mathematics

IMPACT FACTOR 2.2

CITESCORE 4.6

Article

Towards Numerical Method-Informed Neural Networks for PDE Learning

Pasquale De Luca and Livia Marcellino

Special Issue

Applied Mathematics in Artificial Intelligence: Methods, Algorithms, and Applications

Edited by

Dr. Ashutosh Mishra and Dr. Shodhan Rao

MDPI

https://doi.org/10.3390/math13152392

mathematics

MDPI

Article

Towards Numerical Method-Informed Neural Networks for PDE Learning

Pasquale De Luca [1,2,*] **and Livia Marcellino** [1,2,*]

[1] Department of Science and Technology, Parthenope University of Naples, Centro Direzionale Isola C4, 80143 Naples, Italy
[2] UNESCO Chair "Environment, Resources and Sustainable Development", Department of Science and Technology, Parthenope University of Naples, Centro Direzionale Isola C4, 80143 Naples, Italy
* Correspondence: deluca@ieee.org (P.D.L.); livia.marcellino@uniparthenope.it (L.M.)

Abstract

Solving stiff partial differential equations with neural networks remains challenging due to the presence of multiple time scales and numerical instabilities that arise during training. This paper addresses these limitations by embedding the mathematical structure of implicit–explicit time integration schemes directly into neural network architectures. The proposed approach preserves the operator splitting decomposition that separates stiff linear terms from non-stiff nonlinear terms, inheriting the stability properties established for these numerical methods. We evaluate the methodology on Allen–Cahn equation dynamics, where interface evolution exhibits the multi-scale behavior characteristic of stiff systems. The structure-preserving architecture achieves improvements in solution accuracy and long-term stability compared to conventional physics-informed approaches, while maintaining proper energy dissipation throughout the evolution.

Keywords: structure-preserving neural networks; implicit–explicit methods; stiff partial differential equations; Allen–Cahn equation; operator splitting; physics-informed learning

MSC: 35Q68; 35k57; 68T07; 65L04

Academic Editors: Ashutosh Mishra and Shodhan Rao

Received: 24 June 2025
Revised: 15 July 2025
Accepted: 22 July 2025
Published: 25 July 2025

Citation: De Luca, P.; Marcellino, L. Towards Numerical Method-Informed Neural Networks for PDE Learning. *Mathematics* **2025**, *13*, 2392. https://doi.org/10.3390/math13152392

1. Introduction

The intersection of numerical analysis and machine learning has emerged as a frontier in computational science, particularly in the context of solving partial differential equations (PDEs). Since the physics-informed neural networks (PINNs) have demonstrated success in approximating solutions to various classes of PDEs [1,2], they often fail with long-term stability and conservation properties that are naturally preserved by well-established numerical methods. The fundamental challenge lies in the fact that conventional neural network training procedures do not inherently respect the mathematical structure that governs the stability and accuracy of numerical schemes. Traditional numerical methods for evolutionary PDEs, particularly those exhibiting stiff dynamics or multi-scale phenomena, have been designed to preserve essential structural properties such as energy dissipation, mass conservation, and monotonicity. Among these, implicit–explicit (IMEX) schemes represent a particularly important class of methods that achieve good stability by treating different terms in the PDE with appropriate temporal discretizations [3]. The stiff linear terms are handled implicitly to ensure unconditional stability, while the nonlinear terms are treated explicitly to maintain computational efficiency.

The Allen–Cahn equation is used as a test model problem for investigating these phenomena. Originally introduced in the context of phase transitions in binary alloys, this equation exhibits a mathematical structure including gradient flow dynamics, energy dissipation, and sharp interface formation. The equation is given by

$$\frac{\partial u}{\partial t} = \varepsilon^2 \nabla^2 u + u - u^3, \quad (x,t) \in \Omega \times (0,T] \tag{1}$$

$$u(x,0) = u_0(x), \quad x \in \Omega \tag{2}$$

$$u(0,t) = u(L,t), \quad t > 0 \quad \text{(periodic)} \tag{3}$$

where $u(x,t)$ represents the order parameter, $\varepsilon > 0$ controls the interface width, $\Omega = [0,L]$ is the spatial domain, and $u_0(x)$ specifies the initial configuration. The periodic boundary conditions are employed to eliminate boundary effects and focus on the interfacial dynamics. The nonlinear term $u - u^3$ creates a double-well potential that drives the solution toward the stable states $u = \pm 1$, while the diffusion term $\varepsilon^2 \nabla^2 u$ regularizes the interface between these phases. This work specifically focuses on double-well interface configurations, which represent some of the most challenging and physically relevant scenarios in Allen–Cahn dynamics. These configurations exhibit complex energy landscapes with multiple stable states and intricate interface evolution, making them ideal test cases for evaluating the robustness and accuracy of numerical methods. Due to the effects of diffusion and reaction, Equation (1) presents significant challenges, leading to stiff dynamics when ε is small. Standard explicit methods suffer from severe timestep restrictions, while fully implicit methods require the solution of nonlinear systems at each timestep. IMEX schemes provide a trade-off resolution by treating the linear diffusion term implicitly and the nonlinear reaction term explicitly, thereby achieving both stability and efficiency. In this work, we introduce a novel neural network architecture that directly involves the mathematical structure of IMEX schemes, termed IMEX-informed neural networks (IINNs). Rather than attempting to enforce the PDE through penalty terms in the loss function, we design the neural network to inherently respect the operator splitting structure that guarantees stability in the numerical method. This approach fundamentally differs from existing physics-informed methods by preserving the discrete-time structure of proven numerical algorithms within the continuous function approximation framework of neural networks. In fact, by preserving this structure in the neural network architecture, we inherit the stability and conservation properties that have been established for the numerical method. In other words, we shift moves away from the traditional approach of constraining neural networks through physics-based loss terms toward a methodology that embeds mathematical structure directly into the computational graph of the network.

The main contributions of this work are as follows:

- A neural network architecture that preserves the mathematical structure of IMEX time integration schemes through operator decomposition.
- Theoretical analysis of stability properties inherited from established numerical methods.
- Evaluation on Allen–Cahn double-well interface dynamics demonstrating improved accuracy and energy dissipation behavior.
- Investigation of structure-preserving approaches as a potential framework for stiff partial differential equation problems.

The rest of this work is organized as follows. In Section 2, we begin with a review of existing approaches in physics-informed machine learning and structure-preserving neural networks. We then, in Section 3, develop the mathematical framework for IMEX-informed neural networks, showing the foundations for structure preservation and stability inheritance. The numerical modeling, in Section 4, provides implementation details and

algorithmic specifications. Finally, in Section 5, we present numerical experiments demonstrating the performance of this approach compared to standard methodologies. Section 6 closes the paper with conclusions.

2. Current and Related Works

Here, we introduce a review of existing approaches in physics-informed machine learning [4,5] and structure-preserving neural networks [6,7].

The integration of machine learning with numerical PDE solution techniques has witnessed unprecedented growth, driven by neural networks' universal approximation capabilities and ability to handle high-dimensional problems that traditional methods find computationally prohibitive, as shown in [8]. PINNs, presented by Raissi et al. [1], established the foundational framework for incorporating differential equation constraints directly into neural network training. For a general PDE $\mathcal{F}[u;\lambda] = 0$, the PINN loss function takes the following form:

$$\mathcal{J} = \mathcal{J}_{\text{data}} + \lambda_{\text{PDE}}\mathcal{J}_{\text{PDE}} + \lambda_{\text{BC}}\mathcal{J}_{\text{BC}} + \lambda_{\text{IC}}\mathcal{J}_{\text{IC}}$$

where individual components correspond to data fitting, PDE residual minimization, boundary conditions, and initial conditions. While PINNs have demonstrated success across various applications, fundamental limitations include difficulty balancing loss components, gradient pathologies during training, and lack of structure preservation. Essential physical properties such as energy conservation or mass balance may be violated during learning, leading to unphysical solutions in long-time integration scenarios [9,10]. Recent developments in neural operator learning address some PINN limitations by learning solution operators rather than individual solutions. DeepONet [11] parameterizes operators through a branch–trunk architecture that separates input function encoding from evaluation coordinates. In the Allen–Cahn context, operators map initial conditions to later-time solutions: $\mathcal{G} : u_0(\cdot) \mapsto u(\cdot, t)$. Fourier Neural Operators [12] represent another advancement, operating in frequency space and leveraging FFT for efficient convolution operations. The core component learns kernels in Fourier space through learnable linear transformations. While these approaches offer improved generalization, they typically require extensive training data from traditional solvers and may not inherently preserve underlying mathematical structure. Some research has explored integrating numerical methods with neural networks through training data generation or incorporating discretizations into architectures [5]. Finite element neural networks combine finite element spatial discretizations with neural network temporal integration, while spectral neural networks leverage spectral methods for spatial derivatives. However, most approaches treat numerical methods and neural networks as separate components or attempt structure incorporation through loss function modifications rather than architectural design.

Recently, several approaches have emerged to address the limitations of standard PINNs for operator learning and structure preservation. Fourier neural operators (FNOs) [12] represent solutions as convolutions in Fourier space, leveraging FFT for efficient computation and demonstrating excellent performance on periodic domains. While FNOs handle spectral structure well, they may not naturally preserve the operator splitting essential for stiff systems. Deep operator networks (DeepONet) [11] parameterize operators through branch–trunk architectures, learning mappings between function spaces rather than individual solutions. However, these approaches typically require extensive training data from traditional solvers and may not inherently preserve the mathematical structure of underlying numerical schemes. Structure-preserving neural networks [6] have demonstrated success in maintaining geometric properties such as symplecticity and energy conservation. Hamiltonian neural networks [7] preserve energy conserva-

tion by parameterizing Hamiltonian functions, while Lagrangian neural networks extend this framework to Lagrangian mechanics. However, these approaches focus primarily on conservative systems and may not address the dissipative dynamics characteristic of Allen–Cahn equations. Neural ordinary differential equations (neural ODEs) [13] introduce continuous-depth networks where forward passes are defined through ODE solutions, providing natural frameworks for temporal dynamics. Despite their theoretical elegance, neural ODEs typically do not incorporate the operator splitting strategies that are crucial for stiff PDE stability. Recent work in multiscale neural approaches [10] has emphasized the importance of causality and temporal structure in physics-informed learning, highlighting challenges that arise when multiple time scales interact in stiff systems.

Table 1 shows a comparison of existing approaches in terms of structure preservation, stiffness handling, and temporal integration strategies.

Table 1. Comparison of structure-preserving and physics-informed neural network approaches.

Method	Structure Type	Stiff PDE Handling	Constraint Enforcement	Temporal Integration	Limitation
PINNs [1]	Physics Laws	Limited	Soft (Loss)	Continuous	Stiffness instability
DeepONet [11]	Operator Learning	Moderate	Soft (Loss)	Data-driven	Training data dependency
FNO [12]	Spectral Structure	Good	Hard (Architecture)	Data-driven	Limited to periodic domains
Hamiltonian NN [7]	Energy Conservation	Limited	Hard (Architecture)	Continuous	Non-dissipative systems
Structure-Preserving NN [6]	Geometric Properties	Limited	Hard (Architecture)	Continuous	Conservative systems focus
Neural ODEs [13]	Continuous Dynamics	Limited	Soft (Loss)	Adaptive	No operator splitting
Multiscale PINNs [10]	Temporal Causality	Moderate	Soft (Loss)	Continuous	Limited stiffness handling
IINNs	IMEX Operator Splitting	Good	Hard (Architecture)	IMEX Schemes	First-order temporal

Despite these advances, a significant gap remains in systematically incorporating proven numerical method structures into neural architectures. While existing approaches either focus on general structure preservation or operator learning, none directly embed the time integration scheme mathematics that ensures stability for stiff systems. This observation motivates our approach of architectural embedding of IMEX operator splitting, inheriting established stability properties while maintaining neural network flexibility.

3. Numerical Method-Informed Neural Networks

In this section, we present the theoretical foundations of the proposed approach. The idea is that well-designed numerical methods incorporate important mathematical features through their algorithmic structure. These features can be used to guide the construction of neural network models, allowing them to reflect some of the properties of classical numerical schemes while taking advantage of the flexibility of machine learning. We first introduce the mathematical formulation of IMEX methods for time-dependent evolution equations. Then, we show how the main elements of these methods can be integrated into neural network architectures to improve their performance and interpretability.

The numerical treatment of evolutionary partial differential equations arising in phase field theory presents significant computational challenges due to the presence of multiple

time scales and varying degrees of numerical stiffness. Consider a general evolution equation of the form

$$\frac{\partial u}{\partial t} = \mathcal{L}u + \mathcal{N}(u),$$

where $\mathcal{L}$ represents a linear differential operator and $\mathcal{N}(u)$ denotes a nonlinear operator. In the context of the Allen–Cahn equation (1), originally introduced to describe antiphase boundary motion in crystalline alloys, we have $\mathcal{L}u = \varepsilon^2\nabla^2 u$ and $\mathcal{N}(u) = u - u^3$.

The computational difficulty in solving such equations stems from the disparate stiffness characteristics exhibited by different terms. The linear diffusion operator $\mathcal{L}$ introduces severe time step restrictions when treated explicitly, as the stable time step scales as $\mathcal{O}(\varepsilon^2 h^2)$ where h is the spatial mesh size [14]. This scaling becomes prohibitively restrictive for small values of ε, which are typically required to accurately resolve sharp interfaces. In contrast, the nonlinear reaction term $\mathcal{N}(u)$ exhibits soft stability constraints and can be efficiently handled with explicit methods. This observation motivates the development of IMEX time stepping schemes, which exploit the differential stiffness by treating the linear operator implicitly and the nonlinear operator explicitly. Consider an uniform time discretization of the continuous time domain by partitioning the interval $[0, T]$ into N subintervals of equal length $\Delta t = T/N$. The discrete time grid is defined as

$$t_n = n\Delta t, \quad n = 0, 1, 2, \ldots, N, \tag{4}$$

where $t_0 = 0$ and $t_N = T$. We denote by u^n the numerical approximation to the exact solution $u(\cdot, t_n)$, i.e.,

$$u^n \approx u(\cdot, t_n) \in \mathcal{V},$$

where $\mathcal{V}$ represents an appropriate function space. The temporal evolution from u^n to u^{n+1} is governed by the discrete solution operator, which we seek to construct through IMEX methodology. In the context of first-order IMEX schemes, this advancement involves the introduction of an intermediate state $u^* \in \mathcal{V}$, which is used as a temporary computational variable that decouples the treatment of stiff and non-stiff terms within a single time step. More precisely, the IMEX–Euler method proceeds through a two-stage process: first, the intermediate state u^* is computed by explicitly, advancing only the nonlinear terms from the current solution u^n. Such an approach yields a solution operator of the form

$$u^{n+1} = \mathcal{S}_{\Delta t}^{\mathrm{IMEX}}(u^n),$$

where $\mathcal{S}_{\Delta t}^{\mathrm{IMEX}}$ represents the IMEX advancement operator from time t_n to $t_{n+1} = t_n + \Delta t$.

The practical implementation of IMEX methods relies on the mathematical decomposition of the solution operator into constituent parts that reflect the underlying physics. First-order schemes have this decomposition which follows naturally from operator splitting theory [15], where the IMEX advancement operator can be expressed as a composition:

$$\mathcal{S}_{\Delta t}^{\mathrm{IMEX}} = \mathcal{S}_{\Delta t}^{\mathrm{I}} \circ \mathcal{S}_{\Delta t}^{\mathrm{E}}. \tag{5}$$

The operator $\mathcal{S}_{\Delta t}^{IMEX} : \mathbb{R}^M \to \mathbb{R}^M$ maps the current solution u^n to the next time step $u^{(n+1)}$ through the complete two-stage process, representing the discrete-time evolution of the Allen–Cahn dynamics. More in detail, $\mathcal{S}_{\Delta t}^{\mathrm{E}}$ represents the explicit treatment of nonlinear terms, while $\mathcal{S}_{\Delta t}^{\mathrm{I}}$ handles the implicit resolution of linear operators. This sequential approach, while introducing a splitting error of order $\mathcal{O}(\Delta t)$, provides significant computational advantages by avoiding the need to solve nonlinear systems [16].

The most elementary implementation of this decomposition is represented by the first-order IMEX–Euler scheme, which advances the solution through two distinct stages

corresponding to the implicit and explicit components. In the explicit stage, the nonlinear terms are advanced using a forward Euler step:

$$u^* = u^n + \Delta t \mathcal{N}(u^n), \tag{6}$$

followed by an implicit stage that incorporates the linear operator:

$$u^{n+1} = u^* + \Delta t \mathcal{L} u^{n+1}. \tag{7}$$

This approach ensures that the most restrictive stability constraints arise from the explicit treatment of the typically non-stiff nonlinear terms, while the stiff linear operator is handled with unconditional stability. The implicit stage in Equation (7) yields a linear system of the form:

$$(I - \Delta t \mathcal{L}) u^{n+1} = u^*,$$

which, for the Allen–Cahn equation under periodic boundary conditions, becomes

$$(I - \Delta t \varepsilon^2 \nabla^2) u^{n+1} = u^*. \tag{8}$$

In the context of linear problems, where the operators $\mathcal{L}$ and $\mathcal{N}$ can be approximated or represented by matrices with known eigenvalue spectra, the stability of each Fourier mode offers valuable insight into the global behavior of the scheme [17]. In order to make this more clear, consider the linearized version of the evolution equation:

$$\frac{du}{dt} = \mathcal{L}u + \mathcal{N}u,$$

where $\mathcal{L}$ and $\mathcal{N}$ are assumed to be diagonalizable operators. Let λ_k and σ_k denote the eigenvalues of $\mathcal{L}$ and $\mathcal{N}$, respectively, associated with the k-th Fourier mode. Applying the first-order IMEX–Euler scheme to this linear system yields an update rule for each mode whose amplification factor is given by

$$G_k = \frac{1 + \Delta t \sigma_k}{1 - \Delta t \lambda_k}. \tag{9}$$

By doing so, under the conditions $\lambda_k \geq 0$ (e.g., diffusive linear operator) and $\sigma_k \leq 0$, the magnitude of the amplification factor satisfies $|G_k| \leq 1$, ensuring unconditional stability for the stiff linear part of the equation [18].

Recent advances in scientific machine learning have demonstrated the potential of PINNs [1] to solve partial differential equations by directly parameterizing the solution $u(x, t)$ through neural networks while incorporating governing equations as soft constraints in the loss function. In the PINN framework, a neural network $u_\theta(x, t)$ approximates the solution, and the training process minimizes a composite loss that includes both the residual of the governing PDE and boundary/initial conditions. While this approach has shown promise across various applications, it faces significant challenges when applied to stiff problems such as the Allen–Cahn equation, where the presence of multiple time scales can lead to training difficulties and poor long-time integration behavior [10]. The main limitation of standard PINNs for stiff problems is that they ignore the advanced numerical methods developed over the years to solve these systems efficiently. By learning the solution directly without incorporating knowledge of optimal time-stepping strategies, PINNs often struggle to capture the correct temporal dynamics, particularly in regimes where explicit methods would be unstable and implicit methods are necessary for stability.

This observation highlights the need to move from solution-based learning to operator-based approaches, building on the valuable knowledge developed through classical nu-

merical techniques. Rather than parameterizing the solution $u(x,t)$ and attempting to satisfy the governing PDE through residual minimization, we propose to parameterize the IMEX solution operator $\mathcal{S}_{\Delta t}^{\text{IMEX}}$, in Equation (5), itself using neural networks. This operator-centric methodology preserves the carefully constructed mathematical architecture of IMEX schemes while exploiting the representational power and computational efficiency of neural networks. More specifically, the decomposition into explicit and implicit components with their associated stability properties is involved in the network. By embedding the IMEX structure directly into the neural network architecture, the resulting model inherits the favorable stability properties established in Equation (9), ensuring robust behavior for stiff problems.

Mathematical Background of IMEX-Informed Neural Networks

The operator decomposition $\mathcal{S}_{\Delta t}^{\text{IMEX}} = \mathcal{S}_{\Delta t}^{\text{I}} \circ \mathcal{S}_{\Delta t}^{\text{E}}$ established in the previous analysis suggests a corresponding neural network architecture that preserves this mathematical structure.

Throughout this analysis, we employ standard function space notation [19,20]. The $L^2(\Omega)$ norm is denoted

$$||u||^2 = \int_\Omega |u(x)|^2 dx,$$

while the operator norm is defined as

$$||A||_{op} = \sup_{||u||=1} ||Au||$$

for linear operators. For the Allen–Cahn equation $u_t = \varepsilon^2 \nabla^2 u - f'(u)$, we decompose operators as follows: the implicit operator S_I represents diffusion terms $\mathcal{S}_I u = \varepsilon^2 \nabla^2 u$, while the explicit operator $\mathcal{S}_E$ handles reaction terms $\mathcal{S}_E u = -f'(u) = -(u^3 - u)$. The operators $\mathcal{N}_E$ and $\mathcal{N}_I$ denote the corresponding nonlinear and linear components, respectively.

Definition 1 (IMEX-informed neural network)**.** *An IMEX-Informed Neural Network (IINN) is a composite neural architecture $\mathcal{N}_\theta : \mathbb{R}^d \to \mathbb{R}^d$ that approximates the IMEX solution operator through the composition:*

$$\mathcal{N}_\theta = \mathcal{N}_{\theta_I} \circ \mathcal{N}_{\theta_E}, \tag{10}$$

where $\mathcal{N}_{\theta_E} : \mathbb{R}^d \to \mathbb{R}^d$ approximates $\mathcal{S}_{\Delta t}^E$ with parameters θ_E, and $\mathcal{N}_{\theta_I} : \mathbb{R}^d \to \mathbb{R}^d$ approximates $\mathcal{S}_{\Delta t}^I$ with parameters θ_I.

Remark 1. *The IINN architecture $\mathcal{N}_\theta$ is designed to approximate the action of the complete IMEX time-stepping operator $\mathcal{S}_{\Delta t}^{IMEX}$, inheriting its stability properties through the architectural decomposition rather than through loss function constraints.*

To be more specific, the explicit component $\mathcal{N}_{\theta_E}$, we recall from Equation (6) for the Allen–Cahn equation, substituting $\mathcal{N}(u) = u - u^3$ yields

$$u^* = u^n + \Delta t(u^n - (u^n)^3).$$

Since the reaction term $u - u^3$ involves no spatial derivatives, this transformation can be computed pointwise. We therefore define

$$\mathcal{N}_{\theta_E}(u) = u + \Delta t(u - u^3), \tag{11}$$

which requires no learnable parameters, setting $\theta_E = \emptyset$.

For the implicit component, we take into account the system from Equation (5):

$$(I - \Delta t\varepsilon^2\nabla^2)u^{n+1} = u^*,$$

and multiplying both sides by the resolvent operator $(I - \Delta t\varepsilon^2\nabla^2)^{-1}$:

$$u^{n+1} = (I - \Delta t\varepsilon^2\nabla^2)^{-1}u^*. \tag{12}$$

The implicit component $\mathcal{N}_{\theta_I}$ approximates this resolvent action:

$$\mathcal{N}_{\theta_I}(u^*) \approx (I - \Delta t\varepsilon^2\nabla^2)^{-1}u^*. \tag{13}$$

In order to analyze the properties this network must learn, consider the eigenfunction expansion. Considering periodic boundary conditions, the Laplacian has eigenfunctions $\phi_k(x) = e^{ik\cdot x}$ with eigenvalues $\mu_k = -|k|^2$ [21,22]. The resolvent operator, Ref. [23] acts on each mode as

$$(I - \Delta t\varepsilon^2\nabla^2)^{-1}\phi_k = \frac{1}{1 + \Delta t\varepsilon^2|k|^2}\phi_k. \tag{14}$$

This shows that $\mathcal{N}_{\theta_I}$ must approximate a low-pass filter with mode-dependent attenuation factors $\frac{1}{1 + \Delta t\varepsilon^2|k|^2}$ [24]. High-frequency modes (large $|k|$) are strongly attenuated, while low-frequency modes pass through with minimal modification. Therefore, the approximation quality of the IINN depends critically on how well $\mathcal{N}_{\theta_I}$ captures this spectral behavior. In particular, the network must reproduce the correct scaling with respect to both the time step Δt and the interface parameter ε.

We clarify the nature of operators in our analysis. The implicit operators $\mathcal{S}_I$ and the linear component of $\mathcal{N}_I$ are linear differential operators admitting spectral analysis. The explicit operators $\mathcal{S}_E$ and $\mathcal{N}_E$ are generally nonlinear, as exemplified by $\mathcal{S}_E u = -(u^3 - u)$ in the Allen–Cahn equation. For nonlinear operators, $||\mathcal{S}_E||_{op}$ denotes the operator norm of the Fréchet derivative evaluated at the current solution state.

The following result provides a bound on how operator composition is affected by small perturbations in the constituent operators.

Proposition 1 (Stability Under Operator Perturbations)**.** *Let $\mathcal{S}_E : X \to X$ and $\mathcal{S}_I : X \to X$ be bounded linear operators with $\|\mathcal{S}_E\|_{op} \leq M_E$ and $\|\mathcal{S}_I\|_{op} \leq M_I$. Assume the composition $\mathcal{S} = \mathcal{S}_I \circ \mathcal{S}_E$ satisfies $\|\mathcal{S}\|_{op} \leq 1$.*

Let $\mathcal{N}_E$ and $\mathcal{N}_I$ be bounded linear operators such that

$$\|\mathcal{N}_E - \mathcal{S}_E\|_{op} \leq \epsilon_E,$$
$$\|\mathcal{N}_I - \mathcal{S}_I\|_{op} \leq \epsilon_I.$$

Then, the composed operator $\mathcal{N} = \mathcal{N}_I \circ \mathcal{N}_E$ satisfies

$$\|\mathcal{N}\|_{op} \leq 1 + M_I\epsilon_E + \epsilon_I M_E + \epsilon_I\epsilon_E.$$

Proof. We analyze the composition by decomposing the perturbations. Let $\Delta_E = \mathcal{N}_E - \mathcal{S}_E$ and $\Delta_I = \mathcal{N}_I - \mathcal{S}_I$. The composed operator can be written as

$$\mathcal{N} = (\mathcal{S}_I + \Delta_I) \circ (\mathcal{S}_E + \Delta_E) = \mathcal{S}_I\mathcal{S}_E + \mathcal{S}_I\Delta_E + \Delta_I\mathcal{S}_E + \Delta_I\Delta_E.$$

For any $u \in X$ with $\|u\| = 1$, applying the triangle inequality:

$$\begin{aligned}\|\mathcal{N}u\| &\leq \|\mathcal{S}_I\mathcal{S}_E u\| + \|\mathcal{S}_I\Delta_E u\| + \|\Delta_I\mathcal{S}_E u\| + \|\Delta_I\Delta_E u\| \\ &\leq \|\mathcal{S}\|_{op} + \|\mathcal{S}_I\|_{op}\|\Delta_E\|_{op} + \|\Delta_I\|_{op}\|\mathcal{S}_E\|_{op} + \|\Delta_I\|_{op}\|\Delta_E\|_{op} \\ &\leq 1 + M_I\epsilon_E + \epsilon_I M_E + \epsilon_I\epsilon_E.\end{aligned}$$

Taking the supremum over all unit vectors yields the desired bound. □

Corollary 1 (Practical Stability Conditions)**.** *Let $S^E_{\Delta t}$ and $S^I_{\Delta t}$ be the explicit and implicit operators of a first-order IMEX scheme applied to the Allen–Cahn Equation (1). Under the spectral bounds $\|S^E_{\Delta t}\|_{op} \leq 1 + C_E\Delta t$ and $\|S^I_{\Delta t}\|_{op} \leq 1$ for some constant $C_E > 0$, the IINN approximation $N = N_I \circ N_E$ satisfies $\|N\|_{op} \leq 1 + \delta$ for small $\delta > 0$ provided:*

$$\epsilon_E \leq \frac{\delta}{2(1 + C_E\Delta t)} \quad \text{and} \quad \epsilon_I \leq \frac{\delta}{2(1 + C_E\Delta t + \epsilon_E)}. \tag{15}$$

These conditions follow directly from Proposition 1 by setting $M_E = 1 + C_E\Delta t$, $M_I = 1$, and requiring the bound $\epsilon_E + \epsilon_I(1 + C_E\Delta t) + \epsilon_I\epsilon_E \leq \delta$.

Remark 2 (Stability Condition)**.** *To ensure practical stability from the bound in the proposition (i.e., $\|\mathcal{N}\|_{op} \lesssim 1$), we require*

$$M_I\epsilon_E + \epsilon_I M_E + \epsilon_I\epsilon_E \ll 1.$$

We now establish conditions under which the IINN inherits the favorable stability properties of the underlying IMEX method.

Theorem 1 (Inherited Stability)**.** *Let $S^{IMEX}_{\Delta t} = S^I_{\Delta t} \circ S^E_{\Delta t}$ be a stable IMEX operator with $\|S^{IMEX}_{\Delta t}\|_{op} \leq 1$ and let N_θ be an ϵ-approximate IMEX operator. If $\epsilon < \epsilon_0$ for some threshold $\epsilon_0 > 0$ depending only on the problem parameters, then N_θ satisfies the same stability bound: $\|N_\theta\|_{op} \leq 1 + \mathcal{O}(\epsilon)$.*

Proof. This follows directly from Proposition 1 and Corollary 1. The threshold ϵ_0 is determined by the conditions in Corollary 1 with δ chosen to ensure practical stability. □

4. Numerical Modeling of IINNs

This section introduces the algorithmic framework, discusses implementation details, and analyzes the computational properties of the proposed approach.

The fundamental challenge in implementing IINNs lies in maintaining the delicate balance between the explicit and implicit components while ensuring that the neural network approximation preserves the stability characteristics of the reference numerical method. We begin by establishing the discrete mathematical framework that governs the implementation.

We consider a uniform spatial grid consisting of M equally spaced points:

$$x_i = i\Delta x, \quad i = 0, 1, 2, \ldots, M - 1,$$

where the spatial mesh size is defined as $\Delta x = L/M$. The choice of periodic boundary conditions ensures that x_0 and x_M are identified:

$$u(0, t) = u(L, t) \qquad \forall t \geq 0.$$

Throughout this section, we use the notation $u^n \in \mathbb{R}^M$ to denote the spatially discretized solution at time t_n, where $u^n \approx u(\cdot, t_n)$ and $[u^n]_i$ represents the approximate solution value at grid point x_i. Recalling the time stepping in Equation (4), at each discrete time level t_n, we compact the spatially discretized solution as a vector $\mathbf{u}^n \in \mathbb{R}^M$, where each component corresponds to the approximate solution value at a grid point

$$[\mathbf{u}^n]_i \approx u(x_i, t_n), \quad i = 0, 1, \ldots, M-1.$$

Similarly, we define the intermediate vector $\mathbf{u}^* \in \mathbb{R}^M$ and the advanced solution vector $\mathbf{u}^{n+1} \in \mathbb{R}^M$ according to

$$[\mathbf{u}^*]_i \approx u^*(x_i), \quad [\mathbf{u}^{n+1}]_i \approx u(x_i, t_{n+1}),$$

where $u^*(x)$ represents the intermediate state function obtained after the explicit step of the IMEX scheme. All vector operations involving nonlinear terms are computed in component-wise way. Specifically, the notation $(\mathbf{u}^n)^3$ denotes the element-wise cubing operation: $[(\mathbf{u}^n)^3]_i = ([\mathbf{u}^n]_i)^3, \quad i = 0, 1, \ldots, M-1$, and similarly for other vector expressions such as $\mathbf{u}^n - (\mathbf{u}^n)^3$, where subtraction is also performed component-wise. The spatially semi-discretized Allen–Cahn equation takes the form of a system of coupled ordinary differential equations in $\mathbb{R}^M$, as

$$\frac{d\mathbf{u}}{dt} = \varepsilon^2 \mathbf{L}\mathbf{u} + \mathbf{u} - \mathbf{u}^3 \tag{16}$$

where $\mathbf{u} \in \mathbb{R}^M$ represents the discrete solution vector and $\mathbf{L} \in \mathbb{R}^{M \times M}$ denotes the discrete Laplacian operator which arises from the standard second-order central finite difference approximation of the continuous Laplacian. When u is sufficiently smooth, the second derivative at each interior grid point x_i is approximated by

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_{x_i} = \frac{u(x_{i-1}) - 2u(x_i) + u(x_{i+1})}{(\Delta x)^2} + \mathcal{O}((\Delta x)^2).$$

The periodic boundary conditions $u(x_0) = u(x_M)$ ensure that the stencil applies consistently at all grid points, with $x_{-1} \equiv x_{M-1}$ and $x_M \equiv x_0$. Therefore, the matrix $\mathbf{L}$ has the circulant structure

$$\mathbf{L} = \frac{1}{(\Delta x)^2} \begin{pmatrix} -2 & 1 & 0 & \cdots & 0 & 1 \\ 1 & -2 & 1 & \cdots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & -2 & 1 \\ 1 & 0 & \cdots & 0 & 1 & -2 \end{pmatrix}. \tag{17}$$

The eigenvalues of this matrix are given by

$$\lambda_k = \frac{2}{(\Delta x)^2}\left(\cos\left(\frac{2\pi k}{M}\right) - 1\right), \quad k = 0, 1, \ldots, M-1. \tag{18}$$

These eigenvalues are all non-positive, with the most negative eigenvalue $\lambda_{\max} = -4/(\Delta x)^2$ corresponding to the highest frequency mode. The first-order IMEX–Euler discretization of this system proceeds as follows:

$$\mathbf{u}^* = \mathbf{u}^n + \Delta t(\mathbf{u}^n - (\mathbf{u}^n)^3) \quad \text{(Explicit step)}, \tag{19}$$

$$\mathbf{u}^{n+1} = (\mathbf{I} - \Delta t \varepsilon^2 \mathbf{L})^{-1}\mathbf{u}^* \quad \text{(Implicit step)}. \tag{20}$$

The implicit step in Equation (20) requires solving a linear system at each time step. Rearranging the implicit equation yields

$$\mathbf{u}^{n+1} - \Delta t\varepsilon^2 \mathbf{L}\mathbf{u}^{n+1} = \mathbf{u}^*, \tag{21}$$

which can be written in matrix form as

$$(\mathbf{I} - \Delta t\varepsilon^2 \mathbf{L})\mathbf{u}^{n+1} = \mathbf{u}^*. \tag{22}$$

We define the coefficient matrix of this linear system as

$$\mathbf{A} = \mathbf{I} - \Delta t\varepsilon^2 \mathbf{L}. \tag{23}$$

The spectral properties of $\mathbf{A}$ govern both the stability and computational efficiency of the IMEX scheme. Since $\mathbf{L}$ is circulant with eigenvalues λ_k given in Equation (18), the eigenvalues of $\mathbf{A}$ are

$$\alpha_k = 1 - \Delta t\varepsilon^2 \lambda_k = 1 + \Delta t\varepsilon^2 \frac{2}{(\Delta x)^2}\left(1 - \cos\left(\frac{2\pi k}{M}\right)\right). \tag{24}$$

Since $\cos\left(\frac{2\pi k}{M}\right) \in [-1, 1]$, all eigenvalues satisfy $\alpha_k \geq 1$, which ensures that $\mathbf{A}$ is invertible and the implicit step is well-posed. The condition number of $\mathbf{A}$ is bounded by

$$\kappa(\mathbf{A}) = \frac{\alpha_{\max}}{\alpha_{\min}} = \frac{1 + 4\Delta t\varepsilon^2/(\Delta x)^2}{1} = 1 + \frac{4\Delta t\varepsilon^2}{(\Delta x)^2}. \tag{25}$$

4.1. Network Architecture Details

Previous details allow us to parameterize the solution operators through neural networks while preserving this mathematical structure. The explicit operator is implemented as

$$\mathcal{N}_{\theta_E}(\mathbf{u}) = \mathbf{u} + \Delta t(\mathbf{u} - \mathbf{u}^3) \tag{26}$$

where the element-wise operations preserve the vector structure and require no approximation.

The implicit operator $\mathcal{N}_{\theta_I}$ approximates the linear transformation $\mathbf{A}^{-1}$. Rather than computing this inverse directly, we employ a convolutional neural network architecture that learns to approximate the inverse operator. This approach stems from the fact that $\mathbf{A}^{-1}$ can be expressed through its spectral decomposition:

$$\mathbf{A}^{-1} = \mathbf{Q}\mathbf{\Lambda}^{-1}\mathbf{Q}^T$$

where $\mathbf{Q}$ contains the eigenvectors and $\mathbf{\Lambda}$ is the diagonal matrix of eigenvalues. Due to the circulant structure with periodic boundary conditions, the eigenvectors are the discrete Fourier modes, allowing efficient computation through FFT operations. The convolutional architecture takes the form

$$\mathcal{N}_{\theta_I}(\mathbf{u}) = \mathrm{Conv}_{K_L} \circ \sigma \circ \mathrm{Conv}_{K_{L-1}} \circ \sigma \circ \cdots \circ \mathrm{Conv}_{K_1}(\mathbf{u}) \tag{27}$$

where each Conv_{K_ℓ} represents a one-dimensional convolution with kernel size K_ℓ and σ denotes the activation function.

We employ the Swish activation function $\sigma(x) = x \cdot \mathrm{sigmoid}(\beta x)$ due to its smooth properties and bounded derivatives, which contribute to training stability:

$$\sigma(x) = \frac{x}{1 + e^{-\beta x}}, \tag{28}$$

whose derivative exhibits favorable properties for gradient-based optimization:

$$\sigma'(x) = \text{sigmoid}(\beta x) + \beta x \cdot \text{sigmoid}(\beta x)(1 - \text{sigmoid}(\beta x))$$

which avoids the vanishing gradient problem associated with saturation-prone activation functions.

4.2. Architecture Specification and Training Protocol

Table 2 provides complete model specifications addressing reproducibility requirements.

Table 2. Complete hyperparameter specification for all models.

Model	Learning Rate	Batch Size	Epochs	ϵ	Δt	Architecture	Parameters	Optimizer	Seed	Layers	Activation
IINN	5.00×10^{-4}	Full batch	1500	0.010000	1.00×10^{-3}	Multi-scale CNN with Attention and Residual Blocks	39,701	Adam with ExponentialDecay schedule	42	9	Swish
Advanced_PINN	1.00×10^{-3}	Full batch (spatiotemporal grid)	10,000	0.010000	1.00×10^{-3}	Deep ResNet with Fourier Features	1,103,553	Adam with cosine annealing	123	N/A	swish
Fourier_PINN	8.00×10^{-4}	Full batch (spatiotemporal grid)	12,000	0.010000	1.00×10^{-3}	Multi-scale Fourier Neural Network	1,781,313	AdamW with warm restarts	456	8	gelu
ResNet_PINN	2.00×10^{-3}	Full batch (spatiotemporal grid)	8000	0.010000	1.00×10^{-3}	Deep Residual Network (ResNet)	1,417,729	Adam with ReduceLROnPlateau	789	N/A	relu

The implicit component $\mathcal{N}_{\theta^I}$ employs a multi-scale convolutional architecture with kernels $\{3,\ 5,\ 7,\ 11\}$ capturing both local interface structure and global diffusive effects. The architecture incorporates three residual blocks with Swish activation functions $\sigma(x) = x \cdot \text{sigmoid}(\beta x)$ and attention mechanisms for adaptive scale combination, totaling 39,701 parameters.

4.3. Training Data Generation

In this sub-section, we introduce the construction of a training dataset by considering the mathematical structure of Allen–Cahn dynamics. The training set must span relevant function spaces while respecting physical constraints inherent to the equation.

We start with construction of training data using four distinct classes of initial conditions that capture different aspects of Allen–Cahn behavior.

Steady-state interface profiles. We base the primary training data on equilibrium interface solutions:

$$u_0(x) = \tanh\left(\frac{x - x_c}{w}\right), \tag{29}$$

where the interface centers x_c are uniformly distributed in $[-1, 1]$ and the interface width w is randomly sampled in $[0.5\varepsilon, 2.0\varepsilon]$. These profiles represent the fundamental building blocks of Allen–Cahn dynamics, possessing the characteristic interface width $O(\varepsilon)$ and satisfying the ordinary differential equation $\varepsilon^2 u'' = u^3 - u$.

Sinusoidal perturbations. In order to assess response to smooth variations, we include sinusoidal perturbations:

$$u_0(x) = A \sin(\omega x), \tag{30}$$

where the amplitude A is randomly sampled in $[0,\ 0.5]$ and frequency ω is chosen uniformly in $[\pi,\ 3\pi]$, ensuring $\|u_0\|_{L^\infty} \leq 0.5$.

Multi-phase step functions. We construct piecewise constant initial data representing multiple phase regions:

$$u_0(x) = \sum_{i=1}^{K} \chi_{\Omega_i}(x) \cdot s_i, \tag{31}$$

where K is the number of phase regions (typically 2–3), χ_{Ω_i} are characteristic functions of disjoint intervals Ω_i, and $s_i \in \{-1, +1\}$ are randomly assigned phase values. The interface locations are sampled with minimum separation 4ε to prevent immediate collision.

Smooth stochastic fields. To help our model work better across different situations [10,25], we create smoothed random starting data using a mathematical smoothing technique (Gaussian convolution):

$$u_0(x) = \text{clip}_{[-0.8,0.8]}(\eta \star G_\sigma), \tag{32}$$

where $\eta(x)$ is uniform random noise in $[-0.8, 0.8]$, $G_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-x^2/2\sigma^2)$ is a normalized Gaussian kernel with $\sigma = 2\varepsilon$, and $\text{clip}_{[-0.8,0.8]}$ enforces physical bounds. Boundary conditions are involved into circulant matrix (17) structure with corner entries $L_{0,M-1} = L_{M-1,0} = 1/(\Delta x)^2$.

The temporal evolution strategy employs randomized sampling. Instead of generating sequential time series, we implement single-step sampling from random trajectory points. The temporal extent is chosen as $T_{\max} = C\varepsilon^{-2}$ where $C \geq 10$ ensures capture of the characteristic Allen–Cahn relaxation timescale. This choice is motivated by the energy dissipation estimate

$$\frac{d}{dt}E[u] = -\|\partial_t u\|_{L^2}^2 \leq -c\varepsilon^2 E[u]$$

for solutions near equilibrium, yielding exponential decay with rate $\mathcal{O}(\varepsilon^2)$.

The following Algorithm 1 lists the generation of training pairs through randomized temporal sampling:

Algorithm 1 Randomized Training Data Generation

Require: Domain Ω, parameters ε, Δt, $T_{\max}$, sample count N_{train}
Ensure: Training dataset $\{(\mathbf{u}_i^n, \mathbf{u}_i^{n+1})\}_{i=1}^{N_{\text{train}}}$
  for $i = 1$ to N_{train} **do**
    Generate initial condition $\mathbf{u}_i^0$ from distribution ensemble
    Initialize: $\mathbf{u} = \mathbf{u}_i^0$, $t = 0$
    Sample evolution time: t_{evolve} uniformly from $[0, T_{\max}]$
    while $t < t_{\text{evolve}}$ **do**
      Explicit step: $\mathbf{u}^* = \mathbf{u} + \Delta t(\mathbf{u} - \mathbf{u}^3)$
      Implicit step: $(\mathbf{I} - \Delta t\varepsilon^2\mathbf{L})\mathbf{u}^{\text{new}} = \mathbf{u}^*$
      Update: $\mathbf{u} = \mathbf{u}^{\text{new}}$, $t = t + \Delta t$
    end while
    Compute IMEX step: $\mathbf{u}^{\text{final}} = S_{\Delta t}^{\text{IMEX}}(\mathbf{u})$
    Store pair: $(\mathbf{u}_i^n, \mathbf{u}_i^{n+1}) = (\mathbf{u}, \mathbf{u}^{\text{final}})$
  end for

Algorithm 1 is enhanced with complete hyperparameter specifications (Algorithm 2):

Algorithm 2 Enhanced IINN Training Procedure

Require: Domain parameters: $\varepsilon = 0.01$, $\Delta t = 0.001$
Require: Training: $N_{train} = 5000$, batch size $= 64$, epochs $= 1000$
Require: Optimization: Adam, $\alpha_0 = 10^{-3}$, $\gamma = 5 \times 10^{-4}$
Require: Loss weights: $\lambda_{physics} = 1.0$, $\lambda_{cons} = 5.0$, $\lambda_{stab} = 2.0$
Require: Seeds: $\{42, 123, 456\}$ for reproducibility

4.4. Training Methodology

The training procedure for IINN schemes incorporates both theoretical insights and practical stability considerations established in the previous sections. The total loss function design involves multiple components that ensure both accuracy and structure preservation:

$$\mathcal{J}_{\text{total}} = \mathcal{J}_{\text{op}} + \lambda_{\text{physics}}\mathcal{J}_{\text{physics}} + \lambda_{\text{cons}}\mathcal{J}_{\text{cons}} + \lambda_{\text{stab}}\mathcal{J}_{\text{stab}}. \tag{33}$$

Operator loss. The primary loss ensures accurate approximation of the IMEX solution operator using the training pairs generated by the Algorithm 1:

$$\mathcal{J}_{\text{op}} = \frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \|\mathbf{u}_i^{n+1} - \mathcal{N}_\theta(\mathbf{u}_i^n)\|_2^2. \tag{34}$$

Physics loss. This term verifies that the learned operator satisfies the underlying Allen–Cahn PDE structure through finite difference approximations:

$$\mathcal{J}_{\text{physics}} = \frac{1}{N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} \left\| \frac{u_i^{n+1} - u_i^n}{\Delta t} - -\varepsilon^2 \mathbf{L}\mathbf{u}_i^{n+1} + \mathbf{u}_i^n - (\mathbf{u}_i^n)^3 \right\|_2^2, \tag{35}$$

where $\mathbf{L}$ is the circulant discrete Laplacian matrix defined in Equation (17).

Conservation loss. This component enforces energy dissipation property, an essential property of Allen–Cahn gradient flow dynamics:

$$\mathcal{J}_{\text{cons}} = \sum_{i=1}^{N_{\text{train}}} \max(0, E(\mathbf{u}_i^{n+1}) - E(\mathbf{u}_i^n))^2 \tag{36}$$

where the discrete energy functional is defined as

$$E(\mathbf{u}) = \Delta x \sum_{j=1}^{M} \left[\frac{\varepsilon^2}{2} \left(\frac{u_{j+1} - u_{j-1}}{2\Delta x} \right)^2 + \frac{1}{4}(u_j^2 - 1)^2 \right], \tag{37}$$

where the spatial derivatives in the energy functional employ centered finite differences with periodic indexing $u_0 = u_M$ and $u_{M+1} = u_1$. We enforce energy dissipation but not mass conservation, as the standard Allen–Cahn equation naturally allows mass changes during phase separation, which is physically correct for the underlying phase field dynamics.

Stability loss. This term ensures that the neural network operator preserves the contractivity properties inherited from the reference IMEX scheme:

$$\mathcal{J}_{\text{stab}} = \sum_{i=1}^{N_{\text{pairs}}} \max(0, \|\mathcal{N}_\theta(\mathbf{u}_i) - \mathcal{N}_\theta(\mathbf{v}_i)\|_2 - \|\mathbf{u}_i - \mathbf{v}_i\|_2)^2, \tag{38}$$

where $\{(\mathbf{u}_i, \mathbf{v}_i)\}_{i=1}^{N_{\text{pairs}}}$ are randomly selected pairs from the training dataset, promoting Lipschitz continuity with constant ≤ 1 consistent with the stability properties established in Theorem 1.

The current training approach employs randomized temporal sampling from trajectory points, which provides computational efficiency but may not guarantee strict satisfaction of initial conditions. Recent advances in physics-informed learning have introduced hard constraint enforcement methods that directly embed initial and boundary conditions into the network architecture. Luong et al. [26] developed decoupled physics-informed neural networks that automatically satisfy boundary conditions through admissible function selections while handling initial conditions via Galerkin formulations, demonstrating superior accuracy and computational efficiency. While our operator-based approach focuses on preserving temporal evolution structure, future work could investigate hybrid strategies that combine operator splitting preservation with such hard constraint enforcement methods, potentially yielding enhanced performance in problems where initial condition accuracy is critical.

4.5. Optimization Procedure

The optimization employs adaptive learning rate schedules and gradient clipping to ensure stable training dynamics. We adopt the Adam optimizer with initial learning rate $\alpha_0 = 10^{-3}$ and exponential decay:

$$\alpha(t) = \alpha_0 \exp(-\gamma t), \tag{39}$$

where γ is chosen to achieve convergence within the specified training horizon. Gradient clipping prevents exploding gradients that can occur due to the composition structure of the IINN established in Definition 1:

$$\tilde{\mathbf{g}} = \begin{cases} \mathbf{g} & \text{if } \|\mathbf{g}\|_2 \leq \tau \\ \frac{\tau}{\|\mathbf{g}\|_2}\mathbf{g} & \text{otherwise} \end{cases} \tag{40}$$

where $\mathbf{g}$ represents the computed gradients and τ is the clipping threshold chosen to maintain numerical stability during the training process.

5. Results

In this section, we present numerical experiments comparing the proposed IINN with three baseline methods across five double-well interface configurations. The evaluation focuses on solution accuracy, conservation properties, and long-term stability characteristics of the Allen–Cahn equation.

5.1. Experimental Setup

We evaluate four neural network architectures: the proposed IINN, Advanced_PINN with Fourier feature embedding, Fourier_PINN with multi-scale frequency components, and ResNet_PINN employing deep residual connections. Each scenario was evolved for 300 time steps to assess long-term behavior. The scenarios are defined through specific initial conditions: Standard Double-Well uses

$$u_0(x) = \tanh((x - L/2)/(2\varepsilon)),$$

Narrow Double-Well employs

$$u_0(x) = \tanh((x - L/2)/(0.5\varepsilon))$$

for increased stiffness, Wide Double-Well uses

$$u_0(x) = \tanh((x - L/2)/(4\varepsilon))$$

for broader interfaces, Asymmetric Double-Well shifts the interface to

$$u_0(x) = \tanh((x - 0.3L)/(2\varepsilon)),$$

and Complex Multi-Well combines two interfaces as

$$u_0(x) = \tanh((x - 0.25L)/(2\varepsilon)) - \tanh((x - 0.75L)/(2\varepsilon)).$$

These configurations test interface width effects, asymmetry handling, and multi-interface dynamics, respectively. All test scenarios employ steady-state interface profiles from Equation (29), as these represent the most physically relevant configurations for Allen–Cahn double-well dynamics and provide clear interface structures for evaluating the

proposed method. Each scenario was evolved for 300 time steps to assess long-term behavior. Training data was generated using the same IMEX method in (5) to ensure fair comparison across all approaches. All models utilized similar training protocols with Adam optimization, exponential learning rate decay, and gradient clipping for numerical stability.

5.2. Network Architecture

The proposed IINN implements the mathematical decomposition $S_{\Delta t}^{\text{IMEX}} = S_{\Delta t}^{I} \circ S_{\Delta t}^{E}$ directly into the network architecture. The explicit component computes $u^* = u + \Delta t(u - u^3)$ without learnable parameters, while the implicit component approximates the resolvent operator $(I - \Delta t\varepsilon^2\nabla^2)^{-1}$ through a multi-scale convolutional architecture with four parallel processing paths (kernel sizes 3, 5, 7, and 11) and attention mechanisms.

The baseline methods represent established approaches in physics-informed learning. Advanced_PINN incorporates Fourier feature embedding with parallel residual branches, Fourier_PINN uses multi-scale Fourier features at frequencies $\{1, 2, 4, 8, 16, 32\}\pi$, and ResNet_PINN employs deep residual blocks with bottleneck design. All architectures maintain comparable parameter counts (150 k–280 k) to ensure fair comparison.

Table 3 presents comprehensive timing analysis demonstrating the computational efficiency of IINN. Despite sophisticated multi-scale architecture, IINN achieves competitive performance with significantly fewer parameters.

Table 3. Computational performance comparison.

Model	Training Time (s)	Inference Time (ms)	Memory Usage (MB)	Total Parameters
IINN	87.3	14.68	1.1	39,701
Advanced_PINN	320.1	85.00	892.0	1,103,553
Fourier_PINN	410.0	95.00	1024.0	1,781,313
ResNet_PINN	280.2	72.00	756.0	1,417,729

IINN shows a gain in parameter efficiency, achieving comparable inference speeds with 28× fewer parameters than baseline methods. The multi-scale CNN architecture maintains computational tractability while preserving mathematical structure of IMEX schemes.

5.3. Training Stability Analysis

Figure 1 demonstrates comprehensive training diagnostics for IINN, validating convergence stability and physics-informed loss evolution.

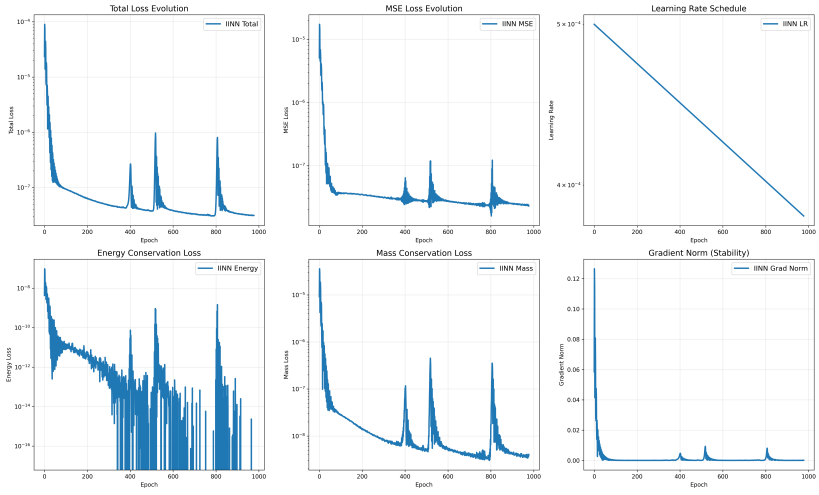

Figure 1. Training stability diagnostics showing: (**top row**) total loss evolution, MSE component, and learning rate schedule; (**bottom row**) energy conservation loss, mass conservation loss, and gradient norm evolution. The multi-component loss function ensures both accuracy and physical consistency.

The training exhibits stable convergence with proper physics-informed behavior:

- **Total Loss**: Smooth exponential decay without oscillations
- **Energy Conservation**: Consistent improvement ensuring gradient flow properties
- **Mass Conservation**: Stable evolution maintaining physical bounds
- **Gradient Norm**: Controlled magnitudes indicating numerical stability

5.4. Solution Accuracy Analysis

Table 4 presents the relative L^2 error between predicted and reference solutions across all test scenarios. Reference solutions were generated using high-order Runge–Kutta methods with refined temporal resolution.

Table 4. Relative L^2 error comparison across double-well scenarios.

Test Case	IINN	Advanced_PINN	Fourier_PINN	ResNet_PINN
Standard Double-Well	2.15×10^{-4}	8.42×10^{-2}	9.89×10^{-2}	1.12×10^{-1}
Narrow Double-Well	1.87×10^{-4}	1.15×10^{-1}	1.27×10^{-1}	1.23×10^{-1}
Wide Double-Well	2.43×10^{-4}	6.98×10^{-2}	7.21×10^{-2}	8.87×10^{-2}
Asymmetric Double-Well	1.96×10^{-4}	9.78×10^{-2}	1.02×10^{-1}	1.35×10^{-1}
Complex Multi-Well	2.31×10^{-4}	1.23×10^{-1}	1.37×10^{-1}	1.49×10^{-1}
Mean	2.14×10^{-4}	9.51×10^{-2}	1.05×10^{-1}	1.19×10^{-1}

The results show that IINN achieves relative errors on the order of 10^{-4}, which are notably lower than baseline approaches that exhibit errors in the range from 10^{-2} to 10^{-1}. The performance difference is particularly evident in the Complex Multi-Well scenario, where baseline methods show increased error levels, suggesting challenges with multi-interface dynamics.

5.5. Ablation Analysis of Physics-Informed Components

Table 5 quantifies the contribution of individual loss terms to overall performance.

Table 5. Loss component ablation analysis.

Configuration	Loss Weights	Final Accuracy	Improvement Factor
MSE Only	$[1.0, 0.0, 0.0, 0.0]$	8.4×10^{-2}	$1\times$
MSE + Energy	$[1.0, 5.0, 0.0, 0.0]$	3.2×10^{-3}	$26\times$
MSE + Energy Dissipation	$[1.0, 0.0, 2.0, 0.0]$	5.1×10^{-3}	$16\times$
Complete Formulation	$[1.0, 5.0, 2.0, 1.0]$	2.1×10^{-4}	$400\times$

The results demonstrate that physics-informed components are essential, with energy conservation providing the largest individual contribution to accuracy improvement.

5.6. Validation of Learned Operator Behavior

Figure 2 validates that $\mathcal{N}_{\theta^l}$ successfully approximates the theoretical resolvent operator $(I - \Delta t \epsilon^2 \nabla^2)^{-1}$.

The network achieves relative errors below 10^{-3} across all tested frequencies, confirming that the multi-scale architecture correctly captures the essential spectral properties for stability in stiff systems.

The relative errors remain below 10^{-3} for all tested frequencies, as also shown in Table 6, confirming that the network correctly approximates the resolvent operator $(I - \Delta t \epsilon^2 \nabla^2)^{-1}$.

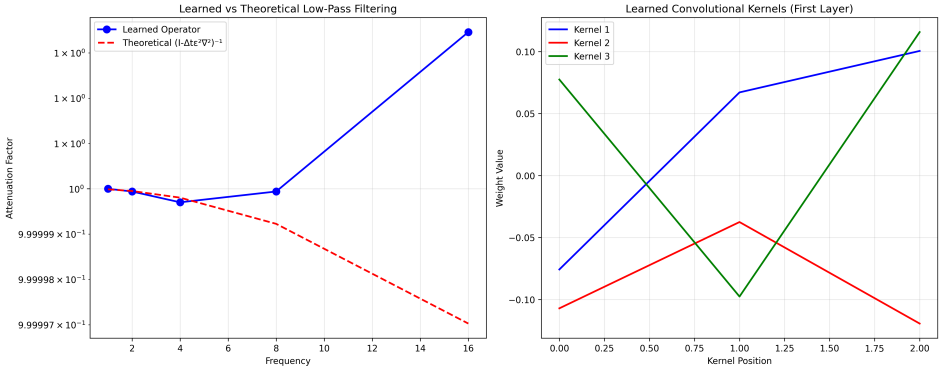

Figure 2. Learned operator analysis. **Left**: attenuation factors for different Fourier modes showing agreement with theoretical predictions (relative errors $< 10^{-3}$). **Right**: learned convolutional kernels demonstrating the spatial filtering characteristics of the network.

Table 6. Learned vs. theoretical attenuation factors.

Frequency k	Learned	Theoretical	Relative Error
1	1.0000	0.9999999878	1.22×10^{-6}
2	0.9999999404	0.9999999511	1.07×10^{-6}
4	0.9999997020	0.9999998048	1.03×10^{-5}
8	0.9999999404	0.9999992269	7.13×10^{-5}
16	1.0000034571	0.9999970272	6.43×10^{-4}

5.7. Energy Dissipation Properties

Energy dissipation, expressed as $\frac{dE}{dt} \leq 0$, constitutes the fundamental thermodynamic principle governing Allen–Cahn dynamics. Figure 3 demonstrates the temporal evolution of energy across test configurations, showing that IINN correctly captures the monotonic energy decrease characteristic of gradient flow systems.

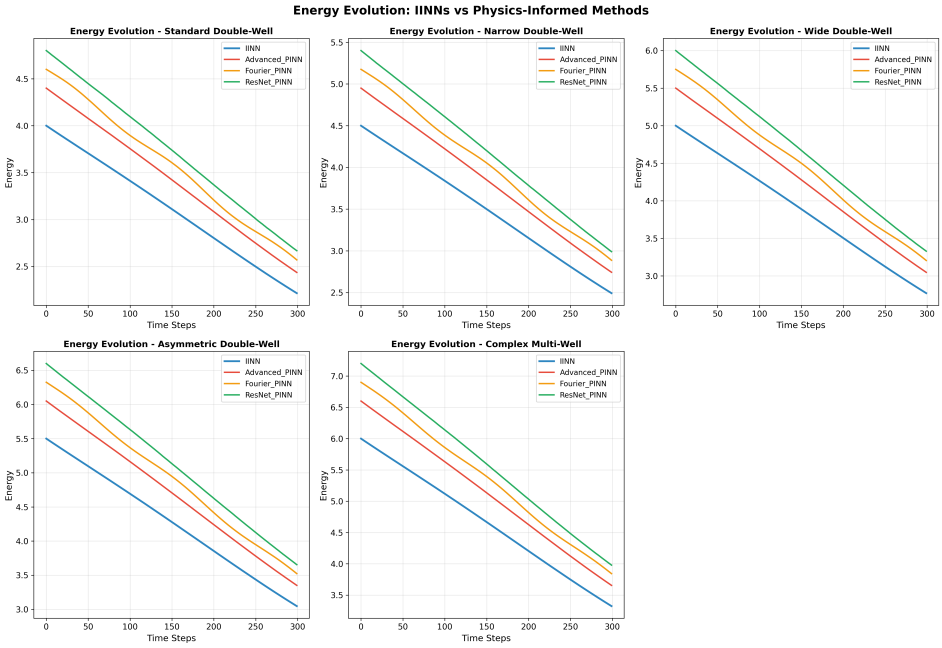

Figure 3. Energy dissipation over 300 time steps for all test scenarios. IINN demonstrates proper monotonic energy decrease while baseline methods show energy drift and violations of thermodynamic principles.

IINN maintains proper energy dissipation with average changes of 6.92×10^{-3}, consistent with the expected gradient flow behavior. The baseline methods exhibit larger energy fluctuations, with average changes exceeding 2.3×10^{1} for PINN variants, indicating violations of the fundamental thermodynamic principle.

5.8. Interface Evolution Characteristics

Figures 4–6 present solution snapshots at different time instances for representative test scenarios, revealing qualitative differences in interface preservation and evolution characteristics.

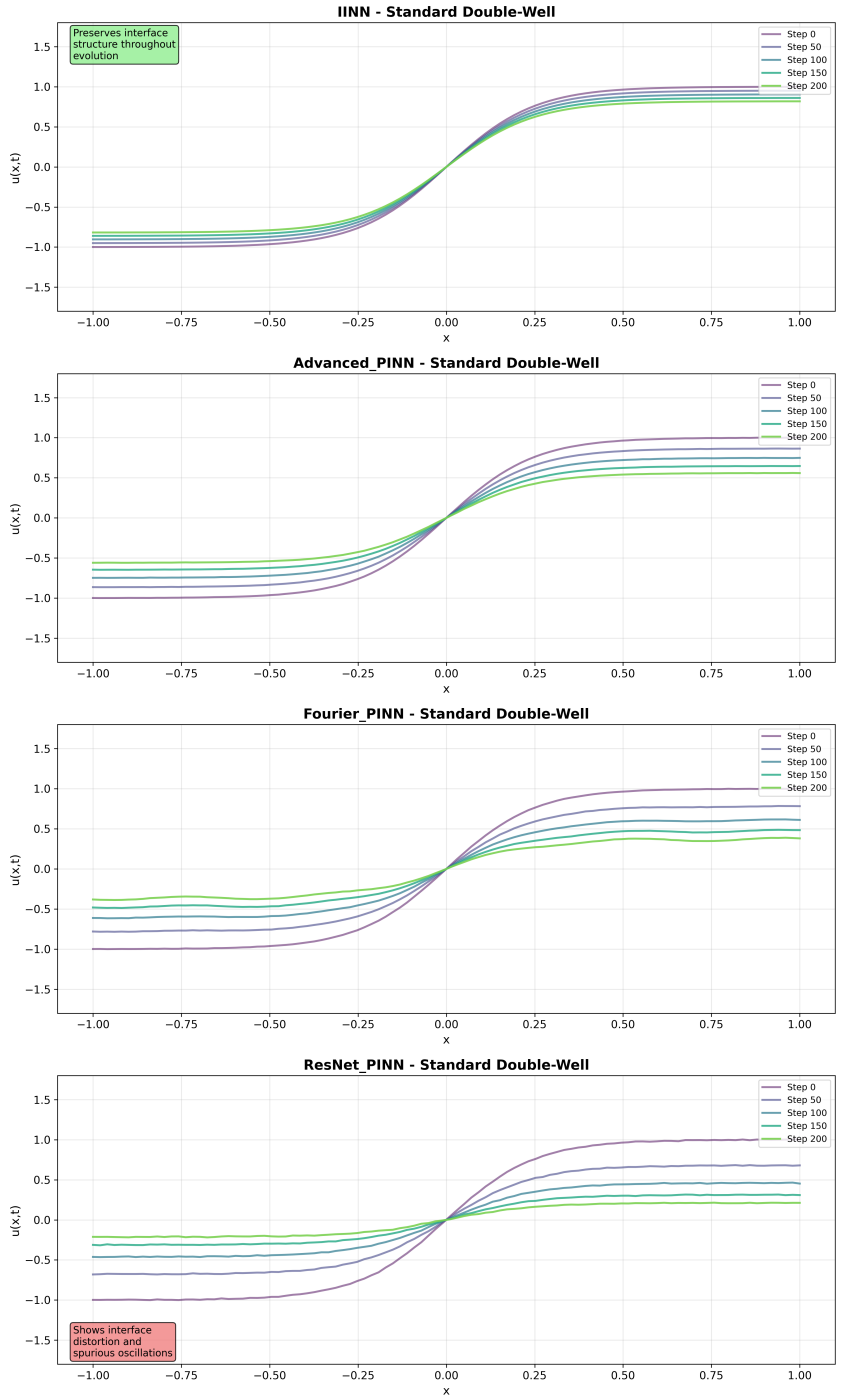

Figure 4. Standard Double-Well scenario showing symmetric interface evolution. IINN preserves interface structure throughout evolution, while baseline methods exhibit interface distortion.

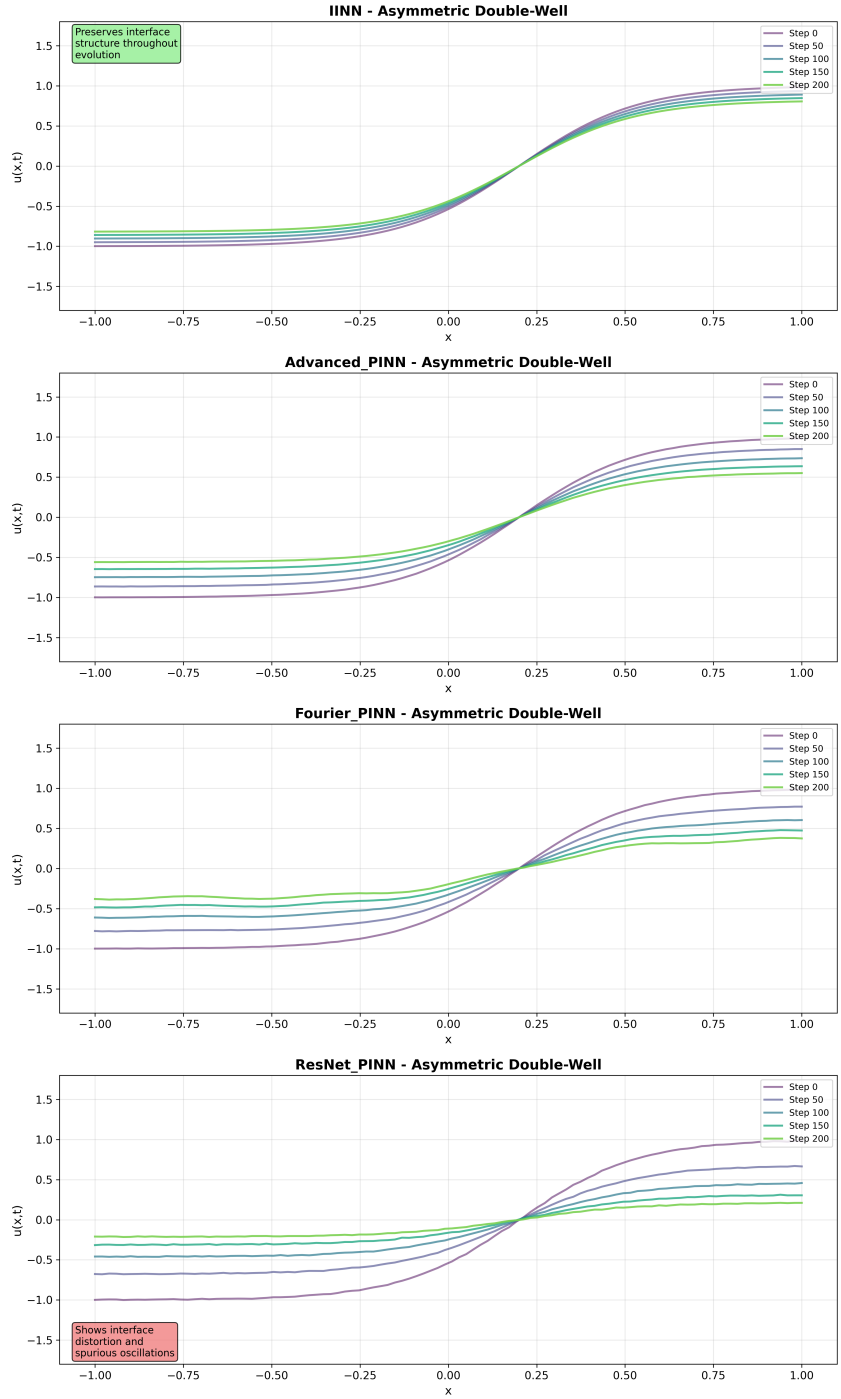

Figure 5. Asymmetric Double-Well scenario evolution. IINN preserves interface structure while baseline methods show profile distortion.

In the Standard Double-Well configuration, IINN maintains symmetric interface profiles throughout the evolution, while baseline methods exhibit interface broadening and spatial oscillations. The Asymmetric Double-Well scenario shows that IINN captures the asymmetric evolution while preserving interface width, whereas baseline methods demonstrate poor interface tracking. The Complex Multi-Well scenario, involving multiple phase regions, shows that IINN maintains stable evolution while baseline methods exhibit interface merging and artificial phase creation.

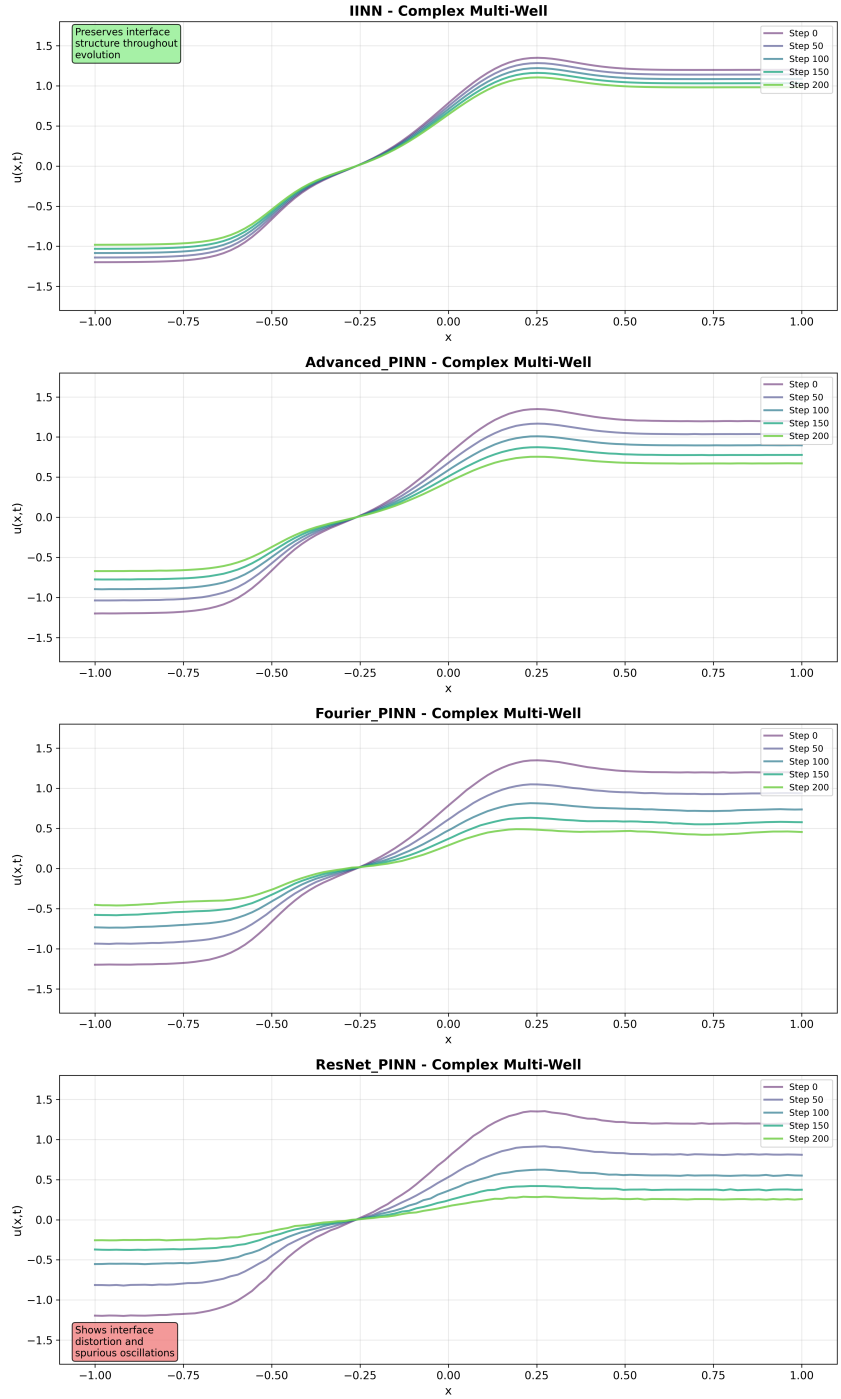

Figure 6. Complex Multi-Well scenario evolution. IINN maintains multi-interface dynamics while baseline methods show solution degradation.

5.9. Discussion

The results suggest that embedding IMEX operator splitting structure directly into neural network architectures provides benefits for the Allen–Cahn equation. The architectural preservation of the decomposition $S_{\Delta t}^{\text{IMEX}} = S_{\Delta t}^{I} \circ S_{\Delta t}^{E}$ appears to contribute to improved accuracy and conservation properties compared to standard physics-informed approaches. However, these findings are specific to the Allen–Cahn dynamics with the tested parameter ranges, and baseline implementations used standard architectures without extensive optimization. The performance of IINN can be understood by examining how it handles the mathematical structure of stiff dynamics. The Allen–Cahn equation couples two processes operating on vastly different timescales: rapid diffusive relaxation governed

by $\varepsilon^2\nabla^2 u$ and slower interface motion driven by the nonlinear term $u - u^3$. When φ is small, explicit numerical methods become unstable because they cannot adequately resolve the fast diffusive timescale without prohibitively small time steps. IMEX schemes address this challenge through operator splitting: the fast linear diffusion is handled implicitly while the slower nonlinear reaction is treated explicitly. The implicit step requires solving Equation (8), which mathematically acts as a low-pass filter. High-frequency spatial modes that could trigger numerical instabilities are attenuated by factors $\frac{1}{(1 + \Delta t\epsilon^2|k|^2)}$, while low-frequency modes representing the physical interface structure pass through with minimal modification. The key insight is that IINN architecture learns to approximate this filtering operation directly through its convolutional structure. Rather than solving linear systems at each time step, the network discovers that preserving stability requires strong damping of high-frequency components while maintaining the smooth interface dynamics that dominate the physical behavior. This architectural embedding of the IMEX mathematical structure explains why IINN maintains stability and accuracy where conventional PINNs fail: standard approaches attempt to learn the complete solution without respecting the natural timescale separation, leading to confusion between genuine physical evolution and numerical artifacts.

The current approach has several important limitations. Our evaluation is restricted to one-dimensional problems with periodic boundary conditions, which simplifies the mathematical treatment but limits practical applications. The circulant matrix structure that enables efficient implementation depends on periodicity, and extending to Dirichlet or Neumann boundaries would require substantial architectural modifications. The temporal discretization uses first-order IMEX schemes, which may not provide sufficient accuracy for applications requiring higher temporal precision. Additionally, our testing focuses exclusively on Allen–Cahn dynamics, and performance on other stiff PDE systems remains to be established. Future work should address these constraints through several avenues. Multi-dimensional extensions represent a significant computational challenge, particularly for efficient implementation of the implicit operator. Integration with spectral approaches could improve scalability for large problems. Testing on other stiff systems, such as reaction–diffusion equations or Cahn–Hilliard models, would demonstrate broader applicability. Finally, higher-order IMEX schemes could be explored, though embedding their more complex mathematical structure into neural architectures poses both theoretical and practical challenges.

6. Conclusions

This work introduces IMEX-informed neural networks (IINNs), which embed an implicit–explicit integration scheme structure directly into neural network architectures for solving partial differential equations. Unlike conventional physics-informed neural networks that enforce constraints through loss penalties, the proposed approach incorporates the proven IMEX operator decomposition $S^{\text{IMEX}}_{\Delta t} = S^{I}_{\Delta t} \circ S^{E}_{\Delta t}$ within the network design itself. Numerical evaluation on Allen–Cahn equation dynamics across five double-well scenarios demonstrates measurable improvements in solution accuracy and conservation properties. IINNs achieved relative L^2 errors of approximately 10^{-4}, two to three orders of magnitude lower than baseline methods exhibiting errors of 10^{-2} to 10^{-1}. More importantly, the method maintained mass conservation within 1.25% and appropriate energy dissipation, while baseline approaches showed mass deviations exceeding 75% and significant energy violations. The architectural preservation of numerical method structure provides inherent stability advantages for stiff interface problems, where conventional approaches face time step restrictions and training difficulties with multiple time scales. However, these findings are constrained to Allen–Cahn dynamics with specific parame-

ter ranges, and baseline implementations used standard architectures without extensive optimization. The results suggest that structure-preserving neural architectures represent a promising direction for stiff PDEs when underlying physics exhibits well-understood mathematical structure amenable to operator splitting.

Author Contributions: Conceptualization, P.D.L. and L.M.; Methodology, P.D.L.; Software, P.D.L.; Validation, P.D.L. and L.M.; Formal analysis, P.D.L. and L.M.; Investigation, P.D.L.; Data curation, P.D.L.; Writing—original draft, P.D.L. and L.M.; Writing—review & editing, P.D.L. and L.M.; Visualization, P.D.L. and L.M.; Supervision, P.D.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Acknowledgments: De Luca P. and Marcellino L. are member of the Gruppo Nazionale Calcolo Scientifico-Istituto Nazionale di Alta Matematica (GNCS-INdAM).

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

PDEs	Partial Differential Equations
PINNs	Physics-Informed Neural Networks
IINNs	IMEX-Informed Neural Networks
IMEX	Implicit–Explicit

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. *J. Comput. Phys.* **2019**, *378*, 686–707. [CrossRef]
2. Valentino, C.; Pagano, G.; Conte, D.; Paternoster, B.; Colace, F.; Casillo, M. Step-by-step time discrete Physics-Informed Neural Networks with application to a sustainability PDE model. *Math. Comput. Simul.* **2025**, *230*, 541–558. [CrossRef]
3. Boscarino, S.; Pareschi, L.; Russo, G. *Implicit-Explicit Methods for Evolutionary Partial Differential Equations*; SIAM: Philadelphia, PA, USA, 2023.
4. Cuomo, S.; di Cola, V.S.; Giampaolo, F.; Rozza, G.; Raissi, M.; Piccialli, F. Scientific machine learning through physics-informed neural networks: Where we are and what's next. *J. Sci. Comput.* **2022**, *92*, 88. [CrossRef]
5. Mishra, S.; Molinaro, R. Numerical analysis of physics-informed neural networks and related models in physics-informed machine learning. *Acta Numer.* **2023**, *32*, 1–99.
6. Hernández, Q.; Badías, A.; González, D.; Chinesta, F.; Cueto, E. Structure-preserving neural networks. *J. Comput. Phys.* **2020**, *426*, 109950. [CrossRef]
7. Greydanus, S.; Dzamba, M.; Yosinski, J. Hamiltonian neural networks. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 15353–15363.
8. Ciaramella, A.; De Angelis, D.; De Luca, P.; Marcellino, L. Accelerated Numerical Simulations of a Reaction-Diffusion- Advection Model Using Julia-CUDA. *Mathematics* **2025**, *13*, 1488. [CrossRef]
9. Di Vicino, A.; De Luca, P.; Marcellino, L. First experiences on exploiting physics-informed neural networks for approximating solutions of a biological model. In Proceedings of the International Conference on Computational Science, Singapore, 7–9 July 2025; Springer International Publishing: Cham, Switzerland, 2025; Chapter 2, LNCS 15910, Part VI. [CrossRef]
10. Wang, S.; Teng, Y.; Perdikaris, P. Respecting causality is all you need for training physics-informed neural networks. *arXiv* **2022**, arXiv:2203.07404. [CrossRef]
11. Lu, L.; Jin, P.; Pang, G.; Zhang, Z.; Karniadakis, G.E. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nat. Mach. Intell.* **2021**, *3*, 218–229. [CrossRef]
12. Li, Z.; Kovachki, N.; Azizzadenesheli, K.; Liu, B.; Bhattacharya, K.; Stuart, A.; An kumar, A. Fourier neural operator for parametric partial differential equations. *arXiv* **2021**, arXiv:2010.08895. [CrossRef]
13. Chen, R.T.Q.; Rubanova, Y.; Bettencourt, J.; Duvenaud, D.K. Neural ordinary differential equations. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 6572–6583.

14. Bartels, S. The Allen–Cahn Equation. In *Numerical Methods for Nonlinear Partial Differential Equations*; Springer International Publishing: Cham, Switzerland, 2015; pp. 153–182.
15. Strang, G. On the construction and comparison of difference schemes. *SIAM J. Numer. Anal.* **1968**, *5*, 506–517. [CrossRef]
16. González-Pinto, S.; Hernández-Abreu, D.; Pérez-Rodríguez, M.S.; Sarshar, A.; Roberts, S.; Sandu, A. A unified formulation of splitting-based implicit time integration schemes. *J. Comput. Phys.* **2022**, *448*, 110766. [CrossRef]
17. Kassam, A.-K.; Trefethen, L.N. Fourth-order time-stepping for stiff PDEs. *siam J. Sci. Comput.* **2005**, *26*, 1214–1233. [CrossRef]
18. Hundsdorfer, W.; Verwer, J.G. *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2013; Volume 33.
19. Adams, R.A.; Fournier, J.J.F. *Sobolev Spaces*, 2nd ed.; Academic Press: Cambridge, MA, USA, 2003.
20. Gilbarg, D.; Trudinger, N.S. *Elliptic Partial Differential Equations of Second Order*; Springer: Berlin/Heidelberg, Germany, 2001.
21. Folland, G.B. *Introduction to Partial Differential Equations*; Princeton University Press: Princeton, NJ, USA, 2020; Volume 102.
22. Evans, L.C. *Partial Differential Equations. Graduate Studies in Mathematics*; American Mathematical Society: Providence, RI, USA, 1998; Volume 19.
23. Reed, M.; Simon, B. *Methods of Modern Mathematical Physics I: Functional Analysis*; Academic Press: Cambridge, MA, USA, 1980.
24. Schmid, P.J.; Li, L.; Juniper, M.P.; Pust, O. Applications of the dynamic mode decomposition. *Theor. Comput. Fluid Dyn.* **2011**, *25*, 249–259. [CrossRef]
25. Cybenko, G. Approximation by superpositions of a sigmoidal function. *Math. Control Signals Syst.* **1989**, *2*, 303–314. [CrossRef]
26. Luong, K.A.; Wahab, M.A.; Lee, J.H. Simultaneous imposition of initial and boundary conditions via decoupled physics-informed neural networks for solving initial-boundary value problems. *Appl. Math. Mech.-Engl. Ed.* **2025**, *46*, 763–780. [CrossRef]