Structure Preserving PINN for Solving Time Dependent PDEs with Periodic Boundary

Baoli Hao[a], Ulisses Braga-Neto[b], Chun Liu[a], Lifan Wang[c], Ming Zhong[a]

[a] Department of Applied Mathematics, Illinois Institute of Technology, Chicago, IL, USA

[b] Department of Electrical and Computing Engineering, Texas A&M University, College Station, TX, USA

[b] Department of Physics and Astrophysics, Texas A&M University, College Station, TX, USA

April 26, 2024

Abstract

We present a structure preserving PINN for solving a series of time dependent PDEs with periodic boundary. Our method can incorporate the periodic boundary condition as the natural output of any deep neural net, hence significantly improving the training accuracy of baseline PINN. Together with mini-batching and other PINN variants (SA-PINN, RBA-PINN, etc.), our structure preserving PINN can even handle stiff PDEs for modeling a wide range of convection-diffusion and reaction-diffusion processes. We demonstrate the effectiveness of our PINNs on various PDEs from Allen Cahn, Gray Scott to nonlinear Schrödinger.

Keywords: Physics-Informed Neural Network; Reaction-Diffusion PDE; Convection-Diffusion PDE; Structure Preserving.

1 Introduction

Time-dependent partial differential equations (PDEs) hold a ubiquitous presence across numerous scientific and engineering models, underpinning various physical phenomena (Allen & Cahn, 1979; Bazant, 2017; Burgers, 1948; Cahn & Hilliard, 1958; Cassani, Monteverde, & Piumetti, 2021; De Kepper, Castets, Dulos, & Boissonade, 1991; Fibich, 2015; Horstmann, Danner, & Bessler, 2013; Hyman, Weber, & Jülicher, 2014; Kato, 1987; Kevrekidis, Rasmussen, & Bishop, 2001; Kim et al., 2016; Kudryashov, 1990; Lee et al., 2014; Michelson, 1986; Miranville, 2017; Nishiura & Ueyama, 1999; Shen & Yang, 2010; Takatori & Brady, 2015; Tian, Mao, Brown, Rutledge, & Hatton, 2015; Whitham, 2011; Zhabotinsky, 2007). Particularly, when coupled with periodic boundary conditions, these PDEs become instrumental in modeling phenomena on infinite physical domains. However, the task of solving such PDEs often poses significant challenges. In response to these challenges, we propose an innovative series of machine learning-based solvers tailored for handling time-dependent PDEs, with a specific emphasis on forward evolution of the data. The utilization of machine learning methodologies has demonstrated notable success in various scientific endeavors, ranging from the intricate process of protein folding Jumper et al. (2021), new drug discoveries Carracedo-Reboredo et al. (2021), to expediting matrix multiplication Fawzi et al. (2022). Traditional data-driven machine learning approaches rely heavily on vast amounts of training data, sometimes yielding models that lack interpretability due to their detachment from the physical understanding of the problem domain. Moreover, in numerous scientific domains, acquiring high-fidelity data proves to be both expensive and time-consuming. The emerging field of physics-informed machine learning presents a promising avenue to address these challenges. By integrating constraints derived from physical laws, physics-informed machine learning enables the prediction of complex system behaviors using sparse data Cai, Wang, Lu, Zaki, and Karniadakis (2021); X. Chen et al. (2022); Y. Chen, Hosseini, Owhadi, and Stuart (2021); Jin, Cai, Li, and Karniadakis (2021); Liu and Wang (2019, 2021); Rad, Viardin, Schmitz, and Apel (2020); Raissi and Karniadakis (2018); Raissi, Perdikaris, and Karniadakis (2019); R. Wang, Zhong, Xu, Sánchez-Cortés, and de Cominges Guerra (2023); Yang, Barajas-Solano, Tartakovsky, and Tartakovsky (2019); Zhu, Liu, and Yan (2021). However, the successful training of physics-informed machine learning models remains a formidable task Coutinho et al. (2023); McClenny and Braga-Neto (2020); S. Wang, Teng, and Perdikaris (2020); S. Wang, Yu, and Perdikaris (2020). Fundamental challenges include the difficulty in propagating information away from data, particularly in scenarios involving stiff PDEs.

In this paper, we introduce a novel algorithm designed to preserve the structural integrity of physics-informed neural networks (PINNs). Our approach involves embedding information regarding initial and boundary conditions directly into the neural network architecture, thereby reducing the reliance on spectral bases in training

various types of losses S. Wang, Yu, and Perdikaris (2020). This architectural modification not only accelerates the training process but also enhances the accuracy of solutions derived from PINNs.

Our key insight lies in recognizing that collocation-based machine learning solvers, utilized for training PINNs, can be viewed as a specialized form of regularized regression. In this framework, the initial and boundary conditions, along with observed data points, serve as the basis for training, while collocation points act as regularization data. However, the multi-objective nature of the training loss often poses challenges in finding global minimizers. Furthermore, the solution manifolds of most time-dependent PDEs exhibit sensitivity to initial and boundary conditions, where slight variations may yield vastly different outcomes. By integrating initial and boundary condition data into the neural network structure—thus preserving the underlying problem structure—we simplify the PINN training process, particularly for stiff time-dependent PDEs. Moreover, our approach can be augmented with existing training enhancement techniques, such as self-adaptive weights McClenny and Braga-Neto (2020), causal PINNs S. Wang, Yu, and Perdikaris (2020), and time-marching PINNs Wight and Zhao (2020), Residual-based Attention PINN Anagnostopoulos, Toscano, Stergiopulos, and Karniadakis (2024), co-training PINN Zhong, Liu, Arroyave, and Braga-Neto (2024), further refining prediction accuracy.

1.0.1 Related Work

The foundational PINN algorithm, while demonstrating remarkable efficacy across numerous applications, occasionally exhibits shortcomings in accurately approximating solutions or achieving convergence, particularly when applied to "stiff" PDEs, where a small time scale is required in order for traditional numerical algorithms to stay stable. Extensive research indicates that such challenges stem from an imbalance within the PINN loss function between its data-fitting and residual components (Shin, Darbon, & Karniadakis, 2020; S. Wang, Teng, & Perdikaris, 2020; S. Wang, Yu, & Perdikaris, 2020; Wight & Zhao, 2020). Notably, gradient descent tends to prioritize minimizing the residual loss over the data-fitting component, impeding convergence to the correct solution. This issue is particularly pronounced in forward problems, where all available data are confined to initial and boundary conditions, with scant information within the PDE domain. For instance, in time-evolution scenarios, neural network training struggles to propagate information from initial conditions to subsequent time steps, as corroborated by several researchers Haitsiukevich and Ilin (2022); Krishnapriyan, Gholami, Zhe, Kirby, and Mahoney (2021); McClenny and Braga-Neto (2020); S. Wang, Sankaran, and Perdikaris (2022); Wight and Zhao (2020).

In addressing these challenges, Wight and Zhao (2020) proposed a method to propagate information forward in time by segmenting the time axis into smaller intervals, sequentially training PINNs on each segment, starting from the interval closest to the initial condition. However, this approach is time-intensive due to the necessity of training multiple PINNs. Furthermore, both Wight and Zhao (2020) and McClenny and Braga-Neto (2020) emphasized the importance of weighting initial condition data and residual points near $t = 0$ to effectively propagate information forward, as evidenced in nonlinear Allen-Cahn PDE benchmarks. Notably, McClenny and Braga-Neto (2020) demonstrated that their self-adaptive PINNs autonomously learn this weighting during training, a phenomenon observed in benchmarks featuring complex initial conditions, such as the Wave equation. Similarly, Krishnapriyan et al. (2021) proposed a "sequence-to-sequence" approach, wherein the PINN sequentially learns each time step in a time-marching scheme, further addressing the propagation challenge. S. Wang et al. (2022) identified a lack of "causality" in standard PINN training and proposed a corrective weighting scheme that gradually prioritizes data from early to later times, leveraging information from the PDE residue itself, akin to the self-adaptive weighting mechanism proposed by McClenny and Braga-Neto (2020). Moreover, recent developments, such as the pseudo-label approach introduced in Haitsiukevich and Ilin (2022), bear resemblance to self-training methodologies in PINNs, further diversifying the arsenal of techniques available for enhancing convergence and accuracy in PINN-based solutions.

Finally, a characteristic-informed neural networks (CINNs) framework was introduced in Braga-Neto (2023) for training PINNs for solving transport equations, where the PDE information is built-in to the structure of the neural networks. Similar approach was proposed in Lagaris, Likas, and Fotiadis (1998) for a Dirichlet BC, our method is essential similar to that approach but with a more careful design of stability in mind.

1.0.2 Contributions of this Work

In this paper, we proposed a structure-preserving PINN algorithm by building the IC/BC information into the structure of neural networks, in order to reduce the training difficulties for PINNs for solving various stiff time-dependent PDEs.

- We develop structure preserving PINN algorithms, and investigate their stability, training speed, and prediction accuracy. The structure preserving PINN is similar to the method proposed recently in Braga-Neto (2023). However in that paper, PDE is built into the structure; whereas ours build the IC/BC into the PINNs.
- We investigate how structure preserving ameliorates the issue of propagating information forward in time, which is a common failure mode of PINNs.
- We propose the first structure preserving PINNs which incorporate both IC and BC data in a neural networks.

2 Methodology

We consider the following setup for a family of time dependent PDEs. Let u be an unknown scalar function defined on $[0,T] \times \Omega$ with $\Omega \subset \mathcal{R}^d$. Here the physical domain Ω comes with a Lipschitz boundary $\partial\Omega$. Let $\mathcal{P}$ be a spatial partial differential operator, i.e. for the heat equation $u_t - \lambda u_{xx} = 0$, $\mathcal{P} = \lambda \frac{\partial^2}{\partial x^2}$. Moreover $\mathcal{P}$ can be nonlinear. Let $\mathcal{B}$ be another partial differential operator on the boundary, i.e. for Dirichlet boundary condition $u(t,x) = g(t,x)$ for $(t,x) \in [0,T] \times \partial\Omega$, $\mathcal{B} = \mathcal{I}$ (the identity). Then we say that u is a solution of an PDE, if u satisfies the following

$$\begin{cases} u_t - \mathcal{P}(u)(t,\mathbf{x}) &= 0, \quad (t,\mathbf{x}) \in (0,T] \times \Omega, \\ u(0,\mathbf{x}) &= u_0(\mathbf{x}), \quad \mathbf{x} \in \bar{\Omega} = \Omega \cup \partial\Omega, \\ \mathcal{B}(u)(t,\mathbf{x}) &= g(t,\mathbf{x}), \quad (t,\mathbf{x}) \in [0,T] \times \partial\Omega, \end{cases} \tag{1}$$

Here the two functions u_0, g are user-input and are assumed to satisfy desired regularity so that the existence and uniqueness of solutions for such a PDE is guaranteed. We also assume that the compactibility condition, i.e., $g(0,\mathbf{x}) = u_0(\mathbf{x})$ for $\mathbf{x} \in \partial\Omega$, is satisfied in order to prevent any ill-conditioning of the PDE problem.

Types of Boundary Conditions: one can consider many kinds of boundary conditions as follows

$$\mathcal{B}(u)(t,\mathbf{x}) = u(t,\mathbf{x}), \quad \text{Dirichlet type},$$

$$\mathcal{B}(u)(t,\mathbf{x}) = \frac{\partial u}{\partial \mathbf{n}}(t,\mathbf{x}), \quad \text{Neumann type, } \mathbf{n} \text{ is the outward normal vector to } \partial\Omega,$$

$$\mathcal{B}(u)(t,\mathbf{x};a,b) = au(t,\mathbf{x}) + b\frac{\partial u}{\partial \mathbf{n}}(t,\mathbf{x}), \quad \text{Robin type}.$$

The periodic boundary condition gives d different equations, and they are given as follows,

$$\mathcal{B}(u)(t,\mathbf{x}) = u(t,\mathbf{x}) - u(t,\mathbf{x} + P\mathbf{e}_i) = 0, \quad i = 1, \cdots, d.$$

Here P is the period and $\mathbf{e}_i$ is the ith standard basis vector in $\mathcal{R}^d$, i.e.

$$\mathbf{e}_i = \underbrace{\begin{bmatrix} 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \end{bmatrix}^\top}_{1 \text{ is the in the } i^{th} \text{ position.}}.$$

One can even consider a mixture of two or all of the above BC conditions, i.e. where $\Omega = \Gamma_1 \cup \cdots \cup \Gamma_K$ and each Γ_i defines a different kind of BC. In this paper, we will mainly focus on the periodic type of boundary conditions.

Classical numerical techniques such as the Finite Difference Method (which approximates derivatives on a point mesh), Finite Element Method (utilizing basis approximation on a triangular mesh with weak formulation), and Spectral Method (employing Fourier basis with FFT on a point mesh) are widely utilized in scientific computations. However, recent advancements in scientific machine learning integrate the principles of physics, particularly the understanding of PDEs, into the training process of machine learning models. This integration enables the development of PINNs to solve for u as follows: one tries to find an approximate solution from a set of deep neural networks (of the same depth, same number of neurons on each hidden layer, and same activation functions on each layer) $\mathcal{H}_{\text{NN}}$, which is a minimizer of the following loss functional

$$\text{Loss}(u_{nn}) = \frac{1}{N_{CL}} \sum_{i=1}^{N_{CL}} |(\frac{\partial u}{\partial t} - \mathcal{P}(u_{nn}))(t_i^{CL}, \mathbf{x}_i^{CL})|^2 + \frac{1}{N_{IC}} \sum_{i=1}^{N_{IC}} |u_{nn}(0, \mathbf{x}_i^{IC}) - u_0(\mathbf{x}_i^{IC})|^2$$
$$+ \sum_{i=1}^{d} \frac{1}{N_{BC}} \sum_{j=1}^{N_{BC}} |u_{nn}(t_j^{BC}, \mathbf{x}_j^{BC}) - u_{nn}(t_j^{BC}, \mathbf{x}_j^{BC} + P\mathbf{e}_i)|^2 \qquad , \quad \forall u_{nn} \in \mathcal{H}_{\text{NN}}$$

Here $\{(t^{CL}, \mathbf{x}^{CL})_i\}_{i=1}^{N_{CL}} \in (0,T] \times \Omega$ are called collocation points, $\{(0, \mathbf{x}^{IC})_i\}_{i=1}^{N_{IC}} \in \{0\} \times \bar{\Omega}$ are the initial condition points, and $\{(t^{BC}, \mathbf{x}^{BC})_i\}_{i=1}^{N_{BC}} \in [0,T] \times \partial\Omega$ are the boundary condition points. The minimizer, denoted as u_{NN}, will an be approximate solution to (1).

2.1 Hard Constrains

The major motivation for us to consider hard constrains transformation for PINNs is the possible ill-conditioning of the PDE solution map's dependence on IC and BC. Traditional numerical methods do not need to encounter this kind of difficulty, as the IC data is exactly satisfied in their solutions. PINNs, on the other hand, use L^2-loss to fit the IC/BC and PDE residual data, causing the solutions to be highly sensitive to the goodness of fit of the IC in the training (as pointed out in McClenny and Braga-Neto (2020)). By moving the IC/BC data into the structure of PINNs, it reduces the ill-conditioning of the solutions. Therefore, We consider the following transformation

$$\tilde{u}(t,\mathbf{x}) = \psi(t,\mathbf{x}) + \phi(t,\mathbf{x})u_{nn}(t,\mathbf{x}).$$

Here the two functions ψ and ϕ are both smooth function with the following properties

$$\phi(0,\mathbf{x}) = 0 \quad \text{for } \mathbf{x} \in \bar{\Omega} \quad \text{and} \quad \phi(t,\mathbf{x}) = 0, \quad \text{for } \mathbf{x} \in \partial\Omega;$$

and

$$\psi(0, \mathbf{x}) = u_0(\mathbf{x}) \quad \text{for } \mathbf{x} \in \bar{\Omega} \quad \text{and} \quad \psi(t, \mathbf{x}) = 0, \quad \text{for } \mathbf{x} \in \partial\Omega.$$

For the Neumann and Robin type, a more complicated ψ and ϕ have to be designed. For the periodic BC, we simply require ψ and ϕ to be P-periodic. If we use such transformation, then the training for u_{nn} is changed, due to the fact that IC/BC can be built into the PDE loss, i.e.

$$\text{Loss}(u_{nn}) = \frac{1}{N_{CL}} \sum_{i=1}^{N_{CL}} |(\tilde{u}_t - \mathcal{P}(\tilde{u}))(t_i^{CL}, \mathbf{x}_i^{CL})|^2, \quad \text{for } \tilde{u} = \psi + \phi u_{nn}.$$

We have reduced the kinds of losses in the training; however the information about IC and BC has then moved into the PDE residual loss through ψ and ϕ. The difficulty in reducing the new PDE residual loss has been increased. Hence, other training enhancement techniques can be used to greatly improve training.

Periodic Neural Networks: even though we require ψ and ϕ to be periodic, we still need to build the information of periodicity into $u_n n$, hence we consider the following composition

$$u_{nn}(t, x) = f_{nn}(v(t, x)), \quad f_{nn} \text{ is a neural network, and } x \in \mathcal{R},$$

and

$$v(t, x) = \begin{bmatrix} t & 1 & \cos(\frac{2\pi}{P}x) & \sin(\frac{2\pi}{P}x) & \cos(\frac{2\pi}{P}2x) & \sin(\frac{2\pi}{P}2x) & \cdots & \cos(\frac{2\pi}{P}mx) & \sin(\frac{2\pi}{P}mx) \end{bmatrix}^\top,$$

where $m > 0$ is a positive integer (another hyper-parameter). With this composition, u_{nn} will be automatically P-periodic (Dong & Ni, 2021). For higher dimensional physical $\mathbf{x}$, similar transformation can be used. (write the $2D$ here).

2.2 Mini-batching and Other Techniques

As we have mentioned in the previous section, although we have reduced the kinds of losses to one PDE residual loss, the information about IC and BC have been transferred into the new PDE residual loss. So we consider mini-batching in the training for decreasing the PDE residual loss. As pointed out in Wight and Zhao (2020), the mini-batching technique is similar to time-marching resampling. But with mini-batching, one do not consider an explicit time-grid. Similar training enhancement can be added, such as SA PINN McClenny and Braga-Neto (2020), RBA PINN Anagnostopoulos et al. (2024), xPINN Jagtap and Karniadakis (2020).

3 Numerical Experiments

In this section, We will evaluate our proposed PINN scheme in the previous section on multiple prototypical time dependent PDEs with periodic boundary.

First of all, we use a Latin hypercube sampling (LHS) strategy to sample the collocation points in the whole domain including the boundary. We next take a simple transformation as follows

$$\tilde{u}(t, \mathbf{x}) = u_0(\mathbf{x}) * \exp(-t) + t * u_{nn}(t, \mathbf{x}), \quad (t, \mathbf{x}) \in [0, T] \times \bar{\Omega},$$

where $\psi(t, \mathbf{x}) = u_0(\mathbf{x})$ and $\phi(t, \mathbf{x}) = t$. Through the construction of the transformed u-network and the specification of appropriate loss function, we train our neural network using the generated points. Next, we utilize the trained model to approximate the solution to the partial differential equations (PDEs).

To ascertain the accuracy of our results, we use Chebfun, a spectral-style system implemented in MATLAB for handling functions in an object-oriented manner. This allows us to obtain solutions with high accuracy against which we can rigorously compare the performance of our scheme. Given the data points $\{x_i, t_i\}_{i=1}^N$ with N the total number of points, we take relative L_2 norm of the "exact" value $u(x_i, t_i)$ at those points and the network output $\mathcal{U}(x_i, t_i)$ at those points to evaluate the accuracy of our trained model:

$$\mathcal{E} = \frac{\sqrt{\sum_{i=1}^N |\mathcal{U}(x_i, t_i) - u(x_i, t_i)|^2}}{\sqrt{\sum_{i=1}^N |u(x_i, t_i)|^2}}.$$

3.1 Solving the Viscous Burger's Equation

The viscous Burger's Equation is a fundamental partial differential equation and convection-diffusion equation used in fluid dynamics and nonlinear waves (Burgers, 1948; Whitham, 2011).

For a given field $u(x, t)$, and viscosity ν, the general form of viscous Burgers' equation is:

$$\begin{aligned} u_t + u u_x &= \nu u_{xx}, \quad (x, t) \in [-1, 1] \times [0, \infty), \\ u(x, 0) &= u_0(x), \quad -1 < x < 1, \\ u(1, t) &= u(-1, t) = 0, \quad t > 0. \end{aligned}$$

The parameters we used to train are in Table 1:

# Co. Points	# Layers	# Neurons
16, 384	7	32

Table 1: PINN Params

It trains with $50k$ Adam steps with learning rate at 5×10^{-3} first, and then uses the L-BFGS-B optimizer to fine-tune the neural network. The results are shown in Figure 1 and Figure 2:

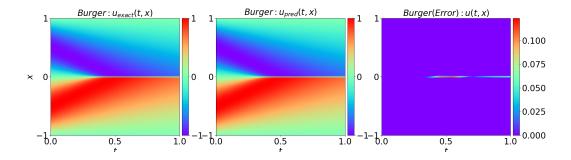Figure 1: Exact solution of the 1D Viscous Burger's Equation with the corresponding network prediction and the absolute error difference.
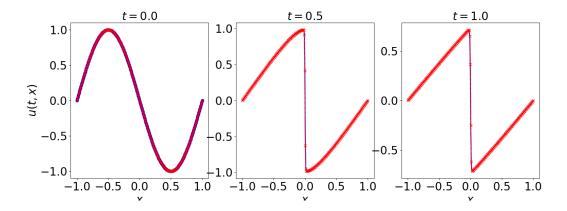

Figure 2: Solutions of 1D Viscous Burger's Equation.

3.2 Solving the Allen-Cahn Equation

Allen-Cahn type PDE is a classical phase-field model, has been widely used to investigate the phenomena of phase separation (Bazant, 2017). The Allen-Cahn equation has numerous practical applications across various fields, such as material science (Allen & Cahn, 1979; Shen & Yang, 2010), biological systems (Hyman et al., 2014; Takatori & Brady, 2015), electro-chemical systems (Horstmann et al., 2013; Tian et al., 2015) etc. We first tested the one-dimensional Allen-Cahn Equation with periodic boundary conditions, as follows:

$$
\begin{aligned}
&u_t - \gamma_1 u_{xx} + \gamma_2 u^3 - \gamma_2 u = 0, \quad (t, x) \in (0, T) \times (a, b), \\
&u(0, x) = u_0(x), \quad x \in [a, b], \\
&u(t, a) = u(t, b), \quad t \in [0, T], \\
&u_x(t, a) = u_x(t, b), \quad t \in [0, T],
\end{aligned}
\tag{2}
$$

where $\gamma_1, \gamma_2 > 0, T > 0, a < b$ are prescribed constants. As γ_2 increases, the transition interface of the solutions is sharper, which makes it harder to solve the AC equation numerically.

3.2.1 Case I

Therefore, we demonstrate the effectiveness of our scheme by testing on AC PDE with a large γ_2:

$$u_0(x) = x^2 \cos(\pi x), \quad T = 1, \quad a = -1, \quad b = 1, \quad \gamma_1 = 0.001, \quad \gamma_2 = 5.$$

We train a PINN solution using the same parameters in Table 1.
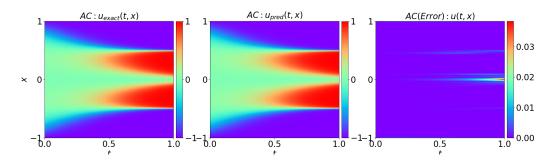
Figure 3 and **??** show the results:

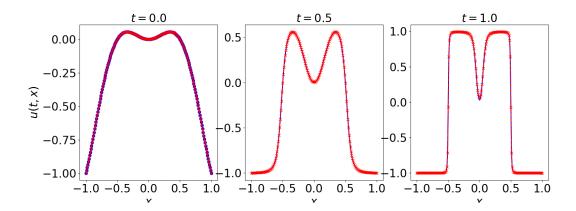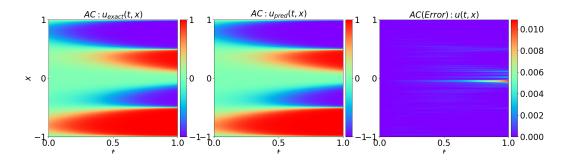Figure 3: Case I: Exact solution of the 1D AC with the corresponding network prediction and the absolute error difference.

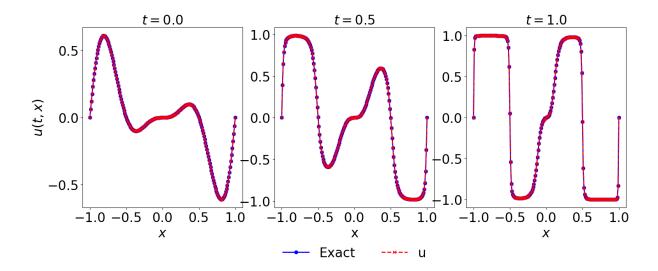Figure 4: Case I: Solutions of the AC equation $\gamma_2 = 5$.

Allen-Cahn	Baseline PINN	Re-sampling (Wight & Zhao, 2020)	Our Approach
Relative L_2	$9.90e\text{-}1$	$2.33e\text{-}2$	$9.16e\text{-}4$
Relative L_1	$9.90e\text{-}1$	$6.20e\text{-}3$	$4.84e\text{-}4$
L_∞ norm	$9.96e\text{-}1$	$2.64e\text{-}1$	$1.43e\text{-}1$

Table 2: Comparison of errors in the learned solutions of the Allen-Cahn equation using various PINN approaches

Table 2 compares the errors among the three approaches: baseline PINN, re-sampling technique proposed by Wight and Zhao (2020) and our approach. Unfortunately, using the standard PINN approach alone, we were not able to learn the accurate solution for the Allen-Cahn equation, and capture the dynamics of equation. The relative error is almost equal to 1. Compared with the technique proposed by Wight and Zhao (2020), the error is much less and the accuracy is improved.

3.2.2 Case II

Next, we explore how our enhanced PINN approach performs in handling scenarios with sharper moving interfaces. Specifically, we modify the initial conditions of the Allen-Cahn equation to introduce more oscillations, and vary

the parameters to observe the method's effectiveness across different problem settings. Instead of Equation 2 in case I, we use the following equations:

$$\begin{aligned}
&u_t - \gamma_1 u_{xx} + \gamma_2 u^3 - \gamma_2 u = 0, \quad (t,x) \in (0,1) \times (-1,1), \\
&u(0,x) = x^2 \sin(2\pi x), \quad x \in [-1,1], \\
&u(t,-1) = u(t,1), \quad t \in [0,1], \\
&u_x(t,-1) = u_x(t,1), \quad t \in [0,1],
\end{aligned} \tag{3}$$

where $\gamma_1 = 0.001, \gamma_2 = 4$.

Figure 5 and 6 show the results:

Figure 5: Case II: Exact solution of the 1D AC with the corresponding network prediction and the absolute error difference.

Figure 6: Case II: Solutions of the AC equation $\gamma_2 = 4$.

3.3 Solving the Cahn-Hilliard Equation

Related to Allen-Cahn Equation, Cahn-Hilliard Equation also describes the process of phase separation, by which the two components of a binary fluid spontaneously separate and form domains pure in each component (Cahn & Hilliard, 1958; Kim et al., 2016; Lee et al., 2014; Miranville, 2017). It is defined as the following equation:

$$\begin{aligned}
&u_t = \epsilon_1(-u_{xx} - \epsilon_2 u_{xxxx} + (u^3)_{xx}), \quad (t,x) \in (0,T] \times (-L,L), \\
&u(0,x) = u_0(x), \quad x \in [-L,L], \\
&u(t,-L) = u(t,L), \quad t \in [0,T],
\end{aligned} \tag{4}$$

where $\epsilon_1 = 10^{-2}, \epsilon_2 = 10^{-4}, T = 1, L = 1$. We specify the initial condition as $u_0(x) = -\cos(2\pi x)$. This PDE has higher order derivatives and is known to be harder to solve than the Allen-Cahn equation. We train the neural network by using 7 layers, 128 neurons and 16,384 collection points with $50k$ steps. The results are shown in the following Figures 7 and Figure 8:
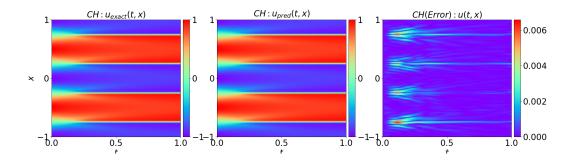
Figure 7: Exact solution of the 1D CH with the corresponding network prediction and the absolute error difference.
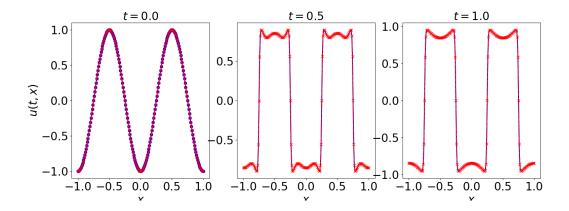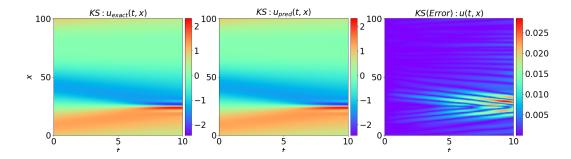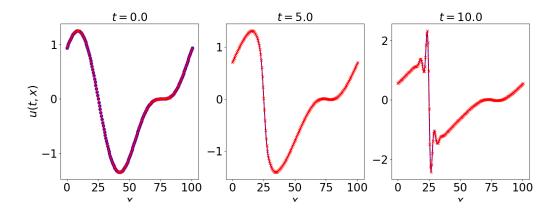

Figure 8: Solutions of the CH equation.

3.4 Solving the Kuramoto-Sivashinsky equation

Kuramoto–Sivashinsky equation is a fourth-order nonlinear partial differential equation, known for its chaotic behavior (Kudryashov, 1990; Michelson, 1986), as follows:

$$
\begin{aligned}
u_t &= -u_{xx} - u_{xxxx} - uu_x, \quad (t,x) \in (0,T] \times (a,b), \\
u(0,x) &= u_0(x), \quad x \in [a,b], \\
u(t,a) &= u(t,b), \quad t \in [0,T],
\end{aligned}
\tag{5}
$$

where $T = 20, a = 0, b = 32\pi$, and $u_0(x) = \cos\frac{x}{16}(1 + \sin\frac{x-1}{16})$. The parameters are the same with Table 1. The results are shown in Figure 9 and 10:

Figure 9: Exact solution of the 1D KS with the corresponding network prediction and the absolute error difference.

8

Figure 10: Solutions of the KS equation.

3.5 Solving the Gray-Scott Equation

Reaction and diffusion of chemical species can produce a variety of patterns (De Kepper et al., 1991; Nishiura & Ueyama, 1999), reminiscent of those often seen in nature. The Gray-Scott type system is one of classical mathematical models for chemical reactions (Gray & Scott, 1983, 1984; Liang, Jiang, Liu, Wang, & Zhang, 2022). The general irreversible GS equations describe such reactions:

$$
\begin{aligned}
U + 2V &\longrightarrow 3V, \\
V &\longrightarrow P.
\end{aligned}
\tag{6}
$$

This system is defined by two equations that describe the dynamics of two reacting substances:

$$
\begin{aligned}
u_t &= \epsilon_1 u_{xx} + b(1-u) - uv^2, \quad (t,x) \in (0,T] \times (-L,L), \\
v_t &= \epsilon_2 v_{xx} - (b+k)v + uv^2, \\
u(0,x) &= u_0(x), \quad v(0,x) = v_0(x), \quad \forall x \in [-L,L], \\
u(t,-L) &= u(t,L), \quad v(t,-L) = v(t,L), \quad \forall t \in [0,T], j
\end{aligned}
\tag{7}
$$

where $T = 20, L = 50, \epsilon_1 = 1, \epsilon_2 = 0.01$ are diffusion rates, $b = 0.02$ is the "feeding rate" that adds U, $k = 0.0562$ is the "killing rate" that removes V. We set our initial conditions as:

$$
u_0(x) = 1 - \frac{\sin(\pi(x-50)/100)^4}{2}, \quad v_0(x) = \frac{\sin(\pi(x-50)/100)^4}{4}.
$$

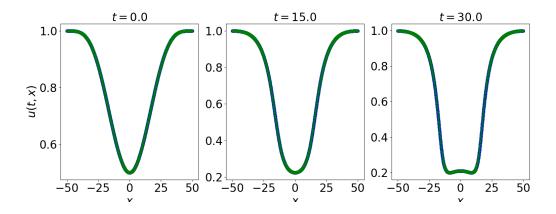The parameters that we used are the same with before. Figures 12 and **??** show the results:
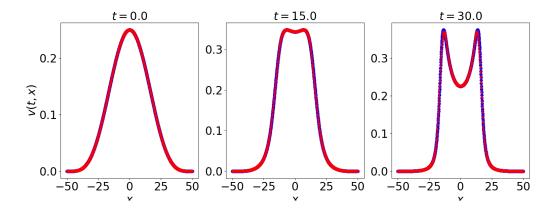

Figure 11: Solutions of the GS-u equation.

Figure 12: Solutions of the GS-v equation.

Figure 13 and Figure14 show the results of exact solutions of the 1D GS Equations that we obtain using Chebfun, with the corresponding network predictions and the absolute error differences:
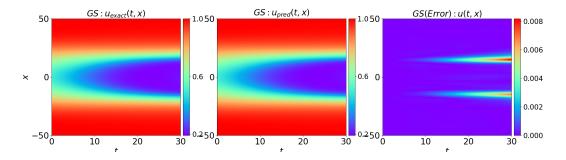

Figure 13: Exact solution of the 1D GS-u Equation with the corresponding network prediction and the absolute error difference.
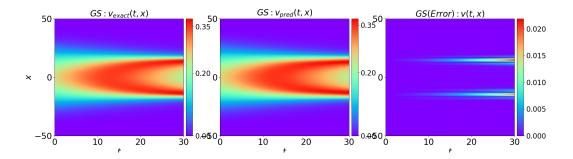

Figure 14: Exact solution of the 1D GS-v Equation with the corresponding network prediction and the absolute error difference.

3.6   Solving the Belousov-Zhabotinsky equation

A Belousov–Zhabotinsky reaction is one of a class of reactions that serve as a classical example of non-equilibrium thermodynamics. It typically involves the oxidation of organic compounds by bromine in an acidic medium

(Cassani et al., 2021; Zhabotinsky, 2007). The system is as follows:

$$u_t = \epsilon_1 u_{xx} + u + v - uv - u^2,$$
$$v_t = \epsilon_2 v_{xx} + w - v - uv,$$
$$w_t = \epsilon_1 w_{xx} + u - w, \quad (t, x) \in (0, T] \times [-L, L] \tag{8}$$
$$u(0, x) = u_0(x), \quad v(0, x) = v_0(x), \quad w(0, x) = w_0(x), \quad \forall x \in [-L, L]$$
$$u(t, -L) = u(t, L), \quad v(t, -L) = v(t, L), \quad w(t, -L) = w(t, L), \quad \forall t \in [0, T],$$

where $T = 3, L = 1, \epsilon_1 = 10^{-5}, \epsilon_2 = 2 \times 10^{-5}$ are diffusion rates. We let the initial conditions be:

$$u_0(x) = \exp(-100(x + 0.5)^2), \quad v_0(x) = \exp(-100x^2), \quad w_0(x) = \exp(-100(x - 0.5)^2).$$
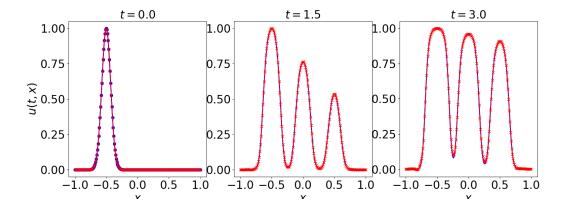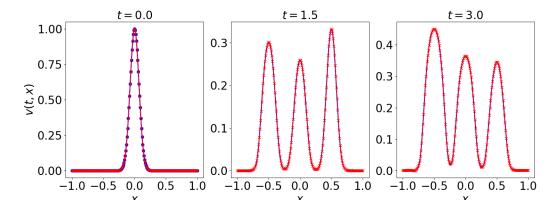
The results are:

Figure 15: Solutions of the BZ-u equation.
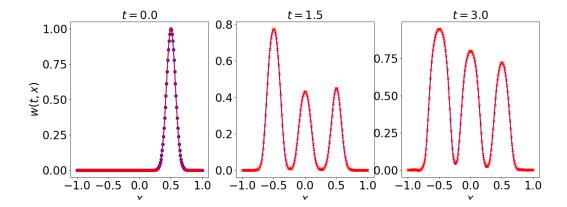

Figure 16: Solutions of the BZ-v equation.

11

Figure 17: Solutions of the BZ-w equation.

The comparison between the exact solutions of (u, v, w) equations with corresponding neural network predictions by our approach on the whole (x, t) domain are shown in Figures 18, 16 and 17:
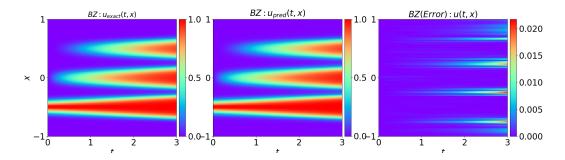

Figure 18: Exact solution of the 1D BZ-u Equation with the corresponding network prediction and the absolute error difference.
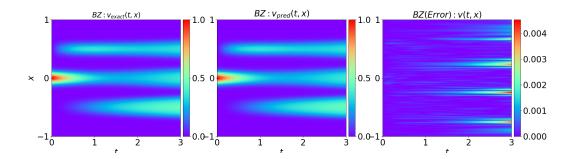

Figure 19: Exact solution of the 1D BZ-v Equation with the corresponding network prediction and the absolute error difference.

3.7 Solving the Nonlinear Schroedinger equation

In theoretical physics, nonlinear schroedinger equation is a nonlinear PDE, applicable to classical and quantum mechanics (Fibich, 2015; Kato, 1987; Kevrekidis et al., 2001). The dimensionless equation of the classical field is

$$
\begin{aligned}
u_t &= iu_{xx} + i|u|^2 u, \quad (t, x) \in [0, 2] \times [-\pi, \pi], \\
u(0, x) &= u_0(x), \quad x \in [-\pi, \pi], \\
u(t, -\pi) &= u(t, \pi), \quad t \in [0, 2],
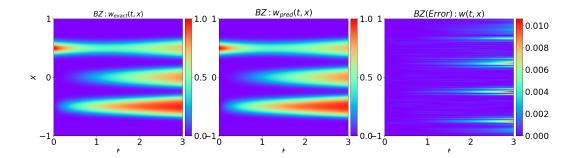\end{aligned}
\tag{9}
$$

Figure 20: Exact solution of the 1D BZ-u Equation with the corresponding network prediction and the absolute error difference.

where we let

$$u_0(x) = \frac{2}{2 - \sqrt{2}\cos(x)} - 1.$$

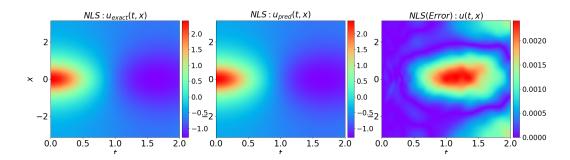Using the same parameters and training step with before, we obtain the following results:

Figure 21: Exact solution of the real part of 1D NLS with the corresponding network prediction and the absolute error difference.
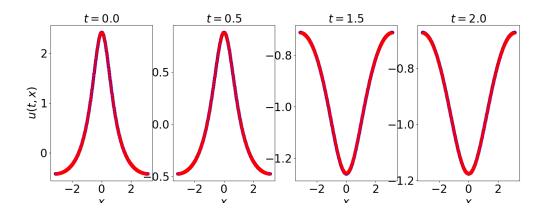

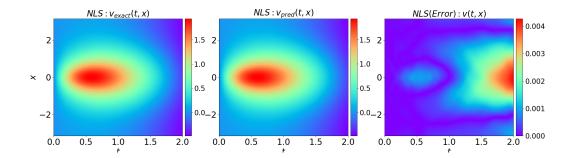Figure 22: Real-part Solutions of the NLS equation.

Figure 23: Exact solution of the imaginary part of 1D NLS with the corresponding network prediction and the absolute error difference.
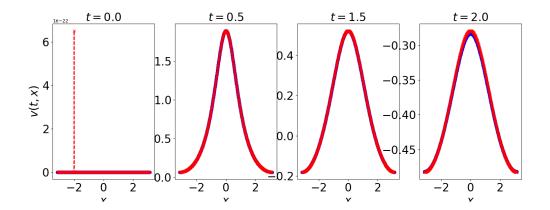

Figure 24: Imaginary-part Solutions of the NLS equation.

4   Conclusion

We introduce a structure-preserving PINN where the information about IC and BC are built-in to the structure of the neural networks. It greatly reduces the training difficulties for seven different kinds of stiff 1-dimensional time-dependent PDEs with periodic boundary condition, ranging scalar output, vector output, and to complex number output. Future work on how to extend such methods to 2- or 3-dimensional PDEs are on the work, and investigation on the optimal choice of ψ and ϕ, i.e. the transformation is ongoing.

Acknowledgements

Funding Statement

References

Allen, S. M., & Cahn, J. W. (1979). A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening. *Acta metallurgica*, *27*(6), 1085–1095.

Anagnostopoulos, S. J., Toscano, J. D., Stergiopulos, N., & Karniadakis, G. E. (2024). Residual-based attention in physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, *421*, 116805. Retrieved from https://www.sciencedirect.com/science/article/pii/S0045782524000616 DOI: https://doi.org/10.1016/j.cma.2024.116805

Bazant, M. Z. (2017). Thermodynamic stability of driven open systems and control of phase separation by electro-autocatalysis. *Faraday discussions*, *199*, 423–463.

Braga-Neto, U. (2023). *Characteristics-informed neural networks for forward and inverse hyperbolic problems.*

Burgers, J. M. (1948). A mathematical model illustrating the theory of turbulence. *Advances in applied mechanics*, *1*, 171–199.

Cahn, J. W., & Hilliard, J. E. (1958). Free energy of a nonuniform system. i. interfacial free energy. *The Journal of chemical physics*, *28*(2), 258–267.

Cai, S., Wang, Z., Lu, L., Zaki, T. A., & Karniadakis, G. E. (2021). DeepM&Mnet: Inferring the electroconvection multiphysics fields based on operator approximation by neural networks. *Journal of Computational Physics*, *436*, 110296.

Carracedo-Reboredo, P., Liñares-Blanco, J., Rodríguez-Fernández, N., Cedrón, F., Novoa, F. J., Carballal, A., ... Fernandez-Lozano, C. (2021). A review on machine learning approaches and trends in drug discovery. *Computational and Structural Biotechnology Journal*, *19*, 4538-4558. Retrieved from https://www.sciencedirect.com/science/article/pii/S2001037021003421 DOI: https://doi.org/10.1016/j.csbj.2021.08.011

Cassani, A., Monteverde, A., & Piumetti, M. (2021). Belousov-zhabotinsky type reactions: The nonlinear behavior of chemical systems. *Journal of mathematical chemistry*, *59*, 792–826.

Chen, X., Jeffery, D. J., Zhong, M., McClenny, L., Braga-Neto, U., & Wang, L. (2022). *Using physics informed neural networks for supernova radiative transfer simulation.*

Chen, Y., Hosseini, B., Owhadi, H., & Stuart, A. M. (2021). Solving and learning nonlinear PDEs with Gaussian processes. *Journal of Computational Physics*, *447*, 110668.

Coutinho, E. J. R., Dall'Aqua, M., McClenny, L., Zhong, M., Braga-Neto, U., & Gildin, E. (2023). Physics-informed neural networks with adaptive localized artificial viscosity. *Journal of Computational Physics*, *489*, 112265. Retrieved from https://www.sciencedirect.com/science/article/pii/S0021999123003601 DOI: https://doi.org/10.1016/j.jcp.2023.112265

De Kepper, P., Castets, V., Dulos, E., & Boissonade, J. (1991). Turing-type chemical patterns in the chlorite-iodide-malonic acid reaction. *Physica D: Nonlinear Phenomena*, *49*(1-2), 161–169.

Dong, S., & Ni, N. (2021). A method for representing periodic functions and enforcing exactly periodic boundary conditions with deep neural networks. *Journal of Computational Physics*, *435*, 110242.

Fawzi, A., Balog, M., Huang, A., Hubert, T., Romera-Paredes, B., Barekatain, M., ... Kohli, P. (2022). Discovering faster matrix multiplication algorithms with reinforcement learning. *Nature*, *610*, 47 - 53.

Fibich, G. (2015). *The nonlinear schrödinger equation* (Vol. 192). Springer.

Gray, P., & Scott, S. (1983). Autocatalytic reactions in the isothermal, continuous stirred tank reactor: isolas and other forms of multistability. *Chemical Engineering Science*, *38*(1), 29–43.

Gray, P., & Scott, S. K. (1984). Autocatalytic reactions in the isothermal, continuous stirred tank reactor: Oscillations and instabilities in the system a+ 2b→ 3b; b→ c. *Chemical Engineering Science*, *39*(6), 1087–1097.

Haitsiukevich, K., & Ilin, A. (2022). Improved training of physics-informed neural networks with model ensembles. *arXiv preprint arXiv:2204.05108*.

Horstmann, B., Danner, T., & Bessler, W. G. (2013). Precipitation in aqueous lithium–oxygen batteries: a model-based analysis. *Energy & Environmental Science*, *6*(4), 1299–1314.

Hyman, A. A., Weber, C. A., & Jülicher, F. (2014). Liquid-liquid phase separation in biology. *Annual review of cell and developmental biology*, *30*, 39–58.

Jagtap, A. D., & Karniadakis, G. E. (2020). Extended physics-informed neural networks (xpinns): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, *28*(5), 2002–2041.

Jin, X., Cai, S., Li, H., & Karniadakis, G. E. (2021). NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations. *Journal of Computational Physics*, *426*, 109951.

Jumper, J., R., E., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., ... Hassabis, D. (2021). Highly accurate protein structure prediction with alphafold. *Nature*, *596*, 583 - 589.

Kato, T. (1987). On nonlinear schrödinger equations. In *Annales de l'ihp physique théorique* (Vol. 46, pp. 113–129).

Kevrekidis, P., Rasmussen, K., & Bishop, A. (2001). The discrete nonlinear schrödinger equation: a survey of recent results. *International Journal of Modern Physics B*, *15*(21), 2833–2900.

Kim, J., Lee, S., Choi, Y., Lee, S.-M., Jeong, D., et al. (2016). Basic principles and practical applications

of the cahn–hilliard equation. *Mathematical Problems in Engineering*, *2016*.

Krishnapriyan, A., Gholami, A., Zhe, S., Kirby, R., & Mahoney, M. W. (2021). Characterizing possible failure modes in physics-informed neural networks. *Advances in Neural Information Processing Systems*, *34*, 26548–26560.

Kudryashov, N. A. (1990). Exact solutions of the generalized kuramoto-sivashinsky equation. *Physics Letters A*, *147*(5-6), 287–291.

Lagaris, I., Likas, A., & Fotiadis, D. (1998). Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, *9*(5), 987–1000. Retrieved from http://dx.doi.org/10.1109/72.712178 DOI: 10.1109/72.712178

Lee, D., Huh, J.-Y., Jeong, D., Shin, J., Yun, A., & Kim, J. (2014). Physical, mathematical, and numerical derivations of the cahn–hilliard equation. *Computational Materials Science*, *81*, 216–225.

Liang, J., Jiang, N., Liu, C., Wang, Y., & Zhang, T.-F. (2022). On a reversible gray-scott type system from energetic variational approach and its irreversible limit. *Journal of Differential Equations*, *309*, 427–454.

Liu, D., & Wang, Y. (2019). Multi-fidelity physics-constrained neural network and its application in materials modeling. *Journal of Mechanical Design*, *141*(12).

Liu, D., & Wang, Y. (2021). A dual-dimer method for training physics-constrained neural networks with minimax architecture. *Neural Networks*, *136*, 112–125.

McClenny, L., & Braga-Neto, U. (2020). Self-adaptive physics-informed neural networks using a soft attention mechanism. *arXiv preprint arXiv:2009.04544*.

Michelson, D. (1986). Steady solutions of the kuramoto-sivashinsky equation. *Physica D: Nonlinear Phenomena*, *19*(1), 89–111.

Miranville, A. (2017). The cahn–hilliard equation and some of its variants. *AIMS Mathematics*, *2*(3), 479–544.

Nishiura, Y., & Ueyama, D. (1999). A skeleton structure of self-replicating dynamics. *Physica D: Nonlinear Phenomena*, *130*(1-2), 73–104.

Rad, M. T., Viardin, A., Schmitz, G., & Apel, M. (2020). Theory-training deep neural networks for an alloy solidification benchmark problem. *Computational Materials Science*, *180*, 109687.

Raissi, M., & Karniadakis, G. E. (2018). Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, *357*, 125–141.

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, *378*, 686–707.

Shen, J., & Yang, X. (2010). Numerical approximations of allen-cahn and cahn-hilliard equations. *Discrete Contin. Dyn. Syst*, *28*(4), 1669–1691.

Shin, Y., Darbon, J., & Karniadakis, G. E. (2020). On the convergence and generalization of physics informed neural networks. *arXiv preprint arXiv:2004.01806*.

Takatori, S. C., & Brady, J. F. (2015). Towards a thermodynamics of active matter. *Physical Review E*, *91*(3), 032117.

Tian, W., Mao, X., Brown, P., Rutledge, G. C., & Hatton, T. A. (2015). Electrochemically nanostructured polyvinylferrocene/polypyrrole hybrids with synergy for energy storage. *Advanced Functional Materials*, *25*(30), 4803–4813.

Wang, R., Zhong, M., Xu, K., Sánchez-Cortés, L. G., & de Cominges Guerra, I. (2023). *Pinns-based uncertainty quantification for transient stability analysis.*

Wang, S., Sankaran, S., & Perdikaris, P. (2022). Respecting causality is all you need for training physics-informed neural networks. *arXiv preprint arXiv:2203.07404*.

Wang, S., Teng, Y., & Perdikaris, P. (2020). Understanding and mitigating gradient pathologies in physics-informed neural networks. *arXiv preprint arXiv:2001.04536*.

Wang, S., Yu, X., & Perdikaris, P. (2020). When and why PINNs fail to train: A neural tangent kernel perspective. *arXiv preprint arXiv:2007.14527*.

Whitham, G. B. (2011). *Linear and nonlinear waves.* John Wiley & Sons.

Wight, C. L., & Zhao, J. (2020). Solving allen-cahn and cahn-hilliard equations using the adaptive physics informed neural networks. *arXiv preprint arXiv:2007.04542*.

Yang, X., Barajas-Solano, D., Tartakovsky, G., & Tartakovsky, A. M. (2019). Physics-informed cokriging: A Gaussian-process-regression-based multifidelity method for data-model convergence. *Journal of Computational Physics*, *395*, 410–431.

Zhabotinsky, A. M. (2007). Belousov-zhabotinsky reaction. *Scholarpedia*, *2*(9), 1435.

Zhong, M., Liu, D., Arroyave, R., & Braga-Neto, U. (2024). *Label propagation training schemes for*

physics-informed neural networks and gaussian processes.

Zhu, Q., Liu, Z., & Yan, J. (2021). Machine learning for metal additive manufacturing: predicting temperature and melt pool fluid dynamics using physics-informed neural networks. *Computational Mechanics*, *67*(2), 619–635.