Commun. Comput. Phys.
doi: 10.4208/cicp.OA-2020-0086

Vol. **29**, No. 3, pp. 930-954
March 2021

Solving Allen-Cahn and Cahn-Hilliard Equations using the Adaptive Physics Informed Neural Networks

Colby L. Wight[1] and Jia Zhao[1,*]

[1] *Department of Mathematics & Statistics, Utah State University, Logan, UT 84322, USA.*

Received 12 May 2020; Accepted (in revised version) 21 July 2020

Abstract. Phase field models, in particular, the Allen-Cahn type and Cahn-Hilliard type equations, have been widely used to investigate interfacial dynamic problems. Designing accurate, efficient, and stable numerical algorithms for solving the phase field models has been an active field for decades. In this paper, we focus on using the deep neural network to design an automatic numerical solver for the Allen-Cahn and Cahn-Hilliard equations by proposing an improved physics informed neural network (PINN). Though the PINN has been embraced to investigate many differential equation problems, we find a direct application of the PINN in solving phase-field equations won't provide accurate solutions in many cases. Thus, we propose various techniques that add to the approximation power of the PINN. As a major contribution of this paper, we propose to embrace the adaptive idea in both space and time and introduce various sampling strategies, such that we are able to improve the efficiency and accuracy of the PINN on solving phase field equations. In addition, the improved PINN has no restriction on the explicit form of the PDEs, making it applicable to a wider class of PDE problems, and shedding light on numerical approximations of other PDEs in general.

AMS subject classifications: 92bxx, 65Mxx

Key words: Phase field, Allen-Cahn equation, Cahn-Hilliard equation, deep neural networks, physics informed neural networks.

1 Introduction

Phase field models have been widely embraced in the past few decades to study various problems, taking the applications in image analysis, material science, engineering, fluid

*Corresponding author. *Email addresses:* `colbywight5@gmail.com` (C. L. Wight), `jia.zhao@usu.edu` (J. Zhao) mechanics, and life science as examples. Among them, two fundamental equations are the Allen-Cahn (AC) equation and Cahn-Hilliard (CH) equation, which are originally introduced to describe the non-conservative and conservative phase variables in the phase separation process, respectively. Both models are recognized as gradient flow systems, for which there exists a Lyapunov function, known as the free energy. From a modeling view, given a specified Lyapunov function, or a free energy function, the Allen-Cahn type equations can be derived as the L^2 gradient flow, and the Cahn-Hilliard type equations can be derived as the H^{-1} gradient flow, respectively. This generality makes the AC and CH type equations extremely useful in modeling many interfacial or multiphase problems. Many well-known PDE models turn out to be their special cases.

Given the nonlinearity in phase field equations, along with the stiff terms due to the small parameters, how to design accurate, efficient, and stable numerical algorithms for their numerical approximations have been intensively studied in the literature. Here is some literature that attracts our attention [7,8,11,12,18,28,29,32,33]. Interested readers are encouraged to read them and the references therein for further information. In this paper, we focus on a new numerical approximation approach by using the deep neural network. Our major goal is to investigate strategies to improve the capabilities of deep neural networks on solving phase field models, in particular, the Allen-Cahn equation and the Cahn-Hilliard equation.

The artificial neural network is named after the fundamental unit of computation inside the mammalian brain [20]. Many neurons inside the brain work together to carry out complex tasks. Similarly, an artificial neural network is composed of multiple connected neurons that work to solve complex tasks. A single neuron in a neural network can take input from multiple neurons (or nodes). Each input has a parameter called a weight associated with it. There is also typically a bias term that doesn't have an input associated with it. The neuron receives the sum of these inputs multiplied by their weights, along with added bias. This weighted sum then goes through an activation function that gives the final output for this neuron. In the brain, a neuron usually doesn't fire unless the total of its input reaches a certain threshold. The output is either on or off. In deep learning, continuous activation functions are more commonly used [21]. The *sigmoid* function can be used as a smoother version of the step function. There are benefits in using differentiable functions like this that will help in "learning" good weights. Other useful activation functions used in deep learning include $relu$, tanh, leaky $relu$ [31] and *swish* [25]. A typical feed-forward, fully connected neural network has input going to and from multiple neurons. The input to the network makes up the input layer. The neurons of the input layer are then sent to other layers of neuron connections called hidden layers, and finally to the last layer, the output layer. See Fig. 1 for a representation of a simple neural network architecture.

Mathematically, the feed-forward neural network could be defined as compositions of nonlinear functions. Give an input $x \in \mathbb{R}^{n_1}$, and denote the output of the l-th layer as $a^{[l]} \in \mathbb{R}^{n_l}$, which is the input for $l+1$-th layer. In general, we can define the neural network