CSIAM Trans. Appl. Math.
doi: 10.4208/csiam-am.SO-2022-0014

Vol. **4**, No. 2, pp. 275-305
June 2023

BINet: Learn to Solve Partial Differential Equations with Boundary Integral Networks

Guochang Lin[1], Pipi Hu[1,3], Fukai Chen[2], Xiang Chen[4], Junqing Chen[2], Jun Wang[5] and Zuoqiang Shi[1,3,*]

[1] *Yau Mathematical Sciences Center, Tsinghua University, Beijing 100084, P.R. China.*
[2] *Department of Mathematical Sciences, Tsinghua University, Beijing 100084, P.R. China.*
[3] *Yanqi Lake Beijing Institute of Mathematical Sciences and Applications, Beijing 101408, P.R. China.*
[4] *Noah's Ark Lab, Huawei, No. 3 Xinxi Road, Beijing 100085, P.R. China.*
[5] *University College London, London, WC1E 6EA, United Kingdom.*

Received 29 March 2022; Accepted 5 September 2022

Abstract. We propose a method combining boundary integral equations and neural networks (BINet) to solve (parametric) partial differential equations (PDEs) and operator problems in both bounded and unbounded domains. For PDEs with explicit fundamental solutions, BINet learns to solve, as a proxy, associated boundary integral equations using neural networks. The benefits are three-fold. Firstly, only the boundary conditions need to be fitted since the PDE can be automatically satisfied with single or double layer potential according to the potential theory. Secondly, the dimension of the boundary integral equations is less by one, and as such, the sample complexity can be reduced significantly. Lastly, in the proposed method, all differential operators have been removed, hence the numerical efficiency and stability are improved. Adopting neural tangent kernel (NTK) techniques, we provide proof of the convergence of BINets in the limit that the width of the neural network goes to infinity. Extensive numerical experiments show that, without calculating high-order derivatives, BINet is much easier to train and usually gives more accurate solutions, especially in the cases that the boundary conditions are not smooth enough. Further, BINet outperforms strong baselines for both one single PDE and parameterized PDEs in the bounded and unbounded domains.

AMS subject classifications: 65N35, 65N80, 68T20

Key words: Partial differential equation, boundary integral, neural network, learning operator.

*Corresponding author. *Email addresses:* lingc19@mails.tsinghua.edu.cn (G. Lin), xiangchen.ai@huawei.com (X. Chen), cfk19@mails.tsinghua.edu.cn (F. Chen), hpp1681@gmail.com (P. Hu), jqchen@tsinghua.edu.cn (J. Chen), jun.wang@cs.ucl.ac.uk (J. Wang), zqshi@tsinghua.edu.cn (Z. Shi)

1 Introduction

Partial differential equations (PDEs) have been widely used in scientific fields and engineering applications, such as Maxwell's equations in optics and electromagnetism [16], Navier-Stokes equations in fluid dynamics [39], Schrödinger equations in quantum physics [36], and Black-Scholes equations for call option pricing in finance [30]. Therefore, finding solutions to PDEs has been a critical topic in many research fields. However, in most cases, the analytical solution of PDEs is infeasible to obtain, such that numerical methods become the major bridge between PDE models and practical applications. Furthermore, many problems require to solve PDEs with high dimensional parameters, which leads to unaffordable computational cost if the parameter space is discretized directly.

In the past decade, deep learning has achieved great success in computer vision, natural language processing, and many other topics [15]. Together with the deep learning revolution, solving PDEs with deep neural networks (DNNs) also enter a period of prosperity. Due to the attractive capability in approximating functions, especially in high dimensional space, DNNs hold great potential in solving PDEs with the promise of providing a good ansatz to represent the solution. In the literature, many efforts have been devoted to developing DNN-based methods for solving various kinds of PDEs, such as DGM [37], PINN and its variants [31,34,35], Deep Ritz [42], and so on. The main idea of these methods is to use neural networks to approximate the solution of the PDE directly, and the two key ingredients that characterize these neural network-based PDE solvers are loss function and network structure.

Regarding the loss function, one natural choice is the residual of PDE. In [35,37], the L_2 norm of the residual is used as the loss function. For elliptic equations, the variation form provides another choice of the loss function. Yu and E proposed to use Ritz variational form as the loss function in [42] and Galerkin variational form was formulated as an adversarial problem in [45]. In [9,29], to avoid high order derivatives in the loss function, high order PDEs are firstly transformed to first-order PDEs system by introducing auxiliary variables, and thus only first-order derivatives need to be computed in the loss function. To solve PDEs, boundary condition has to be imposed properly. One simple way to enforce the boundary condition is to add it to the loss function as a penalty term. In this approach, we must tune a weight to balance the PDEs' residual and boundary conditions. Usually, this weight is crucial and subtle to get good results. The other way is to impose the boundary condition explicitly by introducing a distance function of the boundary [6].

As for network structure, there are also many works recently. A fully connected neural network (FCN) is one of the most frequently used networks. In [42], it is found that residual neural network (ResNet) gives better results. For PDEs with multiscale structures, a multiscale neural network was designed specifically by introducing multiscale structure in the network [8]. The activation function is another important component of neural networks. Choice of the activation function is closely related to the smoothness