AIAA SciTech Forum
January 3-7, 2022, San Diego, CA & Virtual
AIAA SCITECH 2022 Forum

10.2514/6.2022-2353

Critical Investigation of Failure Modes in Physics-informed Neural Networks

Shamsulhaq Basir [*] and Inanc Senocak[†]
University of Pittsburgh, Pittsburgh, PA, 15261

Several recent works in scientific machine learning have revived interest in the application of neural networks to partial differential equations (PDEs). A popular approach is to aggregate the residual form of the governing PDE and its boundary conditions as soft penalties into a composite objective/loss function for training neural networks, which is commonly referred to as physics-informed neural networks (PINNs). In the present study, we visualize the loss landscapes and distributions of learned parameters and explain the ways this particular formulation of the objective function may hinder or even prevent convergence when dealing with challenging target solutions. We construct a purely data-driven loss function composed of both the boundary loss and the domain loss. Using this data-driven loss function and, separately, a physics-informed loss function, we then train two neural network models with the same architecture. We show that incomparable scales between boundary and domain loss terms are the culprit behind the poor performance. Additionally, we assess the performance of both approaches on two elliptic problems with increasingly complex target solutions. Based on our analysis of their loss landscapes and learned parameter distributions, we observe that a physics-informed neural network with a composite objective function formulation produces highly non-convex loss surfaces that are difficult to optimize and are more prone to the problem of vanishing gradients.

I. Nomenclature

Ω	=	bounded domain
$\partial\Omega$	=	domain boundary
$\mathcal{D}$	=	differential operator on the interior part of the domain
g	=	source function on the boundary of the domain
$N_{\partial\Omega}$	=	number of points on the boundary of the domain
N_{Ω}	=	number of collocation points on the interior part of the domain
u	=	variable of interest
$\hat{u}_{\theta}$	=	approximated variable of interest
θ	=	parameters of the neural network model
$\mathcal{N}$	=	physics-based neural network model
λ	=	Lagrange multipliers
$\mathcal{C}$	=	equality constraint vector
μ	=	penalty parameter
μ_{∞}	=	safe-guarding penalty parameter
$L_{2,r}$	=	relative 2-norm of error

[*]Graduate Student, Department of Mechanical Engineering and Materials Science, shb105@pitt.edu, Student Member AIAA
[†]Associate Professor, Department of Mechanical Engineering and Materials Science, senocak@pitt.edu, Associate Fellow AIAA

II. Introduction

Deep learning has shown tremendous success in the pattern recognition [1, 2], speech recognition [3], and natural language processing [4–6]. The success of these models exudes collectively from the fast development of accessible information, increment in computing power, and advanced learning algorithms that have made strides in the understanding of neural networks [7]. The introduction of the universal approximation theorem [8, 9] has stimulated new studies using neural networks to solve ODEs and PDEs. Dissanayake and Phan-Thien [10] pioneered the use of neural networks to solve PDEs, where they assembled the residual form of a given PDE and its boundary conditions as soft-constraints for training their neural network model. van Milligen et al. [11] presented a similar approach and showed its potential on a magnetohydrodynamics plasma equilibrium problem. The general neural network-based approach that was proposed in [10, 11] was applied with satisfactory outcomes to the non-linear Schrodinger equation in [12], to a non-steady fixed bed non-catalytic solid/gas reactor problems in [13], and to the one-dimensional Burgers equation in [14]. Lagaris et al. [15] proposed a neural network-based approach for the solution of ODEs and PDEs on orthogonal box domains by forming trial functions that satisfy the boundary conditions by construction. It is worth mentioning that these early works did not find broader adoption or appreciation by other researchers likely because of a lack of computational resources and an incomplete understanding of neural networks at the time of their introduction. Modern machine learning frameworks with automatic differentiation capabilities [16, 17] have revived the use of neural networks to solve ODEs and PDEs. Several recent investigations [18–20] adopted the overall approach and formulation proposed in [10, 11]. Raissi et al. [19] coined the term physics-informed neural networks (PINNs) to describe this technique, which has sparked considerable interest in the application of neural networks for the solution of problems involving PDEs and ODEs.

In this work, we investigate the failure modes of the PINN approach in the context of elliptic PDEs. We formulate an equivalent but purely data-driven objective function composed of the boundary loss and the domain loss term to contrast it against the loss function formulation adopted in the PINN approach. We then train two neural network models of the same architecture with the physics-informed and the purely data-driven objective functions and investigate their performances by visualizing their loss landscapes and the distribution of their learned parameters. The paper is organized as follows. Section III presents the technical background on supervised and unsupervised learning approaches. In section IV we compare three objective/loss formulations, one of which is the recently introduced *Physics and Equality Constrained Artificial Neural Networks*(PECANNs) [21] through numerical experiments on two elliptic PDE problems and provide insights from these comparisons. Finally, section V concludes the paper with a discussion of our findings and future directions.

III. Technical Background

In this section, we start with describing supervised regression for function approximation using data from the exact solution of the PDE alone. We then describe physics informed/constrained neural networks for the solution of PDEs. Let us consider a scalar function $u(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}$ satisfying the following differential equation

$$\mathcal{D}(\boldsymbol{x}, u) := 0 \text{ in } \Omega, \tag{1}$$

$$u(\boldsymbol{x}) := g(\boldsymbol{x}) \text{ on } \partial\Omega, \tag{2}$$

where Ω is the domain and $\partial\Omega$ is its boundary. We aim to learn a parameterized solution $u_\theta(x)$ represented by a neural network model that satisfies the governing equation and its boundary condition as in Eq. (1). We consider two different learning approaches. In the first approach, the solution u_θ is learned in a supervised fashion should there exist enough training data. For our investigation, we derive the training data from the exact solution to the PDE. In the second approach, the underlying physics through the governing PDE can be accounted for in the objective function for training a neural network model to learn u_θ in an unsupervised fashion. A schematic representation of both learning approaches are depicted in Fig 1.

A. Supervised Learning Method

In this section, we describe a purely data-driven learning approach. We aim to have a composite loss function with no scale disparity between the domain loss and the boundary loss. To this end, we assume that there are sufficient training data on the boundary and in the domain. We describe a purely data-based objective formulation as a sum of

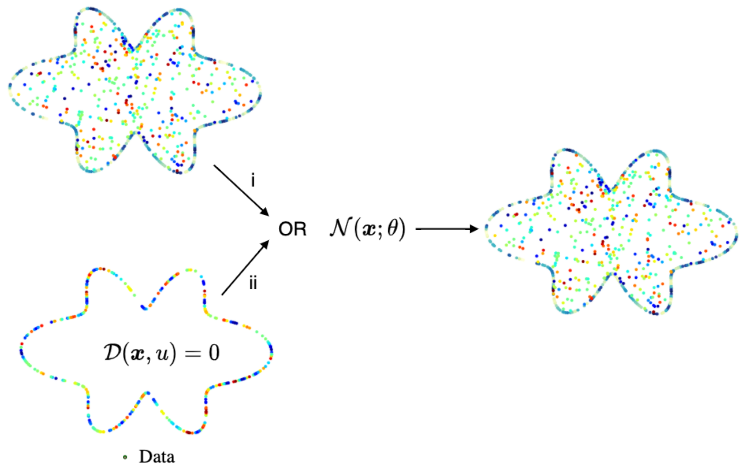
i
ii
OR
$\mathcal{N}(\boldsymbol{x};\theta)$
$\mathcal{D}(\boldsymbol{x},u)=0$
• Data

Fig. 1 Schematic representation of learning approaches with an artificial neural network model $\mathcal{N}$ with its inputs x and its parameters θ: the *i* branch represent supervised learning approach, whereas the *ii* branch represents physics-based unsupervised learning approach

mean-squared-errors (MSE) of loss terms in the domain and on the boundaries as in Eq. (5)

$$\mathcal{L}_\Omega = \frac{1}{N_\Omega}\sum_{i=1}^{N_\Omega}\left[u_\theta(\boldsymbol{x}^{(i)}) - u(\boldsymbol{x}^{(i)})\right]^2, \tag{3}$$

$$\mathcal{L}_{\partial\Omega} = \frac{1}{N_{\partial\Omega}}\sum_{j=1}^{N_{\partial\Omega}}\left[u_\theta(\boldsymbol{x}^{(j)}) - g(\boldsymbol{x}^{(j)})\right]^2, \tag{4}$$

$$\mathcal{L} = \mathcal{L}_\Omega + \mathcal{L}_{\partial\Omega}, \tag{5}$$

where $\mathcal{L}$ is the objective function, $N_{\partial\Omega}$ and N_Ω are the number of training points on the boundary and in the domain respectively.

B. Unsupervised Learning Methods

In this section, we present the popular physics-informed neural network method and a newly introduced, alternative method based on a constrained optimization formulation of the artificial neural networks Basir and Senocak [21].

1. PINN: Physics-informed Neural Networks

Here, we describe the common elements of the physics-informed learning framework presented in the works of Dissanayake and Phan-Thien [10], van Milligen et al. [11], and Raissi et al. [22]. In all these works, the domain and boundary loss terms are represented as a mean squared error (MSE).

$$\mathcal{L}_\Omega = \frac{1}{N_\Omega}\sum_{i=1}^{N_\Omega}\left[\mathcal{D}(\boldsymbol{x}^{(i)}, u_\theta(\boldsymbol{x}^{(i)})\right]^2, \tag{6}$$

$$\mathcal{L}_{\partial\Omega} = \frac{1}{N_{\partial\Omega}}\sum_{j=1}^{N_{\partial\Omega}}\left[g(\boldsymbol{x}^{(j)}) - u_\theta(\boldsymbol{x}^{(j)})\right]^2, \tag{7}$$

$$\mathcal{L} = \mathcal{L}_\Omega + \mathcal{L}_{\partial\Omega} \tag{8}$$

Note that each loss term is scaled or weighted by the number of training points. The aggregate loss $\mathcal{L}$ constitutes a composite objective function.

2. PECANN: Physics & Equality Constrained Artificial Neural Networks

In this section, we describe a recent physics-based unsupervised learning approach that combines a constrained optimization approach with modern deep learning practices for the solution of problems involving PDEs and ODEs [21]. The core idea is to minimize a domain loss based on the residual form of Eq. (1) such that it is constrained by the boundary conditions (i.e. Eq. 2). This formally constrained optimization formulation is then put into practice for training an artificial neural network architecture using the Augmented Lagrangian method (ALM) [23, 24].

$$\mathcal{L}_\Omega = \sum_{i=1}^{N_\Omega} \left[\mathcal{D}(\boldsymbol{x}^{(i)}, u_\theta(\boldsymbol{x}^{(i)})\right]^2 \tag{9}$$

$$\mathcal{L}_\mu = \mathcal{L}_\Omega + \boldsymbol{\lambda}^T \mathcal{C} + \frac{\mu}{2}\|\mathcal{C}\|^2, \tag{10}$$

$$\boldsymbol{\lambda} = \boldsymbol{\lambda} + \mu\mathcal{C}, \tag{11}$$

where μ is a positive penalty parameter with a maximum safeguarding value of μ_∞, $\boldsymbol{\lambda}^T = \{\lambda^{(1)}, \cdots, \lambda^{(N_{\partial\Omega})}\}$ is a vector of Lagrange multipliers, $\mathcal{C}^T = \{[g(\boldsymbol{x}^{(1)}) - u_\theta(\boldsymbol{x}^{(1)})]^2, \cdots, [g(\boldsymbol{x}^{(N_{\partial\Omega})}) - u_\theta(\boldsymbol{x}^{(N_{\partial\Omega})})]^2\}$ is a vector of evaluated boundary constraints and T is transposition.

IV. Numerical Experiments

In this section, we present an in-depth investigation of how supervised and unsupervised neural networks perform through loss landscape visualizations and distributions of learned parameters. We target elliptic problems as they are central to the numerical solution of incompressible flows.

A. Simulation Parameters

We use a feed-forward neural network with eight hidden layers and 20 neurons per layer with tanh activation functions. We use Adam [25] with 10^{-2} initial learning rate and ReduceLROnPlateau(patience=100,factor=0.90) as the learning rate scheduler. Given two sets of vectors $\hat{\boldsymbol{u}}$, the learned solution, and $\boldsymbol{u}$, the exact solution, we define the relative $L_{2,r}$ of $\hat{\boldsymbol{u}}$ as follows,

$$L_{2,r} = \frac{\|\hat{\boldsymbol{u}} - \boldsymbol{u}\|_2}{\|\boldsymbol{u}\|_2}. \tag{12}$$

B. One Dimensional Poisson's Equation

$$\frac{d^2u(x)}{dx^2} = f(x), \text{ in } \Omega, \tag{13}$$

$$u(x) = g(x), \text{ on } \partial\Omega, \tag{14}$$

where $f(x)$ and $g(x)$ are forcing functions, $\Omega = \{x \mid 0 \geq x \leq 1\}$ and $\partial\Omega$ is its boundary. We manufacture a solution of the form $u(x;\omega) = \sin(\omega\pi x)$ and use it to calculate $f(x)$ and $g(x)$. The particular form the manufactured solution is desirable as we can increase its complexity simply by increasing ω. For the purpose of supervised learning, we generate 600 training data in the domain and two boundary data from the boundary conditions and simply fit the data and learn the function. For the purpose of physics based learning, we generate 600 collocation points that we use to calculate on the residual form of Eq. (14) and two boundary points with their respective boundary values from the exact equation at every epoch.

1. Predictions

In this section, we present predictions for increasingly complex functions from the neural network models trained using supervised and unsupervised learning approaches. The results of our experiment is presented in Fig 2.

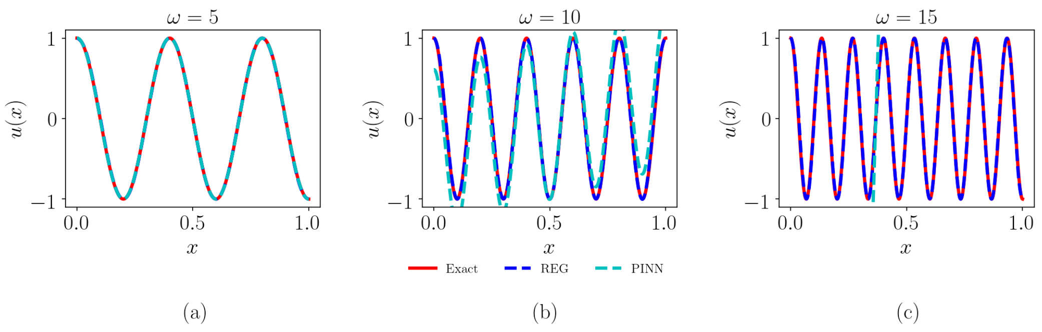

Fig. 2 ***One dimensional Poisson's equation: predicted solution $\hat{u}(x)$ from PINN model (dashed cyan) and purely data-driven regression model (dashed blue)*** **(a) wave number $\omega = 5$ with $L_{2,r} = 1.028 \times 10^{-2}$ and $L_{2,r} = 8.269 \times 10^{-4}$ for PINN and regression model respectively, (b) wave number $\omega = 10$ with $L_{2,r} = 2.087 \times 10^{-2}$ and $L_{2,r} = 3.195 \times 10^{-1}$ for PINN and regression model respectively, (c) wave number $\omega = 15$ with $L_{2,r} = 3.077 \times 10^{-2}$ and $L_{2,r} = 1.652 \times 10^{+1}$ for PINN and regression model respectively**

We observe that as we increase the complexity of the target function, the neural network model trained with a supervised regression approach converges under fixed hyperparameter settings. However, the same behavior is not observed from the same neural network model trained using PINN learning approach. Thus far, we know that for the supervised learning approach, both terms in the loss function have a similar scale without any derivative terms as seen in Eq. (5). However, for the physics-informed learning approach, we have a disparity of scales between each loss term since the calculation of the residual form of the PDE in Eq. (1) involves derivative terms. Therefore, we observe that aggregating two terms with different scales into a single objective function works for a fairly simple target solution and fails or produces erroneous results with challenging target functions. By this experiment, we showed that this failure is not due to inadequacy of expressivity in the neural network architecture but rather due to the formulation of physics-informed objective/loss function.

2. *Investigations*

We now investigate the above failure mode by comparing the histograms of learned parameters by both models as well as their loss surfaces. In Fig. 3, we present histograms of learned parameters by the purely data-driven supervised neural-network model and the unsupervised PINN model. Our aim here is to investigate the differences between the histograms of parameters in both models. It is worth noting that we initialize all the parameters in the network with Xavier initialization technique[26].

From Figs. 3(a-c), we observe that as we increase the complexity of the target function (i.e. larger ω), weight histograms of the neural network model trained using supervised learning approach stay fairly the same for all the cases. However, increasing the complexity of the target function (i.e. larger ω) changes the distribution of learned weights for the unsupervised PINN model as seen in Figs. 3(d-f). We attribute this observation for physics-informed neural network to the derivatives in the governing equation and conclude that learning becomes harder for physics-informed neural networks when the target problem gets complex. We further investigate the loss landscapes by perturbing the trained models across two random directions [27]. We present loss surfaces extracted from the neural-network models trained using both learning approaches in Fig. 4 We observe that as we increase the complexity of target function from $\omega = 5$ to $\omega = 15$, loss surfaces of the neural network model trained using supervised learning approach behave well with a visible minimum as seen in Figs. 4(a)-(c). However, with the increasing complexity of the target function, loss surfaces the PINN model becomes complex and hard to optimize as seen in Fig. 4(d)-(f). These experiments show that for physics-informed neural networks loss surfaces become complex with the complexity of the target function which is not desirable. Because learning complex functions become challenging as in the case of $\omega = 15$.

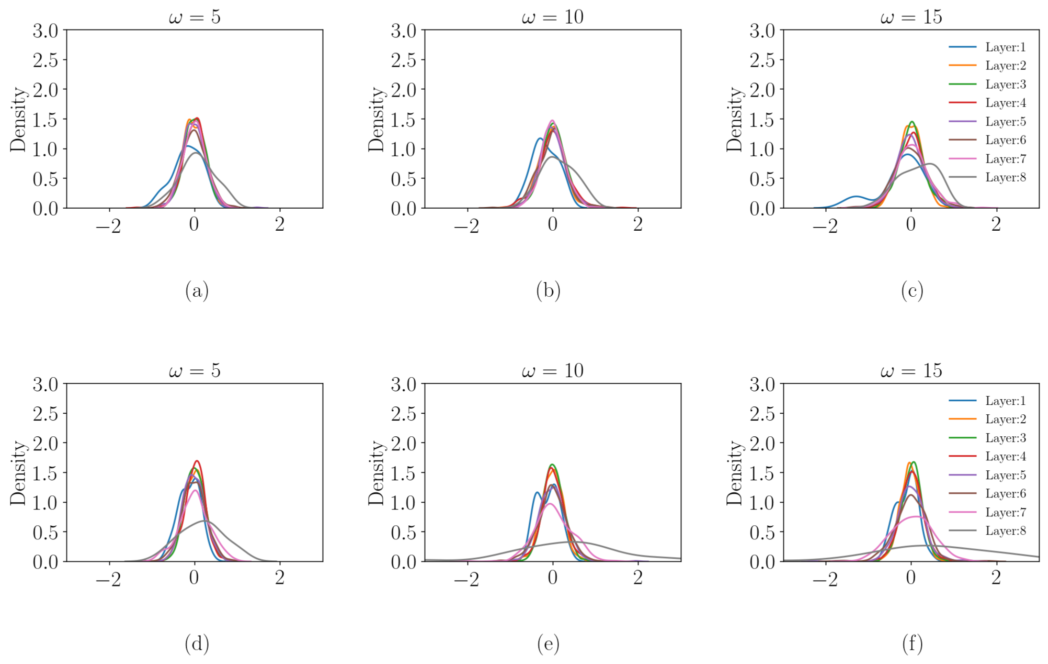

Fig. 3 ***Weight histograms of neural network models trained using two different approaches*** **Top row: using supervised learning approach. (a) wave number $\omega = 5$, (b) wave number $\omega = 10$, (c) wave number $\omega = 15$. Bottom row: using PINN approach. (d) wave number $\omega = 5$, (e) wave number $\omega = 10$, (f) wave number $\omega = 15$.**

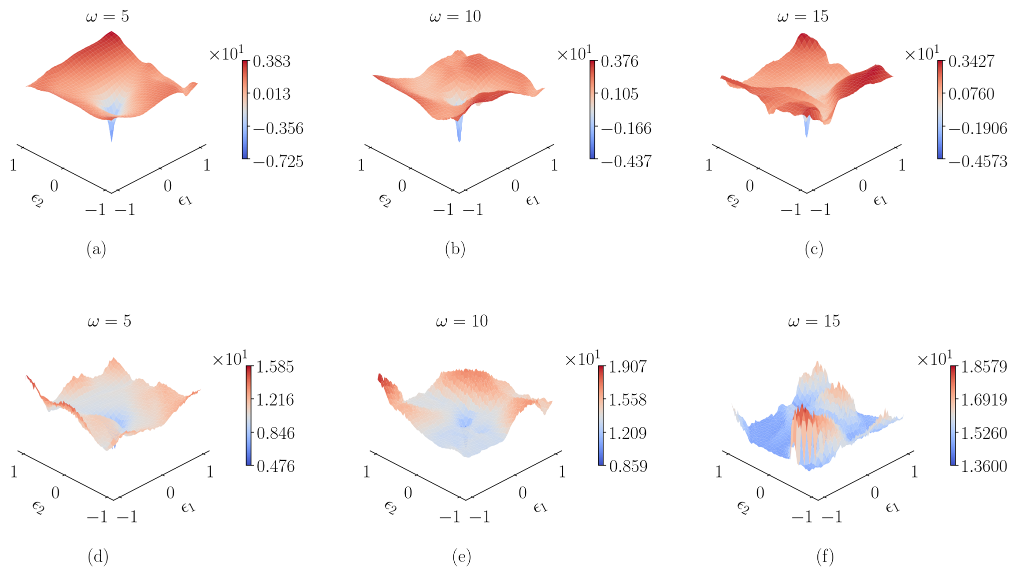

Fig. 4 ***Loss landscape of the neural network model.*** **Top row: purely-data driven regression model. (a) wave number $\omega = 5$, (b) wave number $\omega = 10$, (c) wave number $\omega = 15$. Bottom row: physics-informed neural network model. (d) wave number $\omega = 5$, (e) wave number $\omega = 10$, (f) wave number $\omega = 15$.**

3. Physics and Equality Constrained Artificial Neural Networks

Through numerical experiments, we have observed that it is possible to improve the predictions with the PINN model and its composite objective function formulation by manually tuning the weight of the boundary loss term. However, the composite objective function with user-defined weights is an inchoate optimization problem formulation, to begin with, and it is at the root of several performance issues associated with the PINN model. In Basir and Senocak [21], the authors pursued a constrained-optimization formulation and proposed the Physics& Equality Constrained Artificial Neural Networks (PECANNs). In this framework, the authors use the Augmented Lagrangian method (ALM) [23, 24] to convert a constrained-optimization problem to an unconstrained optimization problem suitable for neural networks. In this work, we show that training the same neural network architecture with the PECANN approach easily captures the target function with increasing difficulty. We present the predictions from our PECANN model and the exact solution in Fig. 5.

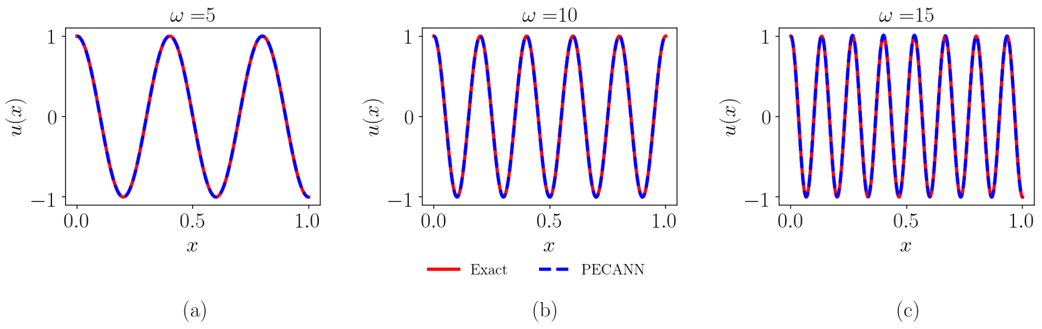

Fig. 5 ***One dimensional Poisson's equation: predicted solution $\hat{u}(x)$ from PECANN model (dashed cyan) versus exact solution* (a) wave number $\omega = 5$ with $L_{2,r} = 1.060 \times 10^{-4}$, (b) wave number $\omega = 10$ with $L_{2,r} = 6.333 \times 10^{-4}$, (c) wave number $\omega = 15$ with $L_{2,r} = 9.627 \times 10^{-3}$**

We further investigate our trained model by presenting the histogram of its parameters for each layer in Fig. 6.

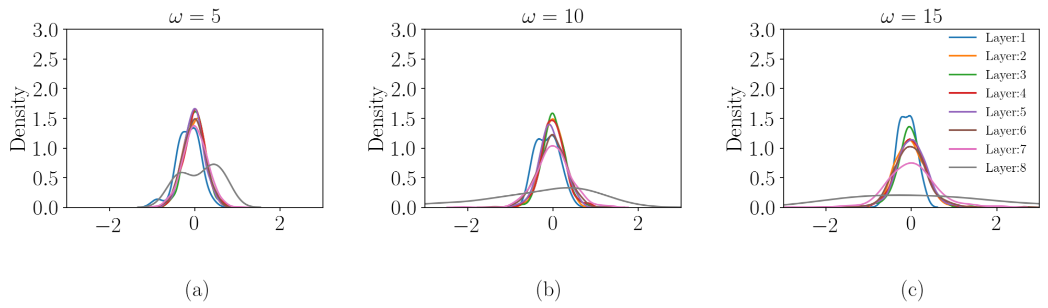

Fig. 6 ***Weight histograms of the PECANN model for one dimensional Poisson's equation.* (a) wave number $\omega = 5$, (b) wave number $\omega = 10$, (c) wave number $\omega = 15$.**

From Fig. 6 we observe that as we increase the complexity of the target function, the histograms of weights change as was the case with the PINN model. This shift of weight distribution between hidden layers as we increase the complexity of the target function is due to structured physics in the objective function. In [21], authors propose a simple residual neural network that has no extra weight or activation function compared with a vanilla neural network that addresses this issue and significantly speeds up training. For further insight, we present loss surfaces for three target functions in Fig. 7. From Fig. 7(a-c) we observe that loss landscapes stay smooth with a clear local minimum that the optimizer has found. Unlike the PINN model, our PECANN model has smooth loss landscapes that are easier to optimize. This is attributed to the unconstrained optimization formulation of the objective function that properly constrains the boundary conditions and addresses the scale discrepancy.

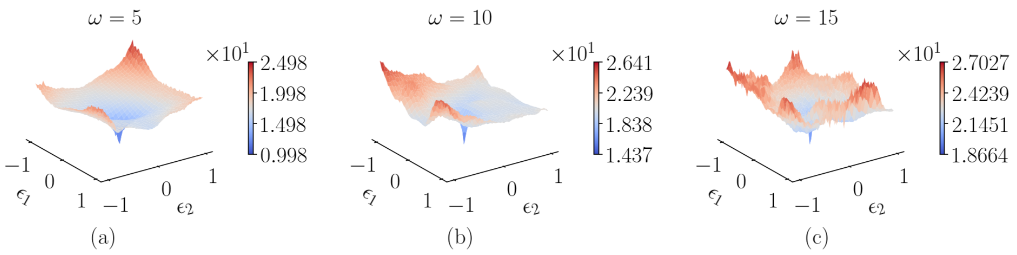

Fig. 7 ***Loss landscape of PECANN model for one dimensional Poisson's equation.*** **(a) wave number $\omega = 5$, (b) wave number $\omega = 10$, (c) wave number $\omega = 15$**

C. Two Dimensional Poisson's Equation

In the previous numerical experiment, we have investigated the effect of scale by comparing a supervised learning approach, in which the domain and the boundary loss terms had the same scale, against an unsupervised learning approach where the domain loss term had derivative terms whereas its boundary loss term did not because of the imposed Dirichlet boundary conditions. We should note that the mean squared error is averaged of the number of training points for the domain and the boundary loss terms. Because we have only two boundary points in a one-dimensional problem, this exercise was not amenable to an in-depth investigation. Therefore, in this section, we consider a two-dimensional problem so that average loss values calculated for the boundary and the domain can be computed over the equal number of points. Hence, the purely data-driven supervised neural network model not only has the same scale for the domain and the boundary loss terms but is also averaged over equal number of points in both of the loss terms. Similarly, for the physics-informed neural network model, we have the equal number of points for both the domain and the boundary loss terms, but they end up having different scales because the domain loss term includes derivatives, whereas the boundary loss term does not because of the Dirichlet type conditions. This aspect of the problem enables us to observe the effect of structures or derivative terms in the domain loss in the physics-informed neural-network model. As for our PECANN model, which is also an unsupervised model, we do not have the issue of scaling anymore as it is taken care of by the augmented Lagrange method. Also regardless of where our PECANN model converges in the space of solutions, we make sure the obtained solutions satisfy the constraints or in other words, the solution is feasible.

We consider the following two-dimensional non-homogeneous Poisson's equation:

$$\nabla^2 u(x, y) = f(x, y), \ \ (x, y) \text{ in } \Omega, \tag{15}$$

$$u(x, y) = g(x, y), \ \ (x, y) \text{ on } \partial\Omega, \tag{16}$$

where $\Omega = \{(x, y) \mid 0 \leq x \leq 1, \ 0 \leq y \leq 1\}$, f and g are source functions across the domain and the boundary respectively. We manufacture a solution as follows

$$u(x, y) = \cos(15\pi x)\exp(-\pi y), \qquad \forall(x, y) \text{ in } \Omega. \tag{17}$$

In the case of supervised learning, we generate 600 training data in the domain and 600 boundary data from the Dirichlet boundary conditions and learn the function by simply fitting the data with a neural network. In the case of unsupervised, physics-based learning, we generate 600 collocation points that we use to calculate the residual form of Eq. (16) and 600 boundary points with their respective boundary values from the exact equation Eq. (17) at every epoch. We use the same network architecture, optimizer, and learning rate scheduler as in the previous problem. We set $\mu_\infty = 500$ in the PECANN model. We present the predicted solutions from three neural network models trained with the purely data-driven supervised learning approach and the unsupervised learning approaches with PINN and PECANN.

From Fig. 8(b), we observe that the purely data-driven supervised neural network model has converged to the exact solution which implies that our neural network architecture is capable of representing the solution function. However, Fig. 8(c), shows that the unsupervised physics-informed neural network model has diverged. We note that the only major difference between the supervised model and the PINN model is the loss term for the domain. Therefore, we see that lumping two loss terms with different scales impedes the convergence of neural network models. However, unlike the PINN model, we observe from Fig. 8(d) that the unsupervised PECANN model has converged to the correct solution. In the PECANN model, the difference in the scale of the domain and boundary loss terms are taken care of by the augmented Lagrangian method which gets updated during training to make sure the network converges to a minimum such that the constraints are satisfied.

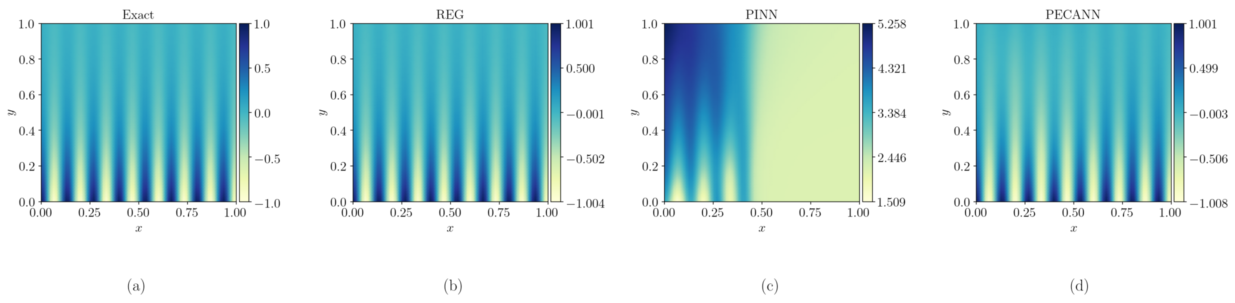

Fig. 8 Two dimensional Poisson's equation: (a) exact solution $u(x, y)$, (b) predicted solution from purely data-driven supervised neural network model, (c) predicted solution from unsupervised physics-informed neural network model, (d) predicted solution from unsupervised physics-and-equality constrained artificial neural network model

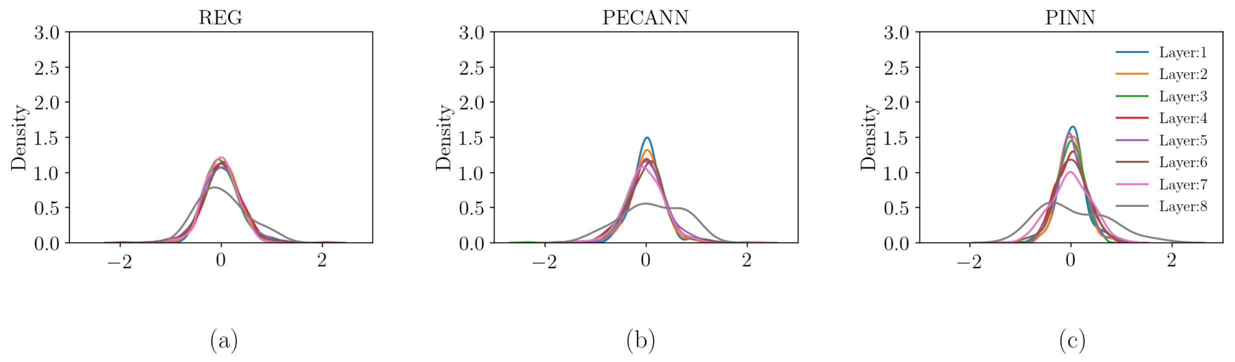

Fig. 9 Weight histograms of neural network models trained for the solution of a two dimensional Poisson's equation: (a) purely data-driven supervised neural network model, (b) unsupervised physics-informed neural network model, (c) unsupervised physics-and-equality constrained artificial neural network model

In Fig. 9, we present histograms of the learned weights for all the neural network models considered for the two-dimensional Poisson's equation problem. From Fig. 9(a), we observe that layers 1 through 7 have similar means and variances. However, having a slightly higher peak in their distribution than layer 8 indicates that they must have suffered from a mild vanishing gradient problem. However, Fig. 9(b -c), show that embedding the physics into the neural network slightly exacerbates the problem of vanishing gradient regardless of the scale difference between the loss terms. Because PECANN model does not have the issue of scale disparity but the parameter distribution of PECANN and PINN do not have any significant difference. This is a crucial observation and is because during training, predictions from the network are naturally noisy and their derivatives end up being noisy as well. Therefore, the neural network model is more vulnerable to the issue of vanishing gradients. As mentioned earlier this issue was addressed in [21] with their proposed residual neural network architecture. So far, we have shown that aggregating multiple loss terms without properly balancing them impedes the convergence of the neural network model and can cause the network to suffer from the vanishing gradient problem. At this point, we visualize the loss landscapes of these three models to gain more insight. From Fig. 10(a), we observe the loss landscape of the purely data-driven supervised neural network model is smooth with a clear minimum that the optimizer was able to find. In Fig. 10(b), we observe that the PECANN model also has a smooth loss landscape with a clear minimum that is found by the optimizer. However, Fig. 10(b) shows the loss landscape for the PINN model is highly non-convex with flat regions. Therefore, the optimizer was not able to get out and find a better local minimum.

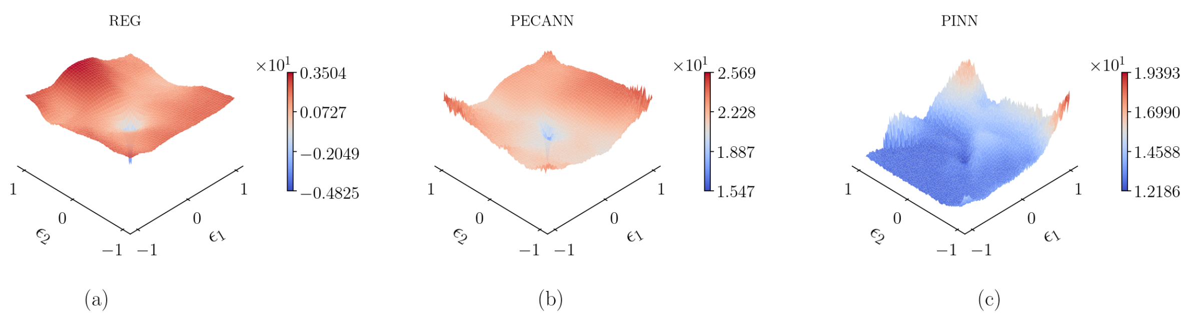

Fig. 10 Loss landscapes of neural network models trained for the solution of a two dimensional Poisson's equation: (a) purely data-driven neural network model, (b) physics-informed neural network model, (c) physics-and-equality constrained artificial neural network model

V. Conclusions

We have investigated a particular network-based approach that aggregates the residual form of the PDE of interest and its boundary conditions into a composite objective function. We demonstrated that this particular formulation can severely limit the predictive power of these networks. Our analysis on the solution of elliptic problems revealed that PINN may work for simple target solutions and but fail to converge for more complex problems. We showed that this failure mode is due to the difference of scales between the boundary and the domain losses. For further insight into how the disparity of scales impact the loss landscapes of the neural network models, we visualized the histograms of their learned parameters and their loss landscapes. We observed that embedding physics into the loss function makes learning hard and causes the parameter distributions to shift in between layers. We observed that for the PINN model, loss surfaces become increasingly complex and hard to optimize for increasingly complex target functions unlike the loss surfaces for the purely data-driven supervised model that stays fairly smooth and easy to optimize. We then demonstrated that this underlying problem can be solved by properly formulating the objective function. We showed that a constrained optimization formulation of a well-posed PDE employing the Augmented Lagrange method successfully learns the target solutions for all the cases.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. (1953204). Research was sponsored in part by the University of Pittsburgh, Center for Research Computing through the computing resources provided.

References

[1] Krizhevsky, A., Sutskever, I., and Hinton, G. E., "Imagenet classification with deep convolutional neural networks," *Adv. Neur. In.*, Vol. 25, 2012, pp. 1097–1105.

[2] He, K., Zhang, X., Ren, S., and Sun, J., "Deep Residual Learning for Image Recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778. https://doi.org/10.1109/CVPR.2016.90.

[3] Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., and Kingsbury, B., "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups," *IEEE Signal Process. Mag.*, Vol. 29, No. 6, 2012, pp. 82–97. https://doi.org/10.1109/MSP.2012.2205597.

[4] Sutskever, I., Vinyals, O., and Le, Q. V., "Sequence to Sequence Learning with Neural Networks," *CoRR*, Vol. abs/1409.3215, 2014.

[5] Weston, J., Chopra, S., and Adams, K., "#TagSpace: Semantic Embeddings from Hashtags," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, Doha, Qatar, 2014, pp. 1822–1827. https://doi.org/10.3115/v1/D14-1194.

[6] Chowdhury, G. G., "Natural language processing," *Annual review of information science and technology*, Vol. 37, No. 1, 2003, pp. 51–89.

[7] Schmidhuber, J., "Deep learning in neural networks: An overview," *Neural Networks*, Vol. 61, 2015, pp. 85–117.

[8] Hornik, K., Stinchcombe, M., and White, H., "Multilayer feedforward networks are universal approximators," *Neural Netw.*, Vol. 2, No. 5, 1989, pp. 359–366.

[9] Leshno, M., Lin, V. Y., Pinkus, A., and Schocken, S., "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function," *Neural Netw.*, Vol. 6, No. 6, 1993, pp. 861–867.

[10] Dissanayake, M. W. M. G., and Phan-Thien, N., "Neural-network-based approximations for solving partial differential equations," *Commun. Numer. Meth. Eng.*, Vol. 10, No. 3, 1994, pp. 195–201.

[11] van Milligen, B. P., Tribaldos, V., and Jiménez, J. A., "Neural Network Differential Equation and Plasma Equilibrium Solver," *Phys. Rev. Lett.*, Vol. 75, No. 20, 1995, pp. 3594–3597.

[12] Monterola, C., and Saloma, C., "Solving the nonlinear Schrodinger equation with an unsupervised neural network," *Opt. Express*, Vol. 9, No. 2, 2001, pp. 72–84.

[13] Parisi, D. R., Mariani, M. C., and Laborde, M. A., "Solving differential equations with unsupervised neural networks," *Chemical Engineering and Processing: Process Intensification*, Vol. 42, No. 8-9, 2003, pp. 715–721.

[14] Hayati, M., and Karami, B., "Feedforward Neural Network for Solving Partial Differential Equations," *J. Appl. Sci.*, Vol. 7, No. 19, 2007, pp. 2812–2817.

[15] Lagaris, I. E., Likas, A., and Fotiadis, D. I., "Artificial neural networks for solving ordinary and partial differential equations," *IEEE Trans. Neural Netw.*, Vol. 9, No. 5, 1998, pp. 987–1000.

[16] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zheng, X., "TensorFlow: A System for Large-Scale Machine Learning," *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, USENIX Association, USA, 2016, p. 265–283.

[17] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al., "Pytorch: An imperative style, high-performance deep learning library," *Advances in Neural Information Processing systems*, Vol. 32, 2019, pp. 8026–8037.

[18] Yu, B., et al., "The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems," *arXiv preprint arXiv:1710.00211*, 2017.

[19] Raissi, M., Perdikaris, P., and Karniadakis, G., "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *J. Comput. Phys.*, Vol. 378, 2019, pp. 686–707.

[20] Sirignano, J., and Spiliopoulos, K., "DGM: A deep learning algorithm for solving partial differential equations," *J. Comput. Phys.*, Vol. 375, 2018, pp. 1339–1364.

[21] Basir, S., and Senocak, I., "Physics and Equality Constrained Artificial Neural Networks: Application to Partial Differential Equations," *arXiv preprint arXiv:2109.14860*, 2021.

[22] Raissi, M., Perdikaris, P., and Karniadakis, G. E., "Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations," *arXiv preprint arXiv:1711.10566*, 2017.

[23] Powell, M. J., "A method for nonlinear constraints in minimization problems," *Optimization; Symposium of the Institute of Mathematics and Its Applications, University of Keele, England, 1968*, edited by R. Fletcher, Academic Press, London,New York, 1969, pp. 283–298.

[24] Hestenes, M. R., "Multiplier and gradient methods," *J. Optim. Theory Appl.*, Vol. 4, No. 5, 1969, pp. 303–320.

[25] Kingma, D. P., and Ba, J., "Adam: A Method for Stochastic Optimization," , 2017.

[26] Glorot, X., and Bengio, Y., "Understanding the difficulty of training deep feedforward neural networks," *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, Proceedings of Machine Learning Research, Vol. 9, edited by Y. W. Teh and M. Titterington, PMLR, Chia Laguna Resort, Sardinia, Italy, 2010, pp. 249–256.

[27] Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T., "Visualizing the loss landscape of neural nets," *arXiv preprint arXiv:1712.09913*, 2017.