PHYSIC INFORMED NEURAL NETWORKS WITH CURRICULUM LEARNING

Serge Kotchourko
Department of Computer Science
University of Stuttgart
Universitätsstraße 38, D–70569 Stuttgart
st155124@stud.uni-stuttgart.de

Amel Vatic
Department of Computer Science
University of Stuttgart
Universitätsstraße 38, D–70569 Stuttgart
st166197@stud.uni-stuttgart.de

February 2, 2024

ABSTRACT

Curriculum Learning for physics-informed neural networks (PINNs) is a training approach that aims to improve the performance of PINNs on complex physical phenomena. The method involves a systematic curriculum, reminiscent of how humans learn, where the PINN is progressively exposed to increasingly challenging scenarios, building on top of previously learned concepts. However, as in humans, noise and the number of examples can have a significant impact on the learning process. For PINNs these effects are rarely explored and most setups assume a clean room with (almost) no noise. This paper investigates the effects of introducing noise into the training step and changing sample size has on the performance of PINNs with curriculum learning. The results indicate that PINNs with curriculum learning are still able to learn the underlying primitive function of the partial differential equation (PDE) for small sample sizes and high noise, yet depended on the optimizer used the sensitivity varies vastly, highlighting that this is an important aspect to consider. Ultimately, this research contributes to the understanding of the robustness and reliability of curriculum regularization, making it a more effective tool in the realm of physics-informed machine learning.

1 Introduction

Ordinary differential equations (ODEs) and partial differential equations (PDEs) are common building blocks in numerical models and are the foundation and driver of modern scientific understanding. However, solving differential equations is not always feasible or possible, especially for complex problems. Thus, numerical methods are used to approximate the solution of the differential equation. These numerical methods are computationally expensive and require a lot of computational resources. Furthermore, the numerical methods are not always accurate and can be unstable. Physics-informed neural networks (PINNs) are a class of neural networks which respect in their output the governing laws of physics described through differential equations [1]. Such models are utilized in two distinct areas: (1) learning the underlying primitive function of a PDE and (2) discovery of PDEs and learning its parameters. Incorporating such governing laws through partial differential equations into the training of models can be achieved via a common property exhibited by such equations:

$$\mathcal{F}(u(\mathbf{x}, t)) = 0 \quad \text{for} \quad \mathbf{x} \in \mathbb{R}^n, t \in \mathbb{R} \tag{1}$$

where $\mathcal{F}$ is a differential operator, u is the solution of the PDE and $\mathbf{x}$ is the spatial coordinate. Hence, due to the form of such PDEs, it is possible to introduce the equation directly into the loss function

$$\min_{\theta} \mathcal{L}(\hat{u}, u \mid \theta) = \min_{\theta} \left(\mathcal{L}_{\text{PDE}}(\hat{u} \mid \theta) + \mathcal{L}'(\hat{u}, u \mid \theta)\right) \quad \text{with} \quad \mathcal{L}_{\text{PDE}}(\hat{u} \mid \theta) = \mathcal{F}(\hat{u}(\mathbf{x}, t)) \tag{2}$$

where $\hat{u}$ is the learned function, u is the actual function, θ are the learned weights and biases of the network, $\mathcal{L}_{\text{PDE}}$ is the loss for the PDE and $\mathcal{L}'$ is some other common loss function (e.g. mean squared error). In general, PINNs are a promising approach in different fields [2, 3, 4, 5, 6], however, in cases of complex PDEs it might fail, as Krishnapriyan, Gholami, et al. [7] showed. To improve the performance of PINNs on complex PDEs, Krishnapriyan, Gholami, et

al. introduced curriculum regularization (from here on referred to as curriculum learning). The method involves a systematic curriculum, reminiscent of how humans learn, where the PINN is progressively exposed to increasingly challenging scenarios, building on top of previously learned concepts. This approach can be thought of as a smart way of initializing the weights and biases of the network. The results of Krishnapriyan, Gholami, et al. indicate that curriculum learning is beneficial for PINNs to learn solutions to complex PDEs. Noise and sample size can have a significant impact on the performance of models and is also a topic of research for PINNs [8, 9, 10, 11, 12], yet, most studies assume noiseless data. This paper aims to investigate the effects of noise and sample size on the performance of PINNs in the context of curriculum learning, to better understand the robustness and reliability of curriculum regularization. As an artifact of this research, we provide a framework for curriculum learning for PINNs, which can be easily extended to other PINN problems [1]. Section 2 introduces PINNs using curriculum learning to improve performance on more complex PDEs, with its first goal to reproduce results of Krishnapriyan, Gholami, et al., by comparing the performance of PINNs with and without curriculum learning. The second goal is to find optimal hyperparameters for the optimizers used in the following experiments. Section 3 investigates the effects of noise and sample size on the performance of PINNs with curriculum learning and Section 4 builds on these results and investigates a slower approach on learning, by increasing the number of curriculum steps, to mitigate the negative effects of noise and sample size. Finally, Section 5 concludes the paper and recaps the results.

2 The Classroom

Krishnapriyan, Gholami, et al. [7] highlighted how using curriculum learning for physics-informed neural networks can improve the performance on both linear and non-linear ordinary differential equations and partial differential equations. In this section, we aim to reproduce the results of Krishnapriyan, Gholami, et al. by comparing the performance of PINNs with and without curriculum learning for a simple Convection-Diffusion Equation (CDE), which will be used as the basis for the following experiments. Furthermore, we aim to find optimal hyperparameter sets for the optimizers used in the following experiments. If not further specified, the evaluation of the results is based on the last curriculum step, where the convection coefficient c is 30, as our goal is to learn a complex PDE, and its overall loss, as it reflects the ability to do so.

2.1 Experimental Setup

In the following, we describe the experimental setup in this section. The aim of the experiments is to reproduce the results of Krishnapriyan, Gholami, et al. and find optimal hyperparameters for the optimizers used in the following experiments.

Convection-Diffusion Equation. The Convection-Diffusion Equation (CDE) is a PDE that describes the diffusion and convection of energy, mass, or particles in a physical space. In this experiment, we use the CDE with no diffusion term and a scalar convection term. Thus, the described CDE is also known as the linear transport equation. The described CDE is given by the partial differential equation

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0 \quad \text{for} \quad x, t, c \in \mathbb{R}_+ \tag{3}$$

where c is the convection coefficient and $u(x, t)$ is the primitive function of the CDE. The CDE is analyzed on the domain $x \in [0, 2\pi]$ and $t \in [0, 1]$ and the initial condition is given by

$$u(x, 0) = \sin(x) \quad \text{and} \quad u(0, t) = u(2\pi, t) \quad \text{for} \quad x \in [0, 2\pi] \quad \text{and} \quad t \in [0, 1] \tag{4}$$

The convection coefficient c is increased from 1 to 30, which increases the complexity of the CDE, as the convection term becomes more dominant. Furthermore, not that for the primitive function u, the PDE evaluates to zero. A deviation from this can be seen as an error, which is the key on how the CDE is used to guide the training through the loss function.

Data. The data is sampled from the domain of the CDE. The corresponding labels are the analytical solution of the CDE. The data is sampled using a random sampling strategy and the corresponding labels are augmented with Gaussian noise to achieve a certain signal-to-noise ratio (SNR). The SNR is defined as

$$\text{SNR} = 10 \log_{10} \left(\frac{\sigma^2_{\text{signal}}}{\sigma^2_{\text{noise}}} \right) \tag{5}$$

[1] The implementation and further details can be found here: `https://github.com/confusedSerge/pinn-curriculum-learning`

Optimizer	Learning Rate	Weight Decay	Momentum	Nesterov	Max Iterations	History Size
SGD	[.0001, .1]	[0, .1]	[.1, .9]	Yes, No	-	-
Adam	[.00001, .1]	[0, .1]	-	-	-	-
L-BFGS	[.01, 2]	-	-	-	5, 10, 20	25, 50, 100

Table 1: Hyperparameters sweep domain for the optimizers. Parameters in square brackets are continuous, while parameters in without brackets are categorical. If a hyperparameter is not listed, the default value of the PyTorch implementation is used.

where σ^2_{signal} and σ^2_{noise} are the variance of the signal and noise respectively. The so calculated variance of the noise is then used to sample from a Gaussian distribution with mean 0 and variance σ^2_{noise} and added to the labels. For this set of experiments, we use a signal-to-noise ration of 50 and sample 100 points from the domain consisting of 10000 equidistant points.

Model. The model is a fully connected neural network with 3 hidden layers, 50 neurons per hidden layer and the hyperbolic tangent function as the activation function. The inputs to the network are the spatial and temporal coordinates (x, t) and the output of the network is $\hat{u}(x, t)$. The model is implemented in PyTorch.

Loss Function. The loss function is the mean squared error (MSE) between the predicted and the analytical solution of the CDE and the PDE of the learned function. The loss is given by

$$\mathcal{L}(\hat{u}, u \mid \theta) = \mathcal{L}_{\text{PDE}}(\hat{u} \mid \theta) + \mathcal{L}_{\text{MSE}}(\hat{u}, u \mid \theta) \tag{6}$$

$$\mathcal{L}_{\text{PDE}}(\hat{u} \mid \theta) = \frac{\partial \hat{u}}{\partial t} + c\frac{\partial \hat{u}}{\partial x} \tag{7}$$

$$\mathcal{L}_{\text{MSE}}(\hat{u}, u \mid \theta) = \frac{1}{n}\sum_{i=1}^{n}(\hat{u}(x_i, t_i) - u(x_i, t_i))^2 \tag{8}$$

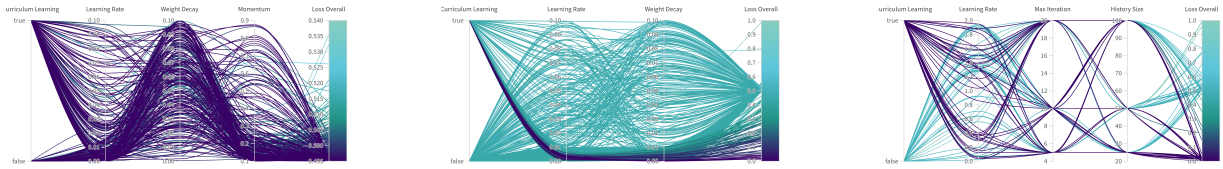
where $\hat{u}$ is the predicted value of the solution to the CDE, u is the analytical solution of the CDE, θ are the learned weights and biases of the network, $\mathcal{L}_{\text{PDE}}$ is the loss function given by the CDE and $\mathcal{L}_{\text{MSE}}$ is the MSE loss.

Optimization. For the initial experiments, we use three different optimizers; Stochastic Gradient Descent (SGD), Adam and Limited-memory Broyden–Fletcher–Goldfarb–Shannon (L-BFGS). SGD is used, as it is a very common and known optimizer. Adam is the natural extension of SGD and tends to perform better in comparison. Furthermore, Adam is not as resource intensive as L-BFGS. L-BFGS is a quasi-Newton method, known to perform better on PINN problems. In preliminary experiments, we found that L-BFGS performs better with the Strong-Wolfe line search method, which is why we use it in our experiments. The hyperparameters for the optimizers are found using sweeps with Bayesian optimization on the hyperparameter search space. The hyperparameter sweep domains are listed in Table 1. If not further specified, the default PyTorch values are used for the hyperparameter of the respective optimizer.

Curriculum Learning. The curriculum learning approach is implemented as described by Krishnapriyan, Gholami, et al. . Between each curriculum step, the convection coefficient c is increased by 1, starting at 1 and ending at 30. Each curriculum step, the model is trained for 250 epochs on the data. Furthermore, we include models without curriculum learning, where the convection coefficient c is fixed at 30 and trained for 250 epochs, referred to as baseline. The baseline is used to compare PINNs with and without curriculum learning.

2.2 Discussion

The results of the hyperparameter sweep are presented in Figure 1. The hyperparameters are color coded, where the color represents the overall loss $\mathcal{L}(\hat{u}, u \mid \theta)$, where the convection coefficient c is 30. We also include markings if the corresponding hyperparameter combination was used in curriculum learning or in the baseline model, to illustrate the effect of curriculum learning.



(a) SGD (b) Adam (c) L-BFGS

Figure 1: Hyperparameter sweep result for the different optimizers. The hyperparameters are color coded, where the color represents the overall loss $\mathcal{L}(\hat{u}, u \mid \theta)$, as it reflects the ability of the model to learn the underlying function to the complex CDE. The hyperparameters used during the sweep are listed in Table 2.

Reproducing Results Figure 1 shows that SGD performs poorly on the CDE, with the overall loss $\mathcal{L}(\hat{u}, u \mid \theta)$ being around 0.5. Furthermore, SGD does not benefit from curriculum learning, as the performance with curriculum learning is on par with the baseline model. Adam and L-BFGS baseline models also tend to perform poorly, as Figure 1b and Figure 1c demonstrate, but clearly benefit from using the curriculum learning approach, as their loss drop as low as .01 and .0005 respectively. This indicates that curriculum learning is beneficial for PINNs to learn the underlying primitive function of (more complex) PDEs, which is on par with the results of Krishnapriyan, Gholami, et al. [7].

Optimal Hyperparameters Figure 1a illustrates, that SGD is an ineffective optimizer for PINNs, as the overall loss $\mathcal{L}(\hat{u}, u \mid \theta)$ is around 0.5 for most hyperparameter combinations, with the exception of a few combinations. This again is in line with the results of Krishnapriyan, Gholami, et al., where they argue that SGD is not ideal for PINNs. Therefore, we will not further consider SGD in the experiments to follow. Adam benefits from a very low learning rate and a very low weight decay, as Figure 1b shows. This is in line with the results of Krishnapriyan, Gholami, et al., where they argue that the loss landscape is very complex, hence a small learning rate can be beneficial to not overstep into a worse local minimum. L-BFGS on the other hand exhibits two pairs of hyperparameters, which perform well, as Figure 1c illustrates. The first pair is a high learning rate and a low history size, while the second pair is a low learning rate and a high history size, while the maximum iterations do not seem to have a significant effect on the performance of L-BFGS. This might be due to the internal optimization of L-BFGS, where for higher learning rates bad steps quickly leave the history and do not affect the optimization for longer periods, while for smaller learning rates the impact of a bad step is not as pronounced.

3 A Crowded and Noisy Classroom

As discussed in the previous Section 2, PINNs with curriculum learning perform better than PINNs without. Note that the experiments in the previous section were done with a sample size of 100 and a SNR of 50. This is an idealized experiment scenario, not reflecting the real world, as noise can be introduced in a variety of ways, for example through experimental setups or trough approximate approaches, yet reflecting the choice of data in this research space. Therefore, the goal of this section is to investigate the effects of sample size and noise in the training data on the performance of PINNs with curriculum learning. For this set of experiments, we use the optimal hyperparameters found in the previous Section 2.2 and vary the sample size and noise in the training data. Intuitively, we expect the performance of PINNs with curriculum learning to decreases with decreasing sample size and increasing noise in the training data, as the co-domain ceases to represent the underlying function sufficiently. A visualization of the effect on the co-domain can be found in Appendix A.

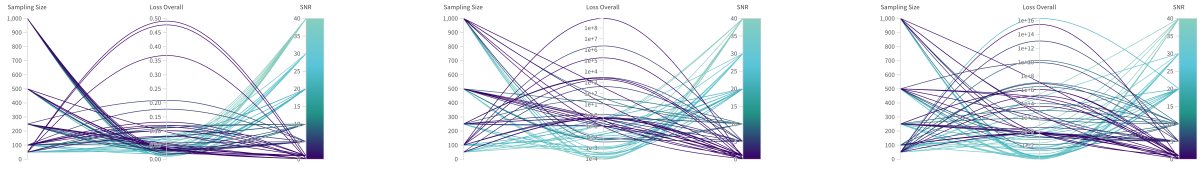
3.1 Experimental Setup

For this set of experiments, the Convection-Diffusion Equation, model, loss function and curriculum learning approach are identical to the previous Section 2.1.

Optimization. For this set of experiments, we use the optimal hyperparameters found in the previous Section 2.2. The hyperparameters are listed in Table 2. Furthermore, in the case of L-BFGS, we use two different hyperparameter sets, one with a high learning rate and a low history size and one with a low learning rate and a high history size. These sets are results from observations made in the previous Section 2.2, where we found that these hyperparameter combinations performed well. For readability, we will refer to the hyperparameter sets as Adam, L-BFGS$_{\text{high}}$ and L-BFGS$_{\text{low}}$.

Optimizer	Learning Rate	Weight Decay	Max Iterations	History Size
Adam	.0025	.0005	-	-
L-BFGS$_{\text{high}}$	1.5	-	10	25
L-BFGS$_{\text{low}}$	.15	-	10	100

Table 2: Hyperparameters found during the hyperparameter search, performing well on first set of experiments. These hyperparameters are also the basis for following experiments.



(a) Adam (b) L-BFGS$_{\text{high}}$ (c) L-BFGS$_{\text{low}}$

Figure 2: The overall loss of the last curriculum step for varying degrees of sample size and noise, with 30 curriculum learning steps. The loss is located in the middle, and for cases of L-BFGS$_{\text{high}}$ and L-BFGS$_{\text{low}}$, the scale is logarithmic.

Sample Size. The samples are randomly sampled from the domain consisting of 10000 equidistant points. The sample size is varied from 10, 50, 100, 500 and 1000 samples.

Noise. The signal-to-noise ratio (SNR) is varied from 0.1, 1, 5, 10, 20, 30 and 40. The noise is added to the labels as described in the previous Section 2.1.

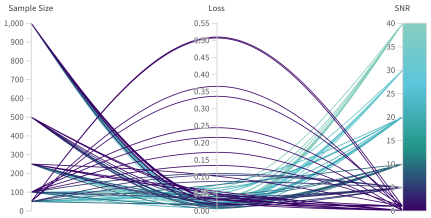
3.2 Discussion

The results of the experiments are presented in Figure 2. Overall, the results indicate that increasing the noise in the sample set increases the loss, as does decreasing the sample size. Furthermore, decreasing the sample size increases the impact of noise in the data on the achieved loss, meaning that noise contributes significantly more to the loss.

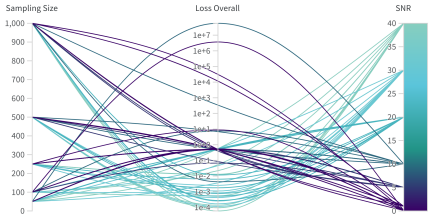
Adam. Figure 2a illustrates how models trained with the Adam optimizers are still able to learn the underlying primitive function even for very small and noisy sample sizes, indicating the robustness of Adam. Increasing the signal-to-noise ratio does not further significantly lower the overall loss, yet increasing sample size does decrease loss and for 1000 sample points completely negates the effect of noise with SNR of .1. In general, for Adam, sample size does not need to be increased significantly for noisy data to achieve similar performances as in the idealized experiment. However, the loss does increase for noisy samples and interferes with the learning of underlying primitive function. Finally, the low learning rate and weight decay seems to help in navigating the even more complex loss landscape.

L-BFGS. In contrast, neither L-BFGS hyperparameter sets (Figure 2b and 2c) are able to achieve comparable performance to the previous Section 2.2 for lower signal-to-noise ratios, in some cases failing completely. For increasing values, this slowly creeps back to its original loss. Noise has a significant impact on the ability to learn the underlying function, as increasing the sampling size under higher noise does not have a significant impact on lowering the noise. Decreasing the sampling size also impacts the loss, albeit not as significant as noise. As in Adam, we are able to observe the negative impacts of lower sample sizes and higher noises on the ability to learn the solution to the CDE. However, the impact is exaggerated in the case for hyperparameter set L-BGFS$_{\text{high}}$ and even more so for L-BGFS$_{\text{low}}$, implying that L-BFGS is less robust against these effects and is not able to reconstruct the co-domain sufficiently.

Overall, the results indicate that PINNs with curriculum learning are still able to learn the underlying solution and depending on the optimizer used, sample size should be weighted against noise in the training data. Furthermore, the results indicate that increasing the sample size can be an approach in mitigating the negative effects, however noise is the driving factor in decrease of performance and should be closely monitored.



(a) Adam



(b) L-BFGS$_{\text{high}}$

Figure 3: Loss for varying degrees of sample size and noise and 60 curriculum learning steps. Per curriculum step, the convection coefficient c is increased by 0.5, up to 30. The overall loss $\mathcal{L}(\hat{u}, u \mid \theta)$ is the loss of the last curriculum step, where the convection coefficient c is 30 and is located in middle.

4 A Slower Approach on Learning the Subject

Introducing noise can have a negative effect on the performance of PINNs with curriculum learning, as discussed in the previous Section 3.2, which can be reduced to some degree by increasing the sample size. This section's goal is to investigate a different approach in reducing the negative effects of noise, by using a slower approach on learning. This slower approach is achieved by increasing the number of curriculum steps, in return reducing the increase of the convection coefficient c per curriculum step and allows the model to learn the solution more gradually.

4.1 Experimental Setup

For this set of experiments, the Convection-Diffusion Equation, Model and Loss Function are identical to the previous Section 2.1. The data is also sampled and augmented with noise in the same way as in the previous Section 3.1. For the optimizer, we only continue to use Adam and L-BFGS$_{\text{high}}$, as they performed well in the previous Section 3.

Curriculum Learning. The curriculum learning steps are increased from 30 to 60. The convection coefficient c is increased by 0.5 per curriculum step, starting at 0.5 and ending at 30. For each curriculum step, the model is trained for 250 epochs as in the previous sections.

4.2 Discussion

Figure 3 shows the results of the experiments. Neither Adam nor L-BFGS$_{\text{high}}$ benefit from the slower approach on learning, as the overall loss does not decrease, considering also that the number of curriculum steps is doubled, intrinsically doubling the number of samples seen by the models during training. This indicates the possibility that with the previous curriculum step size the models capability of learning the underlying primitive function is reached. This is also supported by our observation in preliminary experiments, where decreasing the step size had a negative impact on the ability in learning the primitive function. These results also highlight how detrimental low sampling size and high noise can be on the performance of models and also underline how sensitive L-BFGS$_{\text{high}}$ is to noise and sample size.

5 Conclusion

Curriculum learning, as introduced by Krishnapriyan, Gholami, et al. [7], is a promising approach to improve the performance of physics-informed neural networks with partial differential equations. However, as in other fields of machine learning, noise and sample size are important factors to consider when training a model, yet not often considered for PINNs. This paper investigated these effects on the performance of PINNs with curriculum learning. The results from Section 3 indicate that noise and sample size exhibit similar effects on the performance of PINNs with curriculum learning as in other fields of machine learning. Furthermore, PINNs with curriculum learning are still able to learn the underlying solution to the Convection-Diffusion Equation. However, counter-intuitively, L-BFGS methods, which tend to perform better for PINNs, were less robust under these conditions, while Adam was able to reconstruct the co-domain even under higher noise conditions. This highlights the importance of considering sparsity and noise in data, especially in combination with the optimization approach taken. Finally, in Section 4, we investigated a slower approach on learning, by increasing the number of curriculum steps, in return reducing the increase of the convection coefficient c per curriculum step. However, this approach did not mitigate the negative effects of noise, especially considering that the model was able to observe more samples during training, hinting at that curriculum step

size of one is already able to fully saturated the ability of the model to learn the underlying function. In conclusion, curriculum learning is a promising approach to improve the performance of PINNs in regards to scientific machine learning and usefulness in scientific methodologies, even under sub-optimal conditions regarding the training data.

References

[1] M. Raissi, P. Perdikaris, and G.E. Karniadakis. 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, 2019.

[2] Francisco Sahli Costabal, Yibo Yang, Paris Perdikaris, Daniel E. Hurtado, and Ellen Kuhl. Physics-informed neural networks for cardiac activation mapping. *Frontiers in Physics*, 8, 2020.

[3] Zhiping Mao, Ameya D. Jagtap, and George Em Karniadakis. Physics-informed neural networks for high-speed flows. *Computer Methods in Applied Mechanics and Engineering*, 360:112789, 2020.

[4] Shengze Cai, Zhicheng Wang, Sifan Wang, Paris Perdikaris, and George Em Karniadakis. Physics-Informed Neural Networks for Heat Transfer Problems. *Journal of Heat Transfer*, 143(6):060801, 04 2021.

[5] Zhiwei Fang and Justin Zhan. A physics-informed neural network framework for pdes on 3d surfaces: Time independent problems. *IEEE Access*, 8:26328–26335, 2020.

[6] Minglang Yin, Xiaoning Zheng, Jay D. Humphrey, and George Em Karniadakis. Non-invasive inference of thrombus material properties with physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 375:113603, 2021.

[7] Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael W Mahoney. Characterizing possible failure modes in physics-informed neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 26548–26560. Curran Associates, Inc., 2021.

[8] Luning Sun and Jian-Xun Wang. Physics-constrained bayesian neural network for fluid flow reconstruction with sparse and noisy data. *Theoretical and Applied Mechanics Letters*, 10(3):161–169, 2020.

[9] Liu Yang, Xuhui Meng, and George Em Karniadakis. B-pinns: Bayesian physics-informed neural networks for forward and inverse pde problems with noisy data. *Journal of Computational Physics*, 425:109913, 2021.

[10] Pongpisit Thanasutives, Takashi Morita, Masayuki Numao, and Ken ichi Fukui. Noise-aware physics-informed machine learning for robust pde discovery. *Machine Learning: Science and Technology*, 4(1):015009, feb 2023.

[11] Hao Xu, Junsheng Zeng, and Dongxiao Zhang. Discovery of partial differential equations from highly noisy and sparse data with physics-informed information criterion. *Research*, 6:0147, 2023.

[12] Adan Jafet Garcia Inda, Shao Ying Huang, Nevrez İmamoğlu, Ruian Qin, Tianyi Yang, Tiao Chen, Zilong Yuan, and Wenwei Yu. Physics informed neural networks (pinn) for low snr magnetic resonance electrical properties tomography (mrept). *Diagnostics*, 12(11), 2022.

A Effects of Noise and Sample Size visualized

Figure 4 demonstrates the effects of changing sampling size and noise. It illustrates how detrimental the effect low sample size and high noise is on the co-domain. Intuitively, this highlights how reconstructing the original, de-noised function is a hard process.

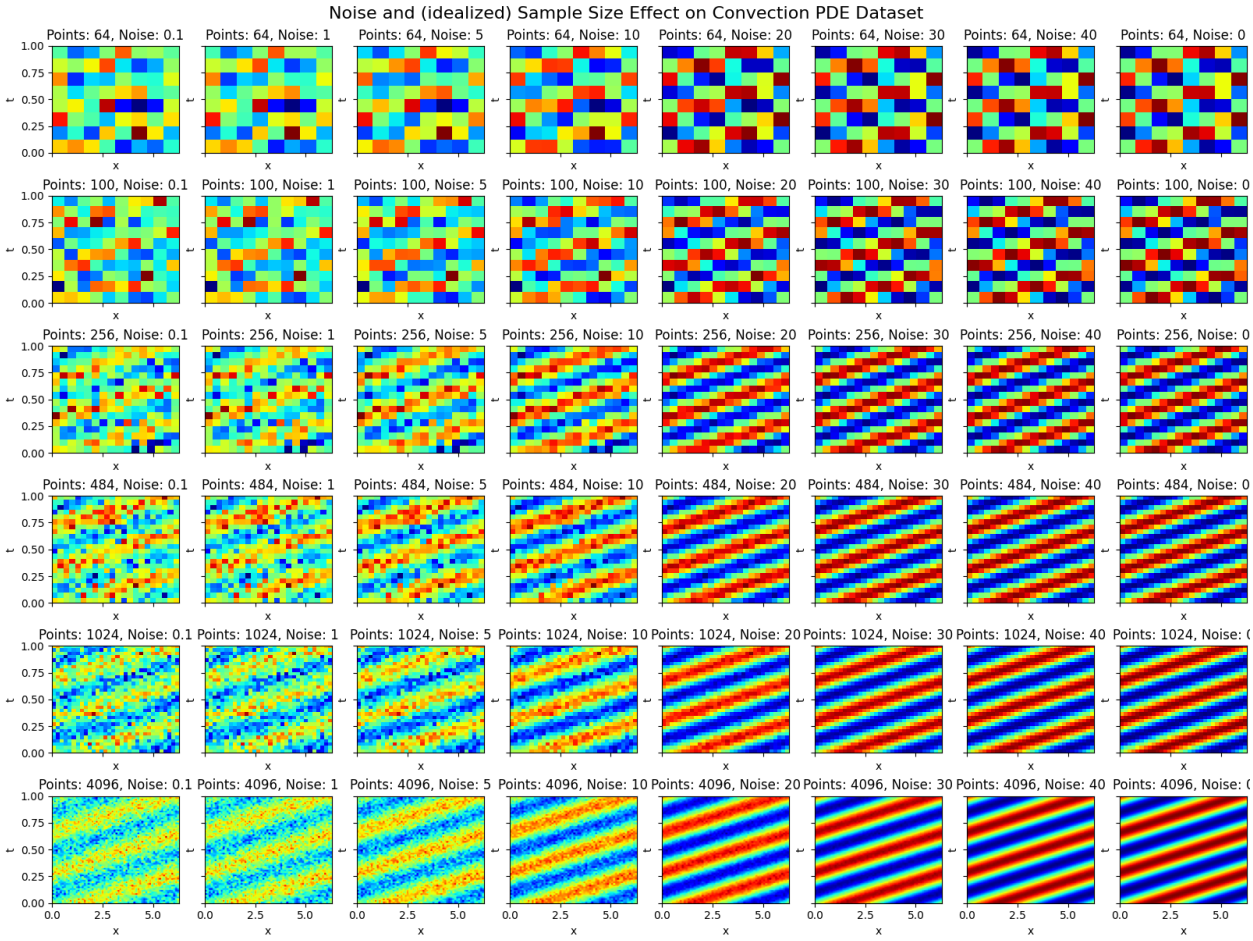


Figure 4: Effects of noise and sample size visualized, where the number of samples are increased from top to bottom and the signal-to-noise ratio is increased from left to right. The samples are equidistantly populated on the co-domain.