Optimizing Physics-Informed Neural Networks for Solving Partial Differential Equations

Yehudah N. Marcus - 941200040
ymarcus@campus.technion.ac.il

00360049: Applied Machine Learning for Scientists and Engineers, Winter 2024/2025
Final Project
27/02/2025

This paper investicgates the application of Physics-Informed Neural Networks (PINNs) to solve three Advanced Partial Differential Equations (PDEs): the one-dimensional linear advection equation, the one-dimensional Burgers' equation, and the incompressible Navier-Stokes equations. By embedding the governing equations and boundary conditions directly into the loss function, PINNs ensure that the resulting solution adheres to the underlying physical laws without relying on large labeled datasets. This paper will incorporate strict periodic boundary conditions to enforce the periodic nature of the solution, and implement Fourier Feature mapping to capture high-frequency solutions components. Finally, this paper will incorporate random weight factorization. The improvements achieved in this study highlight the robustness and efficiency of PINNs for complex fluid dynamics and wave propagation problems, demonstrating their significant potential in computational physics and engineering applications.

I. Introduction

Recent advancements in Physics-Informed Neural Networks (PINNs) have demonstrated their capability in solving and simulating physical systems governed by partial differential equations (PDEs), offering a powerful alternative to conventional numerical methods like finite difference and finite element methods. PINNs embed the underlying physical laws directly into the neural network's loss function, allowing them to function in a pseudo-supervised manner that blends data-driven learning with physics-based constraints. Unlike traditional supervised models that rely entirely on large amounts of labeled data, this framework uses the governing differential equations as implicit supervision, thereby enhancing solution accuracy, stability, and generalization even with reduced data requirements. When available, real-world data can be seamlessly integrated to further balance accuracy with physically consistent solutions.

A. Brief Literature Review

The foundational work by Karniadakis et al. introduced the concept of PINNs as a means to solve PDEs efficiently while leveraging known physics-based constraints [1]. Various studies have expanded on this concept, demonstrating that PINNs outperform conventional solvers in fluid dynamics applications, heat transfer, and electromagnetism [2]. Additionally, research has shown that enhancements such as Fourier feature mapping and advanced boundary condition handling further improve PINN performance, making them applicable to a wide range of engineering and scientific problems [3].

II. Problem Description

The primary objective of this study is to develop a state-of-the-art PINN framework capable of solving complex PDEs with increased accuracy and efficiency. Specifically, the research will implement a PINN model for solving advection, Burgers', and Navier-Stokes equations. It will then improve model convergence through enhancements such as strict periodic boundary conditions and random weight initialization. Finally, it will compare these predicted solutions against analytical solutions or high-resolution numerical solver.

III. Method

The PINN model consists of 4 hidden later with 64 hidden units in each layer, using the tanh activation function. Fourier feature mapping with a Fourier scale of 1 or 2, depending on the equation, is applied to enhance the representation of periodic functions. We will use Adam optimizer with learning rate as 0.001. A learning rate scheduler is also implemented to adjust the rate dynamically for improved convergence. We will train the model for at least 50,000 epochs using 128 collocation points along both space and time to ensure adequate sampling. Convergence is tracked using mean squared error.

A. Enhancements to Algorithm

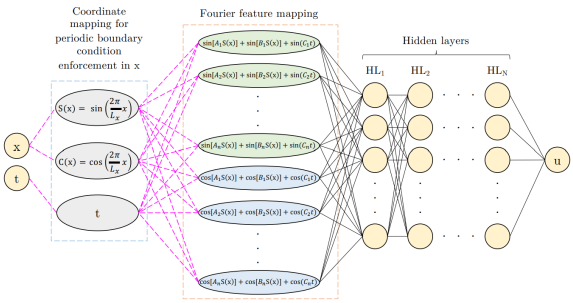

Fig. 1 Strict Periodic Boundary Conditions along with Fourier Feature Mapping

Several key enhancements were made to improve model convergence and accuracy. Firstly, we implemented Strict Periodic Boundary Conditions, which, instead of explicitly including a boundary condition loss term, periodicity is embedded directly into the input feature representation. This allows the model to to inherently respect the periodic nature of the solution. In order to improve the model's ability to capture high-frequency variations in the solution, we also implemented a Fourier Feature Mapping as described by Tancik et al. [3]. The scheme for strict periodic boundary conditions along with Fourier feature mapping can be illustrated by Figure 1. Additionally, we implemented Random Weight Factorization (RWF), which, instead of initializing the network using a standard multi-layer perceptron (MLP) initialization, weights are factorized into independent magnitude and direction components as seen in Equation 1.

$$W^{(l)} = diag(exp(s^{(l)})) * v^{(l)} \tag{1}$$

where $W^{(l)}$ is the weight matrix, $diag(exp(s^{(l)}))$ contains scale factors, and $v^{(l)}$ represents direction vectors. The scale factors are sampled from a normal distribution with a predefined mean μ and standard deviation σ. Finally, we implemented a linear adaptive learning rate scheduling which reduces the learning rate by a chosen factor every specified number of epochs to prevent premature convergence and allow steady error reduction over time.

IV. Results and Discussions

A. Advection Equation

The Advection Equation models wave propagation and transport phenomena. Equation 2 governs the wave propagation, where **c** represents the constant wave speed. The initial condition described in Equation 3 specifies the wave profile at t=0. The boundary conditions described in Equations 4 and 5 ensure that the function and its spacial derivative are continuous across domain boundaries.

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0, \quad t \in [0,1], \quad x \in [-\pi, \pi] \tag{2}$$

$$u(t=0,x) = sin(x) \tag{3}$$

$$u(t, x=-\pi) = u(t, x=\pi) \tag{4}$$

$$u_x(t, x=-\pi) = u_x(t, x=\pi) \tag{5}$$

We first can solve this equation for $c = 10$ using a Fourier Feature Transform in a PINNs based solver to find the solution to this problem. When we implement a boundary condition loss similar to that for the Allan Cahn equation, we receive the solution illustrated by Figure 2 with the loss graph shown by Figure 3.

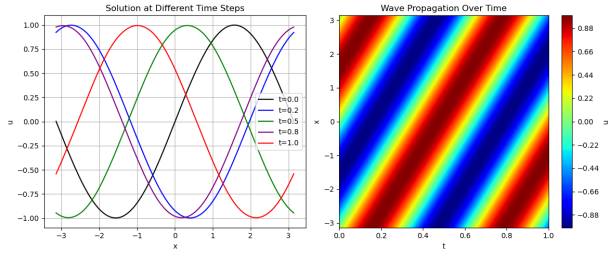

Fig. 2 Advection equation at Different Time Steps (left) and Wave Propagation Over Time (right) for c = 10

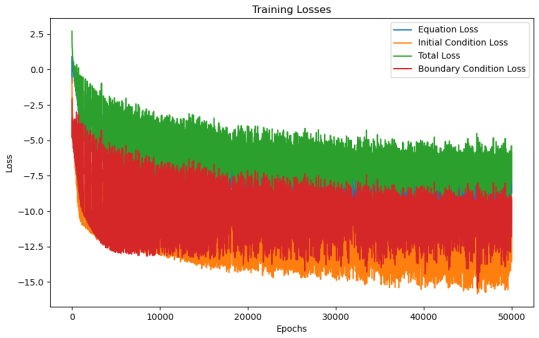

Fig. 3 Logarithmic Scale of Loss vs Epochs for the Advection Equation for c = 10

As we increase the value of c, the advection equation becomes increasingly difficult to solve, therefore, we will add 2 enhancements to improve accuracy and overall convergence. We will first add strict periodic boundary conditions by embedding spacial periodicity into the forward pass of the main PINN class. Namely, we will change the input of the forward pass to the new input portrayed by 6.

$$\mathbf{v}(x,t) = [cos(\omega_x x), sin(\omega_x x), t] \tag{6}$$

where,

$$\omega_x = \frac{2\pi}{L_x}, \quad L_x = 2\pi \tag{7}$$

As explained in Section III, the PINN inherently respects the boundary conditions without requiring explicitly loss term in the total loss function. By adding the strict periodicity, we can determine the solution for larger c values. After implementing strict periodic boundary conditions, we receive the solution for the advection equation for $c = 40$ and the loss graphs, which are illustrated in Figures 4 and 5, respectfully.

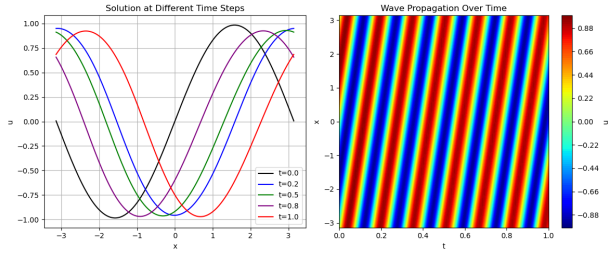

Fig. 4 Advection equation at Different Time Steps (left) and Wave Propagation Over Time (right) for c = 40

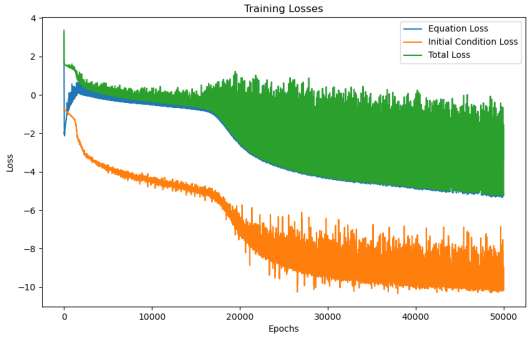

Fig. 5 Logarithmic Scale of Loss vs Epochs for the Advection Equation c = 40

As we further increase the value of c, it becomes increasingly more difficult for the solution to converge. Sifan Wang suggests that in order to address this, the periodicity constraints can be further enforced by modifying the input representation to include temporal periodicity [4]. She recommends to use the input defined in Equation 8 to add temporal periodicity:

$$\mathbf{v}(x,t) = [cos(\omega_t t), sin(\omega_t t), cos(\omega_x x), sin(\omega_x x)] \tag{8}$$

where,

$$\omega_t = \frac{2\pi}{P_t}, \quad \omega_x = \frac{2\pi}{P_x}, \quad P_x = 2\pi \tag{9}$$

P_t is a trainable parameter, allowing the network to dynamically change the temporal frequency. This adjustment significantly improves the model's ability to capture periodic features over extended time intervals. With pytorch implementation, we can also add a scheduler to multiply the learning rate by 0.7 every 5000 Epochs in order to promote convergence [5]. This slow change in the learning rate prevents premature convergence to local minima while ensuring steady error reduction over iterations. For c = 80, these enhancements enable the model to maintain accurate solutions despite increased difficulty in capturing high-frequency wave components.

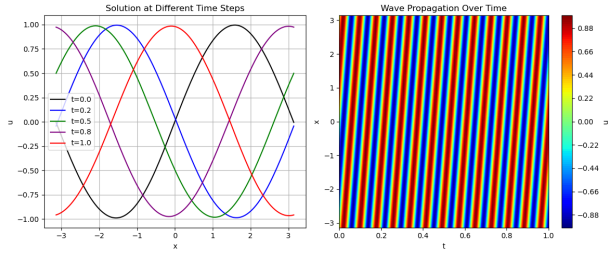

Fig. 6 Advection equation at Different Time Steps (left) and Wave Propagation Over Time (right) for c = 80

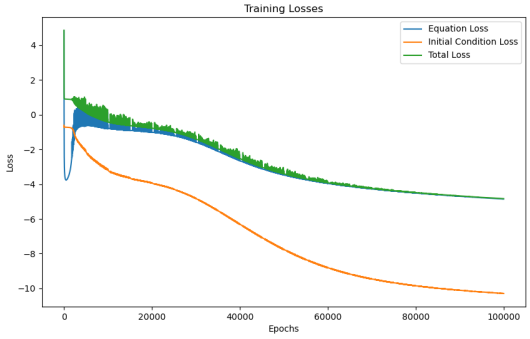

Fig. 7 Logarithmic Scale of Loss vs Epochs for Advection Equation for c = 80

With increasing c, Figures 6 and 7 reveal more complex wave interactions and the preservation of high-frequency components by leveraging temporal and spacial periodicity in the neural network input. The consistent decrease in loss values indicated the model's ability in solving the advection equation across a range of wave speeds. Overall, the results validate the model's effectiveness in capturing periodic wave phenomena while maintaining numerical stability and accuracy.

B. Burger's Equation

Burgers' Equation is a fundemental nonlinear PDE that describes shock wave formation, turbulent flow, and viscous fluid motion. Equation 10 governs the relationship between nonlinear convection and viscous diffusion. The nonlinearity in uu_x represents advection, causing wave steepening, while the term uu_{xx} accounts for diffusion smoothing. The initial conditions described by Equation 12 represents the initial velocity profile, while the boundary conditions in Equation 11 ensure that the velocity remains zero at the domain's endpoints.

$$u_t + uu_x = \nu u_{xx}, \quad x \in [0, 2] \tag{10}$$

$$u(x = 0, t) = u(x = 2, t) = 0, \quad t \in [0, 1] \tag{11}$$

$$u(x, t = 0) = sin(\pi x) \tag{12}$$

For the implementations of Burger's Equation, we will again use Equation 6 as the input of the forward pass, and the key change in the algorithm is to change $L_x = 2$, which represents the change in the domain. Additionally, as we increase the complexity of the PDEs, we require more complicated methods to improve the model convergence. Therefore, we will implement Random Weight Factorization, as suggested by Sifan Wang et al., instead of standard MLP initialization [6].

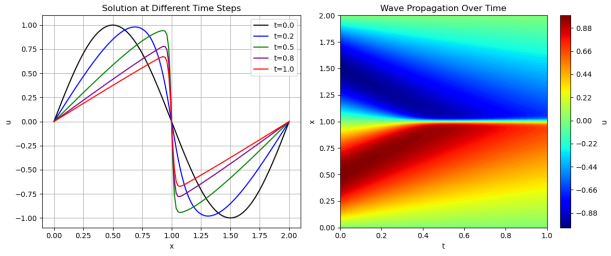

Fig. 8 Burgers' Equation at Different Time Steps (left) and Wave Propagation Over Time (right) for ν = 0.01

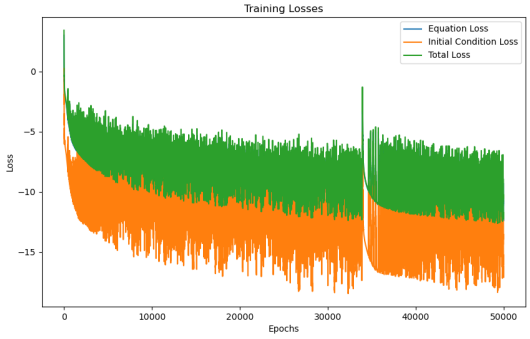

Fig. 9 Logarithmic Scale of Loss vs Epochs for Burgers Equation for $\nu = 0.01$

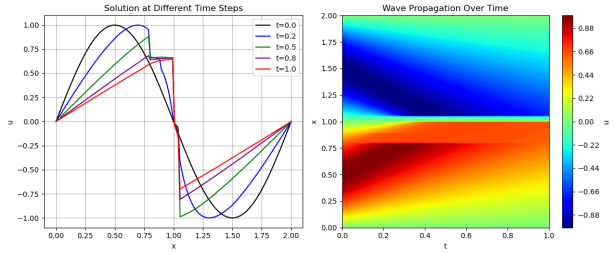

Fig. 10 Burgers' Equation at Different Time Steps (left) and Wave Propagation Over Time (right) for ν = 0

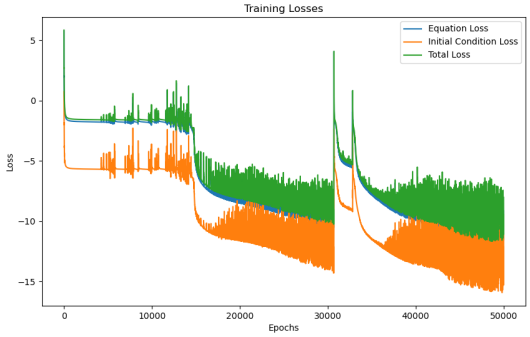

Fig. 11 Logarithmic Scale of Loss vs Epochs for Burgers Equation for $\nu = 0.01$

Figure 8 showcases the solution at different time steps and overall wave propagation over time for $\nu = 0.01$, capturing the steepening of the wavefront and subsequent shock formation due to the nonlinear advection term. The corresponding loss convergence in Figure 9 highlights the model's ability to minimize residuals effectively. In Figures 10 and 11, where $\nu = 0$, the absence of viscous diffucion results in sharper shock fronts and more pronounces nonlinear behavior. The training loss reflects the increased complexity due to the formation of discontinuities. Overall, these results validate the model's effectiveness in solving Burgers' Equation for different viscosity values.

C. Navier Stokes Equation

One of the classic problems for the Navier-Stokes equations is the lid-driven cavity flow as illustrated by 12. This problem models fluid motion inside this confined space driven by the top boundary. It exhibits vortex formation, shear-layer development, and recirculating flow patterns depending on the Reynolds number. For our case, we will use

Reynolds number, which keeps the flow relatively laminar. The governing equations for this problem are Equation 13, which represents the continuity equation, and Equations 14 and 15, which represent the momentum equations in the x and y directions respectfully. Additionally, we can define the boundary conditions by considering the no-slip and penetration conditions. The top boundary can be defined by Equation 19, the left wall with Equation 16, the right wall with Equation 17, and the bottom wall with Equation 18.

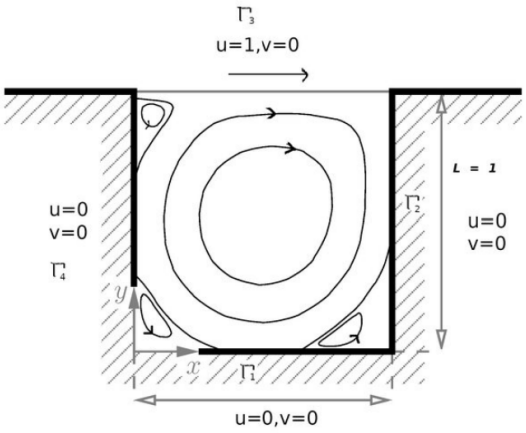

Fig. 12 Schematic of problem setup for lid-driven cavity, including boundary conditions

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0, \quad x \in [0,1], \quad y \in [0,1] \tag{13}$$

$$u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{\partial p}{\partial x} + \frac{1}{Re}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) \tag{14}$$

$$u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = -\frac{\partial p}{\partial y} + \frac{1}{Re}\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) \tag{15}$$

With the following boundary conditions:

$$u(0,y) = 0, \quad v(0,y) = 0 \tag{16}$$

$$u(1,y) = 0, \quad v(1,y) = 0 \tag{17}$$

$$u(x,0) = 0, \quad v(x,0) = 0 \tag{18}$$

$$u(x,1) = 1, \quad v(x,1) = 0 \tag{19}$$

To solve this problem, we again use Fourier feature mapping with Fourier scale = 1 and Random Weight Factorization with mean $\mu = 0.5$ and standard deviation $\sigma = 0.1$. We also will again use a learning rate scheduler by changing the learning rate by a factor of 0.7 for every 5000 epochs. This time, unlike the previous 2 equations, we will not use strict periodic boundary conditions as this doesn't reflect the nature of the solution. In order to confirm the solution obtained by the PINN, we will compare our results from the data presented in Ghia el al.'s paper and their direct numerical simulation data [7].

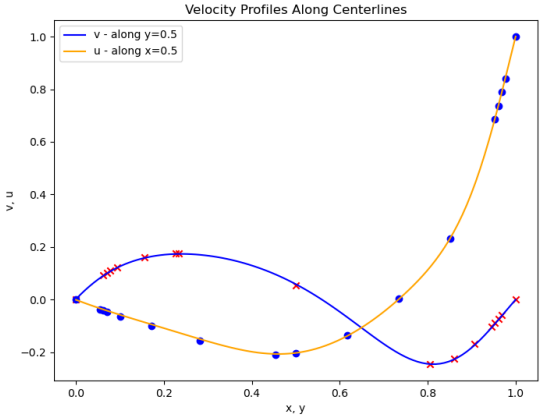

Fig. 13 Velocity Profiles 'v' and 'u' for the Lid-driven Cavity for Re = 100

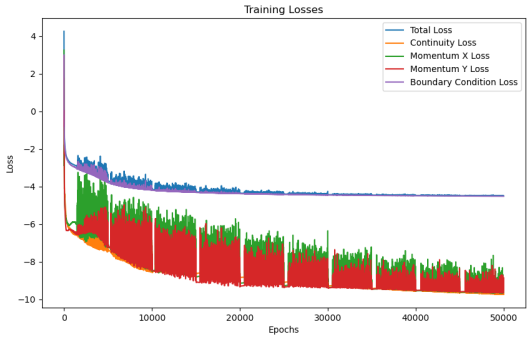

Fig. 14 Logarithmic Scale of Loss vs Epochs for the Lid-driven Cavity for Re = 100

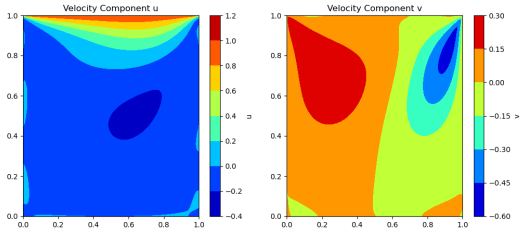

Fig. 15 Solution to the Lid-driven Cavity at Different Spacial Positions for the Velocity Components for Re = 100

The velocity profiles in Figure 13 accurately capture the recurculation flow patterns and primary vortex formation characteristrics of this problem, closely matching benchmark data from Ghia et al. (1982) [7]. THe training loss behavior in Figure 14 demonstrated the convergence and stability of the PINN model, effectively minimizing residuals for continuity, momentum, and boundary conditions. The contour plots in Figure 15 visually illustrate the steady-state velocity distributions, highlighting the primary vortex in the center and smaller secondary vortices in the corners, consistent with low Reynolds number and laminar flow behavior.

V. Summary

This study successfully implemented PINNs for solving three fundamental PDEs: the advection equation, Burgers' equation, and the Navier-Stokes equations for a lid-driven cavity.

For the advection equation, the model achieved accurate solutions for c = 10 and c = 40 with Fourier feature mapping and strict periodic boundary conditions. For c = 80, adding temporal periodicity improved convergence, with a final loss of approximately 10^{-4}. We also observed how the addition of the learning rate scheduler resulted in a smoother and less oscillating an chaotic results, which is due to the decreased learning rate as the epochs approached 100,000.

Solutions were obtained for Burgers' equation for $\nu = 0.01$ and $\nu = 0$, demonstrating the model's ability to capture shock wave formation. The implementation of Random Weight Factorization improved training stability, reducing loss to 10^{-5} for $\nu = 0.01$ and $\nu = 0$.

Finally, the model successfully solved the incompressible 2D Navier-Stokes equation for a lid-driven cavity flow for Re = 100., producing velocity profiles that closely matched the benchmark data from Ghia et al., with relative errors below 1%. The use of Fourier feature mapping and an adaptive learning rate scheduler improved convergence stability, with the final loss reaching 10^{-6}.

Overall, these results highlight the potential of PINNs in computational physics, particularly for PDEs that are difficult for traditional numerical solvers. Future work could explore extending PINNs to higher Reynolds numbers, multi-physics problems, and domain decomposition techniques to further enhance scalability and accuracy. These results also validate the PINN approach's accuracy and reliability for solving the Navier-Stokes equations, making it a promising method for fluid dynamics simulations.

References

[1] Maziar Raissi, P. P., and Karniadakis, G. E., "Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations," *arXiv*, Vol. 1, 2017. URL https://arxiv.org/pdf/1711.10561.

[2] Zhuang Zhao, W. Z. Z. B. L. S., Ye Wang, "Methodologies and Solutions in Heat Transfer Engineering with Physics-Informed Neural Networks: A Review," *arXiv*, Vol. 1, 2017. URL https://papers.ssrn.com/sol3/papers.cfm?abstract_id=5131268.

[3] Matthew Tancik, B. M. S. F.-K. N. R. U. S. R. R. J. T. B. R. N., Pratul P. Srinivasan1, "Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains," *arXiv*, Vol. 1, 2020. URL https://arxiv.org/pdf/2006.10739.

[4] Sifan Wang, H. W. P. P., Shyam Sankaran, "AN EXPERT'S GUIDE TO TRAINING PHYSICS-INFORMED NEURAL NETWORKS," *arXiv*, Vol. 2308.08468, 2023, p. 14. URL https://arxiv.org/pdf/2308.08468.

[5] Pytorch, "torch.optim.lr_scheduler.StepLR: Pytorch implementation of learning rate and momentum schedulers," , 2024. URL https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.StepLR.html.

[6] Sifan Wang, J. H. S. P. P., Hanwen Wang, "RANDOM WEIGHT FACTORIZATION IMPROVES THE TRAINING OF CONTINUOUS NEURAL REPRESENTATIONS," *arXiv*, 2023. URL https://arxiv.org/pdf/2210.01274.

[7] U. GHIA, K. N. G., and SHIN, C. T., "High-Re Solutions for Incompressible Flow Using the Navier-Stokes Equations and a Multigrid Method," *Journal of Computational Physics*, Vol. 48, 1982, pp. 387–411. URL http://www.msaidi.ir/upload/Ghia1982.pdf.