Flow Domain Parameterization and Training of Generalized Physics-Informed Neural Networks for Solving Navier-Stokes Equations

Anonymous authors
Paper under double-blind review

Abstract

Physics-informed neural networks (PINNs) have shown promise in solving the Navier-Stokes equations for fluid flow problems, but most existing approaches require retraining for each new flow case, limiting their applicability to a wide range of scenarios. This study addresses the challenge of multi-dimensional parameterization of flow domain geometries, which has not been extensively explored in previous research. We propose an approach for parameterizing the flow domain for solving the stationary Navier-Stokes equations for Newtonian fluid flow using PINNs. The proposed approach allows scaling PINN for new cases not considered in training and significantly reduces computational costs in comparison with the numerical solution.

1 Introduction

Most existing approaches to solving the Navier-Stokes equations using physics-informed neural networks (PINN) involve retraining for each new flow case. However, the configuration of the neural network allows for additional inputs to describe physical parameters, boundary conditions, and the geometry of the flow domain. Naderibeni et al. (2024) considered parameterization of the Reynolds number for flow past a cylinder. Liu et al. (2024) considered the three-dimensional flow around an arbitrarily rotating sphere subjected to a cross-flow. The authors proposed an approach for parameterizing the Reynolds number and spanwise angular velocity components. However, in these researches, the geometry of the flow domain does not change. Gao et al. (2021) applied an elliptic coordinate mapping to use convolutional neural networks (CNN) in solving partial differential equations (PDEs) on irregular parametrized flow domains. The authors note the high accuracy of the approach, but CNN allows obtaining a solution at a fixed set of points and requires the use of numerical methods for calculating derivatives. Fully connected artificial neural networks (FCNN) allow obtaining values of unknown functions at any point in the computational domain and natively support the calculation of derivatives. Heger et al. (2024) considered flow domains parameterized by a single parameter, such as a T-junction flow with a change in the height of the left junction. The analysis of existing works indicates that the possibility of multi-dimensional parameterization of flow domain geometries has not been previously explored. However, for complex flow domains, the geometry should be described by multiple parameters. Thus, the current study considers the parameterization of the flow domain geometry using several parameters and evaluates the accuracy of interpolation and extrapolation using the example of a two-dimensional flow with stenosis and aneurysm.

2 Methodology

Raissi et al. (2019) proposed a concept of PINN, where the PDE describing the physical system is integrated into the loss function of the artificial neural network. Thus, minimizing this loss function allows the artificial neural networks (ANN's) output to approximate the PDE solution. The peculiarity of the approach is that only the PDE, initial, and boundary conditions are used to obtain the solution without the need for other data. To parameterize the geometry of the flow domain, the ANN's input can be extended with a set of parameters to describe it. Then, in general form, the parameterized PINN considers the solution of the following system of equations:

$$\mathcal{N}[\mathbf{u}] = f_u(x,t,a,b,c),\ x \in \Omega, \tag{1}$$
$$\mathcal{B}[\mathbf{u}] = f_{bc}(x,t,a,b,c), x \in S, \tag{2}$$
$$\mathcal{I}[\mathbf{u}] = f_{ic}(x,t,a,b,c), t = 0, x \in \Omega, \tag{3}$$

where $\mathcal{N}[\cdot]$ is a nonlinear differential operator applied to the PDE solution $\mathbf{u}$, $\mathcal{B}[\cdot]$ defines the boundary conditions, $\mathcal{I}[\cdot]$ defines the initial conditions, x are the coordinates of the points in the computational domain, t is the time, a are the geometric parameters of the computational domain, b are the initial and boundary condition parameters, c are the physical parameters (e.g., viscosity).

The application of PINN involves approximating the solution $\mathbf{u}$ using the ANN $\mathbf{u}_{ann}(x, t,\ a,\ b, c, \Theta)$, where Θ represents the trainable parameters of the ANN. The overall loss function, based on the mean squared error (MSE), takes the following form:

$$L = \omega_u L_u + \omega_{bc} L_{bc} + \omega_{ic} L_{ic}, \tag{4}$$
$$L_u = \frac{1}{N_u}\sum_{i=1}^{N_u}\left(\mathcal{N}[\mathbf{u_i}] - f_u\right)^2, \tag{5}$$
$$L_{bc} = \frac{1}{N_{bc}}\sum_{i=1}^{N_{bc}}\left(\mathcal{B}[\mathbf{u_i}] - f_{bc}\right)^2, \tag{6}$$
$$L_{ic} = \frac{1}{N_{ic}}\sum_{i=1}^{N_{ic}}\left(\mathcal{I}[\mathbf{u_i}] - f_{ic}\right)^2, \tag{7}$$

where L_u represents the loss function based on the PDE residuals evaluated at $N_{\mathbf{u}}$ randomly distributed points within the geometry, L_{bc} is the loss function for the boundary conditions evaluated at N_{bc} randomly distributed points on the geometry's boundary, L_{ic} is the loss function for the initial conditions evaluated at N_{ic} randomly distributed points at $t = 0$ and ω_u, ω_{bc} and ω_{ic} are the weighting coefficients for the loss functions.

For modeling stationary fluid flow, the Navier-Stokes equations are solved without considering time. The fluid flow in the domain Ω with surface S is considered. The boundary conditions consist of three parts: no-slip conditions on the walls, a parabolic velocity profile at the inlet, and zero pressure at the outlet. The Navier-Stokes equations are used in the following form [6]:

$$\rho(\mathbf{v} \otimes \nabla) \cdot\ \mathbf{v} - \mu\nabla^2\mathbf{v} + \nabla p = 0, \tag{8}$$
$$\nabla \cdot \mathbf{v} = 0, \tag{9}$$

where $\mathbf{v} = [v_i]$ and p are the velocity and pressure distributions, μ is the dynamic viscosity coefficient, ρ is the fluid density, $\rho(\mathbf{v} \otimes \nabla) \cdot\ \mathbf{v}$ is the convective term, which allows taking into account the nonlinear properties, $\otimes$ is the tensor product operation. The Navier-Stokes equations can be generalized as follows:

$$\mathcal{N}[\mathbf{v}, p] = 0, x \in \Omega. \tag{10}$$

Separate loss functions for the Navier-Stokes equations, including the incompressibility condition, no-slip condition ($\mathbf{v} = 0$ at the walls) and boundary condition ($p = 0$ at the outlet, $\mathbf{v} = \mathbf{v}^{in}$ at the inlet) are also included in the overall loss function. $\mathbf{v}_\mathbf{i}^{in}$ is the specified parabolic velocity field at the inlet. The overall loss function will take the following form:

$$L = \omega_u L_u + \omega_v L_v + \omega_p L_p + \omega_{ns} L_{ns}, \tag{11}$$

where L_v is the loss function for the velocity field evaluated at N_v randomly distributed points on the inlet, L_p is the loss function for the pressure evaluated at N_p randomly distributed points on the outlet, L_{ns} is the loss function for the no-slip condition evaluated at N_{ns} randomly distributed points on the walls.

To implement the parameterization of the flow domain, a two-dimensional flow in idealized vessels with stenosis and aneurysm is considered. Figure 1 shows the drawing of the flow domain parameterized by parameters h and l. The dimensions are given in millimeters. The parameter h indicates the size of the stenosis and aneurysm in the vessel, while the parameter b indicates the position of the region with stenosis and aneurysm in the longitudinal direction. For flows with stenosis, the parameter h takes negative values. The radius of the rounding is determined as $R = 1/|h|$.

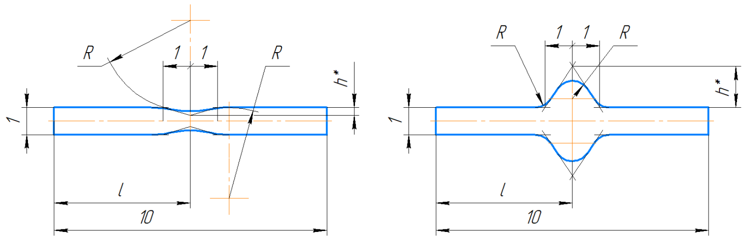

Figure 1: Scheme of the flow regions under consideration, described by the parameters h and b.

To train the parameterized PINN in this work, an FCNN will be used. The SiLU activation function demonstrates high performance for PINN (Safwan et al., 2021). For the considered example, the input data includes the coordinates and the geometric parameters of the flow domain: $\mathbf{x} = [x_1, x_2, h, b] \in \mathbb{R}^4$. The ANN predicts the distributions of the unknown functions $\mathbf{v}$ and p. The application of PINN involves the necessity of computing partial derivatives of the unknown functions. For this purpose, automatic differentiation based on the backpropagation method (Baydin et al., 2015) is used. This method allows for the computation of all necessary components of (Eq. 8, 9). To ensure stability and speed of ANN training, a procedure for normalizing input and output data is applied.

3 Results and Discussion

For training, 18 combinations of flow domains with the following parameters were used with $h = \{-0.45, -0.3, -0.15, 0.25, 0.5, 1\}$ and $l = \{3, 5, 7\}$. To test the model's generalization, 3 interpolation tests ($h = \{-0.35, 0, 0.35\}$, $\{l = [4, 5, 6\}$) and 2 extrapolation tests ($h = \{-0.4999, 1.5\}$, $l = \{2, 8\}$) were conducted. The value $h = 0$ corresponds to a flow domain without distortions. The total number of random points for each flow domain was approximately 1e6.

The flow of Newtonian fluid is considered. A parabolic velocity profile with $v_{max} = 0.3$ m/s is set at the inlet. The viscosity is taken as $\mu = 0.00385$ Pa·s, and the density as $\rho = 1050$ kg/m³. Separate ANNs were used to predict the unknown functions v_i^{ann}, p^{ann}. The trained ANNs included 10 layers with 32 neurons each. The Adam optimizer (Kingma & Ba, 2017) was used for training. The learning rate was set to 0.001. An exponential decay of the learning rate with $\gamma = 0.99$ every 150 epochs was used during training. The number of epochs was 100000. The weighting coefficients were as follows: $w_{res} = 0.2$, $w_{div(v)} = 0.2$, $w_{ns} = 10000$, $w_v = 1000$, $w_p = 1$. The batch size was 3 examples, each including: 500 points inside the computational domain, 250 points on the walls, 100 points at the inlet and outlet. Batches were formed by sampling random points.

Training and testing were performed on a workstation with a GPU 4060 ti, 32 GB RAM, and an CPU Intel i5-12400F. To assess accuracy, numerical solutions using the FEniCS library were used as reference values. The mean absolute error (mae) and relative percent error for the fields v_{abs}, p was evaluated. Table 1 presents the error values for all interpolation and extrapolation tests, as well as for several examples from the training set. Figure 2 shows the visualization of predictions for the interpolation test at $h = 0.35$, $l = 6$ compared to CFD.

The inference time of the trained ANN for 70k points of the computational domain takes 0.1 s on cpu and 0.01 s on gpu, while the numerical solution takes 10 s. The results show a slight difference in metrics for interpolation compared to the training set. The extrapolation tests show a somewhat larger error, but high extrapolation error is a common occurrence for FCNN. However, the extrapolation tests show a result sufficiently close to CFD, which may allow for quick fine tune of PINN on a wider range of parameters.

Table 1: Parameterized PINN error compared to CFD.

l, mm	h, mm	$mae_{v_{abs}}$, cm/s	$mean\,(v_{abs})$ error, %	$max\,(v_{abs})$, cm/s	mae_p Pa	$mean\,(p)$ error, %	$max\,(p)$, Pa
				Training			
3	-0.3	0.643	2.589	41.1	1.346	3.470	97.3
5	-0.15	0.291	1.068	34.0	0.727	1.941	82.4
7	0.5	0.191	0.179	29.9	1.350	4.128	66.8
				Interpolation			
4	-0.35	0.601	2.031	44.3	1.391	2.831	106.4
5	0	0.146	0.601	29.9	0.475	1.181	74.0
6	0.35	0.167	0.055	29.9	1.031	2.985	68.0
				Extrapolation			
2	-0.4999	1.305	4.716	60.0	4.097	8.655	163.6
8	1.5	0.470	2.306	29.9	0.984	0.139	65.1

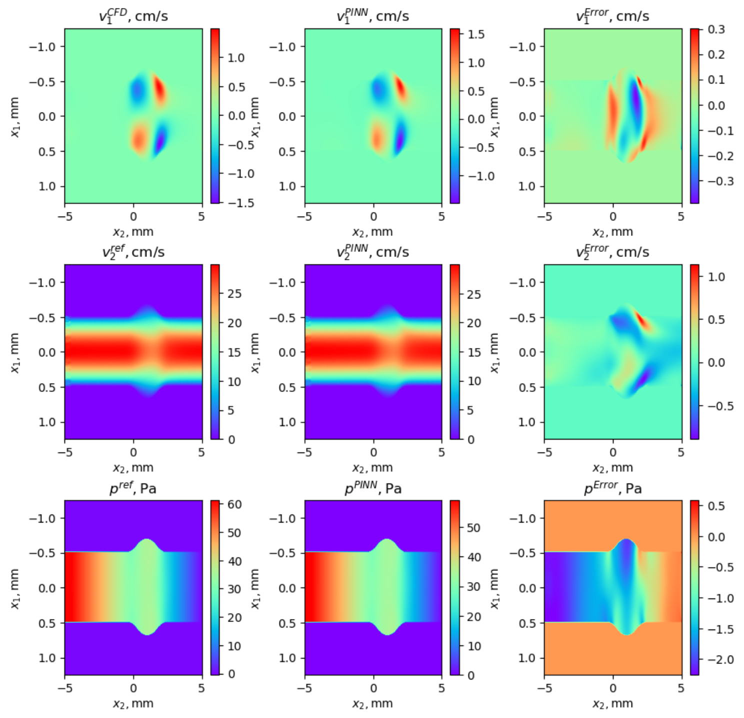

Figure 2: Comparison of parameterized PINN with CFD for $h = 0.35$ and $l = 6$.

This work presents an approach for parameterizing the flow domain for solving the stationary Navier-Stokes equations for Newtonian fluid flow using PINN. The proposed approach allows scaling PINN for new cases not considered in training and significantly reduces computational costs in comparison with the numerical solution. Deep learning methods can be used to describe the geometry of an arbitrary flow domain by transforming it into a set of features, which will enable the application of the method to arbitrary geometries of the flow domain, including realistic flows in blood vessels. Further development of flow domain parameterization may include joint parameterization of boundary conditions, viscosity, and can be used for a wider range of flow domains.

References

Atilim Gunes Baydin, Barak A. Pearlmutter, and Alexey Andreyevich Radul. Automatic differentiation in machine learning: a survey. *CoRR*, abs/1502.05767, 2015. URL `http://arxiv.org/abs/1502.05767`.

Han Gao, Luning Sun, and Jian-Xun Wang. Phygeonet: Physics-informed geometry-adaptive convolutional neural networks for solving parameterized steady-state pdes on irregular domain. *Journal of Computational Physics*, 428:110079, March 2021. ISSN 0021-9991. doi: 10.1016/j.jcp.2020.110079. URL `http://dx.doi.org/10.1016/j.jcp.2020.110079`.

Philip Heger, Daniel Hilger, Markus Full, and Norbert Hosters. Investigation of physics-informed deep learning for the prediction of parametric, three-dimensional flow based on boundary data. *Computers Fluids*, 278:106302, 2024. ISSN 0045-7930. doi: https://doi.org/10.1016/j.compfluid.2024.106302. URL `https://www.sciencedirect.com/science/article/pii/S0045793024001348`.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL `https://arxiv.org/abs/1412.6980`.

Kai Liu, Kun Luo, Yuzhou Cheng, Anxiong Liu, Haochen Li, Jianren Fan, and S. Balachandar. Parameterized physics-informed neural networks (p-pinns) solution of uniform flow over an arbitrarily spinning spherical particle. *International Journal of Multiphase Flow*, 180:104937, 2024. ISSN 0301-9322. doi: https://doi.org/10.1016/j.ijmultiphaseflow.2024.104937. URL `https://www.sciencedirect.com/science/article/pii/S0301932224002143`.

M. Naderibeni, M. J. T. Reinders, L. Wu, and D. M. J. Tax. Learning solutions of parametric navier-stokes with physics-informed neural networks, 2024. URL `https://arxiv.org/abs/2402.03153`.

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. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2018.10.045. URL `https://www.sciencedirect.com/science/article/pii/S0021999118307125`.

A. Safwan, Chao Song, and Umair Bin Waheed. Is it time to swish? comparing activation functions in solving the helmholtz equation using pinns. pp. 1–5, 01 2021. doi: 10.3997/2214-4609.202113254.

A Appendix

Figure 3 shows an example of random points for $h = 1, b = 5$.

Figure 4 shows the training process of parametrized PINN.

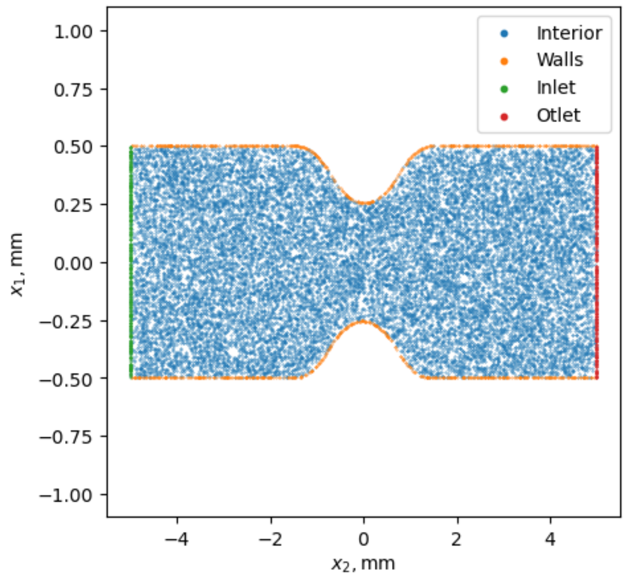

Figure 3: Visualization of randomly distributed points in the computational domain for $h = 1$, $l = 5$. Interior – points inside the flow region; Walls – dots on wall surfaces; Inlet – points on the input surface; Outlet – points on the output surface.

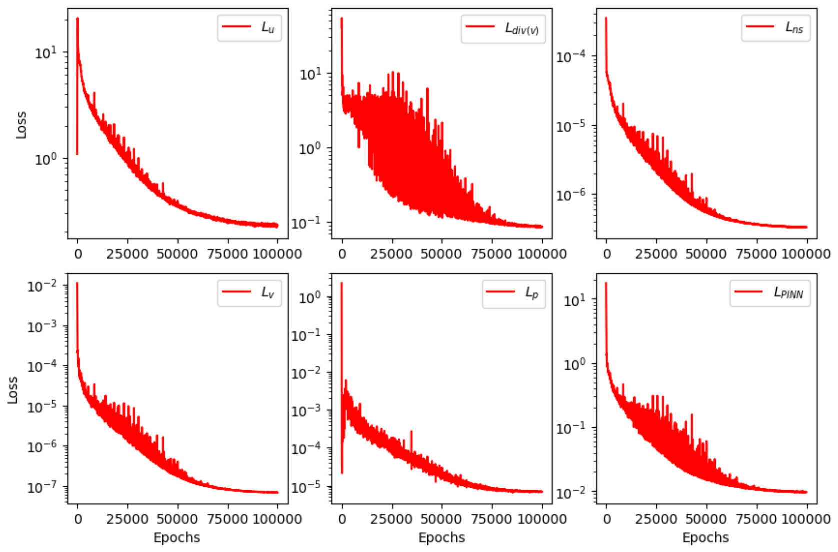

Figure 4: Training process of the parameterized PINN.