Physics-informed machine learning with domain decomposition and global dynamics for three-dimensional intersecting flows

Leslie K. Hwang *
School of Electrical, Computer and Energy Engineering
Arizona State University
Tempe, AZ 85287
lkhwang@asu.edu

Abstract

Physics-informed neural networks (PINNs) have emerged as a promising framework to develop complex scientific surrogate models, yet their scalability and accuracy often degrade in non-canonical geometries, such as non-rectangular domains or three-dimensional (3D) domains with high aspect ratios. These limitations hinder the broader adoption of vanilla PINNs in real-world, practical systems. In this work, we introduce a multi-domain PINN (MDPINN) framework designed to address the scalability and generalization challenges inherent in 3D non-rectangular domains governed by nonlinear fluid dynamics. The target domain consists of intersecting 3D fluid channels with a high aspect ratio, inducing complex flow features such as deflections, mixing, and recirculations. Our approach is grounded in two key innovations: 1) domain decomposition, which partitions the channel volumes into multiple cubic-like subdomains, each modeled by an individual PINN, 2) enforcement of global dynamics (MDPINN-GD), which ensures that the total mass flow rate entering the domain equals that exiting. These innovations reduce the complexity of the problem imposed on individual PINNs and guide effective network optimization toward physically consistent solutions throughout the domain. We demonstrate that our method achieves: 1) 74.8% accuracy improvement over a single-network PINN, and 2) 52.9% accuracy improvement over MDPINN that do not enforce global mass conservation. Furthermore, the MDPINN-GD framework exhibits accurate prediction even in highly complex regions-such as the channel intersecting zone and the outlet zone characterized by intense flow mixing and large velocity gradients-achieving maximum normalized mean absolute errors below 14.9% for velocity predictions compared to simulation results. This work establishes a path towards scalable, physically grounded surrogate modeling approach that is extensible to multiphysics and high-dimensional scientific problems.

1 Introduction

Recent advancements in scientific machine learning have garnered much attention for their potential to bridge modern machine learning methods with traditional scientific computing. This integration is particularly appealing to the engineering design community, where conventional simulation methods such as computational fluid dynamics (CFD) can be computationally expensive for their applications. For instance, CFD-based design optimization of fluidic networks-such as microchannels used for electronic cooling-often involves simulations across a wide range of geometric and physical parameters. Major bottlenecks in these simulations lie in the generation and refinement of computational meshes for large geometric variations, as well as in the computational cost of iterative design searches.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

To address the challenges in engineering designs, physics-informed neural networks (PINNs) have emerged as a viable surrogate modeling approach [1–9].

PINNs are a class of neural networks whose solutions are guided by embedded physical laws rather than being solely determined by data. Unlike traditional data-driven machine learning approaches, it is demonstrated that PINNs do not require large datasets-or any datasets in simple cases-as they are trained using a physics-driven loss function formulated from the residuals of governing partial differential equations (PDEs), boundary conditions, and initial conditions. In particular, in high-dimensional fluid simulations, PINNs offer significant potential to reduce computational costs, where traditional CFD methods often require substantial computational time and resource demand as the dimensionality of a problem increases. In contrast, PINNs enable efficient inference once trained and can further extend scalable acceleration through transfer learning. For example, Tang et al. used transfer learning to model vortex-induced vibrations, reducing the required training data by 87.5% (using only 1/8 of the original data), while achieving a 10-fold decrease in computation time with a maximum relative error of 8.51% [10]. Prantikos et al. used transfer learning to accelerate the prediction of nuclear reactor transients, achieving at least a 16-fold acceleration in training convergence, with prediction times under sub 10 seconds [11]. In another study, Ohashi et al. applied a transfer learning scheme to PINNs for modeling two-dimensional (2D) transient temperature fields, achieving a 9.9-fold reduction in computational time-from 8.32 to 0.84 hours [12]. Lastly, coupled with extreme learning machines (ELM), a physics-informed ELM demonstrated training time reduction from eight hours to a few seconds for simple 1D and 2D linear PDEs [13]. These results highlight the potential of PINNs as forward-predictive models that scale to complex, multi-scale geometries and adapt to diverse flow conditions. Consequently, PINNs are promising methodologies to solve challenging high-dimensional physics problems, such as modeling ill-posed fluid systems where conventional CFD methods struggle due to under-constrained conditions and the existence of multiple plausible solutions [1, 14]. PINNs have been successfully applied to a variety of problems, including steady and unsteady heat conduction, convection, and incompressible Navier–Stokes problems, offering a mesh-free and differentiable alternative to conventional numerical solvers [5, 15].

Despite the advantages of PINNs for fluid simulations, their application to three-dimensional (3D) fluid systems with complex flow structures remains challenging. Chan et al. systematically evaluated the performance of PINNs against a supervised learning approach trained on extensive CFD data for steady and pulsatile flow in a 3D curved tube with varying geometries. Their findings showed that PINNs were only successful under carefully controlled steady-state conditions, including (1) the use of a transformed coordinate system, (2) enforcement of no-slip boundary conditions as hard constraints, and (3) specialized strategies to propagate boundary effects [16]. The complexity of 3D domains amplifies challenges already encountered in 2D simulations—most notably, the handling of corner singularities. In 2D, these singularities typically occur at domain corners where abrupt changes in boundary conditions or geometry lead to discontinuities or sharp gradients in flow variables. In 3D, the problem becomes more pronounced as these singularities extend along edges or lines formed by the intersection of surfaces [17, 18]. Cai et al. explored transient 3D flow reconstruction around a cylinder at Reynolds number 200, which required a large number of simulation data (at least 344,650 data points) and collocation points (3×10^6) for PINN training [5]. In another study, Moser et al. investigated PINNs applied to flow through cylindrical and aneurysmal geometries, using up to 17×10^6 collocation points and enforcing mass conservation via Monte Carlo integration across continuity planes [19]. While Deep Galerkin Methods provided the best performance—particularly for simple geometries—the models still exhibited significant pressure errors in complex aneurysms. Collectively, these studies highlight the current limitations of 3D PINNs, even with relatively simple geometries. Notably, existing work has largely focused on geometries with nearly uniform aspect ratios, where the streamwise and spanwise dimensions are roughly equal.

In most of the literature, PINNs based on a single neural network have shown limited effectiveness in modeling high-aspect-ratio 3D fluid channels [20]. Our separate empirical studies find that this occurs around a channel length that exceeds the depth by a factor greater than five—primarily due to challenges in convergence and solution accuracy. Empirical studies on various geometries can be found in Appendix A.1. High-aspect-ratio domains introduce pronounced disparities in characteristic length scales, often resulting in vanishing gradients during training and significantly imbalanced loss contributions across physical equations along different orientations. As a consequence, single-network PINNs tend to prioritize dominant flow directions and large-scale structures while underrepresenting

transverse or localized features critical to accurate prediction. This imbalance can lead to poor generalization, limited predictive accuracy, and unstable convergence behavior. To address these issues, recent studies have proposed decomposing the computational domain and assigning dedicated subnetworks to individual physical components (e.g., momentum, continuity, energy). Multi-network, multi-domain PINN (MDPINN) architecture with domain decomposition strategies have demonstrated greater effectiveness for such geometrically complex systems. One such example is given by a study investigating 3D blood flow in a three-branch intersection, in which the L_2 error was around 7.8% but required 8×10^5 epochs [21]. Another is given by Shukla et al. in which 2D steady-state lid driven cavity flow and heat conduction were investigated [22]. The authors showed advancements in computational time by leveraging parallel tasks. Next, Laubscher [23] and Merdasi et al. [24] demonstrated that multi-network architectures—where subnetworks are optimized independently within subdomains or for specific governing equations—can significantly improve training stability and accuracy. This method is further described by Dwivedi et al., which introduced distributed PINN for each subdomain with additional flux conditions at cell interfaces as local regularizer when combining together, and demonstrated higher accuracy and better data efficiency over PINNs [25]. Other various interface conditions and irregular domain geometries have also been studied by Jagtap et al. [26, 27]. Due to many interface conditions, Li et al. proposed a meta-learning strategy to dynamically determine the appropriate constraints [28]. However, these studies explored domains limited to 2D with simple constraints. These approaches mitigate the complexity faced by any single network, making them particularly suitable for high-aspect-ratio geometries characterized by anisotropic behavior and multiscale physics. Advancements, such as causal sweeping strategies, loss term balancing, and transfer learning, have improved training time and accuracy to a certain degree, yet more research is required in higher-dimensional domain [29, 20, 30].

Here, we investigate a MDPINN strategy and further improve with global dynamic enforcement, applied to 3D intersecting fluid channels characterized by large aspect ratios of 7.1:1.4:1 (length:width:depth) and strong flow mixing at the intersection. Fluid channel networks that promote flow mixing have broad applications, particularly in cold plates for electronics cooling, where flow mixing disrupts thermal boundary layer development and improves localized heat removal. Another application is in species mixing (e.g., chemical, particulates, fuel/air), in which a homogeneous mixture is desired. The overall model consists of an ensemble of PINNs optimized jointly with parallel parameter updates. To ensure physical consistency across subdomains, we introduce two additional loss components: (1) a global continuity loss (MDPINN-GD) and (2) matching boundary condition losses, which enforce conservation laws and interface compatibility, respectively.

2 Problem setup

2.1 Design domain

Cross-flow channels are widely used in fluidic networks due to their simple design and ability to create localized mixing zones that enhance fluid homogeneity and further heat transfer efficiency. Fig. 1 (a) illustrates a 3D cross-shaped channel configuration, where two inlet ports are positioned on the left and bottom faces, two outlet ports on the right and top faces and two channels intersect in the middle. This configuration establishes a central cross-flow region that promotes fluid interaction and recirculation. The channel dimensions are set to $500\,\mu\mathrm{m}$ in length l, $100\,\mu\mathrm{m}$ in width w, and $70\,\mu\mathrm{m}$ in depth d. Liquid water at 25 °C, with a kinematic viscosity of $\nu = 8.917\times10^{-7}$ m^2/s and density of $\rho_0 = 997$ kg/m^3, enters both inlets at a uniform velocity (U_0) of 2.68 m/s. The objective is unsupervised learning, predicting the velocity and pressure fields within the domain using governing equations and boundary conditions only, without any simulation data or experimental measurements. The flow regime is characterized by the Reynolds number [Re = $U_0 D_H/\nu$], where D_H is the hydraulic diameter. By tuning Re, various flow regimes can be explored, but to limit the scope of this work, Re is fixed to 247.5, with system-wide aspect ratios of $\gamma_1 = l/w = 5$ and $\gamma_2 = l/d = 7.14$. To evaluate the accuracy of the PINN predictions, we use a reference CFD simulation conducted in Ansys Fluent. The CFD domain is discretized at a resolution of $1\,\mu\mathrm{m}$, resulting in approximately 6.3×10^6 cells. The model assumes laminar flow with normalized convergence criteria of 1×10^{-5} for all residuals.

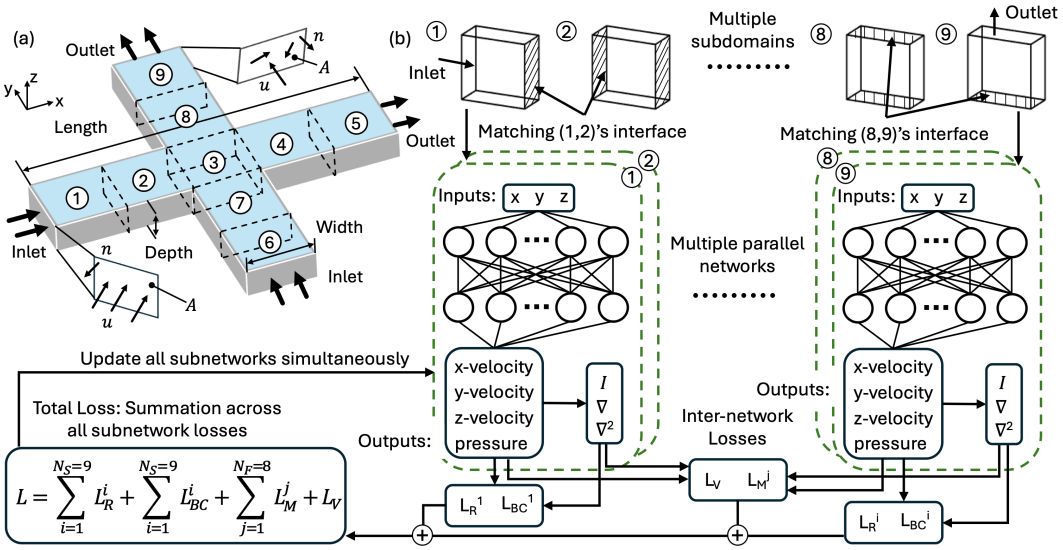

Figure 1: (a) 3D cross-intersecting fluid channel. (b) MDPINN network architecture in which each subdomain is assigned a separate PINN. Network loss includes local and interface terms. Index i runs from 1 through 9, corresponding to the nine subdomains. Index j runs from 1 through 8, corresponding to the eight subdomains interfaces.

2.2 Governing equations

For demonstration purposes, we focus on a simplified flow regime, steady-state laminar flow, governed by the 3D steady-state incompressible Navier-Stokes equations. To improve generalization and numerical stability, the governing PDEs are non-dimensionalized, allowing the system to be characterized in terms of dimensionless parameters. Furthermore, input and output variables are normalized to the range of 0 to 1, which align with the effective range of common activation functions used in PINNs (e.g., tanh and swish [2]). Consequently, the problem formulation incorporates the following normalized terms, denoted by '*' to signify the dimensional parameters with $x = x^*/l$, $y = y^*/w$, $z = z^*/d$, $U = U^*/U_0$, $V = V^*/U_0$, $W = W^*/U_0$, and $P = P^*/\rho_0 U_0^2$. x, y, z are dimensionless coordinates in length, width, and depth directions, respectively, defined by the dimensional variables $x^* \in [0, l]$, $y^* \in [0, w]$, and $z^* \in [0, d]$. U, V, W are the dimensionless velocity components in the x-, y-, and z-directions, respectively, normalized by U_0. P is the dimensionless pressure, normalized by $\rho_0 U_0^2$. The non-dimensional PDEs of continuity, x-, y-, and z-momentum in Cartesian coordinates are listed in order at Eq. 1.

$$
\begin{aligned}
&\frac{\partial U}{\partial x} + \gamma_1 \frac{\partial V}{\partial y} + \gamma_2 \frac{\partial W}{\partial z} = 0 \\
&U\frac{\partial U}{\partial x} + \gamma_1 V \frac{\partial U}{\partial y} + \gamma_2 W \frac{\partial U}{\partial z} + \frac{\partial P}{\partial x} - \frac{1}{\mathrm{Re}}\left(\frac{\partial^2 U}{\partial x^2} + \gamma_1^2 \frac{\partial^2 U}{\partial y^2} + \gamma_2^2 \frac{\partial^2 U}{\partial z^2}\right) = 0 \\
&U\frac{\partial V}{\partial x} + \gamma_1 V \frac{\partial V}{\partial y} + \gamma_2 W \frac{\partial V}{\partial z} + \gamma_1 \frac{\partial P}{\partial y} - \frac{1}{\mathrm{Re}}\left(\frac{\partial^2 V}{\partial x^2} + \gamma_1^2 \frac{\partial^2 V}{\partial y^2} + \gamma_2^2 \frac{\partial^2 V}{\partial z^2}\right) = 0 \\
&U\frac{\partial W}{\partial x} + \gamma_1 V \frac{\partial W}{\partial y} + \gamma_2 W \frac{\partial W}{\partial z} + \gamma_2 \frac{\partial P}{\partial z} - \frac{1}{\mathrm{Re}}\left(\frac{\partial^2 W}{\partial x^2} + \gamma_1^2 \frac{\partial^2 W}{\partial y^2} + \gamma_2^2 \frac{\partial^2 W}{\partial z^2}\right) = 0
\end{aligned}
\tag{1}
$$

3 Multiple domain physics-informed neural network (MDPINN)

To address the optimization challenges associated with training a single-network PINN on 3D domains with high aspect ratios, we adopt a domain decomposition strategy that improves both convergence and scalability. As shown by the black dotted lines in Fig. 1 (a), the domain is divided into nine subdomains with reduced aspect ratios. Each subdomain measures $100\,\mu\mathrm{m}$ in length, $100\,\mu\mathrm{m}$ in width, and $70\,\mu\mathrm{m}$ in depth, resulting in nearly cube-like volumes with an aspect ratio of 1.4:1.4:1. An overview of the MDPINN framework is shown in Fig. 1 (b), where each subdomain is assigned to an

individual subnetwork. Domain decomposition allows each subnetwork to learn within a subdomain with reduced geometric aspect ratios, which helps mitigate vanishing gradient issues and imbalanced loss contributions commonly observed in single-network PINNs applied to elongated geometries. The training loop shows both intra-network losses (within each subnetwork) and inter-network losses (across subnetworks). The losses are aggregated across all subnetworks into a total loss function, which is minimized through a single backpropagation step. This scheme enables simultaneous parameter updates across all subnetworks.

Each subnetwork is implemented as a fully-connected network that takes 3D Cartesian coordinates as input and predicts four flow variables as output: P, U, V, and W. Based on literature and empirical hyperparameter tuning, each subnetwork and training settings are configured as follows with the range explored in parentheses, offering a balance between accuracy and computational efficiency: 1) 7 hidden layers (5~10), 2) 100 neurons per layer (100~150), 3) 100,000 epochs for Adam optimizer (50K~100K), 4) 20,000 collocation points (10K~40K) randomly sampled within each subdomain using Latin hypercube sampling [31], and 5) boundary points uniformly sampled from a grid with $1\,\mu\mathrm{m}$ spacing ($1\,\mu\mathrm{m}$~$4\,\mu\mathrm{m}$). The activation function used is the adaptive sine [32]. Network optimization was conducted in two stages-the Adam optimizer with a learning rate of 1×10^{-4}, followed by the L-BFGS optimizer until convergence. The network was built using PyTorch and its associated libraries. A pseudocode outlining the network training process is included in Appendix A.2.

3.1 Interface-matching loss

To maintain physical consistency across adjacent subdomains, continuity is enforced by applying matching Dirichlet boundary conditions at the interfaces. The interface-matching loss is defined as:

$$L_M = \frac{1}{N_M}\sum_{n=1}^{N_M}(F_{1,n} - F_{2,n})^2 \tag{2}$$

where L_M represents the mean squared error (MSE) of field variables between adjacent subdomain interfaces, F_1 and F_2, across N_M matching points. This formulation ensures the continuity of fluid flow variables (e.g., velocity and pressure) across subdomain interfaces. It is important to note that the current interface-matching condition constrains only the zeroth-order continuity. Improved continuity and physical consistency can be achieved by extending the matching conditions to include higher-order derivatives, such as incorporating first-order derivatives of the flow variables (e.g., volumetric flux). However, further improvements by including higher-order derivatives incur the cost of increased computational complexity.

3.2 Global mass conservation loss (MDPINN-GD)

Whilst subdomain and interface predictions are greatly improved with MDPINN, there is still a tendency to violate global mass conservation when applied to complex 3D fluid systems. To mitigate this, we introduced an additional physics-based constraint into the loss function to enforce global volumetric flow rate conservation. Specifically, a global continuity loss term, denoted as L_V, is incorporated to penalize discrepancies between the total volumetric inflow and outflow across the entire fluid network. To maintain scale invariance and improve numerical stability during training, all flow rates are normalized against a reference volumetric flow rate, U_0A_c, where $A_c = wd$ is the cross-sectional area at the inlet. The volumetric flow rate loss was formulated as:

$$L_V = \left(\frac{1}{U_0A_c}\left(\sum_{l=1}^{N_{in}}(u\cdot n)_l A_l - \sum_{m=1}^{N_{out}}(u\cdot n)_m A_m\right)\right)^2 \tag{3}$$

where u is the local velocity vector, n is the unit normal vector to the boundary surface, and A_l and A_m represent the discretized cross-sectional areas perpendicular to n at the inlet and outlet boundaries, respectively. N_{in} and N_{out} correspond to the number of discrete inlet and outlet segments used in the numerical integration.

3.3 Composite loss function

The total loss function is constructed as a composite of multiple components: the sum of PDE residual loss, boundary condition loss, the interface-matching loss across all subdomains, and global mass

conservation. Thus, the composite loss function, L, to minimize is written as:

$$L = \frac{1}{N_S N_R} \sum_{i=1}^{N_S} \sum_{n=1}^{N_R} \left(e_n^i\right)^2 + \frac{1}{N_S N_{BC}} \sum_{i=1}^{N_S} \sum_{n=1}^{N_{BC}} \left(\xi_n^i - \xi_{0,n}^i\right)^2 + \frac{1}{N_F} \sum_{i=1}^{N_F} L_M^i + L_V \qquad (4)$$

where summation is performed across N_S = 9 subdomains and N_F = 8 interfaces. The number of collocation points, boundary points, PDE residual loss, MDPINN-GD predictions on the boundary, and boundary condition ground truths are represented by N_R N_{BC}, e_n^i, ξ_n^i, and $\xi_{0,n}^i$, respectively. Factors leading the double summations arise due to the MSE function. Additional details on traditional PINN loss functions can be found in the seminal work by Raissi et al. [1].

4 Results and discussion

4.1 Reference flow patterns

Simulated streamlines in three different planes, shown in Fig. 2 (a-c), illustrate the expected flow patterns in this 3D cross-flow channel. In the (a) xy-plane, the flow remains nearly straight until the intersection, where it is deflected into the perpendicular outlet branches, and recirculating flow near the outlet walls is clearly visible. In the (b) yz-plane at the entrance of the outlet section (subdomain 4), the flow moves upward, deflects at the top, and forms recirculating zones near the sidewalls. In the (c) yz-plane at the exit of the outlet section (subdomain 5), strong flow mixing occurs, with a prominent recirculation zone forming at the channel center. Overall, the simulation captures the key transport features of a 3D cross-flow junction, including laminar inlet development, intersection-induced deflection, and outlet mixing and recirculation. The CFD simulation results in Fig. 2 (d) further highlight the overall flow patterns, particularly in the xy-plane view at z = 0.5 (mid-plane). At the inlet boundaries located on the left and bottom, the flow exhibits classical Poiseuille flow features, with maximum stream-wise velocity at the center and decreasing velocity near walls. In contrast, the outlet boundaries on the top and right display more complex flow behavior due to the merging and redirection of streams.

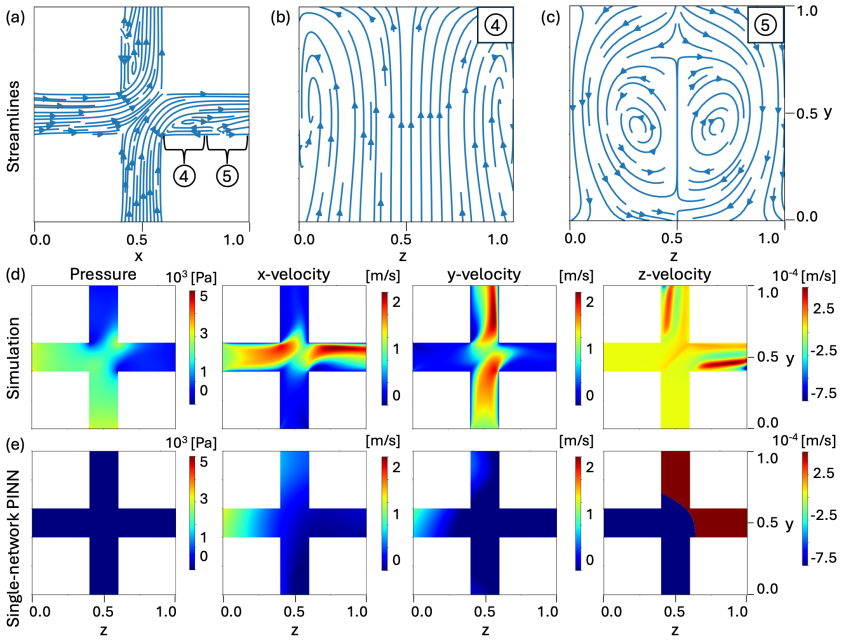

Figure 2: Streamline plots for simulation at: (a) xy-plane at z = 0.5, (b) inlet yz-plane of subdomain 4 at x = 0.6, and (c) outlet yz-plane of subdomain 5 at x = 1. These views demonstrate overall flow patterns and the change in flow profile as it moves downstream. Comparison of pressure, x-velocity, y-velocity, and z-velocity fields between the (d) CFD simulation and (e) single-network PINN model at the z = 0.5, viewed in the xy-plane.

4.2 Single-network PINN

To understand the limitations of single-network PINNs for modeling 3D fluid networks, we first implemented and tested a single-network architecture without domain decomposition. The network was optimized through hyperparameter tuning based on previous studies [5, 15, 33]. The architecture consisted of a fully connected network with 10 hidden layers and 150 neurons per layer, employing adaptive sine activation functions. Training was conducted with 250,000 collocation points (average separation of $2.444\,\mu\mathrm{m}$ between points) and boundary points sampled uniformly along a grid with a spacing of $1\,\mu\mathrm{m}$. The network was trained using the same two-stage optimization (Adam + L-BFGS) as the MDPINN. However, the model produced highly inaccurate predictions, as shown in the xy-plane view in Fig. 2 (e), with additional plane views provided in Appendix A.3. The network failed to capture the expected flow patterns shown in the simulation results: fluid entered only from the left inlet and dissipated rapidly along the channel walls, without meaningful circulation or cross-flow behavior. Notably, even with a low final loss (on the order of 1×10^{-6}), the single-network PINN was unable to resolve the flow features of the intersecting channel geometry. Additional experiments of various geometries are shown in Appendix A.1.

To quantitatively assess the model accuracy, we evaluated two error metrics: the mean absolute error, ϵ, and the maximum absolute error, ϵ_{max}, as given in Eq. 5, where Φ_i, $\Phi_{0,i}$, and N_p represent the PINN prediction, simulation result, and number of nodes in simulation, respectively. For the single-network PINN, ϵ for pressure, x-velocity, y-velocity, and z-velocity were 3.41 kPa, 0.52 m/s, 0.52 m/s, and 0.18 m/s, respectively. The corresponding ϵ_{max} were 23.6 kPa, 2.16 m/s, 2.16 m/s, and 1.79 m/s, respectively. This indicates mean errors of nearly 20% of U_0 for the x- and y-velocity components. Furthermore, the maximum velocity errors reach nearly 81% of U_0, suggesting that the discrepancies in the single-network PINN are on the same order of magnitude as U_0 of the system, which is highly undesirable as errors will impose similar influence on the flow dynamics as U_0.

$$\epsilon = \frac{1}{N_p}\sum_{i=1}^{N_p}|\Phi_i - \Phi_{0,i}| \qquad \epsilon_{\text{max}} = \max\left(|\Phi_i - \Phi_{0,i}|\right) \tag{5}$$

4.3 MDPINN-GD

To identify the most effective optimization strategy for the MDPINN-GD framework, we compared three approaches in terms of their accuracy and computational efficiency. The approaches included: (1) MDPINN_1, a baseline model trained with 20,000 collocation points (average separation of $2.729\,\mu\mathrm{m}$ between points) per subdomain; (2) MDPINN_2, a higher-resolution model using 40,000 collocation points (average separation of $2.165\,\mu\mathrm{m}$ between points) per subdomain; and (3) MDPINN-GD, an optimized model using 20,000 collocation points per subdomain incorporating a global mass conservation constraint. The training loss plot can be found in Appendix A.4. The training times for MDPINN_1, MDPINN_2, and MDPINN-GD were 24.3 hours, 38.8 hours, and 68 hours, respectively. Corresponding GPU memory usage was approximately 40 GB, 60 GB, and 45 GB. All models were trained on a single Nvidia A100 80GB GPU.

The field predictions from the three MDPINN networks are shown in Fig. 3. MDPINN_1 failed to satisfy the mass continuity constraint, as evidenced by zero outlet flows in the y-velocity field. Additionally, the pressure at the bottom-face inlet was lower than that at the right-face outlet, contradicting the expected pressure gradient from inlet to outlet. However, compared to single-network PINN, the results indicate that MDPINN outperforms even with fewer collocation points. MDPINN_2, which utilized doubled collocation point density, showed improved mass continuity and captured general flow patterns. This suggests that increasing the number of collocation points can enhance prediction quality, although it incurs additional computational cost. However, MDPINN_2 still underestimated both velocity magnitudes and pressure values in the outflow region, implying that greater collocation point density alone was insufficient for accurate field prediction in regions of complex flow behavior. These results suggest that using a multi-network architecture without explicit enforcement of global continuity is insufficient for accurately predicting complex flow fields. In contrast, MDPINN-GD successfully demonstrated mass continuity and accurate magnitudes for all flow variables throughout the domain. A further comparison in the channel cross-section (yz-plane view) at the entrance of the outlet section (subdomain 4) revealed that MDPINN-GD accurately captured both x- and y-velocity fields, whereas MDPINN_2 significantly underpredicted the y-velocity relative to the CFD results.

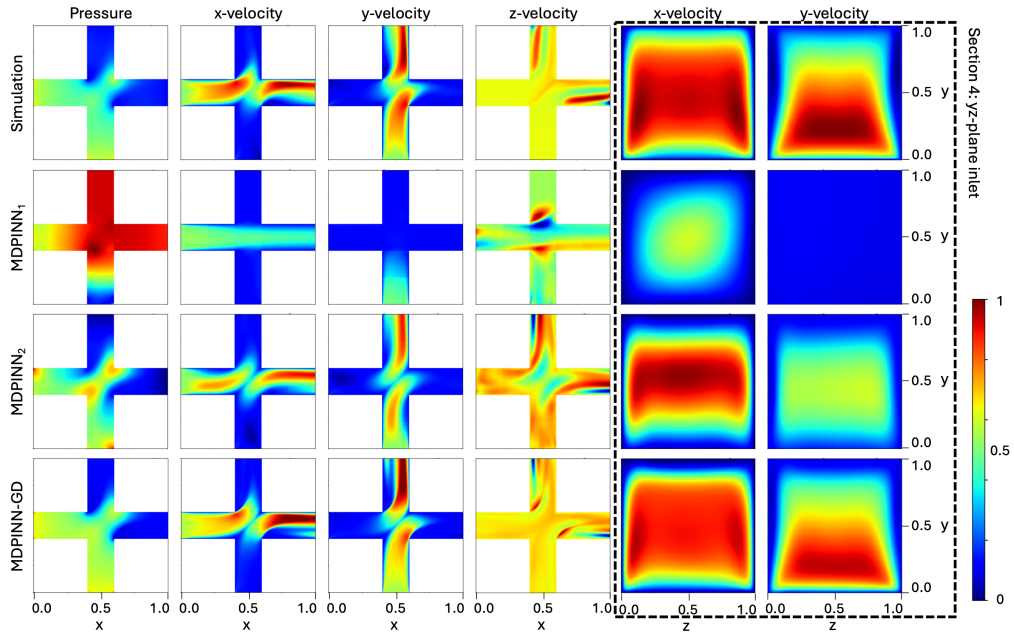

Figure 3: Pressure, x-velocity, y-velocity, and z-velocity field comparisons of CFD simulation and MDPINN models of xy-plane at z = 0.5 (mid-plane). Additional views for x and y-velocity of yz-plane at the inlet of subdomain 4 is also shown to highlight 3D features.

To quantitatively compare the predictive accuracy, we evaluated ϵ and ϵ_{max} across all four fields. The MDPINN-GD achieved a mean ϵ improvement of 76.2% and 52.9% over MDPINN$_1$ and MDPINN$_2$, respectively. Similarly, the mean ϵ_{max} improvement of MDPINN-GD over MDPINN$_1$ and MDPINN$_2$ was 75.2%, and 49%, respectively. These results indicate that improvements are observed evenly across ϵ and ϵ_{max}. From MDPINN$_1$ to MDPINN$_2$, comparison shows that doubling the collocation points results in approximately 50% decrease in error, which yields to comparable flow field predictions. However, it clearly shows that incorporating global constraints provides further substantial accuracy gains without additional collocation points. Overall, MDPINN-GD achieves roughly twice the accuracy of MDPINN$_2$. These results underscore the importance of combining domain decomposition with global continuity enforcement and sufficient sampling to achieve robust and physically consistent PINN predictions in complex 3D flow configurations. Error comparisons for all PINN models can be found in Appendix A.5.

To understand how the complexity of fluid dynamics affects model accuracy, ϵ and ϵ_{max} were evaluated for each subdomain, as shown in Table 1. A key observation is that the largest streamwise errors (i.e., errors aligned with the main flow direction) occurred in the outlet subdomains, where flow deceleration and recirculation introduce additional physical complexity. The largest x- and y-velocity error, ϵ (U) and ϵ (W), is 29.9×10^{-2} m/s, corresponding to 14.9% of U_0. Notably, the largest z-velocity and pressure errors, ϵ (W) and ϵ (P), are observed in the cross junction (subdomain 3), where converging inlet flows subsequently diverge, creating strong velocity gradients in all directions. Complete flow fields at the inlets and outlets of subdomain 3 are shown in Appendix A.6. Analysis of ϵ and ϵ_{max} shows similar trends, with slight deviations of symmetry in the values. For example, comparing ϵ_{max} along the streamwise directions in subdomains 1, 5, 6, and 9 (inlets and outlets) reveals that this asymmetry is likely due to numerical randomness rather than physical effects. Additional error metrics normalized by inlet velocity and inlet dynamic pressure are provided in Appendix A.7.

The MDPINN-GD results are further illustrated in Fig. 4, which presents five views of the x-velocity field alongside corresponding predictions and absolute error distributions. In the xy-plane, error concentrations appear specifically at the diagonal intersection of the channels and the right outlet, likely due to intense mixing and sharp changes in velocity direction. Increased errors are also observed at the inlets of subdomain 4 (yz-plane) and subdomain 8 (xz-plane), particularly near the top and right sides, respectively. These regions correspond to the top-right corner of the channel intersection, where the fluid undergoes sharp directional changes, potentially leading to localized numerical singularities. In addition, the outlet regions of subdomain 5 (yz-plane) and subdomain

Table 1: Subdomain ϵ and ϵ_{max} for MDPINN-GD with **bold** indicating maximum across each row.

		Left inlet		Intersect	Right outlet		Bottom inlet		Top outlet	
Subdomain number		1	2	3	4	5	6	7	8	9
ϵ (P)	$[10^2$ Pa]	1.95	1.04	**16.5**	13.9	2.15	3.31	1.13	14.5	2.25
ϵ (U)	$[10^{-2}$ m/s]	9.65	18.9	21.5	24.4	**29.9**	1.96	2.64	11.7	8.16
ϵ (V)	$[10^{-2}$ m/s]	1.85	2.56	21.4	11.7	8.21	9.84	18.9	24.4	**29.7**
ϵ (W)	$[10^{-2}$ m/s]	2.79	3.46	**7.99**	6.88	6.27	2.92	3.46	6.92	6.16
ϵ_{max} (P)	$[10^2$ Pa]	74.2	12.2	25.4	22.2	4.96	**85.1**	10.2	23.2	4.96
ϵ_{max} (U)	[m/s]	1.51	0.702	1.04	1.41	**1.54**	0.39	0.31	0.69	0.44
ϵ_{max} (V)	[m/s]	0.43	0.41	1.04	0.67	0.44	**1.72**	0.69	1.47	1.60
ϵ_{max} (W)	$[10^{-1}$ m/s]	4.28	2.96	**6.39**	5.49	4.46	4.14	3.04	5.49	4.52

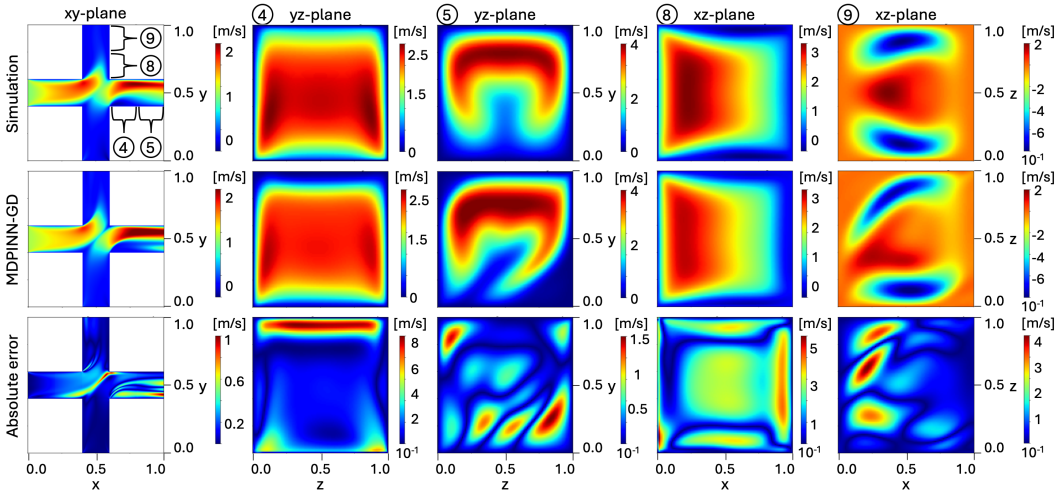

Figure 4: Comparison of x-velocities between CFD simulation and MDPINN-GD with absolute errors. Viewed at: xy-plane at $z = 0.5$ (mid-plane), yz-plane at inlet of subdomain 4, yz-plane at outlet of subdomain 5, xz-plane at inlet of subdomain 8, and xz-plane at outlet of subdomain 9.

9 (xz-plane) exhibit asymmetric errors near the walls, likely resulting from the propagation of numerical artifacts originating at the corners. These artifacts may accumulate along the flow path, contributing to the final prediction errors. Additional visualization and error distributions can be found in Appendix A.6.

5 Conclusion

In summary, this work addresses key limitations in applying PINNs to the modeling of fluid dynamics in complex geometries, i.e., intersecting 3D fluid channels with a high aspect ratio. Traditional PINNs often struggle with convergence and predictive accuracy in high-aspect-ratio geometries, which induce intricate flow features such as deflections, mixing, and recirculations. To overcome these challenges, we first demonstrated that a vanilla MDPINN that leverages domain decomposition to facilitate localized learning, MDPINN$_1$, was able to outperform the single-network PINN with 39% less collocation points while decreasing error by 90.2%. Next, we presented an enhanced MDPINN-GD framework that enforces global dynamics by integrating global mass conservation constraints into the loss formulation. Together, these enhancements led to a 52.9%~74.8% improvement in average predictive accuracy over the single-network PINN and MDPINN architectures without the global mass conservation. Furthermore, mean absolute velocity errors of MDPINN-GD were below 14.9% when normalized by the inlet velocity. This indicates that the proposed method is able to accurately predict the complex flow phenomena, with minimal errors likely caused by the propagation of corner singularities. Overall, the proposed approach effectively captures complex cross-flow interactions and resolves steep inlet pressure gradients, demonstrating its potential as a scalable, high-fidelity and physically consistent surrogate modeling tool for fluid simulations.

Limitations Despite these advancements, several limitations remain. As shown in Fig. 3, the predicted z-velocity exhibits noticeable deviations from the CFD baseline, suggesting a need for finer spatial resolution or improved learning of secondary flow features. This issue underscores the necessity for future research into adaptive loss weighting strategies, multi-scale network architectures, enforcing boundary conditions as hard constraints, and enhanced interface treatments for domain coupling. Furthermore, the demonstrated method has so far been applied to a 3D intersecting channel flow; experiments on other geometries can be helpful to assess its robustness and generalizability. Future research will focus on advancing the MDPINN-GD framework through more adaptive and generalizable learning strategies on various geometries. Additionally, we plan to extend the framework to handle transient and multi-physics systems, thereby demonstrating the scalability of PINNs in dynamic and coupled environments. We also acknowledge that systematic benchmarking of inference speed across various discretization levels are necessary to justify computational efficiency claims. We consider this an important direction for future work.

Acknowledgments and Disclosure of Funding

We would like to recognize Nagahiro Ohashi and Prof. Beomjin Kwon (School for Engineering of Matter, Transport and Energy at Arizona State University) for their substantial intellectual contributions to this work. They have led and were deeply involved in the research design, execution, and preparation of the manuscript. Due to an unintentional clerical error during the submission process, they are not listed as co-authors (Nagahiro Ohashi as the first author and Prof. Beomjin Kwon as the second author) of this NeurIPS version, but they should be regarded as key contributors to the ideas and results presented in this paper. This work was supported by the National Science Foundation grant under Grant No. 2337973. Additionally, this research was partially supported by the Nano & Material Technology Development Program through the National Research Foundation of Korea (NRF), funded by Ministry of Science and ICT (RS-2024-00448639) and Arizona State University startup funds.

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, February 2019.

[2] Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. Scientific Machine Learning Through Physics–Informed Neural Networks: Where we are and What's Next. *Journal of Scientific Computing*, 92(3):88, July 2022.

[3] Zaharaddeen Karami Lawal, Hayati Yassin, Daphne Teck Ching Lai, and Azam Che Idris. Physics-Informed Neural Network (PINN) Evolution and Beyond: A Systematic Literature Review and Bibliometric Analysis. *Big Data and Cognitive Computing*, 6(4):140, December 2022. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute.

[4] Solji Choi, Ikhwan Jung, Haeun Kim, Jonggeol Na, and Jong Min Lee. Physics-informed deep learning for data-driven solutions of computational fluid dynamics. *Korean Journal of Chemical Engineering*, 39(3):515–528, March 2022.

[5] 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, June 2021.

[6] Xiao-dong Bai, Yong Wang, and Wei Zhang. Applying physics informed neural network for flow data assimilation. *Journal of Hydrodynamics*, 32(6):1050–1058, December 2020.

[7] Yikun Yang, Xifeng Wang, Jinfeng Li, and Riletu Ge. A data-physic driven method for gear fault diagnosis using PINN and pseudo-dynamic features. *Measurement*, 236:115124, August 2024.

[8] Félix Fernández de la Mata, Alfonso Gijón, Miguel Molina-Solana, and Juan Gómez-Romero. Physics-informed neural networks for data-driven simulation: Advantages, limitations, and opportunities. *Physica A: Statistical Mechanics and its Applications*, 610:128415, January 2023.

[9] Lu Zhu, Xianyang Jiang, Adrien Lefauve, Rich R. Kerswell, and P. F. Linden. Physics-informed neural network to augment experimental data: an application to stratified flows, September 2023. arXiv:2309.14722 [physics].

[10] Hesheng Tang, Yangyang Liao, Hu Yang, and Liyu Xie. A transfer learning-physics informed neural network (TL-PINN) for vortex-induced vibration. *Ocean Engineering*, 266:113101, December 2022.

[11] Konstantinos Prantikos, Stylianos Chatzidakis, Lefteri H. Tsoukalas, and Alexander Heifetz. Physics-informed neural network with transfer learning (TL-PINN) based on domain similarity measure for prediction of nuclear reactor transients. *Scientific Reports*, 13(1):16840, October 2023. Publisher: Nature Publishing Group.

[12] Nagahiro Ohashi, Leslie K. Hwang, and Beomjin Kwon. Physics-informed neural networks for multi-field visualization with single-color laser induced fluorescence. *AI Thermal Fluids*, 1:100005, March 2025.

[13] Vikas Dwivedi and Balaji Srinivasan. Physics Informed Extreme Learning Machine (PIELM)–A rapid method for the numerical solution of partial differential equations. *Neurocomputing*, 391:96–118, May 2020.

[14] George Em Karniadakis, Ioannis G. Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, June 2021.

[15] Xiaowei Jin, Shengze Cai, Hui Li, and George Em Karniadakis. NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations. *Journal of Computational Physics*, 426:109951, February 2021.

[16] Wei Xuan Chan, Wenhao Ding, Binghuan Li, Hong Shen Wong, and Choon Hwai Yap. Role of physics-informed constraints in real-time estimation of 3D vascular fluid dynamics using multi-case neural network. *Computers in Biology and Medicine*, 190:110074, May 2025.

[17] 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, June 2024.

[18] Gaurav Kumar Yadav, Sundararajan Natarajan, and Balaji Srinivasan. Distributed PINN for Linear Elasticity — A Unified Approach for Smooth, Singular, Compressible and Incompressible Media. *International Journal of Computational Methods*, 19(08):2142008, October 2022. Publisher: World Scientific Publishing Co.

[19] Philipp Moser, Wolfgang Fenz, Stefan Thumfart, Isabell Ganitzer, and Michael Giretzlehner. Modeling of 3D Blood Flows with Physics-Informed Neural Networks: Comparison of Network Architectures. *Fluids*, 8(2):46, February 2023. Number: 2 Publisher: Multidisciplinary Digital Publishing Institute.

[20] Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael W Mahoney. Characterizing possible failure modes in physics-informed neural networks. In *Advances in Neural Information Processing Systems*, volume 34, pages 26548–26560. Curran Associates, Inc., 2021.

[21] Linyan Gu, Shanlin Qin, Lei Xu, and Rongliang Chen. Physics-informed neural networks with domain decomposition for the incompressible Navier–Stokes equations. *Physics of Fluids*, 36(2):021914, February 2024.

[22] Khemraj Shukla, Ameya D. Jagtap, and George Em Karniadakis. Parallel physics-informed neural networks via domain decomposition. *Journal of Computational Physics*, 447:110683, December 2021.

[23] R. Laubscher. Simulation of multi-species flow and heat transfer using physics-informed neural networks. *Physics of Fluids*, 33(8):087101, August 2021.

[24] Arshia Merdasi, Saman Ebrahimi, Xiang Yang, and Robert Kunz. Physics Informed Neural Network application on mixing and heat transfer in combined electroosmotic-pressure driven flow. *Chemical Engineering and Processing - Process Intensification*, 193:109540, November 2023.

[25] Vikas Dwivedi, Nishant Parashar, and Balaji Srinivasan. Distributed physics informed neural network for data-efficient solution to partial differential equations. *Neurocomputing*, 420:299–316, January 2021. arXiv:1907.08967 [cs].

[26] Ameya D. Jagtap, Ehsan Kharazmi, and George Em Karniadakis. Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. *Computer Methods in Applied Mechanics and Engineering*, 365:113028, June 2020.

[27] Ameya D. Jagtap and George Em Karniadakis. Extended Physics-Informed Neural Networks (XPINNs): A Generalized Space-Time Domain Decomposition Based Deep Learning Framework for Nonlinear Partial Differential Equations. *Communications in Computational Physics*, 28(5), November 2020. Institution: Brown Univ., Providence, RI (United States) Publisher: Global Science Press.

[28] Shibo Li, Michael Penwarden, Yiming Xu, Conor Tillinghast, Akil Narayan, Robert M. Kirby, and Shandian Zhe. Meta Learning of Interface Conditions for Multi-Domain Physics-Informed Neural Networks. In *Proceedings of the 40th International Conference on Machine Learning*, pages 19855–19881, July 2023.

[29] Michael Penwarden, Ameya D. Jagtap, Shandian Zhe, George Em Karniadakis, and Robert M. Kirby. A unified scalable framework for causal sweeping strategies for Physics-Informed Neural Networks (PINNs) and their temporal decompositions. *Journal of Computational Physics*, 493:112464, November 2023.

[30] Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and Mitigating Gradient Flow Pathologies in Physics-Informed Neural Networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, January 2021. Publisher: Society for Industrial and Applied Mathematics.

[31] Wei-Liem Loh. On Latin hypercube sampling. *The Annals of Statistics*, 24(5):2058–2080, October 1996. Publisher: Institute of Mathematical Statistics.

[32] Ameya D. Jagtap, Kenji Kawaguchi, and George Em Karniadakis. Adaptive activation functions accelerate convergence in deep and physics-informed neural networks. *Journal of Computational Physics*, 404:109136, March 2020.

[33] Shengze Cai, Zhiping Mao, Zhicheng Wang, Minglang Yin, and George Em Karniadakis. Physics-informed neural networks (PINNs) for fluid mechanics: a review. *Acta Mechanica Sinica*, 37(12):1727–1738, December 2021.

A Technical Appendices and Supplementary Material

A.1 Illustration of PINN failures

Here we show some typical failure points of single-network PINNs, which demonstrates the difficulties of training on non-canonical geometries. A 2D straight channel evaluated at two different aspect ratios are shown. The pressure fields are good indicators of poor network performance, since it is typically the hardest to resolve. Fig. A.1 (a) and (b) show channel aspect ratio of 1:5 and 1:10 (width:length), respectively. It is clearly visible that these high aspect ratios pose an intrinsic challenge due to multi-scale issues. Next, we demonstrate a 3D cubic channel domain. Fig. A.1 (c) shows the yz-plane view PINN predictions at the outlet, in which asymmetry along the z-axis is observed. The likely sources of error are corner singularities that PINNs struggle to approximate.

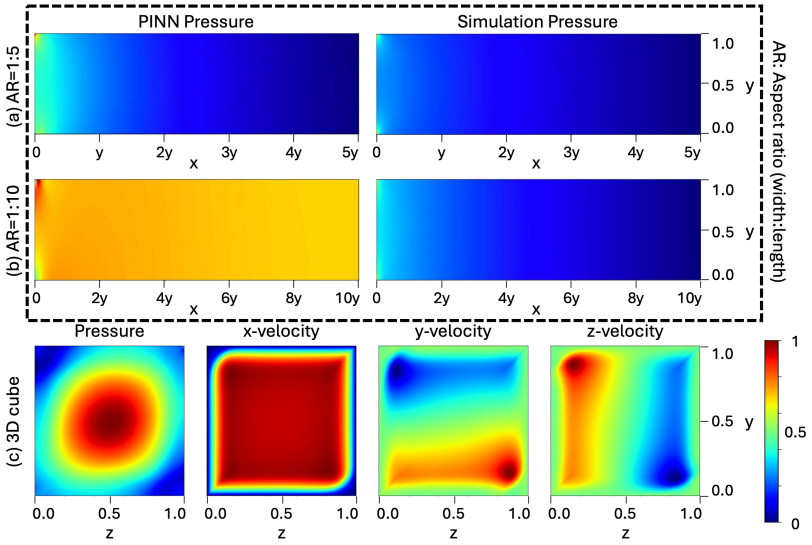

Figure A.1: Demonstration of 2D straight channel PINN and simulation pressure field with aspect ratios (width:length) of (a) 1:5 and (b) 1:10. 3D cubic channel flow fields at the outlet are shown in (c) to demonstrate effects of corner singularities.

A.2 MDPINN-GD pseudocode

Algorithm 1 MDPINN-GD with N_S domain decomposition

Input: Local spatial coordinates: x, y, z
Output: Local fluid parameters: P, U, V, W

Create N_S subdomain geometries
Define subdomain neural network architecture: layers l, neurons n
Define activation function (i.e., adaptive sine)
Define N_R collocation points and boundary points
Define fluid properties, boundary conditions and initial conditions (if any)
Define matching conditions and global continuity constraints
for $s = 0$ to N_S **do**
    Assign to $l \times n$ fully-connected neural network
    Initialize network weights and bias: **W**$\sim$Xavier normal, **b** = 0
end for
Initialize loss values: L_R, L_{BC}, L_M = 0
Load **W** and **b** into Adam and L-BFGS optimizer
for $e = 0$ to max_epoch **do**
    for $s = 0$ to N_S **do**
        Extract s$^{\text{th}}$ subdomain geometric and fluid properties
        Compute fluid parameters on all coordinates with forward propagation
        Calculate PDE loss for s$^{\text{th}}$ subdomain, L_R^s
        Add L_R^s to total L_R
        Extract s$^{\text{th}}$ subdomain boundary conditions
        Calculate boundary loss for s$^{\text{th}}$ subdomain, L_{BC}^s
        Add L_{BC}^s to total L_{BC}
    end for
    for $f = 0$ to N_F **do**
        Extract pairs of subdomain matching conditions
        Calculate matching loss for f$^{\text{th}}$ subdomain pair, L_M^f
        Add L_M^f to total L_M
    end for
    for $i = 0$ to num_inlets **do**
        Calculate volumetric flow rate at inlet face i, Q_{in}^i
        Add Q_{in}^i to total Q_{in}
    end for
    for $o = 0$ to num_outlets **do**
        Calculate volumetric flow rate at outlet face o, Q_{out}^o
        Add Q_{out}^o to total Q_{out}
    end for
    Calculate global continuity loss, $L_V = (Q_{out} - Q_{in})^2$
    Calculate total loss, $L = \frac{L_R}{N_S} + \frac{L_{BC}}{N_S} + \frac{L_M}{N_F} + L_V$
    if $e <$ Adam_max_epoch **then**
        Reset Adam optimizer gradient
        Calculate $\frac{\partial L}{\partial \theta}$ with backpropagation
        Update Adam optimizer
    else
        Reset L-BFGS optimizer gradient
        Calculate $\frac{\partial L}{\partial \theta}$ and $\frac{\partial^2 L}{\partial \theta^2}$ with backpropagation
        Update L-BFGS optimizer
    end if
    if $L <$ convergence_criteria **then**
        End training
    end if
end for

A.3 Model comparisons

Comparisons of single-network PINN, MDPINN$_1$, MDPINN$_2$, and MDPINN-GD for various views are given in the following figures. Flow fields are shown at the outlets of subdomains 5 and 9, and inlets of subdomains 4 and 8.

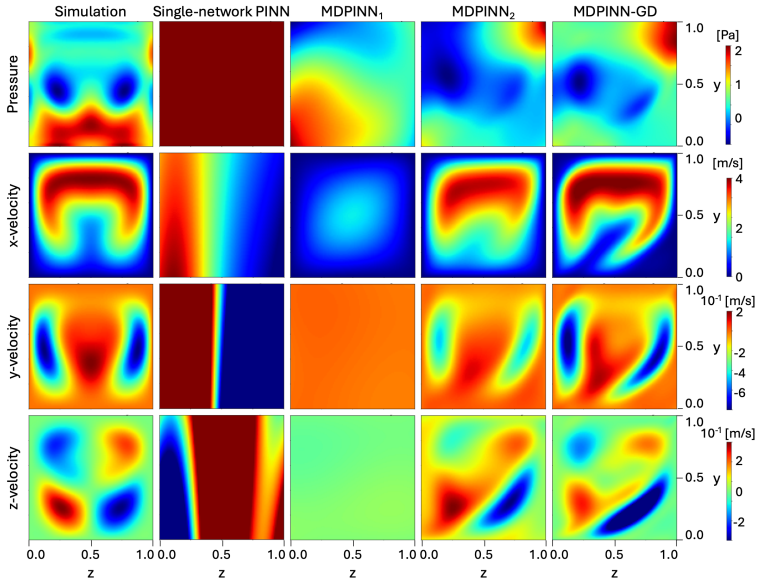

Figure A.2: Comparison of pressure, x-velocity, y-velocity, and z-velocity fields between the simulation and PINN models at outlet of subdomain 5, viewed in the yz-plane.

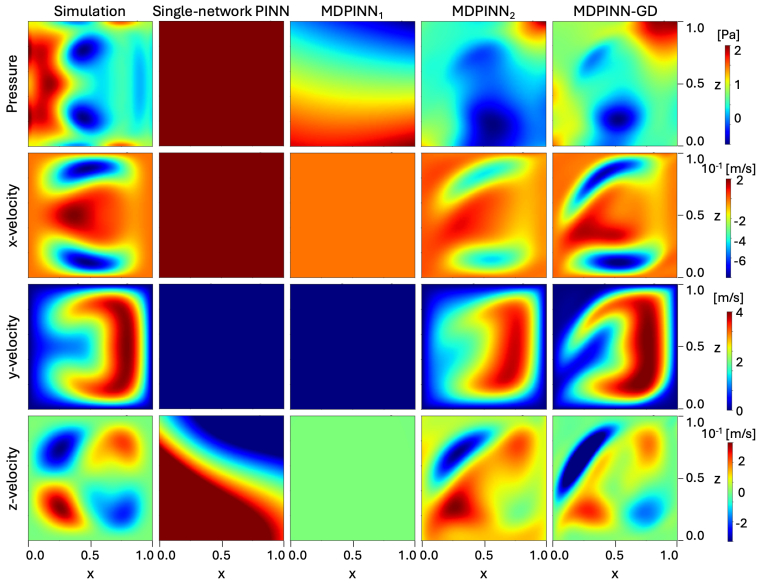

Figure A.3: Comparison of pressure, x-velocity, y-velocity, and z-velocity fields between the simulation and PINN models at outlet of subdomain 9, viewed in the xz-plane.

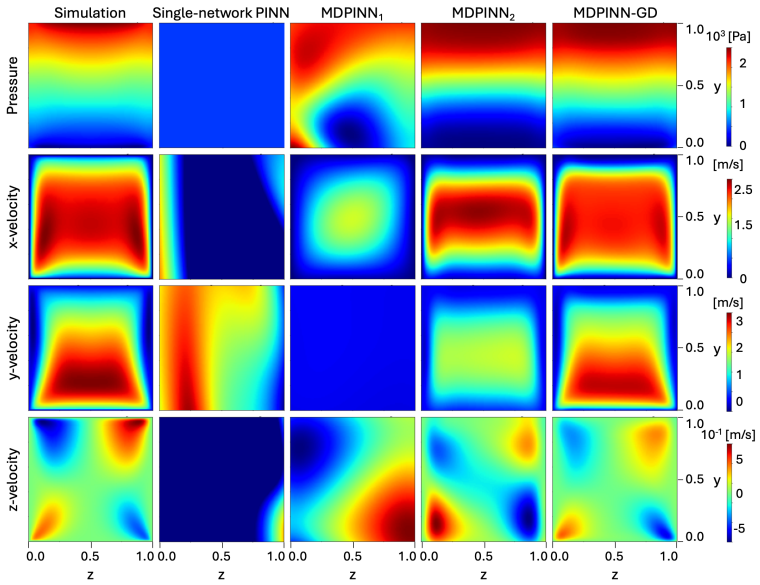

Figure A.4: Comparison of pressure, x-velocity, y-velocity, and z-velocity fields between the simulation and PINN models at inlet of subdomain 4 viewing the yz-plane.

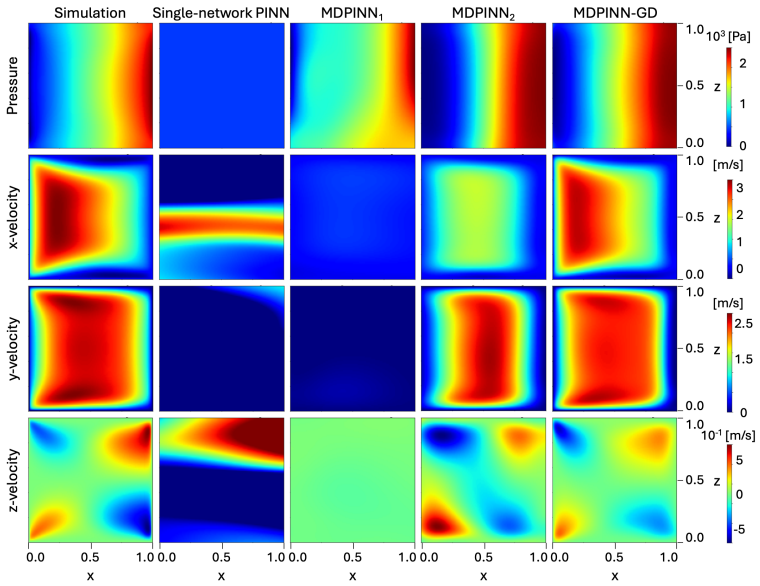

Figure A.5: Comparison of pressure, x-velocity, y-velocity, and z-velocity fields between the simulation and PINN models at inlet of subdomain 8, viewed in the xz-plane.

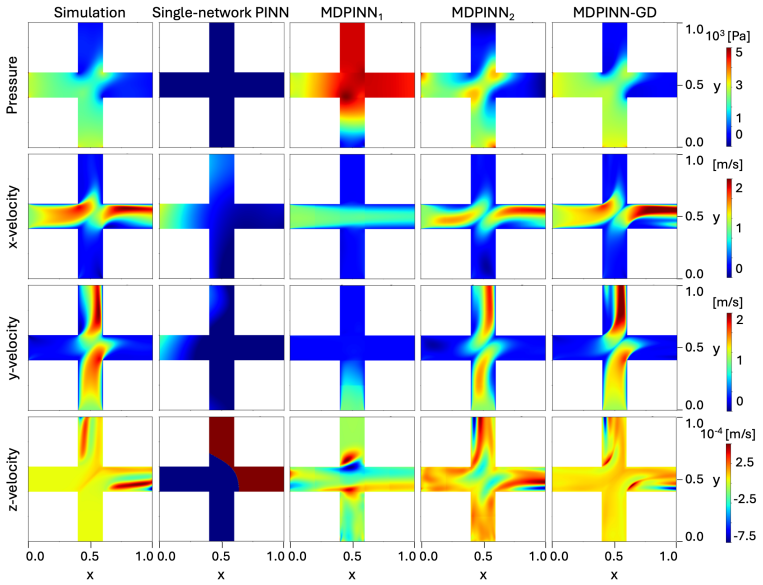

Figure A.6: Comparison of pressure, x-velocity, y-velocity, and z-velocity fields between the simulation and PINN models at the center of the cross-shaped channel, viewed in the xy-plane.

A.4 Loss plot

To understand the training dynamics, the evolution of loss components for MDPINN-GD is shown in Fig. A.7. In Fig. A.7 (a), the total loss gradually decreases, with the most contribution from the boundary loss and one or two orders of magnitude lower contributions from the PDE loss and global continuity loss. Notably, the global continuity loss decreases rapidly by several orders of magnitude from around 1,000 epochs among the 100,000 epoch and continues to drop below 10^{-8}, indicating that the PINN effectively enforces inter-network continuity across subdomains. Meanwhile, the boundary loss decreases more slowly and remains the dominant contributor to the total loss, suggesting that enforcing boundary conditions in as hard constraints may be necessary. Fig. A.7 (b) further decomposes the PDE loss into its physical components. Initially, all components exhibit rapid error reduction, followed by more gradual convergence. Interestingly, the z-momentum loss reaches lower final values compared to the other losses, reflecting the geometric anisotropy of the domain, where flow variations in the z-direction are less pronounced, leading to faster convergence in that component.

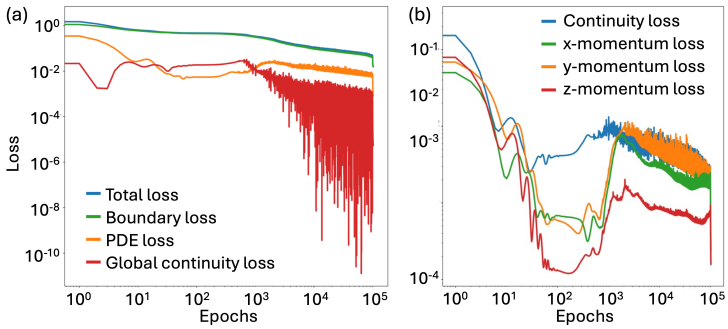

Figure A.7: Loss plots of MDPINN-GD: (a) high-level loss terms: total, boundary condition, PDE, and global continuity loss, (b) PDE loss terms: continuity, x-momentum, y-momentum, and z-momentum.

A.5 Model error comparison

Table A.1: Mean and max absolute errors for all PINN models. MDPINN models are averaged across all subdomains. Overall mean errors are averaged across all fields and percentage improvement of MDPINN-GD are compared against other models.

Mean errors		MDPINN-GD	Single-network PINN	MDPINN_1	MDPINN_2
ϵ (P)	[kPa]	0.62	3.41	3.55	1.76
ϵ (U)	[m/s]	0.14	0.52	0.7	0.33
ϵ (V)	[m/s]	0.14	0.52	0.96	0.33
ϵ (W)	[m/s]	0.052	0.18	0.12	0.076
ϵ_{max} (P)	[kPa]	1.63	23.6	8.11	3.4
ϵ_{max} (U)	[m/s]	0.3	2.16	1.52	0.66
ϵ_{max} (V)	[m/s]	0.3	2.16	2	0.68
ϵ_{max} (W)	[m/s]	0.08	1.79	0.18	0.12
MDPINN-GD mean ϵ improvement			74.8%	76.2%	52.9%
MDPINN-GD mean ϵ_{max} improvement			90.2%	75.2%	49%

A.6 MDPINN-GD error

Comparison of MDPINN-GD predictions to the CFD groundtruth with absolute error fields for various views which include: left and bottom inlets of subdomain 3, right and top outlets of subdomain 3, outlet of subdomains 5 and 9, and xy-plane view at $z = 0.5$ (mid-plane).

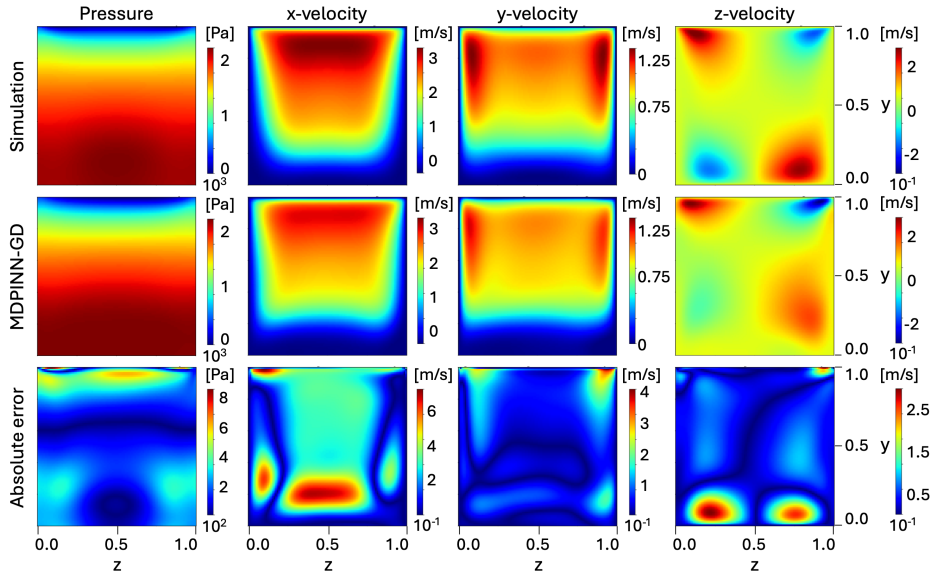

Figure A.8: Pressure, x-velocity, y-velocity, and z-velocity field comparisons of CFD simulation and MDPINN-GD at left inlet of subdomain 3 with absolute error field viewing the yz-plane.

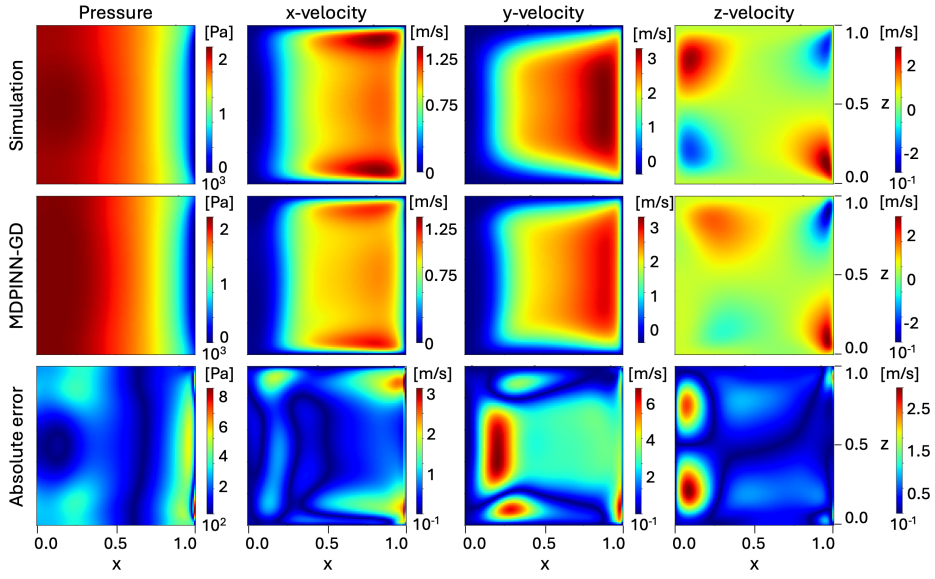

Figure A.9: Pressure, x-velocity, y-velocity, and z-velocity field comparisons of CFD simulation and MDPINN-GD at bottom inlet of subdomain 3 with absolute error field viewing the xz-plane.

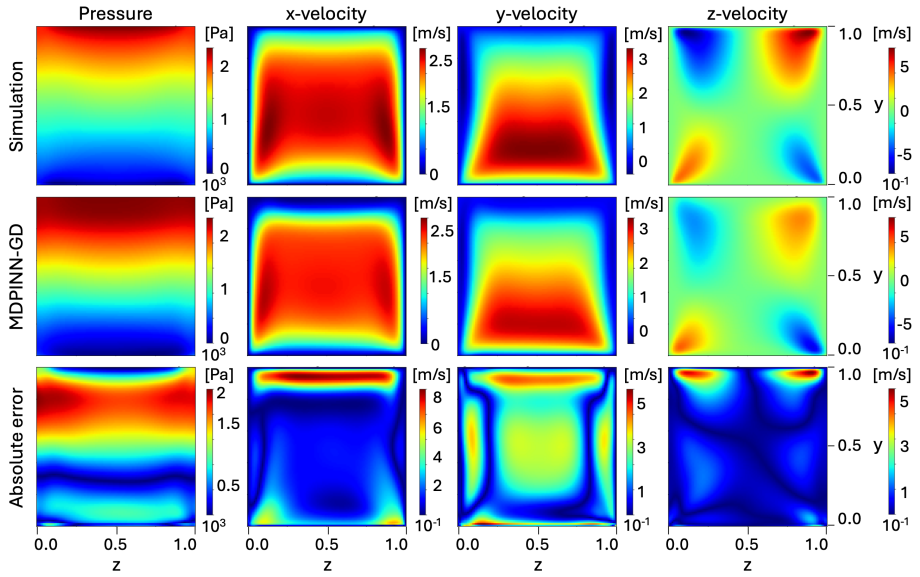

Figure A.10: Pressure, x-velocity, y-velocity, and z-velocity field comparisons of CFD simulation and MDPINN-GD at right outlet of subdomain 3 with absolute error field viewing the yz-plane.

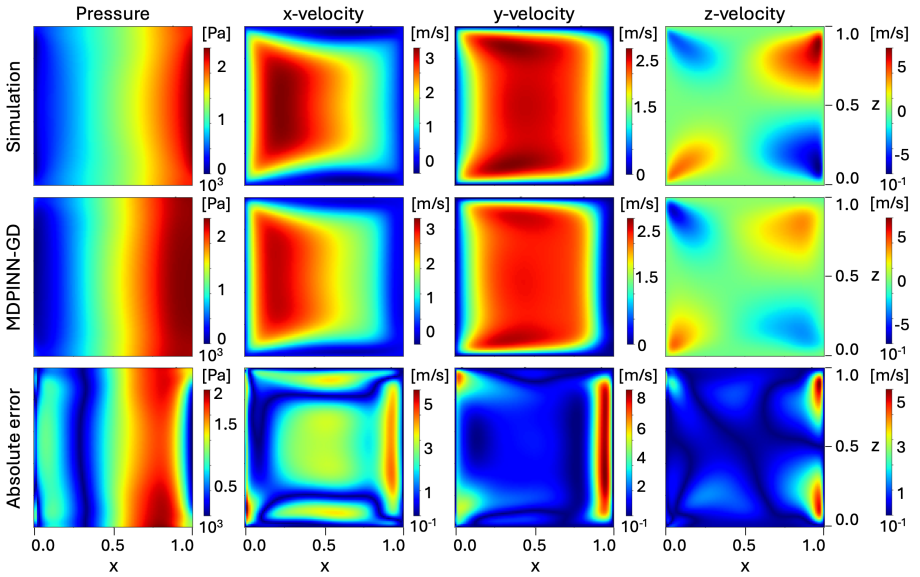

Figure A.11: Pressure, x-velocity, y-velocity, and z-velocity field comparisons of CFD simulation and MDPINN-GD at top outlet of subdomain 3 with absolute error field viewing the xz-plane.

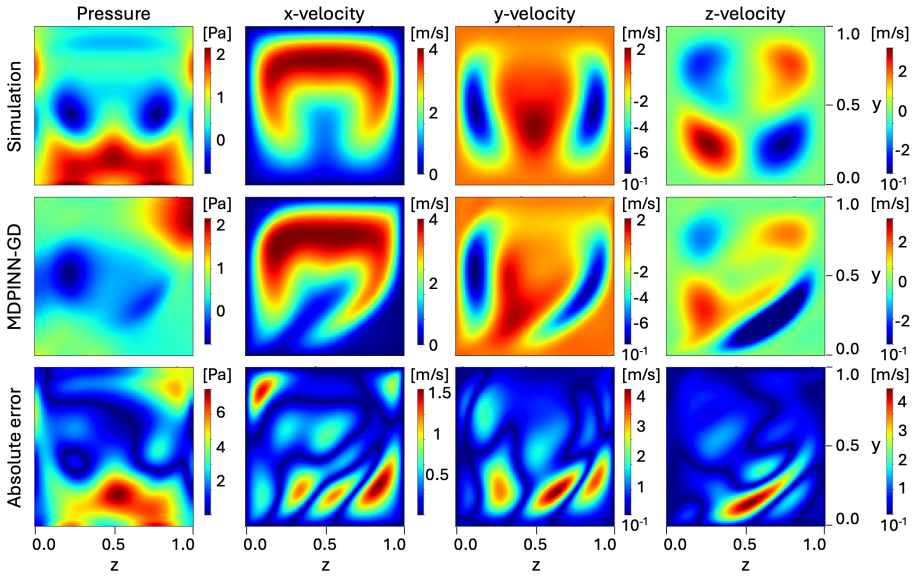

Figure A.12: Pressure, x-velocity, y-velocity, and z-velocity field comparisons of CFD simulation and MDPINN-GD at outlet of subdomain 5 with absolute error field viewing the yz-plane.

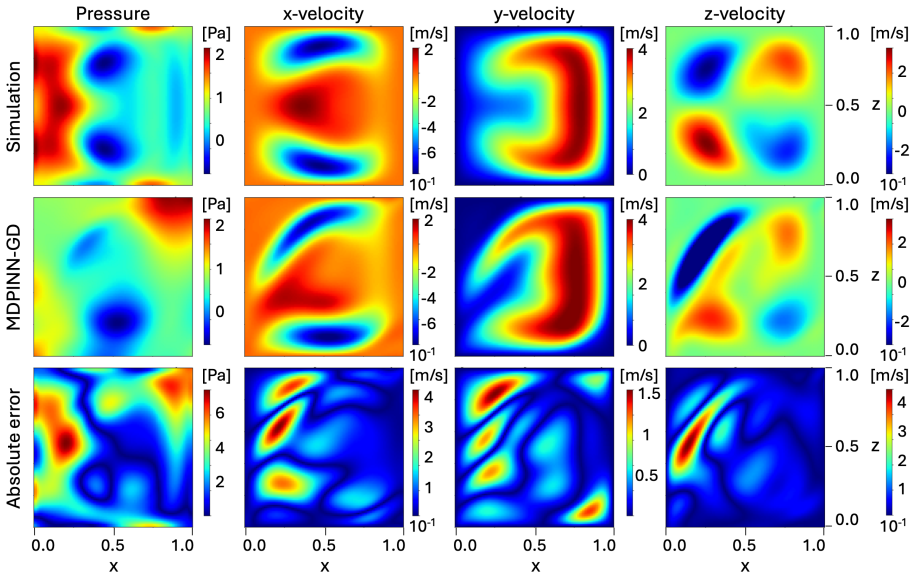

Figure A.13: Pressure, x-velocity, y-velocity, and z-velocity field comparisons of CFD simulation and MDPINN-GD at outlet of subdomain 9 with absolute error field viewing the xz-plane.

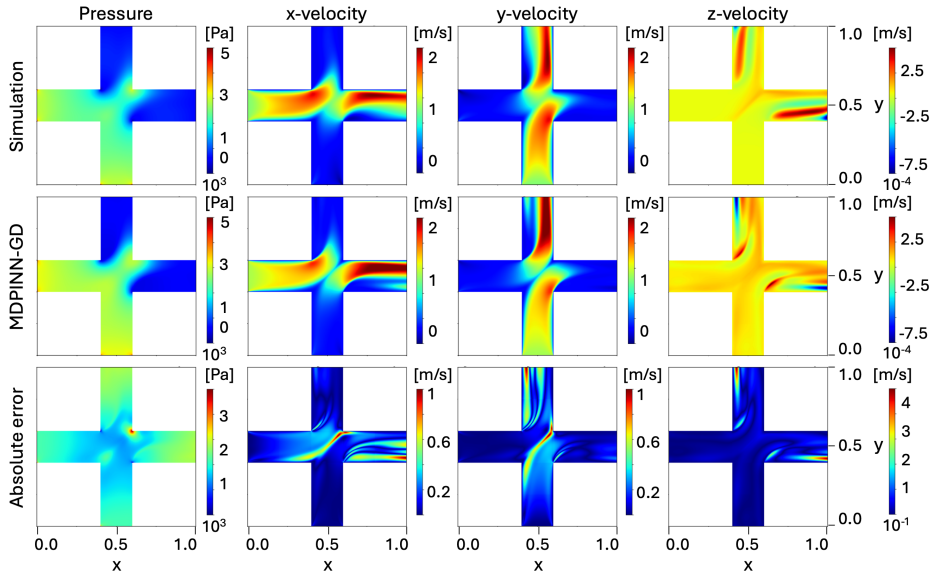

Figure A.14: Pressure, x-velocity, y-velocity, and z-velocity field comparisons of CFD simulation and MDPINN-GD at center of cross-shaped channel with absolute error field viewing the xy-plane.

A.7 MDPINN-GD normalized absolute error

Table A.2: Subdomain mean and max absolute errors for MDPINN-GD with **bold** indicating maximum across each row. Values are given as a percentage of reference inlet velocity (U_0) and reference inlet dynamic pressure ($\rho_0 U_0^2$).

	Left inlet		Intersect	Right outlet		Bottom inlet		Top outlet	
Subdomain number	1	2	3	4	5	6	7	8	9
$\%\epsilon$ (P)	4.86	2.59	**40.83**	34.7	5.36	8.26	2.81	36.2	5.62
$\%\epsilon$ (U)	4.82	9.43	10.7	12.1	**14.9**	0.98	1.31	5.86	4.07
$\%\epsilon$ (V)	0.92	1.27	10.7	5.83	4.10	4.92	9.48	12.18	**14.8**
$\%\epsilon$ (W)	1.39	1.73	**3.99**	3.43	3.13	1.46	1.72	3.45	3.07
$\%\epsilon_{\max}$ (P)	185.4	30.7	63.6	55.6	12.3	**212.6**	25.5	58.0	12.4
$\%\epsilon_{\max}$ (U)	75.6	35.1	52.1	70.5	**76.8**	19.5	15.9	34.7	22.1
$\%\epsilon_{\max}$ (V)	21.6	20.7	52.0	33.8	22.2	**85.9**	34.5	73.5	80.1
$\%\epsilon_{\max}$ (W)	21.3	14.8	**31.9**	27.4	22.3	20.7	15.2	27.4	22.6

A.8 Loss weight ratio tuning

The loss weight ratio between boundary loss and PDE loss were explored for the optimal MDPINN-GD result. The x-velocity field predictions at different views are shown in Fig. A.15, in which $10\times$ PDE loss, $20\times$ PDE loss, and $10\times$ boundary loss were compared against MDPINN-GD (uniform loss). It is evident that uniform loss is optimal in this application.

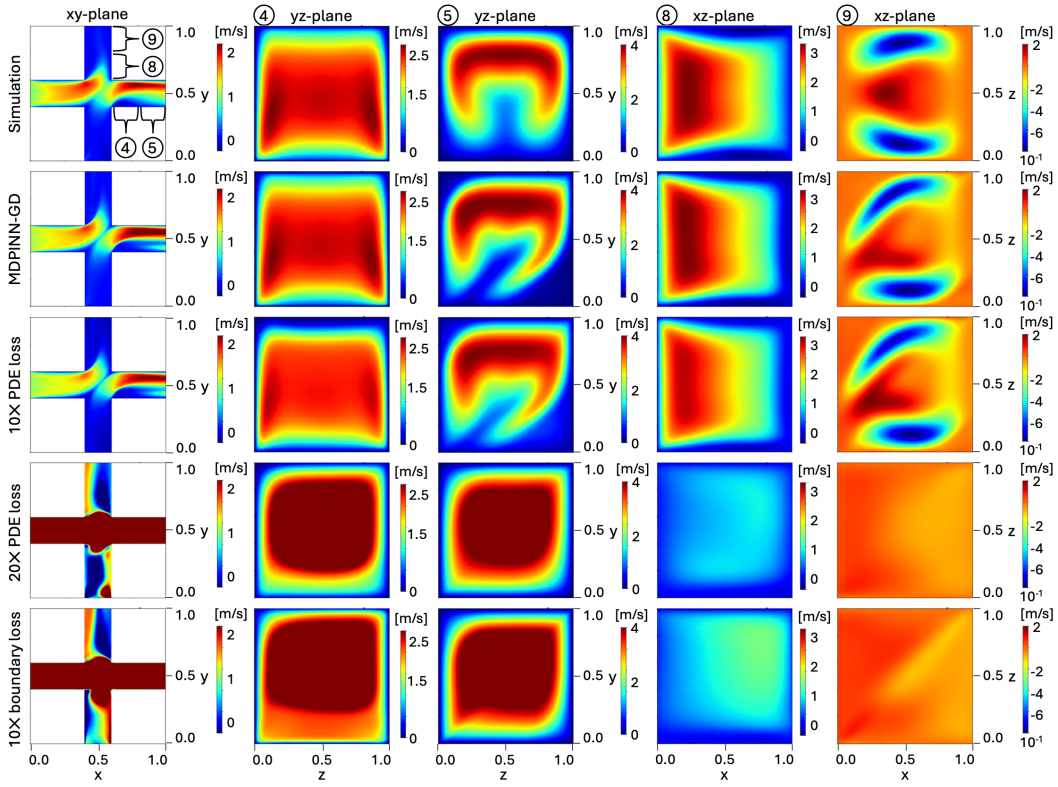

Figure A.15: Comparison of x-velocities between CFD simulation, MDPINN-GD, and MDPINN-GD with different loss ratios with absolute errors. Viewed at: xy-plane at $z = 0.5$ (mid-plane), yz-plane at inlet of subdomain 4, yz-plane at outlet of subdomain 5, xz-plane at inlet of subdomain 8, and xz-plane at outlet of subdomain 9.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims made by the authors include: 1) MDPINN-GD significantly improves fluid prediction results, 2) global continuity loss is necessary for accurate prediction results, and 3) the method demonstrated is applicable to systems with large aspect ratios. These claims have been explored and reflected upon throughout this paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The main limitations of the paper include: 1) MDPINN-GD z-velocity prediction inaccuracies, and 2) demonstrations on other geometries. These limitations have been addressed in the conclusion section with a limitations subsection.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: This paper does not contain any theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: All settings for the simulation and MDPINN-GD network are provided in Section 2: Problem setup under 2.1 and 3. These details are necessary for reproducing the main experimental results. A pseudocode is also provided in the Appendix A.2 for further clarifications.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: We utilized a well-established code for PINN, already available from the original contributors. Our primary contribution lies in adopting multiple PINNs with more robust loss terms. This does not require the sharing of code. Instead, pseudocode is provided in the Appendix A.2 for reproducibility.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: All settings for the simulation and MDPINN-GD network are provided in Section 2.1 and Section 3.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: This paper presents the mean and max values of mean absolute errors. Normalization is also done with respect to the inlet velocity to show the significance of the change in error. These are presented in the Section 4, with additional material provided in the Appendix A.5 and Appendix A.7.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper has indicated the type of GPU, computation time, and memory usage for the experiments. These are provided in Section 4.3.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This paper conforms to all aspects of the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: The Section 1 and Section 5 discusses the positive societal impacts of MDPINN-GD. Negative societal impacts of the proposed network architecture are not anticipated as experiments do not involve sensitive data or information and the proposed methodologies complies with law of physics.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The work presented in this paper poses no risk to misuse.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: References to the previous PINNs and multi-domain methods are properly cited. The use of PyTorch has been properly disclosed. Simulation data used has been generated by the authors, with a disclosure made regarding the use of Ansys Fluent.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: This paper does not release new assets. The framework of MDPINN-GD is well documented throughout the paper and can be referenced without license.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This research and the corresponding paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: This research and the corresponding paper does not involve crowdsourcing nor research with human subjects. No IRB approvals were seeked/needed.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: This research does not involve LLMs as any important, original, or non-standard components. LLM was only used for writing, editing or formatting purposes.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.