Discrete Residual Loss Functions for Training Physics-Informed Neural Networks

J Rishi, Sumanth Kumar, Azhar Gafoor, and Deepak Subramani

Department of Computational and Data Sciences, Indian Institute of Science - Bengaluru, India
{rishij,ksumanth,azharctp,deepakns}@iisc.ac.in

Abstract. The use of neural networks and operators to solve partial differential equations that govern fluid flow is carried out in a simulation-free, physics-informed approach, where the residual of the governing equation calculated via automatic differentiation across the neural network is the loss function used for training. One issue with this approach is that simulating highly nonlinear flows such as high Reynolds number flows is challenging, even in laminar settings. We propose a new simulation-free approach for training PINNs using a residual loss based on a discrete numerical scheme instead of automatic differentiation. This loss function also requires a new grid-based PINNs training strategy. Using the loss landscape, we demonstrate why our new loss function works better than the automatic differentiation-based loss function. We also demonstrate how to implement our grid-based training for complex geometry. Simulations using the new neural model for high Reynolds number fluid flow and complex geometry test cases are showcased and compared with automatic differentiation approaches. The results show that our new discrete loss function and training strategy take less computational time, converge faster than automatic differentiation, and can be used to simulate nonlinear flows efficiently.

Keywords: PINNs · Finite Volume Discrete · Automatic Differentiation.

1 Introduction

Simulating dynamical systems governed by partial differential equations (PDEs) is a major part of engineering design and practice. These simulations are typically performed on a computer using computational methods to solve the governing equations [1]. Recently, deep neural models, called Physics-Informed Neural Networks (PINNs) and Neural Operators (PINO), have been developed to solve PDEs in a semi-supervised manner and have been used in various applications [24,21,30,11,8,4]. The training method is semi-supervised because the loss function is a combination of mean square error of initial/boundary data and the residual of the governing PDEs.

The main innovation of PINNs is the use of Automatic Differentiation (AD; [3,2,34,23])to calculate the residual of PDE. The training is simulation-free and

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_13

does not require the use of expensive computational solvers to obtain training data. However, this process is computationally expensive; calculating the loss function itself requires the calculation of the first and second derivatives of the output (e.g., velocity in a fluid flow problem) with respect to the input (e.g., spatial and temporal coordinates). Furthermore, PINNs are computationally intensive, requiring numerous training epochs with two automatic differentiation operations per epoch, one for loss calculation and another for backpropagation. This algorithm increases the computational time and resources required to train PINNs for practical applications. Thus, new research is required to reduce computation requirements.

In the present paper, we develop a new discrete loss function for training PINNs in a semi-supervised manner. We show that a PINN model trained using a discrete loss function converges in fewer epochs and less time. Here, we develop and use a finite-volume discrete loss function, shortened as FVD loss. However, other discretization schemes could also be utilized. Theoretically, we demonstrate why it works better compared to AD, and experimentally, we demonstrate that the FVD loss performs better than automatic differentiation in simulating highly nonlinear fluid flow and how to solve for complex geometry cases.

Previously, a few papers have proposed the use of a different residual loss function along with automatic differentiation. VPINN [12] is a finite element based method that establishes the loss function by integrating the residue of PDEs multiplied by a set of basis, hp-VPINN [13] uses the same idea as VPINNs, but the basis functions only have local supports, RVPINNs [27] uses quadratic loss functionsl in terms of petrov-galerkin-type variational formulation for PDE, CVRPINNs [19] accelarates implementation of RVPINN, [5] employs least squares functionals as loss function, can-pinn [6] combines automatic difference with finite difference, and other papers [18,7,5,29,16,10] that use different types of residual losses.

The objective of this paper is to theoretically elucidate the reasons why this residual loss function results in reduced computational time and demonstrates faster convergence, supported by experimental results utilizing the FVD loss. Furthermore, we present experimental evidence indicating that these discrete PINNs can achieve accurate results for high Reynolds number scenarios, and we illustrate the application of discrete loss functions for complex geometrical configurations.

The remainder of this paper is as follows. First, we provide a background of PINNs; then, we discuss the FVD loss function along with the grid-based training procedure. In a theoretical context, we demonstrate the advantages of discrete PINNs over traditional PINNs, supported by simulation results that evaluate both of these loss functions. Finally, we show that this development helps simulate high Reynolds-number fluid flows and complex geometries test cases.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_13

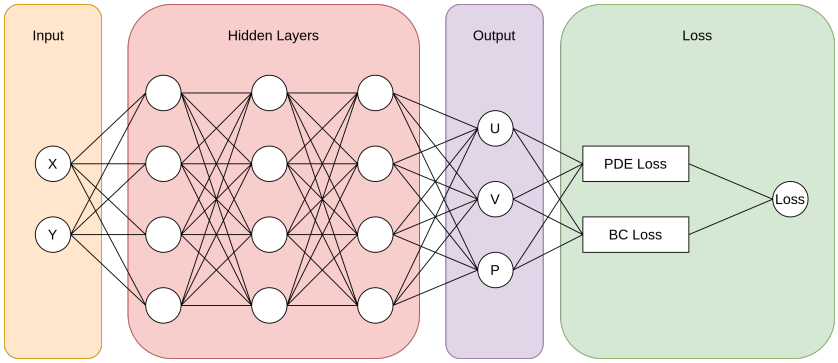

Fig. 1. Schematic diagram of generic PINNs

1.1 Physics Informed Neural Networks: Preliminaries

Consider a generic partial differential equation (PDE) with appropriate initial and boundary conditions,

$$f(x, u, \frac{\partial u}{\partial x}, ...) = 0\,, x \in \Omega\,,$$
$$h(u(x)) = 0\,, x \in \partial\Omega\,,$$

where f is a generic nonlinear function, $x \in \Omega$ is the coordinate in the domain Ω, $\partial\Omega$ is the boundary of the domain, h is the boundary condition function and $u \in \mathcal{U}$ is the solution field in a solution space that satisfies the PDE above. As a specific example, for 2D fluid flow problems, the vector u consists of the flow velocities (horizontal, vertical) and pressure. The domain coordinates consist of x, y, t. For steady problems, the coordinates are x, y.

The Physics-informed Neural Network is a map of the domain to the solution, i.e., $u = \mathcal{H}(x; \theta)$, where $\mathcal{H}$ is a parametrized neural network with parameters θ (Fig. 1).

The PINNs loss comprises a PDE-residual and boundary data loss terms given by

$$L_{PINN} = L_{PDE} + L_{BL}$$
$$L_{PDE} = \frac{1}{N_c}\left[\sum_{i=1}^{N_c}(f(x, u, \frac{\partial u}{\partial x}, ...))^2\right]$$
$$L_{BL} = \frac{1}{N_{bp}}\left[\sum_{i=1}^{N_{bp}}(h(u(x))^2\right]\,,$$

where N_c is the total number of domain collocation points and N_{bp} is the total number of boundary points. Automatic Differentiation evaluates the PDE loss term, and the boundary data loss term is calculated as the mean squared error.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_13

The use of automatic differentiation allows us to solve the PDE without relying on simulation data from numerical solvers, as is done in several simulation-based neural operator methods such as DeepONets [20], and Fourier Neural Operators [17].

2 Finite Volume Discrete Loss for PDE Residual

Instead of using automatic differentiation, we use finite volume discrete(FVD) loss function to evaluate the PDE residual. The FVD loss can be derived for a generic PDE, but for ease of demonstration, we derive the FVD loss for the 2D steady incompressible Navier-Stokes equation [22]. The 2D steady-state Navier-Stokes is given by:

$$(\vec{V}.\nabla)\vec{V} = -\nabla p + \frac{1}{Re}(\nabla^2\vec{V}) \tag{1}$$

$$\nabla \cdot \vec{V} = 0 \tag{2}$$

where $\vec{V}$ is the velocity vector, p is the pressure, Re is the Reynold number, ∇ is the first spatial derivative gradient operator and ∇^2 is the second spatial derivative Laplace operator. We express the above momentum and continuity equations using a finite-volume discretization scheme on a uniform grid (Figure 2). $(\vec{V}.\nabla)\vec{V}$ is called the advection term, and $\nabla^2\vec{V}$ is called the diffusion term.

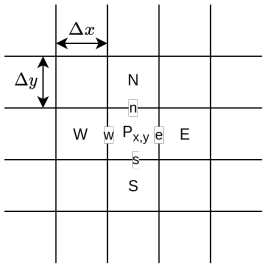

Fig. 2. Schematic of the grid used for deriving the FVD loss

The upwind scheme is used for the advection terms:

$$\hat{\phi}_e = \begin{cases} \phi_P & \text{if } u_E \geq 0 \\ \phi_E & \text{if } u_E < 0 \end{cases} \qquad \hat{\phi}_n = \begin{cases} \phi_P & \text{if } u_N \geq 0 \\ \phi_N & \text{if } u_N < 0 \end{cases}$$

$$\hat{\phi}_s = \begin{cases} \phi_S & \text{if } u_S \geq 0 \\ \phi_P & \text{if } u_S < 0 \end{cases} \qquad \hat{\phi}_w = \begin{cases} \phi_W & \text{if } u_W \geq 0 \\ \phi_P & \text{if } u_W < 0 \end{cases}$$

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_13

where ϕ is a general scalar.

The gradient operator is approximated using the central difference scheme as

$$\begin{aligned} \left.\frac{\partial \phi}{\partial x}\right|_x &= \frac{\phi(x+\Delta x) - \phi(x-\Delta x)}{2\Delta x} \\ \left.\frac{\partial \phi}{\partial y}\right|_y &= \frac{\phi(y+\Delta y) - \phi(y-\Delta y)}{2\Delta y} \end{aligned} \tag{3}$$

The Laplace operator is approximated as

$$\begin{aligned} \left.\frac{\partial^2 \phi}{\partial^2 x}\right|_x &= \frac{\phi(x+\Delta x, y) - 2\phi(x,y) + \phi(x-\Delta x, y)}{\Delta x^2} \\ \left.\frac{\partial^2 \phi}{\partial^2 y}\right|_y &= \frac{\phi(x, y+\Delta y) - 2\phi(x,y) + \phi(x, y-\Delta y)}{\Delta y^2} \end{aligned} \tag{4}$$

Residual loss is calculated as the mean of the square of the finite volume discrete residuals at all interior points.

2.1 Grid-based training procedure

Vanilla PINN models are trained by giving $\{X_{bp}^i\}_{j=1}^{N_{bp}}$ as boundary points and $\{X_c^j\}_{j=1}^{N_c}$ as domain collocation points and loss is calculated as:

$$Loss \quad = \quad \frac{1}{N_c}\left[\sum_{i=1}^{N_c}(f(\{X_{bp}^i\}, u, \frac{\partial u}{\partial x}, ...))^2\right] \quad + \quad \frac{1}{N_{bp}}\left[\sum_{j=1}^{N_{bp}}(h(u(\{X_c^j\}))^2\right] \tag{5}$$

In PINN training, Latin hypercube sampling (LHS) is often used to generate diverse and evenly distributed input samples, promoting better generalization and reducing clustering risk. However, LHS is unsuitable for use with the FVD loss. Calculating FVD loss requires mini-batch training that includes all grid points at once, that is $N_c = N_x * N_y$, where N_x, N_y represents total grid points along the x and y axes. This study uses a rectangular grid and loss function as described in Sec 2.

3 Finite volume discrete vs AD loss

In this section, we will compare computational graphs and loss function plots for FVD and AD, as these are key components used in the analysis of neural network training. We will first examine the computational graph and then proceed to the loss function.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_13

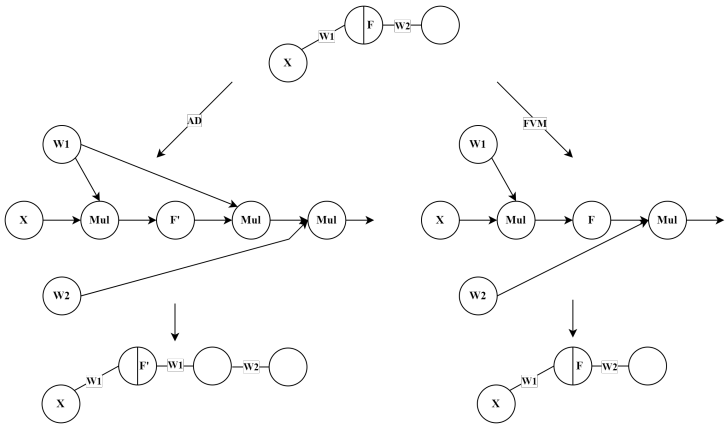

Fig. 3. Computational graph for AD and FVD loss functions

3.1 Computational time

Consider a PDE with only the first derivative term that is being trained on a neural network with two trainable parameters, as shown in Fig. 3. The loss functions for the PINN with AD and FVD loss approaches with activation function F using an input x and producing an output u are given by:

$$\begin{aligned} Loss_{AD} &= \frac{\partial u}{\partial x} \\ &= w_2 w_1 F'(w_1 x) \\ Loss_{FVD} &= \frac{u(x+\Delta x) - u(x-\Delta x)}{2\Delta x} \\ &= \frac{w_2 F(w_1(x+\Delta x)) - w_2 F(w_1(x-\Delta x))}{2\Delta x} \end{aligned}$$

The computational graphs for the above two loss functions are shown in Fig. 3. The figure indicates that in order to calculate loss, the AD method requires a deeper network compared to FVD loss. This suggests a higher computational requirement for AD. We have demonstrated that even when considering w_1 and w_2 as scalars, the AD loss incurs higher computational cost compared to the FVD loss. This difference in computational efficiency remains even when w_1 and w_2 are treated as vectors, as the AD loss involves a greater number of matrix multiplications compared to the FVD loss.

3.2 Faster Convergence

Empirical evidence suggests that the use of the FVD loss in training results in accelerated performance compared to AD. In this study, we investigate the

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_13

underlying reasons for this observation. The primary alteration in the loss function pertains to the residual term, while the boundary term remains constant; therefore, our focus will be on the analysis of the residual term exclusively. The residual loss gradient has been shown to be significantly higher than the boundary loss gradient, as indicated in [33]. This allows for a comparative evaluation of the residual terms associated with both loss functions. For the purpose of conducting an analysis on both residual terms, the one-dimensional Laplace equation will be utilized as a test case.

$$\frac{\partial^2 u}{\partial x^2} = 0$$

To examine the training dynamics of FVD and AD loss, we investigate a network architecture comprising two neurons and two trainable parameters w_1, w_2. The network is structured with a single layer, as shown in Figure 4. For the sake of simplification and ease of analysis, the weights on all other edges are fixed at one. The Swish activation function is used for all hidden neurons [25].

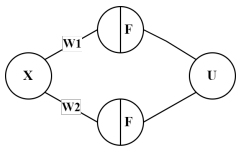

Fig. 4. Neural network of 2 parameters

An input (x) gives the network output (U) as follows.

$$U_{output} = \frac{w_1 x}{1 + e^{(-w_1 x)}} + \frac{w_2 x}{1 + e^{(-w_2 x)}} \tag{6}$$

Now, we will consider the neural network shown in Figure 4 and PDE loss in the 1D Laplace equation.(For FVD loss, Eq: (4) is considered), we discrad the third and fourth order terms in AD and finally we obtain:

$$\begin{aligned} \mathcal{L}_{AD} &\approx \frac{w_1^2 e^{-w_1 x}}{(1 + e^{-w_1 x})^2} + \frac{w_2^2 e^{-w_2 x}}{(1 + e^{-w_2 x})^2} \\ \mathcal{L}_{FVD} &\approx \frac{w_1 \Delta x}{1 + e^{-w_1 (x + \Delta x)}} - \frac{w_1 \Delta x}{1 + e^{-w_1 (x - \Delta x)}} \\ &+ \frac{w_2 \Delta x}{1 + e^{-w_2 (x + \Delta x)}} - \frac{w_2 \Delta x}{1 + e^{-w_2 (x - \Delta x)}} \end{aligned}$$

The efficiency of learning for a neural network through backpropagation depends on the characteristics of the loss function. The residual loss function is

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_13

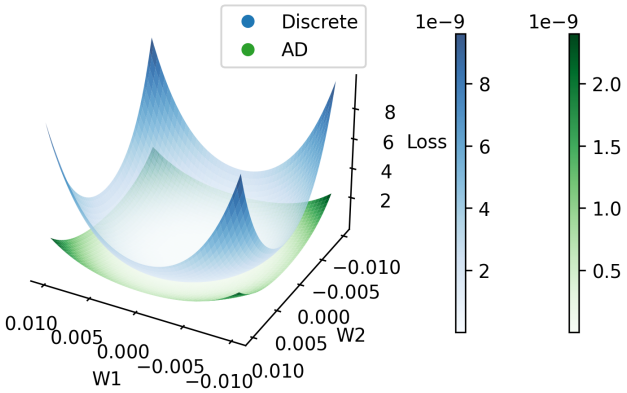

Fig. 5. A comparison of squared Poisson equation Loss function between discrete and automatic differentiation for the neural network shown in Figure 4

plotted with respect to the parameters. Figure 5 presents the loss for a fixed x as a function of w_1 and w_2.

The squared output of the AD network exhibits more plateau regions compared to the FVD network. This causes gradient descent-based optimization algorithms to take longer to converge when using the AD loss compared to the FVD loss. Consequently, FVD loss demonstrates superior learning performance for PINNs. [15,9].

4 Experiments and Results

We conducted three experiments to study the advantage of using FVD loss instead of automatic differentiation loss. The test cases chosen are Kovasznay flow, steady-state lid-driven cavity flow, and flow past a cylinder flow. All PINN simulations are compared with the solution obtained from a numerical CFD solver [31].

4.1 Accelerating Training with FVD Loss

The Kovasznay flow [14,32] is governed by the steady state Navier Stokes equations and has an analytical solution given by

$$\begin{aligned}
\zeta &= \frac{0.5}{\mu} - \sqrt{\frac{1}{4\mu^2} + 4\pi^2}\,, \\
u &= 1 - \exp(\zeta x)\cos(2\pi y)\,, \\
v &= \frac{\zeta}{2\pi}\exp(\zeta x)\sin(2\pi y)\,, \\
p &= 0.5(1 - \exp(2\zeta x))\,,
\end{aligned}$$

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_13

Table 1. Network hyperparameters used in experiments

Hyperparameters			
Flows	Kovasznay flow	Lid-driven cavity	Flow past a cylinder
Layers	5	5	5
Neurons per layer	50	300	32
Learning rate	0.001	0.001	0.0005
Epochs	2000	50000	50000
Optimizer	Adam		
Grid points	64×64	100×100	256×256
Activation function	swish		
Initializer	Glorot Uniform	Glorot Uniform	Glorot Normal

where μ is the viscosity, x and y are the cartesian coordinate system, u and v are horizontal and vertical components of the velocity, and p is pressure. The boundary condition of the Navier-Stokes equation is given by the analytical solution evaluated at the boundary points.

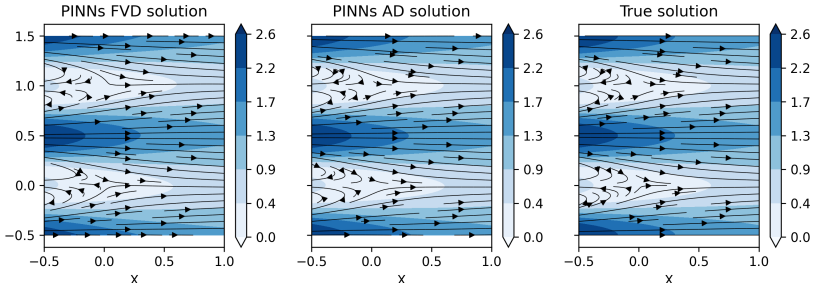

Fig. 6. The solution of PINNs trained with FVD, AD is shown next to the true solution of Kovazsnay flow with Re=40. Flow streamlines are overlaid on a background of velocity magnitude.

We solve the Kovasznay flow using the PINN approach in a square grid with $x \in [-0.5, 1.0]$, $y \in [-0.5, 1.5]$ and $\mu = 0.025$. The loss function consists of the residual of the Navier Stokes and the boundary loss and other hyperparameters chosen are given in Table 1[26].

Table 2. Relative error in u and v with their confidence interval of Kovasznay flow.

Relative error		
Loss	U Error	V Error
AD	0.025±0.0004	0.006±0.00003
FVD	0.018±0.0002	0.009±0.00007

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_13

The training time for 2000 epochs is 1.49 minutes for the FVD loss, while it is 2.52 minutes for the AD loss on NVIDIA RTX A6000. This suggests that the training time is shorter for the FVD loss function compared to the AD loss function. Table 2 presents relative error metrics for this experiment. The results demonstrate that the FVD and AD approaches achieve comparable accuracy in solving the Kovasznay flow problem.

4.2 Accelerating Convergence with FVD Loss: Achieving Faster Results in Fewer Epochs

To demonstrate the convergence of FVD and AD loss, we have considered the lid-driven cavity problem. The right panel of Fig. 7 shows the flow setup. A square fluid-filled cavity in the $x - z$ plane is subject to a horizontal motion on the top surface, corresponding to the lid moving to the right at a specified velocity of 1 non-dimensional unit.

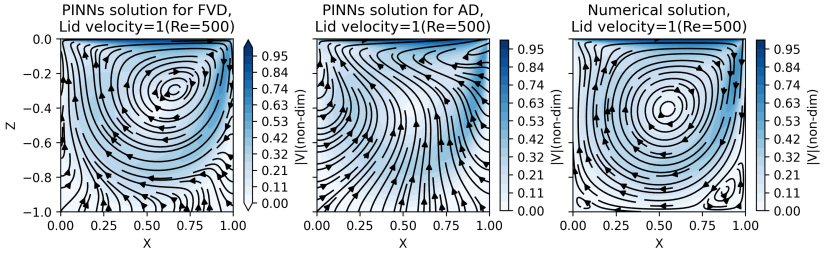

Fig. 7. Flow streamlines are overlaid on a background of velocity magnitude for the lid-driven cavity problem for top-velocity = 1 and Re = 500 for 30000 epochs. The solution of PINNs trained with FVD loss is on the left. PINNs trained with AD-based loss are in the middle. The numerical solution is shown for reference.

The results obtained using AD loss were compared with those from FVD loss. Figure 8 presents the learning curves corresponding to the FVD loss and AD for a network consisting of five hidden layers, depicted by solid red and dotted red lines, respectively. Figure 7 illustrates the PINNs simulation employing the FVD loss and AD-based training alongside the associated numerical CFD solution. Both networks underwent training for 30,000 epochs. At this stage, the FVD simulation aligned with the CFD solution, whereas the AD-based training failed to achieve such concordance. Even after extending the training to 50,000 epochs, the AD-based training did not yield a solution consistent with the CFD solution. To achieve 50,000 epochs of training, the AD network required 30 minutes, which is approximately six times longer than the duration required by the FVD loss to complete the same number of epochs in 5 minutes on the identical GPU. Table 3 presents the tabulated relative error associated with this experiment. The findings suggest that, training using loss based on FVD demonstrates superior effectiveness compared to traditional PINNs trained with AD.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_13

Table 3. Relative error in u and v with their confidence interval for Re=500

Relative error		
Loss	U Error	V Error
AD	0.108±0.006	0.109±0.008
FVD	0.054±0.002	0.051±0.002

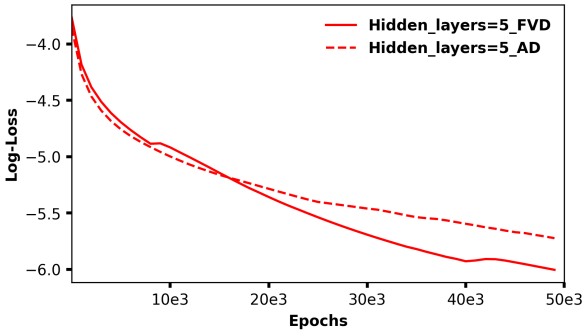

Fig. 8. Loss plot for FVD and AD for Re = 500

4.3 Enhancing Efficiency in High Reynolds Number Simulations: The Effectiveness of FVD Loss

In the present study, PINN with FVD loss was used to simulate fluid flow with a high Reynolds number of Re = 1000 within a lid-driven cavity. The hyper-parameters used in this study are consistent with those presented in Table 1, except for the modifications made to the learning rate and the number of epochs. The training process incorporated an exponential decay learning rate scheduler and spans 100,000 epochs. Figure 9 illustrates the flow streamlines and velocity magnitude based on FVD and AD training. Table 4 presents the tabulated relative error associated with this experiment. The findings suggest that, at high Reynolds numbers, the training using loss based on FVD demonstrates superior effectiveness compared to traditional PINNs trained with AD.

Table 4. Relative error in u and v along with their confidence interval for Re=1000.

Relative error		
Loss	U Error	V Error
AD	0.11±0.013	0.10±0.013
FVD	0.04±0.001	0.04±0.001

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_13

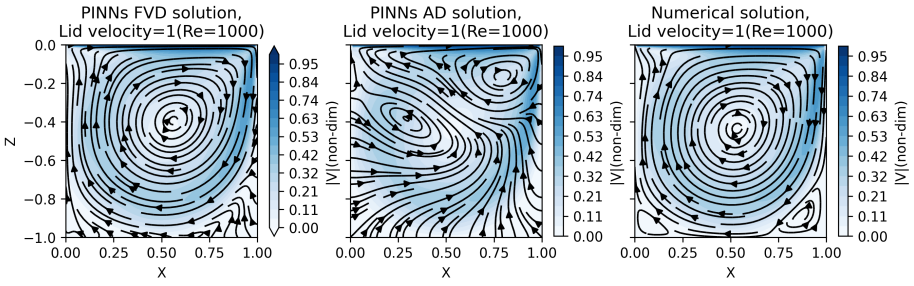

Fig. 9. Flow streamlines are overlaid on a background of velocity magnitude for the lid-driven cavity problem for top-velocity = 1 and Re = 1000. The solution of PINNs trained with FVD loss is on the left. PINNs trained with AD-based loss are in the middle. The numerical solution is shown for reference.

4.4 Complex geometry

Thus far, we have utilized a rectangular grid to implement the FVD loss. For complex geometries, a rectangular grid is unsuitable; alternative approaches are required. To demonstrate FVD loss in complex geometries, we use the Flow Past Cylinder (FPC) scenario, choosing a cylinder of radius=0.5 with dimensions $x = 20$ and $y = 3$, and detail the utility of FVD loss (Figure 10).

The typical method for computing FVD loss in cylinder flow involves computing the residual loss on a rectangular grid that covers the entire domain. The masking operation is then applied to the designated region 1 (see Figure 10), with boundary conditions implemented in a manner similar to that of traditional PINNs. However, this technique produces markedly higher errors near the cylinder, as conditions are not enforced within the cylinder.

We addressed this issue by implementing a geometric loss approach similar to boundary loss, setting loss to zero within the cylinder while applying a 1.5 residual loss weighting around the cylinder (region 2 in Figure 10), with hyperparameters detailed in Table 1. The weights for the boundary loss and the residual loss are considered from this [28]. Figure 11 illustrates that the incorporation of discrete loss, together with our training mechanism, successfully addresses complex geometric scenarios.

5 Conclusion and Future work

Grid-based training mechanism combined with a discrete residual loss of finite volume is used for PINNs. Our results suggest that this method exhibits superior advantages over AD training, as it requires fewer epochs to achieve convergence and reduces computational time. We provide a reasoning that improved the performance of the FVD loss compared to the AD loss. Moreover, we provided evidence that training utilizing the FVD residual loss yields remarkable performance in solving flows with a high Reynolds number. Additionally, we

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_13

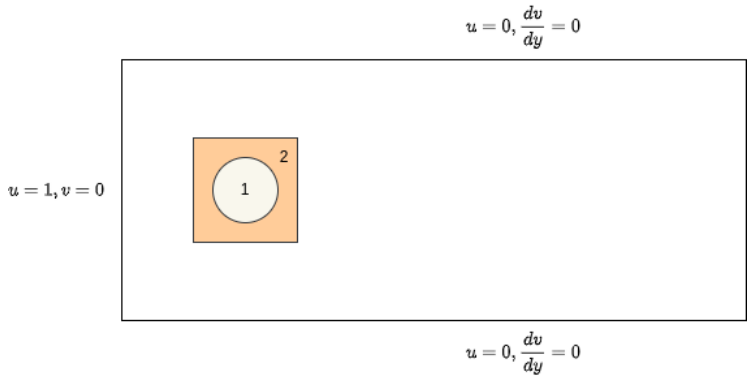

Fig. 10. Schematic diagram of flow past a cylinder. Region 1 denotes a cylinder, while Region 2 indicates an area where the residual loss is scaled.

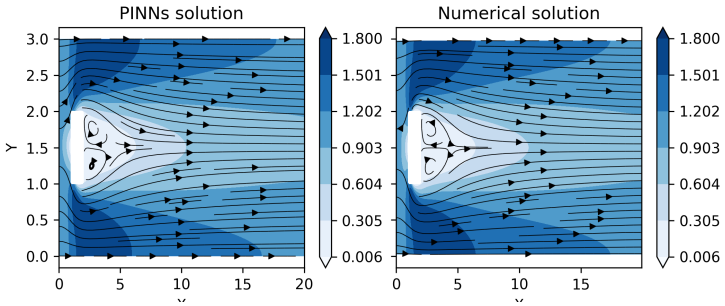

Fig. 11. Flow streamlines are overlaid on a background of velocity magnitude for the flow past a cylinder problem for left-velocity = 1 and Re = 50. The solution of PINNs trained with FVD loss and the numerical solution is shown.

demonstrate the application of FVD loss in scenarios with complex geometries. Future research will evaluate this approach for high Reynolds number flows in complex geometries.

The codes and model weights can be accessed on GitHub[1]

Acknowledgment. This research is partially supported by a research grant from the Ministry of Earth Sciences (MoES/36/OOIS/Extra/84/2022). We are also grateful for the PMRF fellowship support to Azhar Gafoor and TCS PhD Fellowhip to J Rishi.

[1] https://github.com/quest-lab-iisc/PINN-FVD

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_13

References

1. Anderson, J.D., Wendt, J.: Computational fluid dynamics, vol. 206. Springer (1995)
2. Baydin, A.G., Pearlmutter, B.A., Radul, A.A., Siskind, J.M.: Automatic differentiation in machine learning: a survey. Journal of Marchine Learning Research **18**, 1–43 (2018)
3. Berg, J., Nyström, K.: A unified deep artificial neural network approach to partial differential equations in complex geometries. Neurocomputing **317**, 28–41 (2018)
4. Boya, S.K., Subramani, D.: A physics-informed transformer neural operator for learning generalized solutions of initial boundary value problems. arXiv preprint arXiv:2412.09009 (2024)
5. Cai, Z., Chen, J., Liu, M., Liu, X.: Deep least-squares methods: An unsupervised learning-based numerical method for solving elliptic pdes. Journal of Computational Physics **420**, 109707 (2020)
6. Chiu, P.H., Wong, J.C., Ooi, C., Dao, M.H., Ong, Y.S.: Can-pinn: A fast physics-informed neural network based on coupled-automatic–numerical differentiation method. Computer Methods in Applied Mechanics and Engineering **395**, 114909 (2022)
7. Fang, Z.: A high-efficient hybrid physics-informed neural networks based on convolutional neural network. IEEE Transactions on Neural Networks and Learning Systems **33**(10), 5514–5526 (2021)
8. Gafoor CTP, A., Kumar Boya, S., Jinka, R., Gupta, A., Tyagi, A., Sarkar, S., Subramani, D.N.: A physics-informed neural network for turbulent wake simulations behind wind turbines. Physics of Fluids **37**(1) (2025)
9. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics. pp. 249–256. JMLR Workshop and Conference Proceedings (2010)
10. Horuz, C.C., Karlbauer, M., Praditia, T., Butz, M.V., Oladyshkin, S., Nowak, W., Otte, S.: Physical domain reconstruction with finite volume neural networks. Applied Artificial Intelligence **37**(1), 2204261 (2023)
11. Jin, X., Cai, S., Li, H., Karniadakis, G.E.: Nsfnets (navier-stokes flow nets): Physics-informed neural networks for the incompressible navier-stokes equations. Journal of Computational Physics **426**, 109951 (2021)
12. Kharazmi, E., Zhang, Z., Karniadakis, G.E.: Variational physics-informed neural networks for solving partial differential equations. arXiv preprint arXiv:1912.00873 (2019)
13. Kharazmi, E., Zhang, Z., Karniadakis, G.E.: hp-vpinns: Variational physics-informed neural networks with domain decomposition. Computer Methods in Applied Mechanics and Engineering **374**, 113547 (2021)
14. Kovasznay, L.I.G.: Laminar flow behind a two-dimensional grid. Mathematical Proceedings of the Cambridge Philosophical Society **44**(1), 58–62 (1948). https://doi.org/10.1017/S0305004100023999
15. Levin, E., Fleisher, M.: Accelerated learning in layered neural networks. Complex systems **2**(625-640), 3 (1988)
16. Li, T., Zou, Y., Zou, S., Chang, X., Zhang, L., Deng, X.: Learning to solve pdes with finite volume-informed neural networks in a data-free approach. Journal of Computational Physics **530**, 113919 (2025)
17. Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., Anandkumar, A.: Fourier neural operator for parametric partial differential equations. arXiv preprint arXiv:2010.08895 (2020)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_13

18. Lim, K.L., Dutta, R., Rotaru, M.: Physics informed neural network using finite difference method. In: 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC). pp. 1828–1833. IEEE (2022)
19. Łoś, M., Służalec, T., Maczuga, P., Vilkha, A., Uriarte, C., Paszyński, M.: Collocation-based robust variational physics-informed neural networks (crvpinn). arXiv preprint arXiv:2401.02300 (2024)
20. Lu, L., Jin, P., Pang, G., Zhang, Z., Karniadakis, G.E.: Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. Nature machine intelligence **3**(3), 218–229 (2021)
21. Mao, Z., Jagtap, A.D., Karniadakis, G.E.: Physics-informed neural networks for high-speed flows. Computer Methods in Applied Mechanics and Engineering **360**, 112789 (2020)
22. Moukalled, F., Mangani, L., Darwish, M., Moukalled, F., Mangani, L., Darwish, M.: The finite volume method. Springer (2016)
23. Niaki, S.A., Haghighat, E., Campbell, T., Poursartip, A., Vaziri, R.: Physics-informed neural network for modelling the thermochemical curing process of composite-tool systems during manufacture. Computer Methods in Applied Mechanics and Engineering **384**, 113959 (2021)
24. Raissi, M., Perdikaris, P., Karniadakis, G.: 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). https://doi.org/https://doi.org/10.1016/j.jcp.2018.10.045, https://www.sciencedirect.com/science/article/pii/S0021999118307125
25. Ramachandran, P., Zoph, B., Le, Q.V.: Searching for activation functions. arXiv preprint arXiv:1710.05941 (2017)
26. Rishi, J., Gafoor, A., Kumar, S., Subramani, D.: On the training efficiency of shallow architectures for physics informed neural networks. In: International Conference on Computational Science. pp. 363–377. Springer (2024)
27. Rojas, S., Maczuga, P., Muñoz-Matute, J., Pardo, D., Paszyński, M.: Robust variational physics-informed neural networks. Computer Methods in Applied Mechanics and Engineering **425**, 116904 (2024)
28. Sankaran, S., Wang, H., Guilhoto, L.F., Perdikaris, P.: On the impact of larger batch size in the training of physics informed neural networks. In: The Symbiosis of Deep Learning and Differential Equations II (2022)
29. Su, Z., Liu, Y., Pan, S., Li, Z., Shen, C.: Finite volume physical informed neural network (fv-pinn) with reduced derivative order for incompressible flows. arXiv preprint arXiv:2411.17095 (2024)
30. Sun, L., Gao, H., Pan, S., Wang, J.X.: Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data. Computer Methods in Applied Mechanics and Engineering **361**, 112732 (2020)
31. Ueckermann, M.P., Lermusiaux, P.F.: 2.29 finite volume matlab framework documentation. MSEAS report **14** (2012)
32. Wang, C.: Exact solutions of the steady-state navier-stokes equations. Annual Review of Fluid Mechanics **23**(1), 159–177 (1991)
33. Wang, S., Teng, Y., Perdikaris, P.: Understanding and mitigating gradient flow pathologies in physics-informed neural networks. SIAM Journal on Scientific Computing **43**(5), A3055–A3081 (2021)
34. Yang, L., Meng, X., Karniadakis, G.E.: B-pinns: Bayesian physics-informed neural networks for forward and inverse pde problems with noisy data. Journal of Computational Physics **425**, 109913 (2021)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97554-7_13