Physics-Informed Neural Networks for Hyperbolic Conservation Laws

Project Report

Sagar Prakash Barad

Under the Guidance of

Prof. Victor Roy & Prof. Subhasish Basak

April 24, 2024

School of Physical Sciences
National Institute of Science Education and Research, Bhubaneswar

Contents

Abstract

In this work we have integrated finite volume methods (FVM) with physics-informed neural networks (PINNs) in the FV-PINNs framework to address the limitations of traditional PINNs in handling discontinuous solutions, particularly in hyperbolic conservation laws. By combining the high-order accuracy of FVM schemes with the generalizability of PINNs, FV-PINNs provide an efficient and versatile tool for solving partial differential equations (PDEs). The methodology can be extended to other numerical methods such as finite difference methods (FDM), finite element methods (FEM), and discrete element methods (DEM). Future work includes testing FV-PINNs on various hydrodynamics problems to assess its effectiveness and conducting comparative studies with traditional methods and other PINNs approaches.

1 Introduction

Since the introduction of Physics-Informed Neural Networks (PINNs) by Raissi et al. in [RPK19], there has been a significant increase in interest in this technique across various scientific fields. PINNs have the ability to solve both forward problems, where the solution is found given the initial and boundary conditions, and inverse problems, where the parameters of the problem are determined from data, for partial differential equations (PDEs). They act as space and time-efficient and easy to implement alternatives to traditional methods. Although PINNs may not perform as accurately or quickly in terms of training computing time as compared to current state-of-the-art methods, they present an attractive alternative for addressing challenges that are difficult for conventional techniques. For example, PINNs have shown promising results in solving inverse problems and parametric PDEs, where traditional methods often struggle. PINNs are being used in diverse number fields including parameter identification, material properties identification, source identification, medical imaging, and seismic inversion [Cuo+22]. They have been used to identify the parameters of PDEs, material properties of structures, location and strength of pollutant sources, subsurface structure based on seismic data, and image reconstruction from under-sampled data[Hao+23].
Physics-Informed Neural Networks (PINNs) have limitations when it comes to solving conservation laws, which often result in discontinuous solutions. This is because the universal approximation theorem for neural networks only guarantees that they can approximate continuous solutions to partial differential equations (PDEs). As of now, there is no theoretical backing that generalizes a neural network's ability to approximate discontinuous solutions. This raises significant complexity when designing a neural network to solve conservation laws, such as the compressible Euler equations, which describe gas and fluid flow by relating density, velocity, and pressure. These equations are widely used in modeling shock-tube problems and explosions. The finite volume method (FVM) [Tho99] is commonly used to solve these problems, but it requires adjustments to its original numerical formulation to accurately propagate shock and rarefaction waves. This is achieved through the use of flux-limiting, which resolves artificial dispersion and dissipation created by FVM schemes. Similarly, PINNs without modifications can develop similar phenomena when computing for solutions that include shocks and rarefaction fans[Kar+21]. Therefore, in order for PINNs to be competitive with the FVM using flux-limiting, it is necessary to modify the method to solve complex problems in fluid dynamics, particularly hydrodynamic shock-tube problems. This would involve incorporating techniques similar to flux-limiting into

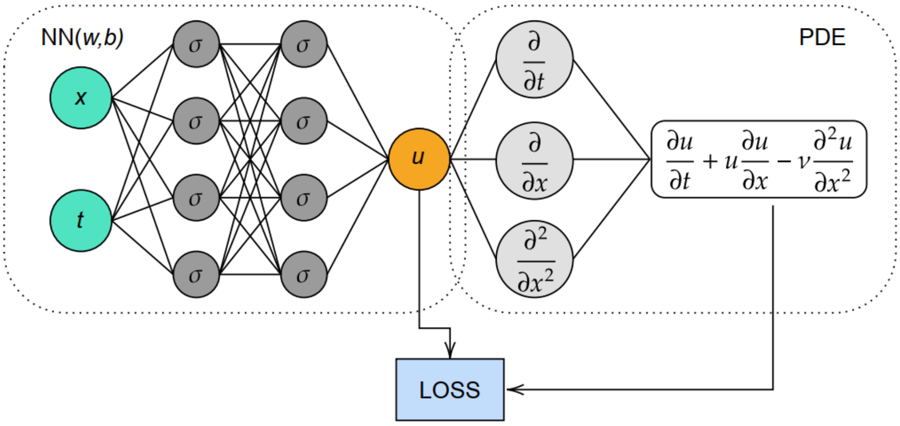

Figure 1: Schematic of a physics-informed neural network.

the neural network architecture and training process, in order to accurately capture discontinuities in the solution.

Problem Statement

Is it possible to develop a hybrid method that combines the flux-limiting capabilities of higher-order Finite Volume Methods (FVM) with the generalizability of Physics-Informed Neural Networks (PINNs) for solving hyperbolic equations?

Contributions

This work proposes a simple modification to Physics-Informed Neural Networks (PINNs) to improve their accuracy in solving a wide range of hydrodynamic shock-tube problems. While we began working on this independently, there have been previous attempts at similar modifications by other researchers such as [**FV-PINNs**]. Our implementation involves replacing the HLLE flux solver used in [**FV-PINNs**] with a Rusanov solver, taking inspiration from [Lin03]. The goal is to accurately solve the Sod shock-tube problem in both 1D and 2D using our modified PINNs, outperforming traditional Vanilla PINNs and Finite Volume Methods (FVMs) in terms of accuracy.

2 Physics Informed Neural Networks (PINNS)

The Universal Approximation Theorem (UAT) states that any continuous function and its derivatives can be approximated arbitrarily well by a single- or multi-layered neural network with enough hidden neurons, providing a theoretical foundation for using neural networks to approximate complex functions. This theorem suggests and proves that it is possible to approximate a solution to a PDE by using neural networks. Hence, PINNs becomes an eligible method for providing solutions to PDEs as well as their inverse problems.
Physics-Informed Neural Networks (PINNs) are a type of neural network designed to solve partial differential equations (PDEs). They leverage automatic differentiation, a feature built into most

deep learning frameworks such as TensorFlow and PyTorch, to calculate derivatives with respect to input coordinates. This is done without the need for mesh generation, making PINNs more flexible and efficient than traditional numerical methods. Instead of generating a mesh, a point cloud is generated in the domain and on the boundaries. Automatic differentiation can then be used to calculate the derivatives needed using these coordinates. PINNs can be applied to both forward and inverse problems. In forward problems, PINN approximates the physics in a domain by considering the differential equation, with only boundary and/or initial condition data provided. For inverse problems, PINN can learn nonlinear continuous operators from a relatively small dataset.
To illustrate the methodology of PINN, let's consider a general partial differential equation:

$$u_t + N[u] = 0, \quad x \in \Omega, t \in [0, T \tag{2.1}$$

$$u(x,t) = g(x,t), \quad x \in \partial\Omega, t \in [0, T] \tag{2.2}$$

$$u(x,0) = h(x), \quad x \in \Omega \tag{2.3}$$

where $u(x,t)$ represents the hidden solution, $\mathcal{N}[]$ is a generalized nonlinear operator, $g(x,t)$ represents the boundary conditions, and $h(x)$ is the initial condition. The hidden solution can be approximated by a feedforward neural network, $\hat{u}(x,t)$. However, for PINN, the objective function is a composite loss function in the form:

$$L = w_R L_R + w_{BC} L_{BC} + w_{IC} L_{IC}, \tag{2.15}$$

where w terms are specific weighting on the total loss. Using mean squared error for the loss function, the terms become:

$$L_R = \frac{1}{N_R} \sum_{i=1}^{N_R} |u_t^{\wedge} + N[\hat{u}]|^2 \tag{2.4}$$

$$L_{BC} = \frac{1}{N_{BC}} \sum_{i=1}^{N_{BC}} |\hat{u}(x_i, t_i) - g(x_i, t_i)|^2 \tag{2.5}$$

$$L_{IC} = \frac{1}{N_{IC}} \sum_{i=1}^{N_{IC}} |\hat{u}(x_i, 0) - h(x_i)|^2 \tag{2.6}$$

where L_R, L_{BC}, and L_{IC} represent the residual of the governing PDE, the boundary conditions, and the initial condition, respectively. N_R, N_{BC}, and N_{IC} are the number of sampled data points for different terms. Using automatic differentiation to calculate the derivatives with respect to input coordinates, this formulation forms the total loss with the residual, boundary, and initial conditions. Optimization algorithms minimize this objective function, leading to predicting a field that represents the governing physical laws.
In PINNs, collocation points are used to calculate the residual of the governing PDE, but insufficient initial/boundary data can lead to trivial solutions. To converge to the correct solution, PINN must propagate from known initial/boundary points to collocation points. Common sampling methods include random sampling from a uniform distribution or Latin hypercube sampling. However, dynamic random sampling, which samples a random set of collocation points at every iteration, is used in this project for its simplicity and computational efficiency.

3 FINITE VOLUME BASED PHYISCS INFORMED NEURAL NETWORK (FV-PINNS)

Although there is no formal theory that suggest that PINNs can handle non differenitable points and this is evident To minimize the reliance on automatic differentiation, we need a hybrid finite volume (FVM) and PINNs technique, lets call it, **FV-PINNs**. Once a FV-PINN is trained, the neural network provides an implicit solution (implicit neural representation), or rather, a parameterization of the approximate solution to the PDE. This allows for reevaluating the solution on subsets of the original training spatial-temporal domain, regardless of how irregular, and obtain solutions in seconds. Moreover, FV-PINNs allow for concrete error and stability analysis from standard FVM, and a simplistic construction of the deep neural network (DNN), by minimizing tedious hyperparameter tuning. currently we will discussing intergration of underlying FVM scheme with PINNs. However, the methodology presented can be generalized to other numerical methods such as the finite difference method (FDM) and the finite element methods (FEM).

3.1 FINITE VOLUME METHOD WITH FLUX LIMITING

For a particular cell $\boldsymbol{i}$, we take the volume integral over the total volume of the cell, v_i, which gives:

$$\int_{v_i} \frac{\partial \boldsymbol{U}}{\partial t}\, dv + \int_{v_i} \boldsymbol{\nabla} \cdot \boldsymbol{f}(\boldsymbol{U})\, dv = 0; \quad \boldsymbol{f} = (F, G); \quad \boldsymbol{r} = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right)$$

Integrating the first term to get the volume average and applying the divergence theorem to the second yields:

$$v_i \frac{d\bar{U}}{dt} + \int_{S_i} \boldsymbol{f}(\boldsymbol{U}) \cdot \boldsymbol{dn}\, dS = 0$$

where S_i represents the total surface area of the cell and $\boldsymbol{n}$ is a unit vector normal to the surface and pointing outward. Therefore, the generalized finite-volume formulation is as follows:

$$\frac{d\bar{U}}{dt} + \frac{1}{v_i} \int_{S_i} \boldsymbol{f}(\boldsymbol{U}) \cdot \boldsymbol{dn}\, dS = 0$$

In Computational Fluid Dynamics (CFD), researchers are constantly designing new numerical schemes to solve complex equations. Choosing the right scheme, especially for problems involving shock waves, rarefaction fans, and supersonic flow, can be challenging. To integrate in time, we employ a two-step Runge-Kutta method (RK-2). For computing fluxes, we use the Rusanov flux solver, known for its simplicity and stability. This solver, an upwind flux, considers the flow direction across cell interfaces, making it a popular choice in many simulations.

For simplicity, we define the Rusanov flux solver in 1-D, as it is easily generalizable to higher dimensions. Hence, $\boldsymbol{f}(\boldsymbol{U}) = F$. We discretize the equation such that $S_i = x_{j-1/2}, x_{j+1/2}$, and $v_i = \Delta x$ is the spatial step. Hence, we have:

$$\frac{\partial \bar{U}}{\partial t} + \frac{1}{\Delta x}\left(F_{j+1/2} - F_{j-1/2}\right)$$

where:

$$F_{j+1/2} = f(U_{j+1/2})$$

Define $U_{j+1/2}$ to be:

$$U_{j+1/2} = -\frac{f(U_{j+1}) - f(U_j)}{r_{j+1/2} - l_{j+1/2}} \left(r_{j+1/2}U_{j+1} - l_{j+1/2}U_j\right)$$

where $l_{j+1/2} < r_{j+1/2}$ are the left and right speeds. Next, we define the right centered flux as:

$$F_{j+1/2} = \frac{b^+_{j+1/2} f(U_j) - b^-_{j+1/2} f(U_{j+1})}{b^+_{j+1/2} - b^-_{j+1/2}} + \frac{b^+_{j+1/2} b^-_{j+1/2}}{b^+_{j+1/2} - b^-_{j+1/2}}$$

where:

$$b^+_{j+1/2} = \max(0, r_{j+1/2}); \quad b^-_{j+1/2} = \min(0, l_{j+1/2})$$

We compute the so-called Roe averages as follows:

$$\bar{u}_{j+1/2} = \frac{\sqrt{\rho_j} u_j + \sqrt{\rho_{j+1}} u_{j+1}}{\sqrt{\rho_j} + \sqrt{\rho_{j+1}}}; \quad \bar{H}_{j+1/2} = \frac{\sqrt{\rho_j} H_j + \sqrt{\rho_{j+1}} H_{j+1}}{\sqrt{\rho_j} + \sqrt{\rho_{j+1}}}; \quad \bar{c}_{j+1/2} = \sqrt{\frac{\gamma - 1}{2}\left(\bar{H}_{j+1/2} - \frac{1}{2}\bar{u}^2_{j+1/2}\right)}$$

Thus, the newly defined speeds are as follows:

$$l_{j+1/2} = \min\left(\hat{u}_{j+1/2} - \hat{c}_{j+1/2}, u_j - c_j\right)$$

$$r_{j+1/2} = \max\left(\hat{u}_{j+1/2} + \hat{c}_{j+1/2}, u_{j+1} - c_{j+1}\right)$$

Lastly, the monotonized central flux limiter is defined by:

$$\varphi_{\mathrm{mc}}(r) = \max\left[0, \min\left(2r, \frac{1+r}{2}, 2\right)\right]; \quad r_j = \frac{U_j - U_{j-1}}{U_{j+1} - U_j}$$

Finite Volume Physics-Informed Neural Networks (FV-PINNs) combine the accuracy of traditional finite volume schemes with the generalizability of a quintessential Physics-Informed Neural Network (PINN). The FVM with Rusanov flux reconstruction and flux limiting provides highly accurate solutions to compressible CFD problems. Instead of differentiating the neural network with respect to the governing Partial Differential Equation (PDE), we discretize the neural network and minimize the Mean Squared Error (MSE) of the neural network evaluated on the respective numerical scheme.

3.2 DIFF Loss

$$DIFF\_loss = \sum_{i,j} G(\theta)^2, \quad G(\theta) = U^{n+1}_{i,j}(\theta) - \left(U^n_{i,j}(\theta) + \frac{\Delta t}{\Delta x} R\left(F^n_{i+1/2}, F^n_{i-1/2}\right)\right) \tag{3.1}$$

Here, $DIFF\_loss$ is the total loss, $G(\theta)$ is the difference between the predicted solution at time level $n+1$ and the updated solution based on the finite volume method update formula, $U^{n+1}_{i,j}(\theta)$ and $U^n_{i,j}(\theta)$ are the approximate solutions at time levels $n+1$ and n, respectively, for grid point (i, j), $R()$ is the numerical flux function used in the finite volume method to approximate the convective term, Δt is the time step size, and Δx is the spatial step size. By minimizing this total loss over all grid points, we can train a neural network with parameters θ, which will provide an implicit solution or parametrization of the approximate solution to the PDE. The resulting FV-PINN model can then be used for prediction and uncertainty quantification in complex systems with discontinuous solutions.

Algorithm 1 FV-PINNS Algorithm with Rusanov Flux Solver

Generate weights $\theta \in \mathbb{R}^k$ and a deep neural network (DNN), $U(\boldsymbol{x}_i, \boldsymbol{y}_j, t_n, \theta) = U^n_{i,j}$, where $(\boldsymbol{x}_i, \boldsymbol{y}_j, t_n)$ are inputs to the network, and $U^n_{i,j} = [\rho^n_{i,j}, u^n_{i,j}, v^n_{i,j}, p^n_{i,j}]$ are the outputs.
Minimize $\|U^0 - \hat{U}^0\|^2$ via stochastic gradient descent.
while $t < T$ **do**
    $G(\theta) = U^{n+1}_{i,j} - \left(U^n_{i,j} + \frac{\Delta t}{\Delta x} R\left(F^n_{i+1/2}, F^n_{i-1/2}\right)\right)$
    Update θ by performing stochastic gradient descent: $\theta = \theta - \eta \nabla_\theta G(\theta)$, where η is the learning rate.
    Employ boundary conditions.
    $t_{n+1} = t_n + \Delta t$
end while

Figure 2: Sod shock tube at t=0s. On the left-hand side region with high density, pressure and temperature

3.3 Algorithm

Once the neural network has been trained for the following time step, instead of evaluating the gradient of the flux using automatic differentiation, we compute the numerical flux with respect to the neural network using the Rusanov flux solver and monotonized central flux limiter. The computation of the numerical flux is parallelized using Message Passing Interface (MPI). To train the neural network, we use an X-PINNs approach, which decomposes the spatial grid into N regions and trains individual neural networks on each subgrid. Once each region has been trained, we assemble the global neural network approximation.

3.4 Example: 1D Sod Shocktube

Consider the example of 1-D compressible Euler equations, with the initial state of the Sod shocktube problem (Sod, 1978). FV-PINNs are able to simuate 2-D and 3-D compressible flow with discontinuous initial states, but we will only discuss till 2D.

$$\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}}{\partial x} = \boldsymbol{0}, \quad (x,t) \in \boldsymbol{\Omega} \times (0,T] \tag{3.2}$$

where, $T > 0$, $\boldsymbol{\Omega} \equiv (0,1)$ and

$$\boldsymbol{U} = \begin{bmatrix} \rho \\ \rho u \\ E \end{bmatrix}, \quad \boldsymbol{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ (E+p)\,u \end{bmatrix}, \tag{3.3}$$

The initial state for the Sod problem is as follows:

$$U_0 = (\rho_0, u_0, p_0) = \begin{cases} (1.0, 0.0, 1.0), & x \le 0.5 \\ (0.125, 0.0, 0.1), & x > 0.5 \end{cases}$$

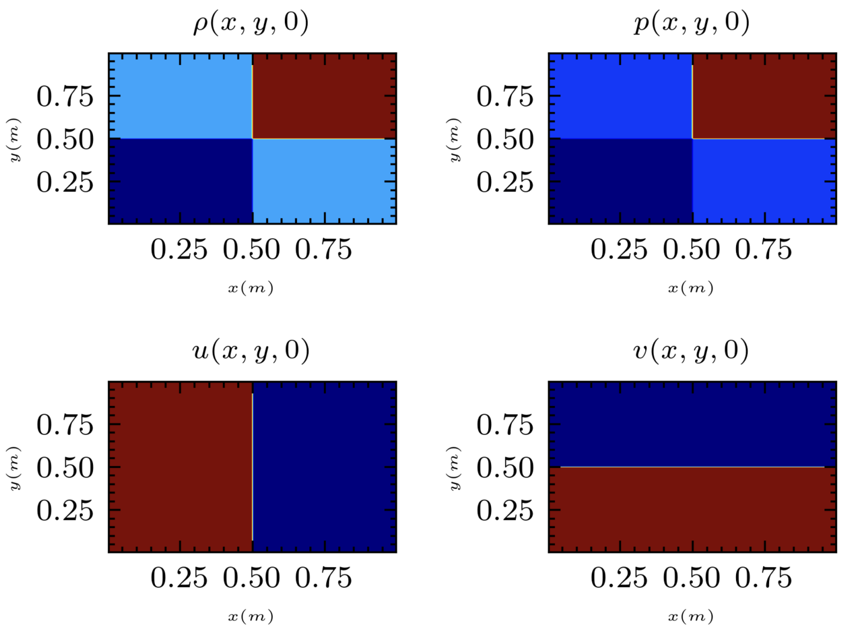

Figure 3: Liska 2D shock tube at t=0s. All the 4 quadrants have different density, pressure and temperature

3.5 EXAMPLE: 2D LISKA SHOCKTUBE

For the 2D Liska shock tube problem, the governing equations are:

$$\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}}{\partial x} + \frac{\partial \boldsymbol{G}}{\partial y} = \boldsymbol{0}, \quad (x, y, t) \in \boldsymbol{\Omega} \times (0, T]$$

Where:

- $T > 0$ denotes the final time.
- $\boldsymbol{\Omega} \equiv (0, 1) \times (0, 1)$ represents the spatial domain.

The vector of conserved variables and the flux vectors are given by:

$$\boldsymbol{U} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ E \end{bmatrix}, \tag{3.4}$$

$$\boldsymbol{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ (E + p)u \end{bmatrix}, \tag{3.5}$$

$$\boldsymbol{G} = \begin{bmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ (E + p)v \end{bmatrix} \tag{3.6}$$

The initial state for the Liska shock tube problem is given by:

$$U_0 = (\rho_0, u_0, v_0, p_0) = \begin{cases} (1.0, 0.0, 0.0, 1.0), & \text{if } x \leq 0.5 \text{ and } y \leq 0.5 \\ (0.125, 0.0, 0.0, 0.1), & \text{if } x > 0.5 \text{ and } y \leq 0.5 \\ (0.125, 0.0, 0.0, 0.1), & \text{if } x \leq 0.5 \text{ and } y > 0.5 \\ (0.1, 0.0, 0.0, 1.0), & \text{if } x > 0.5 \text{ and } y > 0.5 \end{cases}$$

4 Results & Analysis

We optimize all loss functions using L-BFGS, a quasi-Newton, full-batch gradient-based optimization algorithm. For larger datasets, we can employ a more computationally efficient mini-batch setting using stochastic gradient descent and its modern variants. Despite the lack of theoretical guarantee for convergence to a global minimum, empirical evidence shows that our method achieves good prediction accuracy with a sufficiently expressive neural network architecture and a sufficient number of collocation points N_f when the given partial differential equation is well-posed and its solution is unique.
All the code is available in this Github repository.

Sod Shock-Tube Problem

The Sod shock-tube problem is a standard test for numerical methods in hydrodynamics. It includes features like rarefaction fans, contact discontinuities, and shocks, which are challenging to capture accurately due to artificial dispersion or dissipation. Figure 4 illustrates that our FV-PINNs method effectively captures these components for density and pressure, although some dissipation near the right shock is observed in velocity. Overall, the results demonstrate the method's capability in accurately resolving discontinuities, promising for obtaining precise numerical solutions.

Liska Shock-Tube Problem

The 2D Liska shock tube problem, much like the Sod shock-tube problem, serves as a benchmark for evaluating numerical methods in hydrodynamics. It presents challenges in accurately capturing features such as rarefaction fans, contact discontinuities, and shocks, primarily due to artificial dispersion or dissipation. Figure 5 showcases our FV-PINNs method's effectiveness in capturing these components for density and pressure. Although some dissipation near the right shock is evident in velocity, the overall performance underscores the method's capability in accurately resolving discontinuities, offering promising prospects for precise numerical solutions.

5 Conclusion

The integration of finite volume methods (FVM) with physics-informed neural networks (PINNs) in the FV-PINNs framework offers a promising approach to address the limitations of traditional PINNs in handling discontinuous solutions, particularly in hyperbolic conservation laws. By combining the high-order accuracy of FVM schemes with the generalizability of PINNs, FV-PINNs provide an efficient and versatile tool for solving partial differential equations (PDEs).

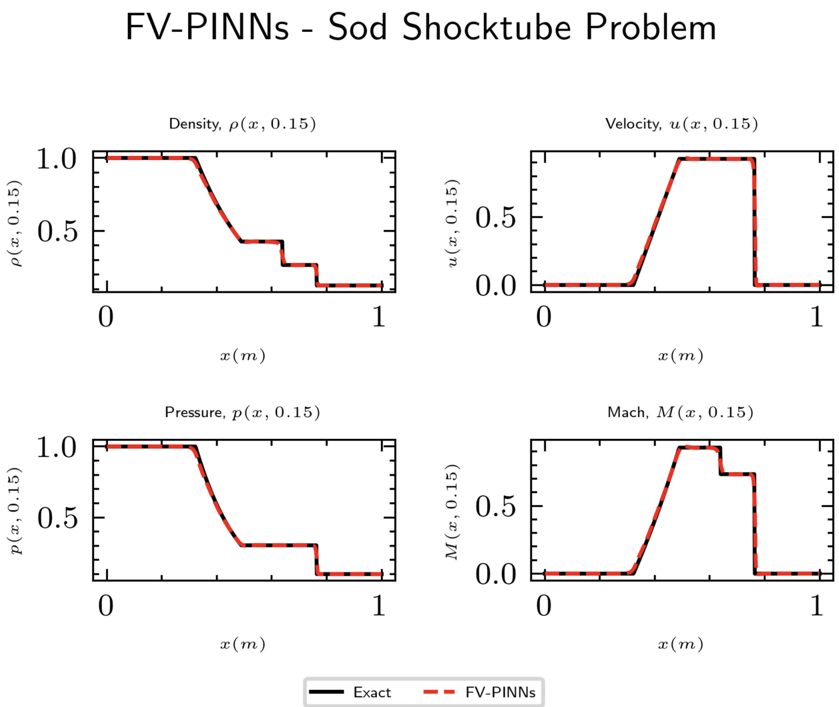

Figure 4: *Sod Shock-Tube Problem*: Solution of the velocity, pressure, and density at final time, $t = .15$.

FV-PINNs offer several advantages over traditional PINNs, including rapid evaluation of solutions on subsets of the original training domain and refined grids, concrete error and stability analysis based on standard FVM techniques, and simplified construction of deep neural networks with reduced hyperparameter tuning. The implicit neural representation obtained through FV-PINNs allows for efficient computation of solutions in diverse scientific and engineering domains. Future work includes testing FV-PINNs on various hydrodynamics problems to assess its effectiveness and conducting comparative studies with traditional methods and other PINNs approaches. These investigations will provide valuable insights into the performance and potential applications of FV-PINNs, which can be extended to other numerical methods such as finite difference methods (FDM), finite element methods (FEM), and discrete element methods (DEM). Overall, FV-PINNs offer a promising approach for solving PDEs with discontinuous solutions.

References

[Tho99] J. W. Thomas. *Numerical partial differential equations : conservation laws and elliptic equations*. eng. Texts in applied mathematics ; 33. New York: Springer, 1999. ISBN: 0387983465.

[Lin03] Chi-Tien Lin. “New High-Resolution Central-Upwind Schemes for Nonlinear Hyperbolic Conservation Laws”. In: *Hyperbolic Problems: Theory, Numerics, Applications*. Ed. by Thomas Y. Hou and Eitan Tadmor. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 705–715. ISBN: 978-3-642-55711-8.

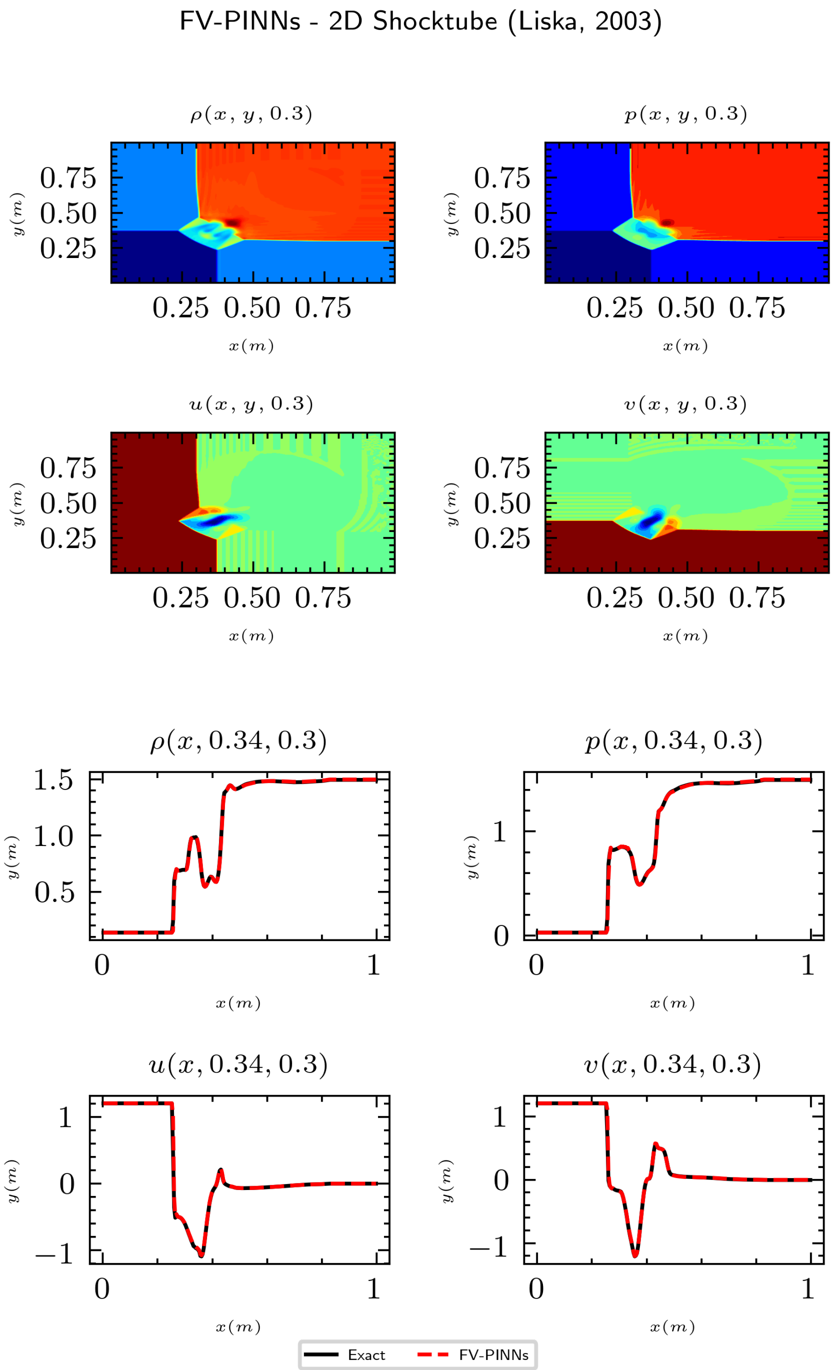

Figure 5: *Liska Shock-Tube Problem*: Top: Solution of the velocity, pressure, and density at final time, t = 0:2. Bottom: FV-PINNs solution of each physical quantity at $(x; t)$. FV-PINNs and PINNs had sample sizes of $N_f = 10,500$, $N_{IC} = 1000$.

[RPK19] 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". In: *Journal of Computational Physics* 378 (2019), pp. 686–707. issn: 0021-9991. doi: https://doi.org/10.1016/j.jcp.2018.10.045. url: https://www.sciencedirect.com/science/article/pii/S0021999118307125.

[Kar+21] George Em Karniadakis et al. "Physics-informed machine learning". In: *Nature Reviews Physics* 3.6 (May 2021). issn: 2522-5820. doi: 10.1038/s42254-021-00314-5. url: https://www.osti.gov/biblio/1852843.

[Cuo+22] Salvatore Cuomo et al. *Scientific Machine Learning through Physics-Informed Neural Networks: Where we are and What's next*. 2022. arXiv: 2201.05624 [cs.LG].

[Pap22] Alexandros Papados. *Finite-Volume Physics-Informed Neural Networks*. Mar. 2022. doi: 10.13140/RG.2.2.19087.05286.

[Hao+23] Zhongkai Hao et al. *Physics-Informed Machine Learning: A Survey on Problems, Methods and Applications*. 2023. arXiv: 2211.08064 [cs.LG].