XLVI CILAMCE
VITÓRIA - ES - BRAZIL - 2025
XLVI IBERO-LATIN AMERICAN CONGRESS ON COMPUTATIONAL METHODS IN ENGENEERING

Use of Physics-Informed Neural Networks to compute velocity and pressure fields around an airfoil in transonic flow

Thiago S. Batista[1], Vinícius P. M. Souza[1], Vinicius C. Rispoli[1]

[1]*Faculdade de Ciências e Tecnologia, Universidade de Brasília*
St. Leste Projeção A - Gama Leste, 72444-240, Brasília, Federal District, Brazil
thiagodsbat@gmail.com, passerimoraes@gmail.com, vrispoli@unb.br

Abstract. This work applies Physics-Informed Neural Networks (PINNs) to simulate transonic flow (Mach 0.8–1.2) around an airfoil, a regime characterized by coexisting subsonic and supersonic regions, shock waves, and potential boundary layer separation. Such nonlinear phenomena pose challenges for conventional Computational Fluid Dynamics (CFD) methods, which often require fine shock-aligned meshes and high computational cost. In this approach, the compressible Euler equations are embedded directly into the network's loss function, enabling mesh-free solution of the flow field. The model, implemented with the DeepXDE library, uses a two-dimensional wind tunnel domain with no-slip boundary conditions on the airfoil and far-field conditions at the outer boundaries. Residual-based adaptive collocation sampling mitigates mesh dependency and focuses training where the physics are most complex. To improve shock resolution and training stability, adaptive loss weighting is incorporated. Preliminary results indicate that the PINN approximately predicts shock location and overall pressure distribution, demonstrating its potential as a mesh-independent solver for aerodynamic problems where traditional methods face meshing or stability limitations.

Keywords: Airfoil, Transonic Flow, Physics-Informed Neural Networks, DeepXDE

1 Introduction

Physics-Informed Neural Networks (PINNs) have emerged as a powerful paradigm for solving partial differential equations (PDEs) by seamlessly integrating physical principles into deep learning frameworks. Unlike traditional numerical methods, PINNs leverage automatic differentiation to embed the residuals of PDEs, along with initial and boundary conditions, directly into the loss function. This approach not only enforces physical constraints but also enables the learning of continuous spatiotemporal fields without reliance on discretized grids (Raissi et al. [1]). PINNs have demonstrated remarkable success across diverse applications, including fluid dynamics (Jin et al. [2]), heat transfer (Lu et al. [3]), and inverse problems (Karniadakis et al. [4]). However, challenges persist in scaling PINNs to high-dimensional problems and noisy data, prompting ongoing research into hybrid architectures and adaptive training strategies (Wang et al. [5]).

One of the promising applications of PINNs lies in Aerodynamics, particularly in flow estimation, as a replacement for or complement to conventional Computational Fluid Dynamics (CFD) methods. Their advantages include the elimination of the need for refined and numerically stable meshes, as well as reduced computational cost in inverse problems compared to traditional approaches (Anderson [6]). Moreover, PINNs incorporate conservation laws directly into the learning process, enabling a natural fusion of sparse data and continuous models. Nonetheless, significant challenges remain, especially in accurately representing discontinuities and shock waves (Jagtap et al. [7]).

The transonic flow regime is characterized by the coexistence of subsonic and supersonic regions around the airfoil, for Mach numbers close to unity (typically between 0.8 and 1.2), with the formation of shock waves that can induce boundary layer separation and a sharp increase in drag (Anderson [8]).

Within this context, the present study formulates and applies a PINN, to compute the two-dimensional compressible flow around an airfoil in the transonic regime, investigating its potential and limitations in comparison with classical CFD references.

The proposed implementation employs the DeepXDE library, which provides a high-level *API* for solving both forward and inverse problems, supports complex geometries via constructive solid geometry, offers residual-based adaptive sampling strategies, and integrates with different automatic differentiation backends (Lu et al. [3]).

These capabilities enable rapid prototyping and flexible customization of architectures and loss functions.

2 Problem Studied

2.1 Physical Model

The transonic flow regime is classified as viscous and compressible. Therefore, three partial differential equations are employed to characterize it: the continuity equation, representing mass conservation; the Navier–Stokes equation, representing momentum conservation; and the energy conservation equation (Anderson [6]).

For the present problem, certain simplifications can be made. In the flow around an airfoil, the problem becomes two-dimensional, and for the purposes of this analysis, a steady-state regime is considered, meaning that temporal variations in the flow are neglected. Thus, the formulation proposed and employed by Wassing et al. [9] is adopted, which reduces the aforementioned three equations to the so-called compressible Euler equations (Eqs. 1 and 2):

$$\frac{\partial \mathbf{F}_x}{\partial x} + \frac{\partial \mathbf{F}_y}{\partial y} = 0 \tag{1}$$

where

$$\mathbf{F}_x = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uE + pu \end{pmatrix}, \quad \mathbf{F}_y = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vE + pv \end{pmatrix}. \tag{2}$$

Here, ρ denotes the fluid density (air in this work), u and v are the horizontal and vertical velocity components, respectively, p is the static pressure, and E is the total energy of the flow, expressed as:

$$E = \frac{1}{\gamma - 1}\frac{p}{\rho} + \frac{u^2 + v^2}{2}, \tag{3}$$

where γ is the specific heat ratio, taken as $\gamma = 1.4$ for air.

It is worth noting that in this problem the fluid viscosity—commonly accounted for by adding a viscous stress tensor (Anderson [6])- is neglected. This simplification is acceptable within the present context, since viscosity modeling is a significant concern in conventional computational methods, but is not always necessary when working with PINNs (Raissi et al. [1]).

The airfoil selected for this study was the SD8020, developed within the context of thin airfoil profiles and belonging to the Selig/Donovan family (Selig et al. [10]). It exhibits a maximum thickness of approximately 10.1% of the chord length, located at about 27.5% of the chord from the leading edge. The coordinates defining the geometry of the airfoil were obtained from the Airfoil Tools platform. The choice of this airfoil is further justified by its symmetric shape, which makes it suitable for analyses aimed at eliminating the influence of camber on pressure distribution, thereby isolating flow-related phenomena.

2.2 Boundary Conditions

Partial differential equations describe a broad class of problems, whose specific features are determined by the initial and boundary conditions. Since the present analysis focuses on a steady-state problem, it is fully defined by its boundary conditions.

Far-field boundary conditions, similar to those used to simulate wind tunnel conditions, are imposed along the domain left boundary where the temperature and pressure are typically set to ambient values (298 K and 1 atm). In addition, no-slip condition is applied on the airfoil surface.

Accordingly:

$$\begin{aligned} p_\infty &= 101325 \quad [\mathrm{Pa}] \\ \rho_\infty &= 1.225 \quad [\mathrm{kg/m^3}] \\ u_\infty &= 272 \quad [\mathrm{m/s}] \\ v_\infty &= 0 \quad [\mathrm{m/s}] \end{aligned} \tag{4}$$

The freestream velocity of $272\ \mathrm{m/s}$ is selected to yield a Mach number of 0.8, which corresponds to the lower limit of the transonic regime. Furthermore, the analysis considers only an angle of attack of $0°$, so that only the horizontal velocity component is relevant.

The no-slip condition applied to the surface of the SD8020 airfoil profile are expressed as:

$$u(x_{\text{airfoil}}, y_{\text{airfoil}}) = v(x_{\text{airfoil}}, y_{\text{airfoil}}) = 0 \tag{5}$$

3 Methods

The neural network was implemented using the Python library DeepXDE (Lu et al. [3]) with PyTorch backend. Each governing equation is constructed using the library's automatic differentiation tools, resulting in four equations incorporated into the training process. The selected trainable parameters are ρ, u, v, and p. The network architecture consists of 6 hidden layers with 65 neurons each, for a total of 21,909 trainable parameters, employing the hyperbolic tangent activation function and Glorot uniform initialization of the trainable parameters.

In line with the literature, the network training is performed in two stages: the first using the ADAM optimizer (Adaptive Moment Estimation), with 10,000 iterations and learning rate 10^{-5}, and the second employing L-BFGS (Limited-memory Broyden–Fletcher–Goldfarb–Shanno), which stopped automatically with 15,540 iterations.

The physical domain chosen is the rectangle $[-15, 50] \times [50, 50]$ with the airfoil in the vertical center. Inside the domain, 6000 collocation points were uniformly sampled, and 1000 were sampled in the contour. In addition, during the training process, at every 100 iterations, the collocation points were resampled to enhance convergence.

The equations were written in adimensional form. In this approach, some terms of the momentum and energy equations were multiplied by the adimensional factor $h = \dfrac{p_\infty}{\rho_\infty \cdot c^2}$, where c denotes the speed of sound in the medium, taken as $340\ \mathrm{m/s}$. The adimensional far-field boundary conditions are therefore expressed as $p_\infty = 1$, $\rho_\infty = 1$, $u_\infty = 0.8$ and $v_\infty = 0$.

All experiments were conducted on Google Colab using a T4 GPU. The implementation can be found on GitHub: https://github.com/labviaunb/pinns.

4 Results and Discussion

Figure 1 shows the behavior of the train and test losses during training. Here is observed an adequate decay, showing the convergence of the technique. The final training and test losses were, respectively, 1.96×10^{-6} and 2.66×10^{-6}. The training process took approximately one hour to finish in a T4 GPU in Google Colab. On the other hand, the results presented in Figures 2a, 2b, 3a and 3b represents the velocity magnitude and pressure, respectively, obtained after 10,000 iterations with the ADAM optimizer followed by an additional 15,540 iterations with the L-BFGS optimizer.

Although the results fall short of the expected accuracy, the ability of the PINN to reproduce fluid behavior is a strong indication of its potential as an alternative to conventional CFD methods. The velocity field visualization reveals evidence of boundary layer separation occurring significantly upstream of the flow–airfoil interaction, accompanied by indications of a potential shock wave formation within the pressure distribution of the flow, as well as a stagnation point at the leading edge. These are characteristic features of transonic flows, which combine subsonic phenomena—such as boundary layer development—with supersonic phenomena—such as shock wave formation. The main shortcoming, however, is that these features are observed far upstream of the airfoil, rather than in its expected vicinity. With further refinement and parameter tuning, the method could more closely approximate the expected behavior of such flows.

It is important to note that several key strategies were not adopted in this initial work. First, adaptive resampling of collocation points was not implemented to concentrate them in regions with higher loss values. Additionally, a higher point density should be sampled near the airfoil surface, as this is where the dependent variables exhibit the strongest gradients. Crucially, while no-slip boundary conditions were implemented, the no-penetration

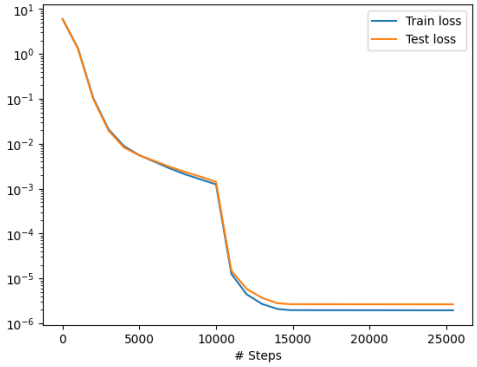
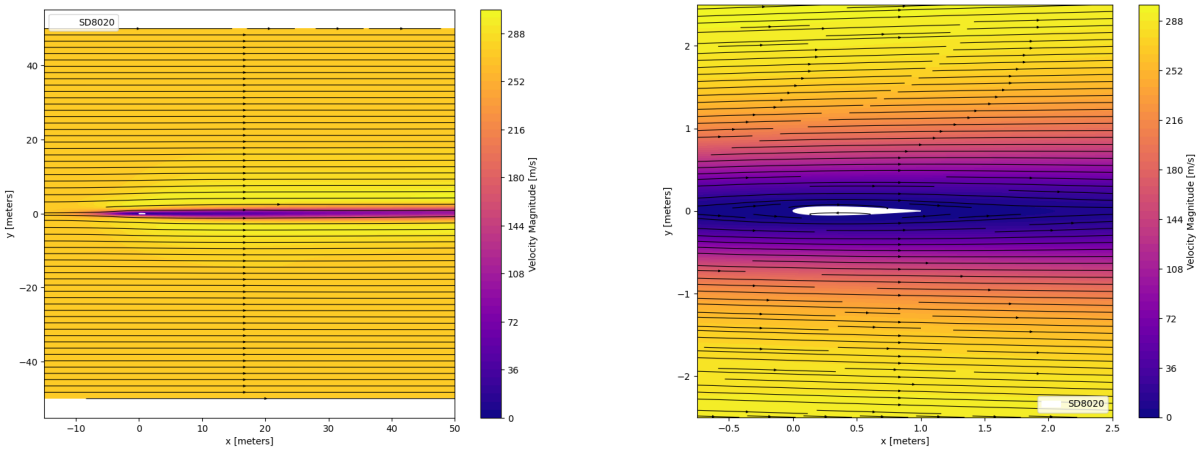
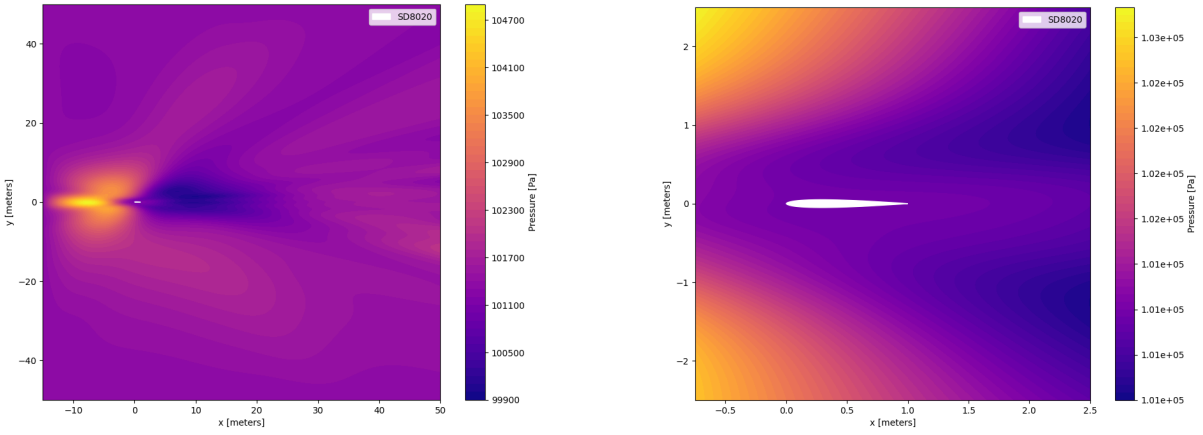


Figure 1. Training and testing loss values throughout training.



(a) (b)

Figure 2. (a) Far field flow velocity magnitude around the airfoil; (b) Near airfoil flow velocity magnitude.



(a) (b)

Figure 3. (a) Far field pressure distribution around the airfoil; (b) Near airfoil pressure distribution.

condition was not enforced in this pilot study—a requirement that must be addressed in future work. Finally, we intentionally avoided using a transformed domain (unlike the approach in Wassing et al. [9]) to evaluate whether satisfactory results could be obtained in a standard Cartesian rectangular domain.

5 Conclusions

The development and application of a Physics-Informed Neural Network (PINN) to the transonic flow problem around an airfoil allowed the identification of both strengths and limitations of the approach. Throughout the process, it was observed that factors such as the specification of boundary conditions, the scaling of physical parameters, and the sampling strategy have a direct influence on training stability and convergence.

The occurrence of gradient explosion during the initial stages highlighted the importance of preprocessing techniques, such as nondimensionalization of variables, to enable stable network training. Furthermore, the need for dynamic resampling reinforced that the distribution of collocation points is crucial to effectively direct computational effort toward the most relevant regions of the domain.

These insights provide guidance for future work, in which the use of techniques specifically designed to handle discontinuities, more rigorous hyperparameter tuning, and the integration of experimental or high-fidelity CFD data could significantly enhance the quality of PINN solutions in transonic regimes.

Acknowledgements. The authors would like to thank Professor Olexiy Shynkarenko from the University of Brasília for his contribution to the initial conception of this study.

Authorship statement. The authors hereby confirm that they are the sole liable persons responsible for the authorship of this work, and that all material that has been herein included as part of the present paper is either the property (and authorship) of the authors, or has the permission of the owners to be included here.

References

[1] M. Raissi, P. Perdikaris, and G. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019.

[2] X. Jin, S. Cai, H. Li, and G. E. Karniadakis. Nsfnets (navier-stokes flow nets): Physics-informed neural networks for the incompressible navier-stokes equations. *Journal of Computational Physics*, vol. 426, pp. 109951, 2021.

[3] L. Lu, X. Meng, Z. Mao, and G. E. Karniadakis. Deepxde: A deep learning library for solving differential equations. *SIAM Review*, vol. 63, n. 1, pp. 208–228, 2021.

[4] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang. Physics-informed machine learning. *Nature Reviews Physics*, vol. 3, n. 6, pp. 422–440, 2021.

[5] S. Wang, Y. Teng, and P. Perdikaris. When and why pinns fail to train: A neural tangent kernel perspective. *Journal of Computational Physics*, vol. 449, pp. 110768, 2022.

[6] J. Anderson. *Computational Fluid Dynamics: The Basics with Applications*. McGraw-Hill, 1995.

[7] A. Jagtap, E. Kharazmi, and G. 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*, vol. 365, pp. 113028, 2020.

[8] J. Anderson. *Modern Compressible Flow: With Historical Perspective*. McGraw-Hill, 2003.

[9] S. Wassing, S. Langer, and P. Bekemeyer. Physics-informed neural networks for transonic flows around an airfoil. *arXiv:2408.17364*, 2024.

[10] M. S. Selig, J. F. Donovan, and D. B. Fraser. *Summary of Low Speed Airfoil Data*, volume 1. SoarTech Publications, Virginia Beach, VA, 1995.