The 9th European Congress on Computational Methods in Applied Sciences and Engineering
ECCOMAS Congress 2024
3–7 June 2024, Lisboa, Portugal

ON PHYSICAL-CONSTRAINTS IN SCIENTIFIC MACHINE LEARNING FOR FLOW FIELD PREDICTIONS

MANUEL CABRAL[1], BERNAT FONT[1], GABRIEL D. WEYMOUTH[1]

[1] Faculty of Mechnical Engineering, Delft University of Technology, Mekelweg 2, 2628CD Delft, the Netherlands

Key words: Scientific Machine Learning, Physics-Constrained Neural Networks, Incompressibility, Non-Dimensionalization, Fluid Dynamics

Summary. Deep learning models have demonstrated remarkable capabilities at producing fast predictions of complex flow fields. However, incorporating known physics is essential to ensure that physical solutions can generalize to flow regimes not used for training.

In this study, a formulation that, by construction, enforces flow incompressibility and respects the invariance of physical laws across different unit systems is introduced. We demonstrate that this approach can achieve performance improvements of up to 100 times compared to purely data-driven methods, all while maintaining fidelity to other crucial physical quantities. Moreover, we show that for canonical flow test cases, such a physics-constrained model can yield accurate results even with training datasets as small as a few hundred points and neural networks containing only a handful of neurons.

It is also shown, however, that physics-constrained machine learning models are not silver bullets out of the box, and require careful consideration in their application and integration with other constraints. Specifically, this study addresses how a problem that is mathematically simple may not necessarily be straightforward in machine learning terms, and discusses ongoing efforts to bridge this gap.

We conclude by discussing the place of physics-constrained machine learning models within a landscape primarily dominated by physics-informed approaches, in particular in the context of real-world problems where data and computational resources are often limited.

1 INTRODUCTION

Despite exponential growth in high-performance computational capabilities and advances in numerical methods [1], computational fluid dynamics (CFD) remains prohibitively expensive for design applications. Fully resolved fluid simulations often require up to trillions of grid points and tens of millions of CPU hours [2], making them unfeasible for practical use beyond fundamental studies. Turbulence models such as RANS or LES can reduce this computational burden, albeit at the cost of accuracy. Nonetheless, these models remain too slow for applications where a large number of simulations are required, such as topological optimization.

More recently, the advent of machine learning (ML) has spurred interest in data-driven methods for fluid predictions. ML has been successfully applied across various fields, from computer vision to large language models, and everything in between, due to its ability to find patterns and structure within the training data without explicit models and provide fast predictions during inference. This capability makes ML, in theory, an ideal candidate to replace or complement

CFD [3]. However, this approach, by itself, has significant limitations. If the training data does not adequately cover the design space, ML models can produce large prediction errors. Additionally, ML models suffer from the law of diminishing returns: achieving incremental improvements in accuracy requires exponentially larger training datasets and increasingly complex models [4]. These larger models are more prone to over-fitting, training pathologies, and issues such as vanishing or exploding gradients in the case of neural networks. As a result, while pure data-driven models tend to yield low field errors (i.e., *good looking solutions*), they exhibit high physical errors, such as flow through boundaries, divergence issues, and incorrect physical quantities like lift and drag.

The field of physics-based machine learning (PBML) aims to combine the speed of ML algorithms with the assurance of physically meaningful solutions by integrating prior domain knowledge into the algorithm (fig. 1). There are two ways of incorporating physics into a ML model: via soft-constraints or hard-constraints (or a combination of both). Nomenclature on the literature diverges. Here, we will use physics-informed machine learning (PIML) when referring to soft constrained architectures, and physics-constrained machine learning (PCML) when referring to hard constrained ones. PIML penalizes violations of physical laws by adding a term to the loss function, ensuring the solution remains as close as possible to the physically relevant manifold. It has the advantage of not requiring specific knowledge about the law itself, as the procedure is consistent across different laws. However, since PIML applies physical constraints only during training, it offers no guarantees during inference. Additionally, incorporating more physics into the model increases the complexity of balancing different loss terms and makes the loss landscape more challenging to navigate. This fact has the effect of a dichotomy in the field: from plenty of success stories on one hand [5], to many others on the struggle to converge in well understood problems [6, 7]. PCML, on the other hand, integrates physical prior knowledge directly into the model architecture, ensuring the solution inherently resides on the physical manifold. In this approach, physical laws are enforced at all stages. This added structure reduces dependency on the training data and allows for the use of smaller models. However, developing methods to enforce physical laws in this manner is challenging and requires substantial field knowledge. Due to the apparent ease of use of PIML, particularly its well-known variant Physics-Informed Neural Networks (PINNs) [8], contributions to the more case-specific PCML have been relatively scarce [9, 10]. Additionally, PCML applications are often used in combination with other methods that can obscure its potential limitations. This motivates the need for a more fundamental study of the successes and limitations of each added constraint, which this study attempts to provide.

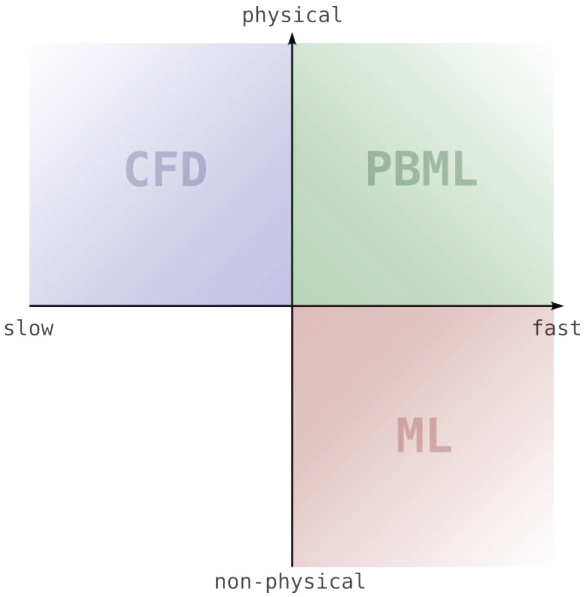

Figure 1: The landscape of methods in fluid dynamics.

This study introduces a PCML formulation where incompressibility is enforced by design, and learning is conducted in a physically meaningful, non-dimensional, low-dimensional space. Future research avenues, including the addition of boundary conditions to this modular architecture, are also discussed.

2 Methodology

Consider the general ML problem, where one is given N data points over k input variables and the goal is to learn the mapping f from the inputs, x_i, to the outputs, y_i. All examples presented in this paper involve 2-dimensional incompressible flows, so $\vec{y} = \{u, v\}$. The inputs consist of two spatial coordinates, x and y, along with other physical parameters relevant to the problem, such as free-stream velocity U_∞ or viscosity ν. In the examples analyzed, the data will be sampled uniformly across the input space, drawn from analytical (or close to it) expressions. The data-driven approach - which will serve as the baseline for comparing ways of adding physics - involves training the model to minimize the general L_2 loss function over the training data:

$$\mathcal{L}_0(\vec{x};\vec{\theta}) = ||\hat{f}(\vec{x}) - f(\vec{x})||_2^2 \tag{1}$$

Consider now the physics-informed variant, that modifies 1 in order to penalize violations to physical principles. Let's consider the case of incompressibility, $\vec{\nabla} \cdot \vec{u} = 0$, as all the cases that will be analyzed in this study involve incompressible flow. The natural way to impose this condition is to integrate it over the whole domain to condition our system, via Lagrange multipliers, to live close to this physical manifold. In practice, we need to approximate the value of this integral via Monte Carlo sampling, which results in the PIML modification of 1:

$$\mathcal{L}(\vec{x};\vec{\theta}) = \mathcal{L}_0(\vec{x};\vec{\theta}) + \lambda||\vec{\nabla} \cdot \vec{u}||_2^2 \tag{2}$$

The parameter λ quantifies the importance of maintaining incompressibility relative to adhering closely to the training data. Selecting an appropriate λ can be challenging and often involves a trial-and-error process, especially when multiple loss terms are involved. It's important to note that when physics constraints are incorporated into the loss function, they are only enforced during training. There is no guarantee that the model will remain within the physical feasibility when deployed for predictions. Moreover, introducing extra physics-based terms into the loss function makes it more non-convex, complicating the search for an optimal minimum during training. Note also that when employing a no-physics, data-driven method, this incompressibility condition is easily violated, with the magnitude of the violation often proportional to the local size of the gradients .

This study will discuss the formulation shown in fig. 2. This *constrained* architecture differs from the no-physics baseline by incorporating a stream-function step, shown in yellow, and a non-dimensionalization step, shown in purple. In this study, the model used, shown in green, will always be a neural network (NN), of varying size depending on the problem in question; all other properties will be kept fixed. The approach is, however, general, and any other differentiable optimization model can replace the NN. The code for this model, along with all analyzed test cases, can be found here.

2.1 Incompressibility

The PCML approach involves leveraging the mathematical properties of the law that need to be enforced. Specifically, in the case of incompressible flow, the divergence-free equation implies

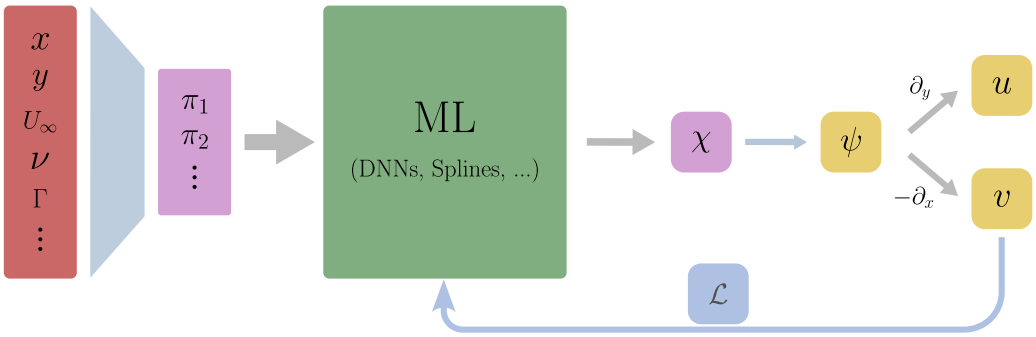

Figure 2: Architecture of a incompressibility-enforcing, non-dimensionalized neural network.

that the velocity field can be derived from a scalar field, the stream function ψ:

$$\vec{u} = \vec{\nabla} \times \vec{\psi} \qquad \text{, where } \vec{\psi} = (0, 0, \psi). \tag{3}$$

This can be achieved by setting up the model to produce a single output instead of the previous two outputs. Using automatic differentiation, the velocity field can then be derived from the curl of this single output: $\vec{u} = (\partial_y \psi, -\partial_x \psi)$, as shown in yellow in fig. 2. This approach guarantees that we always obtain an incompressible solution at any stage. Furthermore, the loss function is now once again the simple L_2 loss, $\mathcal{L} = \mathcal{L}_0$. By incorporating physics into the architecture, the model remains straightforward and therefore easier to optimize during training.

This way of incorporating incompressibility has been done recently [9, 10]. However, it has always been combined with other physics, so its effect and peculiarities are hidden. Namely, the fact this transformation implies the coupling of the two outputs u and v means that if these two targets have significantly different scales, it may be necessary to introduce an appropriate scaling transformation to ensure the stability of the model. This aspect will be discussed in more detail later on.

2.2 Non-dimensionalization

Scientific machine learning (SciML) differs from typical machine learning in that the model's solutions must obey physical laws. One consequence of this distinction is that our inputs and outputs should be dimensionless, as physical laws are invariant to the choice of units. Therefore, one can leverage the principle that nature is indifferent to the observer, allowing us to reduce the dimensionality of the space by constructing these non-dimensional variables. Unlike other dimensionality reduction techniques such as PCA, POD or auto-encoders, this transformation incurs no loss of information: we simplify the mapping to be learned at no additional cost.

To construct these non-dimensional variables, it suffices that the new set of variables forms a basis of the null-space of the units matrix. One effective method to derive them is by employing the Buckingham-Π theorem [11]. This involves selecting a set of repeating variables (always two in the cases shown in this paper, corresponding to the units of length and time) and following an algorithmic procedure to generate the $k - 2$ Π variables. It is important to note that this process should be applied to the outputs as well, ensuring that learning occurs within a non-dimensional space. This step is represented in purple in fig. 2. It is crucial to note that while selecting repeating variables simplifies the problem mathematically, not all choices will simplify the machine learning task. Certain choices may lead to convergence issues. This is primarily

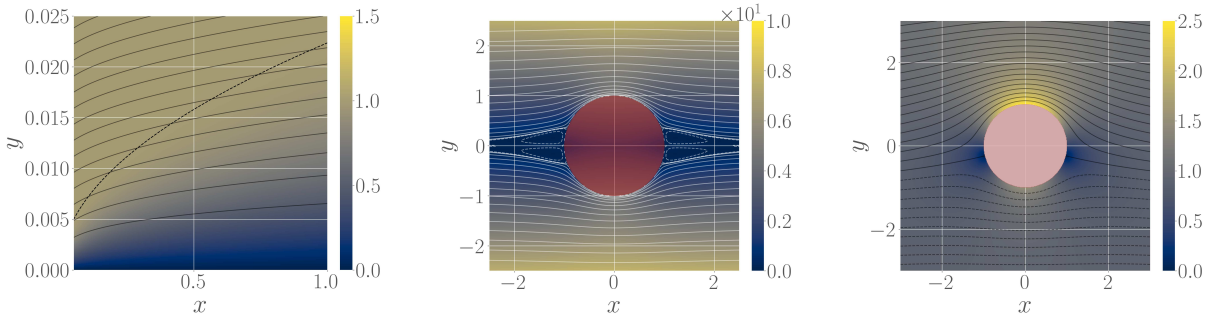

Figure 3: Norm of velocity and streamlines for the three test cases. *left*: boundary layer over a flat plate (*Blasisus solution*); *center*: constant vorticity flow around a cylinder; *right*: potential flow around a cylinder with circulation.

influenced by two factors: transformations that result in skewed distributions and the need for additional scaling of the output in some scenarios. A positive answer to these challenges would allow an algorithmic non-informed non-dimensionalization, reducing the training's complexity, when compared to other data-driven discovery of these non-dimensional groups that has recently been done [12, 13].

3 Test cases and choice of variables

To assess the model's successes and failures, three distinct test cases of increasing complexity are presented. Each of these three cases asks for different things from the model in order to evaluate its capabilities. These are: viscous flow over a flat plate, constant vorticity flow around a cylinder and potential flow around a cylinder with circulation. Stream-plots for all three cases can be seen in fig. 3. The last two cases have analytical solutions, while the first only requires solving a simple ODE. The datasets used in training were generated via uniform sampling in the parameter space of each example. An extra information was provided to the framework by adding to the output the background flow and thus having the model effectively learning the perturbations caused to this background flow, rather than the complete flow field.

3.1 Boundary layer over a flat plate

The first test case is viscous flow over a flat plate. The main challenge is to determine whether the model can learn the self-similar properties of the boundary layer and generalize to different Reynolds numbers. For simplicity, it will be used the *Blasius solution* [14], which has the following governing equation:

$$\psi = \sqrt{U_\infty \nu x} f \tag{4}$$

where f is the solution of the ODE $2f''' + f''f = 0$. In this case, the two components of the velocity have tipically significantly different scales, which poses a challenge of the PCML model, given the coupling between the two outputs. The physical quantities of interest the will be used to quantify the accuracy of the different models are the two velocity profiles, measured at 90% of the chord.

For this case, we chose $\{\nu x, U_\infty\}$ as the repeating variables. These generate the following non-dimensional variables via the Buckingham-Π theorem:

$$\pi_1 = \sqrt{\frac{U_\infty x}{\nu}}, \quad \pi_2 = y\sqrt{\frac{U_\infty}{\nu x}}; \qquad \chi = \frac{\psi}{\sqrt{U_\infty \nu x}} \tag{5}$$

It can be argued that the choice of repeating variables is somewhat informed, when we would like to drop as much as possible any expert knowledge. While this is true, it is also true that this transformation is equivalent to the more natural scaling of the coordinates by the boundary layer thickness:

$$\pi_1 = \frac{x}{\delta(x)}, \quad \pi_2 = \frac{y}{\delta(x)}; \qquad \chi = \frac{\psi}{\delta(x) U_\infty} \tag{6}$$

where $\delta(x) = \sqrt{\frac{\nu x}{U_\infty}}$ is the boundary layer thickness. Furthermore, other choices of repeating variables can work **if** combined with the appropriate output scaling. This is still target of current research and will be covered in future work. In terms of these new variables, eq. 4 takes the trivial form $\chi = f(\pi_2)$. Note that, even though the dimensionality reduction gives two input variables, π_1 and π_2, under the right output transformation, the problem is completely defined with a single input variable, π_2. This information is not given to the model.

3.2 Constant vorticity flow around a cylinder (*Fraenkel's flow*)

The second test case is of constant vorticity flow over a cylinder, also referred to as Fraenkel's flow [15]. In this case, there is a linearly varying inflow velocity profile across all y, and the interaction with the cylinder creates slow re-circulation zones near the body, posing a significant challenge to any algorithm. This paper considers the case $U_\infty = 0$, resulting in only four input variables: x, y, R and ω. The physical quantities of interest will be used to quantify the accuracy of the different models are the size of these recirculation zones, as a function of the vorticity ω. Note that, despite the re-circulation zones, this flow has an analytical solution defined on the upper plane, which can be mirrored to obtain the complete solution.

Here, the choices of repeating variables are more intuitive: the vorticity ω, being the only input with units of time, must be one of them; the other can be either x, y or R. For numerical stability reasons, the chosen set is $\{R, \omega\}$, which generates the following non-dimensional variables:

$$\pi_1 = \frac{x}{R}, \quad \pi_2 = \frac{y}{R}; \qquad \chi = \frac{\psi}{\omega R^2} \tag{7}$$

While the governing equation with the transformed variables appears to have similar complexity to the original, we now benefit from having only one effective radius and vorticity to learn, regardless of the actual radius of the cylinder and the vorticity of the inflow. Note also that, since ω is the only input variable with units of time, it can only scale the output and therefore does not directly enter the model.

3.3 Potential flow around a cylinder with circulation

The third and final test case is potential flow around a cylinder with circulation. Without the inclusion of circulation, the problem in machine learning terms would closely resemble the

boundary layer example. However, the addition of a third unconstrained dimension introduces a distinct challenge, particularly in terms of the model's generalization capabilities. The physical quantities of interest that will be used to quantify the accuracy of the different models are the lift and drag, as well as the generalization error as a function of the Reynolds number. In polar coordinates, the governing equation takes the form:

$$\psi = U_\infty r \sin(\theta)\left(1 - \frac{R^2}{r^2}\right) - \frac{\Gamma}{2\pi}\log\left(\frac{r}{R}\right) \tag{8}$$

Here, the natural choice of repeating variables is $\{R, U_\infty\}$, which generates the following non-dimensional variables:

$$\pi_1 = \frac{x}{R}, \quad \pi_2 = \frac{y}{R}, \quad \pi_3 = \frac{\Gamma}{U_\infty R}; \qquad \chi = \frac{\psi}{U_\infty R} \tag{9}$$

Note that this is the first example where we have three reduced variables, and that the incompressibility constraint only acts on π_1 and π_2. π_3 is *free*. In terms of these new variables, eq. 8 takes the form:

$$\chi = \pi_2\left(1 - \frac{1}{\pi_1^2 + \pi_2^2}\right) + \frac{\pi_3}{4\pi}\log\left(\pi_1^2 + \pi_2^2\right) \tag{10}$$

The letter π without a subscript represents the mathematical constant.

4 Results

4.1 Boundary Layer

Consider the example of Section 3.1. For this case, thanks to the insights gained from non-dimensionalization, effective results can be achieved with a minimal amount of training data and a very small neural network. In this section, we used only 500 data points, and the neural network consists of just six neurons in a single hidden layer. In all three examples, training consisted of 2000 epochs (1000 epochs with a first order optimizer, ADAM, and 1000 epochs of a second order optimizer, LBFGS); no scheduling was performed. A LayerNorm was used, and the background flow was added to the output. Table 4.1 compares the field errors of the two velocity components for the PCML model with those of the no-physics baseline model. It is important to note that if no normalization is applied to the baseline model's output, the difference in scales between u and v causes the neural network to focus solely on the larger component. An alternative approach is to scale the outputs so that they have the same order of magnitude. This method balances the errors between the two components, at the cost of increased error along u. The PCML model shows approximately a 100-fold improvement in error compared to this scaled baseline approach.

Boundary layer over a flatplate

model	u error (%)	v error (%)
physics-constrained	0.07	0.16
no-physics	1.40	56.90
no-physics (scaled)	11.04	12.45

The ideal scenario would have a uniform error across the domain. The error plots in fig. 4 show distinctive features that origin from having only 6 neurons. Additionally, these figures demonstrate how effectively the self-similarity characteristic is applied, which is expected given the choice of non-dimensionalization. Note also, that the error in v is much smaller than the error in u, similarly to their difference in scales, and thus reflecting a balanced error.

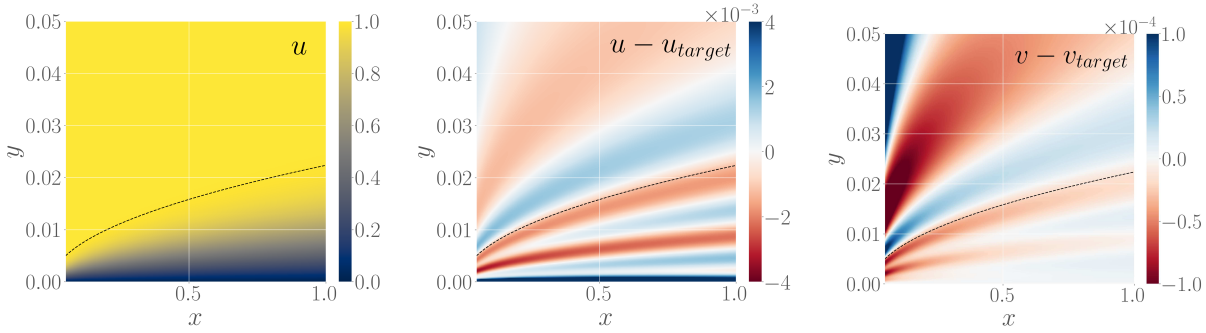

Figure 4: Boundary layer field plots. *left*: velocity field; *center*: error for u component (0.20%); *right*: error for v component (0.57%).

When it comes to the physical quantities, we are interested in seeing how well the velocity profiles and no-slip conditions are respected. Fig.5 shows this comparison between the three previously discussed models. It is clear that the PCML version perfectly captures the velocity profiles, while the baseline model diverges and fails to obey the no-slip condition on the plate.

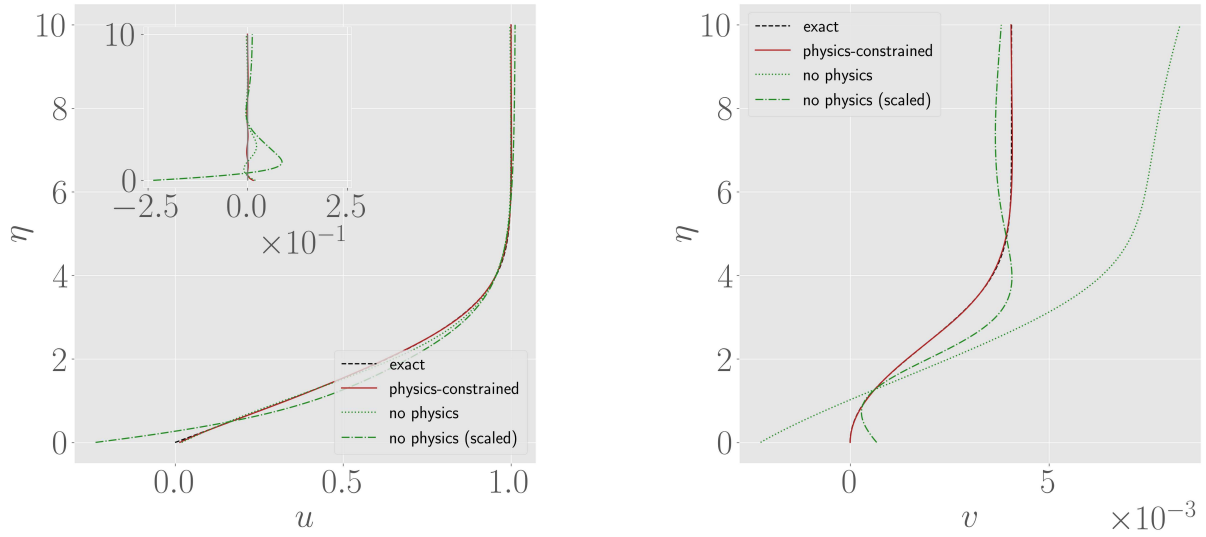

Figure 5: Velocity profiles for PCML model and no-physics baseline. *left*: u component velocity profile; *right*: v component velocity profile.

4.2 Fraenkel's Flow

Consider the example 3.2. For this case, the non-dimensionalization is less informative (it only *compresses* the R and U_∞ dimensions), thus demanding a slighter bigger dataset and NN. In this section, we used 1024 data points, and the neural network consists of 16 neurons per layer and two hidden layers. In this problem, the challenge is to accurately capture the re-circulation zones. These zones not only differ in scale compared to other flow characteristics but also exhibit significantly slower velocities. In fig. 6, it is evident that while the model captures the re-circulation zones, its accuracy is limited. The most significant errors occur near the x-axis, indicating that the chosen non-dimensionalization transformation may not be optimal. Employing a scaling method that better addresses these dissipative zones would likely yield improved results.

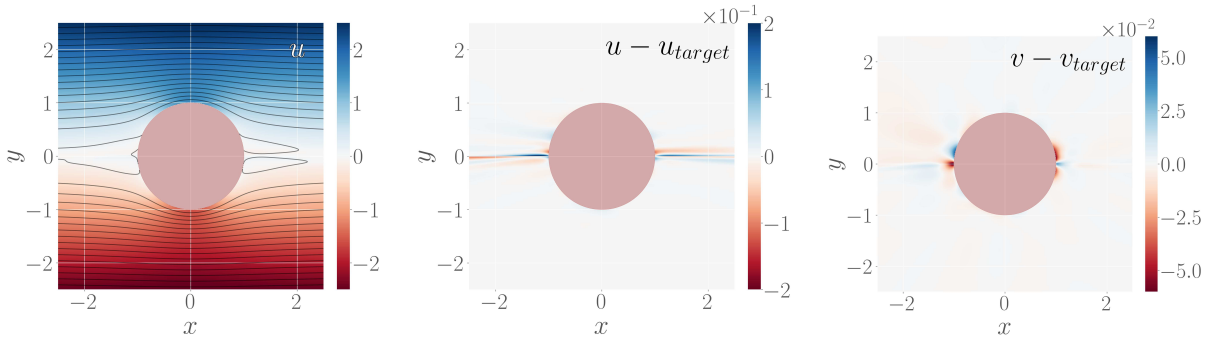

Figure 6: Fraenkel's flow field plots. *left*: velocity field; *center*: error of u component; *right*: error of v component

In this example, the physical quantity of interest is the size of the re-circulation zones. These zones can be defined by *sweeping* along the x-axis until the predicted velocity profile matches the actual profile within a given tolerance. However, as illustrated in fig. 7 (*left* and *center*), this approach is not feasible because the baseline never converges to the actual profile. Instead, we opted to represent the relative error of these profiles. The *right* figure of fig. 7 shows that the errors rapidly increase and diverge for the baseline. In contrast, the non-dimensionalization scaling by ω (which in this example is forced) ensures a constant relative error, regardless of how far we deviate from the training data.

4.3 Potential flow around a cylinder with circulation

The last example is the one presented in 3.2. Like in the previous example, the non-dimensionalization step here *only* eliminates the dependency on R and U_∞. The distinctive element of this test case is the inclusion of a third non-dimensional input, instead of two as in the previous cases. Additionally, the extra dimension is one along which the divergence condition does not apply, providing us with an unconstrained dimension. In this section, 1024 data points were used, and the neural network consists of 16 neurons per layer and two hidden layers. It is evident from fig. 8 that the largest errors are concentrated in areas with the highest gradients: the stagnation points and the suction/pressure side of the cylinder.

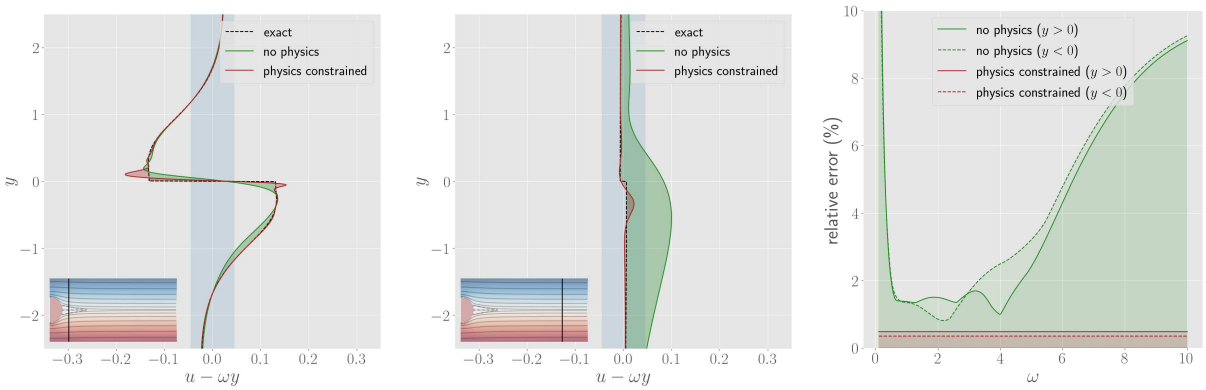

Figure 7: Velocity profile for PCML and no-physics model. The PCML approach is able to guarantee a small relative error for any inputs. Note that the background flow, $u_{bg} = \omega y$, has been subtracted in the figures *left* and *center*. *left*: velocity profile along y, at $x = 1.5R$; *center*: velocity profile along y, at $x = 8R$; *right*: relative errors of velocity profile predictions, as a function of ω.

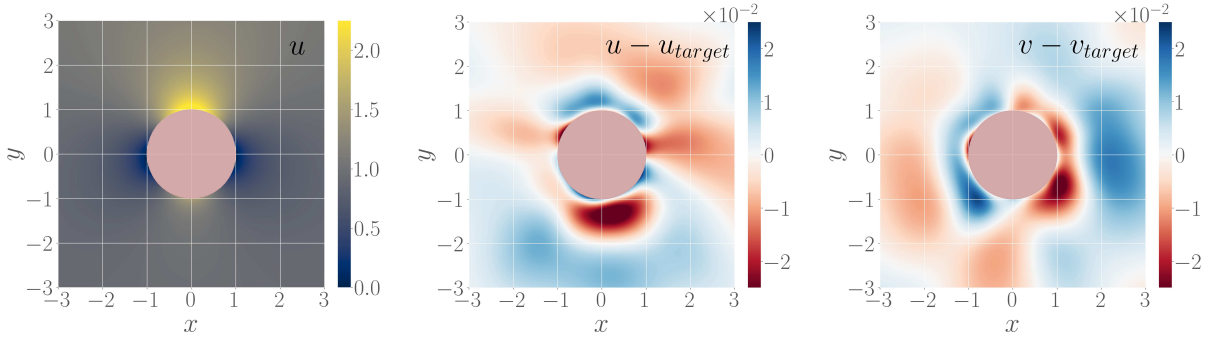

Figure 8: Potential flow around a cylinder with circulation, for $\Gamma = 3$. *left*: velocity field; *center*: error of u component (0.59%); *right*: error of v component (3.32%).

The physical quantities of interest in this example are the coefficients of lift and drag. Fig. 9 shows the results for these variables on values of circulation Γ spanning a wide range, from -15 to 15. This range is significantly wider than that of the training data, which is $[-5, 5]$. The reason for this choice is two-fold: first, we aim to evaluate the model's generalization capabilities outside the training range to identify where the model *breaks*. Second, we want to observe the model's behavior past the phase transition that occurs at $\Gamma = \pm 4\pi$, where the stagnation points merge into a single point and detach from the body. Within the training range, both the PCML model and the no-physics model accurately capture the lift. However, for drag, the PCML model performs considerably better, which reflects a superior understanding of the field's y-axis symmetry. Outside the training range, both models quickly diverge from physicality. The reasons for this divergence are not fully understood yet. One hypothesis is that the PCML model's predictions stop evolving outside the training range because the model, already meeting the imposed constraints, lacks the incentive to continue adapting and thus stagnates. Another

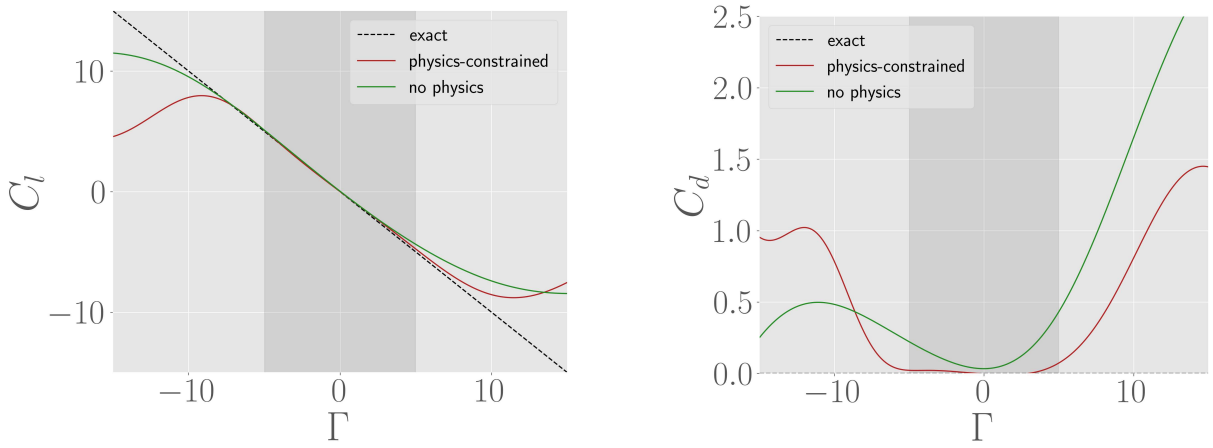

Figure 9: A comparison of the lift and drag predictions between the two models. *left*: coefficient of lift; *right*: coefficient of drag. The dark grey area represents the training range (interpolation), while the light grey area indicates the extrapolation range.

potential explanation involves the saturation of the activation function. Further analysis is needed to determine the precise cause of these large errors.

5 Discussion and conclusions

The results presented in this study clearly demonstrate the superiority of the physics-constrained approach over purely data-driven methods. Not only does it achieve a smaller global field error, but more importantly, it captures the physics of the phenomena significantly better. However, this study also highlighted that these constraints are not simply "plug-and-play" as they might be with PINNs. Hard constraints require careful implementation.

The first distinctive characteristic of the PCML model is its efficiency with data and parameters. Due to the added structure, it requires only a few data points and a limited number of degrees of freedom to develop a robust and fast model, as it can be seen on the three examples analyzed. Even in cases where non-dimensionalization provides minimal information, such as the rotating cylinder example where the primary difference among models is the incompressibility constraint, the PCML model consistently outperforms the baseline. This study also uncovered several issues that require further investigation. Firstly, the common normalization of inputs and outputs is insufficient when enforcing incompressibility via the stream function, particularly when the different output components have varying scales. Non-dimensionalization offers a natural, physical method to simplify and balance the problem. However, since this step is not unique, identifying the optimal coordinate transformation that simplifies both the mathematical and machine learning aspects of the problem is not trivial. Secondly, example 3.3 demonstrated that when there is an unconstrained dimension, the model tends to fail quickly when generalizing along that dimension. While in this example adding boundary conditions should be enough to force the model to evolve beyond the training range, it raises the question of whether there is a way to soften this issue to achieve better generalization. Additionally, the examples provided use noiseless data. Understanding how PCML models handle noise is crucial for their successful

deployment in real-world applications, and will be addressed in future work.

This study introduced a fast, accurate, and robust model that inherently enforces incompressibility and the non-dimensionalization required by physical laws. It demonstrated the model's data and computational efficiency, improved generalization capabilities, and its ability to provide physically accurate solutions. The approach is general enough to be applied to any differentiable model, including non machine learning ones, and is modular, allowing for the easy incorporation of additional elements such as boundary conditions.

REFERENCES

[1] J. Sevilla, L. Heim, A. Ho, T. Besiroglu, M. Hobbhahn, and P. Villalobos, "Compute Trends Across Three Eras of Machine Learning," in *2022 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, July 2022. ISSN: 2161-4407.

[2] D. Modesti, S. Sathyanarayana, F. Salvadore, and M. Bernardini, "Direct numerical simulation of supersonic turbulent flows over rough surfaces," *Journal of Fluid Mechanics*, vol. 942, p. A44, July 2022.

[3] S. L. Brunton, B. R. Noack, and P. Koumoutsakos, "Machine Learning for Fluid Mechanics," *Annual Review of Fluid Mechanics*, vol. 52, pp. 477–508, Jan. 2020. Publisher: Annual Reviews.

[4] J. Hestness, S. Narang, N. Ardalani, G. Diamos, H. Jun, H. Kianinejad, M. M. A. Patwary, Y. Yang, and Y. Zhou, "Deep Learning Scaling is Predictable, Empirically," Dec. 2017. arXiv:1712.00409 [cs, stat].

[5] H. Eivazi, M. Tahani, P. Schlatter, and R. Vinuesa, "Physics-informed neural networks for solving Reynolds-averaged Navier–Stokes equations," *Physics of Fluids*, vol. 34, p. 075117, July 2022.

[6] P.-Y. Chuang and L. A. Barba, "Experience report of physics-informed neural networks in fluid simulations: pitfalls and frustration," July 2022. arXiv:2205.14249 [physics].

[7] S. Wang, X. Yu, and P. Perdikaris, "When and why PINNs fail to train: A neural tangent kernel perspective," *Journal of Computational Physics*, vol. 449, p. 110768, Jan. 2022.

[8] 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*, vol. 378, pp. 686–707, Feb. 2019.

[9] A. T. Mohan, N. Lubbers, M. Chertkov, and D. Livescu, "Embedding hard physical constraints in neural network coarse-graining of three-dimensional turbulence," *Physical Review Fluids*, vol. 8, p. 014604, Jan. 2023. Publisher: American Physical Society.

[10] C. Rao, H. Sun, and Y. Liu, "Physics-informed deep learning for incompressible laminar flows," *Theoretical and Applied Mechanics Letters*, vol. 10, no. 3, pp. 207–212, 2020.

[11] E. Buckingham, "On Physically Similar Systems; Illustrations of the Use of Dimensional Equations," *Physical Review*, vol. 4, pp. 345–376, Oct. 1914. Publisher: American Physical Society.

[12] J. Bakarji, J. Callaham, S. L. Brunton, and J. N. Kutz, "Dimensionally consistent learning with Buckingham Pi," *Nature Computational Science*, vol. 2, pp. 834–844, Dec. 2022.

[13] X. Xie, A. Samaei, J. Guo, W. K. Liu, and Z. Gan, "Data-driven discovery of dimensionless numbers and governing laws from scarce measurements," *Nature Communications*, vol. 13, p. 7562, Dec. 2022.

[14] H. Blasius, "Grenzschichten in Flüssigkeiten mit kleiner Reibung," 1908.

[15] L. E. Fraenkel, "On corner eddies in plane inviscid shear flow," *Journal of Fluid Mechanics*, vol. 11, pp. 400–406, Nov. 1961.