PINNfluence: Influence Functions for Physics-Informed Neural Networks

Jonas R. Naujoks[1,*] Aleksander Krasowski[1,*] Moritz Weckbecker[1]
Thomas Wiegand[1,2,3] Sebastian Lapuschkin[1] Wojciech Samek[1,2,3,†]
René P. Klausen[1,†]

[1]Fraunhofer HHI, 10587 Berlin, Germany
[2]Technische Universität Berlin, 10587 Berlin, Germany
[3]BIFOLD – Berlin Institute for the Foundations of Learning and Data, 10587 Berlin, Germany
[†]corresponding authors: {wojciech.samek, rene.pascal.klausen}@hhi.fraunhofer.de
[*]equal contribution.

Abstract

Recently, physics-informed neural networks (PINNs) have emerged as a flexible and promising application of deep learning to partial differential equations in the physical sciences. While offering strong performance and competitive inference speeds on forward and inverse problems, their black-box nature limits interpretability, particularly regarding alignment with expected physical behavior. In the present work, we explore the application of influence functions (IFs) to validate and debug PINNs post-hoc. Specifically, we apply variations of IF-based indicators to gauge the influence of different types of collocation points on the prediction of PINNs applied to a 2D Navier-Stokes fluid flow problem. Our results demonstrate how IFs can be adapted to PINNs to reveal the potential for further studies. The code is publicly available at https://github.com/aleks-krasowski/PINNfluence.

1 Introduction

Time and time again, deep learning approaches have proven themselves to be exceptionally capable at solving problems that were considered complicated and time-consuming, yielding fast inference times bundled with strong performance. Physics-informed neural networks (PINNs) [28] represent another recent iteration of that trend applied to the realm of partial differential equations (PDEs). By incorporating prior physical knowledge in the form of differential equations [7; 13; 14; 28], they admit a wide range of possible applications such as fluid mechanics, electromagnetics, disease modelling and optics, inter alia [2; 3; 5; 23]. Despite the power of PINNs, there remains the challenge of understanding and improving model behavior when things go wrong [19; 31; 32]. PINNs minimize complex composite losses, which complicates the tracing of poor performance to specific training points or conditions; this is the main motivation for this work.

Influence functions [16] (IFs) systematically assess the contribution of individual training points to a model's behavior. Recent studies have shown that incorporating physical concepts like temporal causality improves PINN performance [10; 34], highlighting the role of underlying physical principles. By identifying key points, such as those near boundaries or critical regions, we can assess whether the PINN's learned parameters align with the physics of the problem. For many PDEs, practitioners possess an intuitive understanding of the underlying physics. For example, in magnetostatics, the magnetic field is fundamentally generated by source terms. A well-trained PINN could be expected to reflect this relation. We aim to show how domain experts can use this approach to validate the model by identifying key training points, boundary conditions or physical principles that have the greatest impact on the model's behavior. This also allows them to estimate the effect of inductive biases and observed data on the model's predictions. Notably, PINNs also provide a powerful application domain for IFs and other data attribution methods, as models and datasets are usually much smaller than those used in computer vision or natural language processing. This makes the calculation of IFs computationally feasible and the effect of individual data points on the fit of the model is expected to be greater.

Our Contributions

- We adapt influence functions to PINNs as a tool for improving their interpretability.
- We derive heuristically motivated indicators based on influence functions to showcase how they can be leveraged by domain-knowledgeable users to test and validate PINNs.
- We empirically demonstrate their usefulness in disambiguating between three different versions of PINNs applied to a Navier-Stokes problem.

2 Theoretical Background

Physics-Informed Neural Networks [28] constitute a machine-learning-based approach to solving PDEs, which are ubiquitous in physical sciences. Let $\Omega \subseteq \mathbb{R}^n$ be an open domain and consider the initial boundary value problem (IBVP)

$$\mathcal{N}[u](\boldsymbol{x}) = 0, \quad \boldsymbol{x} \in \Omega \tag{1}$$

$$\mathcal{B}[u](\boldsymbol{x}) = 0, \quad \boldsymbol{x} \in \partial\Omega \quad . \tag{2}$$

Here, $\mathcal{N}$ and $\mathcal{B}$ are differential operators acting on the solution of the PDE $u : \Omega \to \mathbb{R}^d$, where $\boldsymbol{x}$ typically represents the spatial as well as possible temporal coordinates. Note that Eq. (2) is formulated to capture i.a. Dirichlet as well as von-Neumann conditions.

The goal of PINNs is to approximate the solution of a given differential equation using a neural network $\phi(\boldsymbol{x};\theta)$, which depends on parameters $\theta \in \Theta$ and an input $\boldsymbol{x}$. These parameters are optimized by training ϕ using a composite loss function, consisting of the PDE residual and boundary conditions $\mathcal{L} = \mathcal{L}_{\text{pde}} + \mathcal{L}_{\text{bc}}$, with $\mathcal{L}_{\text{pde}} = \frac{1}{N_{\text{pde}}} \sum_{i=1}^{N_{\text{pde}}} |\mathcal{N}[\phi(\boldsymbol{x}_i;\theta)]|^2$, $\mathcal{L}_{\text{bc}} = \frac{1}{N_{\text{bc}}} \sum_{i=1}^{N_{\text{bc}}} |\mathcal{B}[\phi(\boldsymbol{x}_i;\theta)]|^2$, and where N_{pde} as well as N_{bc} are the numbers of randomly sampled collocation from Ω and $\partial\Omega$, respectively. Note that one can also consider a data-driven regression loss, which we drop for the sake of simplicity.

Influence Functions The question being addressed by influence functions [6; 11; 16] is *how would the model's behavior change if certain training points were not present in the training set?* By linearly approximating the effect of leave-one-out retraining, IFs provide a possibility of studying a model through the lens of its training data. In order to adapt the influence functions to PINNs we will slightly generalize the approach of [16; 17]. We provide a full derivation and proof, including details on all necessary assumptions, in Appendix A.1.

Suppose we have trained a model by minimizing a loss $\frac{1}{N}\Sigma_i L(\boldsymbol{x}_i;\theta)$ on a training data set $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ and derived an optimal parameter $\hat{\theta} \in \Theta$. Furthermore, consider a function $f : A \times \Theta \to \mathbb{R}$, which depends on the parameter $\theta \in \Theta$ and some other parameter $\boldsymbol{z} \in A$ (such as in [16], the loss L for a single test point $\boldsymbol{z}$). Before discussing the case of removing and adding multiple training points, we begin with the case of adding a single training point to $\mathcal{X}$. Thus, we aim to approximate $f(\boldsymbol{z};\hat{\theta}) - f(\boldsymbol{z};\hat{\theta}^+)$ if $\hat{\theta}^+$ were optimized over an amended training data set that includes an additional training point $\boldsymbol{x}^+$. This can be approximated by

$$\operatorname{Inf}_{f(\boldsymbol{z};\hat{\theta})}(\boldsymbol{x}^+) := \nabla_\theta f(\boldsymbol{z};\hat{\theta})^\top \cdot \mathcal{H}_{\hat{\theta}}^{-1} \cdot \nabla_\theta L(\boldsymbol{x}^+;\hat{\theta}) \quad , \tag{3}$$

where $\mathcal{H}_{\hat{\theta}} := \nabla_\theta^2 \frac{1}{N} \sum_{i=1}^N L(\boldsymbol{x}_i;\hat{\theta})$ denotes the Hessian w.r.t. θ. This approach can be extended to the addition and removal of multiple training data points: Let $\mathcal{X}^+ = \{\boldsymbol{x}_1^+, \ldots, \boldsymbol{x}_{m^+}^+\}$ be the set of training points we want to add and $\mathcal{X}^- = \{\boldsymbol{x}_1^-, \ldots, \boldsymbol{x}_{m^-}^-\} \subseteq \mathcal{X}$ the set we want to remove from our training set. The effect of retraining the model with added and removed training points can then be approximated by $\operatorname{Inf}_{f(\boldsymbol{z};\hat{\theta})}(\mathcal{X}^+, \mathcal{X}^-) := \sum_{i=1}^{m^+} \operatorname{Inf}_{f(\boldsymbol{z};\hat{\theta})}(\boldsymbol{x}_i^+) - \sum_{i=1}^{m^-} \operatorname{Inf}_{f(\boldsymbol{z};\hat{\theta})}(\boldsymbol{x}_i^-)$. Hence, we can study the influence of single sampling points as well as regions on loss functions, any type of predictions or derived quantities. By studying the influence resulting therefrom, one can compare the model's behavior with physical intuition. Using this method, we can also estimate the effect of adding and removing entire loss terms for composite losses, as well as several other types of manipulations (see Eqs. (15), (18), (21) and (23) in the appendix).

3 Experiments

Navier-Stokes Problem The Navier-Stokes equations describe the motion of viscous Newtonian fluids and represent the cornerstone of fluid dynamics. Consequently, they are widely used in various applications. In the present work, we focus on a laminar, incompressible and time-independent fluid flow around a cylinder in a rectangular cavity with open lateral boundaries in 2D [30], see Fig. 1. The stationary equations of motions as well as the continuity equation are given as follows:

$$(\boldsymbol{u} \cdot \nabla)\boldsymbol{u} + \frac{1}{\rho}\nabla p - \nu \nabla^2 \boldsymbol{u} = 0\,, \qquad \operatorname{div} \boldsymbol{u} = 0\,, \tag{4}$$

where $\boldsymbol{u}(x, y) \in \mathbb{R}^2$ denotes the velocity field, $p(x, y) \in \mathbb{R}$ represents the pressure, ρ is the density, and ν is the kinematic viscosity. The domain is given by $\Omega = (x_{\min}, x_{\max}) \times (y_{\min}, y_{\max}) \setminus C_r$, where C_r denotes the cylinder with radius r. The boundary conditions at the bottom, top, and cylinder are given by $\boldsymbol{u} = 0$. The inflow condition on the left has a parabolic profile. For the rightward outflow condition, we use a directional "do nothing" condition [24]: $\nu \nabla u_1(x_{\max}, y) - (\frac{p}{\rho}, 0)^\top = 0$. The used parameters of the geometry and PDE are given in A.2. The task of the PINN is then to predict $\boldsymbol{u}(x, y)$ and $p(x, y)$.

Experimental Setup We train fully-connected PINNs consisting of 4 layers with 64 nodes each with GELU activation functions and 3 outputs. These correspond to the solution of Eq. (4) that we want to predict: u_1, u_2 and p. For the training data set, we randomly draw N_{pde} points inside the domain Ω and N_{bc} from the boundary using Hammersley sampling. In essence, these are pairs of (x, y) coordinates as we consider a 2D PDE. We train three models, (1) *good* model ϕ_{good} with the correct PDE and $N_{\text{pde}} = 7500$ and $N_{\text{bc}} = 2500$ that solves the task well compared to a reference solution [33]; (2) a *broken* model ϕ_{broken} with the same number of training points but with the $(u_1 \cdot \nabla)u_1$ part of Eq. (4) missing, thus trained on wrong physics; (3) and a *bad* model ϕ_{bad} with the correct PDE but with $N_{\text{pde}} = 1500$ and $N_{\text{bc}} = 500$, such that it consequently performs badly. Each model is trained in two phases. First, `Adam` [15] is used as an optimizer for a total of 100k steps, followed by further refining the model parameter's using `L-BFGS` [21] for up to 25k steps. We slightly adapted the `DeepXDE` library [22] to allow for more fine-grained gradient manipulation while using `PyTorch` [26] as our backend. For the test set, we draw a total of 36934 points from Ω and 1288 points from $\partial\Omega$ [33]. The influences are calculated using the `NaiveInfluenceFunction` from `captum` [18].

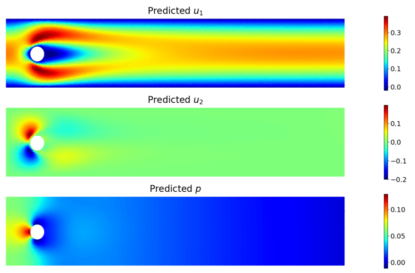


Figure 1: Trained model (ϕ_{good}) with predictions: u_1, u_2, p (top to bottom).

Empirical Indicators In this study, we aim to answer the following question using influence functions: *Can we test whether the PINN has learned certain aspects of the underlying physical principles of the Navier-Stokes problem?* Since predictions alone do not reveal the influence of training data or boundary conditions, we devise a set of heuristic indicators.

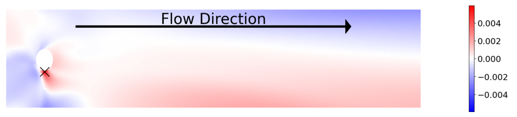


Figure 2: Influence heatmap on the prediction of u_1 of a single training point ($\times$) close to the cylinder (bottom).

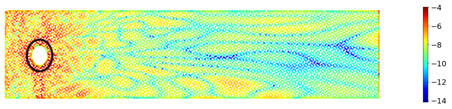
We propose two IF-based indicators that are designed to capture aspects of the physical processes inherent to the PDE (4) and to serve as exemplary metrics showcasing how expert users can design tests to validate PINNs against their prior knowledge of the problem. As a first indication metric, we concentrate on the flow direction of the Navier-Stokes problem: the inflow is given at the left boundary of the domain and the outflow lies on the opposite side. Figure 2 shows the influence pattern of a training point close to the cylinder.

We thus test a model on whether its influence patterns reflect this behavior. To define the directional indicator (DI), which captures the relevance share associated with the collocation point $\boldsymbol{x}_{\text{test}}$ on the test points that follow in the direction of the flow (i.e. having larger x-component), we write:

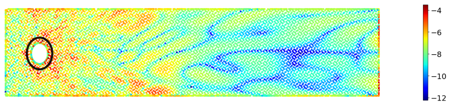
$$\text{DI}(\boldsymbol{x}_{\text{test}}) := \frac{\sum_{\boldsymbol{x}_{\text{train}}: x_{\text{train}} > x_{\text{test}}} \left| \text{Inf}_{f(\boldsymbol{x}_{\text{test}};\hat{\theta})}(\boldsymbol{x}_{\text{train}}) \right|}{\sum_{\boldsymbol{x}_{\text{train}}} \left| \text{Inf}_{f(\boldsymbol{x}_{\text{test}};\hat{\theta})}(\boldsymbol{x}_{\text{train}}) \right|} \in [0, 1] \tag{5}$$

A larger value of DI implies that a given test point mainly influences the prediction further down in the direction of the flow rather than before and implies that the model has learned this underlying physical phenomenon of directed flows.

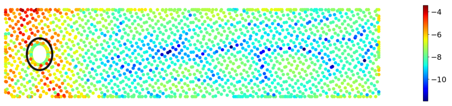
As a second indicator, we are interested in the fraction of influence that is associated with a given object in the input domain. In the present Navier-Stokes problem, the cylinder plays a central role in the flow field: it obstructs the flow. Thus, the overall velocity and pressure fields should be heavily influenced by its position. To this end, we devise a region identifier, which is designed to capture the influence associated with a given object or region in the domain (in our case around the cylinder).



(a) ϕ_{good}



(b) ϕ_{broken}



(c) ϕ_{bad}

Figure 3: Average log values of $|\text{Inf}_{\sum_i \|u(\boldsymbol{x}_i;\theta)\|}(\boldsymbol{x})|$ over the domain $\overline{\Omega}$. The area of $C_{1.5r}$ is outlined in black.

For the region indicator (RI), which sums up the relevance of removing a region $\Xi \subseteq \overline{\Omega}$, we write:

$$\text{RI}(\Xi) := \frac{\sum_{\boldsymbol{x}_{\text{train}} \in \Xi} \left| \text{Inf}_{f(\boldsymbol{x}_{\text{test}};\hat{\theta})}(\boldsymbol{x}_{\text{train}}) \right|}{\sum_{\boldsymbol{x}_{\text{train}}} \left| \text{Inf}_{f(\boldsymbol{x}_{\text{test}};\hat{\theta})}(\boldsymbol{x}_{\text{train}}) \right|} \in [0, 1] \tag{6}$$

A higher value of RI means that the region in question, such as the cylinder in our case, has an increased influence on the prediction for all other points. This is to be expected for a model that aims to accurately reflect the physics. In this investigation, we choose $\Xi = C_{1.5r}$, meaning that we remove the area corresponding to a circle with 1.5 times the radius of the cylinder, see Fig. 3.

Results We evaluate the three different iterations of PINNs on the Navier-Stokes task. Figures 3a to 3c show the influence of individual training points on the whole test set, as heatmaps. For ϕ_{good}, the most influential points are distributed close to the inflow, on the outflow and especially around the cylinder. This is expected, as these areas determine the fluid flow. Furthermore, we observe that for ϕ_{bad}, the upper left boundary is overly influential. In contrast, ϕ_{broken}'s influences are again distributed mostly around the cylinder. In Table 1, the aggregated mean values of the indicators are presented. Both ϕ_{good} and ϕ_{bad} show stronger DI performance. This suggests that both models have successfully learned to capture the dependence of the fluid flow on preceding points. When it comes to the RI indicator, this trend is even clearer. Here, the good model achieves superior performance, which suggests that the cylinder here has the biggest absolute influence, reflecting its importance for the fluid flow. It is important to

Table 1: Mean values for the Direction $\overline{\text{DI}}$ and Region $\overline{\text{RI}}$ Indicators, aggregated over all $\boldsymbol{x}_{\text{test}}$. A higher score indicates a closer alignment with the indicator's assumptions.

	$\overline{\text{DI}}$			$\overline{\text{RI}}(C_{1.5r})$		
$f(x;\theta)$	ϕ_{good}	ϕ_{bad}	ϕ_{broken}	ϕ_{good}	ϕ_{bad}	ϕ_{broken}
u_1	**0.8**	**0.8**	0.71	**0.23**	0.18	0.13
u_2	**0.82**	0.77	0.69	**0.26**	0.17	0.14
p	0.83	**0.87**	0.79	**0.28**	0.18	0.22
$\|\boldsymbol{u}\|$	**0.8**	0.79	0.71	**0.23**	0.18	0.13
$\mathcal{L}_{\text{pde}}$	**0.79**	0.76	0.73	**0.21**	0.14	0.16
$\mathcal{L}_{\text{bc}}$	**0.75**	0.71	0.65	**0.25**	0.16	0.15
$\mathcal{L}$	**0.79**	0.76	0.73	**0.21**	0.14	0.16

note that these indicators are tools for investigating model behavior. Performing well on an indicator does not imply that the model is entirely correct, but rather that it does not fail in the specific aspect the indicator evaluates. Expert knowledge is essential for designing meaningful indicators and further investigation is needed to validate their robustness. Additionally, if the indicator's assumptions are flawed, the results may be misleading and fail to provide sensible insights.

4 Conclusion

We showcased how influence functions can help to evaluate PINNs. In particular, we formulated heuristically motivated indicators designed to test certain physical concepts. By applying this method to a Navier-Stokes problem, we observed that the best-performing model consistently aligned with these indicators, confirming their utility in assessing model performance. Still, further exploration of IF-based markers is indicated to rigorously validate the broad applicability of the presented approach. Here, applying novel evaluation frameworks such as [1] could be helpful in future examinations. Using influence functions to improve PINN training through methods like resampling training points [20], enforcing temporal causality [34], or dynamically reweighing loss terms [4] – combined with IF-based training time techniques like `TracIN` [27] – presents a promising direction for future research. Additionally, further studies on the influence on parts of the loss as well as on different physical quantities, e.g. the vorticity, as well as applying the framework to different PDEs such as the heat equation, Maxwell equations and others appear to be logical next steps.

Acknowledgments

The authors want to thank Galip Ümit Yolcu for fruitful discussions. This work was supported by the Fraunhofer Internal Programs under Grant No. PREPARE 40-08394.

Code Availability

The code is available at https://github.com/aleks-krasowski/PINNfluence.

References

[1] Dilyara Bareeva, Galip Ümit Yolcu, Anna Hedström, Niklas Schmolenski, Thomas Wiegand, Wojciech Samek, and Sebastian Lapuschkin. Quanda: An Interpretability Toolkit for Training Data Attribution Evaluation and Beyond, October 2024. arXiv:2410.07158.

[2] Andrés Beltrán-Pulido, Ilias Bilionis, and Dionysios Aliprantis. Physics-informed neural networks for solving parametric magnetostatic problems. *IEEE Transactions on Energy Conversion*, 37(4):2678–2689, December 2022. arXiv:2202.04041.

[3] Sarah Berkhahn and Matthias Ehrhardt. A physics-informed neural network to model COVID-19 infection and hospitalization scenarios. *Advances in Continuous and Discrete Models*, 2022 (1):61, October 2022.

[4] Rafael Bischof and Michael Kraus. Multi-Objective Loss Balancing for Physics-Informed Deep Learning. October 2021. arXiv:2110.09813.

[5] Shengze Cai, Zhiping Mao, Zhicheng Wang, Minglang Yin, and George Em Karniadakis. Physics-informed neural networks (PINNs) for fluid mechanics: A review. *Acta Mechanica Sinica*, 37(12):1727–1738, January 2022. arXiv:2105.09506.

[6] R. Dennis Cook and Sanford Weisberg. *Residuals and Influence in Regression*. Chapman and Hall, 1982. ISBN 0-412-24280-0.

[7] Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. Scientific Machine Learning Through Physics–Informed Neural Networks: Where we are and What's Next. *Journal of Scientific Computing*, 92(3):88, September 2022. arXiv:2201.05624.

[8] Yann N. Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 2933–2941, Cambridge, MA, USA, December 2014. MIT Press. arXiv:1406.2572.

[9] Klaus Fritzsche and Hans Grauert. *From Holomorphic Functions to Complex Manifolds*, volume 213 of *Graduate Texts in Mathematics*. Springer, New York, NY, 2002. ISBN 978-1-4419-2983-9 978-1-4684-9273-6.

[10] Jia Guo, Haifeng Wang, Shilin Gu, and Chenping Hou. TCAS-PINN: Physics-informed neural networks with a novel temporal causality-based adaptive sampling method. *Chinese Physics B*, 33(5):050701, April 2024.

[11] Frank R. Hampel. The Influence Curve and its Role in Robust Estimation. *Journal of the American Statistical Association*, 69(346):383–393, June 1974.

[12] Hidehiko Ichimura and Whitney K. Newey. The Influence Function of Semiparametric Estimators. *Quantitative Economics*, 13(1):29–61, July 2021. arXiv:1508.01378.

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

[14] Sung Wook Kim, Iljeok Kim, Jonghwan Lee, and Seungchul Lee. Knowledge Integration into deep learning in dynamical systems: an overview and taxonomy. *Journal of Mechanical Science and Technology*, 35(4):1331–1342, April 2021.

[15] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, January 2017. arXiv:1412.6980.

[16] Pang Wei Koh and Percy Liang. Understanding Black-box Predictions via Influence Functions. *Proceedings of Machine Learning Research*, 70:1885–1894, 06–11 Aug 2017. arXiv:1703.04730.

[17] Pang Wei Koh, Kai-Siang Ang, Hubert H. K. Teo, and Percy Liang. On the Accuracy of Influence Functions for Measuring Group Effects. *Advances in Neural Information Processing Systems*, 22, December 2019. arXiv:1905.13289.

[18] Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, and Orion Reblitz-Richardson. Captum: A unified and generic model interpretability library for PyTorch, September 2020. arXiv:2009.07896.

[19] Aditi S. Krishnapriyan, Amir Gholami, Shandian Zhe, Robert M. Kirby, and Michael W. Mahoney. Characterizing possible failure modes in physics-informed neural networks. *Advances in Neural Information Processing Systems*, 34:26548–26560, December 2021. arXiv:2109.01050.

[20] Gregory Kang Ruey Lau, Apivich Hemachandra, See-Kiong Ng, and Bryan Kian Hsiang Low. PINNACLE: PINN Adaptive ColLocation and Experimental points selection, April 2024. arXiv:2404.07662.

[21] Dong C. Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1):503–528, August 1989.

[22] Lu Lu, Xuhui Meng, Zhiping Mao, and George Em Karniadakis. DeepXDE: A Deep Learning Library for Solving Differential Equations. *SIAM Review*, 63(1):208–228, January 2021.

[23] V. Medvedev, A. Erdmann, and A. Rosskopf. Modeling of Near-and Far-Field Diffraction from EUV Absorbers Using Physics-Informed Neural Networks. In *2023 Photonics & Electromagnetics Research Symposium (PIERS)*, pages 297–305, July 2023.

[24] Malte Braack And Piotr Boguslaw Mucha. Directional Do-Nothing Condition for the Navier-Stokes Equations. *Journal of Computational Mathematics*, 32(5):507–521, June 2014.

[25] Oswaldo Rio Branco de Oliveira. The Implicit and the Inverse Function theorems: easy proofs, December 2012. arXiv:1212.2066.

[26] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems*, 32, December 2019. arXiv:1912.01703.

[27] Garima Pruthi, Frederick Liu, Mukund Sundararajan, and Satyen Kale. Estimating Training Data Influence by Tracing Gradient Descent. *Advances in Neural Information Processing Systems*, 33:19920–19930, December 2020. arXiv:2002.08484.

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

[29] Andrea Schioppa, Katja Filippova, Ivan Titov, and Polina Zablotskaia. Theoretical and Practical Perspectives on what Influence Functions Do. *Advances in Neural Information Processing Systems*, 36, December 2024. arXiv:2305.16971.

[30] M. Schäfer, S. Turek, F. Durst, E. Krause, and R. Rannacher. Benchmark Computations of Laminar Flow Around a Cylinder. In Ernst Heinrich Hirschel, Kozo Fujii, Bram Van Leer, Michael A. Leschziner, Maurizio Pandolfi, Arthur Rizzi, Bernard Roux, and Ernst Heinrich Hirschel, editors, *Flow Simulation with High-Performance Computers II*, volume 48, pages 547–566. Vieweg+Teubner Verlag, Wiesbaden, 1996. ISBN 978-3-322-89851-7 978-3-322-89849-4.

[31] Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5): A3055–A3081, 2021. arXiv:2001.04536.

[32] Sifan Wang, Xinling Yu, and Paris Perdikaris. When and why PINNs fail to train: A neural tangent kernel perspective. *Journal of Computational Physics*, 449:110768, January 2022.

[33] Sifan Wang, Shyam Sankaran, Hanwen Wang, and Paris Perdikaris. An Expert's Guide to Training Physics-informed Neural Networks, August 2023. arXiv:2308.08468.

[34] Sifan Wang, Shyam Sankaran, and Paris Perdikaris. Respecting causality for training physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 421: 116813, March 2024. arXiv:2203.07404.

A Appendix / Supplemental Material

A.1 Influence Functions

In this part, we aim to formulate the key steps for influence functions, initially introduced in 1974 [11] to study robust estimators, and which were recently applied to neural networks [16; 17]. However, as already observed in [29], the functions typically encountered in neural networks require a more exhaustive analysis. Therefore, we will carefully redraw the results proposed in [16] and generalize them slightly for the application to PINNs. In particular, we will formulate influence functions in a manner that enables the study of the effect of various modifications of the problem (changes in the training set, in the loss weighting or in the PDE) on the prediction and any derived quantity. Furthermore, we will provide rigorous proofs for the statements made in Section 2.

We will start with a crucial lemma for influence functions, which was already noted in [6]. As remarked in [29], influence functions are usually stated for strictly convex functions[1] and relates to global minima. However, in practice, we will typically only reach saddle points [8], which is the reason why the following lemma is extended to stationary points.

[1]Note, that for the application of influence functions with global minima (e.g. [16]), strict convexity is a necessary, but not a sufficient criterion.

Lemma 1. *For $\Theta \subseteq \mathbb{R}^n$ and $U \subseteq \mathbb{R}$ open, let $g : \Theta \times U \to \mathbb{R}$ be a twice continuously differentiable function for which there exists an $(\theta_0, \epsilon_0) \in \Theta \times U$ such that θ_0 is a non-degenerate stationary point of the function $g(\cdot, \epsilon_0)$, i.e. $\nabla_\theta g(\theta_0, \epsilon_0) = 0$ and the Hessian $\mathcal{H}_\theta(\theta_0, \epsilon_0) := \nabla^2_\theta g(\theta_0, \epsilon_0)$ is invertible. Then there exists a non-empty, open set $U_0 \subseteq U$ with $\epsilon_0 \in U_0$ such that $g(\cdot, \epsilon)$ has a unique stationary point in the neighbourhood of θ_0 for each $\epsilon \in U_0$ and we call the function describing those stationary points by $h : U_0 \to \Theta$. This function h is continuously differentiable with derivative*

$$\frac{\partial h(\epsilon)}{\partial \epsilon} = -\mathcal{H}_\theta(h(\epsilon), \epsilon)^{-1} \cdot \nabla_\theta \frac{\partial}{\partial \epsilon} g(\theta, \epsilon)\Big|_{\theta = h(\epsilon)} \quad . \tag{7}$$

If g is analytic in a neighbourhood of (θ_0, ϵ_0), then we can choose U_0 such that h is analytic as well. Furthermore, let $B \subseteq U$ be an open, connected set with $\epsilon_0 \in B$, such that for every $\epsilon \in B$ there exists a non-degenerated stationary point of $g(\cdot, \epsilon)$. Then h is extendable to B and this function $h : B \to \Theta$ is continuous differentiable as well with Eq. (7).

Proof. By $s : \Theta \times U \to \mathbb{R}^n$ we denote the gradient $s(\theta, \epsilon) := \nabla_\theta g(\theta, \epsilon)$, which is continuously differentiable by assumption. Since (θ_0, ϵ_0) is a stationary point, this will be a root point of s. By the implicit function theorem (see e.g. [25, thm. 2]) we can conclude that there is an open set $U_0 \subseteq U$ around ϵ_0 such that there exists a unique function $h : U_0 \to \Theta$ with $h(\epsilon_0) = \theta_0$ and $s(h(\epsilon), \epsilon) = 0$ for all $\epsilon \in U_0$. Therefore, for any $\epsilon \in U_0$, there is only one unique stationary point in the neighbourhood of θ_0. Furthermore, h is continuously differentiable, and its derivative is given by Eq. (7).

If g is analytic, then the same holds for its gradient s. By the holomorphic implicit function theorem (see e.g. [9, sec. 7.6]) the function h is also analytic.

If the assumptions of the lemma are satisfied for any $\epsilon \in B$, we can apply the lemma in all of these points. The function $h : B \to \Theta$ is continuous differentiable, since it is continuous differentiable in every point and Eq. (7) extends to all points in B. Moreover, $h(B)$ is connected, since h is continuous and B is connected by assumption. $\square$

In a general sense, influence functions study the effect of a perturbation of a function, such as a loss function, through a functional, such as the argmin operator. Therefore, we will consider a functional $F : C(\Theta \times U) \times U \to \Theta$ and a continuous function $g : \Theta \times U \to \mathbb{R}^m$ with $\Theta \subseteq \mathbb{R}^n$ and a neighbourhood $U \subseteq \mathbb{R}$ containing the origin. We assume that $g_0(\theta) := g(\theta, 0)$ describes the initial problem, whereas $g(\theta, \epsilon)$ for $\epsilon > 0$ corresponds to a perturbed problem, e.g. changes in the training set. The influence function can then be formally expressed as

$$\operatorname{Inf} F[g(\cdot, \cdot)] := \lim_{\epsilon \to 0} \frac{F[g(\cdot, 0)] - F[g(\cdot, \epsilon)]}{\epsilon} \tag{8}$$

provided that this limit exists.

In the case of stationary points (in particular for global minima) we obtain a much more convenient expression for the influence function by means of Lemma 1.

Theorem 1. *Let F be the functional of stationary points[2] (alternatively the argmin operator) and let $g_0 : \Theta \to \mathbb{R}$ and $\kappa : \Theta \to \mathbb{R}$ be twice continuously differentiable functions. Assume that g_0 has a non-degenerated, stationary point $\hat{\theta} = F[g_0]$ (a global minimum, respectively). Then the influence function is given by*

$$\operatorname{Inf} F[g_0 + \epsilon\kappa] = \mathcal{H}_{\hat{\theta}}^{-1} \nabla_\theta \kappa(\hat{\theta}) \quad , \tag{9}$$

with the Hessian $\mathcal{H}_{\hat{\theta}} = \nabla^2_\theta g_0(\hat{\theta})$.

Usually we are interested in the effect of a change in the training process on the evaluation of a function f, such as the loss or the prediction. Therefore, before proving Theorem 1, we will add the following direct consequence of the definition (8).

Corollary 1. *Let $f : \Theta \to \mathbb{R}^d$ be a differentiable function. Then the influence function satisfies a chain rule*

$$\operatorname{Inf} f(F[g(\cdot, \cdot)]) = \nabla_\theta f(\theta)\Big|_{\theta = F[g(\cdot, 0)]} \cdot \operatorname{Inf} F[g(\cdot, \cdot)] \quad . \tag{10}$$

[2]To be a functional, F have to map every function to one stationary point or one minimum, respectively. Hence, we might have to adjust e.g. the domain of the input functions to get unique stationary points.

Proof. If the limit in (8) exists, it is nothing else than the derivative. Thus, by applying Lemma 1, the derivative exists and is given by

$$\operatorname{Inf} F[g_0 + \epsilon\kappa] = - \left.\frac{dF(g_0 + \epsilon\kappa)}{d\epsilon}\right|_{\epsilon=0} = \mathcal{H}_{\hat\theta}^{-1} \nabla_\theta \kappa(\hat\theta) \quad . \tag{11}$$

The corollary directly follows from the chain rule for differentiation. □

Note that the left-hand side of Eq. (9) is the Gâteaux derivative of F at g_0 in the direction of κ, which provides a convenient mathematical setting to study influence functions for linear perturbations. We refer to [12] for the study of those relations.

In practice, we often use infinitesimal changes expressed by $\operatorname{Inf} f(F[g(\cdot,\epsilon)])$ to approximate some finite change in $f(F[g(\cdot,\epsilon)])$ for an $\epsilon > 0$. From this perspective, the influence function is the linear approximation term of a Taylor series

$$f(F[g(\cdot,\epsilon)]) = f(F[g(\cdot,0)]) - \epsilon \cdot \operatorname{Inf} f(F[g(\cdot,\cdot)]) + r(\epsilon) \quad , \tag{12}$$

where the remainder term $r(\epsilon)$ vanishes with

$$\lim_{\epsilon\to 0} \frac{r(\epsilon)}{\epsilon^2} = 0 \tag{13}$$

If, moreover, the functions f and g are analytic, then by Lemma 1, also $f(F[g(\cdot,\epsilon)])$ is analytic and the Taylor series converges uniformly for every compact subset of its convergence region, i.e. for any compact subset of $\mathbb{R}$ we have a constant $C > 0$ such that the remainder is bounded there by

$$|r(\epsilon)| \leq C\epsilon^2 \quad . \tag{14}$$

In the remaining part of this section, we will study concrete settings for influence functions, which appears in particular for PINNs, and give convenient reformulations of the statement of Theorem 1.

Adding & removing training points: Let $\mathcal{X}$ be a finite, non-empty set of training points and consider a finite set of points $\mathcal{X}^+$, which we would like to add to the training set, as well as a set of points $\mathcal{X}^- \subsetneq \mathcal{X}$, which we would like to remove from the training set. Moreover, let $L(\boldsymbol{x},\cdot) : \Theta \to \mathbb{R}$ be a twice continuously differentiable loss function assigning a value to each (potential) training point of $\mathcal{X} \cup \mathcal{X}^+$. The empirical risk for our original dataset is given by $g_0(\theta) = \frac{1}{N}\sum_{\boldsymbol{x}\in\mathcal{X}} L(\boldsymbol{x};\theta)$ and let $\kappa(\theta) = \sum_{\boldsymbol{x}\in\mathcal{X}^+} L(\boldsymbol{x};\theta) - \sum_{\boldsymbol{x}\in\mathcal{X}^-} L(\boldsymbol{x};\theta)$ be the perturbation in risk corresponding to the change in the training set. N denotes the size of training points in $\mathcal{X}$. Assume that there exists a stationary point[3] (a global minimum, respectively) $\hat\theta = F[g_0]$ inside Θ and let F be the functional which maps to those stationary points. In addition, let $f(\boldsymbol{z},\cdot) : \Theta \to \mathbb{R}^m$ be an arbitrary continuously differentiable function (e.g. the loss or the prediction), possibly with a further dependency expressed by a variable $\boldsymbol{z}$ (e.g. a test point).

According to Theorem 1, we can then approximate counterfactual effect of training on a perturbed training set $\mathcal{X}' = (\mathcal{X} \cup \mathcal{X}^+) \setminus \mathcal{X}^-$ by

$$f(\boldsymbol{z};\hat\theta) - f(\boldsymbol{z};\hat\theta') = \frac{1}{N} \operatorname{Inf}_{f(\boldsymbol{z};\hat\theta)}(\mathcal{X}^+,\mathcal{X}^-) + r\big(N^{-2}\big) \quad , \tag{15}$$

where we introduce the shorthand notation

$$\begin{aligned} \operatorname{Inf}_{f(\boldsymbol{z};\hat\theta)}(\mathcal{X}^+,\mathcal{X}^-) &:= \operatorname{Inf} f(\boldsymbol{z}; F[g_0 + \epsilon\kappa]) \\ &= \nabla_\theta f(\boldsymbol{z};\hat\theta) \cdot \mathcal{H}_{\hat\theta}^{-1} \cdot \nabla_\theta \left(\sum_{\boldsymbol{x}\in\mathcal{X}^+} L(\boldsymbol{x};\hat\theta) - \sum_{\boldsymbol{x}\in\mathcal{X}^-} L(\boldsymbol{x};\hat\theta) \right) \quad , \end{aligned} \tag{16}$$

which is simply the signed sum of the influences of the individual training points due to the linearity of Influence Functions.

[3]Thus, in practice, we assume that our training has progressed to the stage where it has reached a stationary point in the loss landscape.

Removing a complete loss term: Moreover, we can also remove a complete loss term and study the influence of this operation to the training. For this, consider a total loss as a sum of k loss terms

$$g_0(\theta) = \mathcal{L}(\mathcal{X};\theta) = \sum_{j=1}^{k} \mathcal{L}_j(\mathcal{X};\theta) = \frac{1}{N}\sum_{\boldsymbol{x}\in\mathcal{X}}\sum_{j=1}^{k} L_j(\boldsymbol{x};\theta) \tag{17}$$

and a perturbation direction $\kappa(\theta) = -\sum_{\boldsymbol{x}\in\mathcal{X}} L_i(\boldsymbol{x};\theta)$. Similar to the previous paragraph, we write $\hat{\theta} = F[g_0]$ for the optimal parameter without the perturbation and $\hat{\theta}_{\cancel{\mathcal{L}_i}}$ for the optimal parameter, when the i-th loss is removed. We obtain

$$f(\boldsymbol{z};\hat{\theta}) - f(\boldsymbol{z};\hat{\theta}_{\cancel{\mathcal{L}_i}}) = \frac{1}{N}\operatorname{Inf} f(\boldsymbol{z}; F[g_0 + N^{-1}\kappa]) + r\big(N^{-2}\big) \tag{18}$$

with

$$\operatorname{Inf} f(\boldsymbol{z}; F[g_0 + N^{-1}\kappa]) = -\nabla_\theta f(\boldsymbol{z};\hat{\theta}) \cdot \mathcal{H}_{\hat{\theta}}^{-1} \cdot \nabla_\theta \sum_{\boldsymbol{x}\in\mathcal{X}} L_i(\boldsymbol{x};\hat{\theta}) \quad . \tag{19}$$

Influence of loss weights: In analogous fashion, we can easily extend this framework to study the effect of upweighting or downweighting individual loss terms. This is in particular interesting for PINNs because a wrong weighting of the loss terms is known to reduce the accuracy of PINNs drastically [31].

Thus, assume that for some weights $w_j > 0$ we consider a total loss in the form of

$$g_0(\theta) = \mathcal{L}(\mathcal{X};\theta) = \sum_{j=1}^{k} w_j \mathcal{L}_j(\mathcal{X};\theta) \tag{20}$$

where we now perturb the weight w_i with $\kappa(\theta) = w_i \mathcal{L}_i(\mathcal{X};\theta)$. The influence of changing the weight w_i is then given by

$$\operatorname{Inf} f(\boldsymbol{z}; F[g_0 + \epsilon\kappa]) = w_i \nabla_\theta f(\boldsymbol{z};\hat{\theta}) \cdot \mathcal{H}_{\hat{\theta}}^{-1} \cdot \nabla_\theta \mathcal{L}_i(\mathcal{X},\hat{\theta}) \quad . \tag{21}$$

In comparison to the two previous discussed cases, we can also consider infinitesimal changes of the weight so that we can omit a discussion of the error bounds.

Influence of parameters in the PDE: As a last option to use influence functions for PINNs, we want to discuss the impact of parameters in the PDE. Therefore, we will consider a basic objective of the form

$$g(\theta,\rho) = \frac{1}{N}\sum_{\boldsymbol{x}\in\mathcal{X}} L(\boldsymbol{x};\rho;\theta) \tag{22}$$

with some additional parameter $\rho \in \mathbb{R}$, and we will assume that $\mathcal{L}$ is also continuously differentiable w.r.t. ρ. To apply Lemma 1, we will assume, that there is an optimal or stationary parameter $\hat{\theta} = F[g(\cdot,\rho_0)]$ at a certain value of the parameter ρ_0 we are interested in. Then, the influence of the parameter ρ to the optimal parameters $\hat{\theta}$ at a certain point ρ_0 is given by

$$\begin{aligned}\operatorname{Inf} F[g] &= \lim_{\epsilon\to 0} \frac{F[g(\theta,\rho_0)] - F[g(\theta,\rho_0+\epsilon)]}{\epsilon} = -\left.\frac{dF[g(\theta,\rho)]}{d\rho}\right|_{\rho=\rho_0} \\ &= \mathcal{H}_\theta^{-1}(\hat{\theta},\rho_0) \cdot \nabla_\theta \frac{\partial}{\partial\rho} g(\hat{\theta},\rho_0) \quad .\end{aligned} \tag{23}$$

A.2 Navier-Stokes Parametrization

The rectangular cavity characterizing the 2D Navier-Stokes problem formulation is given in Table 2.

The parabolic inflow is defined as

$$\mathbf{u}(x_{\min}, y) = \begin{pmatrix} U_{\max} y \frac{y_{\max}-y}{y_{\max}^2} \\ 0 \end{pmatrix}, \tag{24}$$

where $U_{\max} = 0.3\,\mathrm{m/sec}$. Furthermore, the fluid density ρ was set to 1 and the kinematic viscosity ν to 0.001.

Table 2: Geometric Parameters of the Cavity

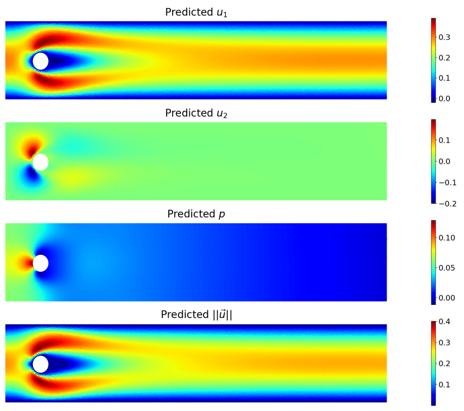
Parameter	Value $[\mathrm{m}]$
$x_{\min}$	0
$x_{\max}$	2.2
$y_{\min}$	0
$y_{\max}$	0.41
Cylinder radius	0.05
Cylinder position (x_c, y_c)	(0.2,0.2)

A.3 Supplemental Figures

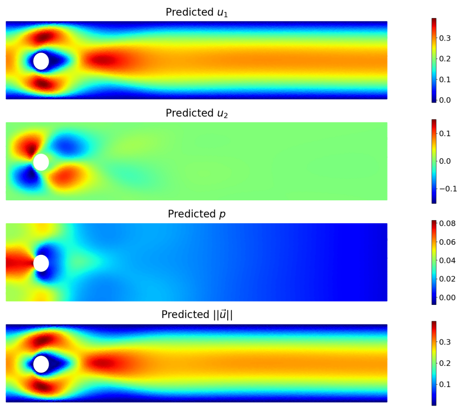
Figure 4 is supplementary to Figure 1 and shows the model predictions additionally for $\|\boldsymbol{u}\|$ and for ϕ_{broken}, ϕ_{bad} as well as target values precomputed with the finite element method (FEM) program FEniCS sourced from [33].

Figure 5 shows the absolute errors between model predictions and the target values depicted in Figure 4d. Note that ϕ_{broken} is evaluated against correct target values.

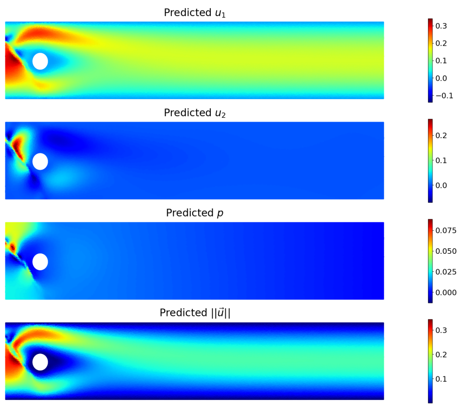
Figure 6 shows the influence heatmap produced by ϕ_{broken} and ϕ_{bad} for the same training point as used in Figure 2.



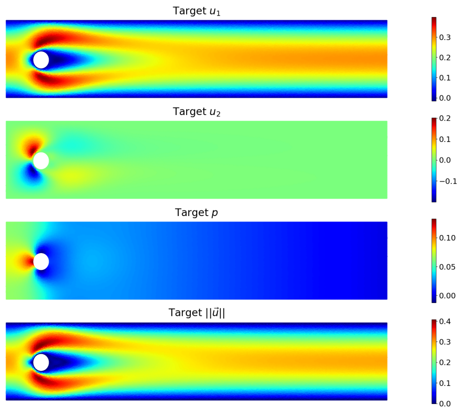
(a) ϕ_{good}



(b) ϕ_{broken}

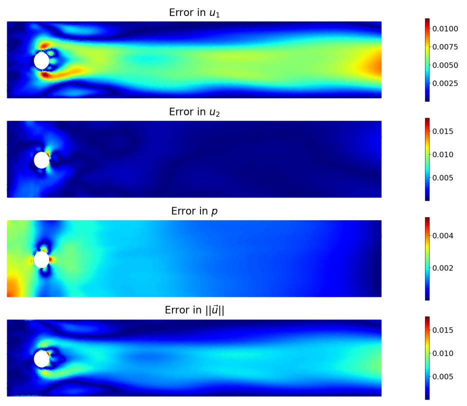


(c) ϕ_{bad}

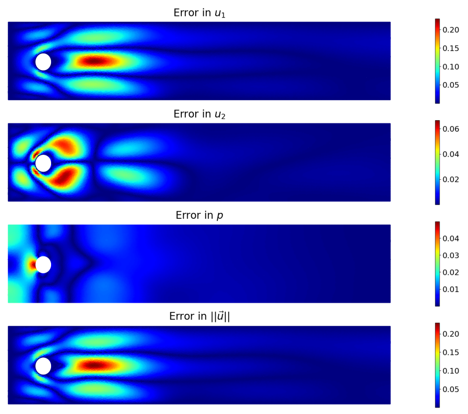


(d) Target values computed with FEniCS.

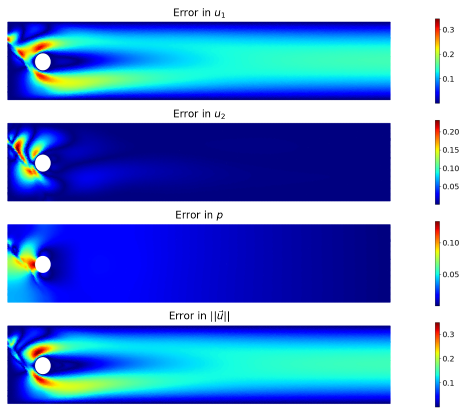
Figure 4: Trained models and target values with respective predictions for u_1, u_2, p, $\|\boldsymbol{u}\|$ (top to bottom).



(a) ϕ_{good}

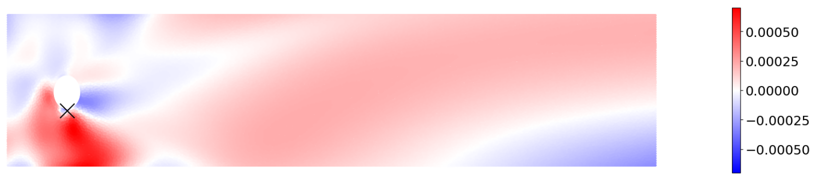


(b) ϕ_{broken}

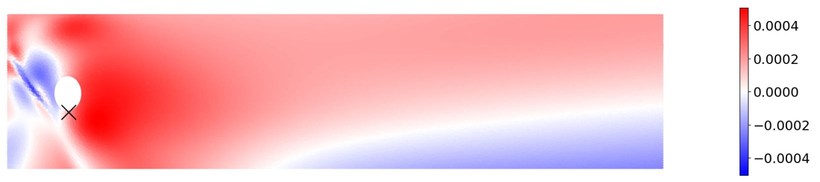


(c) ϕ_{bad}

Figure 5: Absolute errors between respective model predictions for each output dimension w.r.t. precomputed target values.



(a) ϕ_{broken}



(b) ϕ_{bad}

Figure 6: Influence heatmaps on the prediction of u_1 of a single training point close to the cylinder (bottom).