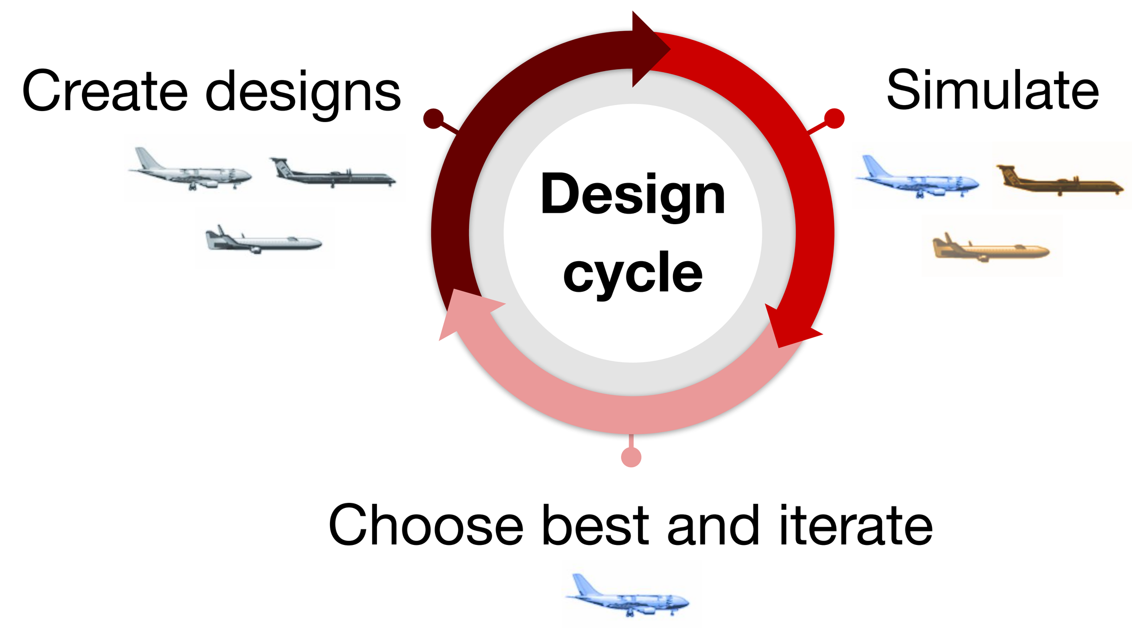
Informing neural networks with simplified physics for better flow prediction

Fedor Sergeev
Supervisors: Prof. Pascal Fua, Dr. Jonathan Donier

1. Introduction

Modern engineering needs methods for quick design evaluation:



While numerical methods are accurate and reliable, they can be prohibitively expensive to use for rapid design iterations (e.g., require hours per simulation). Deep neural networks (DNNs) trained on simulation data are much quicker (minutes per prediction), but may produce nonphysical results and struggle on out-of-distribution samples. Physics-informed Neural Networks (PINNs) potentially combine the advantages of both approaches but are often difficult to tune.

We propose using a hybrid model that combines an unsupervised PINN, trained on a simplified physics system, with a surrogate geometric DNN. This method can potentially be more accurate than a DNN, while being easier to train than a PINN.

Incorporating physics into ML through

Data	Model architecture	Training procedure
Augment data to capture observational biases	Modify the output to satisfy boundary conditions (BCs)	Use PDE residuals as loss terms
e.g., nondimensionalization	e.g., hard Dirichlet constraints	e.g., unsupervised PINN on simplified physics

2. Simplifying physics

Consider continuous incompressible 1-phase fluid at a constant temperature without external forces and heat sources. Its flow is described by the Navier-Stokes equations. They are challenging to solve using PINNs. Therefore, we consider a simplified system (potential flow) by assuming that the fluid motion is irrotational.

Navier-Stokes equations

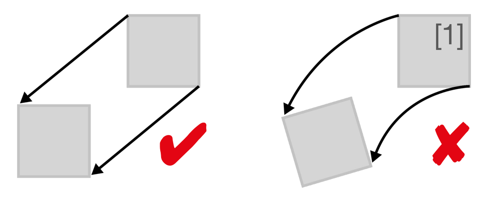
$$\begin{cases} \frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u} - \nu\nabla^2\boldsymbol{u} = -\frac{1}{\rho}\nabla p, \\ \nabla\cdot\boldsymbol{u} = 0, \\ \text{initial condition}, \\ \text{boundary conditions (BCs).} \end{cases}$$

+ irrotationality

$$\boldsymbol{\omega} = \nabla\times\boldsymbol{u} = 0$$

Potential flow

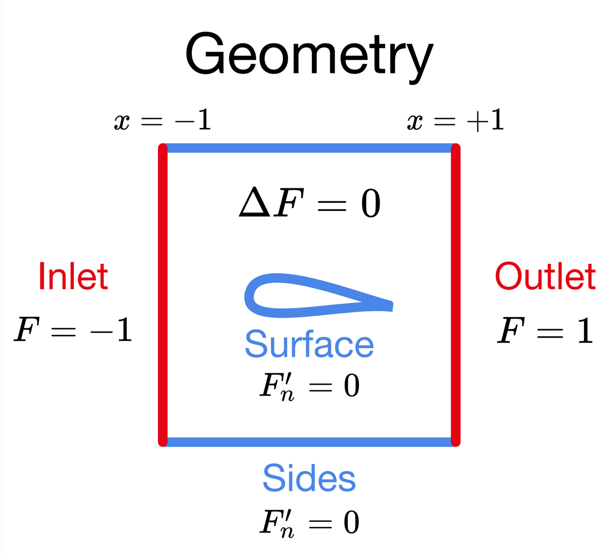
$$\Delta F = 0$$
$$\boldsymbol{u} = \nabla F$$

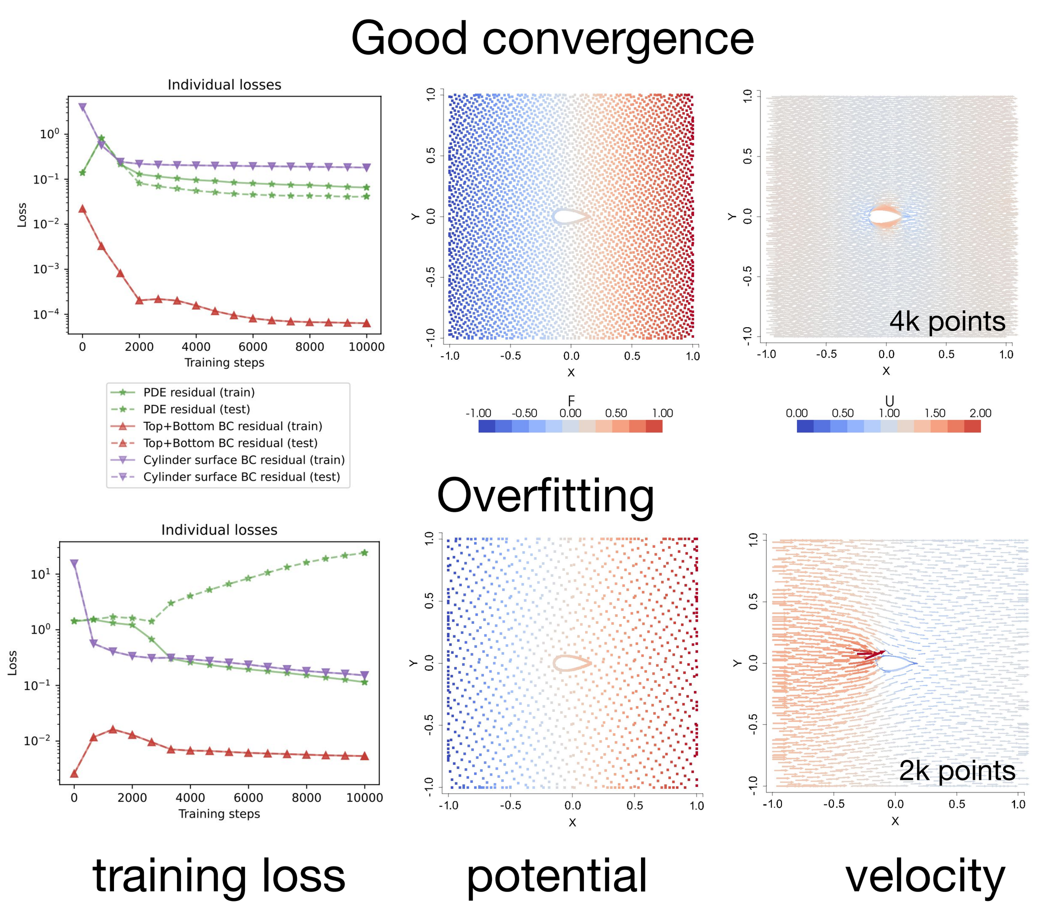


Here $\boldsymbol{u}$ is the velocity vector, p – pressure, ν – viscosity, ρ – density, F – scalar potential.

3. Solving potential flow with PINNs

Fully-connected NN is trained in an unsupervised manner to minimize residuals of the PDE and Neumann BC. Dirichlet BC is imposed with hard constraints. Velocity is computed from potential with autodifferentiation. Note that it is important to sample enough points in the domain to avoid overfitting, as it leads to non-physical predictions.)





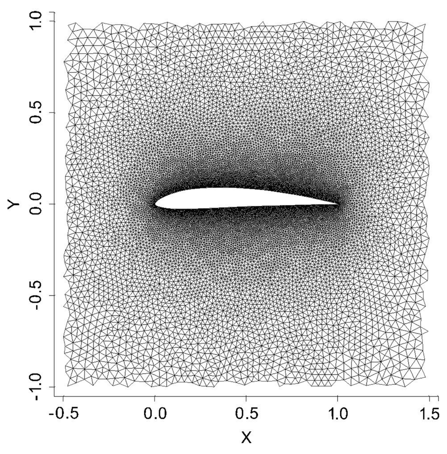
4. Solving viscous flow with PINNs+DNNs

Data

The data set of incompressible fluid flows around 2D NACA airfoils consists of 1096 samples (90% train).

Each sample is generated for a selected airfoil shape and freestream velocity $\boldsymbol{V}$ [2].

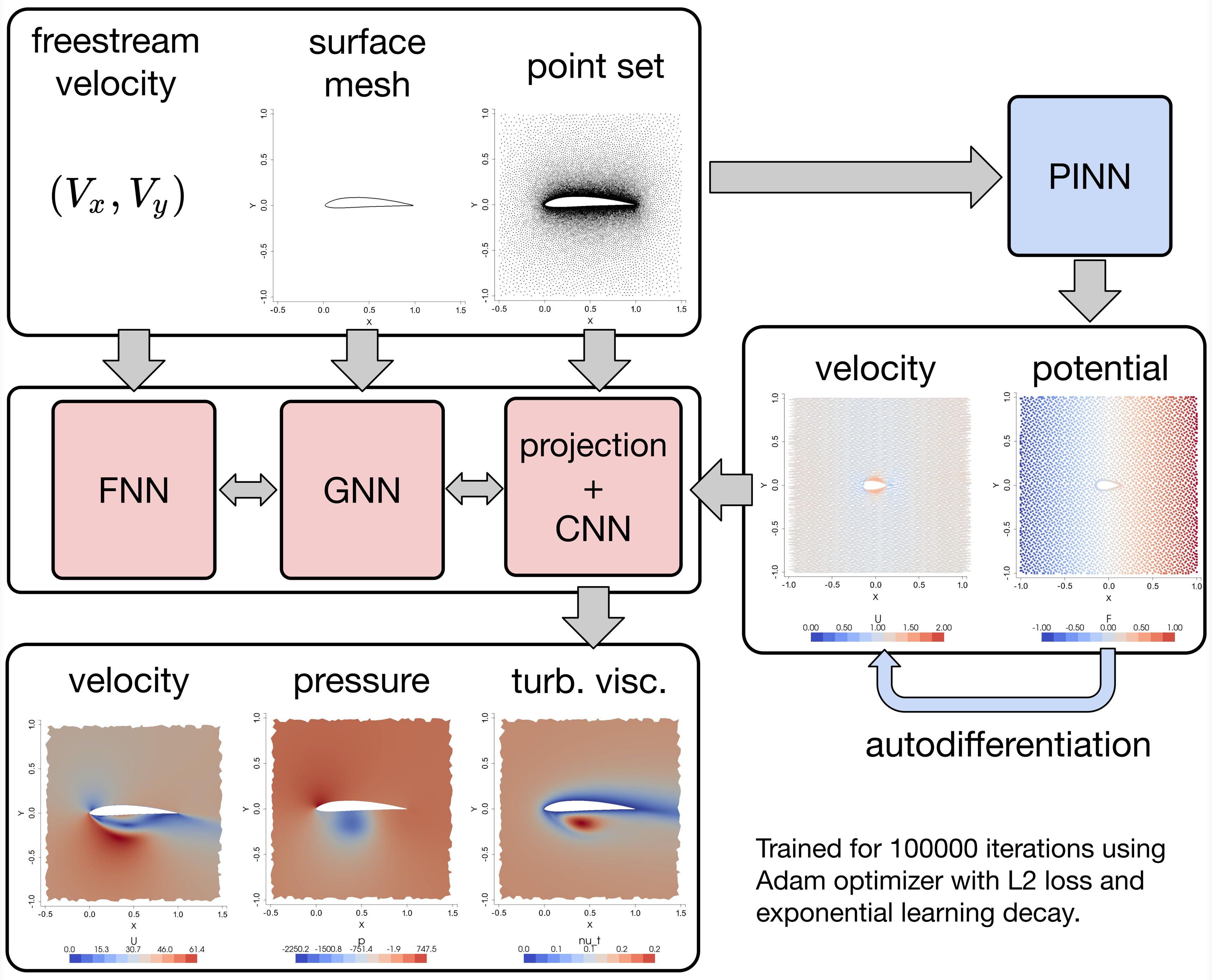
$$\boldsymbol{V} = \begin{pmatrix} V\cos\varphi \\ V\sin\varphi \end{pmatrix}$$
$$V \in [10, 100]$$
$$\varphi \in \left[-\frac{\pi}{8}, \frac{\pi}{8}\right]$$



Task: given freestream velocity and airfoil surface mesh, predict velocity, pressure, and turbulent viscosity on a point set.

Model

We use a geometric DNN proposed in [3] as a surrogate model and a similar PINN to the one used in the previous section.

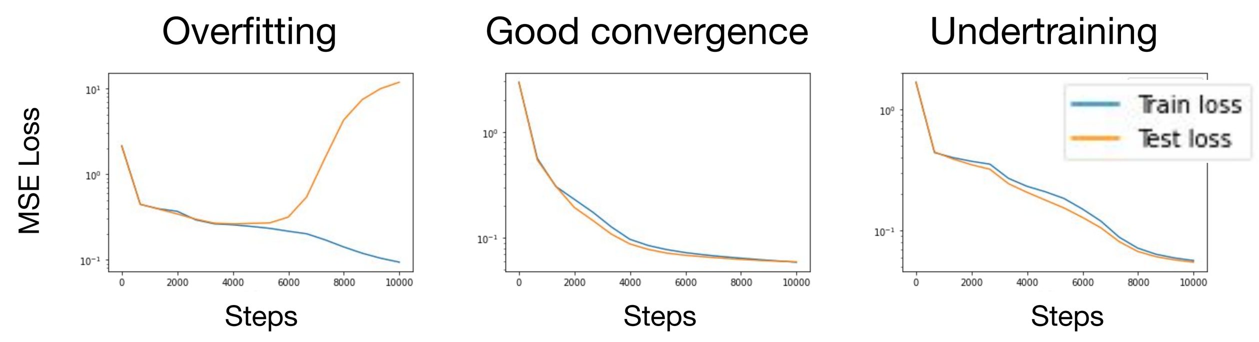


Trained for 100000 iterations using Adam optimizer with L2 loss and exponential learning decay.

Results

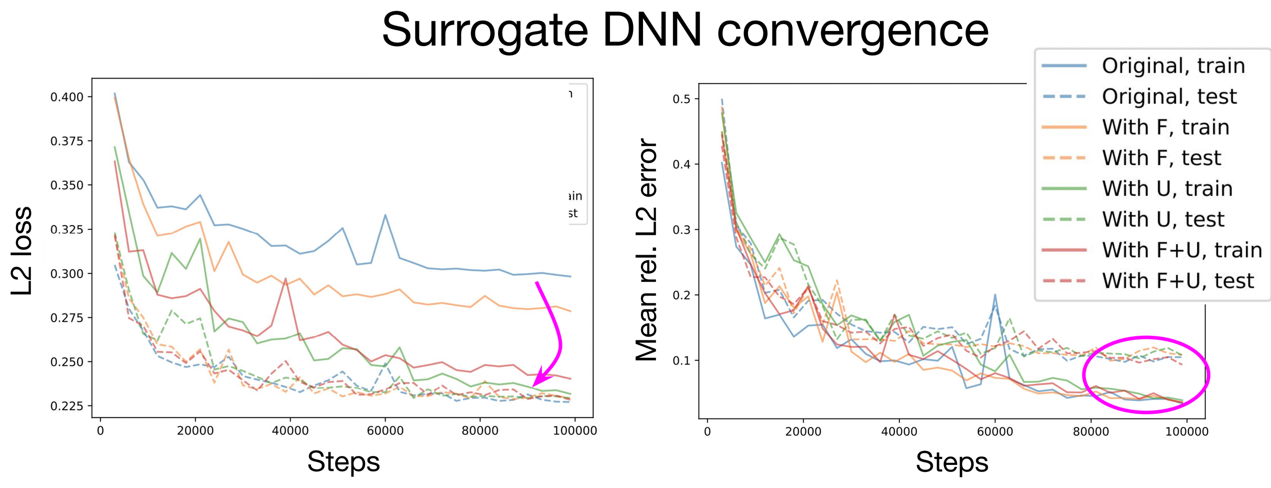
Training a PINN on a simplified physics problem requires around 2 minutes per sample (30h for the whole dataset). During training. we notice that in some cases overfitting and undertraining occur.

PINN convergence examples



The DNN with the addition of the PINN output performs largely the same as the original geometric DNN from [2]. However, our models have significantly smaller distances between train and test curves.

Surrogate DNN convergence



Overall, these results are promising, and the approach can be improved by considering directional potential flow. Future studies can consider the application of our approach for 3D problems as well as transfer learning.

5. References

[1] Modified from Flocess, own work, CC BY 4.0 https://creativecommons.org/licenses/by/4.0, via Wikimedia Commons.
[2] N. Thuerey, K. Weißenow, L. Prantl, and X. Hu, "Deep learning methods for Reynolds-averaged Navier-Stokes simulations of airfoil flows," AIAA Journal, vol. 58, no. 1, pp. 25–36, 2020.
[3] L. Zampieri, "Geometric deep learning for volumetric computational fluid dynamics," EPFL, Politecnico di Milano, Tech. Rep., 2019.