



# NeuroSymbolic Artificial Intelligence at Scale

Marco Fanfani, [marco.fanfani@unifi.it](mailto:marco.fanfani@unifi.it)

<https://www.disit.org/>

Parte: 2.5 (2025-26) Physics-Informed Neural Networks  
Research @ DISIT Lab



# Physics-Informed Neural Networks (PINNs)

Solving Partial Differential Equations through Scientific Machine Learning

Results from Botarelli et al. "Using Physics-Informed neural networks for solving Navier-Stokes equations in fluid dynamic complex scenarios"

# Research @ DISIT Lab

- Research on PINN carried out at the DISIT Lab (Dept. Information Eng.) in collaboration with the T-Group (Dept. Industrial Eng.)
- **Objective:** demonstrating the applicability of PINN on complex fluid-dynamic scenarios
- T. Botarelli, M. Fanfani, P. Nesi, L. Pinelli, "*Using Physics-Informed neural networks for solving Navier-Stokes equations in fluid dynamic complex scenarios,*" Engineering Applications of Artificial Intelligence, Volume 148, 2025, 110347, ISSN 0952-1976, <https://doi.org/10.1016/j.engappai.2025.110347>
- Videos: <https://www.snap4city.org/drupal/node/1010>

Engineering Applications of Artificial Intelligence

The International Journal of Intelligent Real-Time Automation

Engineering Applications of Artificial Intelligence 148 (2025) 110347

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: [www.elsevier.com/locate/engappai](http://www.elsevier.com/locate/engappai)

Using Physics-Informed neural networks for solving Navier-Stokes equations in fluid dynamic complex scenarios

Tommaso Botarelli<sup>a,\*</sup>, Marco Fanfani<sup>a</sup>, Paolo Nesi<sup>a,b</sup>, Lorenzo Pinelli<sup>b</sup>

<sup>a</sup> 2007 Lab, Department of Information Engineering, University of Florence, Via S. Marta 2, 50139, Florence, Italy  
<sup>b</sup> T-Group, Department of Industrial Engineering, University of Florence, Via S. Marta 2, 50139, Florence, Italy

www.elsevier.com/locate/engappai

ARTICLE INFO

**Keywords:**  
 Physics-informed neural networks  
 Deep learning  
 Fluid dynamic  
 Partial differential equations  
 Navier-Stokes

**ABSTRACT**  
 Navier-Stokes equations used to model fluid dynamic processes are fundamental to address several real-world problems related to energy production, aerospace applications, automotive design, industrial process, etc. However, since in most cases they do not admit any analytical solution, numerical simulations are required in industrial contexts to assess fluid dynamic behaviors in specific setups. Computational Fluid Dynamics (CFD) methods, like those using finite volume or element approaches, are exploited to find Navier-Stokes solutions and carry out simulations. However, such methods require expensive hardware resources, relevant computational times, and manual efforts for the definition of dense meshes on which equations are evaluated iteratively for each time step of the simulation. Physics-Informed Neural Networks (PINNs), which are deep neural networks whose physical laws are directly embedded into the training process, offer a promising approach for solving Navier-Stokes equations, thus alleviating hardware and time requirements. PINNs bypass some CFD limitations by using neural networks to produce solutions based on governing equations, thus reducing the need for large datasets, dense meshing, and iterative estimation over time. This paper evaluates the application of PINNs in near real-world scenarios, while considering various geometries. The study focuses on the achieved accuracy, by comparing PINN estimates with CFD solutions obtained via OpenFOAM, and the required training times; this includes evaluating different neural network architectures, activation functions, and numbers of sampling points. Additionally, several training strategies such as fine-tuning, multi-resolution learning, and parameterized training are proposed to enhance efficiency and obtain speed up. Results demonstrate that PINNs can achieve comparable accuracy to CFD methods (with a velocity magnitude mean absolute error inferior to 10<sup>-3</sup>) and significantly reduce computational costs. Our findings demonstrated that with appropriate training techniques PINNs can be effectively used in industrial applications requiring rapid and accurate fluid dynamic simulations, thus paving the way for their broader adoption in practical engineering problems.

1. Introduction

Modelling Fluid Dynamic (FD) problems is fundamental in several disciplines like astrophysics, chemistry, environmental sciences, civil, biomedical, and mechanical engineering. The goal is to simulate complex processes to make predictions on specific scenarios, while avoiding the necessity of conducting expensive real-world experiments in the design phase. FD problems are modeled by the Navier-Stokes (N-S) equations, a system of partial differential equations (PDEs) describing the momentum and energy balances, together with the conservation of mass. The application of N-S equation to engineering studies can range from external aerodynamic problems (aircraft fuselage and wing, racing cars, wind turbines, etc.) to internal flow problems (aircraft engines, piping, injectors, anode/cathode curing, etc.), as well as fluid mixture investigations (steam condensing) and reacting flows (combustion systems). Except for some simplified specific problems, the general form of N-S equations does not admit known analytical solutions and may be solved by means of numerical approaches. Computational Fluid Dynamics (CFD) methods based on finite elements/finite difference (Zienkiewicz et al., 2005; Stabile and Iaschetta, 2021) have been widely used for solving N-S equations to estimate flow velocity and pressure fields given specific boundary and initial Conditions (IC and IC). However, such approaches have strong limitations, since they require considerable time for both their instantiation and execution (Pinelli

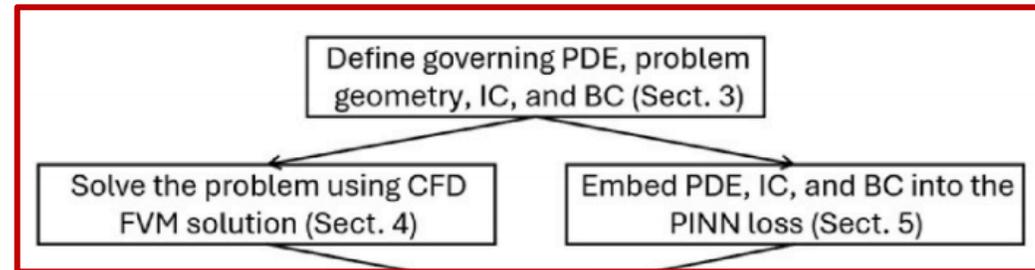
\* Corresponding author.  
 E-mail address: [pinelli@unifi.it](mailto:pinelli@unifi.it) (P. Nesi).

<https://doi.org/10.1016/j.engappai.2025.110347>  
 Received 11 August 2024; Received in revised form 18 January 2025; Accepted 15 February 2025  
 Available online 7 March 2025  
 0952-1976/© 2025 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

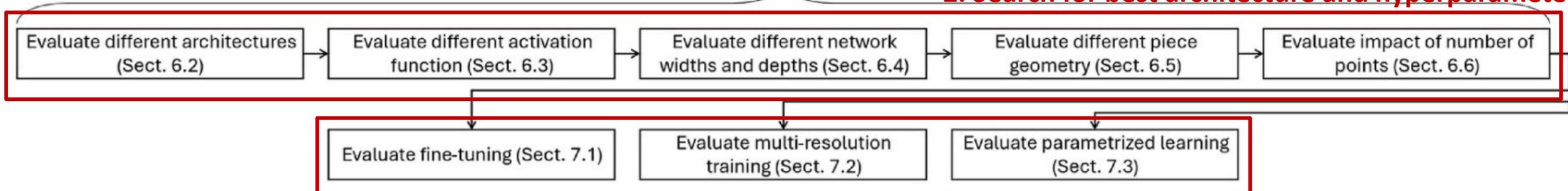
# Experiments @ DISIT Lab

- Research project structure

## 1. Introduction and problem definition



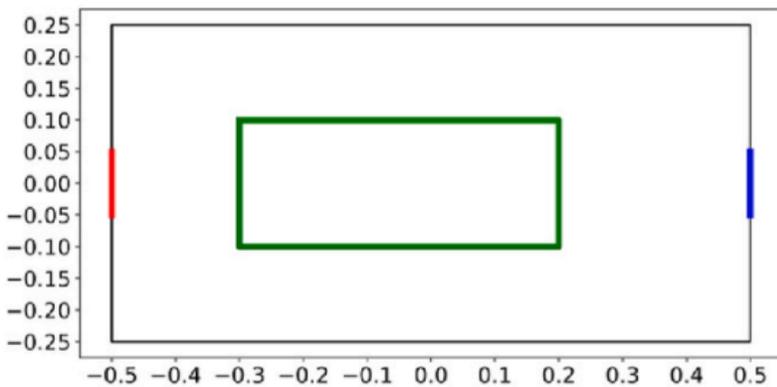
## 2. Search for best architecture and hyperparameters



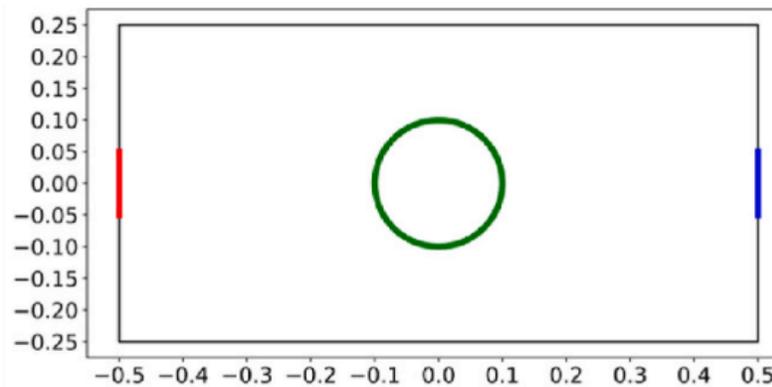
## 3. Advanced training techniques

# Experiments @ DISIT Lab

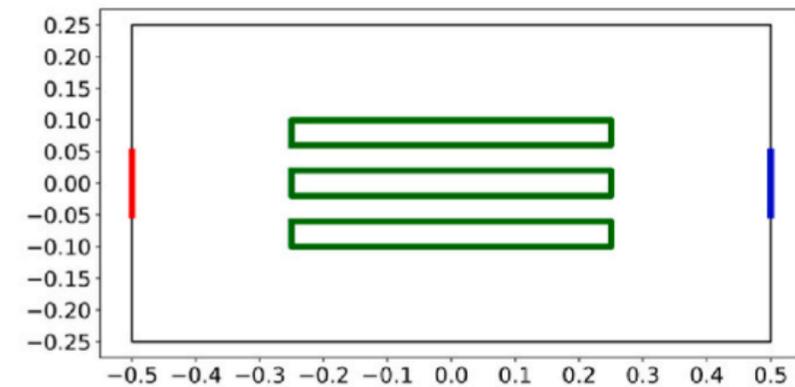
- Case studies
  - Solving **Navier-Stokes** estimating  $u_x$ ,  $u_y$ , and  $p$  in a forward problem
  - Considering **three different domain** setup
  - Exploiting **ReLoBraLo** and **multiple time windows**



(a)



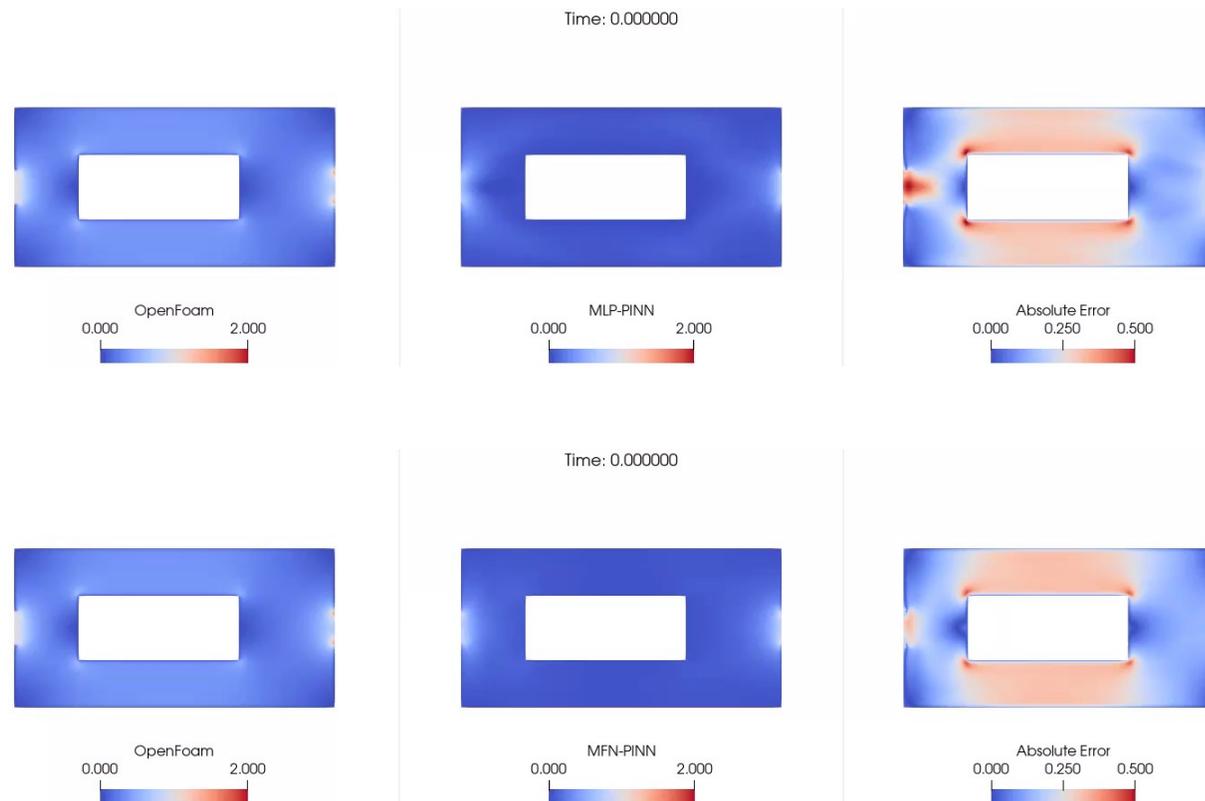
(b)



(c)

# Network Architecture

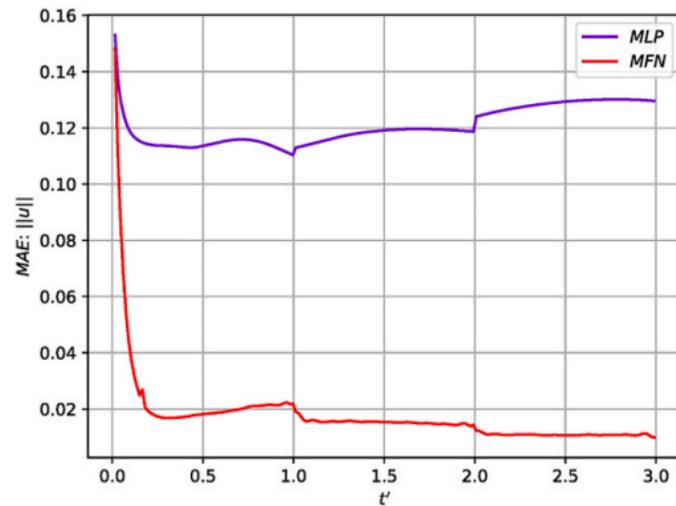
- Multi Layer Perception vs Modified Fourier Network



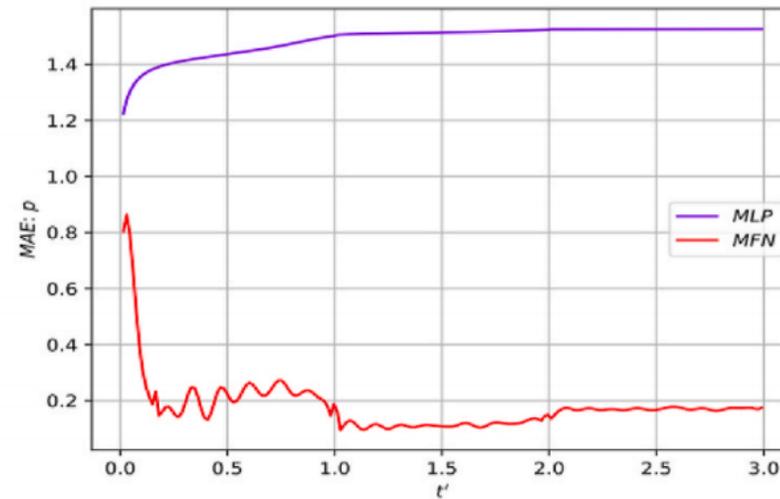
- 6 layers with 512 neurons
- Tanh activation

# Network Architecture

- Multi Layer Perception vs Modified Fourier Network



(a)



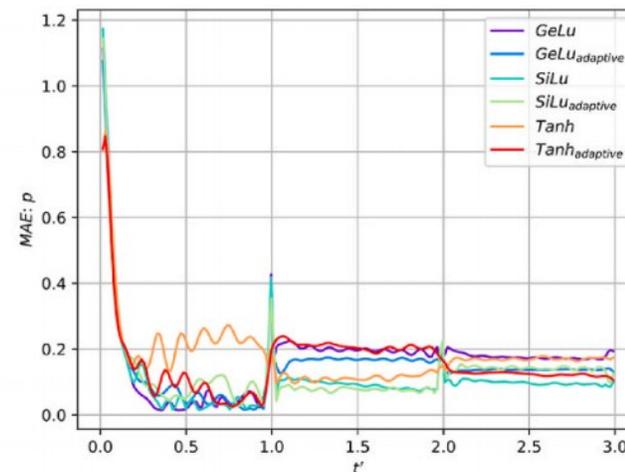
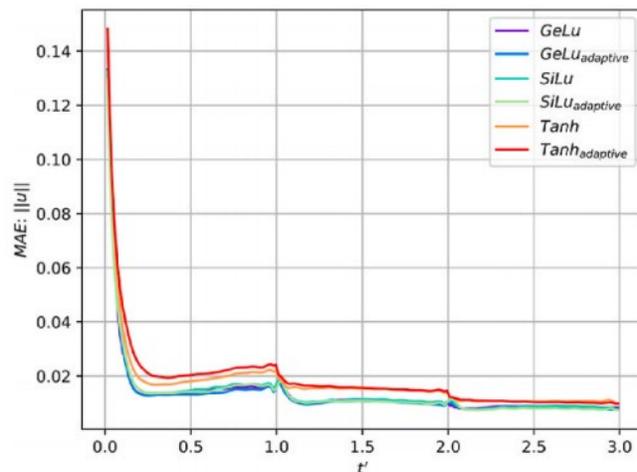
(b)

Architecture	$MAE_{avg}$				$MAE_{st}$				Training Times (hh:mm)
	$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	
MLP-PINN	0.121	0.111	0.053	1.489	0.130	0.119	0.054	1.526	03:32
MFN-PINN	0.017	0.017	0.009	0.179	0.011	0.010	0.005	0.171	04:55

# Network Architecture

- Activation Functions

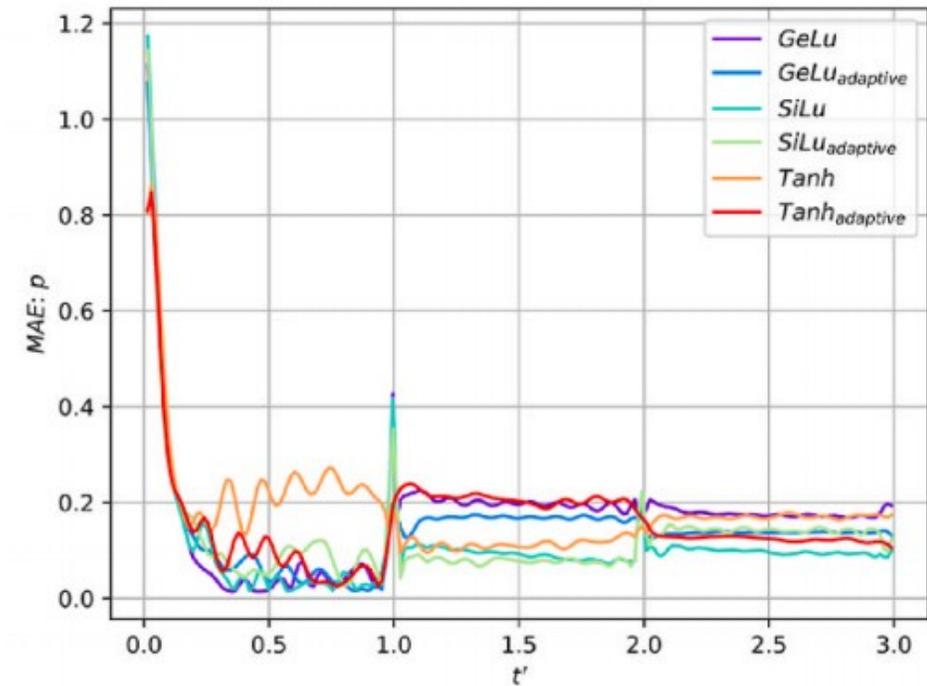
Activations	$MAE_{avg}$				$MAE_{st}$				Training Times (hh:mm)
	$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	
Tanh	0.017	0.017	0.009	0.179	0.011	0.010	0.005	0.171	04:55
Tanh Adaptive	0.018	0.017	0.011	0.158	0.010	0.010	0.005	0.119	05:32
SILU	0.014	0.013	<b>0.007</b>	<b>0.107</b>	0.009	0.008	0.005	<b>0.093</b>	05:47
SILU Adaptive	<b>0.013</b>	0.013	<b>0.007</b>	0.126	<b>0.008</b>	<b>0.007</b>	<b>0.004</b>	0.138	06:10
GELU	<b>0.013</b>	0.013	<b>0.007</b>	0.166	0.009	0.008	0.005	0.174	05:26
GELU Adaptive	<b>0.013</b>	<b>0.012</b>	<b>0.007</b>	0.145	<b>0.008</b>	0.008	<b>0.004</b>	0.138	05:49



- MFN
- 6 layers with 512 neurons
- Varying activations

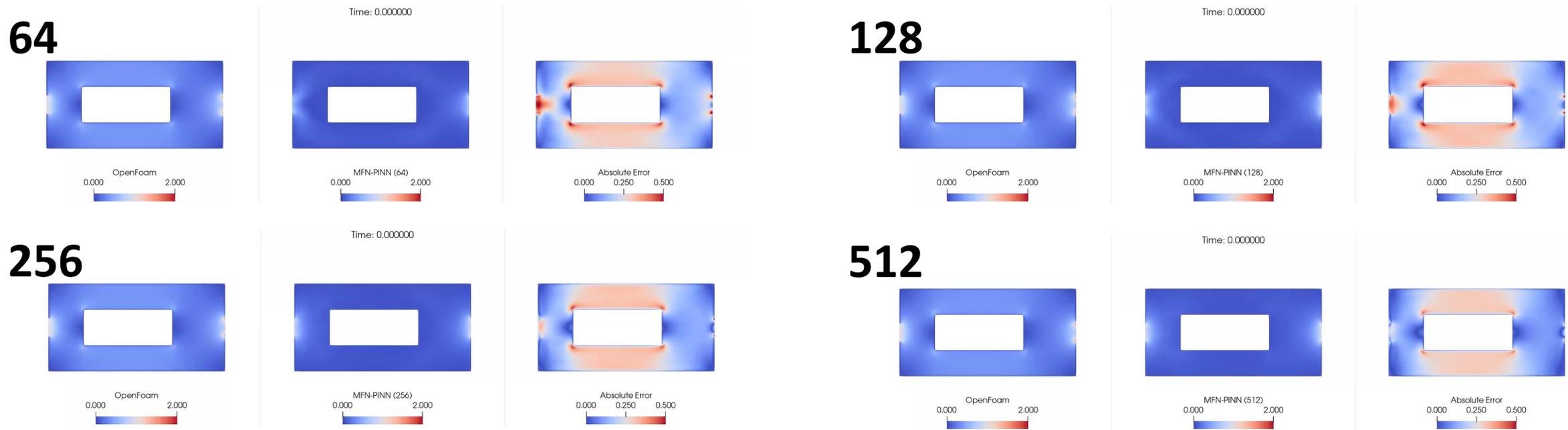
# Observation on Pressure

- Pressure estimate always show **more erratic results**
- Estimating pressure is more challenging for PINNs primarily because **no specific pressure constraints are typically defined** in the Initial Conditions (IC) or Boundary Conditions (BC)
- The system is left to manifest pressure conditions solely as a **byproduct of the fluid flow solution**
- Additionally, the **dynamic range of pressure values in these scenarios is significantly wider** (e.g., between -8.816 and 1.390) compared to velocity magnitudes (0.000 to 1.996), which leads to less regular trends and higher relative errors during training



# Network Architecture

- Network layer width

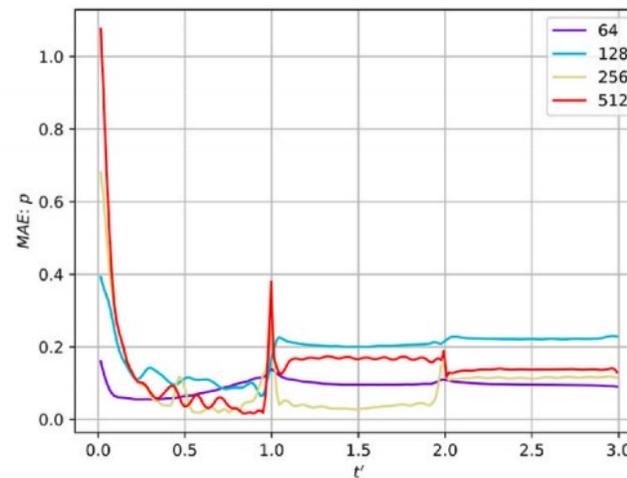
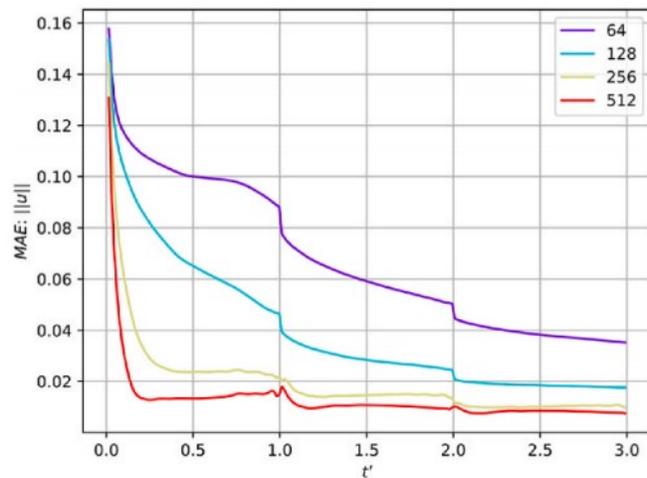


- MFN
- 6 layers with different width
- Adaptive GELU

# Network Architecture

- Network layer width

Layer width	$MAE_{avg}$				$MAE_{st}$				Training Times (hh:mm)
	$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	
64	0.068	0.064	0.032	0.092	0.037	0.034	0.018	0.094	03:24
128	0.040	0.038	0.020	0.187	0.018	0.017	0.009	0.223	03:38
256	0.019	0.019	0.010	<b>0.091</b>	0.010	0.010	0.005	0.116	04:25
512	<b>0.013</b>	<b>0.012</b>	<b>0.007</b>	0.145	<b>0.008</b>	<b>0.008</b>	<b>0.004</b>	0.138	05:49

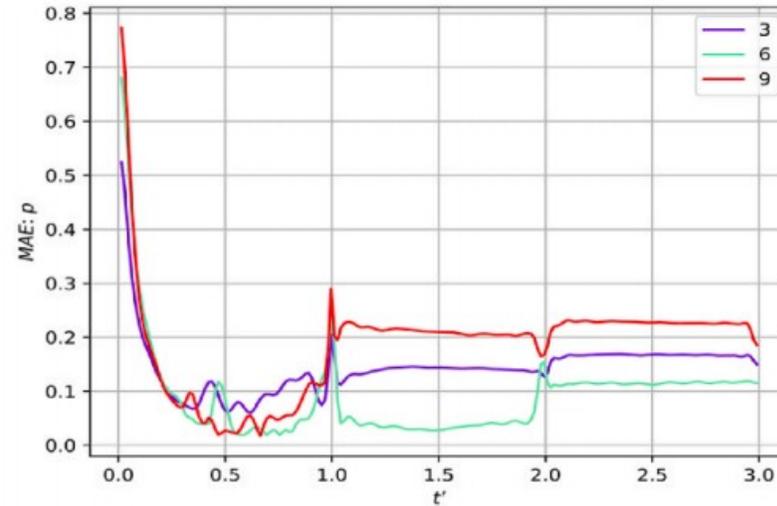
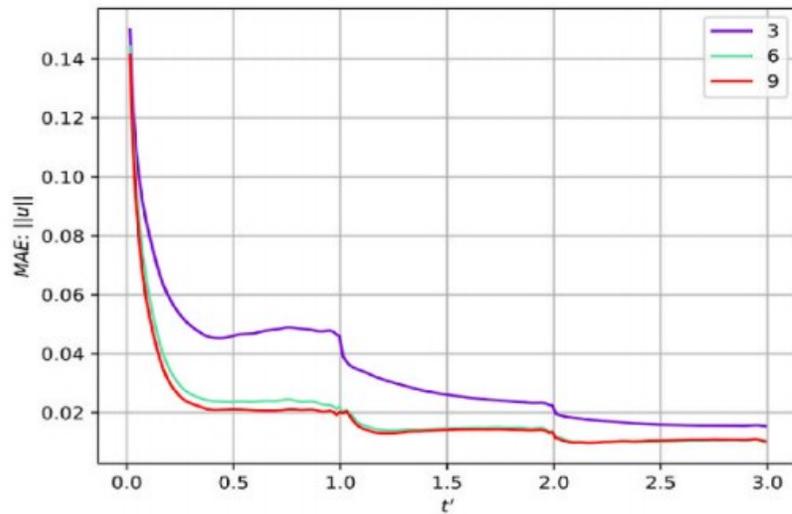


- MFN
- 6 layers with different width
- Adaptive GELU

# Network Architecture

- Network depth (layer number)

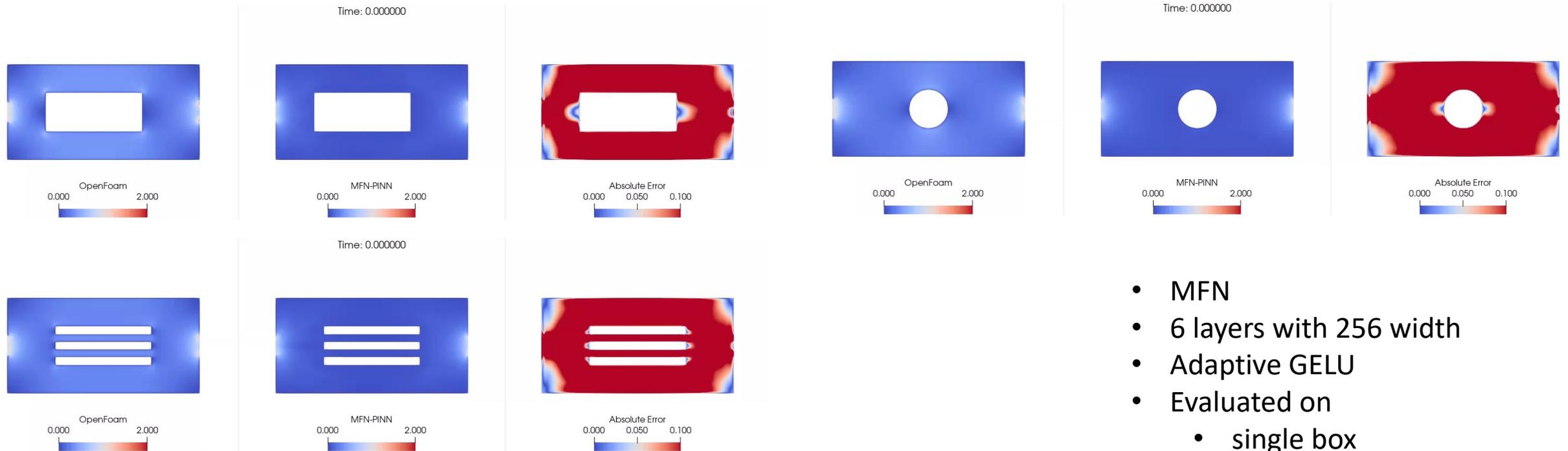
Hidden layer number	$MAE_{avg}$				$MAE_{st}$				Training Times (hh:mm)
	$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	
3	0.033	0.031	0.017	0.144	0.016	0.015	0.007	0.166	<b>03:22</b>
6	0.019	0.019	0.010	0.091	0.010	0.010	0.005	0.116	04:25
9	0.018	0.017	0.010	0.185	0.010	0.010	0.005	0.234	05:18



- MFN
- Different number of layers with 256 width
- Adaptive GELU

# Domains

- Evaluation on different domains

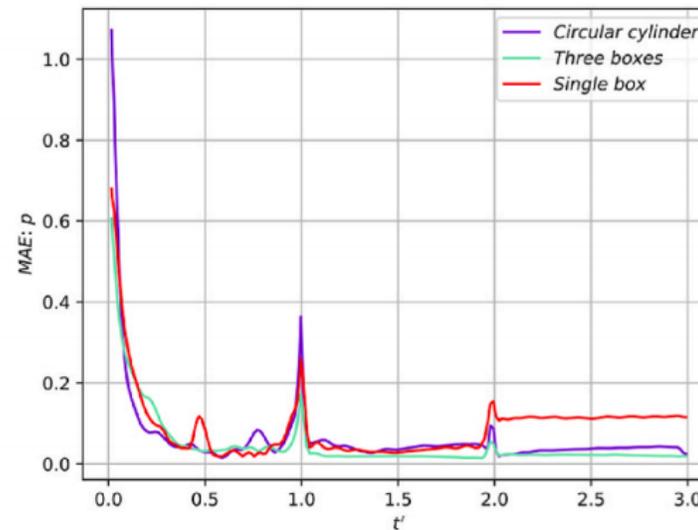
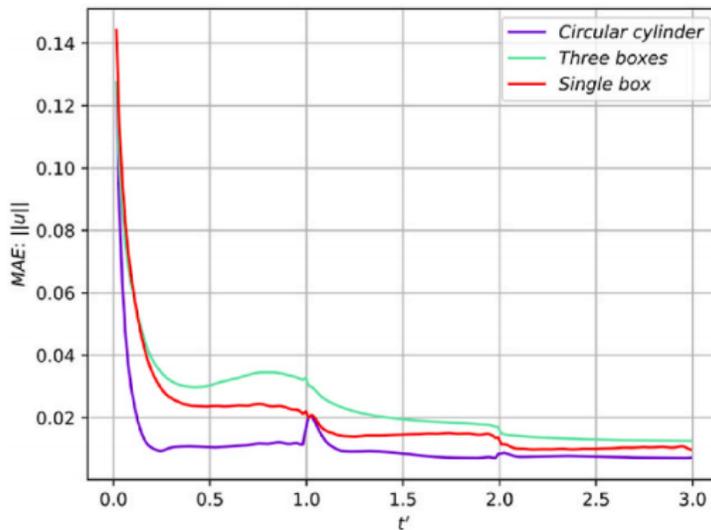


- MFN
- 6 layers with 256 width
- Adaptive GELU
- Evaluated on
  - single box
  - cylinder
  - three-boxes

# Domains

- Evaluation on different domains

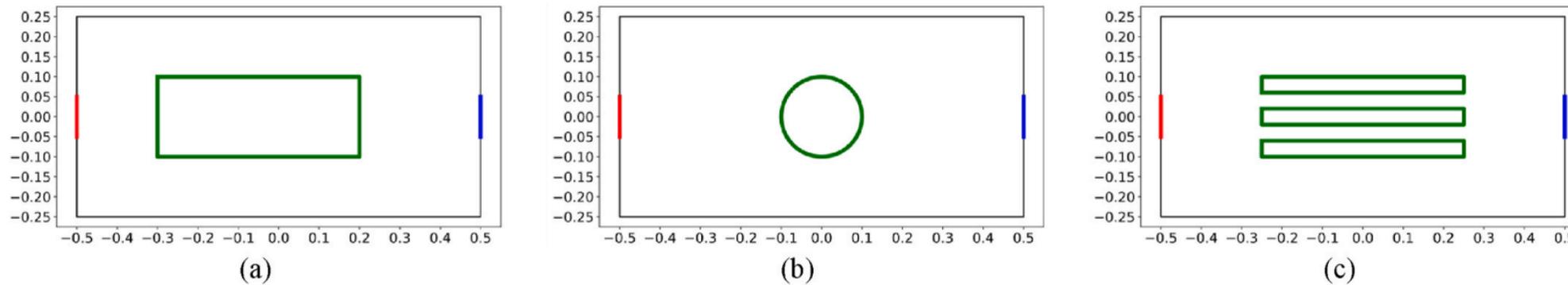
Piece	$MAE_{avg}$				$MAE_{st}$			
	$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$
Single box	0.019	0.019	0.010	0.091	0.010	0.010	0.005	0.116
Circular cylinder	0.011	0.010	0.006	0.065	0.007	0.006	0.004	0.040
Three boxes	0.024	0.023	0.014	0.048	0.013	0.012	0.006	0.021



- MFN
- 6 layers with 256 width
- Adaptive GELU
- Evaluated on
  - single box
  - cylinder
  - three-boxes

# Collocation Points

- Impact of different number of collocation points



Area	High	Mid	Low
Inlet	500	250	100
Outlet	500	250	100
Channel walls	2500	1250	500
Piece walls	1250	600	250
Channel interior	5000	2500	1000
<b>Total</b>	<b>9750</b>	<b>4850</b>	<b>1950</b>

- MFN
- 6 layers with 256 width
- Adaptive GELU
- Evaluated on
  - single box
  - cylinder
  - three-boxes
- In previous tests the Mid set was used

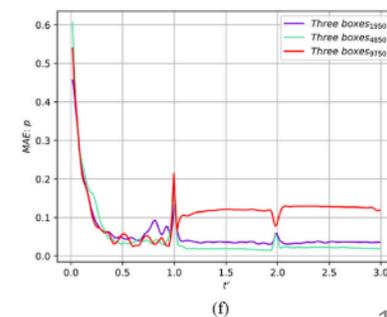
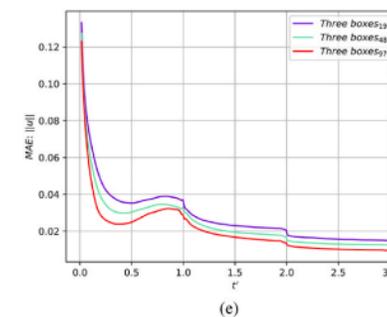
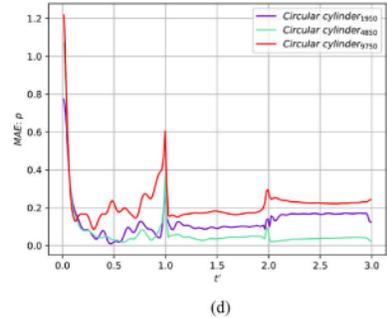
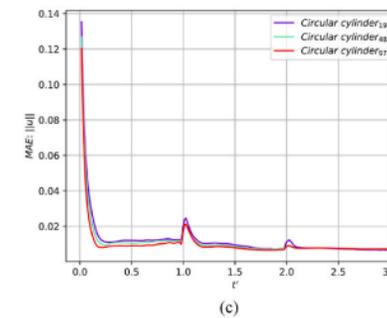
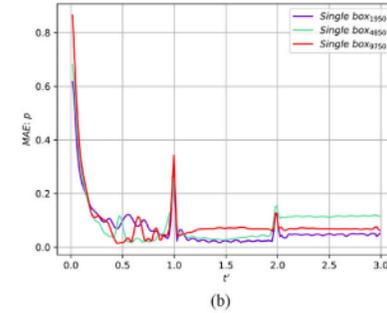
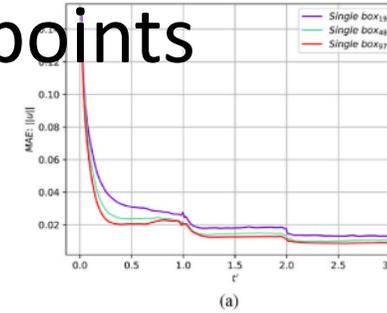
# Collocation Points

- Impact of different number of collocation points

Geometry	Collocation points	$MAE_{avg}$				Training times (hh:mm)
		$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	
Single box	Low	0.024	0.023	0.013	0.071	
	Mid	0.019	0.019	0.010	0.091	
	High	0.017	0.016	0.009	0.089	
Circular cylinder	Low	0.012	0.012	0.007	0.128	
	Mid	0.011	0.010	0.006	0.065	
	High	0.010	0.010	0.006	0.219	
Three boxes	Low	0.028	0.026	0.016	0.059	
	Mid	0.024	0.023	0.014	0.048	
	High	0.021	0.019	0.012	0.112	

Geometry	Collocation points	$MAE_{st}$				Training times (hh:mm)
		$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	
Single box	Low	0.013	0.013	0.006	0.048	02:47
	Mid	0.010	0.010	0.005	0.116	04:25
	High	0.009	0.008	0.005	0.068	07:14
Circular cylinder	Low	0.007	0.006	0.004	0.166	02:47
	Mid	0.007	0.006	0.004	0.040	04:24
	High	0.007	0.006	0.004	0.226	07:11
Three boxes	Low	0.015	0.014	0.008	0.036	02:48
	Mid	0.013	0.012	0.006	0.021	04:24
	High	0.010	0.009	0.005	0.126	07:47

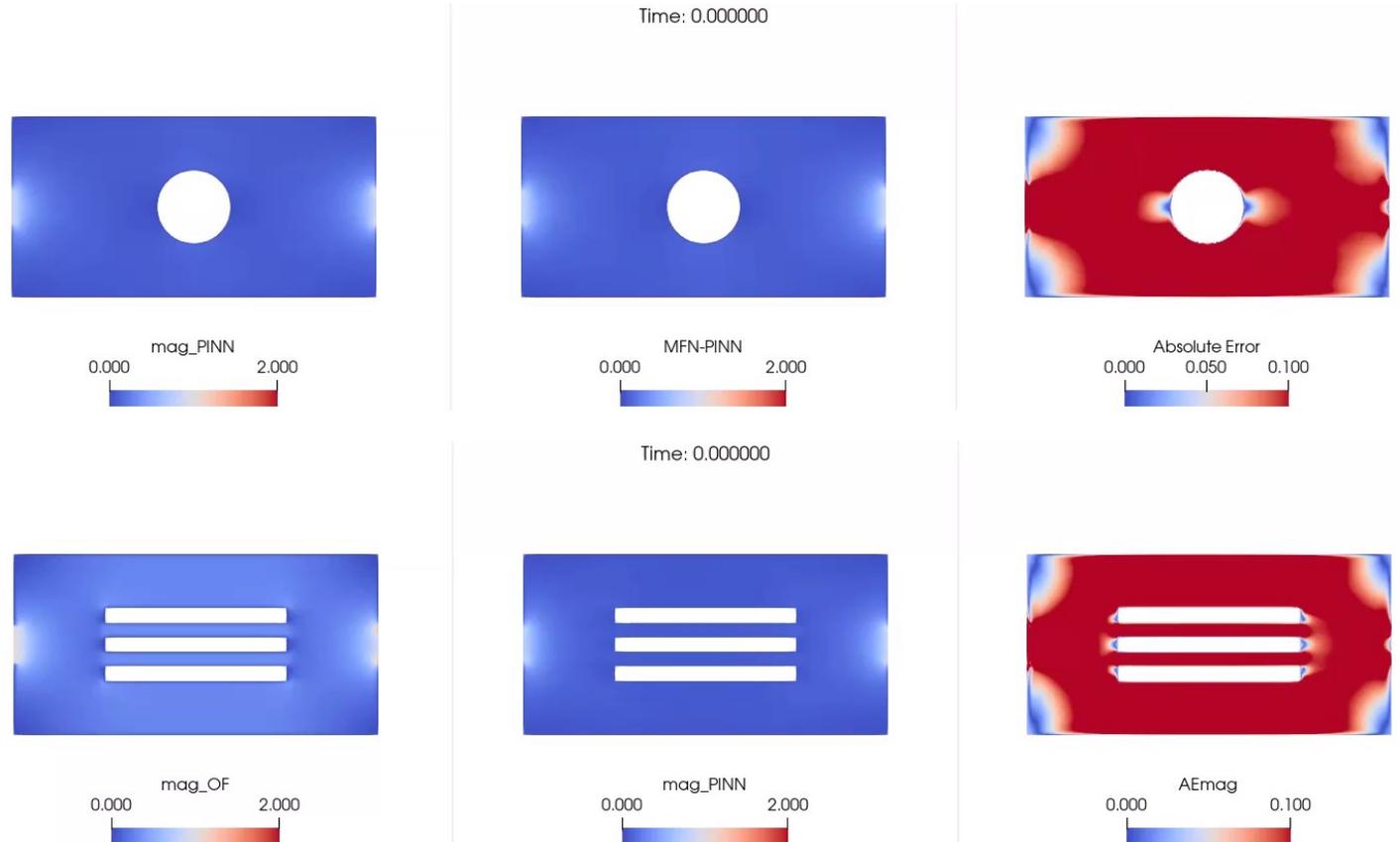


# Advanced Training Strategies

- **Fine-tuning**
  - fine-tune the net trained on the single-box geometry to apply it to the other geometries
  - apply fine-tuning between successive time windows
- **Multi-resolution**
  - number of collocation points changes in different time windows (focus on steady-state solution)
- **Parametrized training**
  - parametrize the piece geometry to learn to predict different (similar) domain without retraining

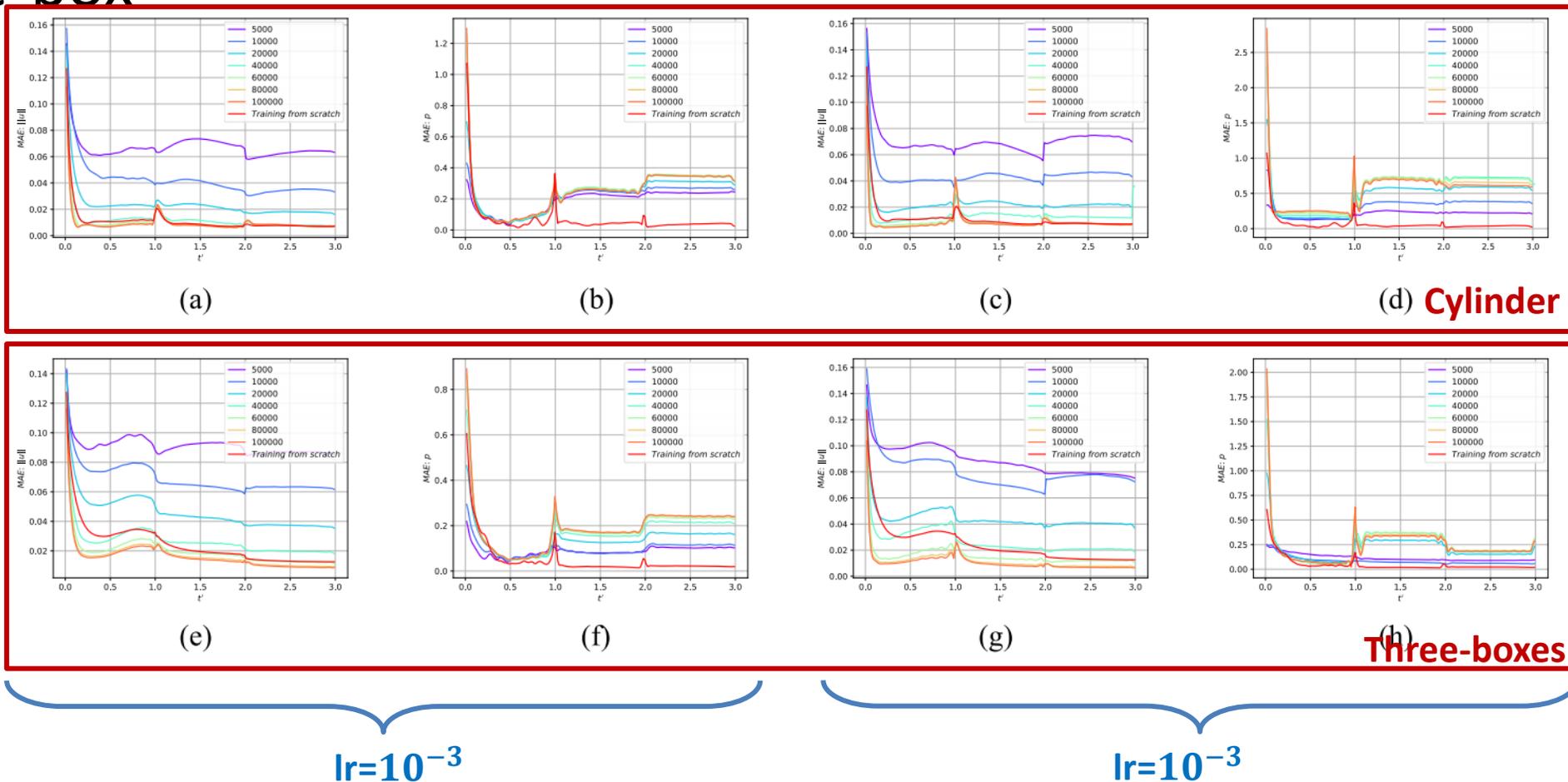
# Fine-tuning

- Train on single-box
- Fine-tune on
  - cylinder
  - three-boxes



# Fine-tuning

- Train on single-box
- Fine-tune on
  - cylinder
  - three-boxes



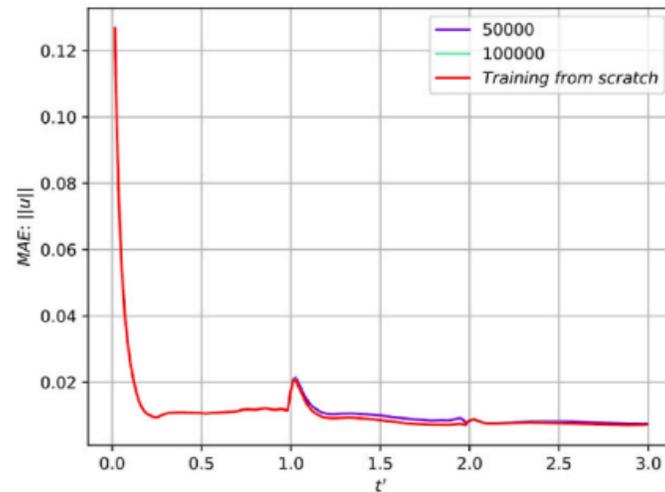
# Fine-tuning

Geometry	Training type	Training epochs	$MAE_{avg}$				$MAE_{st}$				Training Times (hh:mm)
			$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	
Single Box to Circular cylinder	Fine-tuning ( $lr = 10^{-3}$ )	5000	0.067	0.065	0.041	0.188	0.064	0.062	0.039	0.240	00:13
		10000	0.042	0.040	0.025	0.209	0.035	0.033	0.022	0.270	00:29
		20000	0.023	0.022	0.014	0.229	0.018	0.017	0.011	0.310	00:59
		40000	0.012	0.012	0.007	0.248	0.008	0.007	0.005	0.342	02:01
		60000	0.010	0.010	0.006	0.252	<b>0.007</b>	<b>0.006</b>	<b>0.004</b>	0.350	03:00
		80000	0.010	0.009	0.005	0.248	<b>0.007</b>	<b>0.006</b>	<b>0.004</b>	0.346	04:01
		100000	0.009	0.009	0.005	0.247	<b>0.007</b>	<b>0.006</b>	<b>0.004</b>	0.343	05:01
	Fine-tuning ( $lr = 10^{-2}$ )	5000	0.070	0.067	0.037	0.205	0.074	0.071	0.039	0.222	00:13
		10000	0.045	0.042	0.025	0.313	0.046	0.044	0.027	0.380	00:29
		20000	0.023	0.021	0.014	0.457	0.022	0.021	0.013	0.585	00:59
		40000	0.014	0.013	0.008	0.561	0.012	0.011	0.007	0.722	02:01
		60000	0.009	0.009	0.005	0.568	<b>0.007</b>	<b>0.006</b>	<b>0.004</b>	0.705	03:01
		80000	<b>0.008</b>	<b>0.008</b>	<b>0.004</b>	0.548	<b>0.007</b>	<b>0.006</b>	<b>0.004</b>	0.651	04:01
		100000	<b>0.008</b>	<b>0.008</b>	<b>0.004</b>	0.531	<b>0.007</b>	<b>0.006</b>	<b>0.004</b>	0.609	05:02
Training from scratch	100000	0.011	0.010	0.006	<b>0.065</b>	<b>0.007</b>	<b>0.006</b>	<b>0.004</b>	<b>0.040</b>	04:24	
Single Box to Three boxes	Fine-tuning ( $lr = 10^{-3}$ )	5000	0.091	0.083	0.042	0.088	0.087	0.078	0.04	0.103	00:13
		10000	0.069	0.064	0.033	0.096	0.063	0.058	0.026	0.116	00:29
		20000	0.046	0.043	0.023	0.135	0.037	0.034	0.017	0.165	00:59
		40000	0.027	0.025	0.015	0.168	0.019	0.018	0.009	0.214	02:01
		60000	0.020	0.018	0.011	0.181	0.012	0.012	0.006	0.233	03:01
		80000	0.017	0.016	0.010	0.186	0.010	0.009	0.005	0.24	04:01
		100000	0.016	0.015	0.010	0.187	0.009	0.008	0.005	0.242	05:02
	Fine-tuning ( $lr = 10^{-2}$ )	5000	0.089	0.083	0.038	0.117	0.078	0.072	0.033	0.091	00:13
		10000	0.080	0.074	0.036	0.084	0.076	0.069	0.034	0.059	00:29
		20000	0.044	0.041	0.022	0.202	0.040	0.038	0.017	0.157	00:59
		40000	0.027	0.025	0.014	0.235	0.021	0.02	0.009	0.182	02:01
		60000	0.016	0.015	0.009	0.244	0.012	0.011	0.005	0.197	03:01
		80000	0.012	<b>0.011</b>	0.007	0.236	0.008	0.007	<b>0.004</b>	0.192	04:01
		100000	<b>0.011</b>	<b>0.011</b>	<b>0.006</b>	0.231	<b>0.007</b>	<b>0.006</b>	<b>0.004</b>	0.190	05:02
Training from scratch	100000	0.024	0.023	0.014	<b>0.048</b>	0.013	0.012	0.006	<b>0.021</b>	04:24	

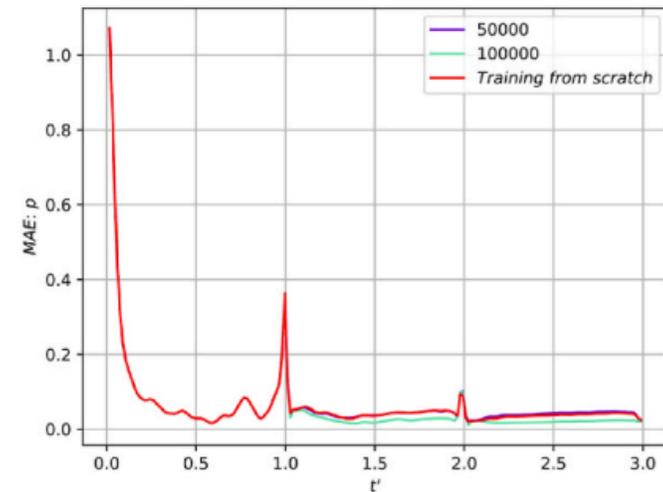
# Fine-tuning

- Fine-tuning between time-windows (single-box)

Training epochs	$MAE_{avg}$				$MAE_{st}$				Training Times for 2nd and 3rd windows (hh:mm)
	$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	
50,000	0.012	0.011	0.007	0.066	0.008	0.007	0.005	0.044	<b>01:38</b>
100,000	<b>0.011</b>	0.011	<b>0.006</b>	<b>0.055</b>	<b>0.007</b>	<b>0.006</b>	<b>0.004</b>	0.022	03:17
Training from scratch	<b>0.011</b>	<b>0.010</b>	<b>0.006</b>	0.065	<b>0.007</b>	<b>0.006</b>	<b>0.004</b>	<b>0.040</b>	03:17



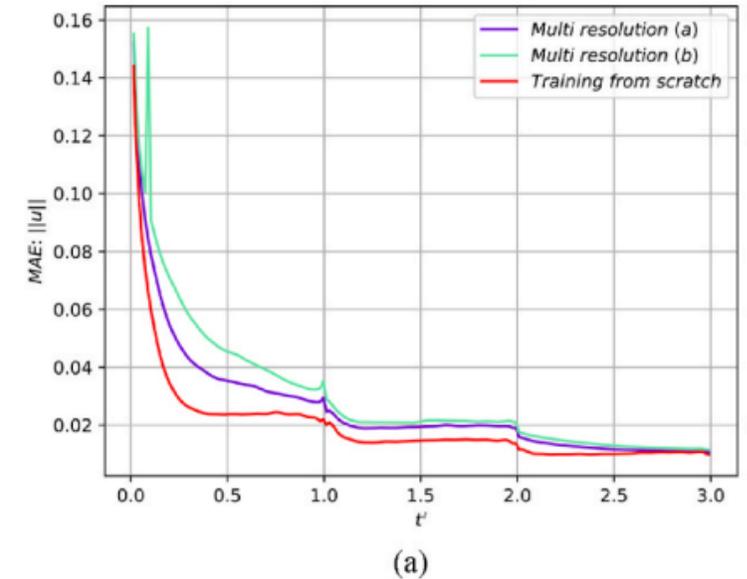
(a)



(b)

# Multi-resolution

- Using a reduced number of collocation points in the first two time windows
  - case (a): first and second windows use the Low point configuration (1950)
  - case (b): first and second windows use only 1000 points (Inlet:75, Outlet: 75, Channel Walls: 250, Piece Walls: 100, Channel Interior: 500)
  - in both cases in the last window the High setup is used



Experiment	$MAE_{avg}$				$MAE_{st}$				Training Times (hh:mm)
	$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	
Multi resolution (a)	0.026	0.024	0.014	0.057	0.011	0.011	0.006	0.026	01:57 + 02:14
Multi resolution (b)	0.030	0.029	0.015	0.110	0.012	0.012	0.006	<b>0.101</b>	<b>01:36 + 02:14</b>
Training from scratch	<b>0.019</b>	<b>0.019</b>	<b>0.010</b>	<b>0.091</b>	<b>0.010</b>	<b>0.010</b>	<b>0.005</b>	0.116	04:25

Last row uses the Mid setup

Time pass from **04h:25m** to **03h:50m** in case (b)

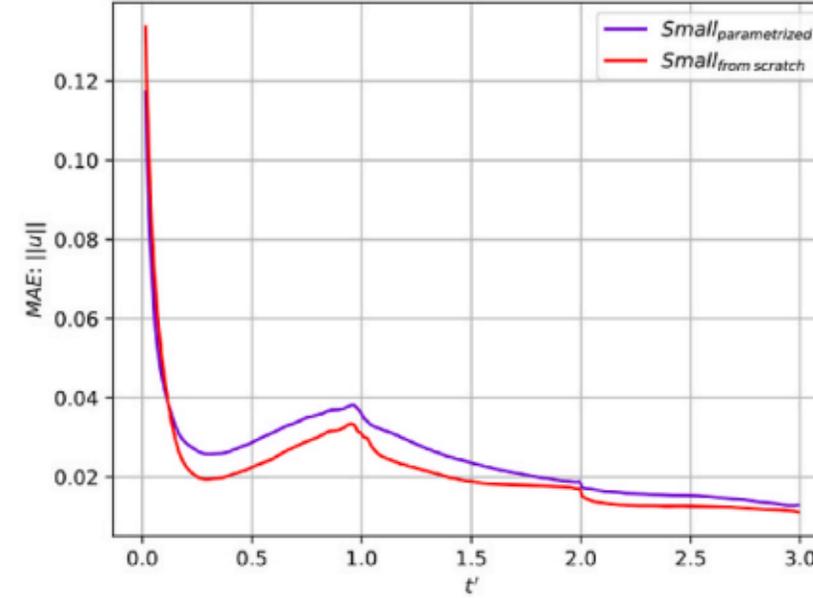
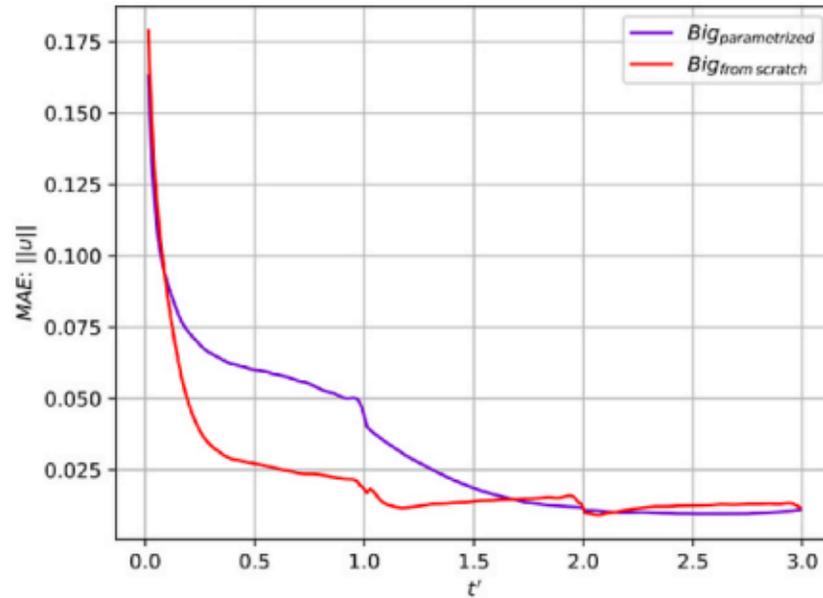
# Parametrized learning

- Box geometry parametrized with
  - width [0.4, 0.6]
  - height [0.15, 0.25]
  - High setup + 150000 epoch (18h:23m)
- Tests considering
  - Big with  $w=0.6$  and  $h=0.25$
  - Small with  $w=0.4$  and  $h=0.15$

Geometry	Training type	$MAE_{avg}$				$MAE_{st}$			
		$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$
Big rectangle	Parameterized	0.032	0.030	0.015	<b>0.157</b>	<b>0.010</b>	<b>0.009</b>	<b>0.005</b>	0.163
	Training from scratch	<b>0.022</b>	<b>0.021</b>	<b>0.010</b>	0.192	0.013	0.012	<b>0.005</b>	<b>0.137</b>
Small rectangle	Parameterized	0.025	0.023	0.017	<b>0.165</b>	0.014	0.013	0.009	<b>0.172</b>
	Training from scratch	<b>0.021</b>	<b>0.020</b>	<b>0.014</b>	0.331	<b>0.012</b>	<b>0.011</b>	<b>0.007</b>	0.474

Last row uses the Mid setup + 100000 epochs (04h:24m)

# Parametrized learning



Geometry	Training type	$MAE_{avg}$				$MAE_{st}$			
		$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$	$\ \mathbf{u}\ $	$u_x$	$u_y$	$p$
Big rectangle	Parameterized	0.032	0.030	0.015	<b>0.157</b>	0.010	0.009	0.005	0.163
	Training from scratch	<b>0.022</b>	<b>0.021</b>	<b>0.010</b>	0.192	0.013	0.012	<b>0.005</b>	<b>0.137</b>
Small rectangle	Parameterized	0.025	0.023	0.017	<b>0.165</b>	0.014	0.013	0.009	<b>0.172</b>
	Training from scratch	<b>0.021</b>	<b>0.020</b>	<b>0.014</b>	0.331	<b>0.012</b>	<b>0.011</b>	<b>0.007</b>	0.474

Last row uses the Mid setup + 100000 epochs (04h:24m)