# Physics-Informed PointNet

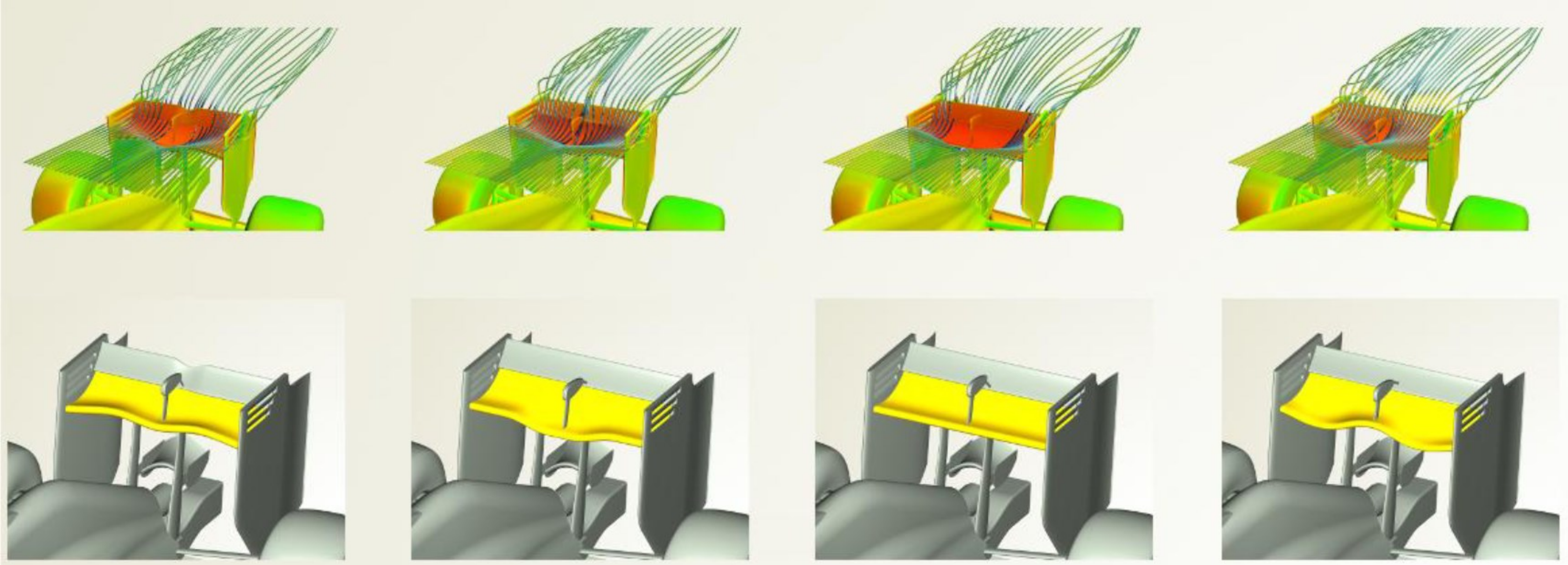
## A deep learning solver for incompressible flows on multiple sets of irregular geometries

**Ali Kashefi**

**Stanford University**

# Physics-Informed PointNet (PIPN)

## Introduction & Motivation



What is the best (optimized) shape of rear wing?

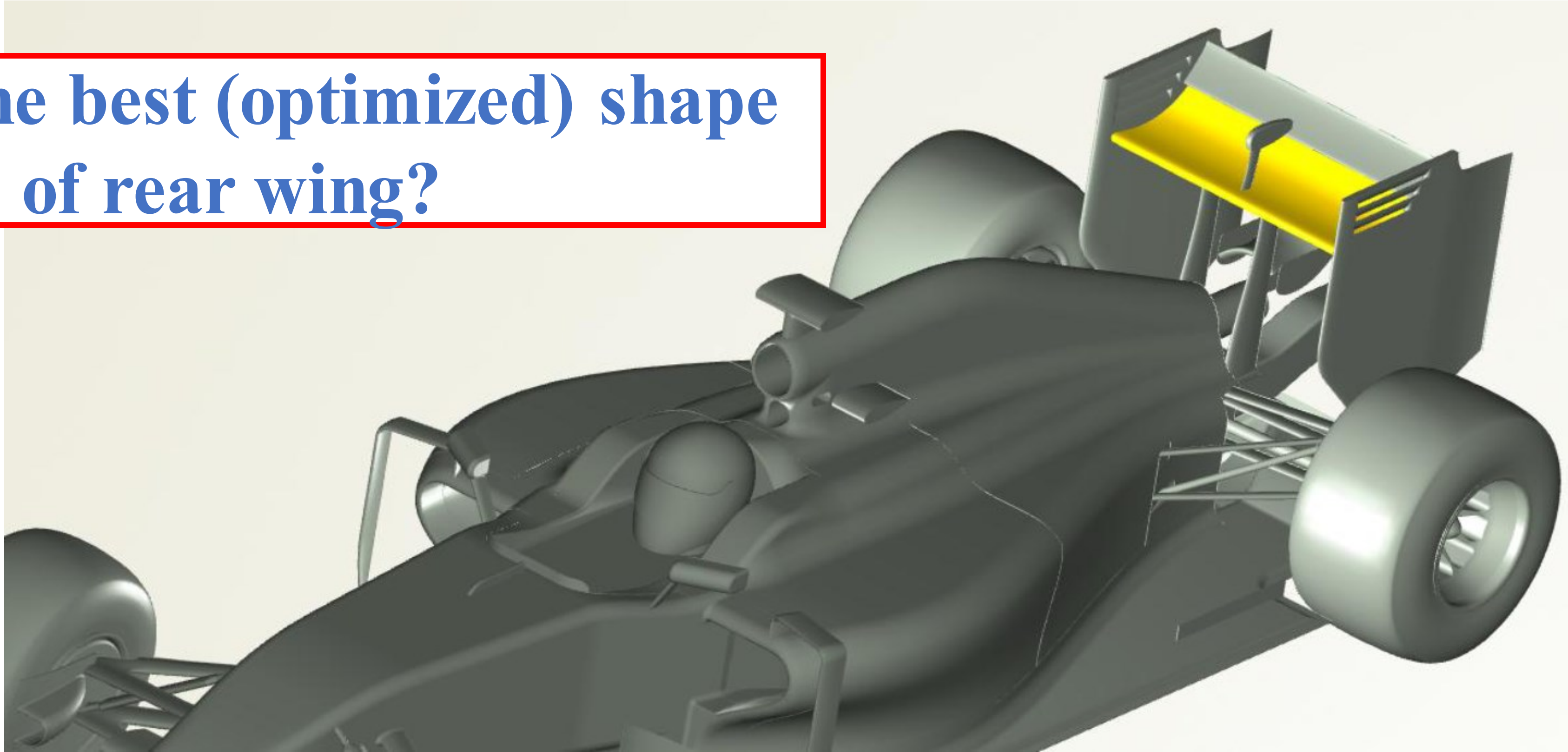
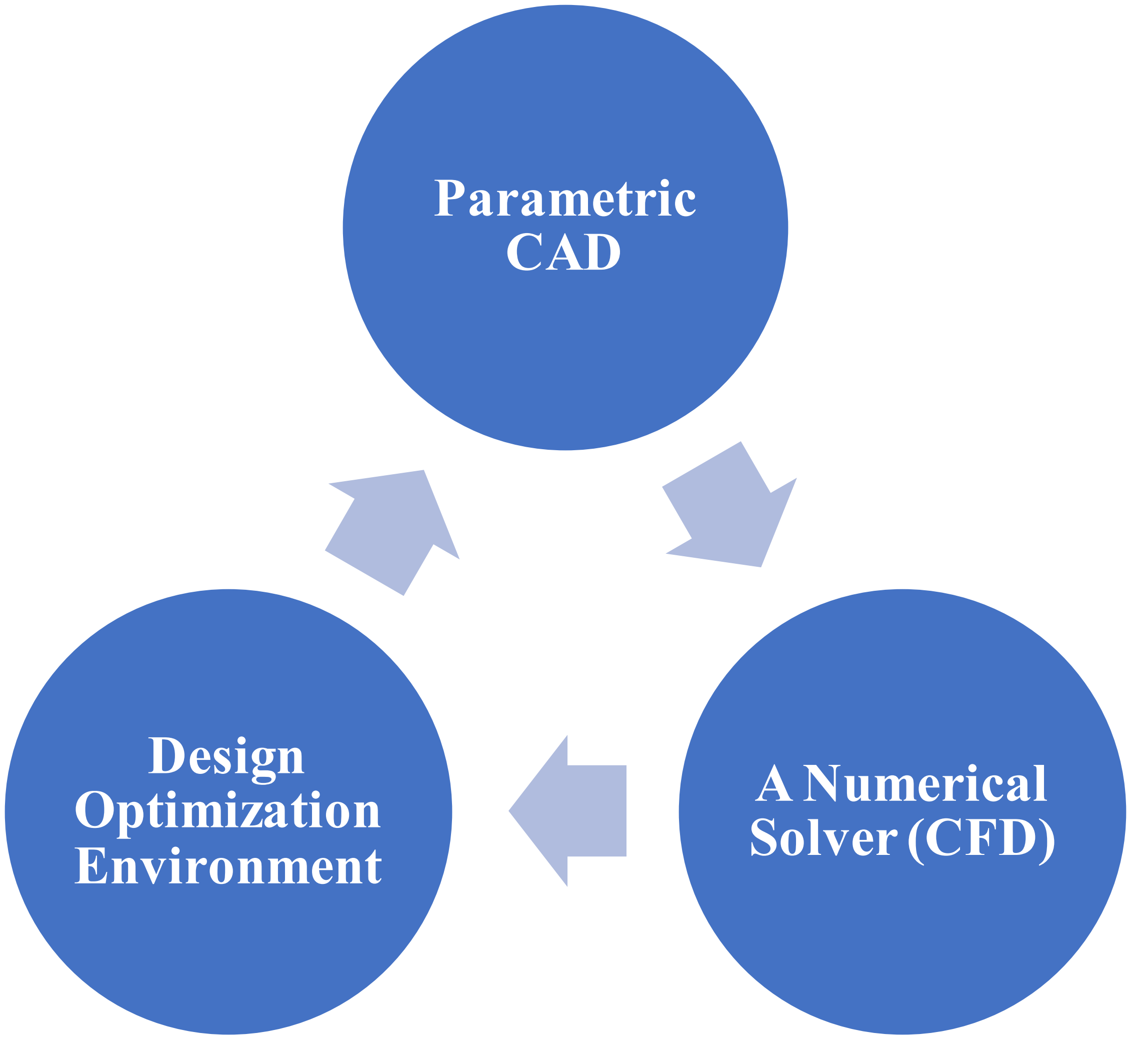
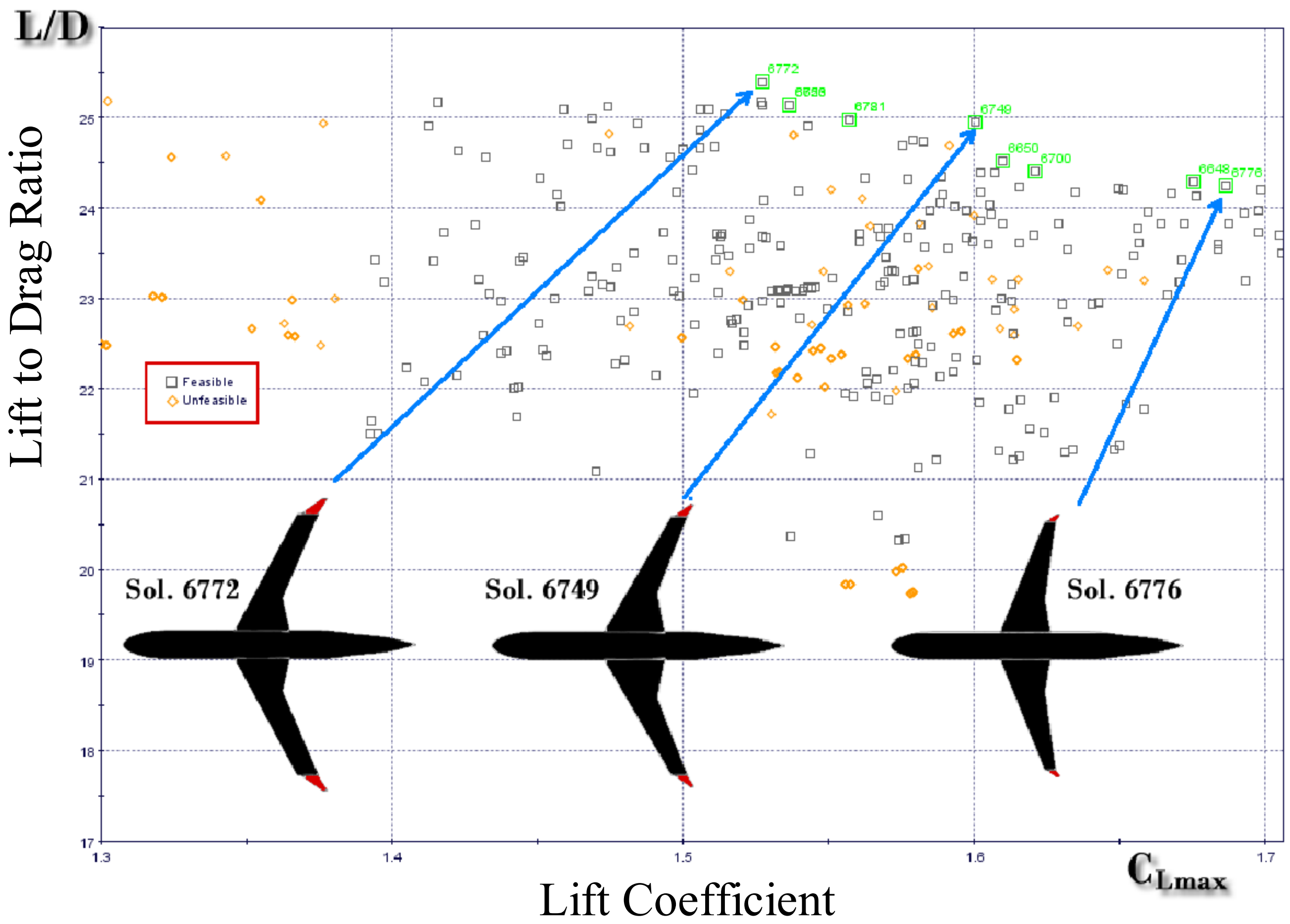


Image credits belongs to the company

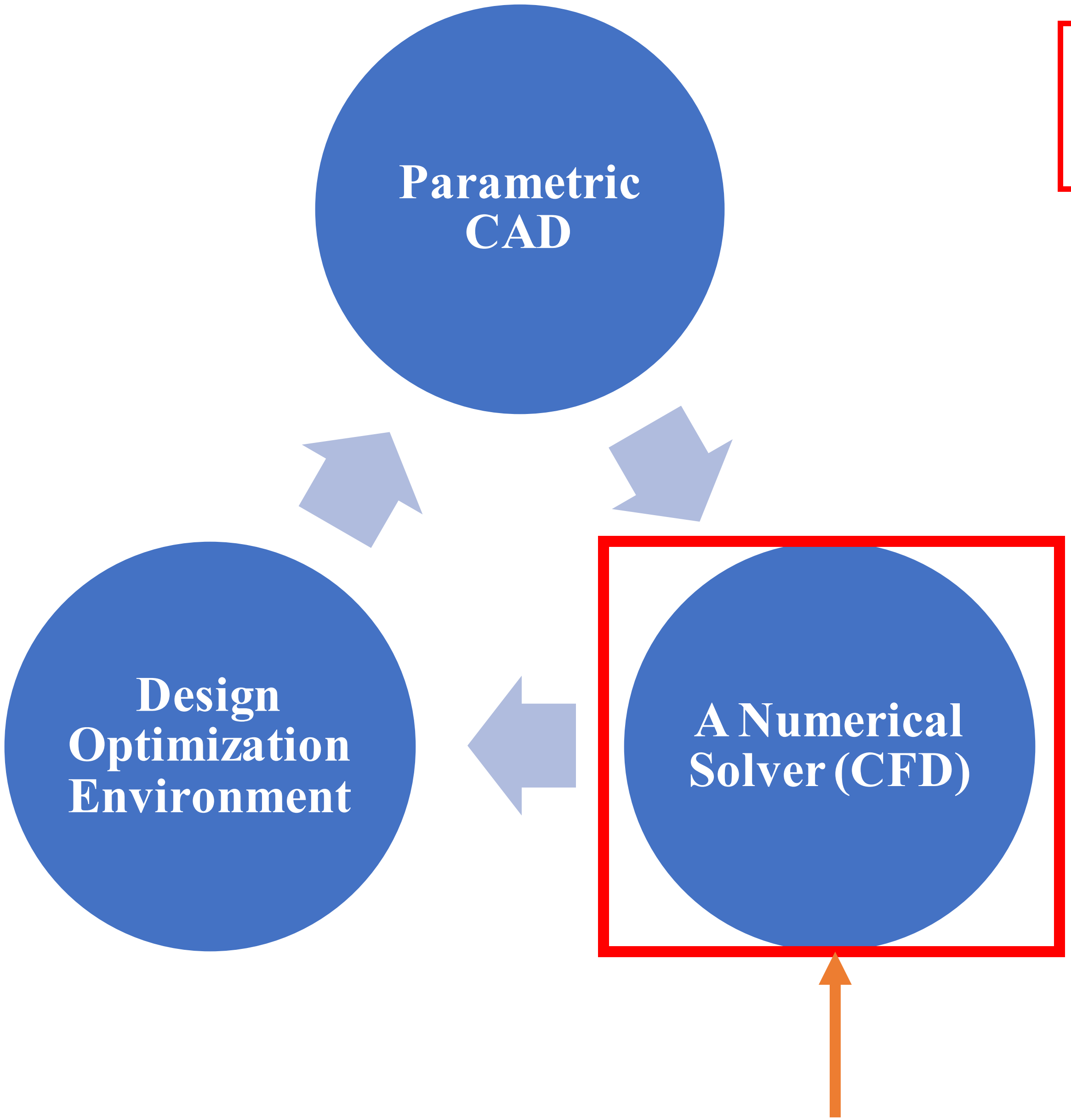


## What is the optimized shape of wing?

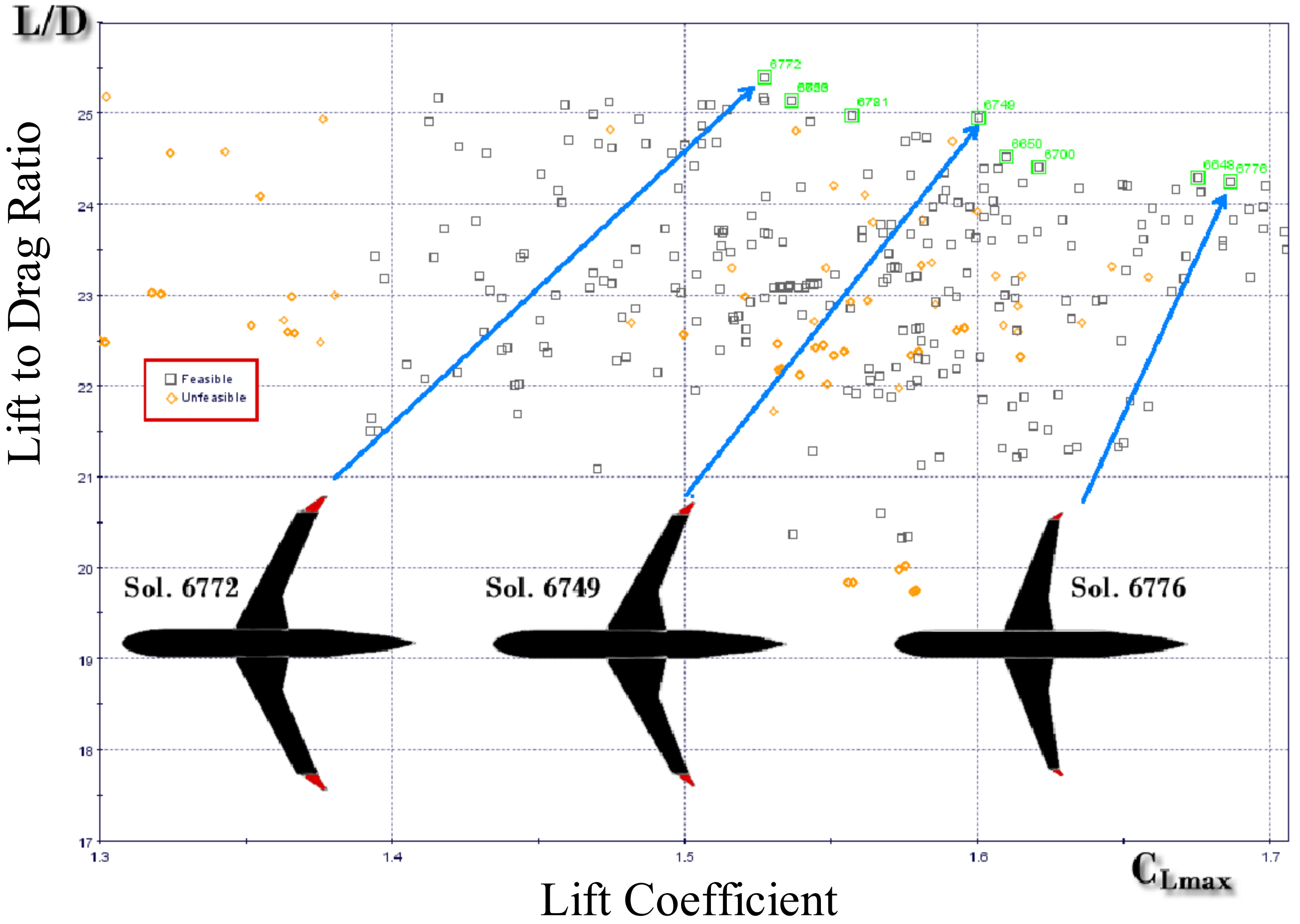


Mattos et al. (2015)

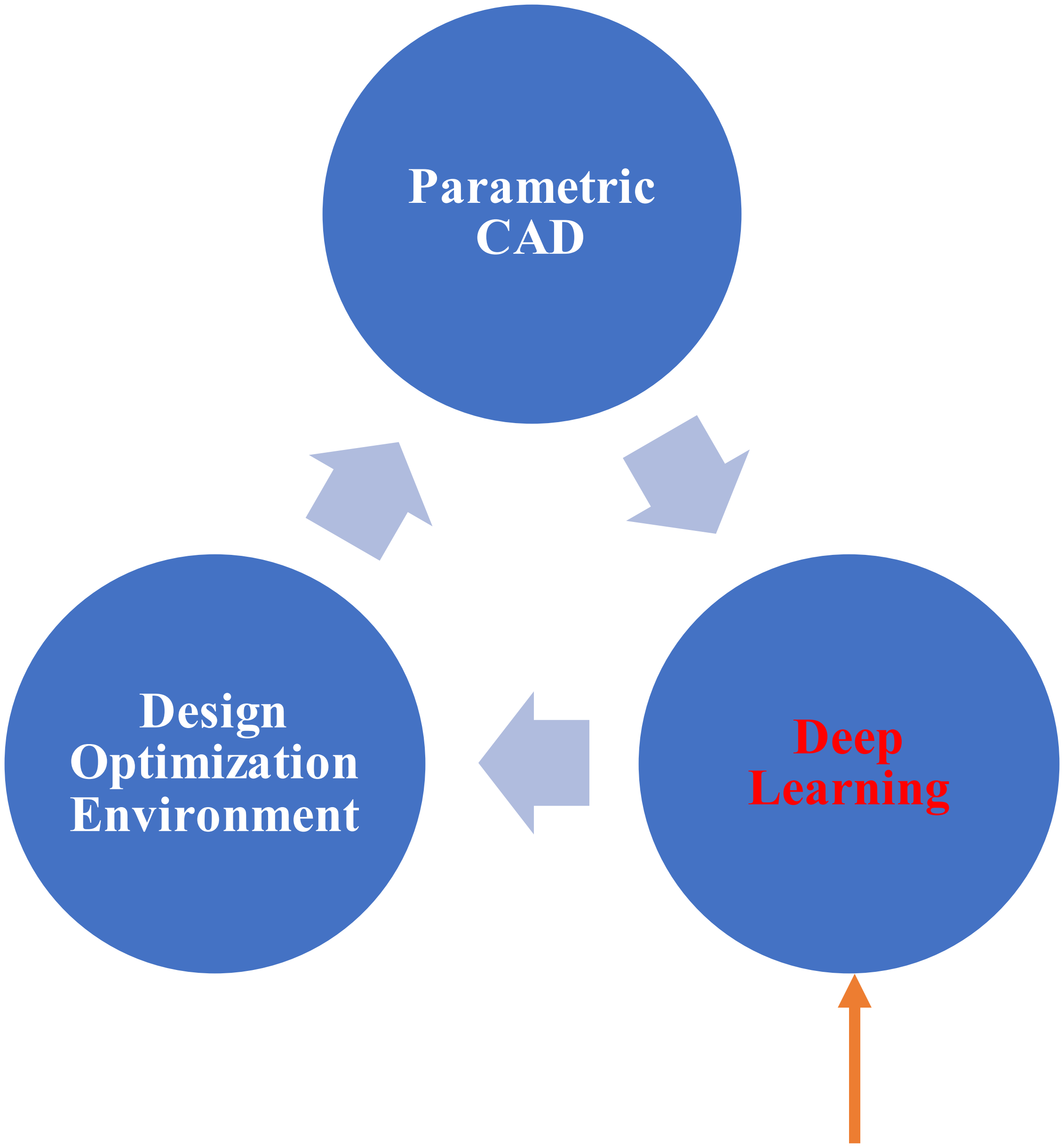
# What is the optimized shape of wing?



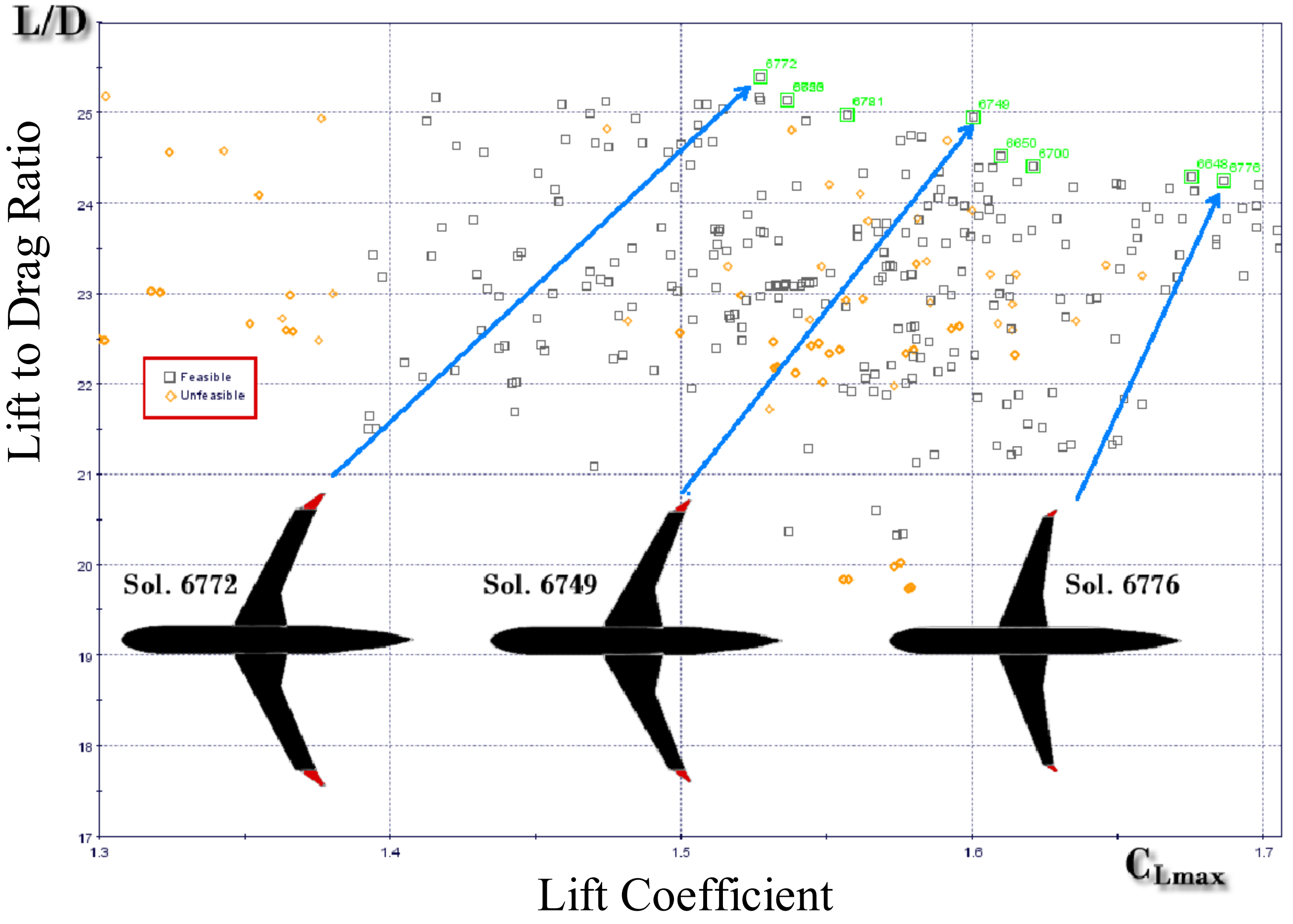
The most time consuming part



Mattos et al. (2015)

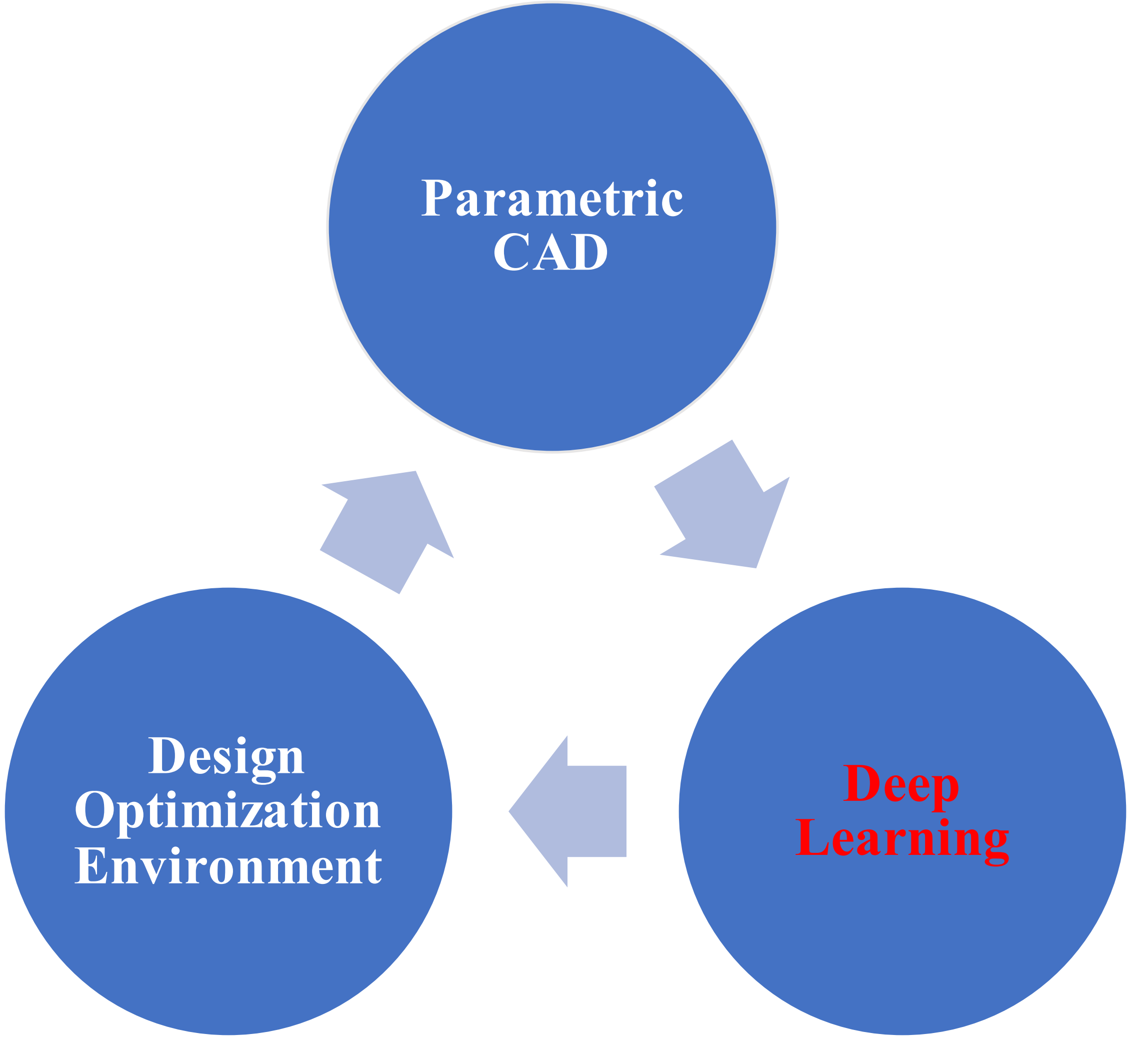


## What is the optimized shape of wing?



Accelerating by Machine Learning

Mattos et al. (2015)



Parametric CAD

Deep Learning

Design Optimization Environment

1. Supervised learning

plentiful labeled data (observations)

2. Unsupervised/weakly supervised learning

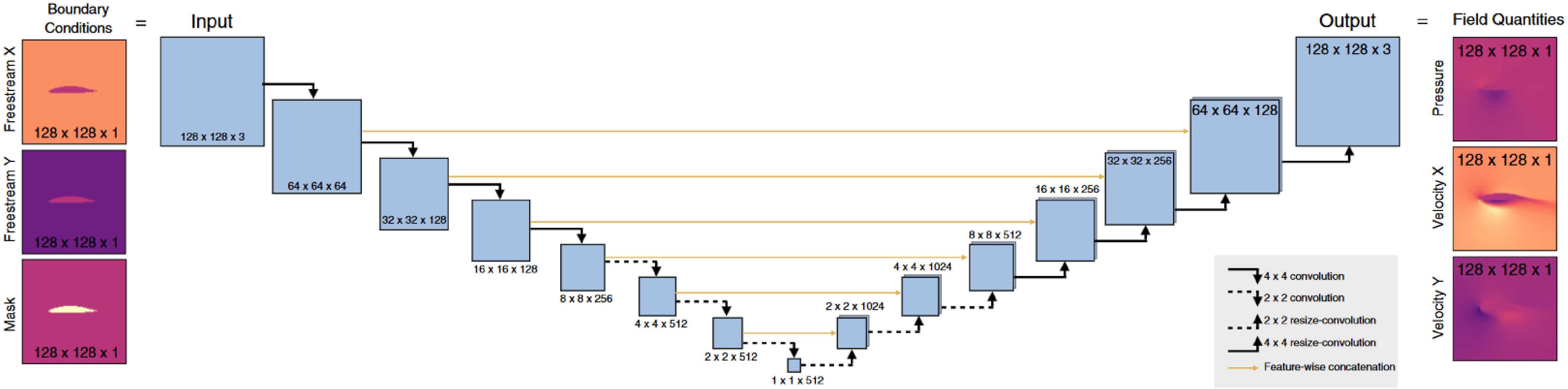
no labeled data or only sparse labeled data (observations)

# Deep Learning Methods for Reynolds-Averaged Navier–Stokes Simulations of Airfoil Flows

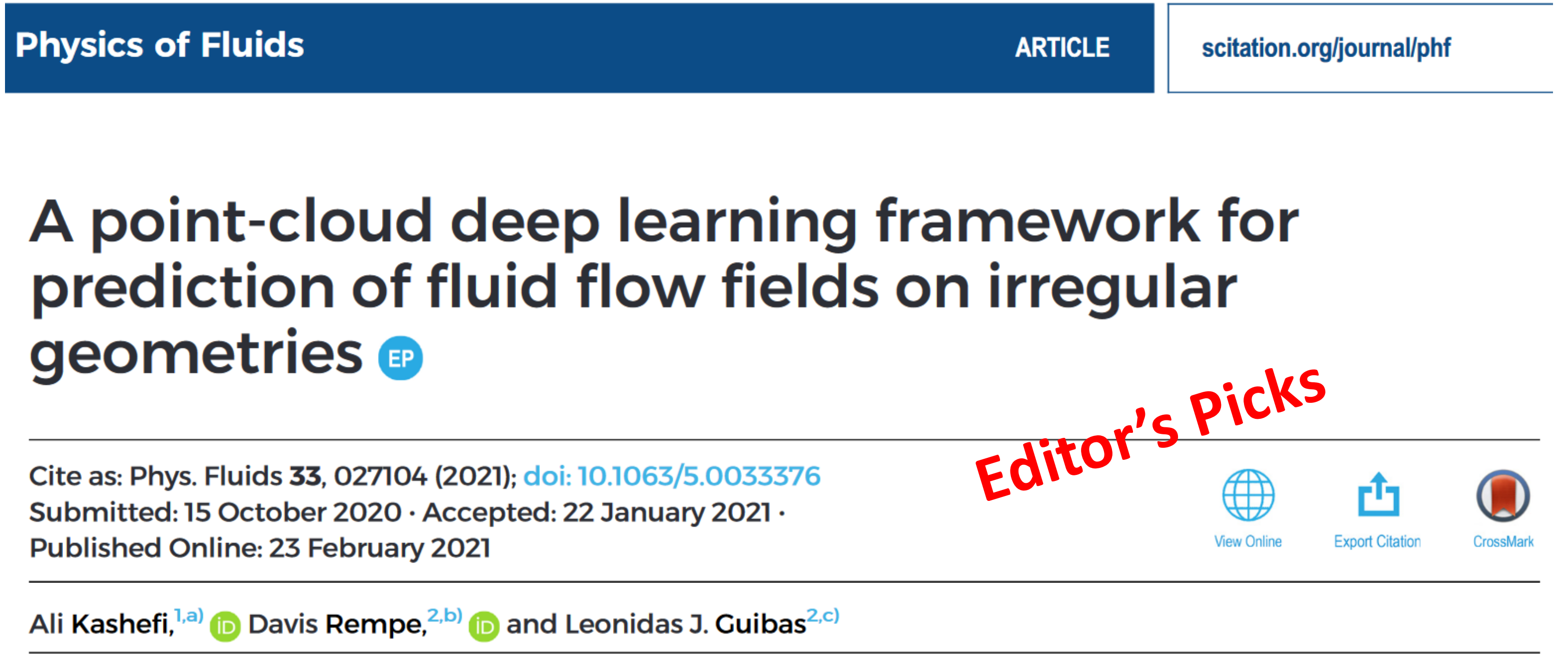
Supervised Learning

N. Thuerey et al. (2020)

*Technical University of Munich*



**Supervised Learning**



Physics of Fluids ARTICLE scitation.org/journal/phf

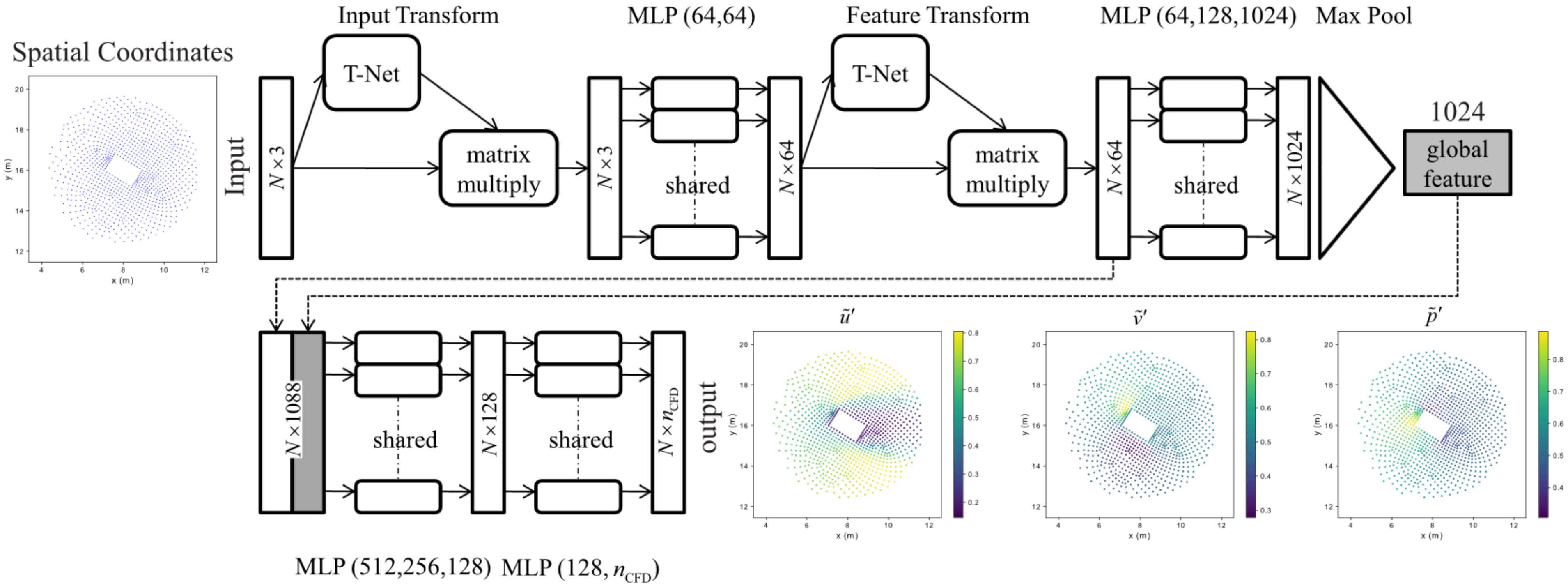
# A point-cloud deep learning framework for prediction of fluid flow fields on irregular geometries EP

Editor's Picks

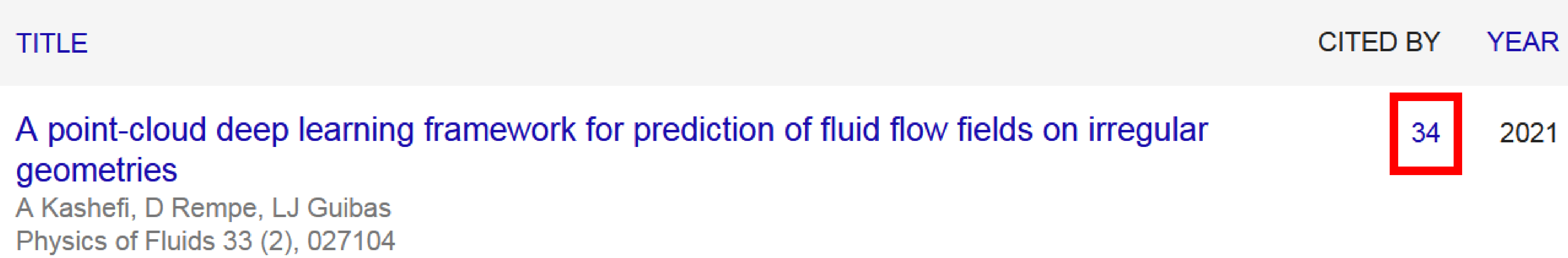
Cite as: Phys. Fluids **33**, 027104 (2021); doi: 10.1063/5.0033376
Submitted: 15 October 2020 · Accepted: 22 January 2021 ·
Published Online: 23 February 2021

View Online Export Citation CrossMark

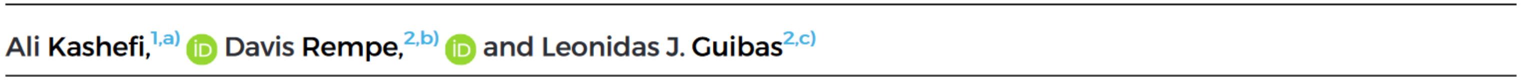
Ali Kashefi,[1,a)] Davis Rempe,[2,b)] and Leonidas J. Guibas[2,c)]



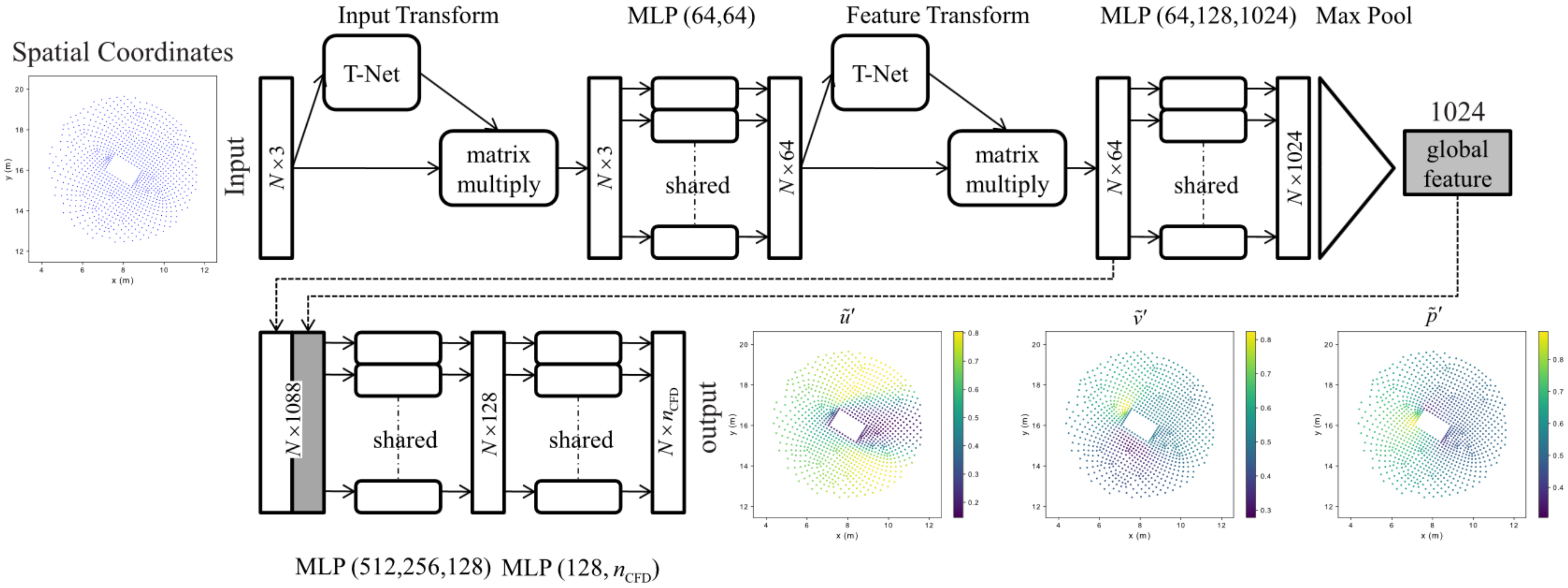
Physics of Fluids ARTICLE scitation.org/journal/phf



| TITLE | CITED BY | YEAR |
|---|---|---|
| A point-cloud deep learning framework for prediction of fluid flow fields on irregular geometries<br>A Kashefi, D Rempe, LJ Guibas<br>Physics of Fluids 33 (2), 027104 | 34 | 2021 |



Ali Kashefi,[1,a] Davis Rempe,[2,b] and Leonidas J. Guibas[2,c]



## Supervised Learning Framework

*Let's explain it by an example:*
Imagine we would like to obtain the velocity fields around 1000 airfoils with different geometries

Step#1 Obtain the velocity fields for 900 domains using a numerical solver (or a lab experiment)

Step#2 Train a neural network on these 900 domains (training set)

Step#3 By the neural network, predict the velocity fields on the remaining 100 domains (test set)

## Supervised Learning Framework

Producing plentiful labeled data is expensive!
Sometimes, labeled data are not accessible!

Step#1 Obtain the velocity fields for 900 domains using a numerical solver (or a lab experiment)

Step#2 Train a neural network on these 900 domains (training set)

Step#3 By the neural network, predict the velocity fields on the remaining 100 domains (test set)

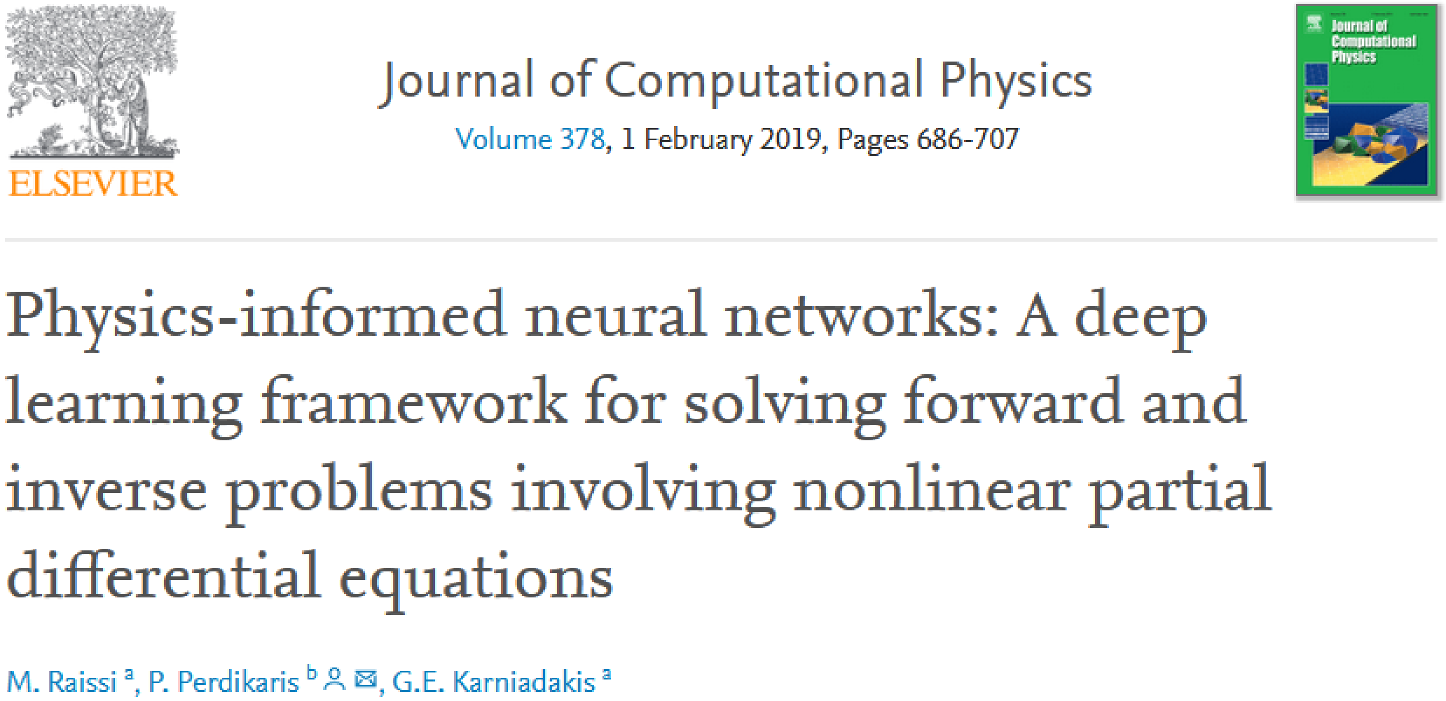
# Unsupervised/Weakly Supervised Learning Framework

**Our goal: Designing a neural network to predict the solution on multiple domains without plentiful labeled data**

Step#1 Obtain the velocity fields for 900 domains using a numerical solver (or a lab experiment)

Step#2 Train a neural network on these 900 domains

Step#3 By the neural network, predict the velocity fields on the remaining 100 domains



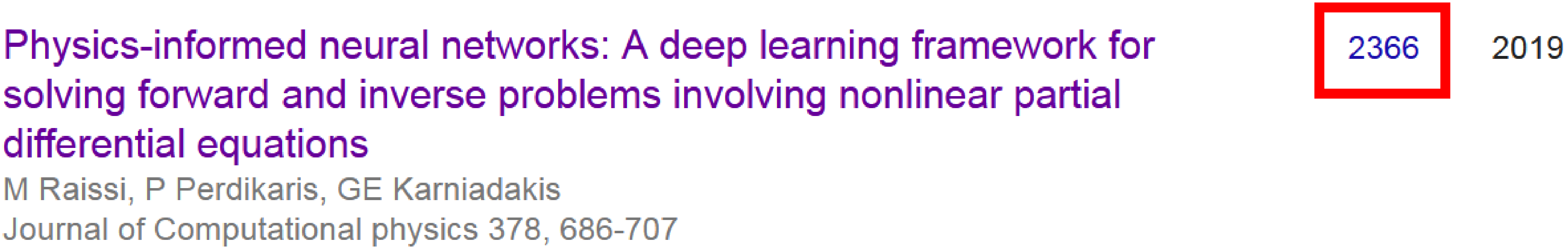
Journal of Computational Physics
Volume 378, 1 February 2019, Pages 686-707

# Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations

M. Raissi [a], P. Perdikaris [b], G.E. Karniadakis [a]

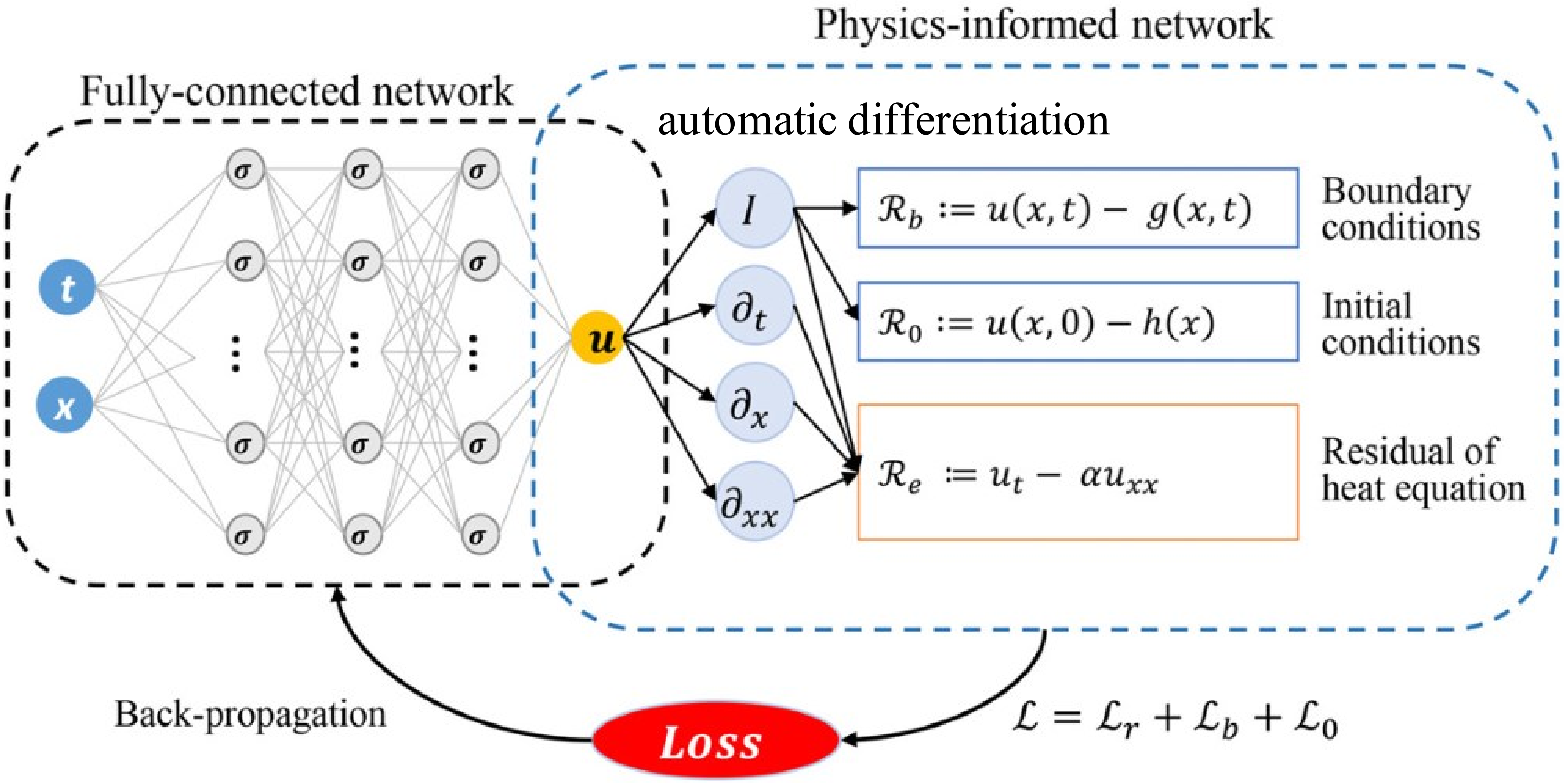
BROWN

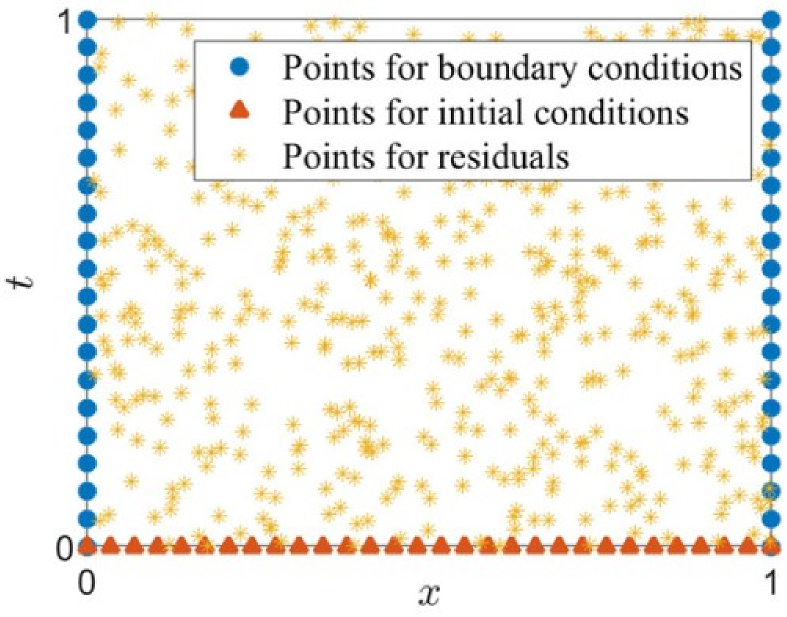
Crunch Group



Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations
M Raissi, P Perdikaris, GE Karniadakis
Journal of Computational physics 378, 686-707

2366 2019

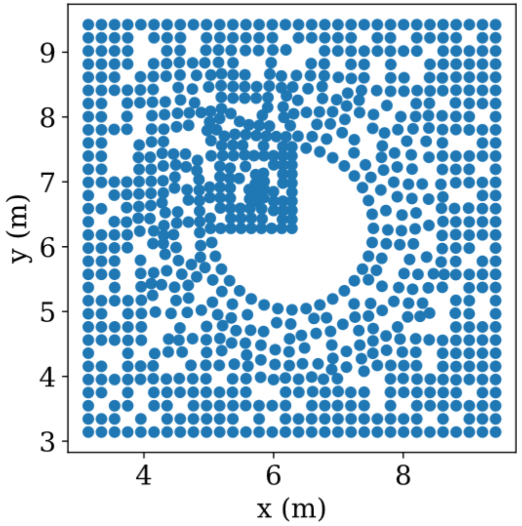

Physics-informed network
Fully-connected network
automatic differentiation
t
x
σ
u
I
$\partial_t$
$\partial_x$
$\partial_{xx}$
$\mathcal{R}_b := u(x,t) - g(x,t)$
Boundary conditions
$\mathcal{R}_0 := u(x,0) - h(x)$
Initial conditions
$\mathcal{R}_e := u_t - \alpha u_{xx}$
Residual of heat equation
Back-propagation
Loss
$\mathcal{L} = \mathcal{L}_r + \mathcal{L}_b + \mathcal{L}_0$

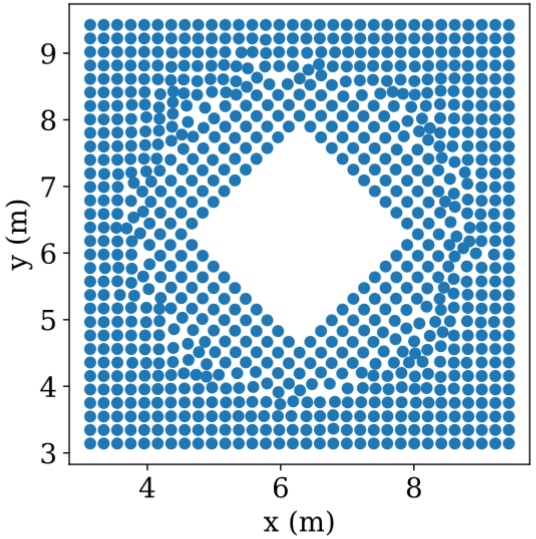


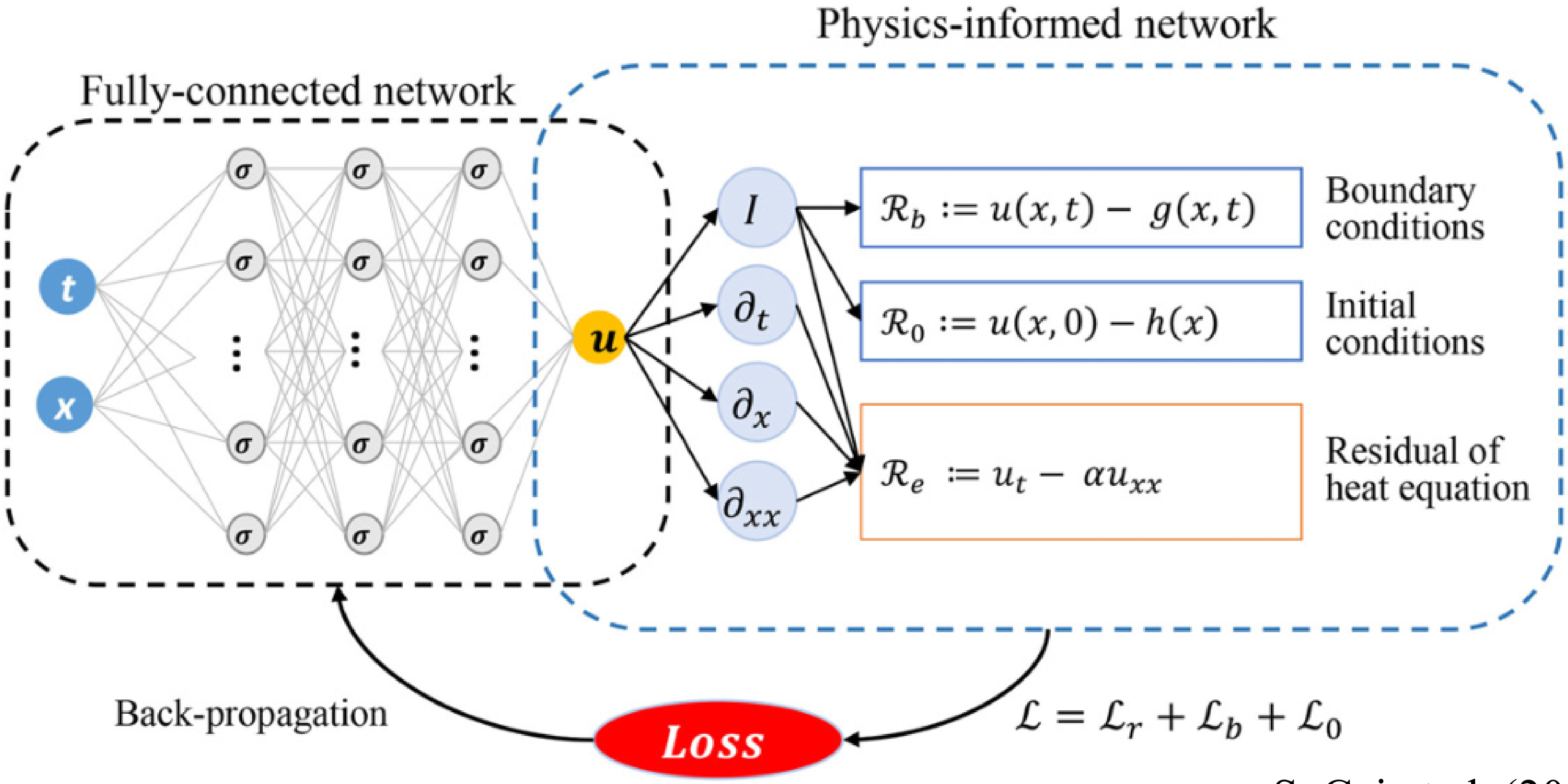
S. Cai et al. (2021)

**Loss function = governing equations + boundary conditions + initial conditions + sparse observation**

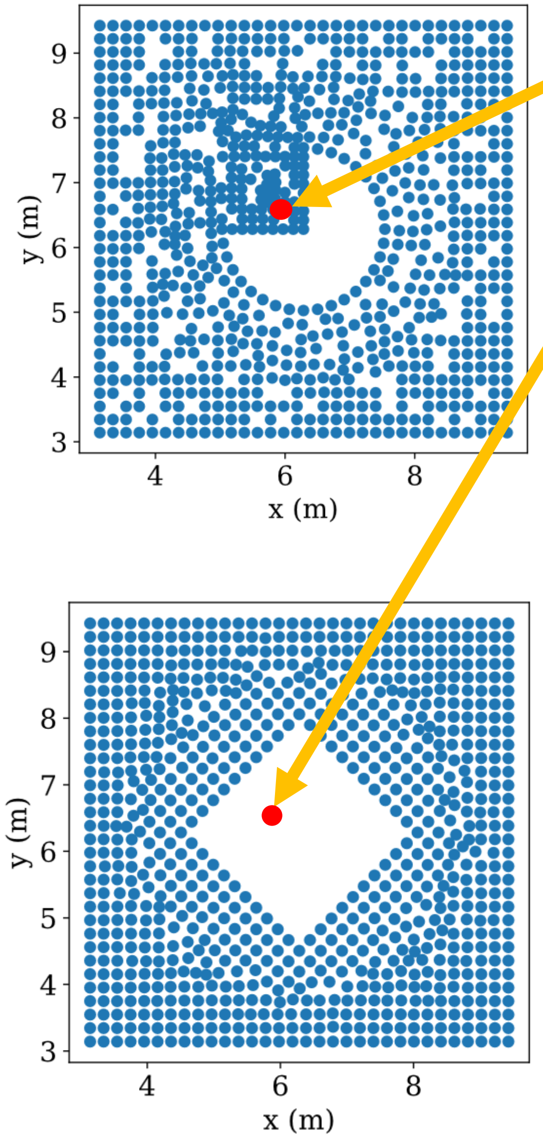
## Can PINN obtain the solution over more than one different geometries simultaneously (i.e., in one training set)?



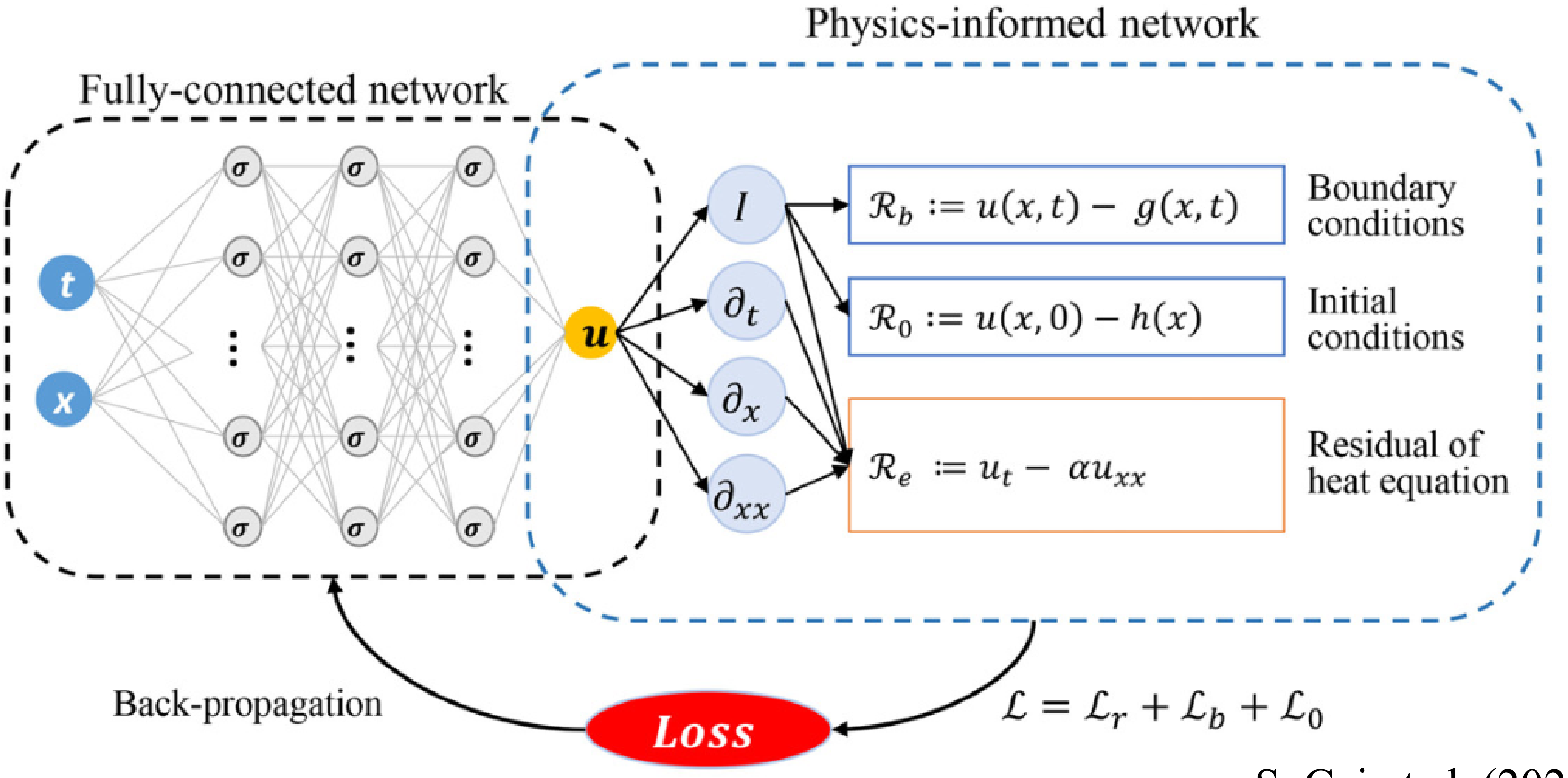



Physics-informed network
Fully-connected network
$t$
$x$
$\sigma$
$u$
$I$
$\partial_t$
$\partial_x$
$\partial_{xx}$
$\mathcal{R}_b := u(x,t) - g(x,t)$
Boundary conditions
$\mathcal{R}_0 := u(x,0) - h(x)$
Initial conditions
$\mathcal{R}_e := u_t - \alpha u_{xx}$
Residual of heat equation
Back-propagation
Loss
$\mathcal{L} = \mathcal{L}_r + \mathcal{L}_b + \mathcal{L}_0$

S. Cai et al. (2021)

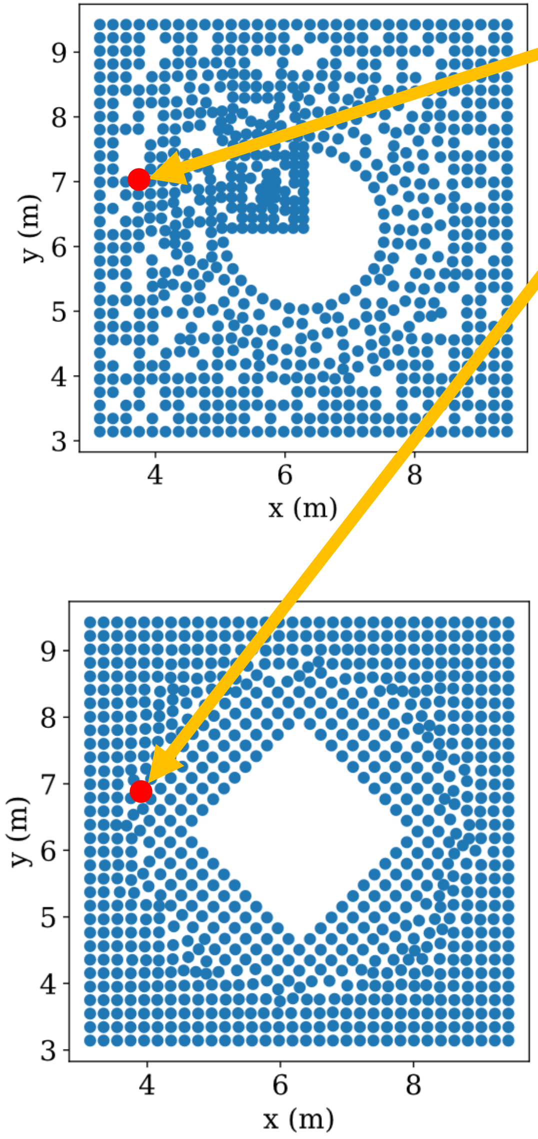


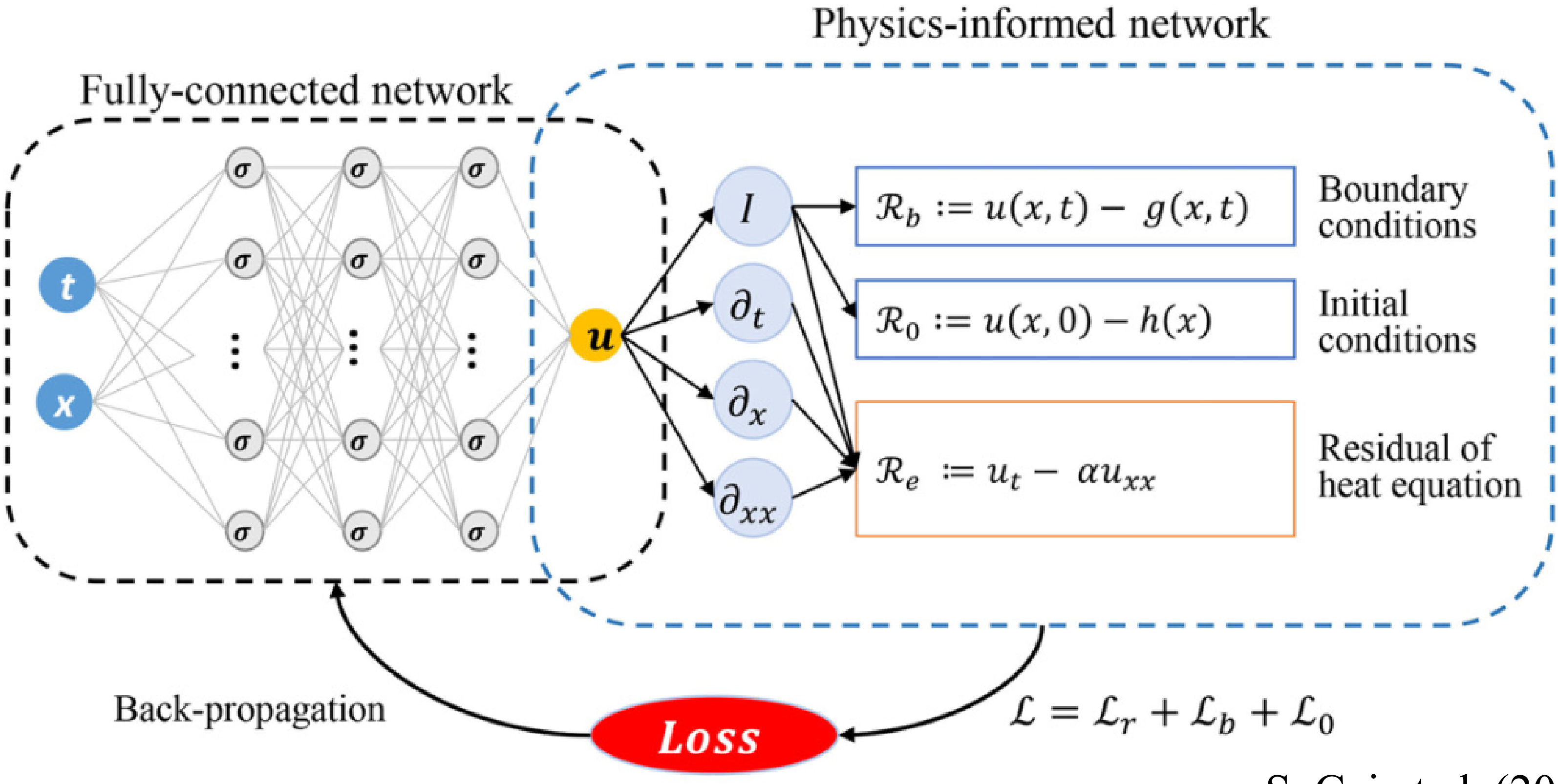
Uncommon points in these two domain


Physics-informed network
Fully-connected network
$t$
$x$
$\sigma$
$u$
$I$
$\partial_t$
$\partial_x$
$\partial_{xx}$
$\mathcal{R}_b := u(x,t) - g(x,t)$
Boundary conditions
$\mathcal{R}_0 := u(x,0) - h(x)$
Initial conditions
$\mathcal{R}_e := u_t - \alpha u_{xx}$
Residual of heat equation
Back-propagation
Loss
$\mathcal{L} = \mathcal{L}_r + \mathcal{L}_b + \mathcal{L}_0$

S. Cai et al. (2021)

Even in common points, the solution might be different in each of these two domains

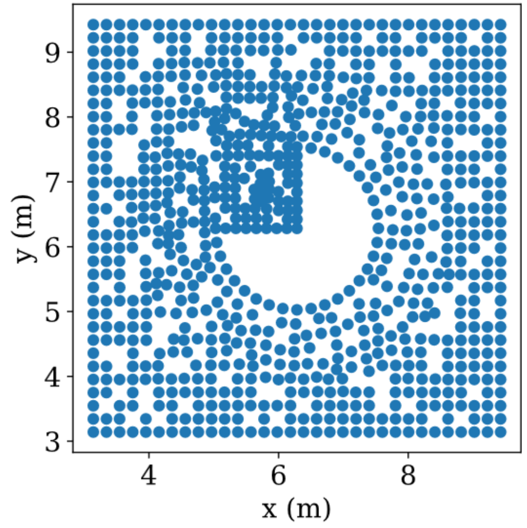


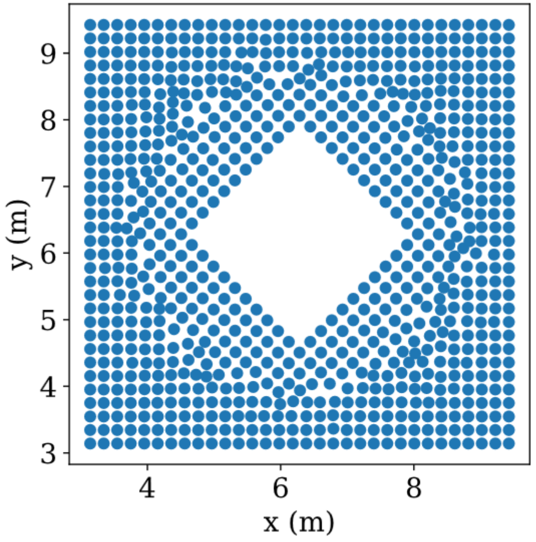


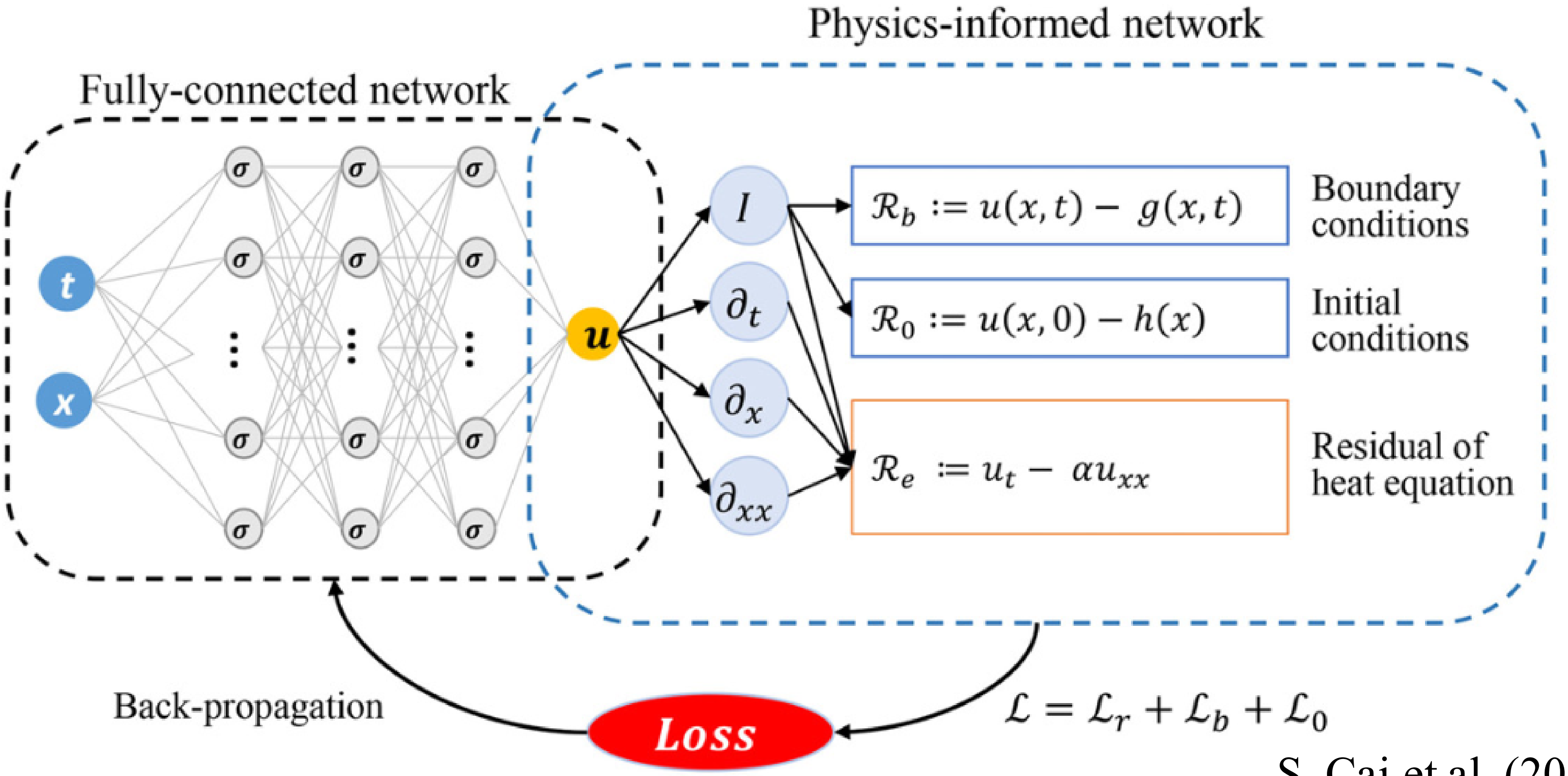
Physics-informed network
Fully-connected network
$t$
$x$
$\sigma$
$u$
$I$
$\partial_t$
$\partial_x$
$\partial_{xx}$
$\mathcal{R}_b := u(x,t) - g(x,t)$
Boundary conditions
$\mathcal{R}_0 := u(x,0) - h(x)$
Initial conditions
$\mathcal{R}_e := u_t - \alpha u_{xx}$
Residual of heat equation
Back-propagation
Loss
$\mathcal{L} = \mathcal{L}_r + \mathcal{L}_b + \mathcal{L}_0$

S. Cai et al. (2021)

## Can PINN obtain the solution over more than one different geometries simultaneously (i.e., in one training set)?





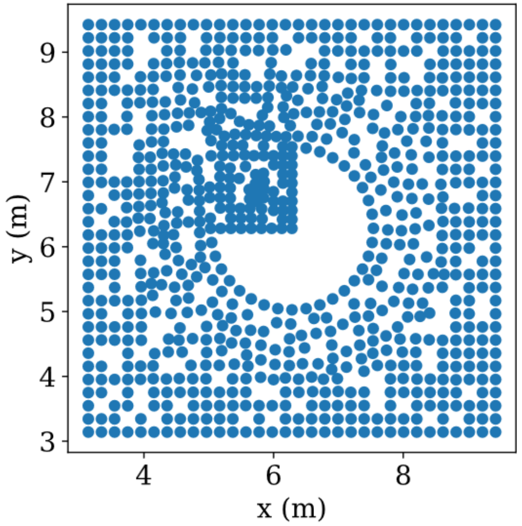


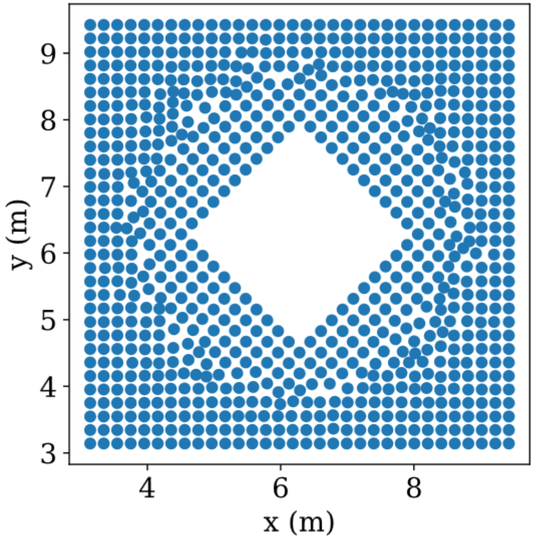
Physics-informed network
Fully-connected network
t
x
σ
u
I
$\partial_t$
$\partial_x$
$\partial_{xx}$
$\mathcal{R}_b := u(x,t) - g(x,t)$
Boundary conditions
$\mathcal{R}_0 := u(x,0) - h(x)$
Initial conditions
$\mathcal{R}_e := u_t - \alpha u_{xx}$
Residual of heat equation
Back-propagation
Loss
$\mathcal{L} = \mathcal{L}_r + \mathcal{L}_b + \mathcal{L}_0$

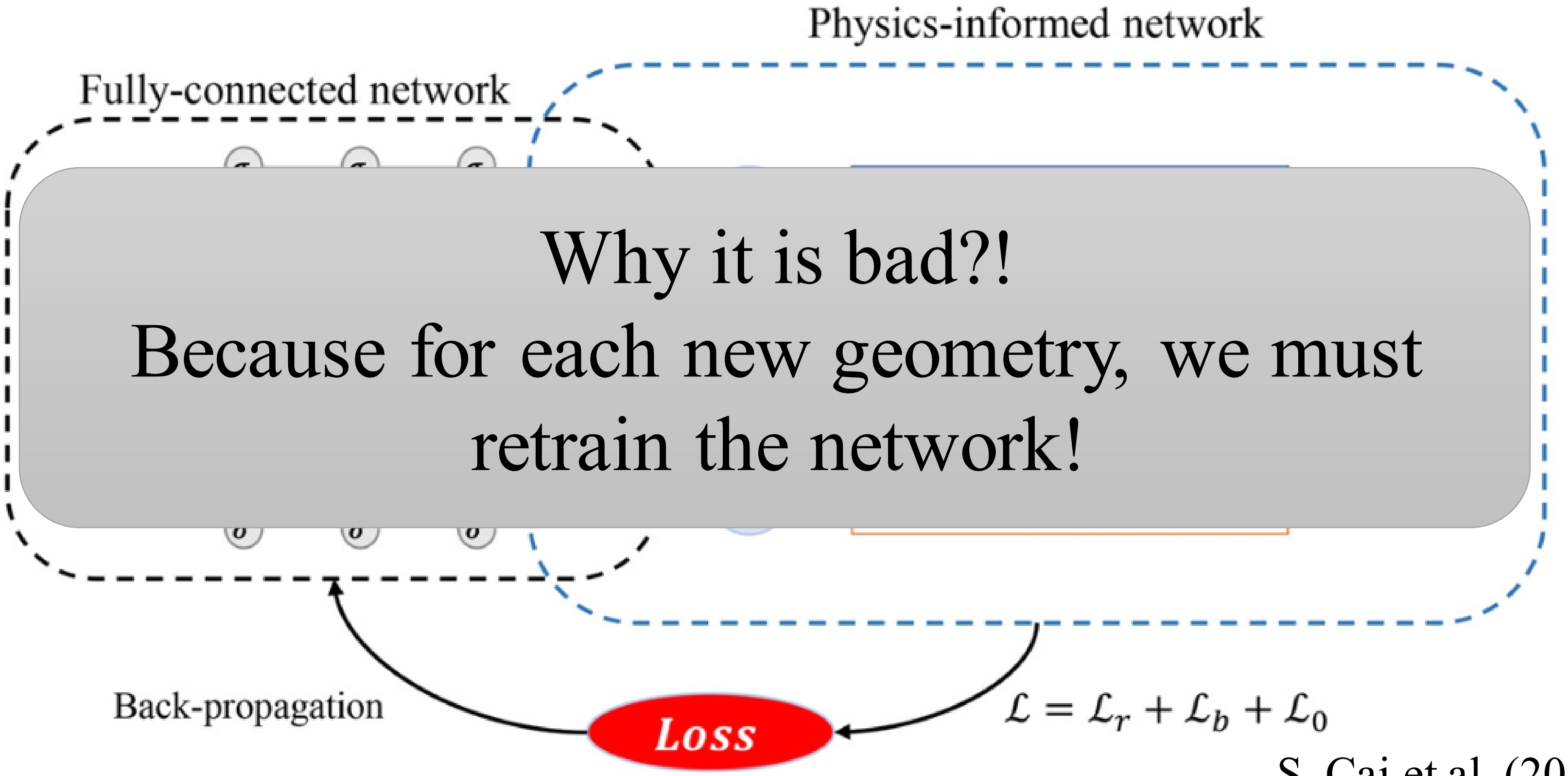
S. Cai et al. (2021)

**No, because there is no mechanism in the fully connected network to capture geometric variations!**

**Can PINN obtain the solution over more than one different geometries simultaneously (i.e., in one training set)?**





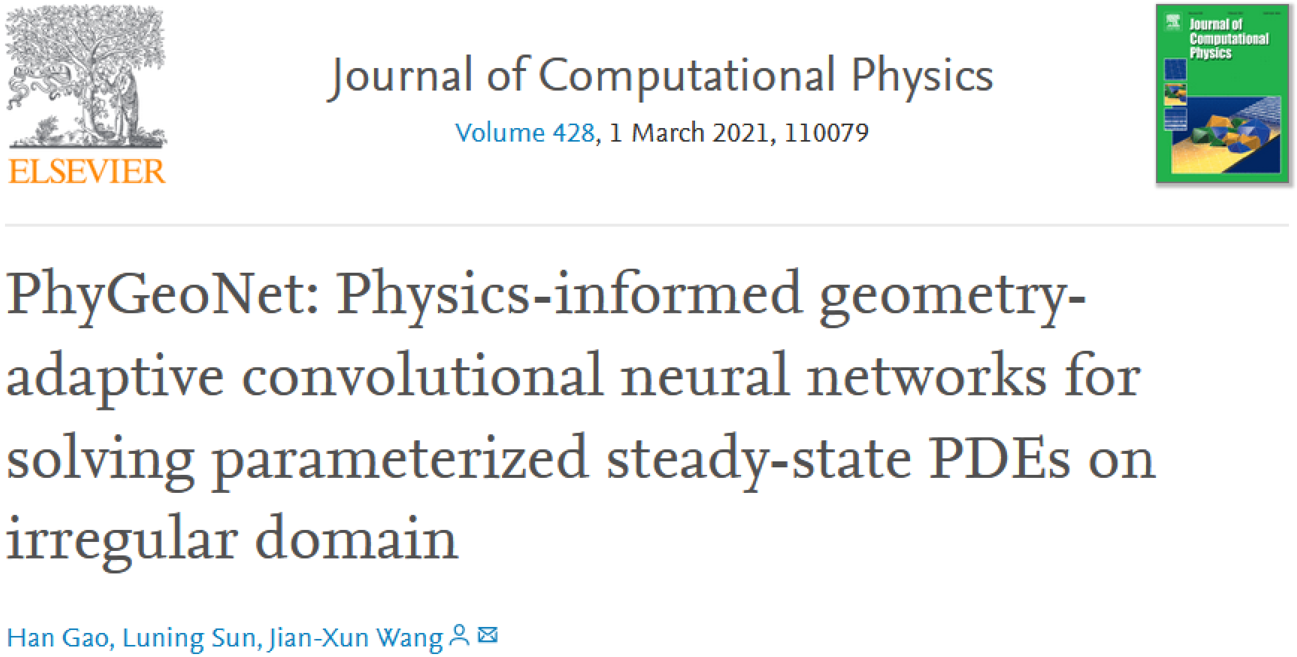


Why it is bad?!

Because for each new geometry, we must retrain the network!

S. Cai et al. (2021)

**No, because there is no mechanism in the fully connected network to capture geometric variations!**



ELSEVIER

Journal of Computational Physics

Volume 428, 1 March 2021, 110079

## PhyGeoNet: Physics-informed geometry-adaptive convolutional neural networks for solving parameterized steady-state PDEs on irregular domain

Han Gao, Luning Sun, Jian-Xun Wang

# PhyGeoNet strategies:

- Capturing geometric features using encoders in CNNs
- Using non-trainable filter representing a finite difference stencil, instead of using automatic differentiation
- Using elliptic coordinate transformations for irregular geometries

# PhyGeoNet architecture

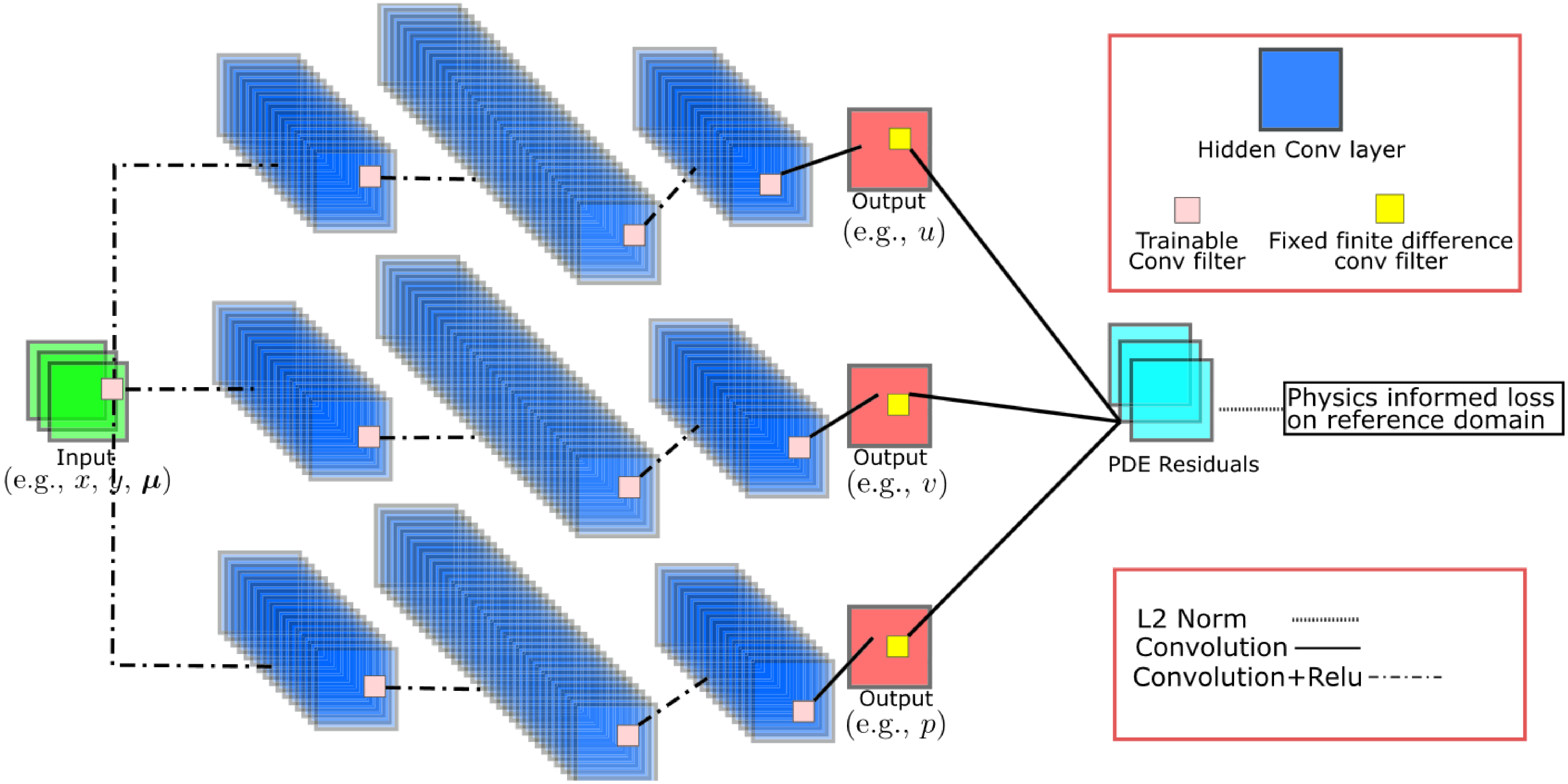


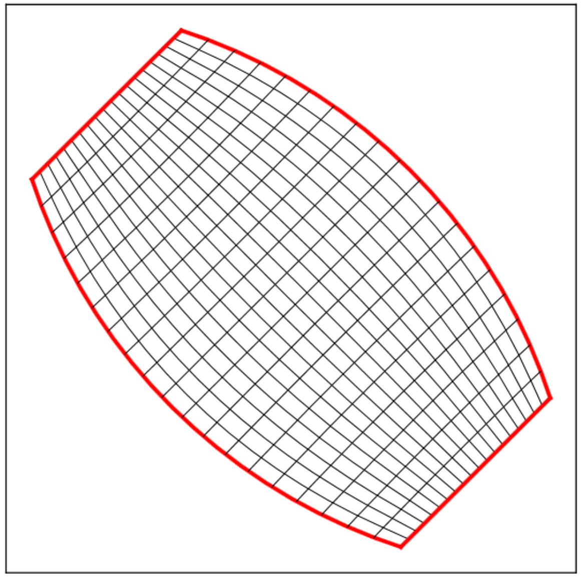
Figure taken from the PhyGeoNet journal paper (Gao et al. 2021)

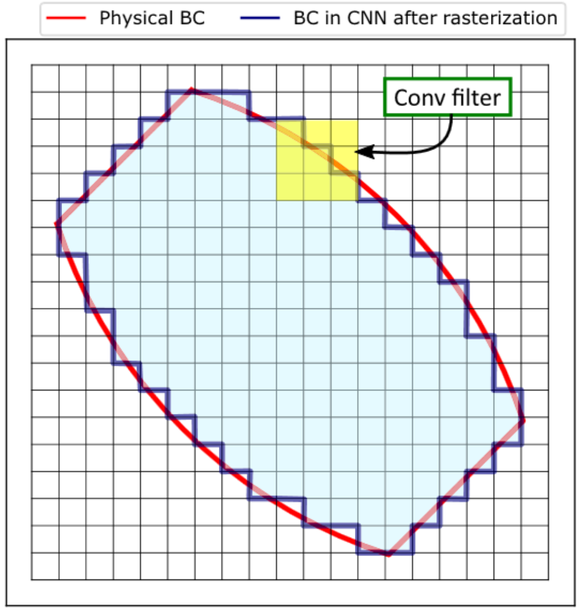
# PhyGeoNet limitations:

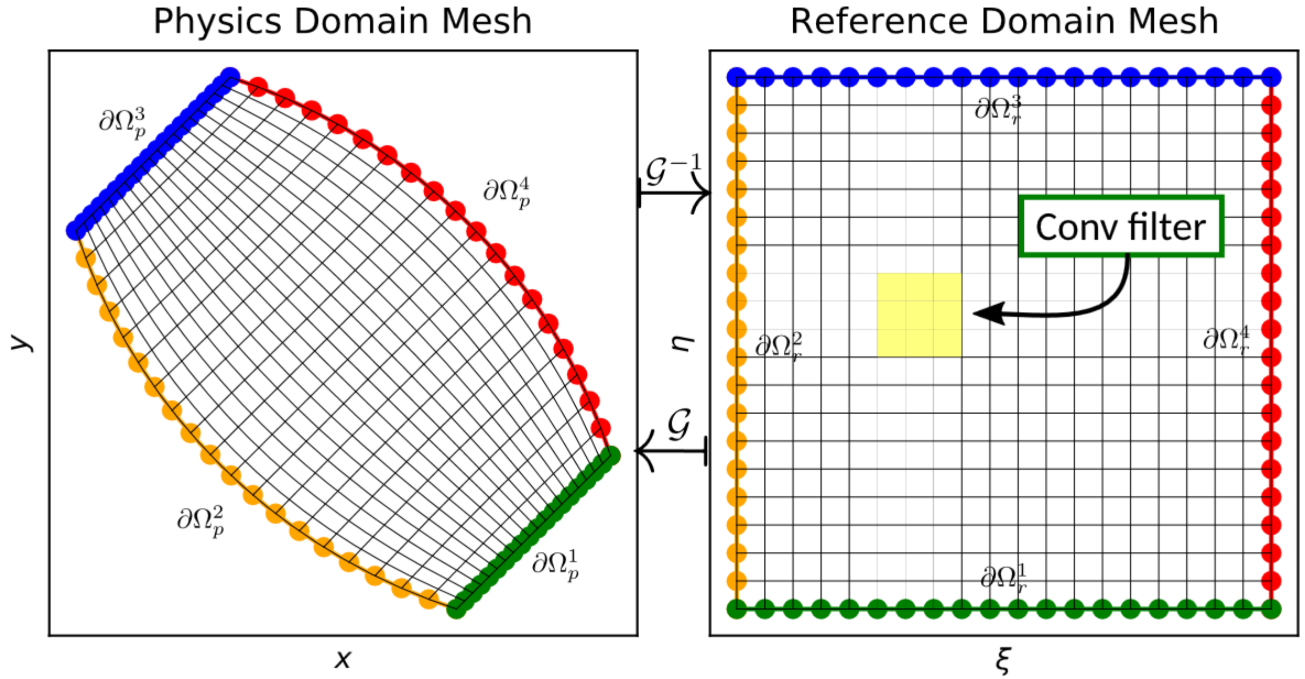
- Limitations of finite difference schemes such as order of accuracy and issues of high order methods near boundaries

- Elliptic coordinate transformations require offline efforts

- Cannot handle more than five $C_0$ continuous boundaries



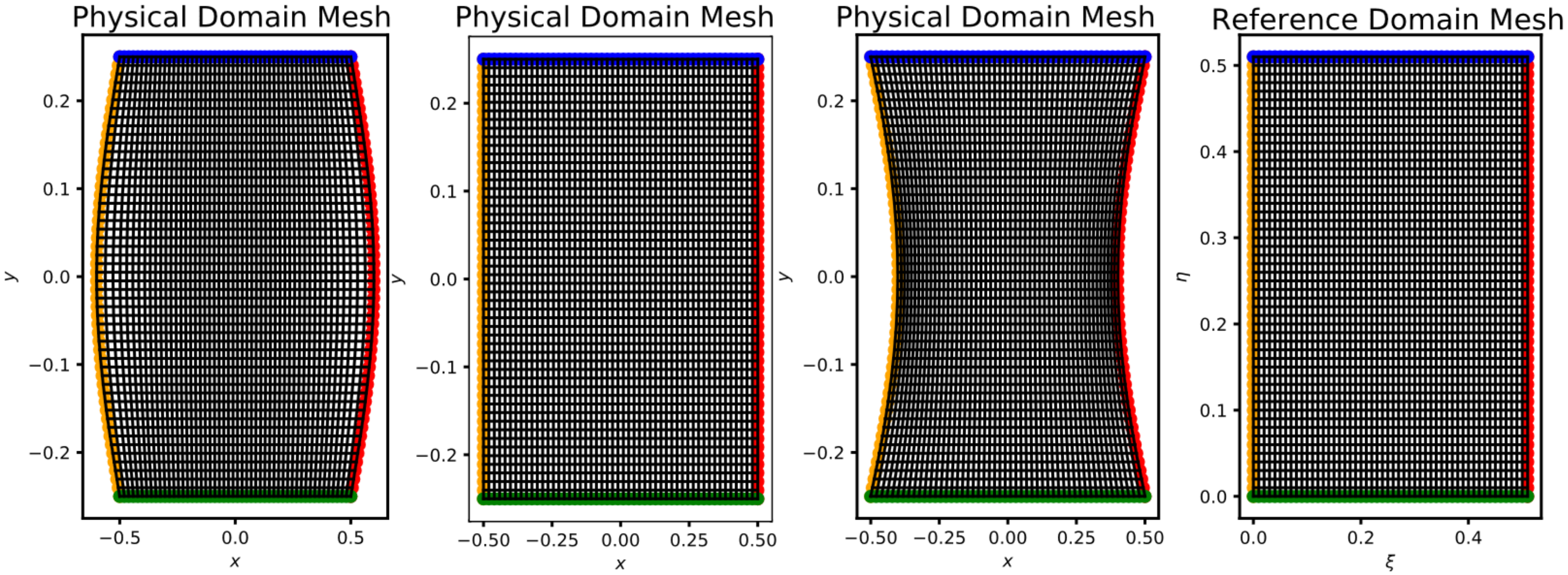


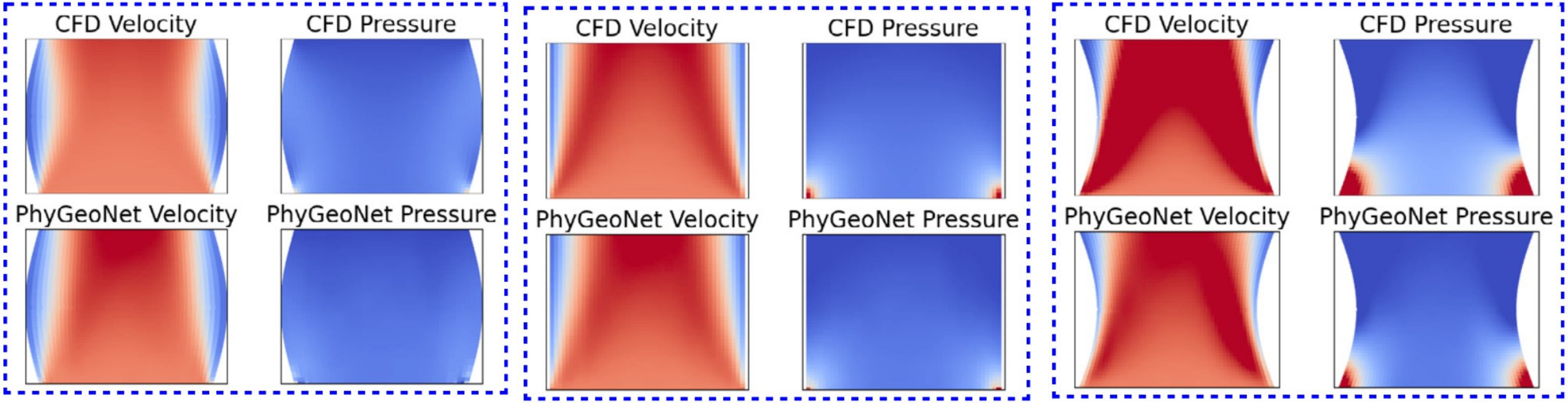


# parameterized vascular geometries

$$x_l = s\cos(2\pi y_l) - 0.5$$

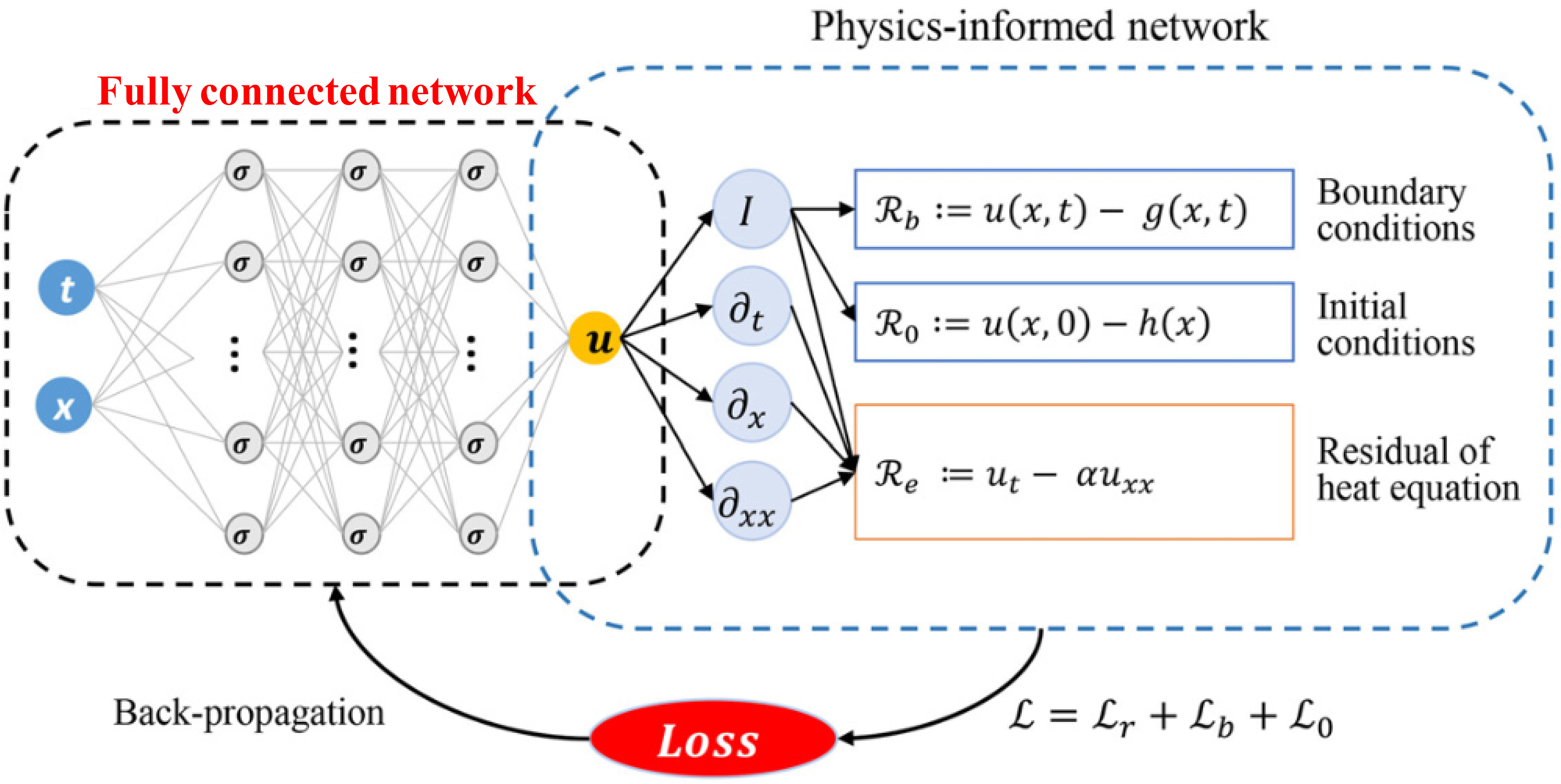
$$x_r = -s\cos(2\pi y_r) + 0.5$$





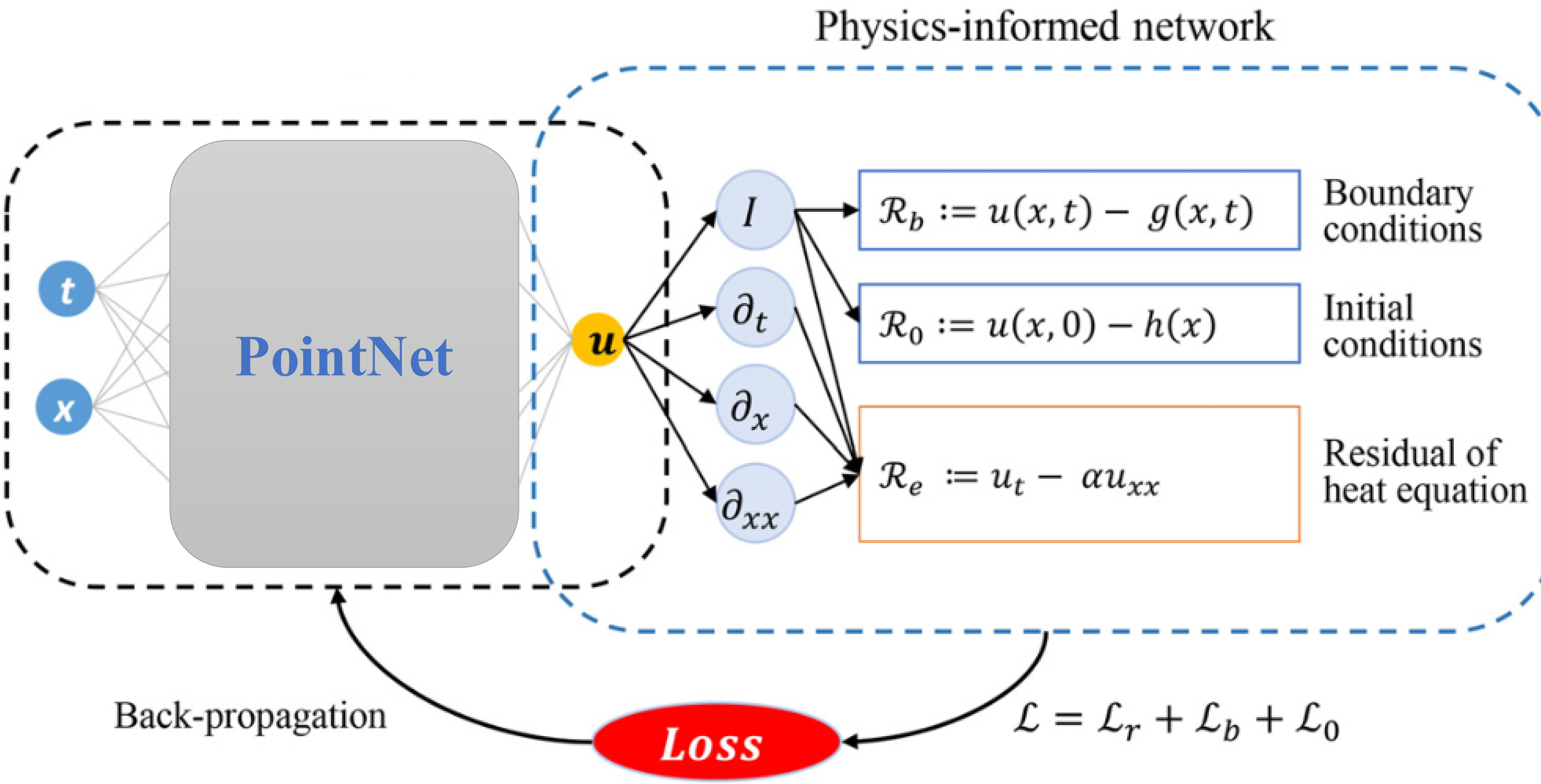
Our proposed solution:

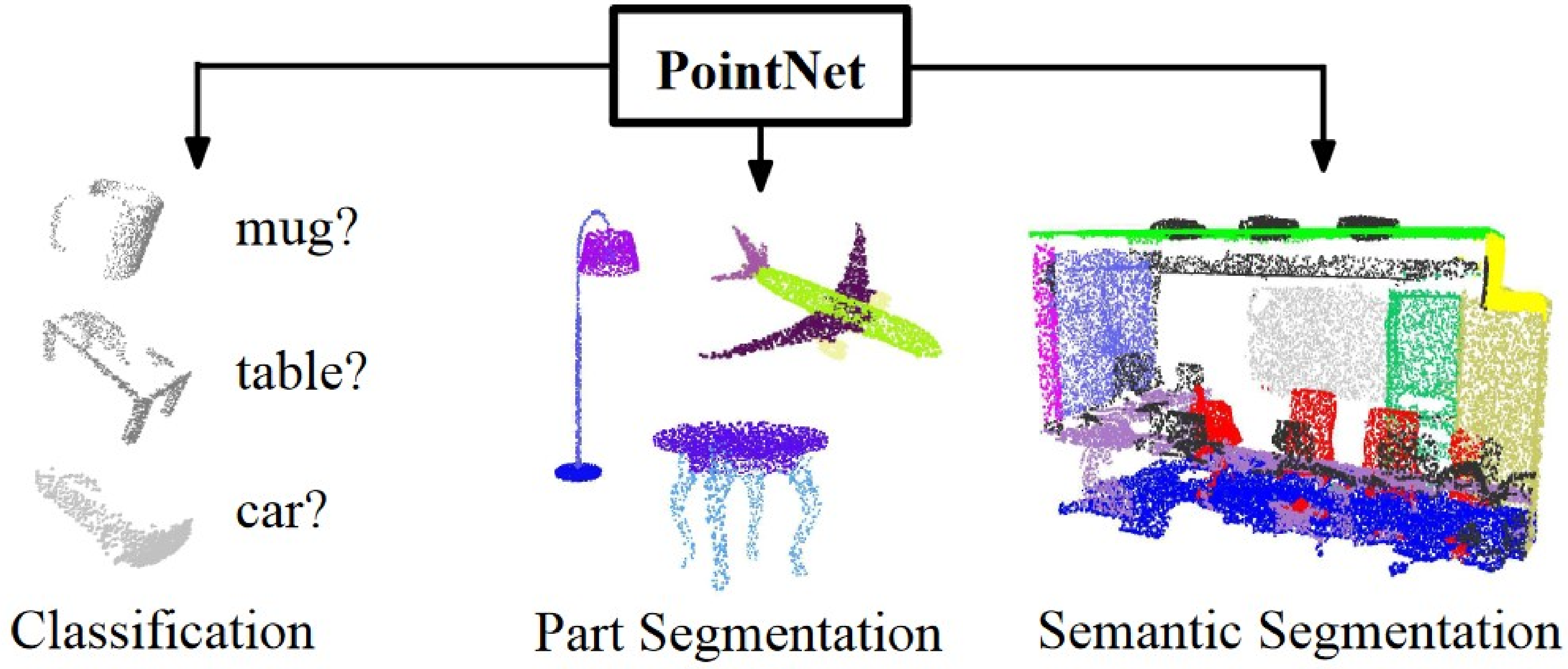
Use PointNet instead of simple fully connected networks


Physics-informed network
Fully connected network
$t$
$x$
$\sigma$
$u$
$I$
$\partial_t$
$\partial_x$
$\partial_{xx}$
$\mathcal{R}_b := u(x,t) - g(x,t)$
Boundary conditions
$\mathcal{R}_0 := u(x,0) - h(x)$
Initial conditions
$\mathcal{R}_e := u_t - \alpha u_{xx}$
Residual of heat equation
Back-propagation
Loss
$\mathcal{L} = \mathcal{L}_r + \mathcal{L}_b + \mathcal{L}_0$

Our proposed solution:

Use PointNet instead of simple fully connected networks


Physics-informed network
$t$
$x$
PointNet
$u$
$I$
$\partial_t$
$\partial_x$
$\partial_{xx}$
$\mathcal{R}_b := u(x,t) - g(x,t)$
Boundary conditions
$\mathcal{R}_0 := u(x,0) - h(x)$
Initial conditions
$\mathcal{R}_e := u_t - \alpha u_{xx}$
Residual of heat equation
Back-propagation
Loss
$\mathcal{L} = \mathcal{L}_r + \mathcal{L}_b + \mathcal{L}_0$



PointNet

mug?

table?

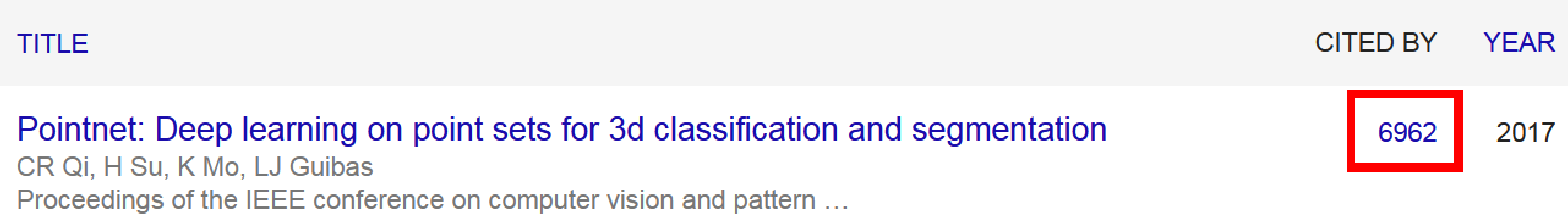
car?

Classification

Part Segmentation

Semantic Segmentation

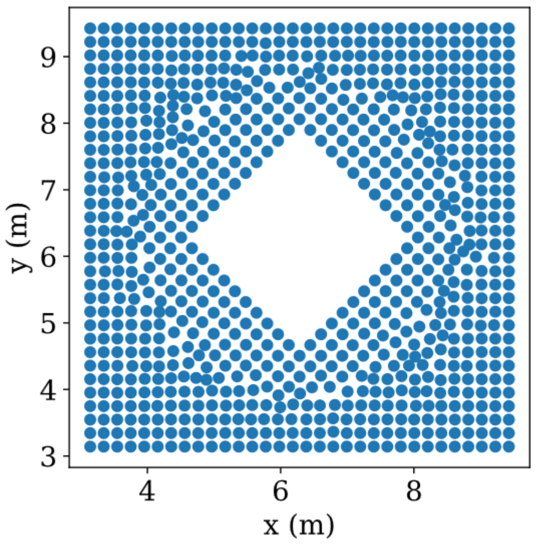
Geometric Computing Lab

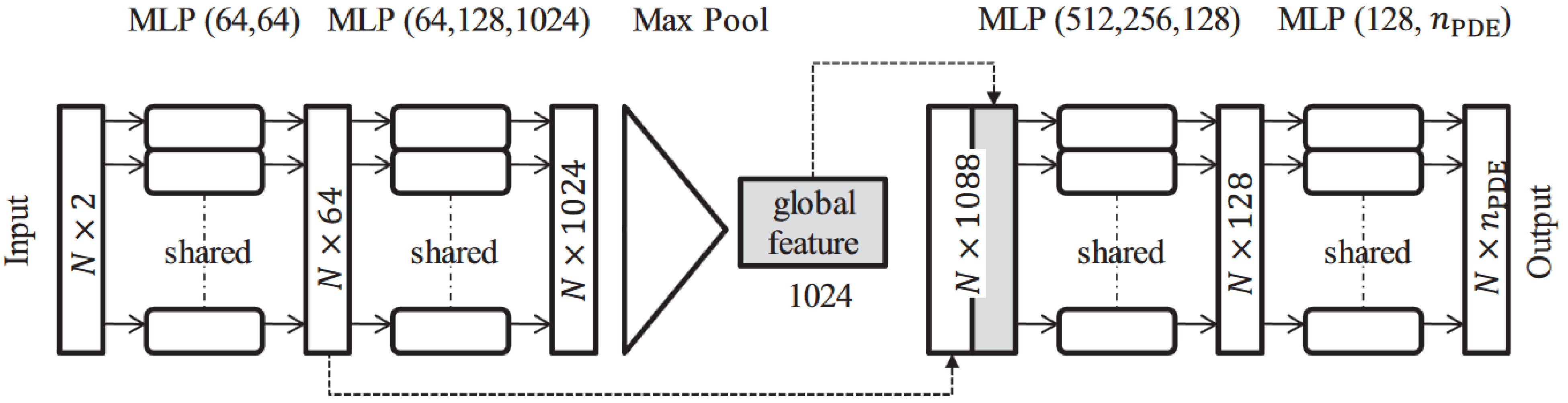


| TITLE | CITED BY | YEAR |
|---|---|---|
| Pointnet: Deep learning on point sets for 3d classification and segmentation<br>CR Qi, H Su, K Mo, LJ Guibas<br>Proceedings of the IEEE conference on computer vision and pattern … | 6962 | 2017 |

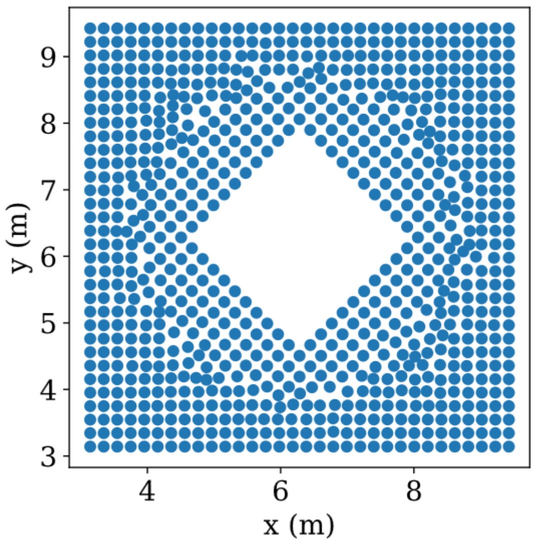
# Physics-Informed PointNet (PIPN)

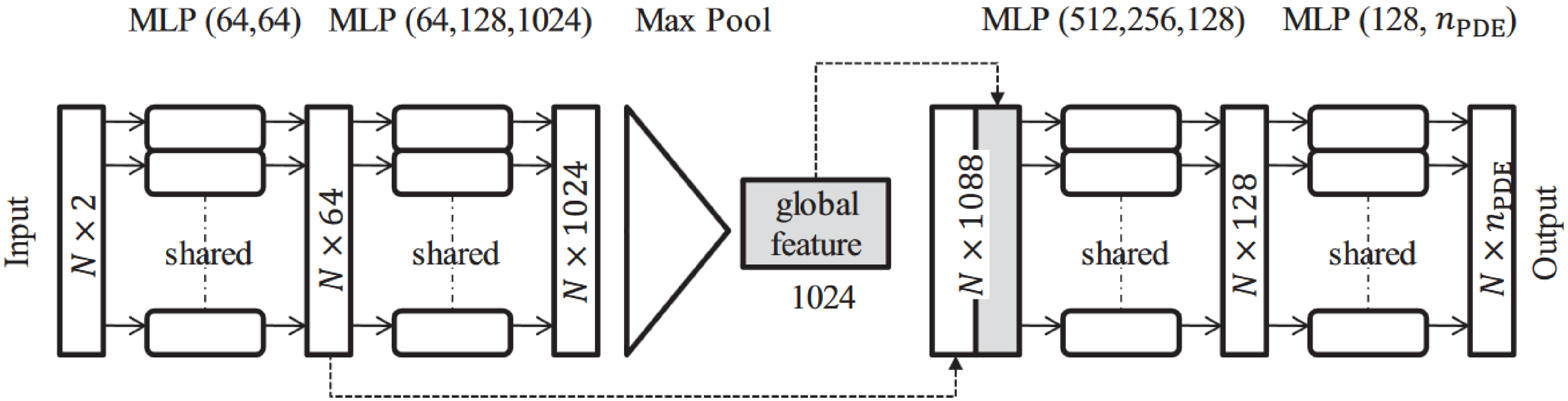
## Methodology





Representing each domain as a set of points $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$

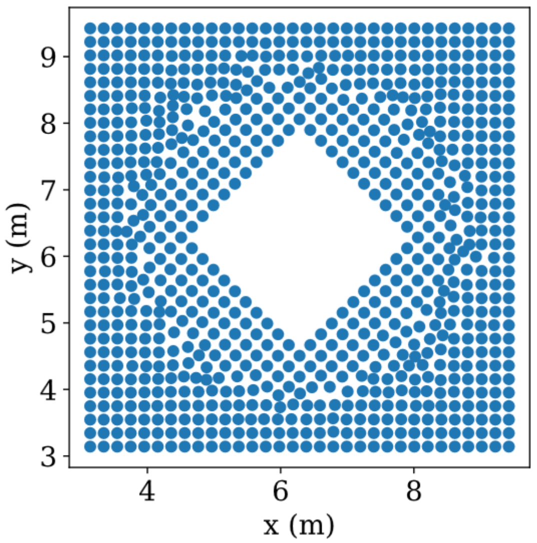


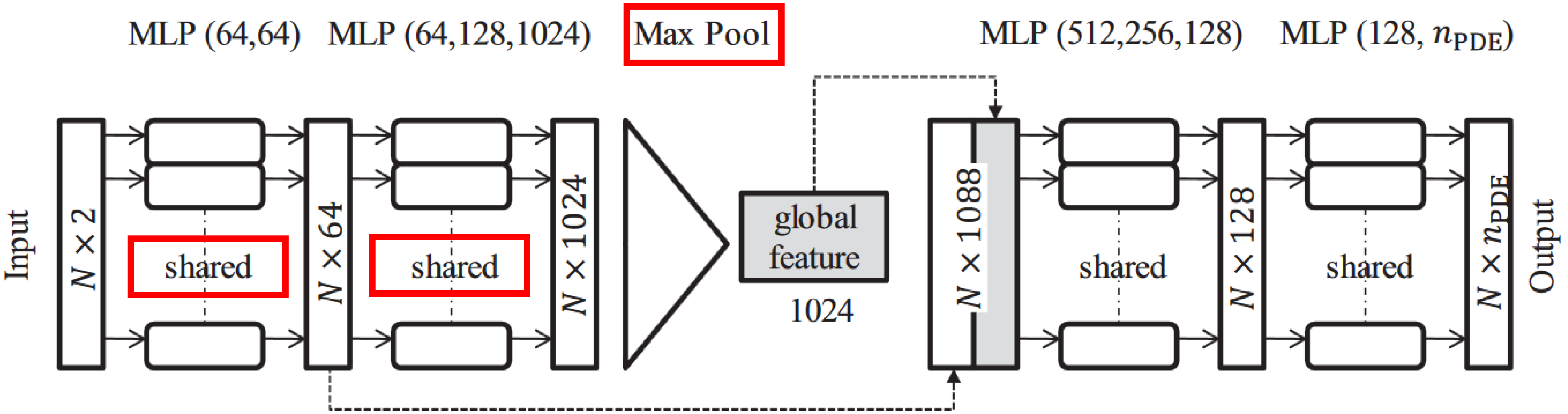


Representing each domain as a set of points $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$

$N$ is number of points in each domain

For example in 2D, we have $\mathbf{x}_1 = (x_1, y_1)$

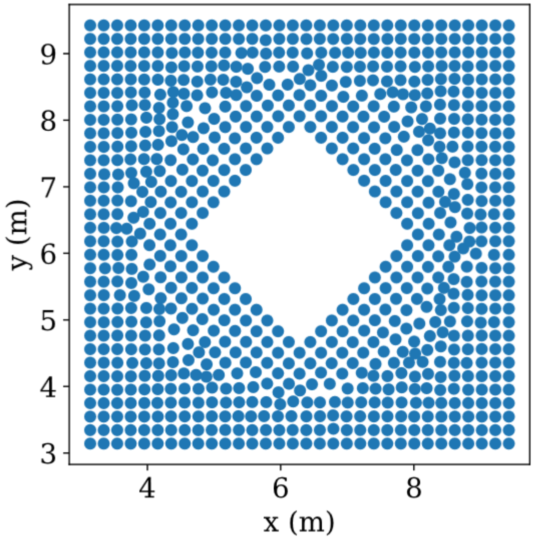


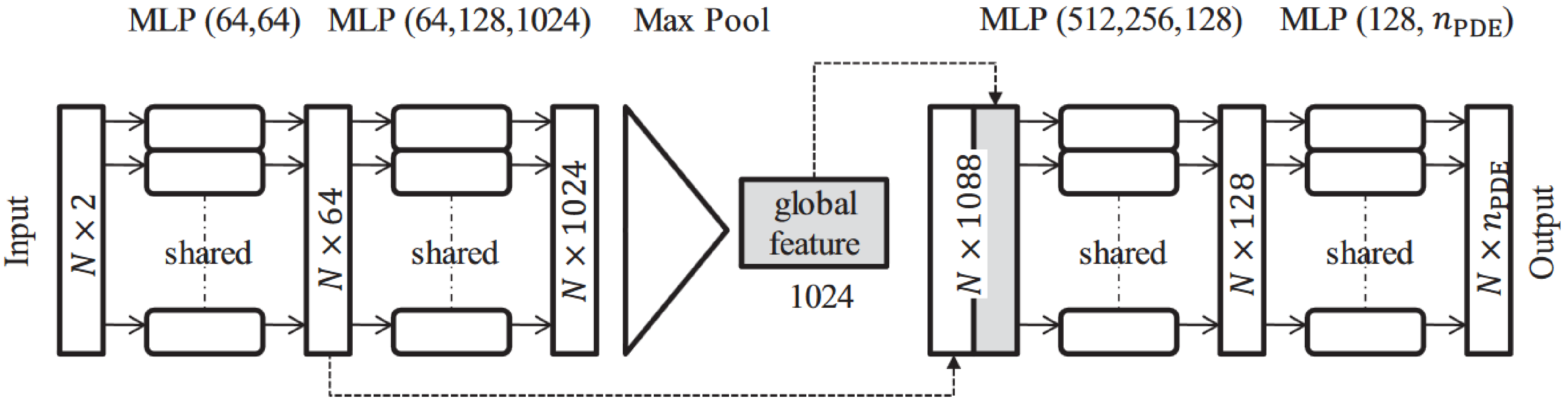


Representing each domain as a set of points $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$

Approximating geometric feature of the domain by $g(X) \approx \max\big(h(\mathbf{x}_1), h(\mathbf{x}_2), \cdots, h(\mathbf{x}_N)\big)$

global feature Max Pool shared MLP





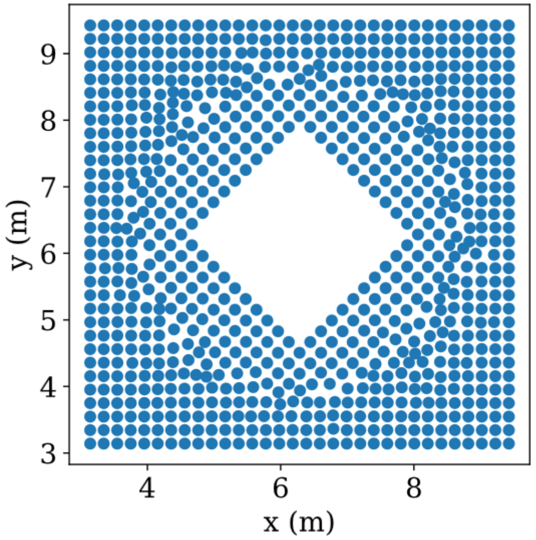
Representing each domain as a set of points $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$

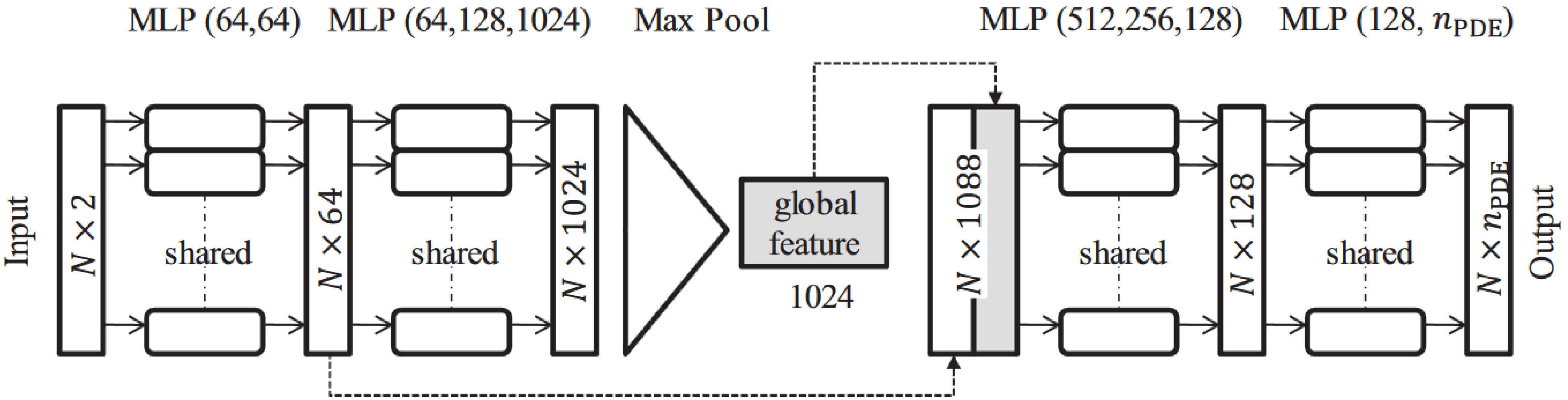
Approximating geometric feature of the domain by $g(X) \approx \max\big(h(\mathbf{x}_1), h(\mathbf{x}_2), \cdots, h(\mathbf{x}_N)\big)$

$$u_i = f\big(\mathbf{x}_i, g(X)\big)$$

PointNet prediction (output)

The PointNet output at each point depends on both the spatial coordinates and the geometric feature of the whole domain.





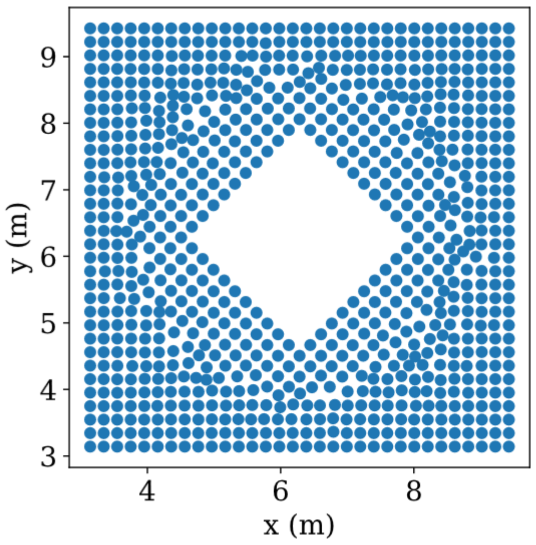
Representing each domain as a set of points $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$

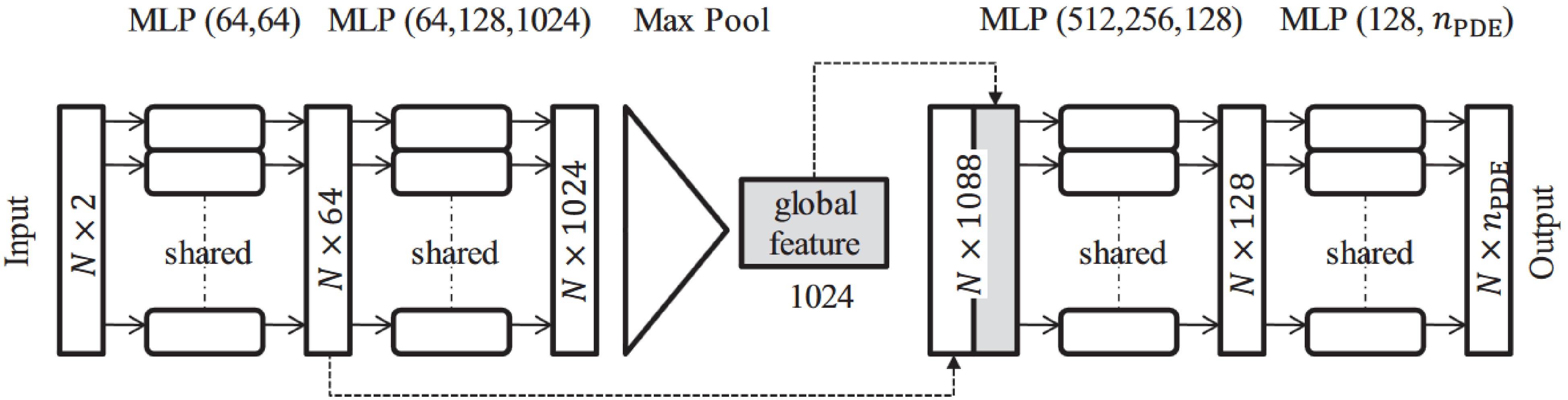
Approximating geometric feature of the domain by $g(X) \approx \max\big(h(\mathbf{x}_1), h(\mathbf{x}_2), \cdots, h(\mathbf{x}_N)\big)$

$$u_i = f\big(\mathbf{x}_i, g(X)\big)$$

$$\frac{\delta u_i}{\delta x_i} = \frac{\delta}{\delta x_i}\Big(f\big(\mathbf{x}_i, g(X)\big)\Big)$$

$\delta$ is the automatic differentiation





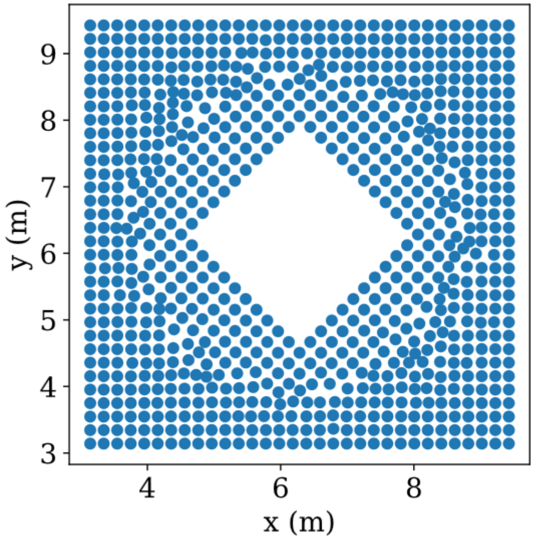
Representing each domain as a set of points $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$

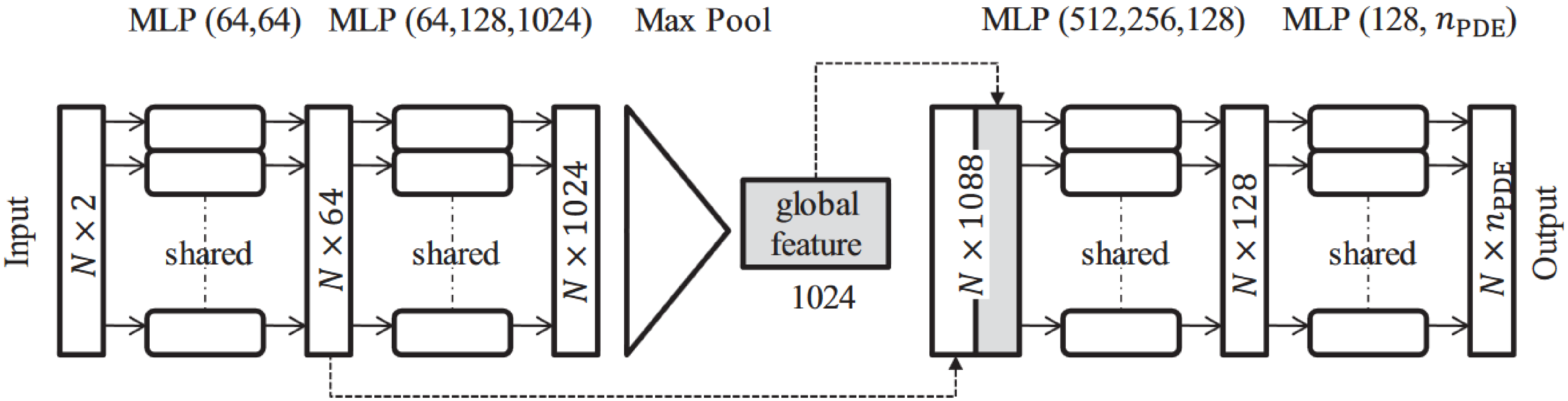
Approximating geometric feature of the domain by $g(X) \approx \max\big(h(\mathbf{x}_1), h(\mathbf{x}_2), \cdots, h(\mathbf{x}_N)\big)$

$$u_i = f\big(\mathbf{x}_i, g(X)\big)$$

$$\frac{\delta u_i}{\delta x_i} = \frac{\delta}{\delta x_i}\Big(f\big(\mathbf{x}_i, g(X)\big)\Big)$$

$$r^{\text{continuity}} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{\delta u_i}{\delta x_i} + \frac{\delta v_i}{\delta y_i}\right)^2$$





Representing each domain as a set of points $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$

Approximating geometric feature of the domain by $g(X) \approx \max\big(h(\mathbf{x}_1), h(\mathbf{x}_2), \cdots, h(\mathbf{x}_N)\big)$

$$u_i = f\big(\mathbf{x}_i, g(X)\big)$$

$$\frac{\delta u_i}{\delta x_i} = \frac{\delta}{\delta x_i}\Big(f\big(\mathbf{x}_i, g(X)\big)\Big)$$

Thus, the weights of PIPN is updated based on all the domains at each epochs during training.

$$r^{\text{continuty}} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{\delta u_i}{\delta x_i} + \frac{\delta v_i}{\delta y_i}\right)^2 \longrightarrow Loss = \frac{1}{m}\sum_{j=1}^{m} r_j^{\text{continuty}}$$

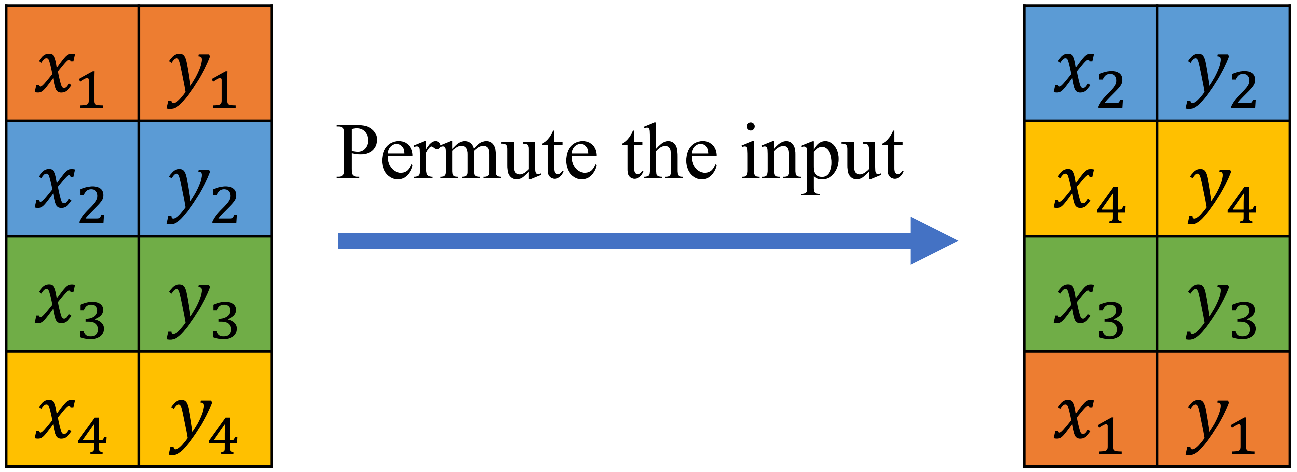
where $m$ is the number of domains

# Important features of PointNet (and PIPN)

Let’s imagine we have a 2D domain representing by

| | |
|---|---|
| $x_1$ | $y_1$ |
| $x_2$ | $y_2$ |
| $x_3$ | $y_3$ |
| $x_4$ | $y_4$ |

# Important features of PointNet (and PIPN)



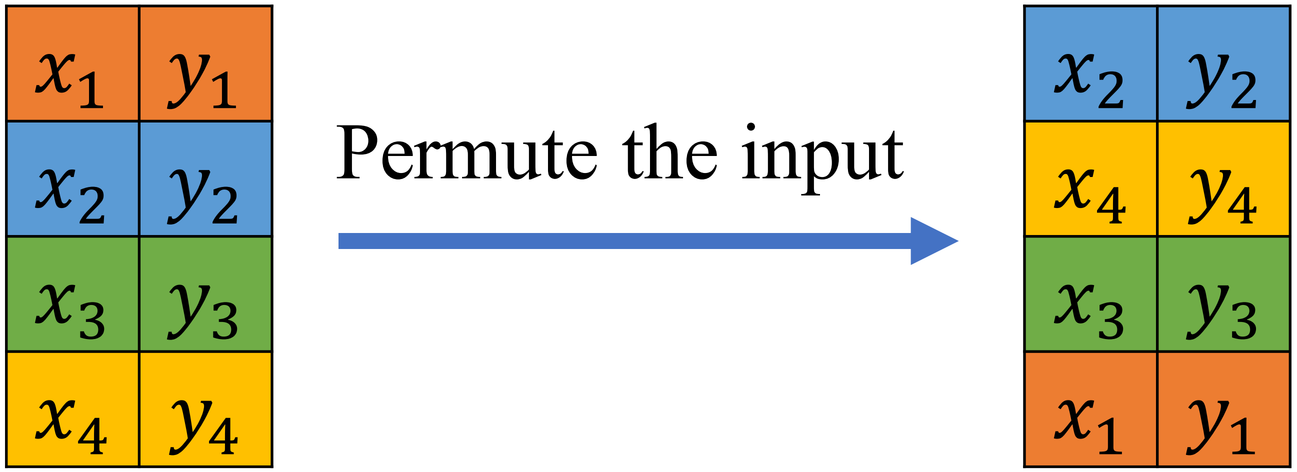
|  |  |
|---|---|
| $x_1$ | $y_1$ |
| $x_2$ | $y_2$ |
| $x_3$ | $y_3$ |
| $x_4$ | $y_4$ |

Permute the input

|  |  |
|---|---|
| $x_2$ | $y_2$ |
| $x_4$ | $y_4$ |
| $x_3$ | $y_3$ |
| $x_1$ | $y_1$ |

The network output should not change!

# Important features of PointNet (and PIPN)



| | |
|---|---|
| $x_1$ | $y_1$ |
| $x_2$ | $y_2$ |
| $x_3$ | $y_3$ |
| $x_4$ | $y_4$ |

Permute the input

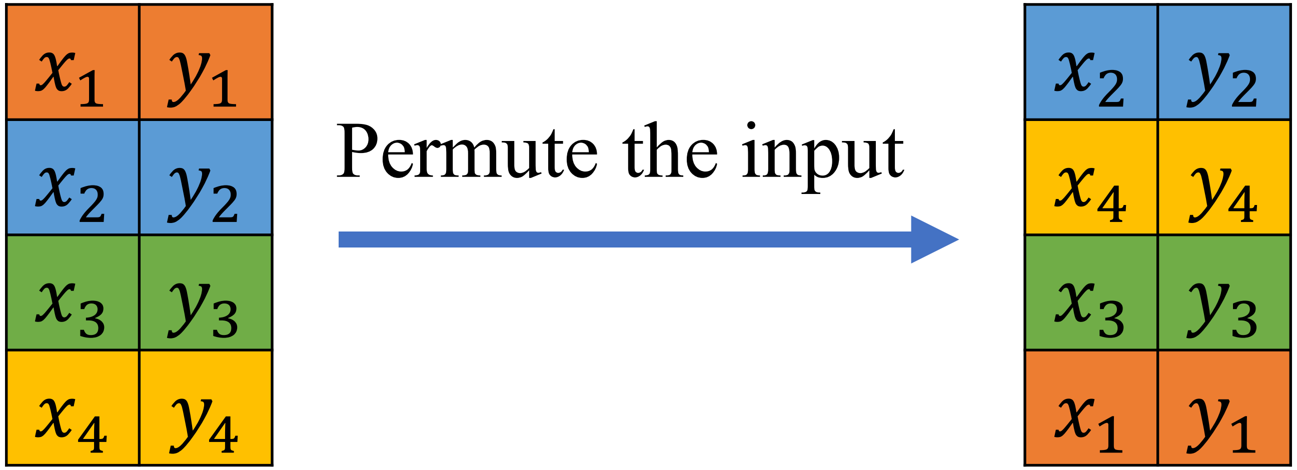
| | |
|---|---|
| $x_2$ | $y_2$ |
| $x_4$ | $y_4$ |
| $x_3$ | $y_3$ |
| $x_1$ | $y_1$ |

The network output should not change!

PointNet (and consequently, PIPN) is invariant to $N!$ permutations using two features:

1. Shared MLPs
2. A symmetric function

# Important features of PointNet (and PIPN)



|  |  |
|---|---|
| $x_1$ | $y_1$ |
| $x_2$ | $y_2$ |
| $x_3$ | $y_3$ |
| $x_4$ | $y_4$ |

Permute the input

|  |  |
|---|---|
| $x_2$ | $y_2$ |
| $x_4$ | $y_4$ |
| $x_3$ | $y_3$ |
| $x_1$ | $y_1$ |

The network output should not change!

PointNet (and consequently, PIPN) is invariant to $N!$ permutations using two features:

1. Shared MLPs
2. A symmetric function

$$g(X) \approx \underset{\substack{\uparrow \\ \text{symmetric} \\ \text{function}}}{\max}\Big(h(\mathbf{x}_1), h(\mathbf{x}_2), \cdots, \underset{\substack{\uparrow \\ \text{shared MLP}}}{h}(\mathbf{x}_N)\Big)$$

# Important features of PointNet (and PIPN)

**Shared MLPs** are "not" **dense layers** (TensorFlow terminology)!

Implementation of **shared MLPs** is explained in

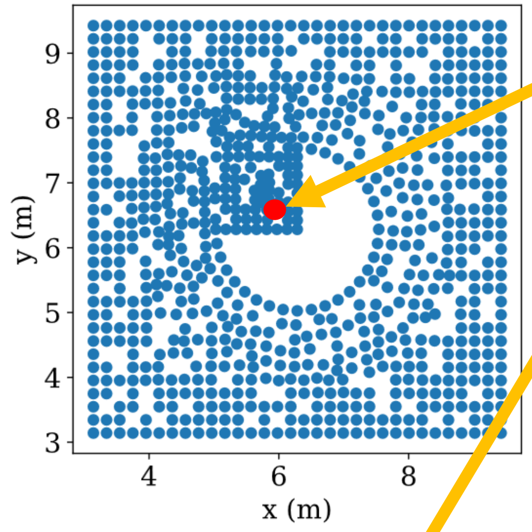
- ➢ the PointNet source code (https://github.com/charlesq34/pointnet)
- ➢ and the manuscript "Network in Network" (cited 6563), (https://arxiv.org/abs/1312.4400)

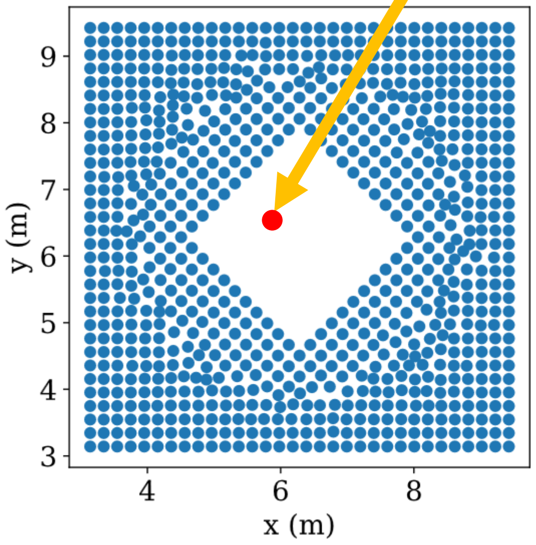
2. A symmetric function

symmetric function

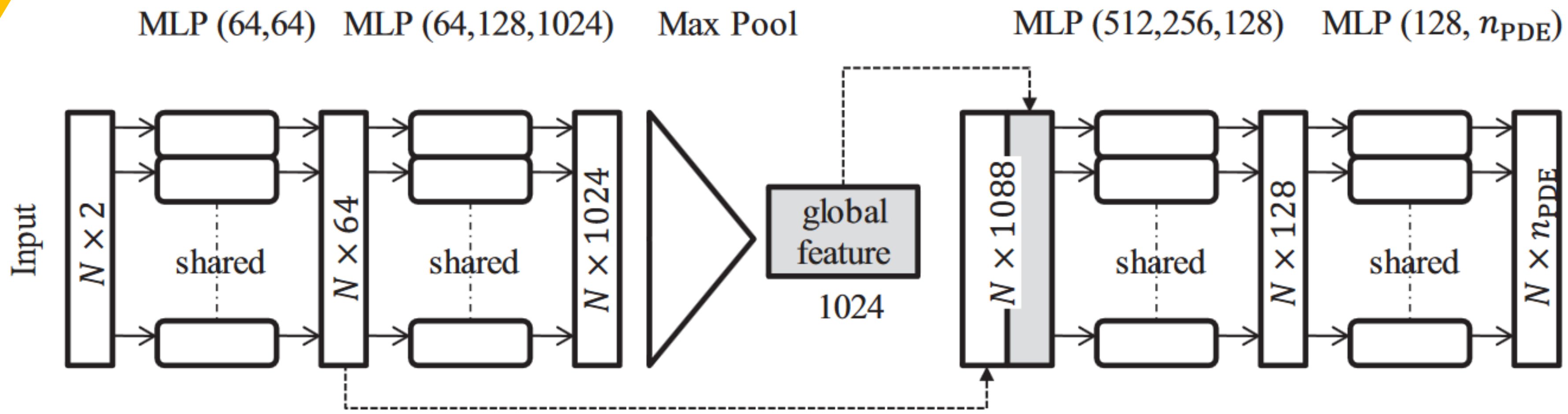
shared MLP

Uncommon points in these two domain

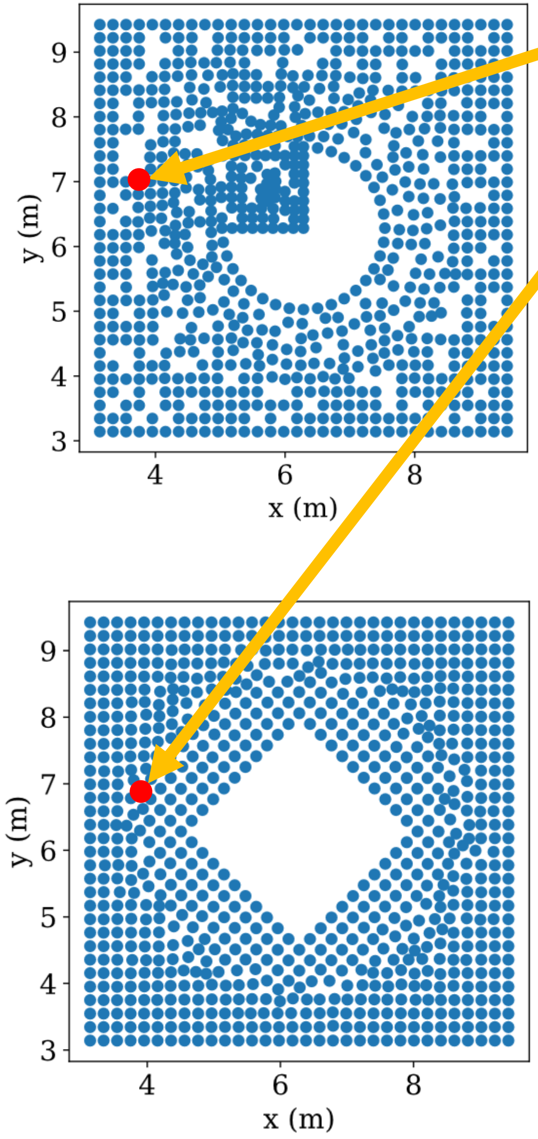




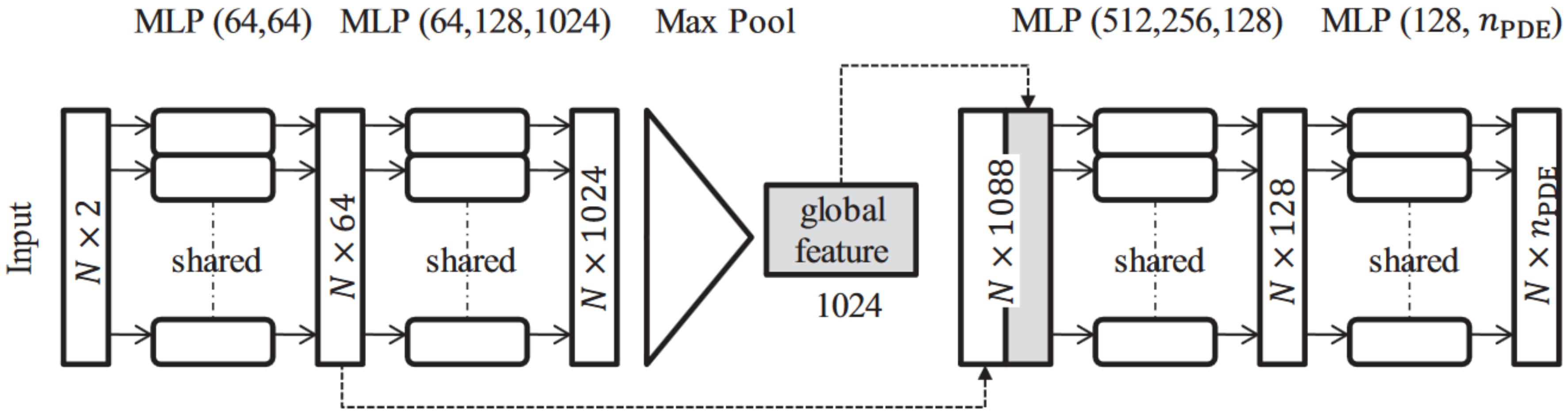
It is not an issue anymore!



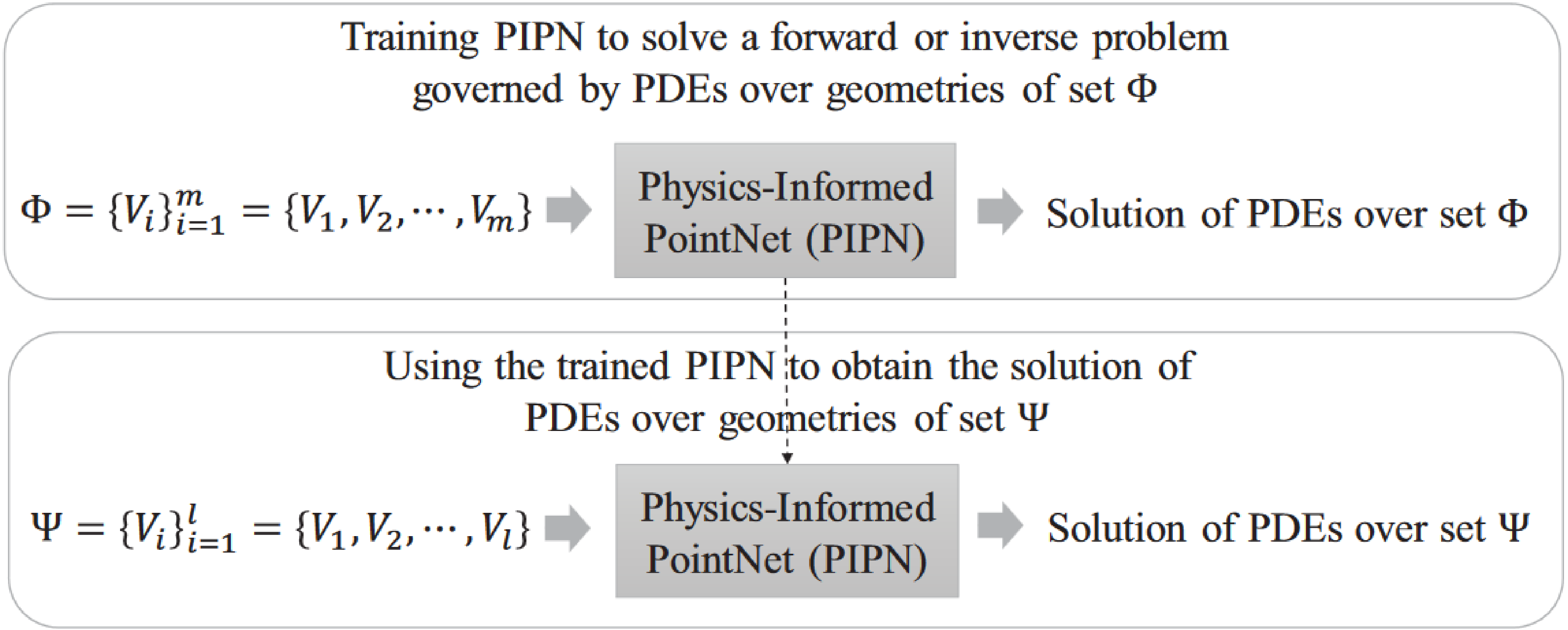
Even in common points, the solution might be different in each of these two domains



It is not an issue anymore!



# PIPN framework



Training PIPN to solve a forward or inverse problem governed by PDEs over geometries of set $\Phi$

$\Phi = \{V_i\}_{i=1}^{m} = \{V_1, V_2, \cdots, V_m\}$

Physics-Informed PointNet (PIPN)

Solution of PDEs over set $\Phi$

Using the trained PIPN to obtain the solution of PDEs over geometries of set $\Psi$

$\Psi = \{V_i\}_{i=1}^{l} = \{V_1, V_2, \cdots, V_l\}$

Physics-Informed PointNet (PIPN)

Solution of PDEs over set $\Psi$

Set $\Psi = \{V_i\}_{i=1}^{l}$ contains unseen geometries from seen and unseen categories with reference to the set $\Phi = \{V_i\}_{i=1}^{m}$.

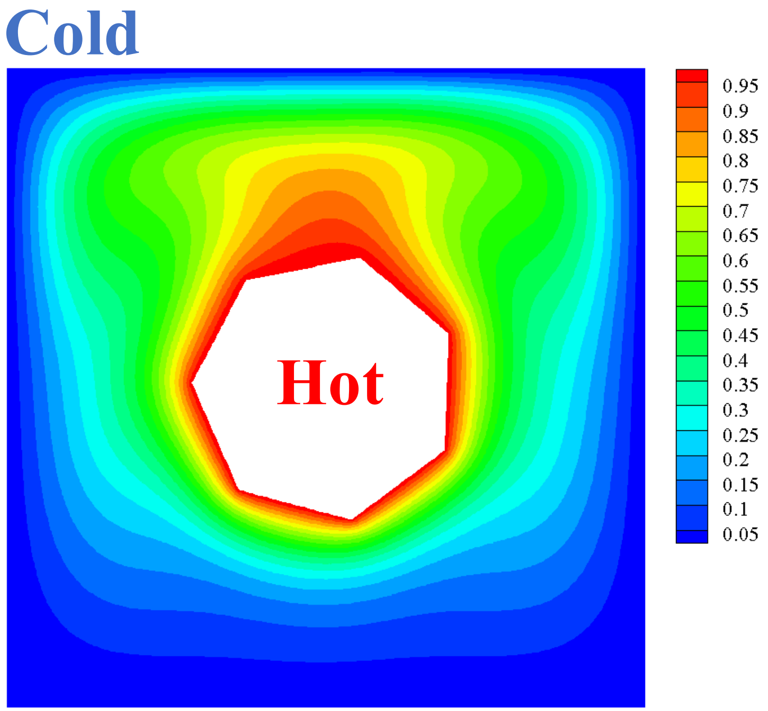
# Physics-Informed PointNet (PIPN)

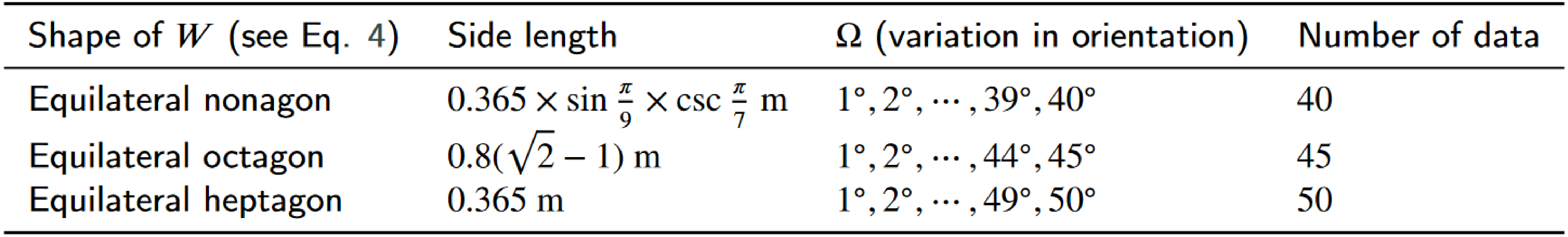
## Results & Discussion

# Natural convection in a square enclosure with a cylinder

**Inner cylinders have different shapes and different poses!**

**We do not know the temperature distribution on the surface of inner cylinder!**

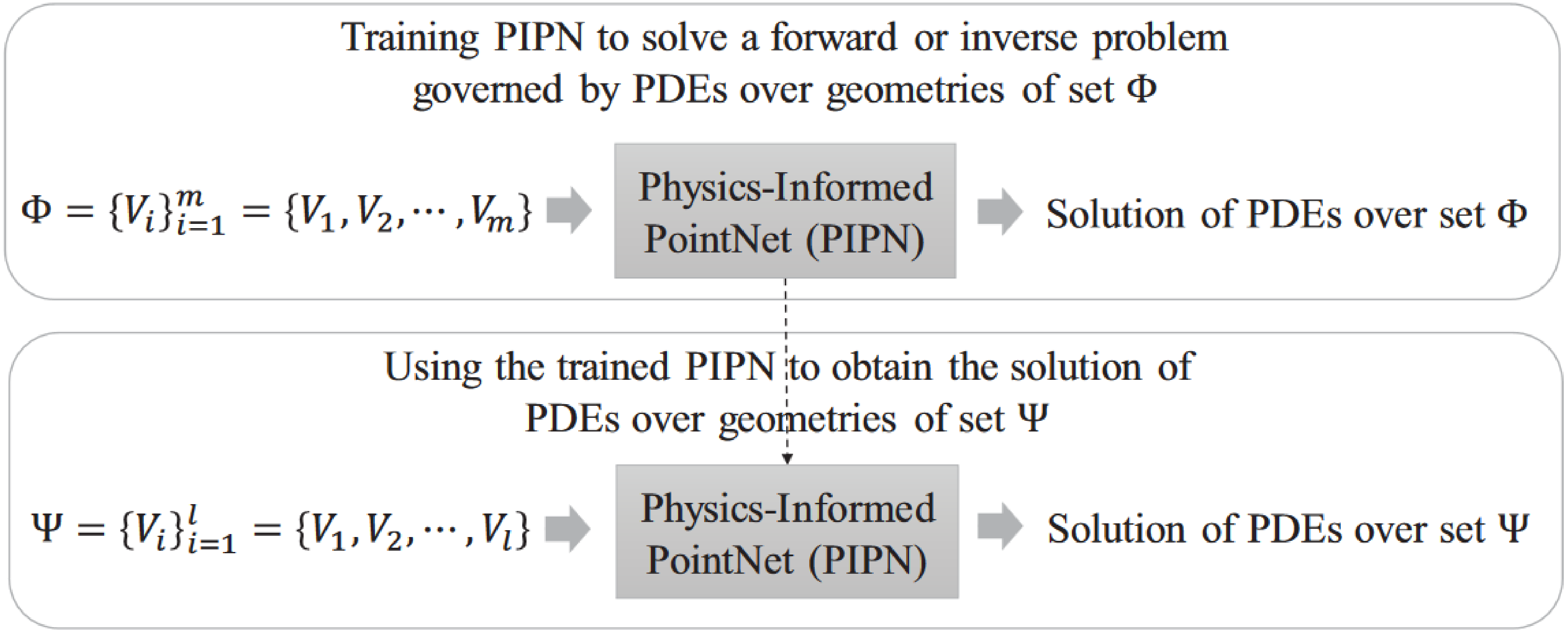




| Shape of $W$ (see Eq. 4) | Side length | $\Omega$ (variation in orientation) | Number of data |
|---|---|---|---|
| Equilateral nonagon | $0.365 \times \sin\frac{\pi}{9} \times \csc\frac{\pi}{7}$ m | $1°, 2°, \cdots, 39°, 40°$ | 40 |
| Equilateral octagon | $0.8(\sqrt{2} - 1)$ m | $1°, 2°, \cdots, 44°, 45°$ | 45 |
| Equilateral heptagon | 0.365 m | $1°, 2°, \cdots, 49°, 50°$ | 50 |

# Natural convection in a square enclosure with a cylinder

- We would like to obtain the velocity, temperature, and pressure fields for 135 domains with different geometries for the inner cylinder.
- For 108 domains, we have sparse observations in sensor locations (set Φ)
- For 27 dominos, we have no observations (set Ψ)



Training PIPN to solve a forward or inverse problem governed by PDEs over geometries of set Φ

$\Phi = \{V_i\}_{i=1}^{m} = \{V_1, V_2, \cdots, V_m\}$

Physics-Informed PointNet (PIPN)

Solution of PDEs over set Φ

Using the trained PIPN to obtain the solution of PDEs over geometries of set Ψ

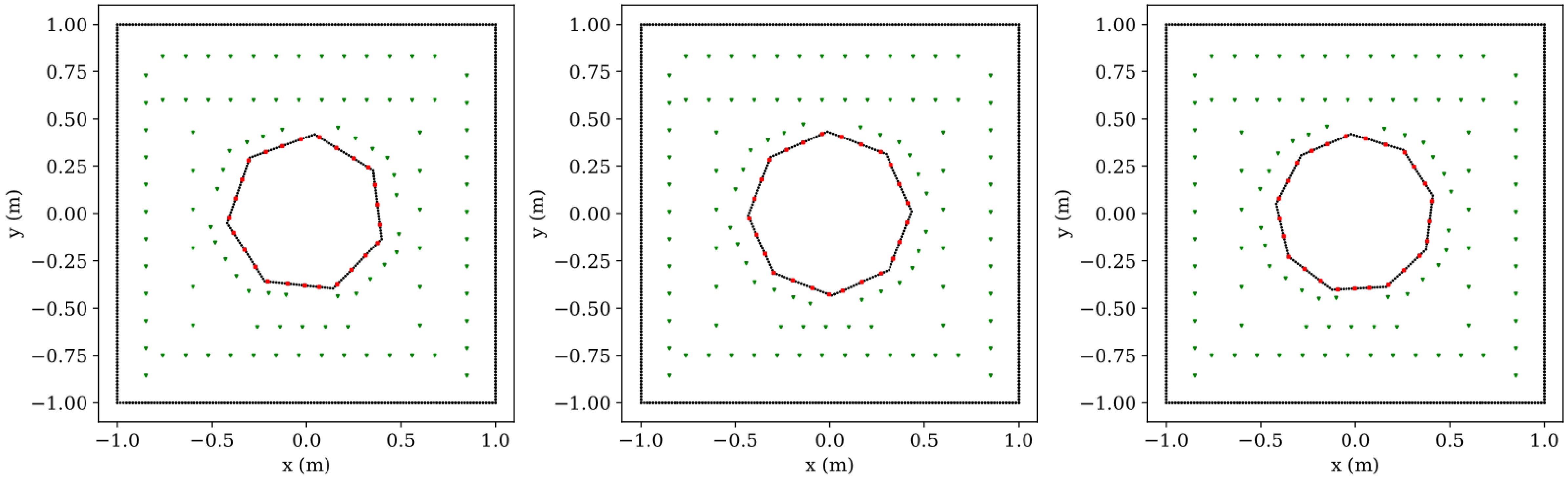
$\Psi = \{V_i\}_{i=1}^{l} = \{V_1, V_2, \cdots, V_l\}$

Physics-Informed PointNet (PIPN)

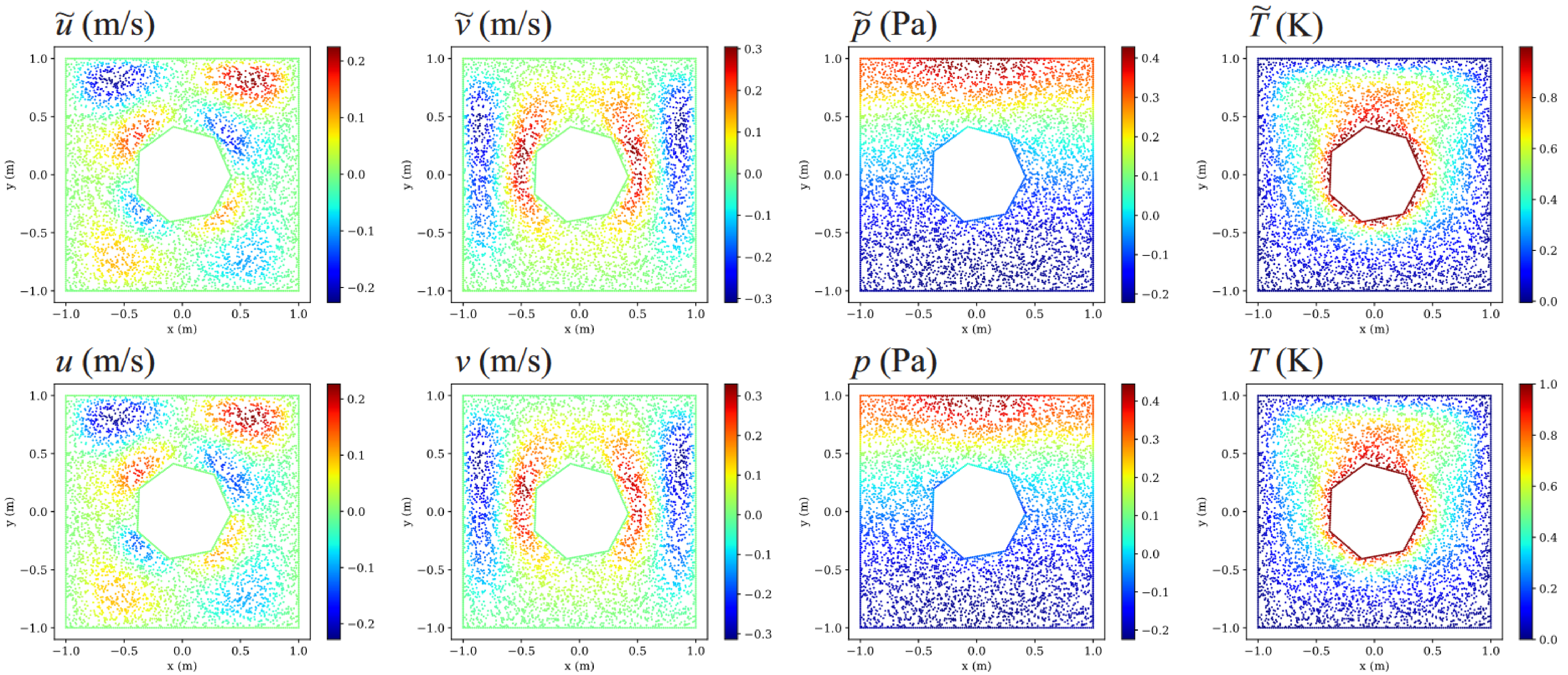
Solution of PDEs over set Ψ

**Loss function for the set $\Phi$ = conservation of mass**
**+ conservation of momentum + conservation of energy**
**+ velocity boundary conditions**
**+ temperature boundary condition of the outer cylinder**
**+ sparse observation of velocity/temperature/pressure in sensor locations**

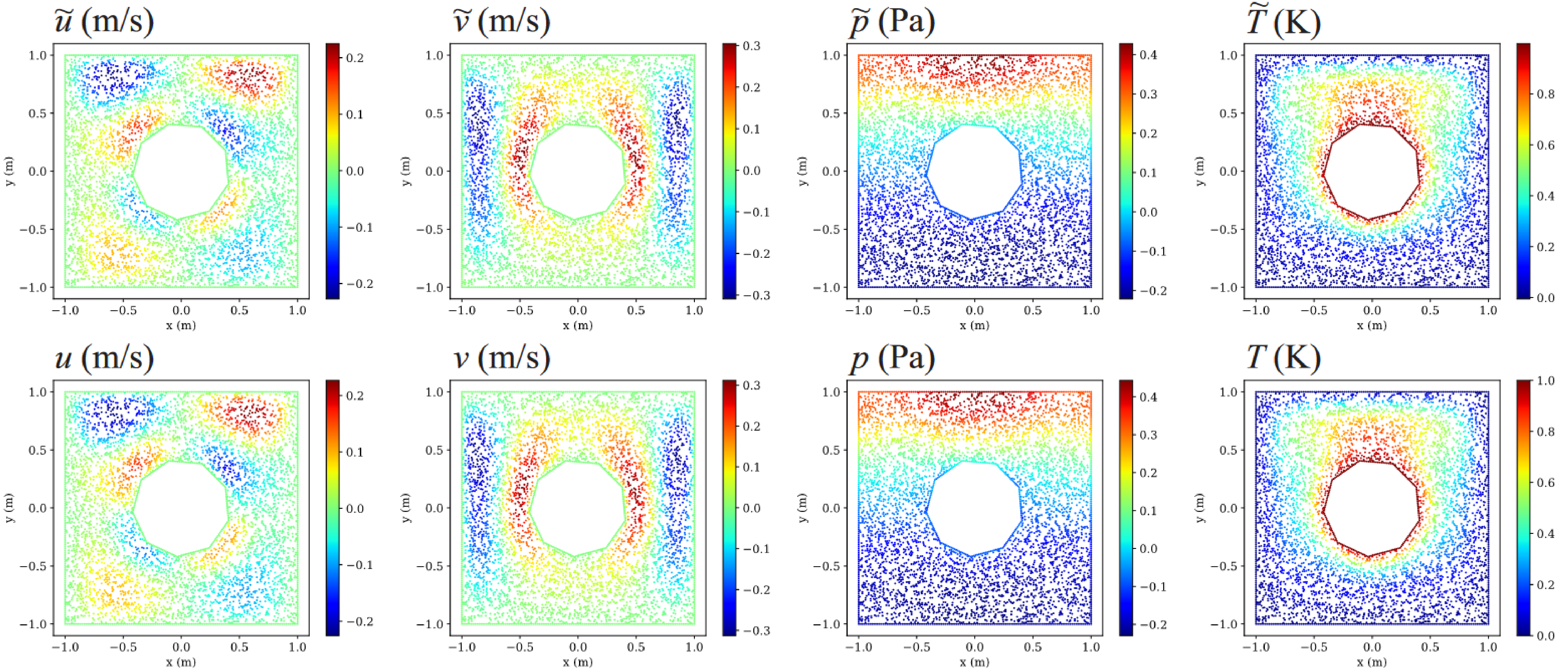
examples of sensor locations in the set $\Phi = \{V_i\}_{i=1}^{108}$



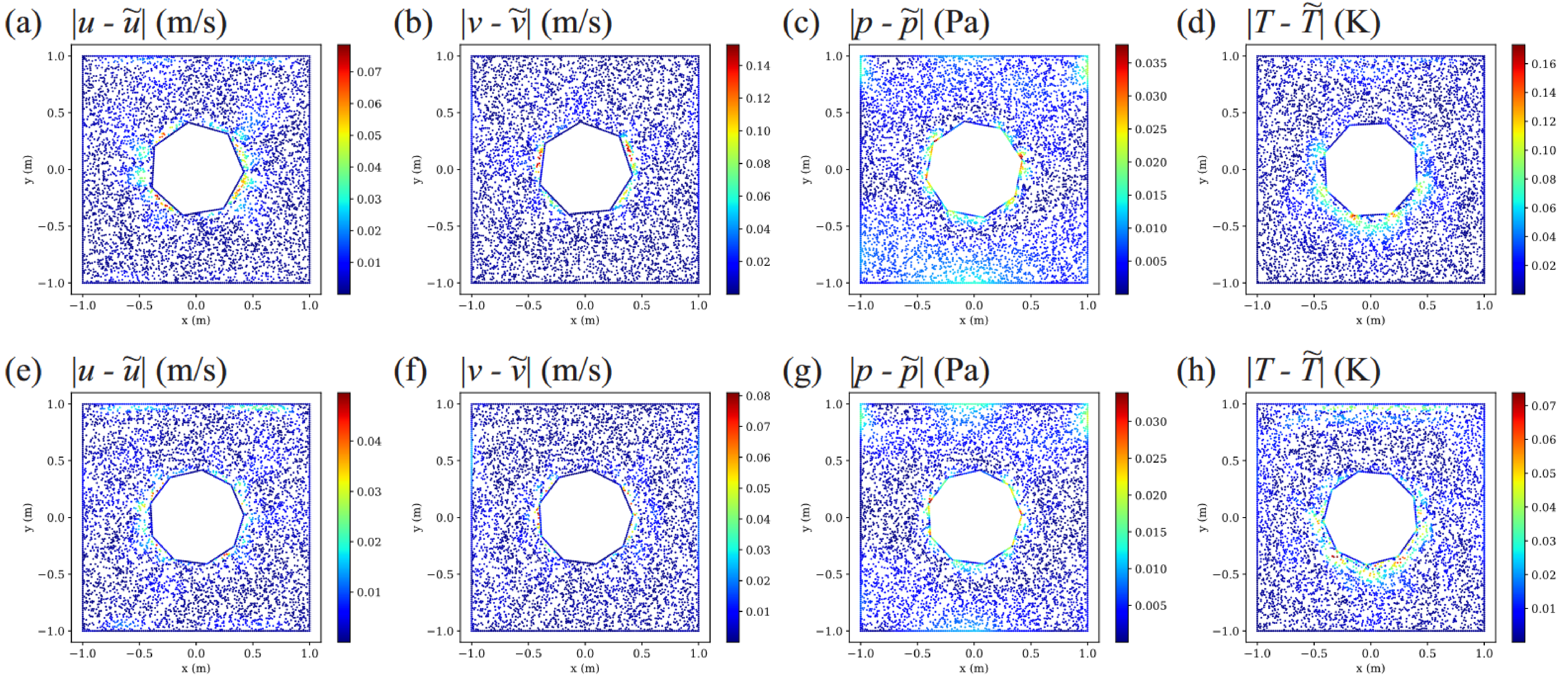
# Prediction by PIPN vs. the ground truth for a domain of the set $\Phi = \{V_i\}_{i=1}^{108}$



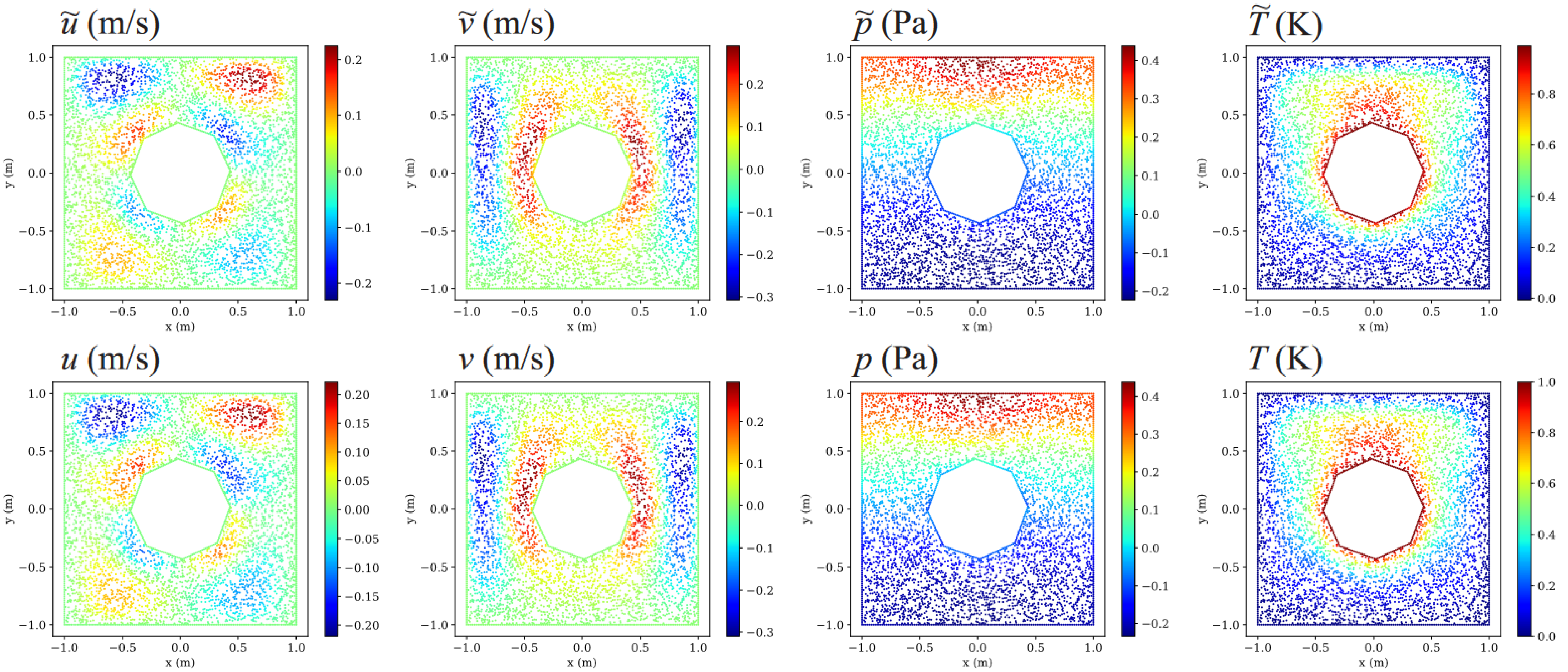
# Prediction by PIPN vs. the ground truth for a domain of the set $\Phi = \{V_i\}_{i=1}^{108}$



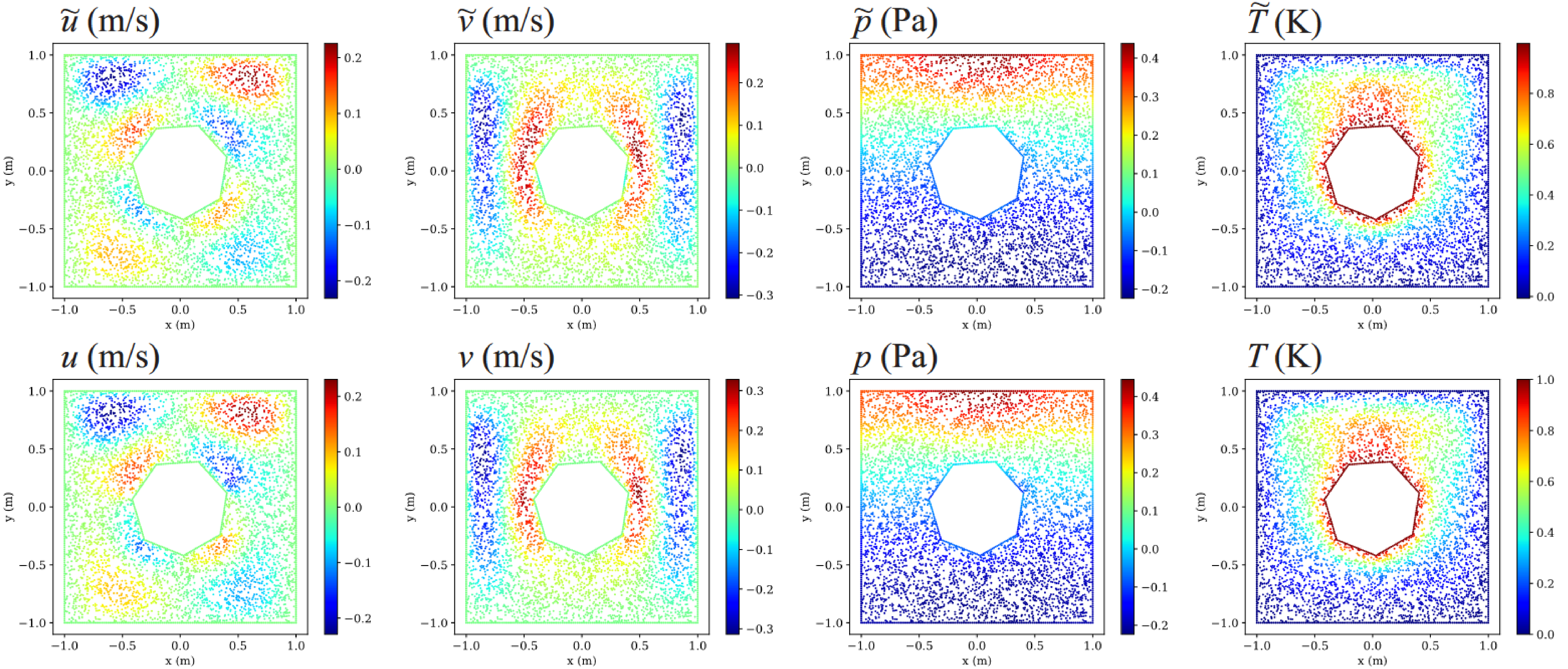
Absolute pointwise error distribution for geometries with maximum and minimum errors for the set $\Phi = \{V_i\}_{i=1}^{108}$



# Prediction by PIPN vs. the ground truth for a domain of the set $\Psi = \{V_i\}_{i=1}^{27}$



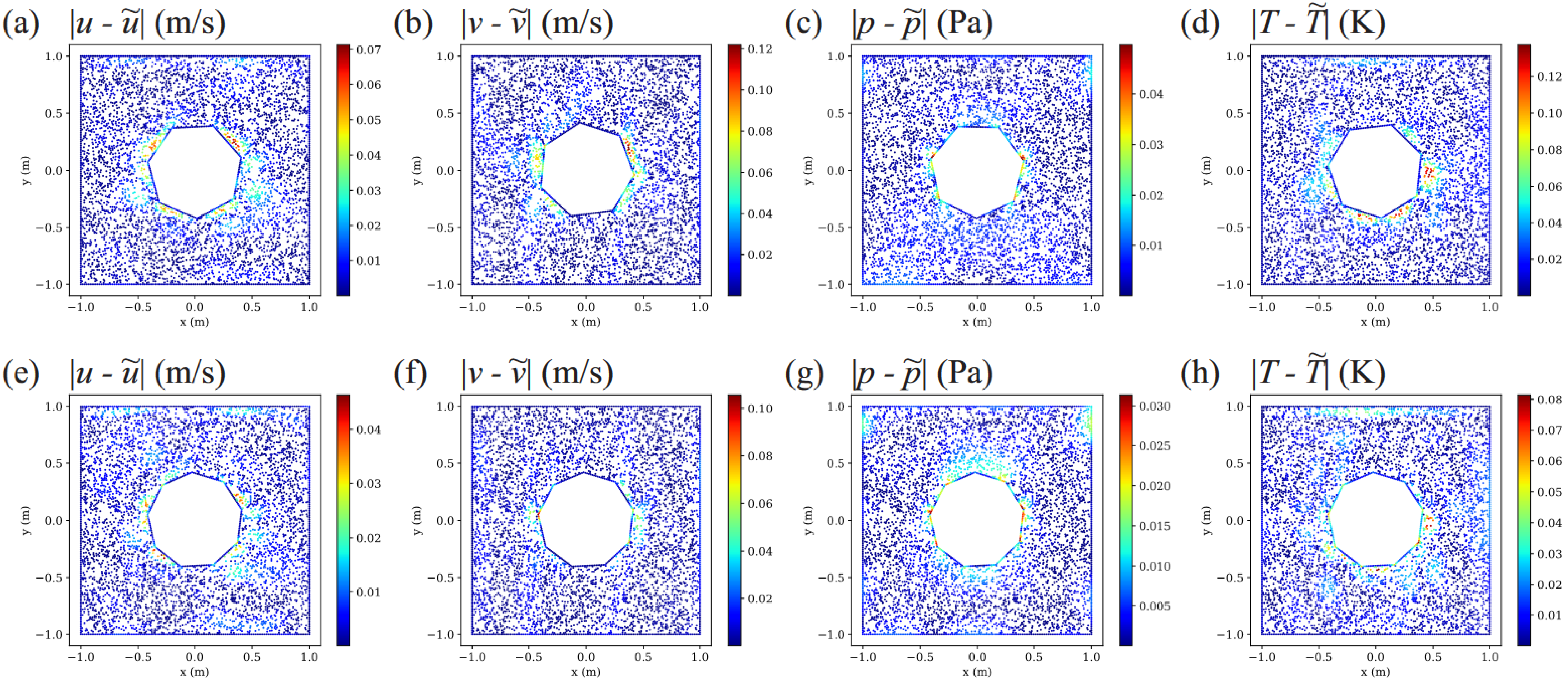
# Prediction by PIPN vs. the ground truth for a domain of the set $\Psi = \{V_i\}_{i=1}^{27}$



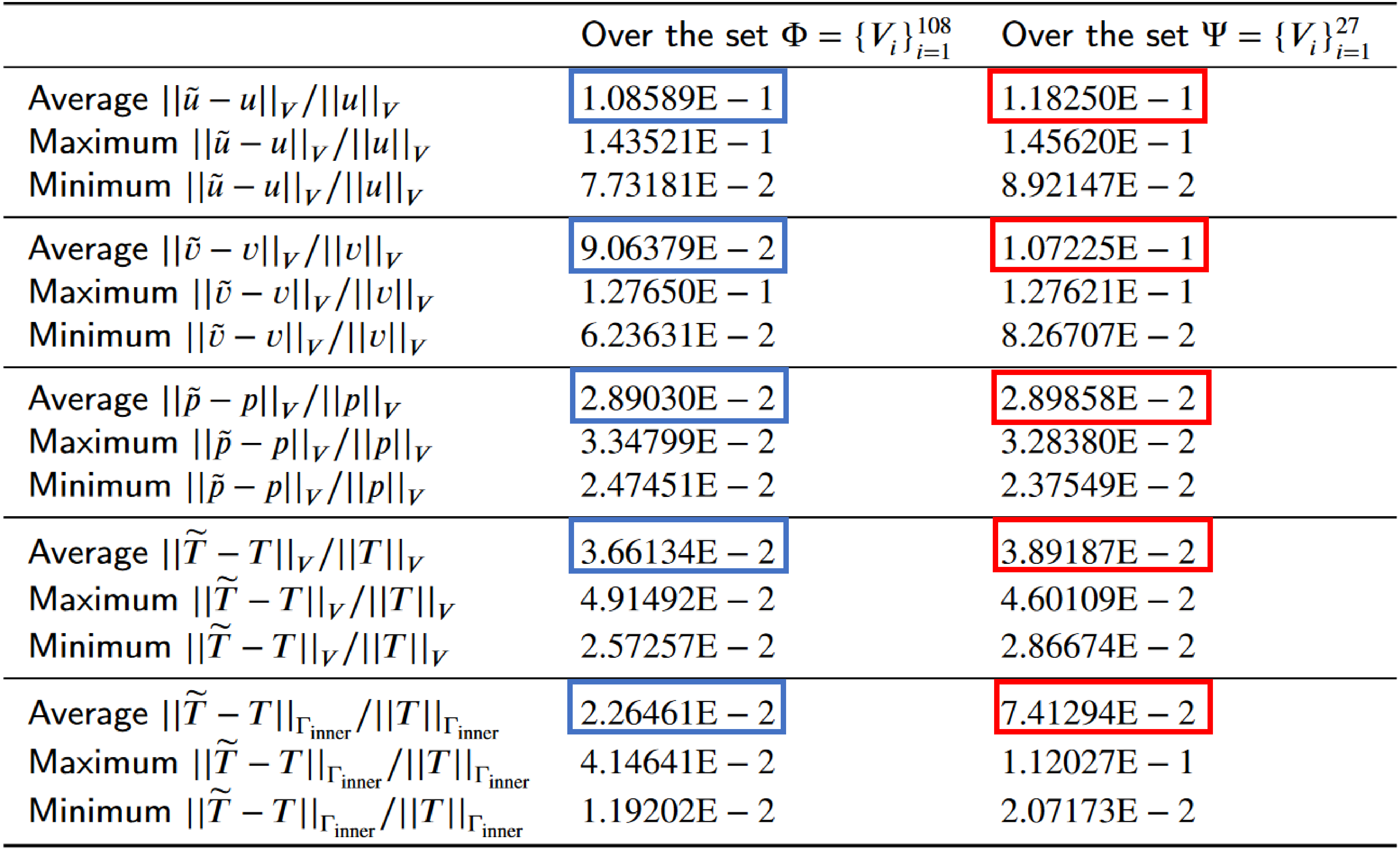
$\tilde{u}$ (m/s) $\tilde{v}$ (m/s) $\tilde{p}$ (Pa) $\tilde{T}$ (K)

$u$ (m/s) $v$ (m/s) $p$ (Pa) $T$ (K)

# Absolute pointwise error distribution for geometries with maximum and minimum errors for the set $\Psi = \{V_i\}_{i=1}^{27}$

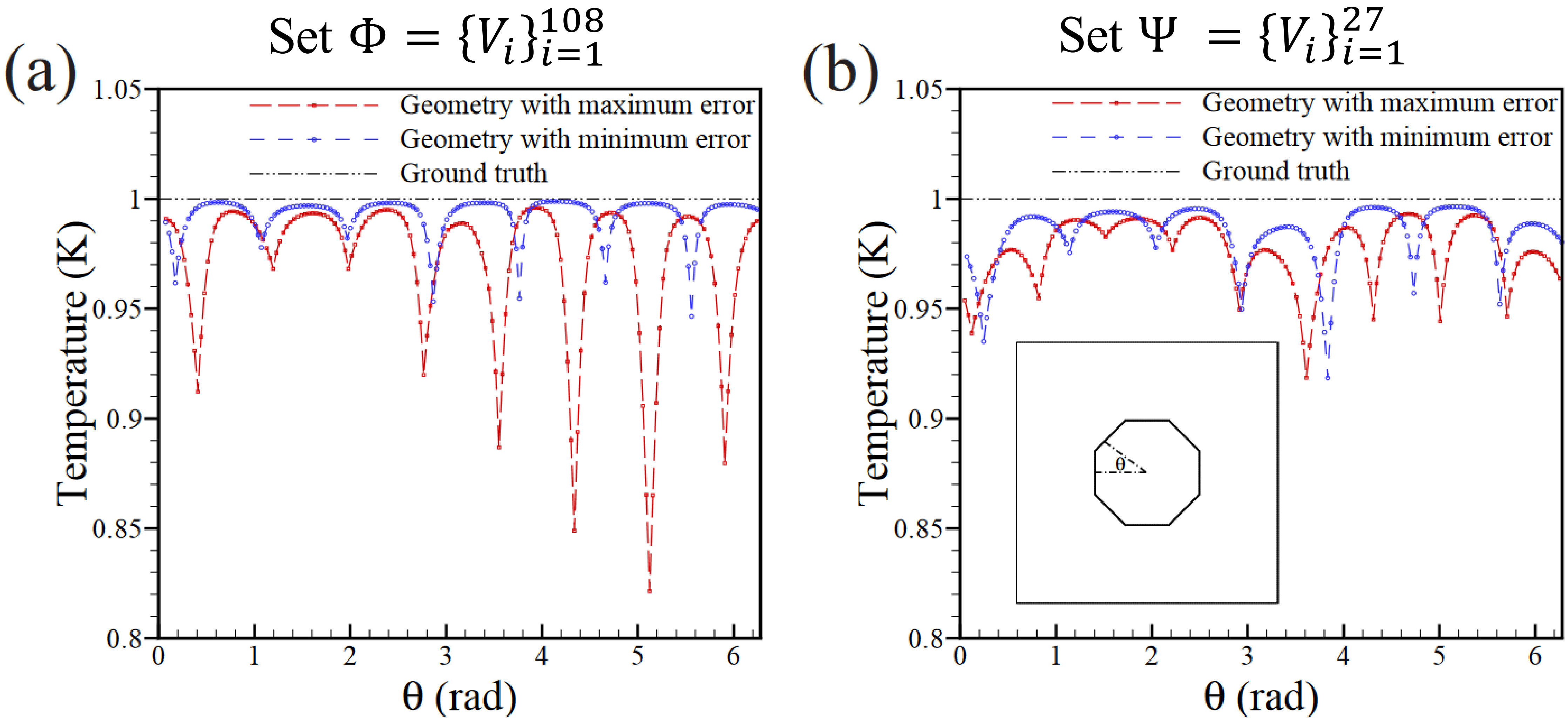


# Error analysis



| | Over the set $\Phi = \{V_i\}_{i=1}^{108}$ | Over the set $\Psi = \{V_i\}_{i=1}^{27}$ |
|---|---|---|
| Average $\|\|\tilde{u} - u\|\|_V / \|\|u\|\|_V$ | 1.08589E − 1 | 1.18250E − 1 |
| Maximum $\|\|\tilde{u} - u\|\|_V / \|\|u\|\|_V$ | 1.43521E − 1 | 1.45620E − 1 |
| Minimum $\|\|\tilde{u} - u\|\|_V / \|\|u\|\|_V$ | 7.73181E − 2 | 8.92147E − 2 |
| Average $\|\|\tilde{v} - v\|\|_V / \|\|v\|\|_V$ | 9.06379E − 2 | 1.07225E − 1 |
| Maximum $\|\|\tilde{v} - v\|\|_V / \|\|v\|\|_V$ | 1.27650E − 1 | 1.27621E − 1 |
| Minimum $\|\|\tilde{v} - v\|\|_V / \|\|v\|\|_V$ | 6.23631E − 2 | 8.26707E − 2 |
| Average $\|\|\tilde{p} - p\|\|_V / \|\|p\|\|_V$ | 2.89030E − 2 | 2.89858E − 2 |
| Maximum $\|\|\tilde{p} - p\|\|_V / \|\|p\|\|_V$ | 3.34799E − 2 | 3.28380E − 2 |
| Minimum $\|\|\tilde{p} - p\|\|_V / \|\|p\|\|_V$ | 2.47451E − 2 | 2.37549E − 2 |
| Average $\|\|\widetilde{T} - T\|\|_V / \|\|T\|\|_V$ | 3.66134E − 2 | 3.89187E − 2 |
| Maximum $\|\|\widetilde{T} - T\|\|_V / \|\|T\|\|_V$ | 4.91492E − 2 | 4.60109E − 2 |
| Minimum $\|\|\widetilde{T} - T\|\|_V / \|\|T\|\|_V$ | 2.57257E − 2 | 2.86674E − 2 |
| Average $\|\|\widetilde{T} - T\|\|_{\Gamma_{\text{inner}}} / \|\|T\|\|_{\Gamma_{\text{inner}}}$ | 2.26461E − 2 | 7.41294E − 2 |
| Maximum $\|\|\widetilde{T} - T\|\|_{\Gamma_{\text{inner}}} / \|\|T\|\|_{\Gamma_{\text{inner}}}$ | 4.14641E − 2 | 1.12027E − 1 |
| Minimum $\|\|\widetilde{T} - T\|\|_{\Gamma_{\text{inner}}} / \|\|T\|\|_{\Gamma_{\text{inner}}}$ | 1.19202E − 2 | 2.07173E − 2 |

# Temperature distribution


Set $\Phi = \{V_i\}_{i=1}^{108}$

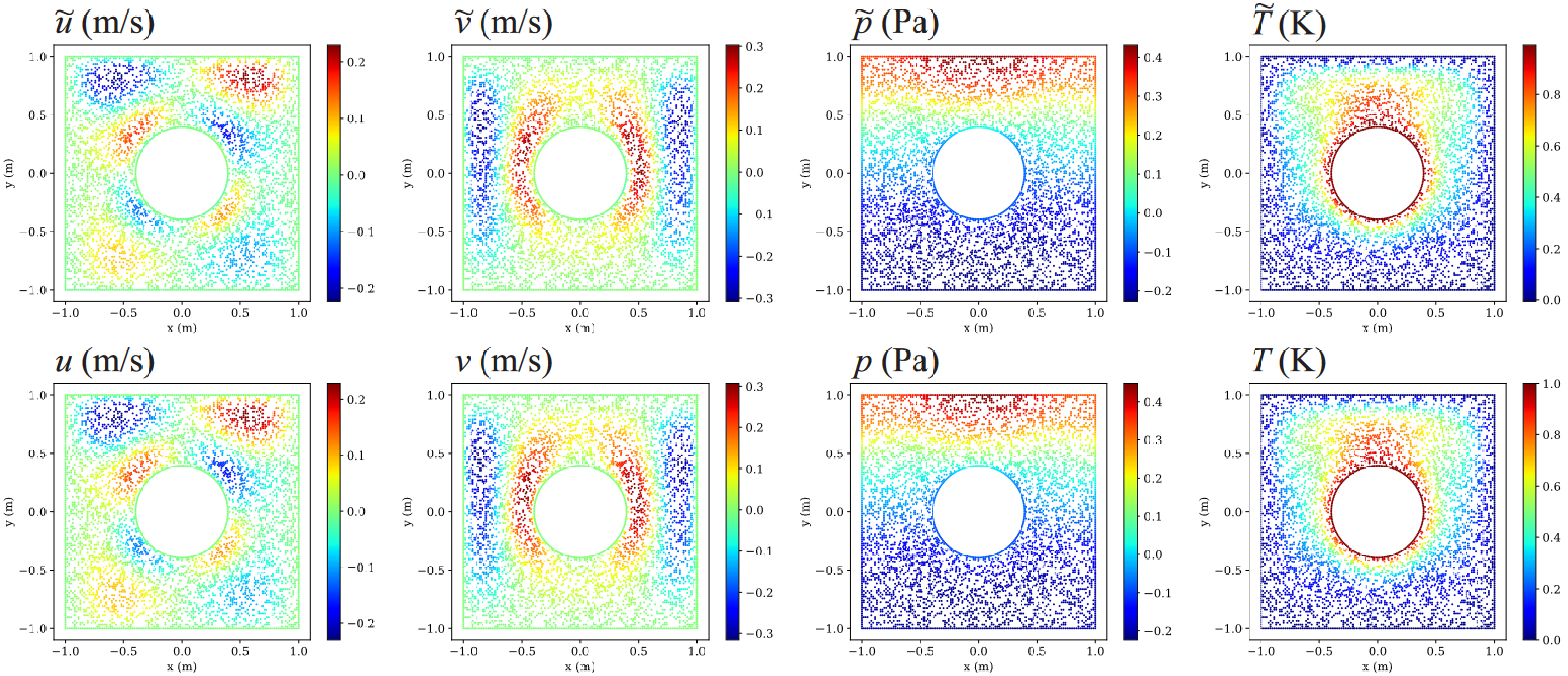
Set $\Psi = \{V_i\}_{i=1}^{27}$

# Natural convection in a square enclosure with a cylinder

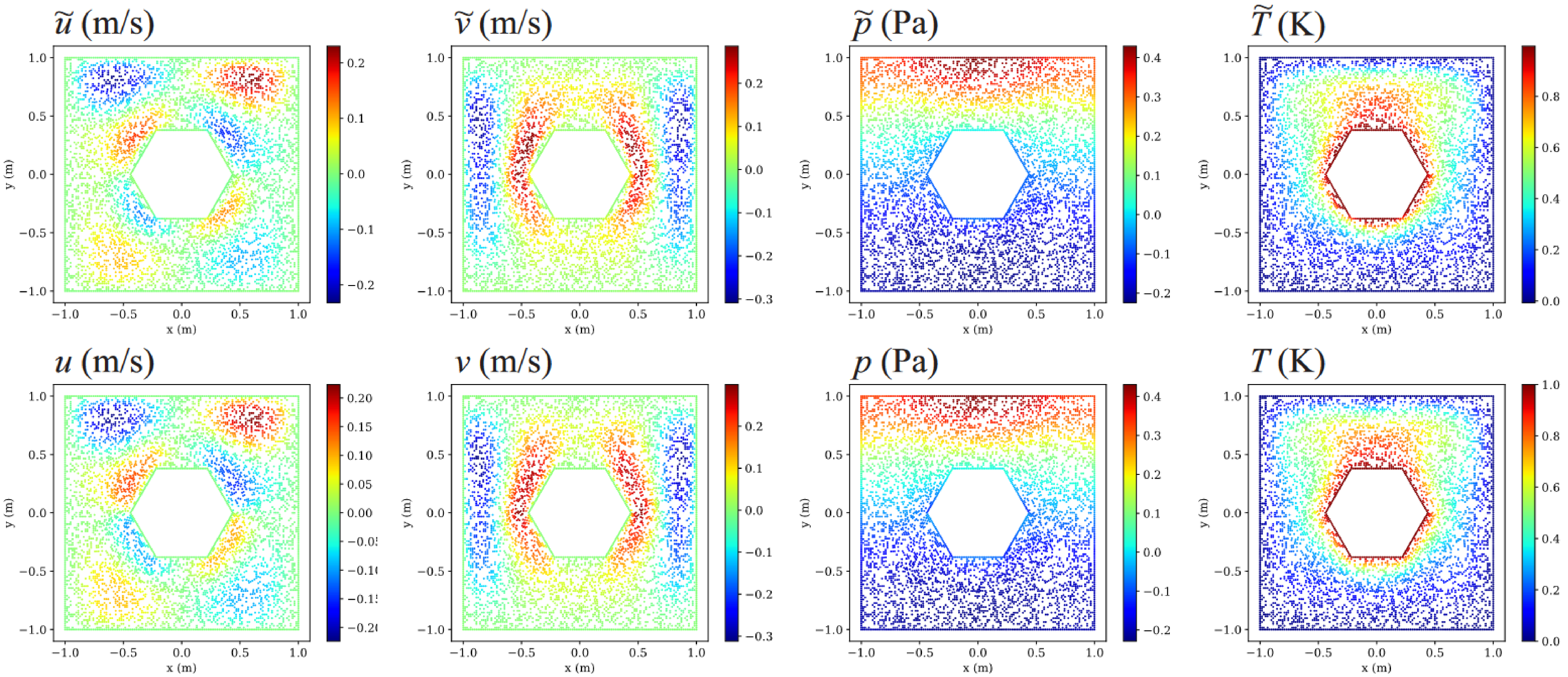
So far the set $\Psi = \{V_i\}_{i=1}^{27}$ has been established from unseen geometries but from seen categories with reference to the set $\Phi = \{V_i\}_{i=1}^{108}$ such as heptagonal, octagonal, nonagonal inner cylinders

Now we set $\Psi = \{V_i\}_{i=1}^{2}$ from unseen geometries from unseen categories with reference to the set $\Phi = \{V_i\}_{i=1}^{108}$ such as circular and hexagonal inner cylinders.

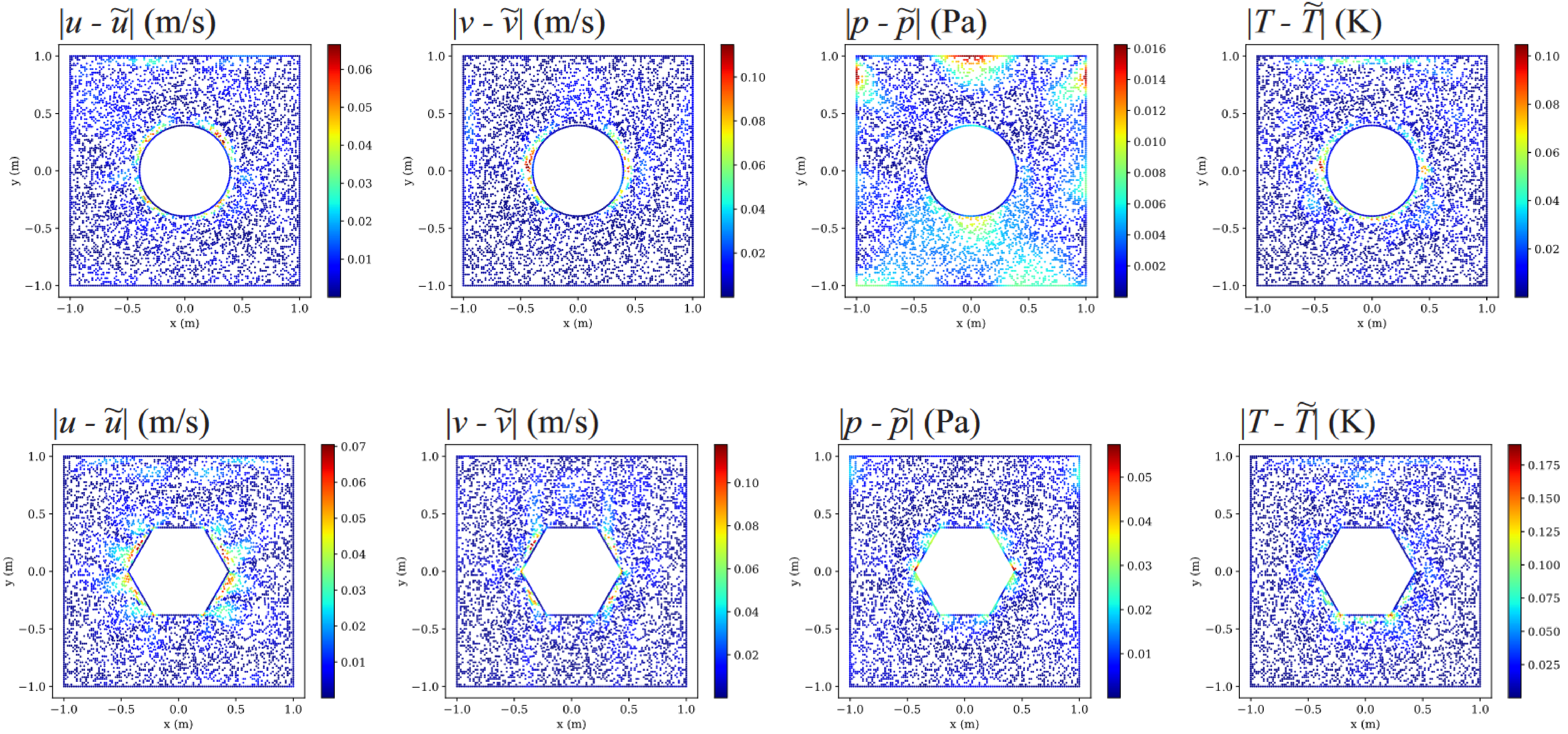
# Prediction by PIPN vs. the ground truth for the circular inner cylinder



# Prediction by PIPN vs. the ground truth for the hexagonal inner cylinder



# Absolute pointwise error distribution



$|u - \widetilde{u}|$ (m/s)

$|v - \widetilde{v}|$ (m/s)

$|p - \widetilde{p}|$ (Pa)

$|T - \widetilde{T}|$ (K)

$|u - \widetilde{u}|$ (m/s)

$|v - \widetilde{v}|$ (m/s)

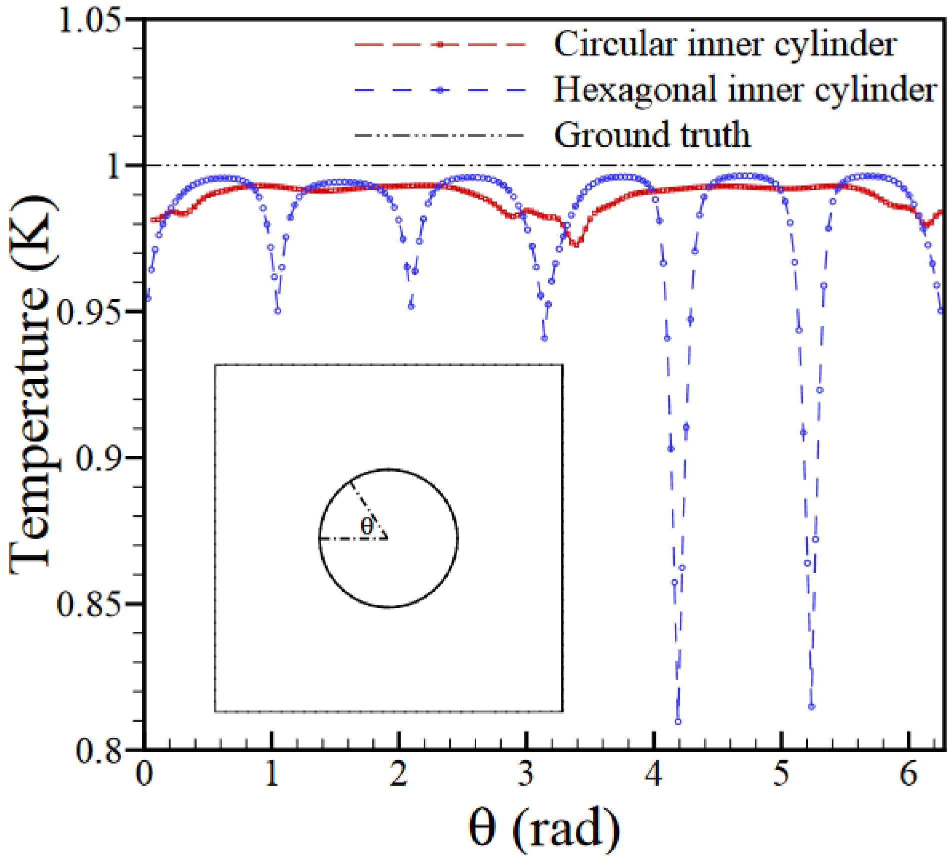
$|p - \widetilde{p}|$ (Pa)

$|T - \widetilde{T}|$ (K)

# Error analysis

| Shape of $W$ (see Eq. 4) | $\frac{\|\|\tilde{u}-u\|\|_V}{\|\|u\|\|_V}$ | $\frac{\|\|\tilde{v}-v\|\|_V}{\|\|v\|\|_V}$ | $\frac{\|\|\tilde{p}-p\|\|_V}{\|\|p\|\|_V}$ | $\frac{\|\|\tilde{T}-T\|\|_V}{\|\|T\|\|_V}$ | $\frac{\|\|\tilde{T}-T\|\|_{\Gamma_{\text{inner}}}}{\|\|T\|\|_{\Gamma_{\text{inner}}}}$ |
|---|---|---|---|---|---|
| Circle | 1.21878E – 1 | 1.08121E – 1 | 2.09506E – 2 | 3.02129E – 2 | 1.15890E – 2 |
| Equilateral hexagon | 1.65543E – 1 | 1.15348E – 1 | 3.32434E – 2 | 4.36705E – 2 | 3.55274E – 2 |

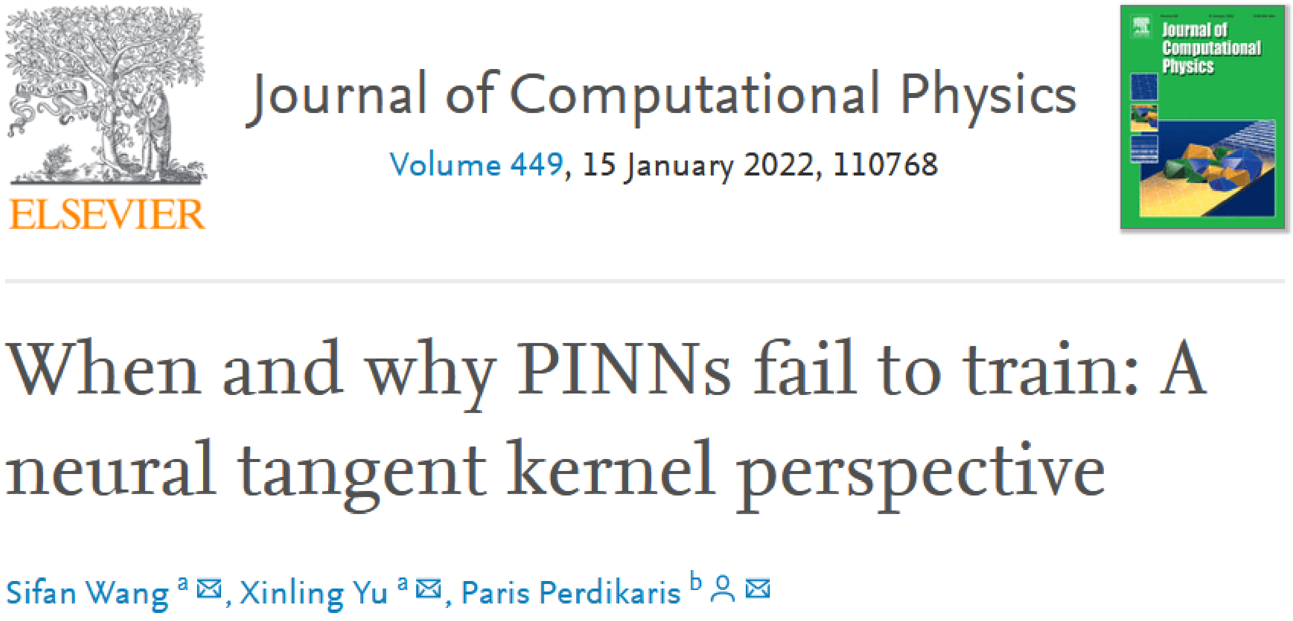
# Temperature distribution



# Physics-Informed PointNet (PIPN)

## New Research Questions

- When and Why PIPN fails to train?!



Journal of Computational Physics

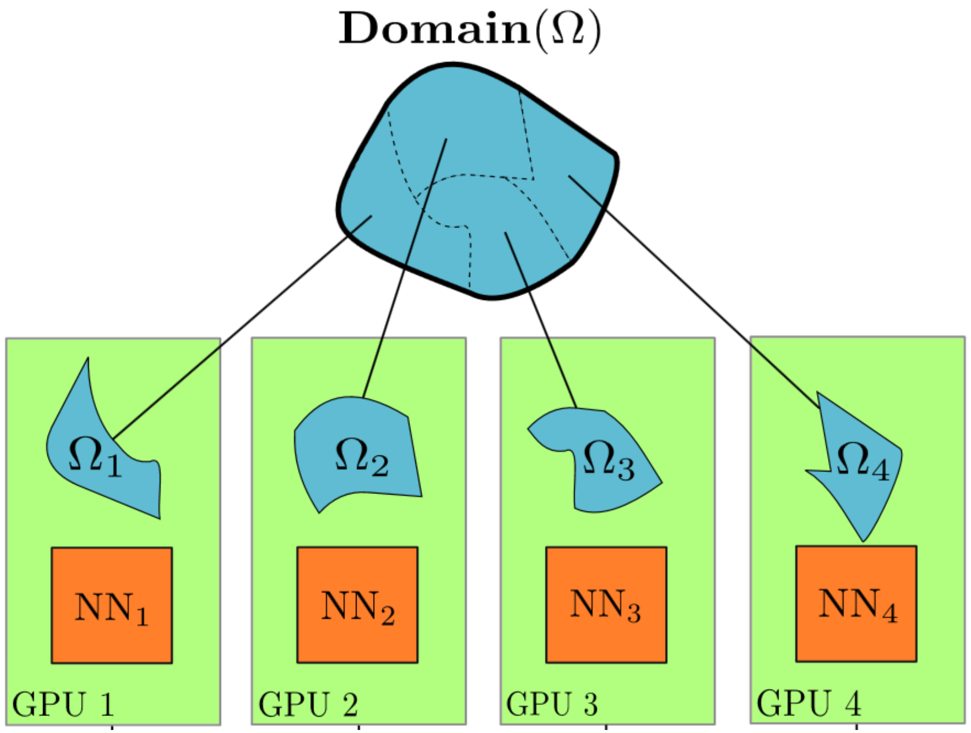
Volume 449, 15 January 2022, 110768

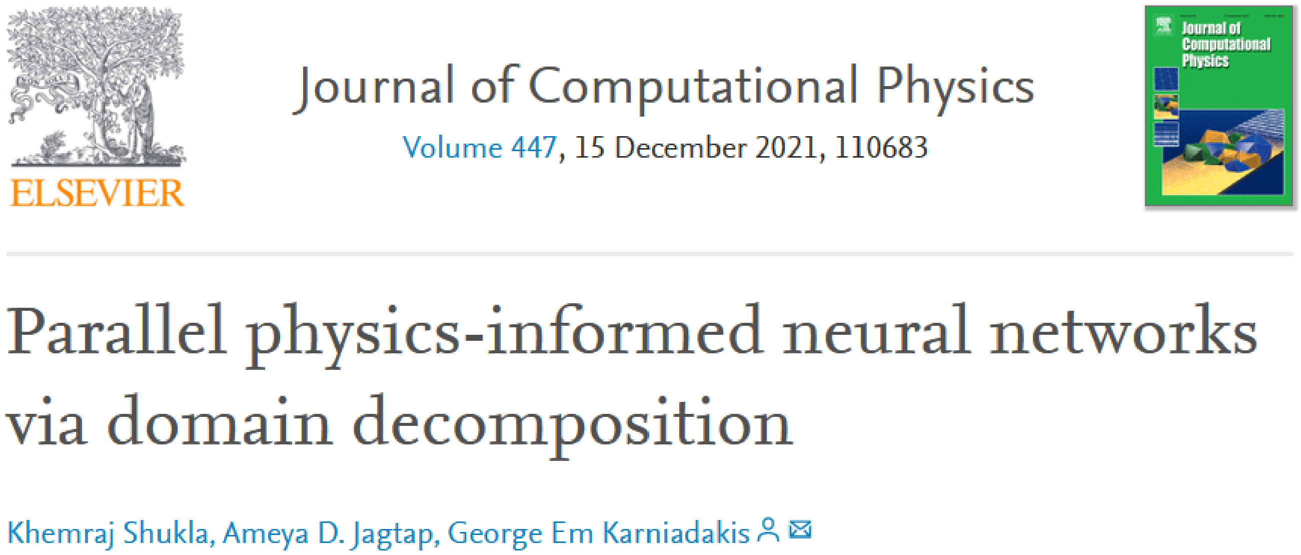
## When and why PINNs fail to train: A neural tangent kernel perspective

Sifan Wang [a], Xinling Yu [a], Paris Perdikaris [b]

How these error analysis change with the presence of "shared" MLPs and the "Max" function?!

- When and Why PIPN fails to train?!
- Parallel physics-informed PointNet via domain decomposition?!





ELSEVIER

Journal of Computational Physics

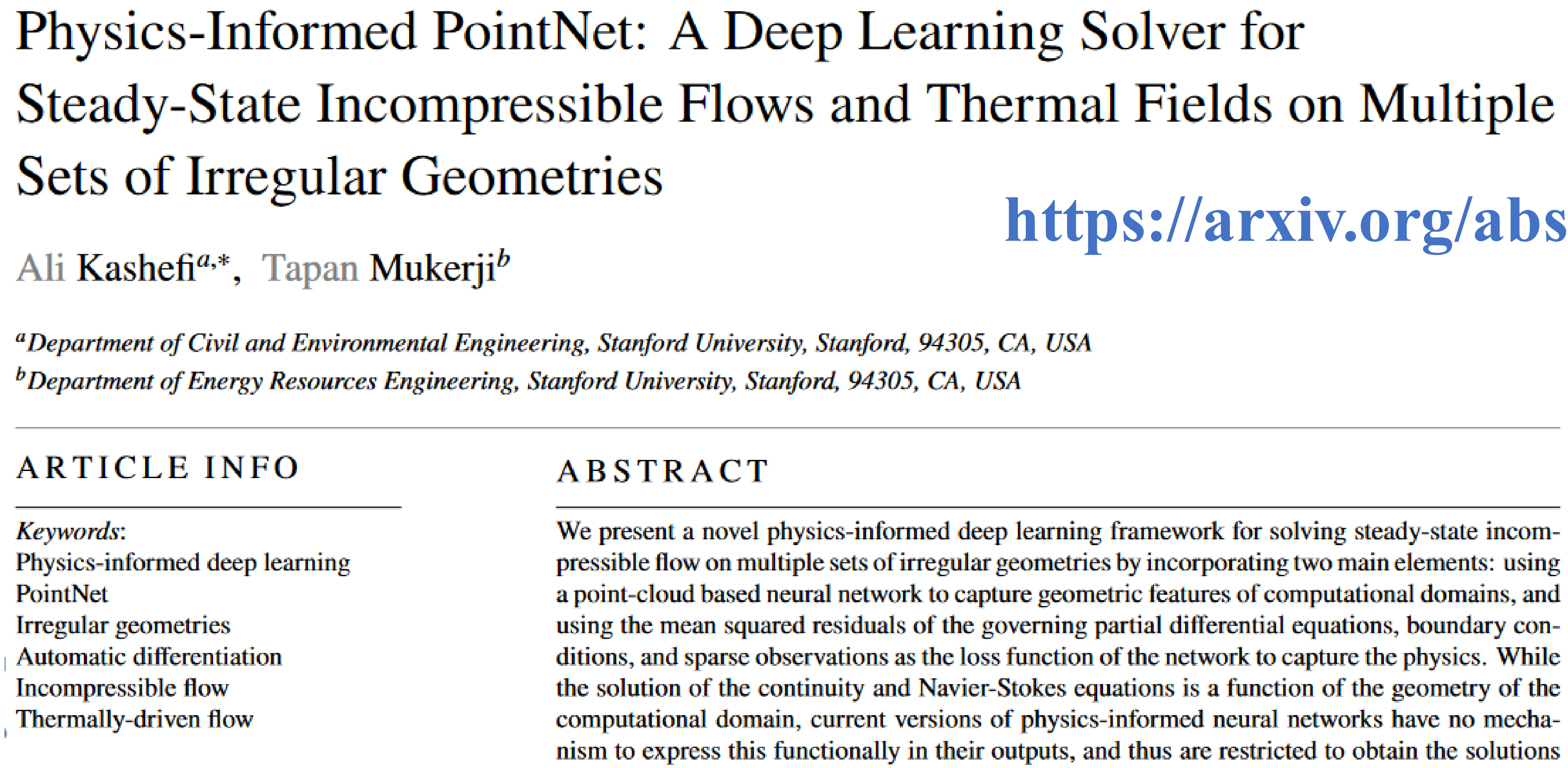
Volume 447, 15 December 2021, 110683

Parallel physics-informed neural networks via domain decomposition

Khemraj Shukla, Ameya D. Jagtap, George Em Karniadakis

How we should decompose "multiple domains" while the solution of each domain depends on its geometry?!

- When and Why PIPN fails to train?!
- Parallel physics-informed PointNet via domain decomposition?!
- Extension to 3D and unsteady problems with applications to other areas such as compressible flows, linear and nonlinear elasticity, …



# Physics-Informed PointNet: A Deep Learning Solver for Steady-State Incompressible Flows and Thermal Fields on Multiple Sets of Irregular Geometries

https://arxiv.org/abs/2202.05476

Ali Kashefi[a,*], Tapan Mukerji[b]

[a] *Department of Civil and Environmental Engineering, Stanford University, Stanford, 94305, CA, USA*

[b] *Department of Energy Resources Engineering, Stanford University, Stanford, 94305, CA, USA*

ARTICLE INFO

*Keywords*:
Physics-informed deep learning
PointNet
Irregular geometries
Automatic differentiation
Incompressible flow
Thermally-driven flow

ABSTRACT

We present a novel physics-informed deep learning framework for solving steady-state incompressible flow on multiple sets of irregular geometries by incorporating two main elements: using a point-cloud based neural network to capture geometric features of computational domains, and using the mean squared residuals of the governing partial differential equations, boundary conditions, and sparse observations as the loss function of the network to capture the physics. While the solution of the continuity and Navier-Stokes equations is a function of the geometry of the computational domain, current versions of physics-informed neural networks have no mechanism to express this functionally in their outputs, and thus are restricted to obtain the solutions

# Physics-Informed PointNet (PIPN)

## Summary

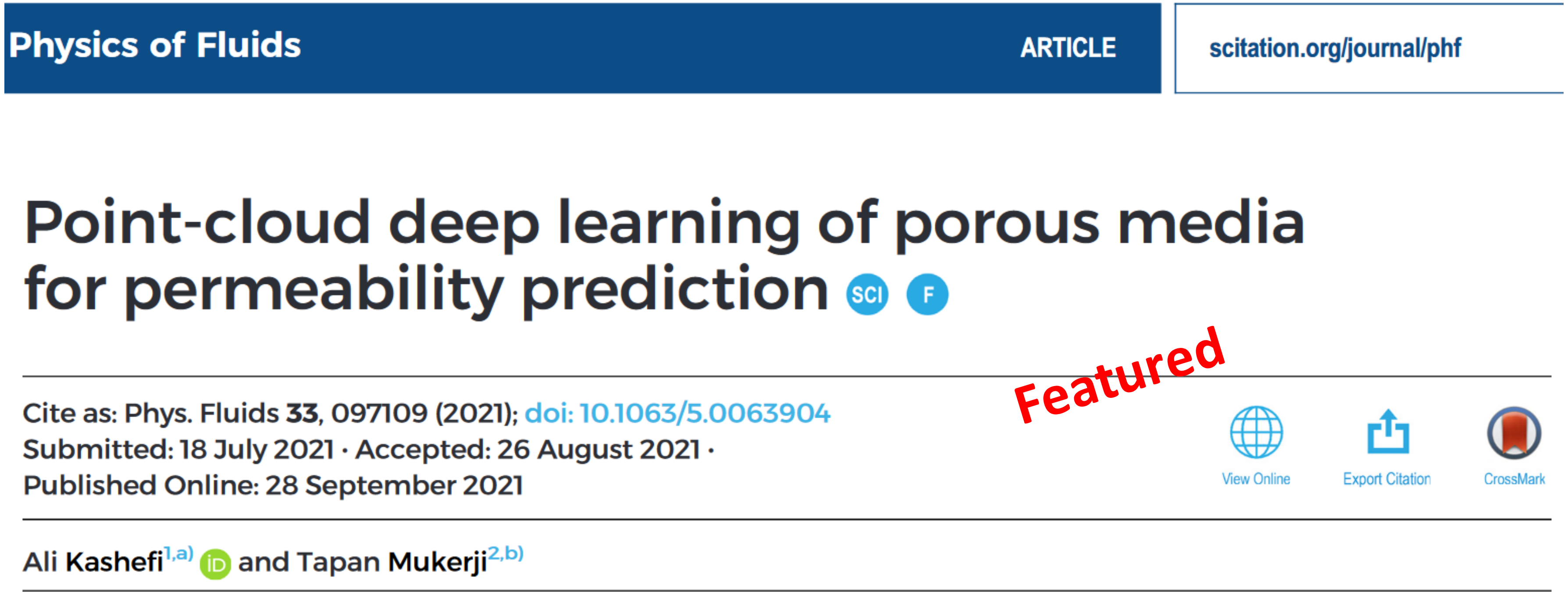
- We proposed PIPN as a novel physics-informed deep learning framework.

- PIPN solves forward and inverse time-independent problems on several irregular domains by training only once.

- PIPN overcomes the shortcoming of regular PINNs that need to be retrained for any single domain with a new geometry.

- We showed applications of PIPN for incompressible flows and thermal fields.

# A Short Advertisement



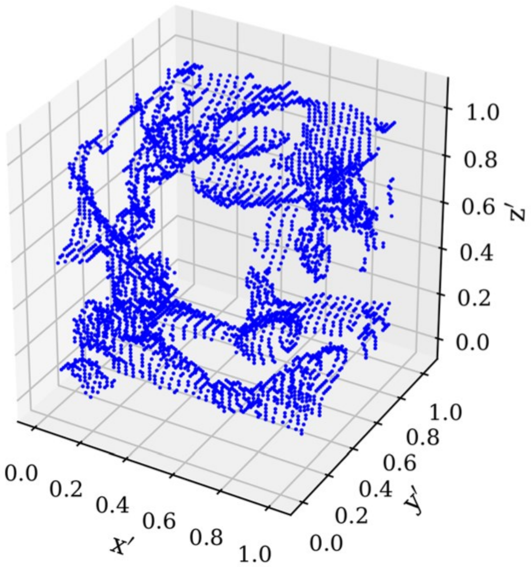
Physics of Fluids ARTICLE scitation.org/journal/phf

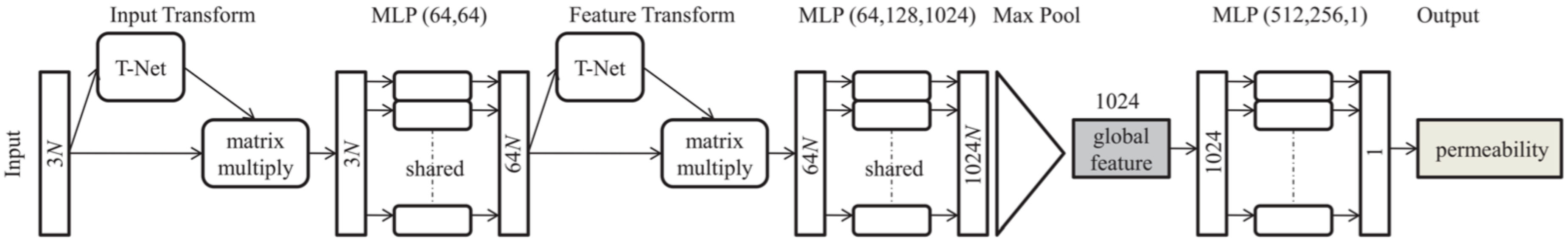
# Point-cloud deep learning of porous media for permeability prediction

Featured

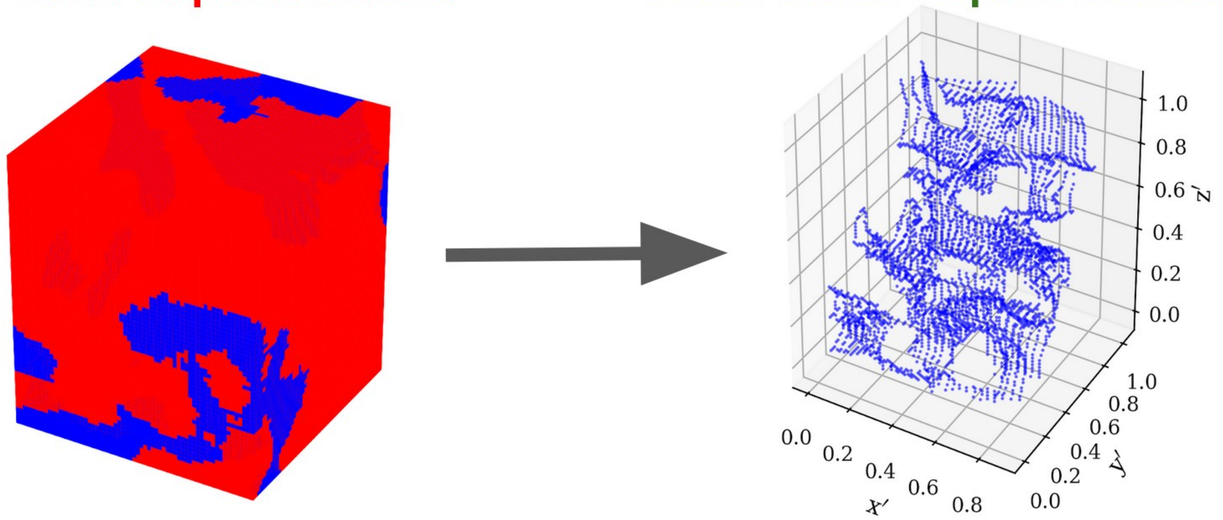
Cite as: Phys. Fluids **33**, 097109 (2021); doi: 10.1063/5.0063904
Submitted: 18 July 2021 · Accepted: 26 August 2021 ·
Published Online: 28 September 2021

Ali Kashefi[1,a) and Tapan Mukerji[2,b)





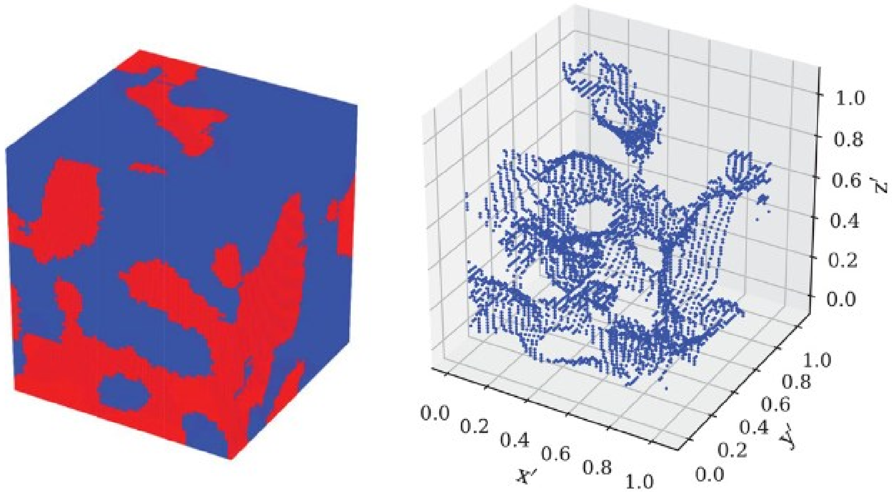
**Voxel Representation** **Point Cloud Representation**

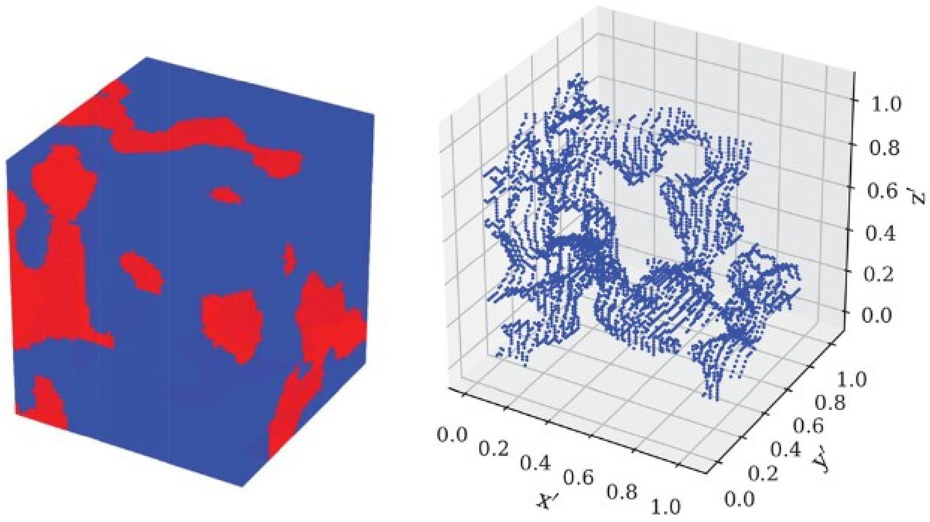


Data Size: Data Size:

64x64x64 = 262144 4000x3 = 12000

**Needs Huge GPU Memory** ● **Needs Low GPU Memory**

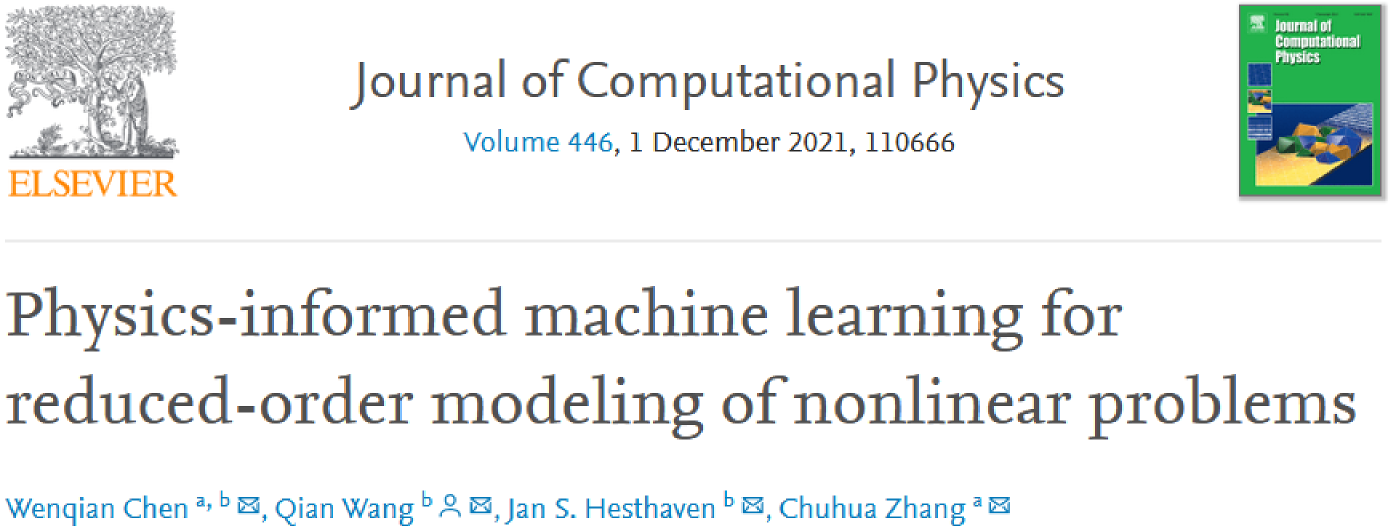




Thank you!

Q&A

# Supportive Materials

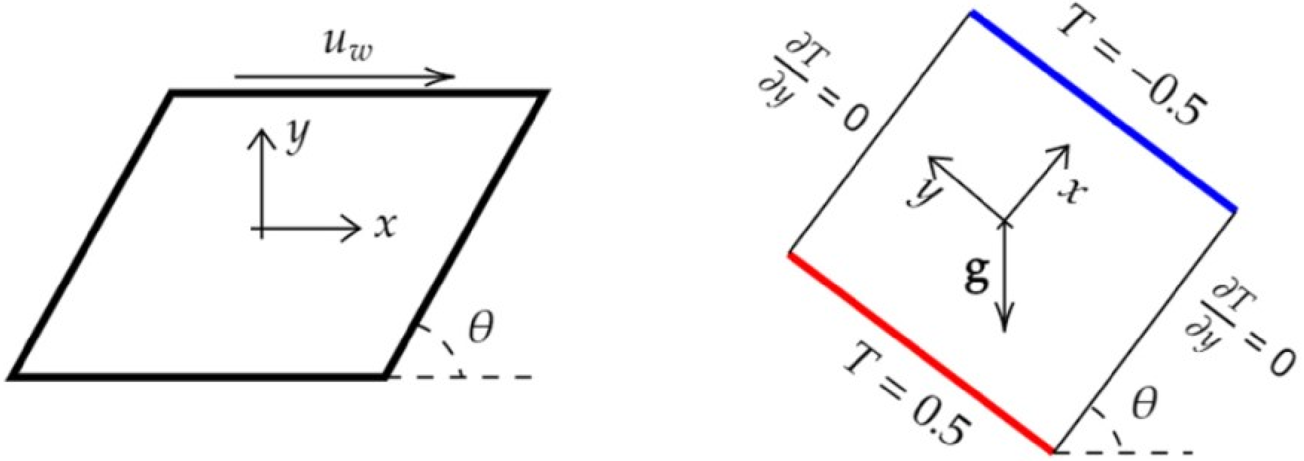


Journal of Computational Physics

Volume 446, 1 December 2021, 110666

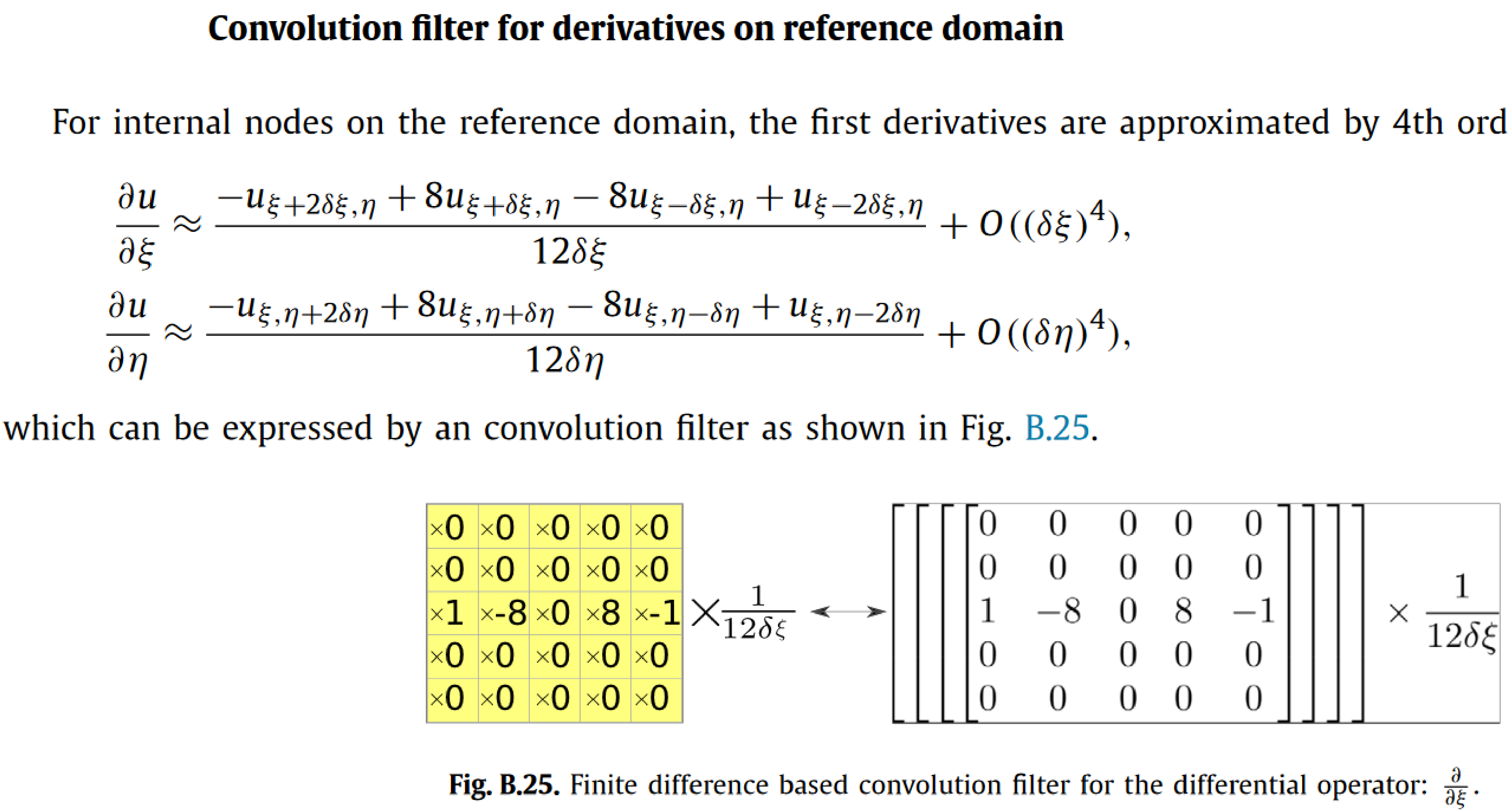
# Physics-informed machine learning for reduced-order modeling of nonlinear problems

Wenqian Chen [a, b], Qian Wang [b], Jan S. Hesthaven [b], Chuhua Zhang [a]


$u_w$
$y$
$x$
$\theta$
$\frac{\partial T}{\partial y} = 0$
$T = -0.5$
$y$
$x$
$\mathbf{g}$
$\frac{\partial T}{\partial y} = 0$
$T = 0.5$
$\theta$

(a)

(b)



### Convolution filter for derivatives on reference domain

For internal nodes on the reference domain, the first derivatives are approximated by 4th order central differences,

$$\frac{\partial u}{\partial \xi} \approx \frac{-u_{\xi+2\delta\xi,\eta} + 8u_{\xi+\delta\xi,\eta} - 8u_{\xi-\delta\xi,\eta} + u_{\xi-2\delta\xi,\eta}}{12\delta\xi} + O((\delta\xi)^4),$$

$$\frac{\partial u}{\partial \eta} \approx \frac{-u_{\xi,\eta+2\delta\eta} + 8u_{\xi,\eta+\delta\eta} - 8u_{\xi,\eta-\delta\eta} + u_{\xi,\eta-2\delta\eta}}{12\delta\eta} + O((\delta\eta)^4),$$

which can be expressed by an convolution filter as shown in Fig. B.25.

$$\begin{array}{ccccc} \times 0 & \times 0 & \times 0 & \times 0 & \times 0 \\ \times 0 & \times 0 & \times 0 & \times 0 & \times 0 \\ \times 1 & \times\text{-}8 & \times 0 & \times 8 & \times\text{-}1 \\ \times 0 & \times 0 & \times 0 & \times 0 & \times 0 \\ \times 0 & \times 0 & \times 0 & \times 0 & \times 0 \end{array} \times \frac{1}{12\delta\xi} \longleftrightarrow \left[\left[\left[\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & -8 & 0 & 8 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}\right]\right]\right] \times \frac{1}{12\delta\xi}$$

**Fig. B.25.** Finite difference based convolution filter for the differential operator: $\frac{\partial}{\partial \xi}$.

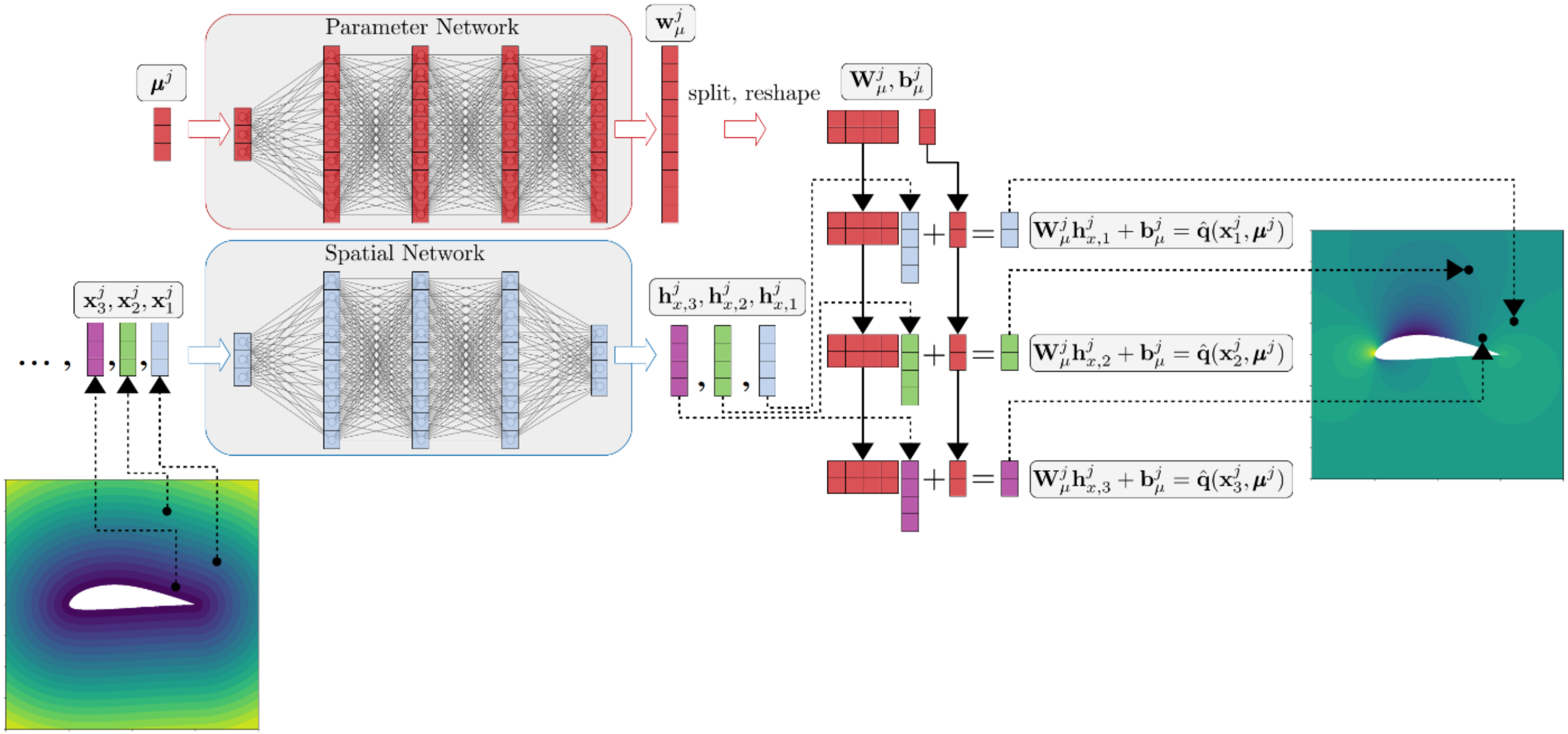
The snapshot taken from the PhyGeoNet journal paper (Gao et al. 2021)

# Non-linear Independent Dual System (NIDS) for Discretization-independent Surrogate Modeling over Complex Geometries

**Supervised Learning**

J. Duvall et al. (2021) on arXiv

*University of Michigan*



Parameter Network
$\mathbf{w}^j_\mu$
$\boldsymbol{\mu}^j$
split, reshape
$\mathbf{W}^j_\mu, \mathbf{b}^j_\mu$
Spatial Network
$\mathbf{x}^j_3, \mathbf{x}^j_2, \mathbf{x}^j_1$
$\dots,$
$\mathbf{h}^j_{x,3}, \mathbf{h}^j_{x,2}, \mathbf{h}^j_{x,1}$
$\mathbf{W}^j_\mu \mathbf{h}^j_{x,1} + \mathbf{b}^j_\mu = \hat{\mathbf{q}}(\mathbf{x}^j_1, \boldsymbol{\mu}^j)$
$\mathbf{W}^j_\mu \mathbf{h}^j_{x,2} + \mathbf{b}^j_\mu = \hat{\mathbf{q}}(\mathbf{x}^j_2, \boldsymbol{\mu}^j)$
$\mathbf{W}^j_\mu \mathbf{h}^j_{x,3} + \mathbf{b}^j_\mu = \hat{\mathbf{q}}(\mathbf{x}^j_3, \boldsymbol{\mu}^j)$

# IMPROVED ARCHITECTURES AND TRAINING ALGORITHMS FOR DEEP OPERATOR NETWORKS

S. Wang et al. (2021) on arXiv

*University of Pennsylvania*

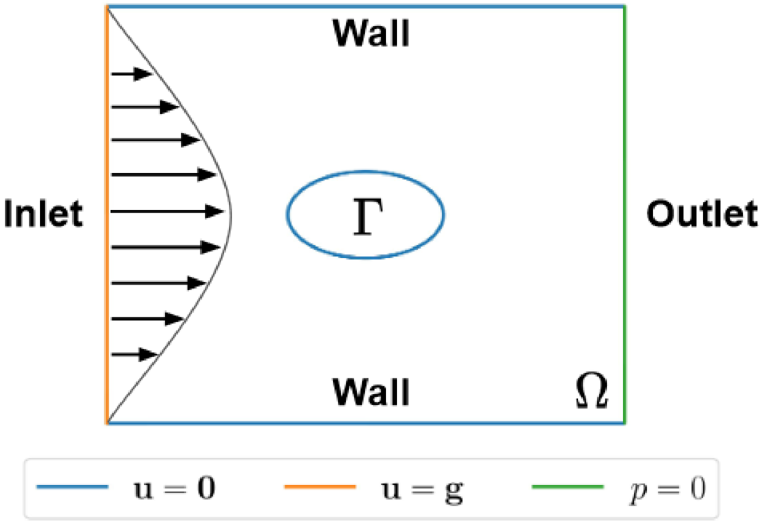


Figure 11: *Stokes equation:* Illustration of the computational domain and boundary conditions.

$$\partial\Gamma = \partial\Gamma(\phi) = (a\cos(\phi) + \frac{1}{2}, b\sin(\phi) + \frac{1}{2}), \quad \phi \in [0, 2\pi),$$

# Method of manufactured solutions in non-trivial geometries

A divergence free velocity field:

$$u = \cos(x)\sin(y)$$

$$v = -\cos(y)\sin(x)$$

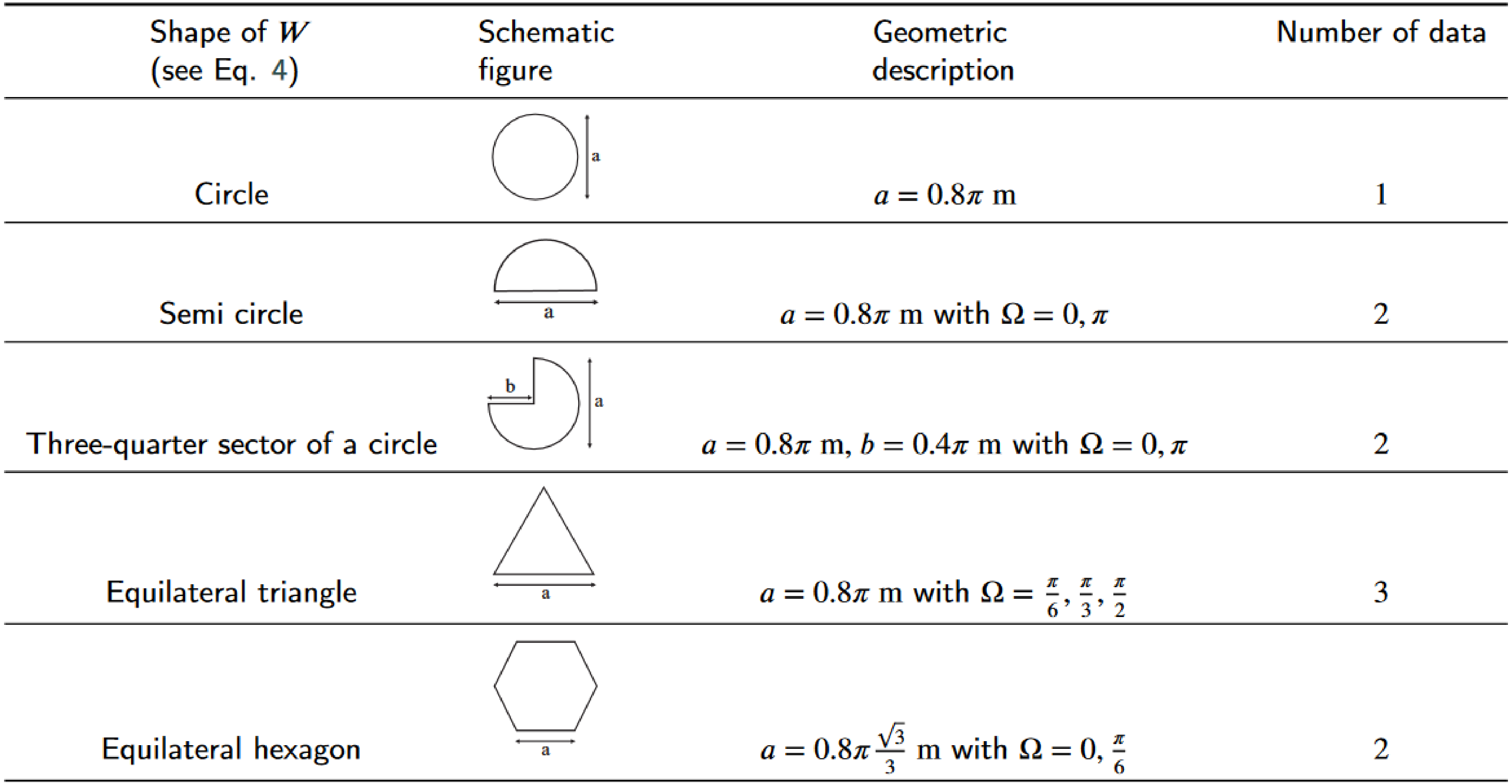
with an arbitrary pressure field:

$$p = -\frac{\rho}{4}[\cos(2y) + \cos(2x)]$$

The set $\Phi = \{V_i\}_{i=1}^{26}$ contains 26 geometries, where we know the velocity and pressure Dirichlet boundary conditions!

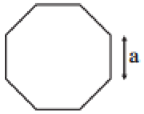
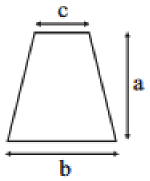
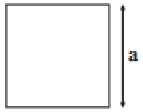
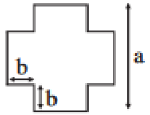
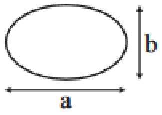
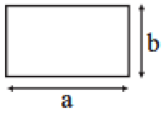
**Loss function for the set $\Phi$ = conservation of mass + conservation of momentum + velocity boundary conditions + pressure boundary conditions**

# Geometric descriptions of domains of the set $\Phi = \{V_i\}_{i=1}^{26}$

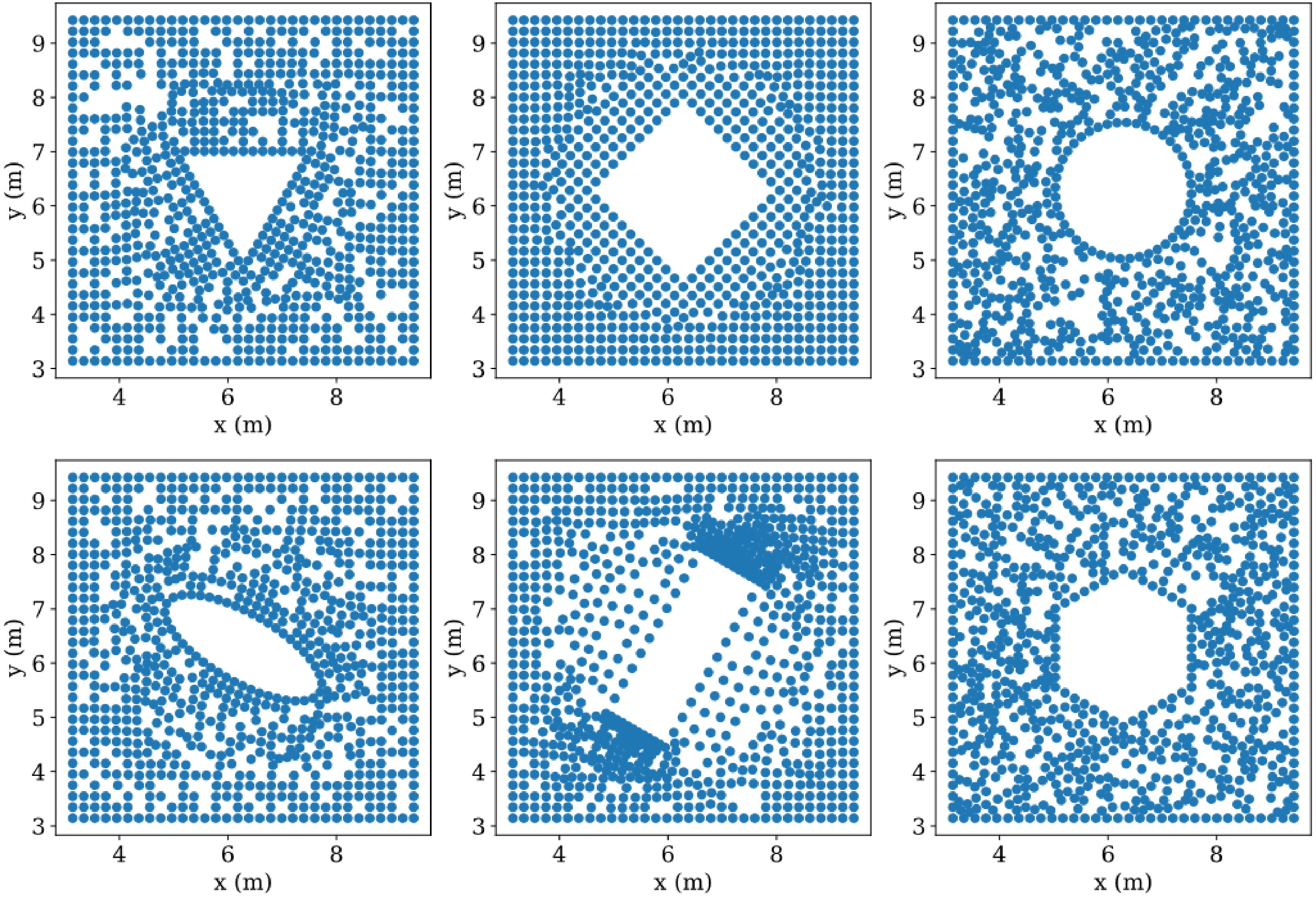


| Shape of $W$ (see Eq. 4) | Schematic figure | Geometric description | Number of data |
|---|---|---|---|
| Circle | | $a = 0.8\pi$ m | 1 |
| Semi circle | | $a = 0.8\pi$ m with $\Omega = 0, \pi$ | 2 |
| Three-quarter sector of a circle | | $a = 0.8\pi$ m, $b = 0.4\pi$ m with $\Omega = 0, \pi$ | 2 |
| Equilateral triangle | | $a = 0.8\pi$ m with $\Omega = \frac{\pi}{6}, \frac{\pi}{3}, \frac{\pi}{2}$ | 3 |
| Equilateral hexagon | | $a = 0.8\pi \frac{\sqrt{3}}{3}$ m with $\Omega = 0, \frac{\pi}{6}$ | 2 |

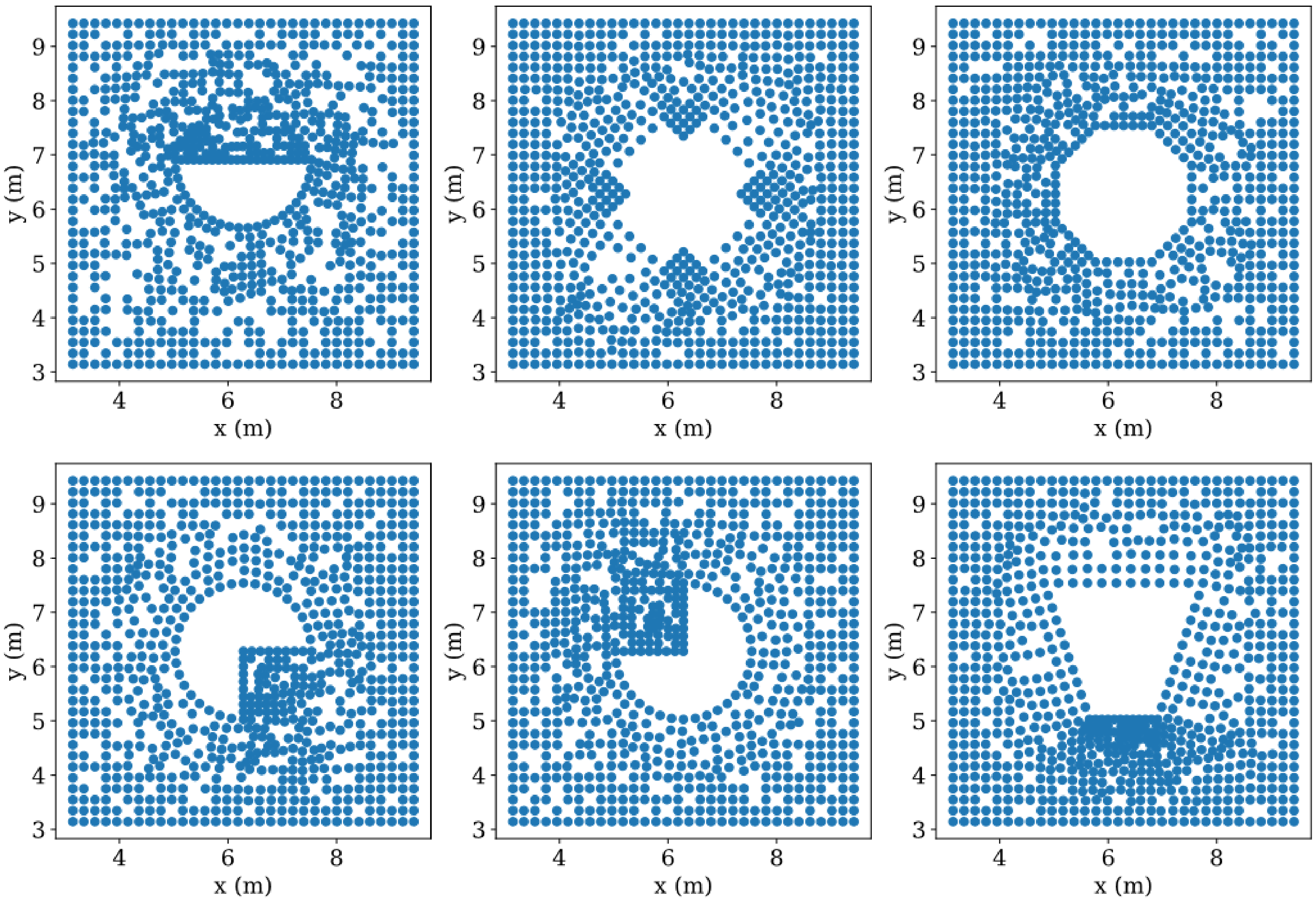
# Geometric descriptions of domains of the set $\Phi = \{V_i\}_{i=1}^{26}$

| | | | |
|---|---|---|---|
| Equilateral octagon |  | $a = 0.8\pi(\sqrt{2} - 1)$ m with $\Omega = 0, \frac{\pi}{8}$ | 2 |
| Trapezoid |  | $a = 0.8\pi$ m, $b = \frac{14}{15}\pi$ m,<br>$c = 0.4\pi$ m with $\Omega = 0, \pi$ | 2 |
| Square |  | $a = 0.8\pi$ m with $\Omega = 0, \frac{\pi}{4}$ | 2 |
| Symmetrical star |  | $a = 0.8\pi$ m, $b = 0.16\pi$ m with $\Omega = 0, \frac{\pi}{4}$ | 2 |
| Ellipse |  | $a = 1.04\pi$ m, $b = 0.4\pi$ m with $\Omega = \frac{\pi}{6}, \frac{5\pi}{6}$<br>and $a = 0.96\pi$ m, $b = 0.4\pi$ m with $\Omega = 0, \frac{\pi}{2}$ | 4 |
| Rectangle |  | $a = 1.04\pi$ m, $b = 0.4\pi$ m with $\Omega = \frac{\pi}{3}, \frac{2\pi}{3}$<br>and $a = 0.96\pi$ m, $b = 0.4\pi$ m with $\Omega = \frac{\pi}{6}, \frac{5\pi}{6}$ | 4 |

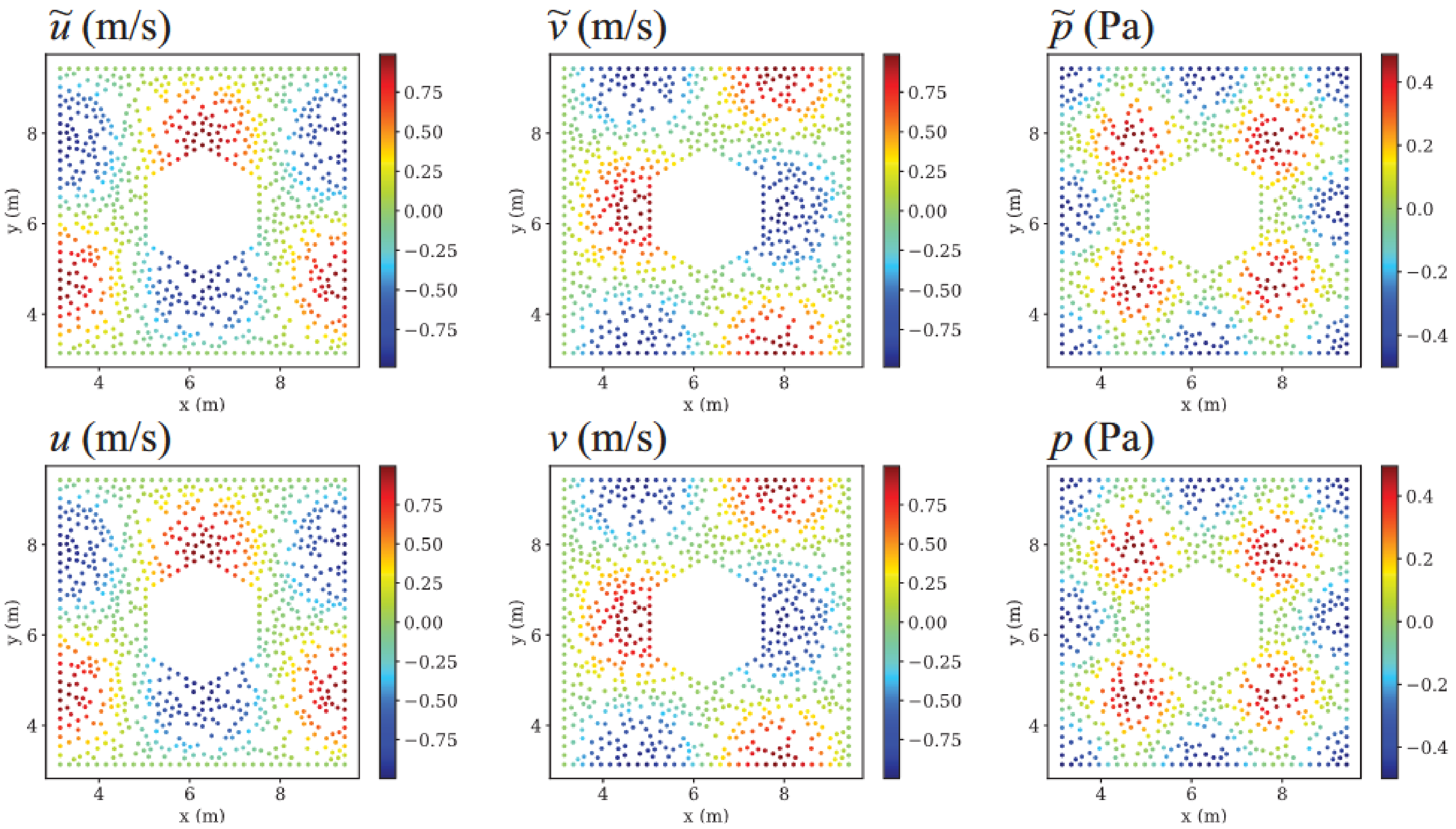
# Examples of point-cloud representations of domains of the set $\Phi = \{V_i\}_{i=1}^{26}$



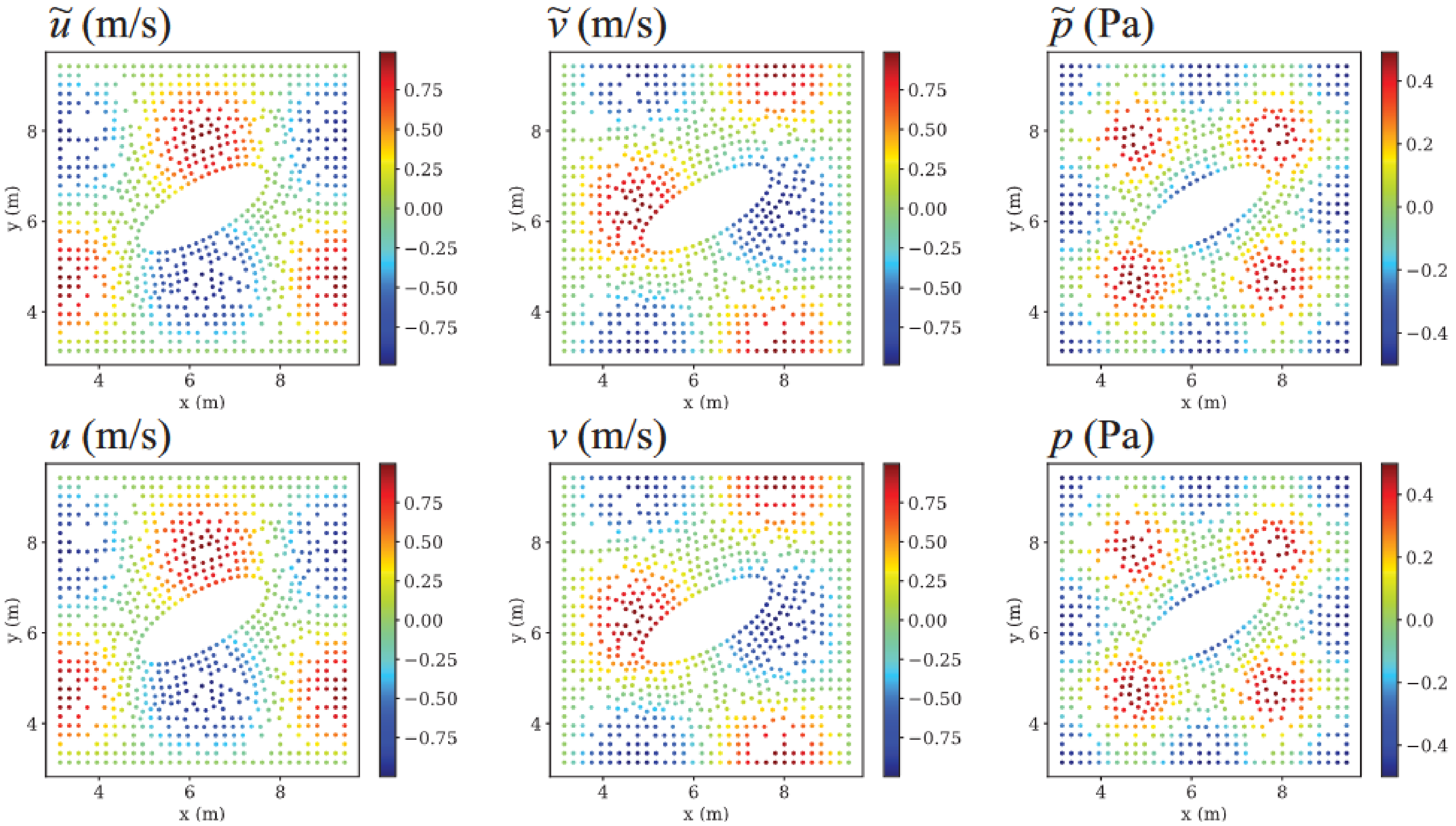
# Examples of point-cloud representations of domains of the set $\Phi = \{V_i\}_{i=1}^{26}$



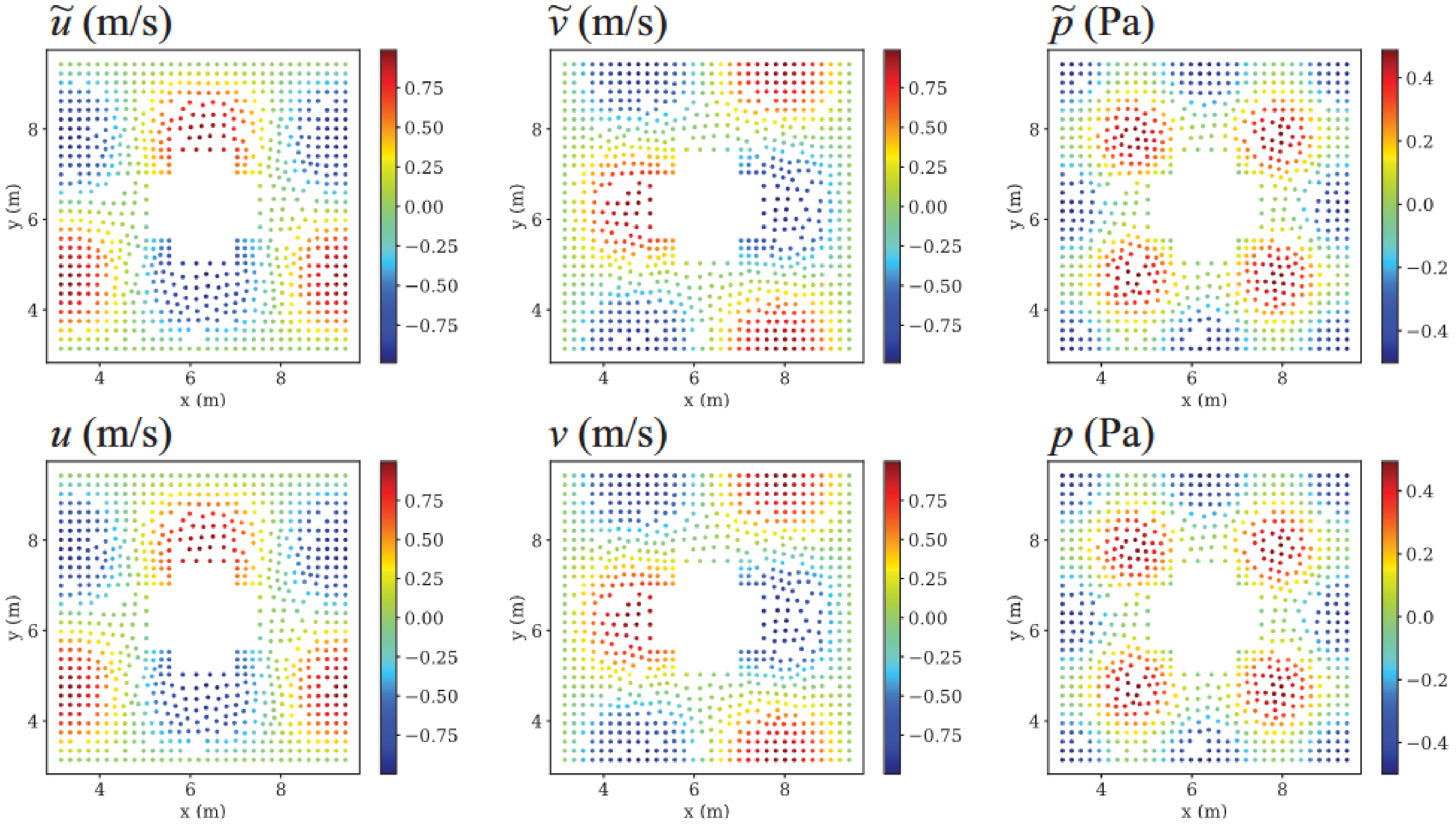
# Prediction by PIPN vs. the ground truth for a domain of the set $\Phi = \{V_i\}_{i=1}^{26}$



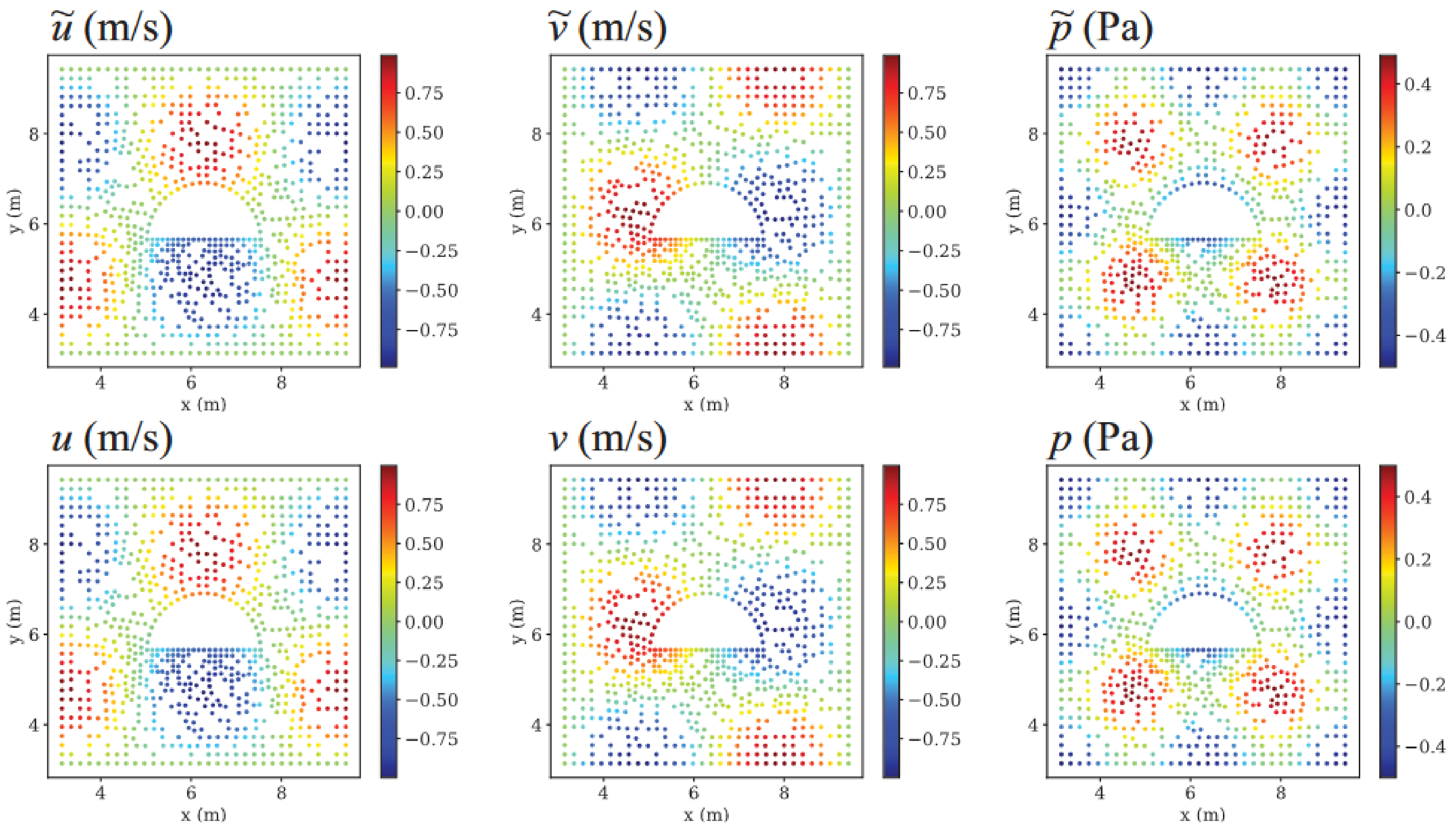
# Prediction by PIPN vs. the ground truth for a domain of the set $\Phi = \{V_i\}_{i=1}^{26}$



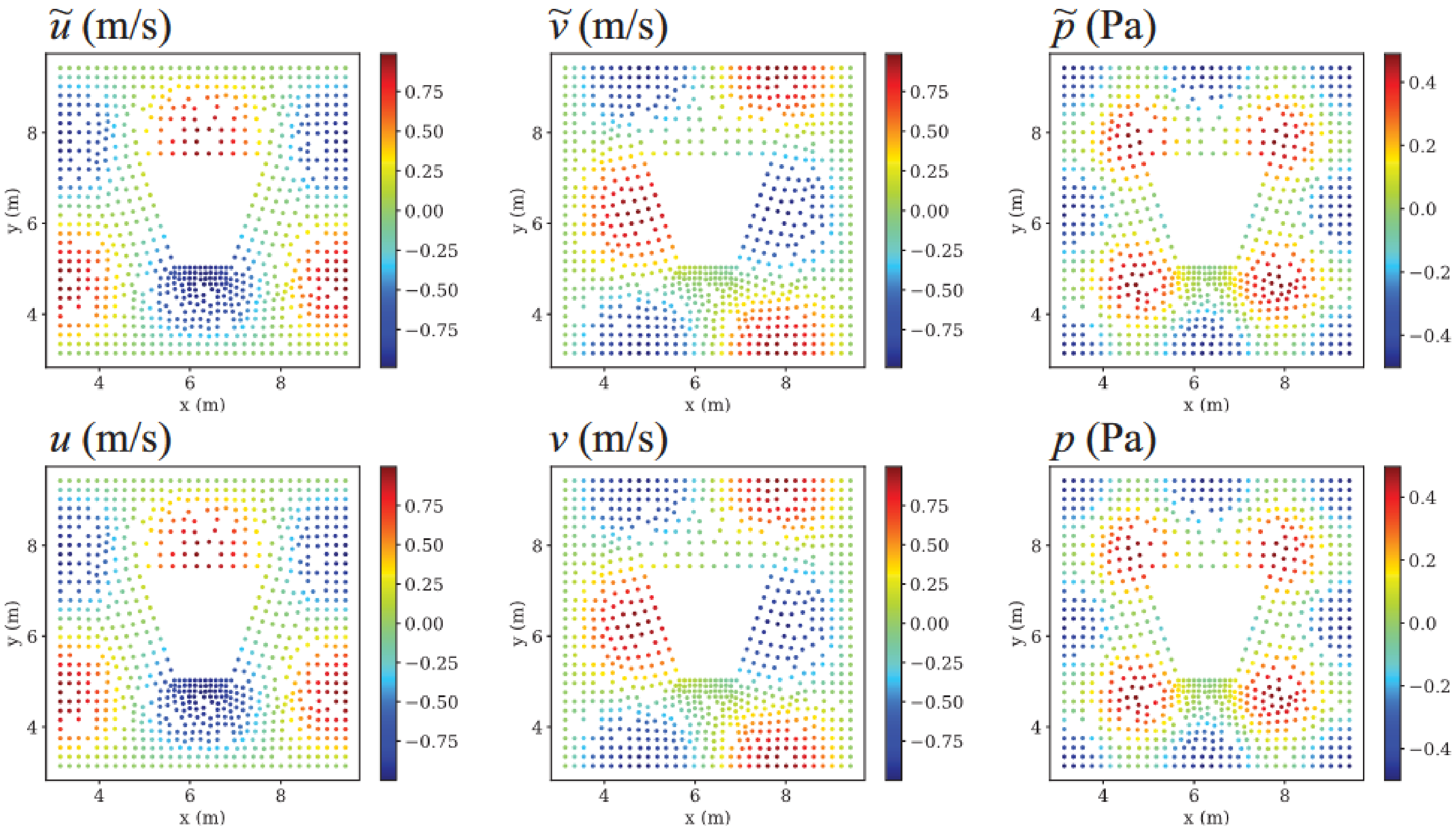
# Prediction by PIPN vs. the ground truth for a domain of the set $\Phi = \{V_i\}_{i=1}^{26}$



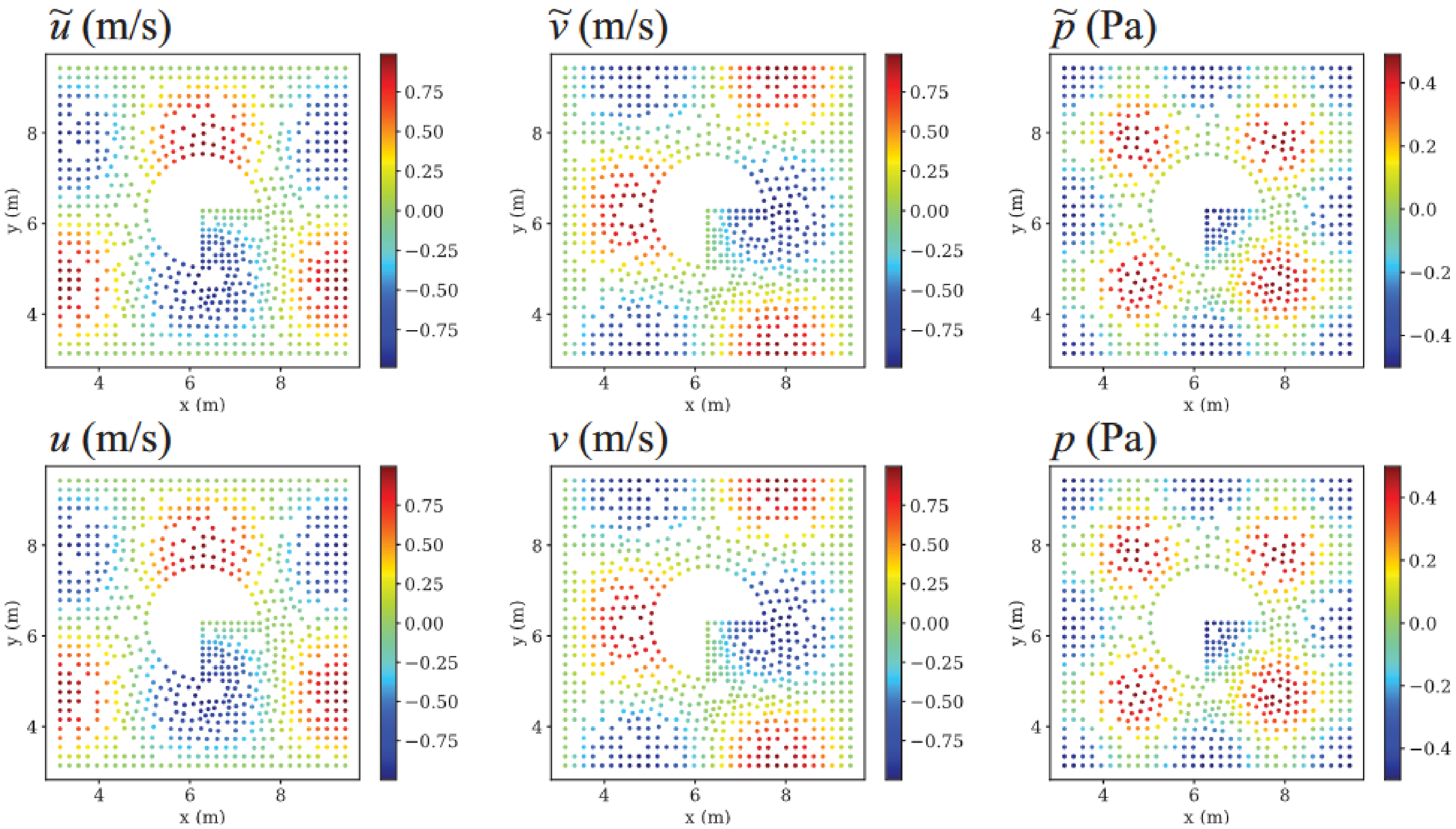
# Prediction by PIPN vs. the ground truth for a domain of the set $\Phi = \{V_i\}_{i=1}^{26}$



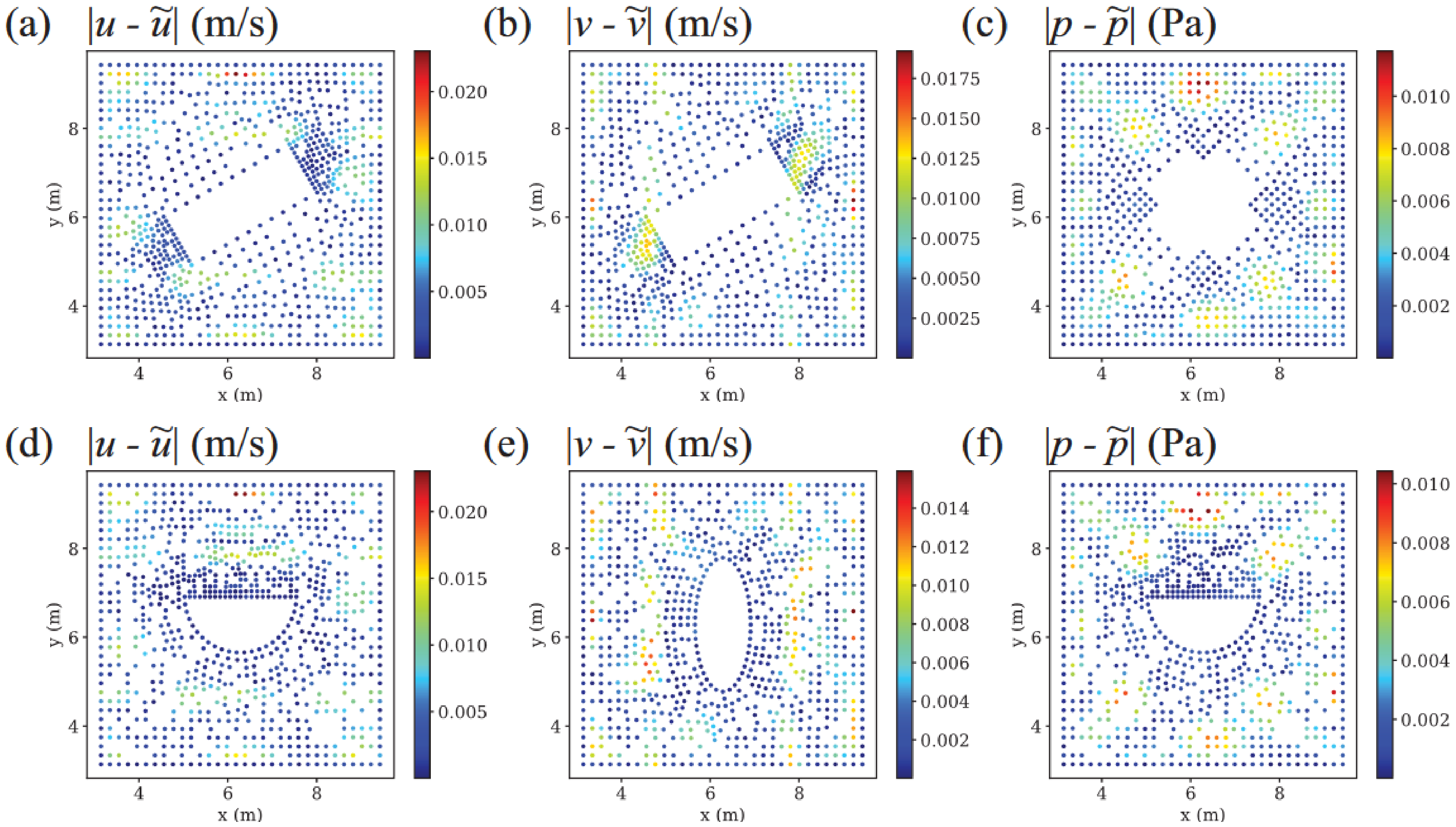
# Prediction by PIPN vs. the ground truth for a domain of the set $\Phi = \{V_i\}_{i=1}^{26}$



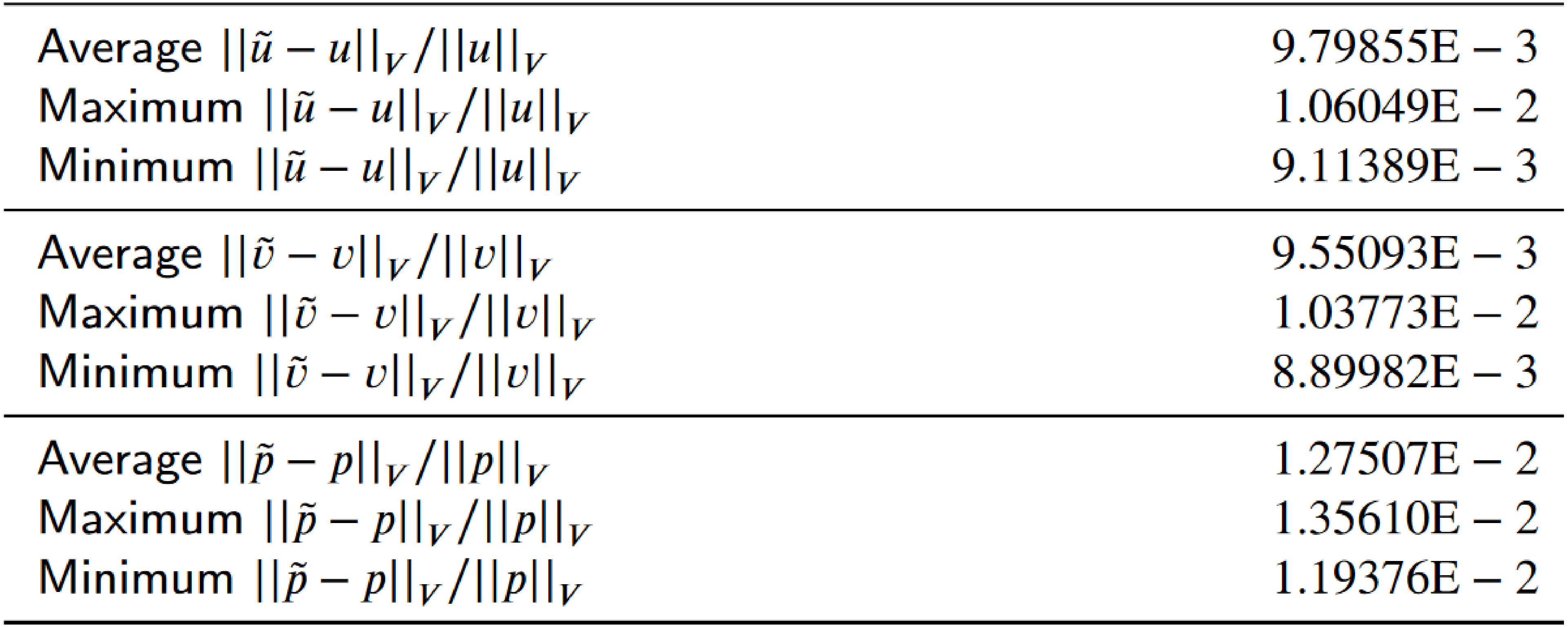
# Prediction by PIPN vs. the ground truth for a domain of the set $\Phi = \{V_i\}_{i=1}^{26}$



Absolute pointwise error distribution for geometries with maximum and minimum errors for the set $\Phi = \{V_i\}_{i=1}^{26}$



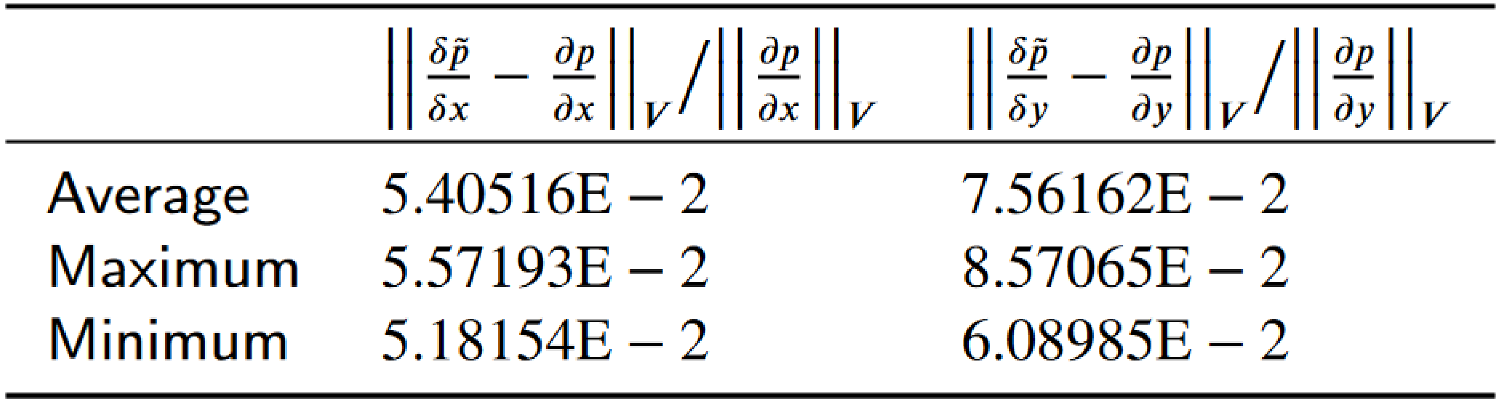
# Error analysis



| | |
|---|---|
| Average $\|\|\tilde{u}-u\|\|_V/\|\|u\|\|_V$ | 9.79855E − 3 |
| Maximum $\|\|\tilde{u}-u\|\|_V/\|\|u\|\|_V$ | 1.06049E − 2 |
| Minimum $\|\|\tilde{u}-u\|\|_V/\|\|u\|\|_V$ | 9.11389E − 3 |
| Average $\|\|\tilde{v}-v\|\|_V/\|\|v\|\|_V$ | 9.55093E − 3 |
| Maximum $\|\|\tilde{v}-v\|\|_V/\|\|v\|\|_V$ | 1.03773E − 2 |
| Minimum $\|\|\tilde{v}-v\|\|_V/\|\|v\|\|_V$ | 8.89982E − 3 |
| Average $\|\|\tilde{p}-p\|\|_V/\|\|p\|\|_V$ | 1.27507E − 2 |
| Maximum $\|\|\tilde{p}-p\|\|_V/\|\|p\|\|_V$ | 1.35610E − 2 |
| Minimum $\|\|\tilde{p}-p\|\|_V/\|\|p\|\|_V$ | 1.19376E − 2 |

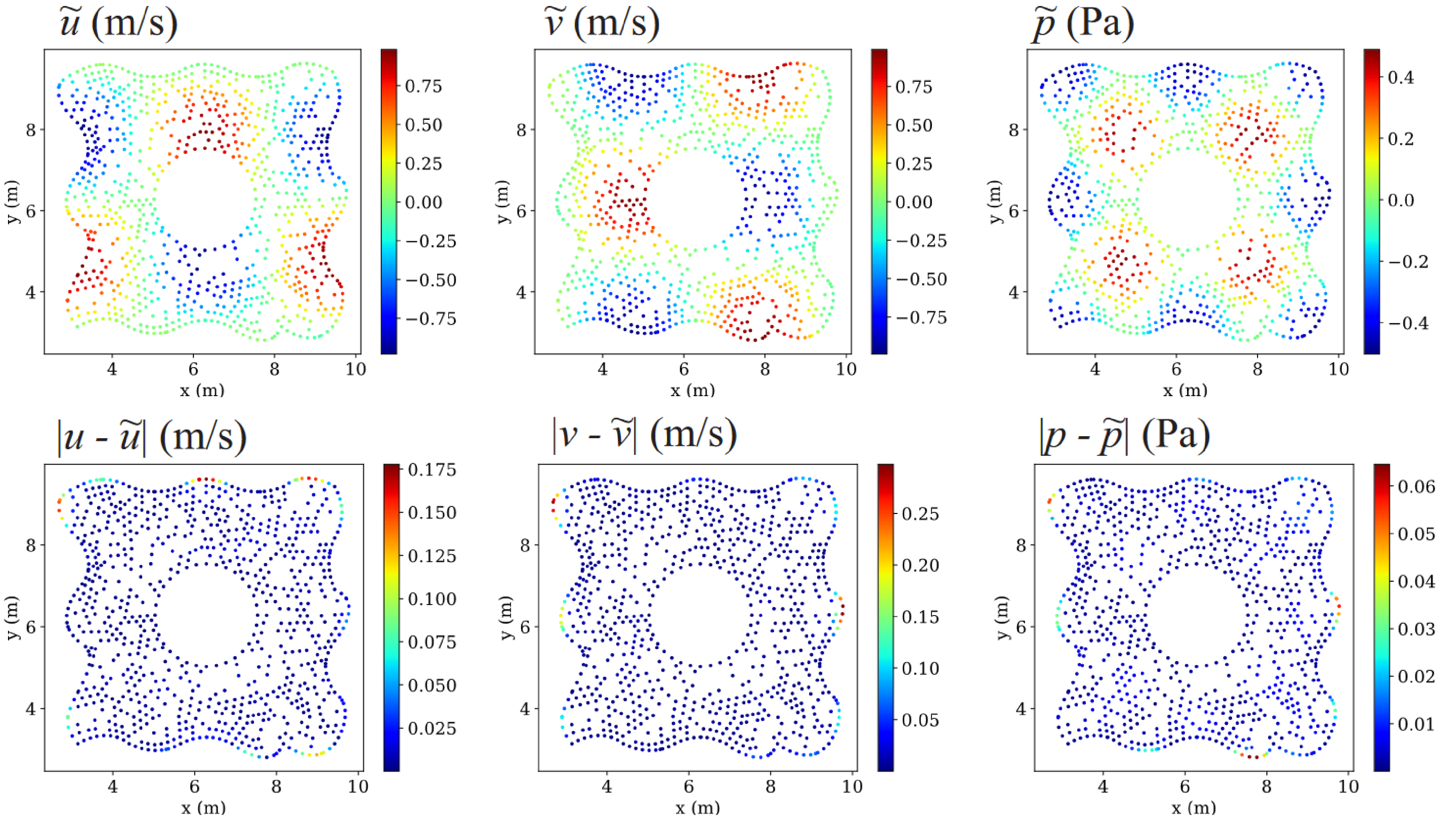
# Error analysis

The effect of removing the Dirichlet pressure boundary conditions from the loss function; PIPN preservers the pressure gradient.



| | $\left\|\frac{\delta\tilde{p}}{\delta x}-\frac{\partial p}{\partial x}\right\|_V \Big/ \left\|\frac{\partial p}{\partial x}\right\|_V$ | $\left\|\frac{\delta\tilde{p}}{\delta y}-\frac{\partial p}{\partial y}\right\|_V \Big/ \left\|\frac{\partial p}{\partial y}\right\|_V$ |
|---|---|---|
| Average | 5.40516E – 2 | 7.56162E – 2 |
| Maximum | 5.57193E – 2 | 8.57065E – 2 |
| Minimum | 5.18154E – 2 | 6.08985E – 2 |

# Prediction by PIPN and the absolute error for a domain of the set $\Psi = \{V_i\}_{i=1}^3$


$\tilde{u}$ (m/s)
$\tilde{v}$ (m/s)
$\tilde{p}$ (Pa)
$|u - \tilde{u}|$ (m/s)
$|v - \tilde{v}|$ (m/s)
$|p - \tilde{p}|$ (Pa)
x (m)
y (m)
4
6
8
10
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
0.4
0.2
0.0
−0.2
−0.4
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.25
0.20
0.15
0.10
0.05
0.06
0.05
0.04
0.03
0.02
0.01