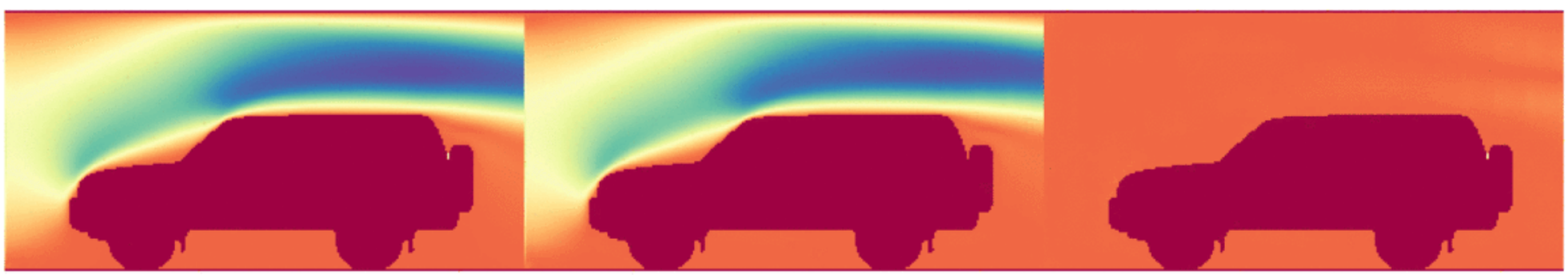
Uncertainty Quantification Group
at the University of Illinois at Urbana-Champaign

University of Illinois at Urbana-Champaign
Department of Civil and Environmental Engineering
UIUC UQ Group Presentation
Fall 2018

Physics-Informed Regularization of Deep Neural Networks

MOHAMMAD AMIN NABIAN

PhD Candidate
Sustainable and Resilient Infrastructure Systems Program



Contents

- Regression using deep neural networks
- Regularization of deep neural networks
- Physics-Informed (PI) regularization
- Numerical examples

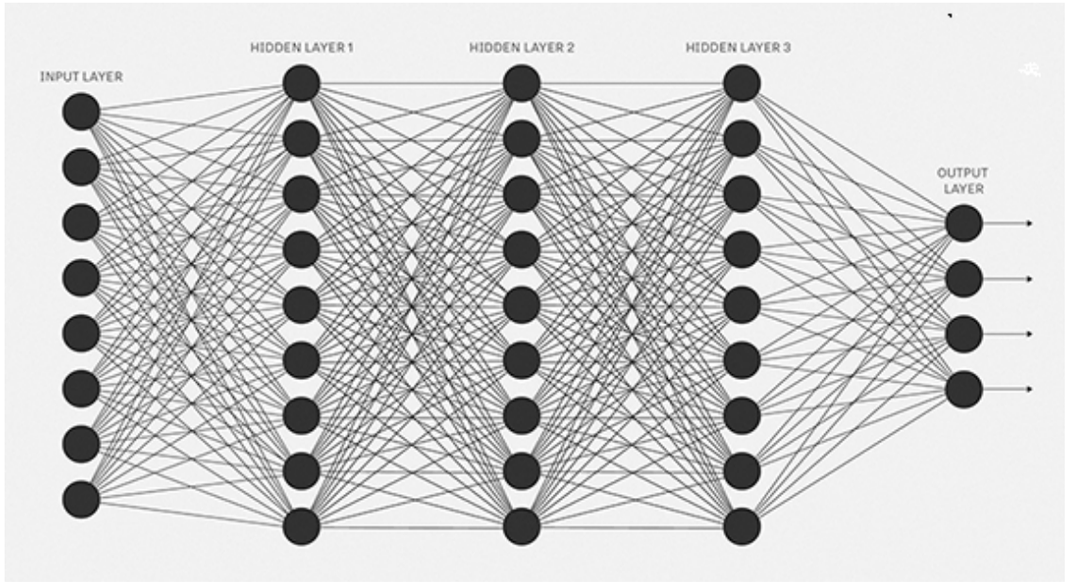
Regression Using Deep Neural Networks

Let us consider the following relationship holds:

$$\boldsymbol{y}_i = u\left(\boldsymbol{x}_i\right) + \epsilon_i$$

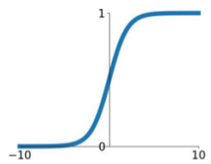
Our objective is to approximate this function by a deep neural network:

$$\boldsymbol{y}_i \approx \tilde{u}\left(\boldsymbol{x}_i; \boldsymbol{\Theta}\right) = \sigma(\boldsymbol{x}_i \boldsymbol{W}_1 + \boldsymbol{b}_1)\boldsymbol{W}_2 + \boldsymbol{b}_2$$

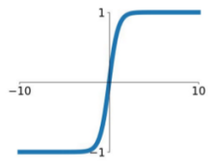


Activation Functions

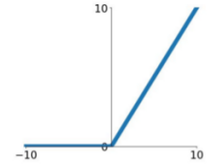
Sigmoid
$\sigma(x) = \frac{1}{1+e^{-x}}$



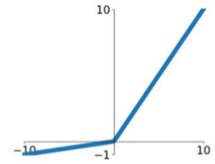
tanh
$\tanh(x)$



ReLU
$\max(0, x)$

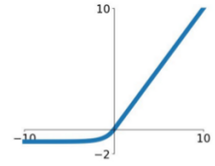


Leaky ReLU
$\max(0.1x, x)$



Maxout
$\max(w_1^T x + b_1, w_2^T x + b_2)$

ELU
$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



In order to calibrate the weight matrices and biases, we may use Euclidean loss function as follows

$$J(\boldsymbol{\Theta}; \boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{2n} \sum_{i=1}^{n} \left\| \boldsymbol{y}_i - \tilde{u}\left(\boldsymbol{x}_i; \boldsymbol{\Theta}\right) \right\|^2$$

The model parameters can be calibrated according to the following optimization problem

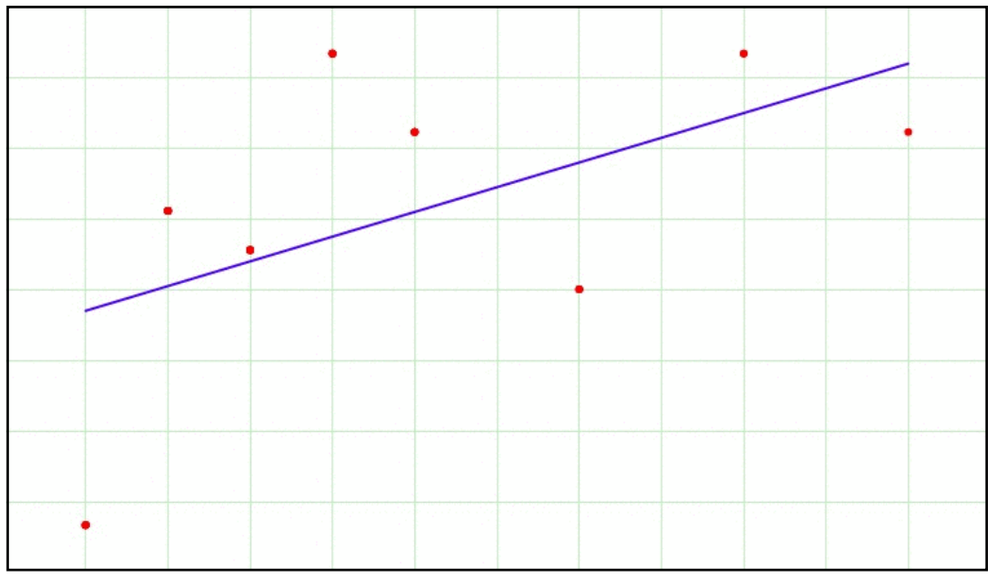
$$\boldsymbol{\Theta}^* = \underset{\boldsymbol{\Theta}}{\operatorname{argmin}}\ J(\boldsymbol{\Theta}; \boldsymbol{X}, \boldsymbol{Y})$$

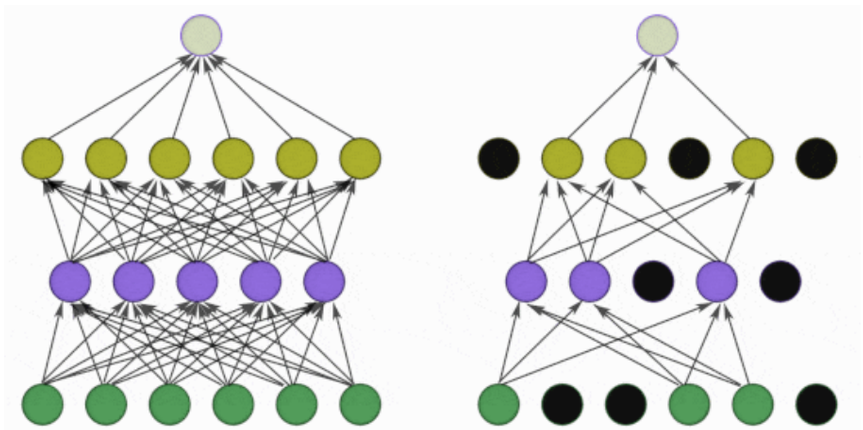
Using SGD, at the i^{th} iteration, the model parameters are updated according to

$$\boldsymbol{\Theta}^{(i+1)} = \boldsymbol{\Theta}^{(i)} - \eta^{(i)} \nabla_{\boldsymbol{\Theta}} J^{(i)}(\boldsymbol{\Theta}^{(i)}; \boldsymbol{X}, \boldsymbol{Y})$$

Regularization of Deep Neural Networks

Deep neural networks suffer from overfitting.





L1 Regularization

$$J_{L^1}(\boldsymbol{\Theta}; \boldsymbol{X}, \boldsymbol{Y}) = J(\boldsymbol{\Theta}; \boldsymbol{X}, \boldsymbol{Y}) + \lambda_1 ||\boldsymbol{W}||_1$$

L2 Regularization

$$J_{L^2}(\boldsymbol{\Theta}; \boldsymbol{X}, \boldsymbol{Y}) = J(\boldsymbol{\Theta}; \boldsymbol{X}, \boldsymbol{Y}) + \frac{\lambda_2}{2} \boldsymbol{W}^T \boldsymbol{W}$$

Dropout

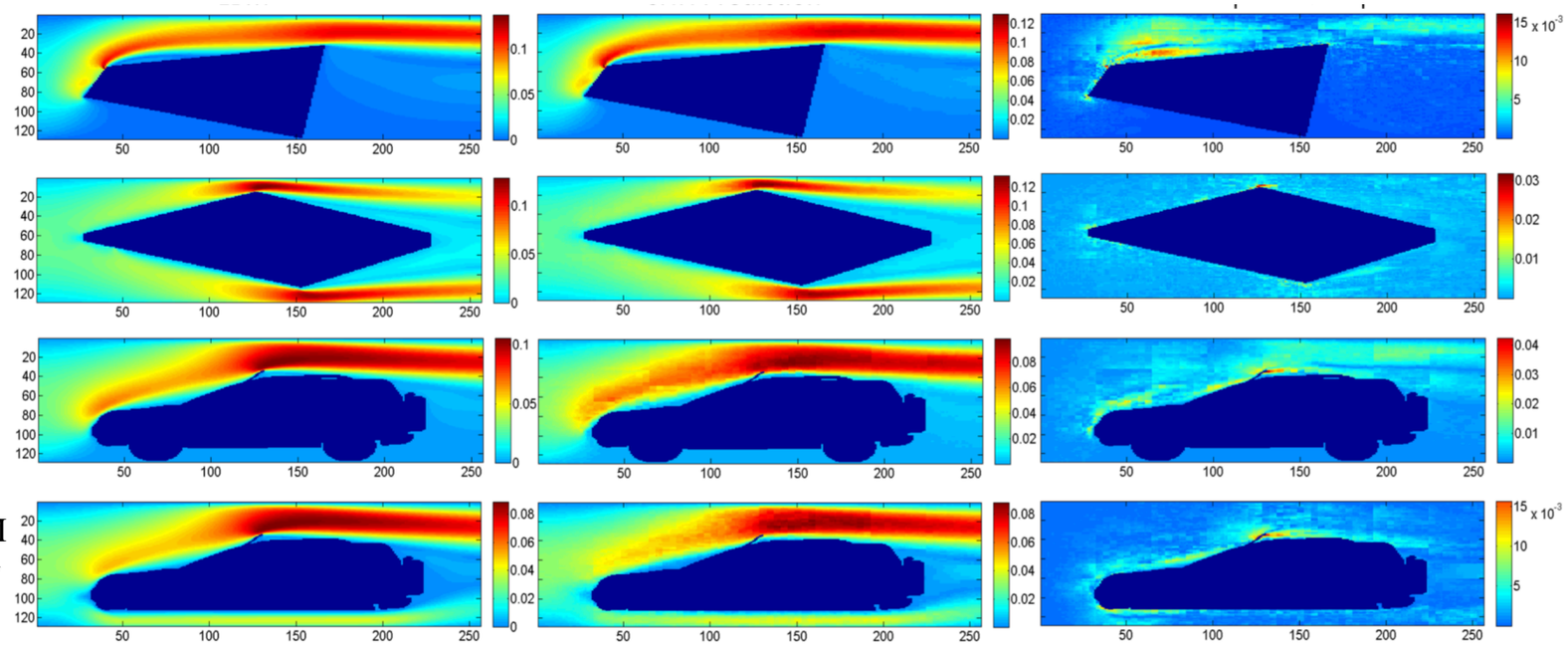
$$\boldsymbol{y} = \boldsymbol{r} \cdot \sigma(\boldsymbol{x}\boldsymbol{W}_1 + \boldsymbol{b}_1)\boldsymbol{W}_2 + \boldsymbol{b}_2$$

Physics-Informed (PI) Regularization

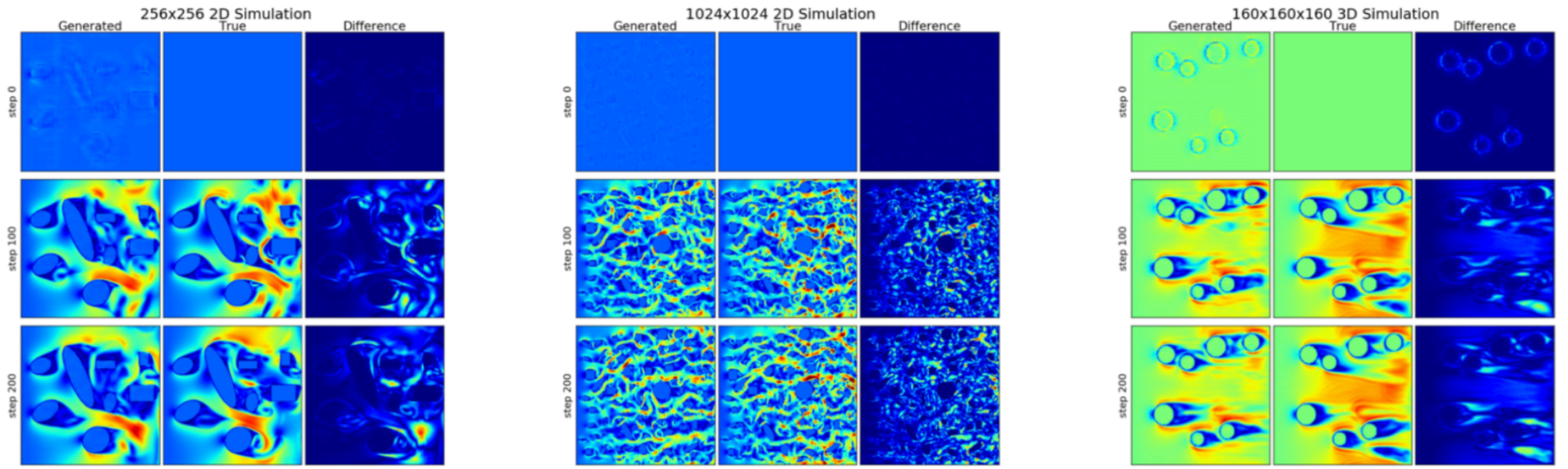
- Many science and engineering problems require repetitive simulation runs of a model with different input values.
- Examples are design optimization, model calibration, sensitivity analysis, what-if analysis, and design space exploration problems.
- However, in many real-world problems, obtaining a reliable outcome requires large number of these solves (typically for a partial differential equation), which can be prohibitive given the available resources.
- One way to alleviate this burden is to construct surrogate models that mimic the solution or response surface.

The state of the practice is to use pure data-driven regularization techniques, such as dropout, L2 and L1 regularizations.

Examples:



X. Guo, W. Li, F. Iorio, Convolutional neural networks for steady flow approximation, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016, pp. 481–490.



O. Hennigh, Lat-net: Compressing lattice boltzmann flow simulations using deep neural net- works, arXiv preprint arXiv: 1705.09036.

- We presents a novel physics-informed approach for the regularization of deep neural network surrogates for systems that are subject to known governing laws which are in the form of a PDE.

- These governing equations are obtained using
  - First principles
  - Empirically-validated laws
  - Knowledge obtained by domain expertise

- In data-driven modeling of physical and biological systems, this prior knowledge is usually available, but not directly used in training of the models.

We consider cases where the response function a governing law, as follows

$$\mathcal{L}(\boldsymbol{x}_i, u(\boldsymbol{x}_i)) = 0, \quad \boldsymbol{x}_i \in \mathbb{R}^d$$

The PI-regularized loss function is then defined as follows

$$J_{\text{PI}}(\boldsymbol{\Theta}; \boldsymbol{X}, \boldsymbol{Y}, \mathcal{L}) = J(\boldsymbol{\Theta}; \boldsymbol{X}, \boldsymbol{Y}) + \lambda_{\mathcal{L}} J_{\mathcal{L}}(\boldsymbol{\Theta}; \boldsymbol{X})$$

the physics-informed penalty term is defined as

$$J_{\mathcal{L}}(\boldsymbol{\Theta}; \boldsymbol{X}) = \frac{1}{2n} \sum_{i=1}^{n} \left[\mathcal{L}(\boldsymbol{x}_i, \tilde{u}(\boldsymbol{x}_i; \boldsymbol{\Theta}))\right]^2$$

By Adding the PI-regularization term to the standard loss function, the standard supervised learning task is converted to a **semi-supervised learning** task:

- The **supervised objective** minimizes the mean squared differences between model prediction and measurements.
- The **unsupervised objective** minimizes divergence from the governing laws.

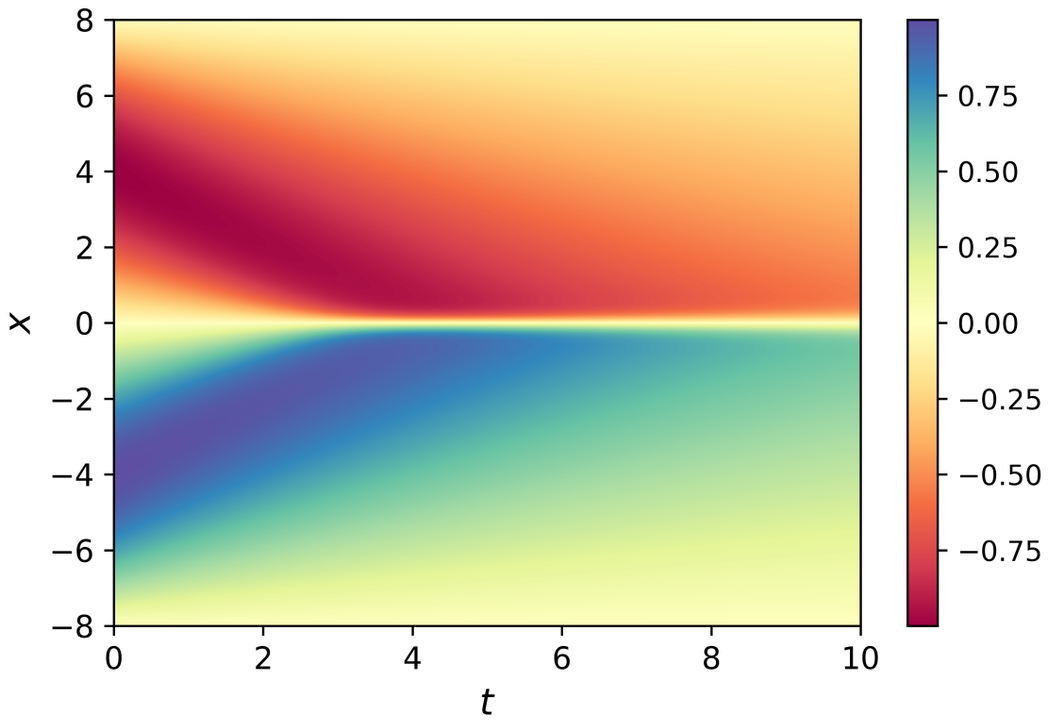
The proposed PI regularization method effectively:

- Prevents deep neural networks from overfitting
- Results in surrogates that are better physically interpretable.

Example 1 : Burger's equation

Let us start with the Burgers' equation, which arises in various areas of engineering, such as traffic flow, fluid mechanics :

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} - 0.1\frac{\partial^2 u}{\partial x^2} = 0, \qquad x \in [-8, 8],\ t \in [0, 10],$$
$$u(0, x) = -\sin(\pi x/8),\ u(t, -8) = 0,\ u(t, 8) = 0.$$



500 samples for training

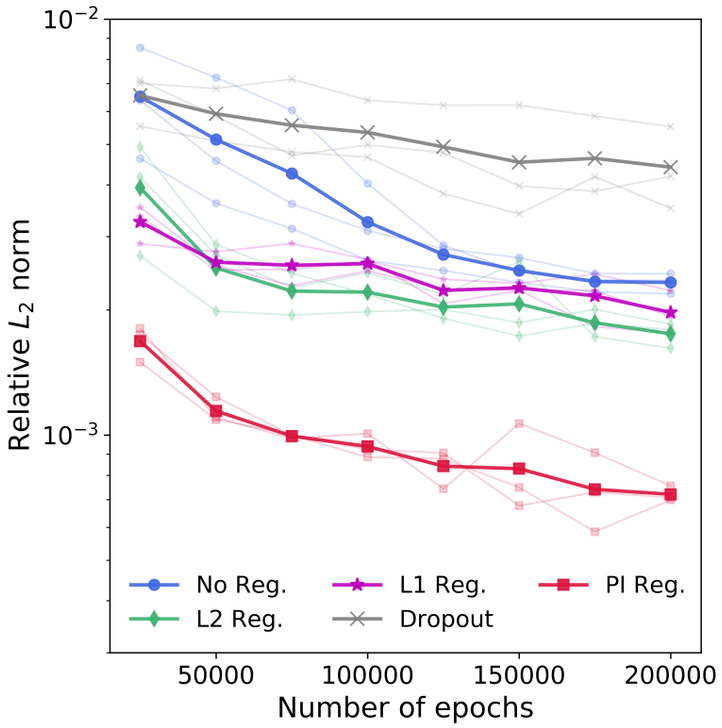
5,000 samples for evaluation

5,000 samples for testing

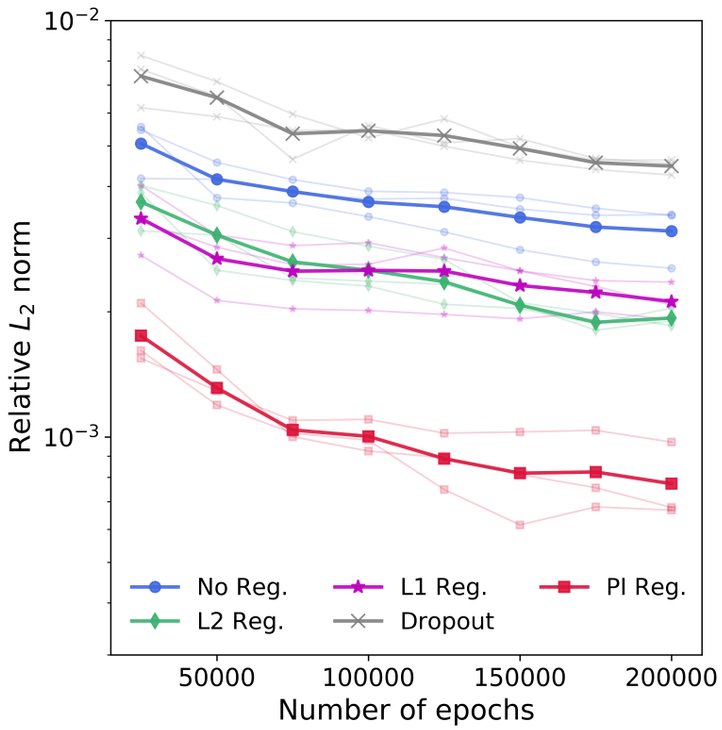
PI regularizer tem:

$$J_{\mathcal{L}}(\boldsymbol{\Theta}; \boldsymbol{X}) = \frac{1}{2n}\sum_{i=1}^{n}\left[\frac{\partial \tilde{u}}{\partial t}(\boldsymbol{X}; \boldsymbol{\Theta}) + \tilde{u}(\boldsymbol{X}; \boldsymbol{\Theta})\frac{\partial \tilde{u}}{\partial x}(\boldsymbol{X}; \boldsymbol{\Theta}) - 0.1\frac{\partial^2 \tilde{u}}{\partial x^2}(\boldsymbol{X}; \boldsymbol{\Theta})\right]^2$$

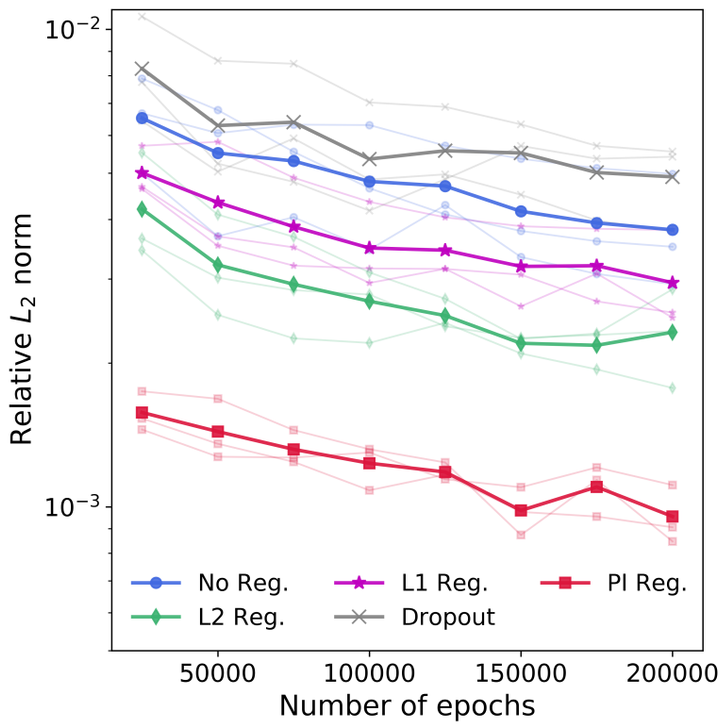
Comparison between the performance of different regularization methods



$\gamma = 0$

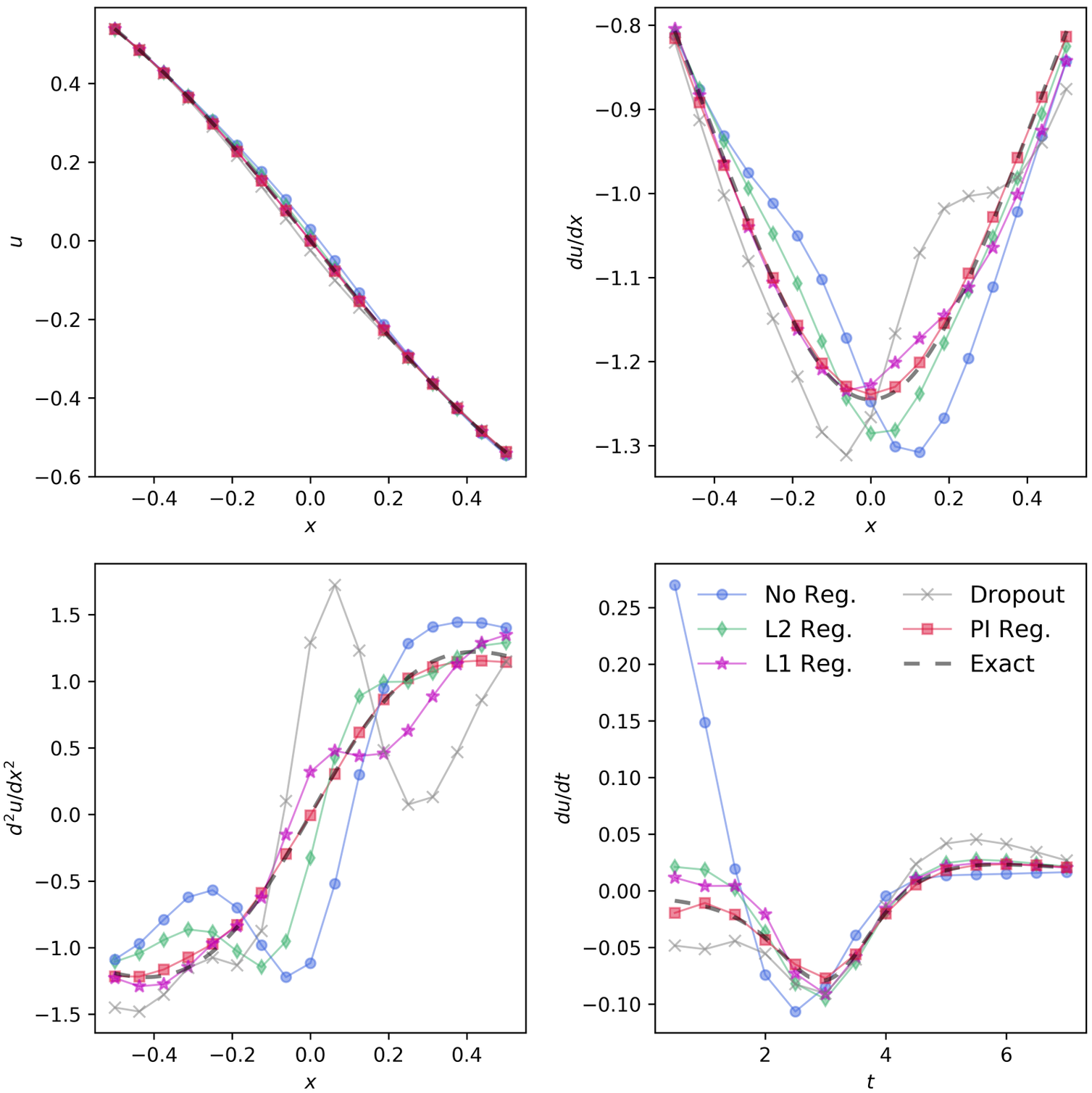


$\gamma = 0.15$



$\gamma = 0.25$

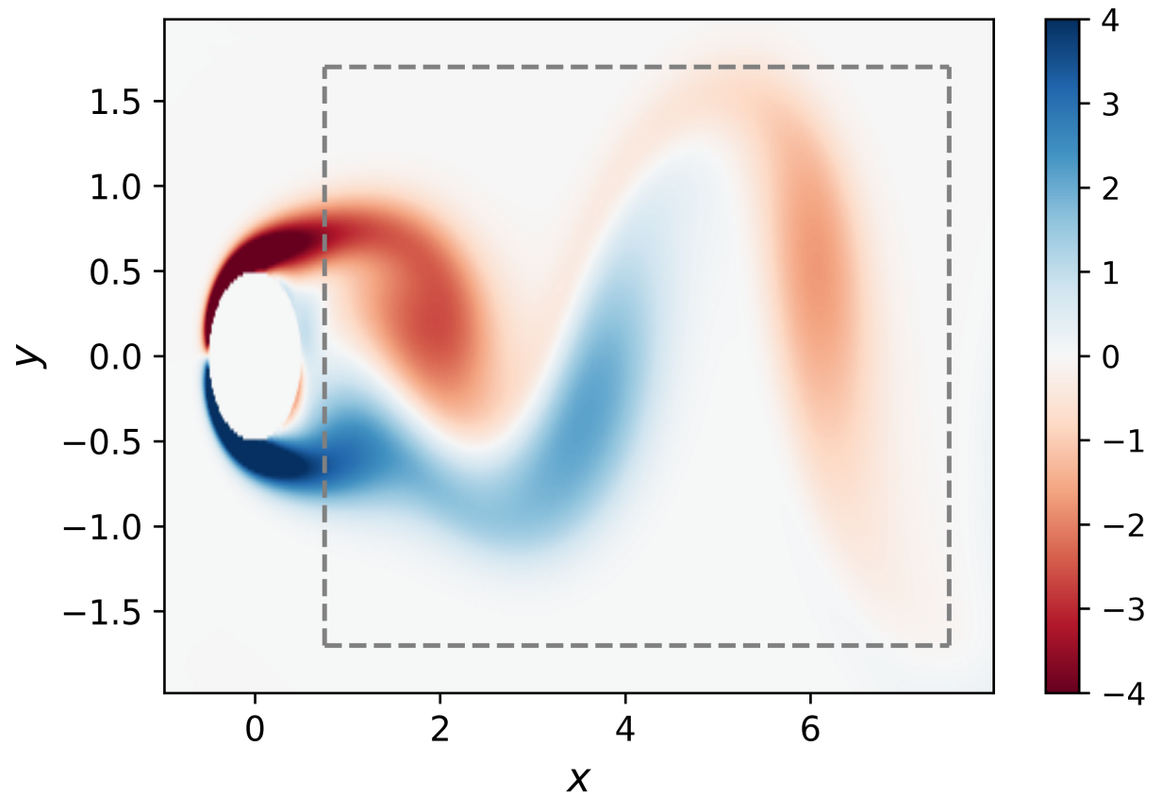
Comparison between the performance of different regularization methods in accurate prediction of first- and second-order derivatives



Example 2 : Navier-Stokes Equation

In this example we consider the vorticity equation given explicitly by

$$\frac{\partial \omega}{\partial t} = -u\frac{\partial \omega}{\partial x} - v\frac{\partial \omega}{\partial y} + 0.01\left(\frac{\partial^2 \omega}{\partial x^2} + \frac{\partial^2 \omega}{\partial y^2}\right)$$



5,000 samples for training

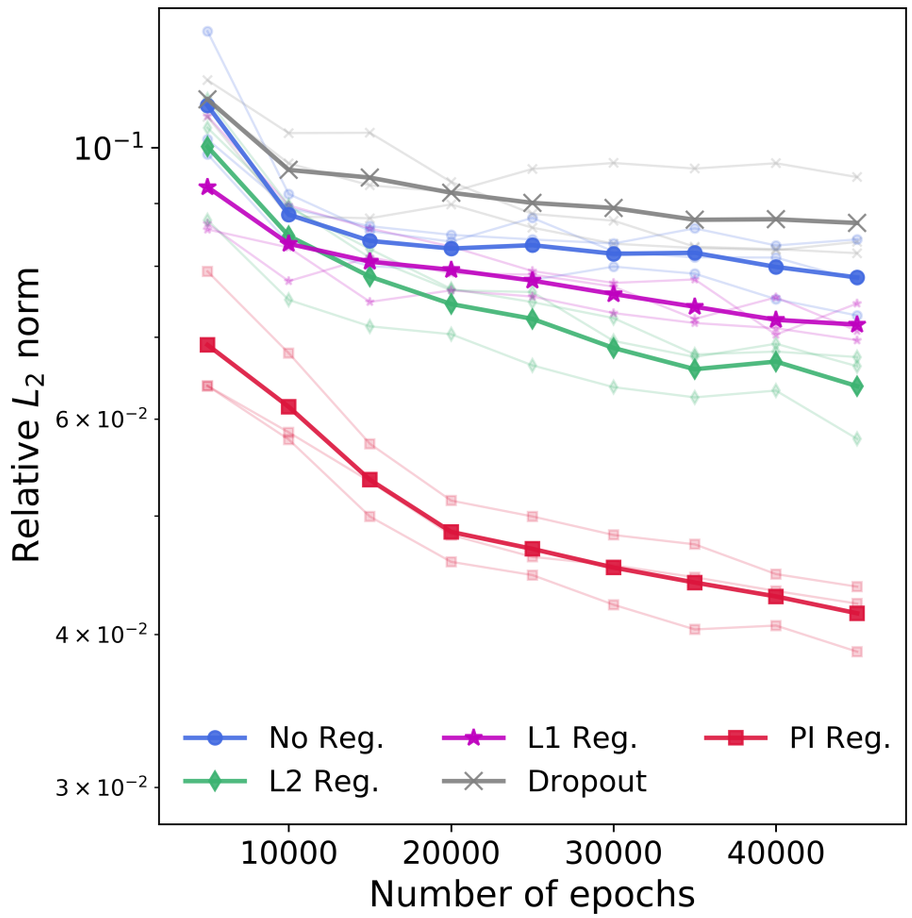
15,000 samples for evaluation

30,000 samples for testing

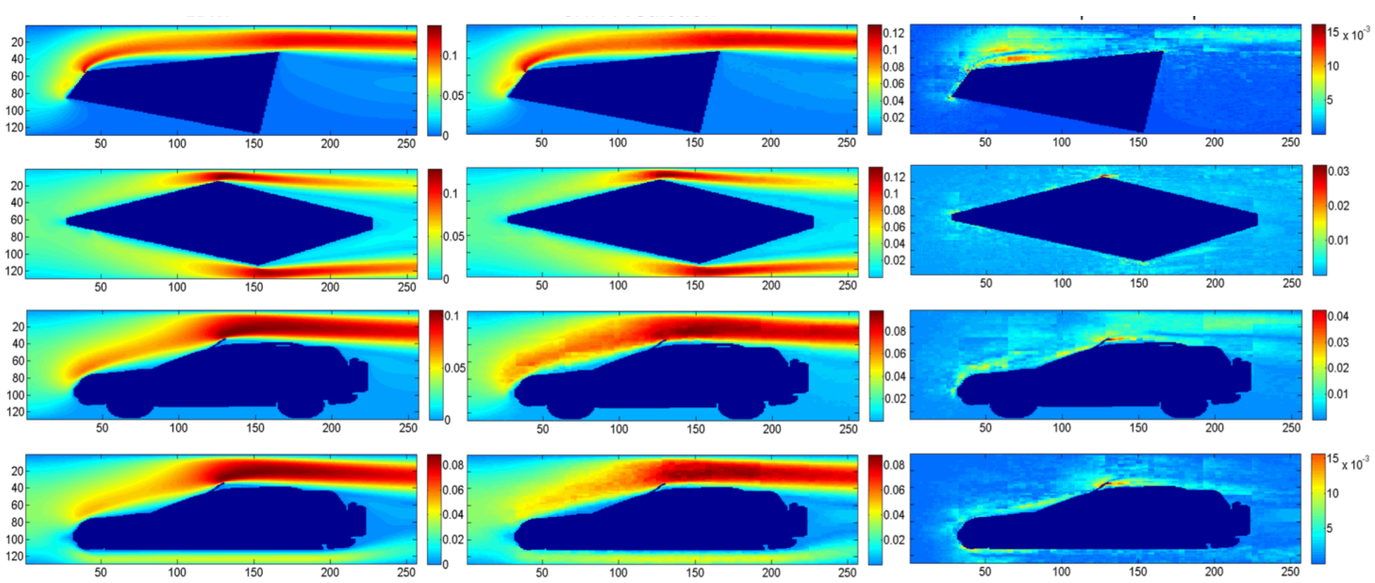
PI regularizer tem:

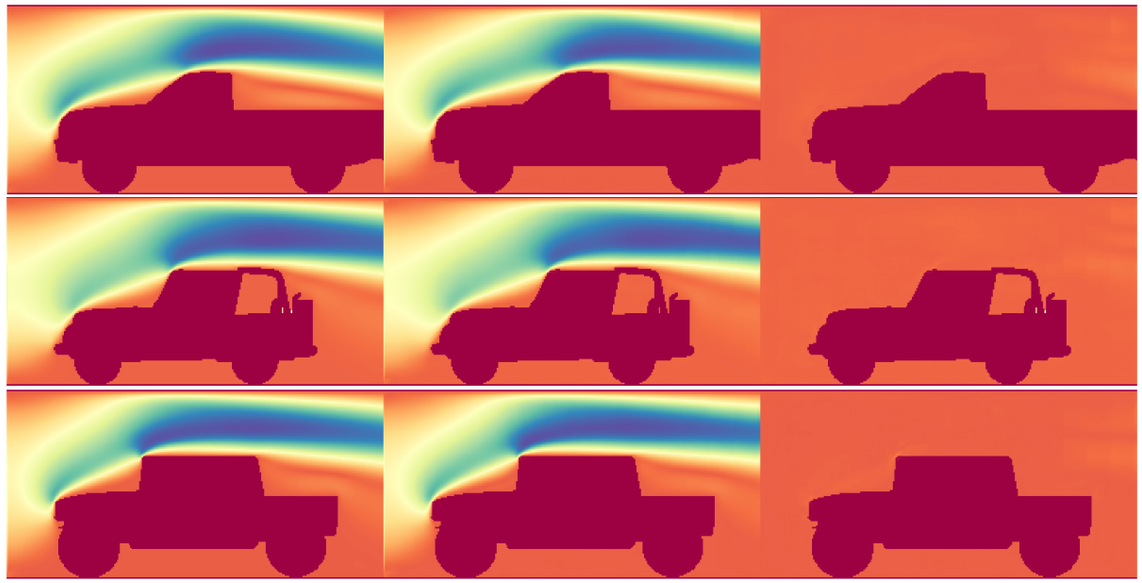
$$J_{\mathcal{L}}(\boldsymbol{\Theta}; \boldsymbol{X}) = \frac{1}{2n}\sum_{i=1}^{n}\left[\frac{\partial \tilde{\omega}}{\partial t}(\boldsymbol{X}) + \tilde{u}(\boldsymbol{X})\frac{\partial \tilde{\omega}}{\partial x}(\boldsymbol{X}) + \tilde{v}(\boldsymbol{X})\frac{\partial \tilde{\omega}}{\partial y}(\boldsymbol{X}) - 0.01\left(\frac{\partial^2 \tilde{\omega}}{\partial x^2}(\boldsymbol{X}) + \frac{\partial^2 \tilde{\omega}}{\partial y^2}(\boldsymbol{X})\right)\right]^2$$

Comparison between the performance of different regularization methods



Example 3 : Surrogate modeling in CFD-based design optimization





Regularization method	No Reg.	Dropout ($P = 0.7$)	PI Reg. ($\lambda_{\mathcal{L}} = 0.1$)	Dropout ($P = 0.7$) & PI Reg. ($\lambda_{\mathcal{L}} = 0.1$)
Relative L_2 norm	1.41×10^{-5}	1.35×10^{-5}	1.36×10^{-5}	1.15×10^{-5}

Thank you!