A brief introduction of deep learning algorithms applied to mechanics

Hanfeng Zhai

Department of Mechanics, Shanghai University

www.hanfengzhai.net

April 20, 2021

Machine Learning & Neural Networks

Machine learning (ML) is the study of computer algorithms that improve automatically through experience and by the use of data.

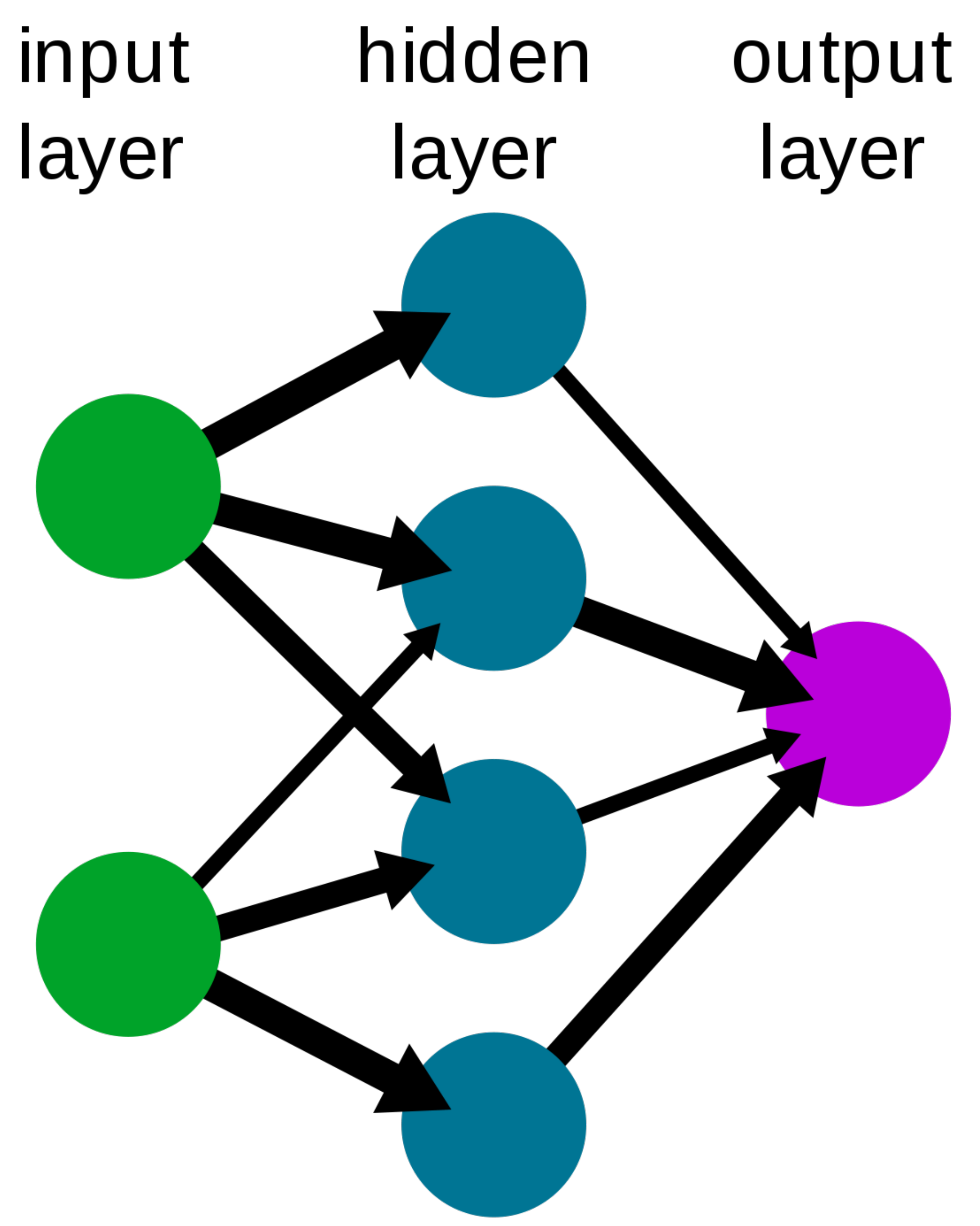
Mitchell, T. 1997

A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates.

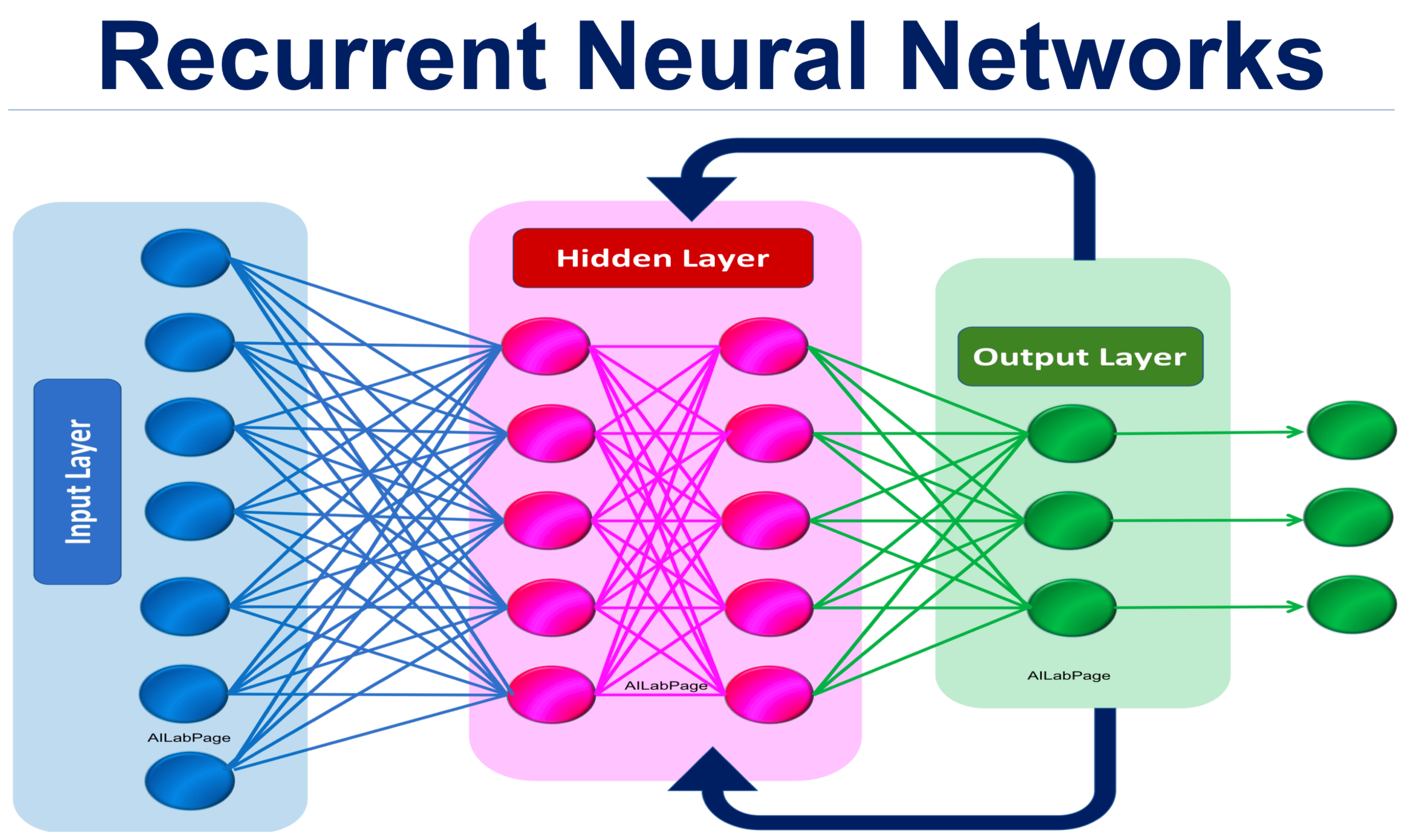
Chen, J. 2020

Machine Learning & Neural Networks

A simple neural network

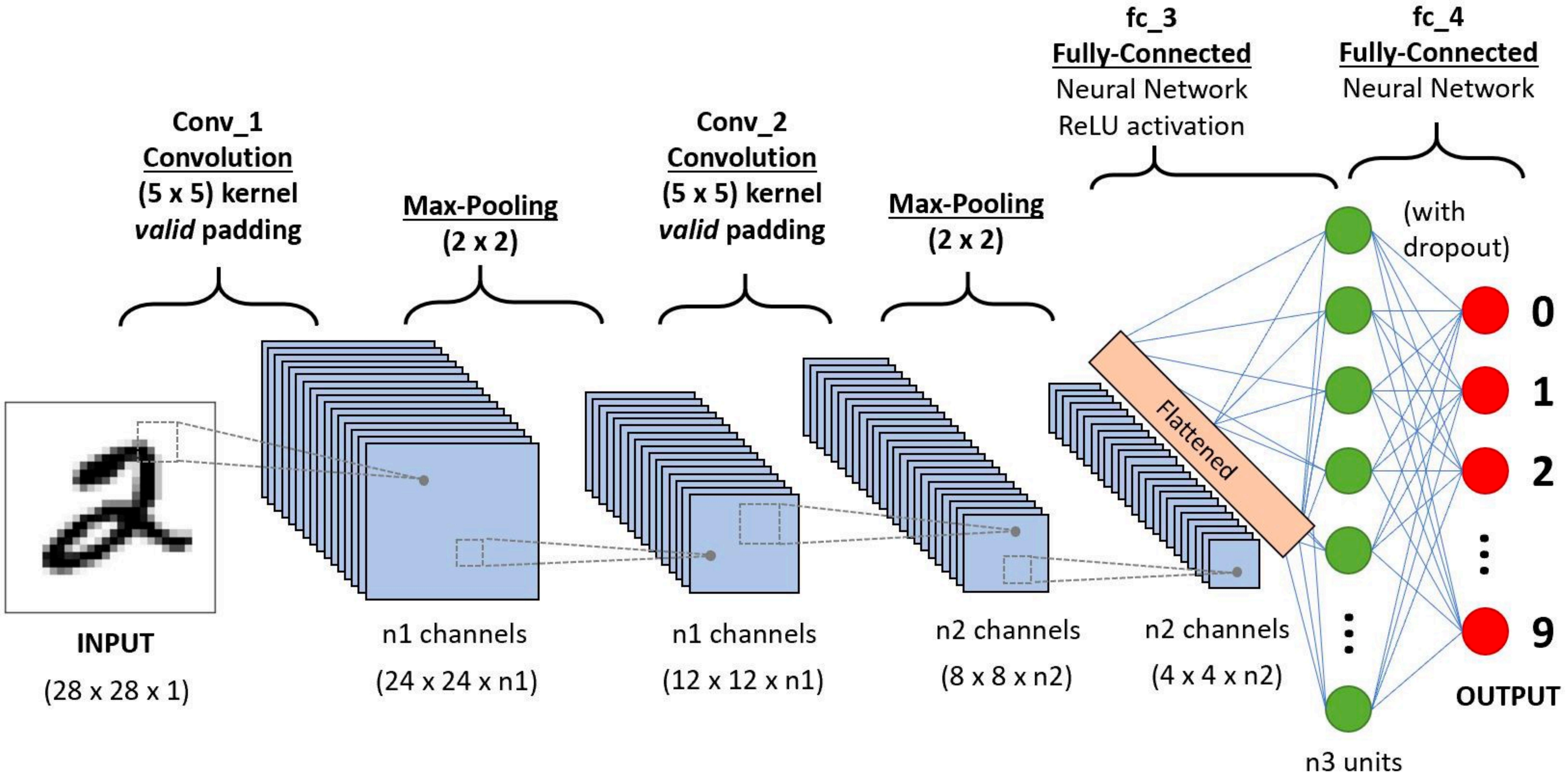


https://en.wikipedia.org/wiki/Neural_network



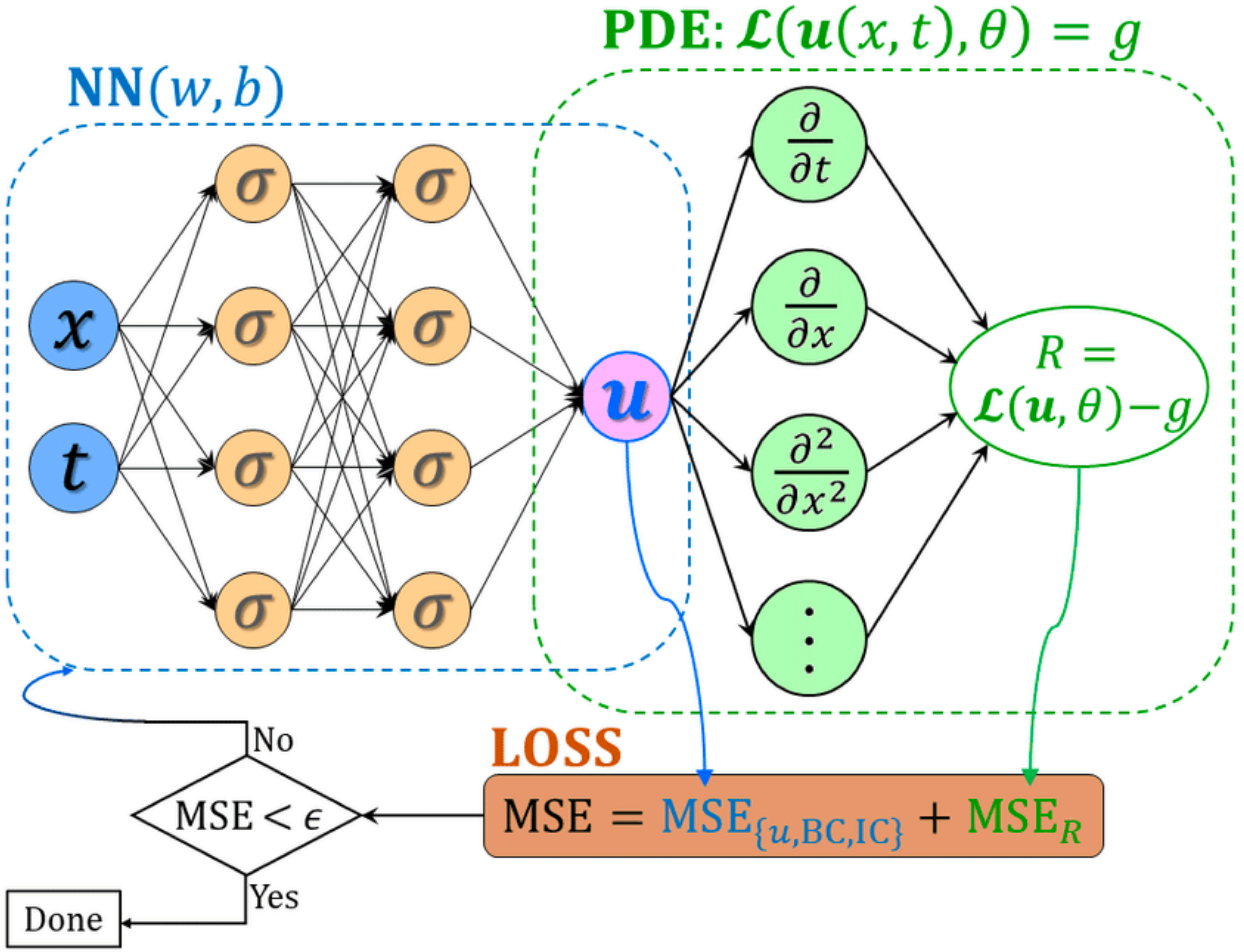
https://ailabpage.com/2019/01/08/deep-learning-introduction-to-recurrent-neural-networks/

Machine Learning & Neural Networks



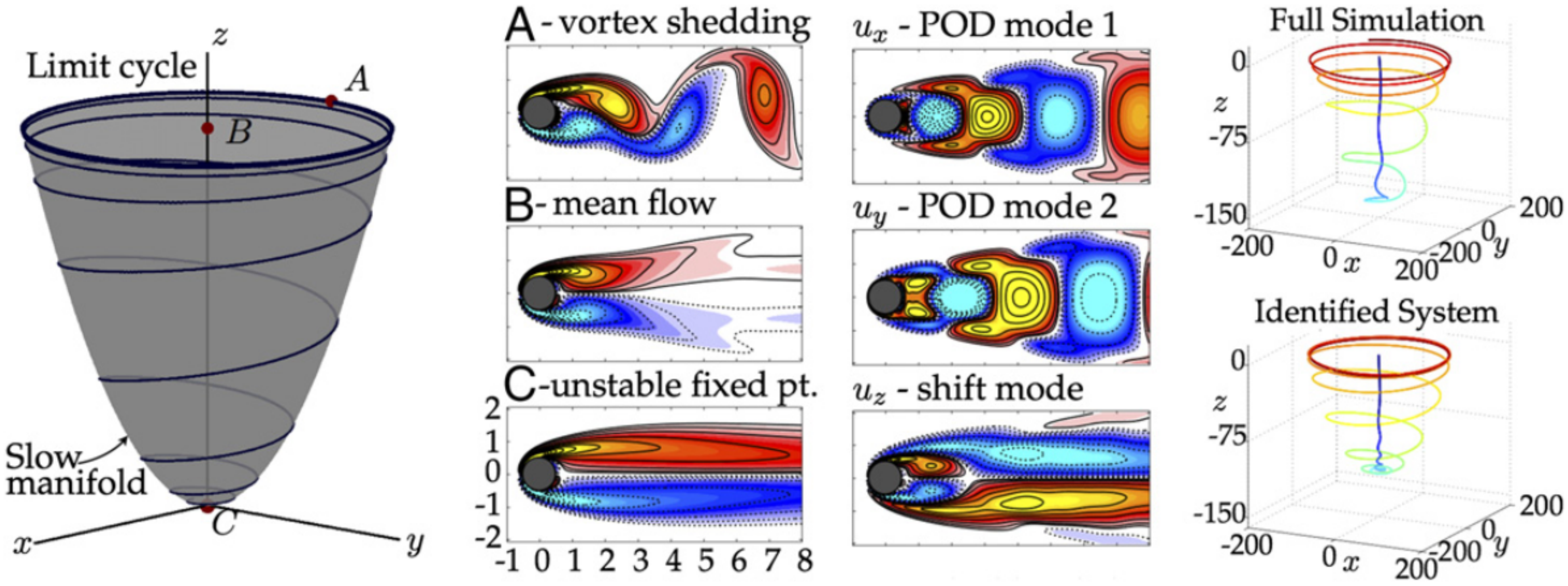
https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

Physics-Informed Neural Network



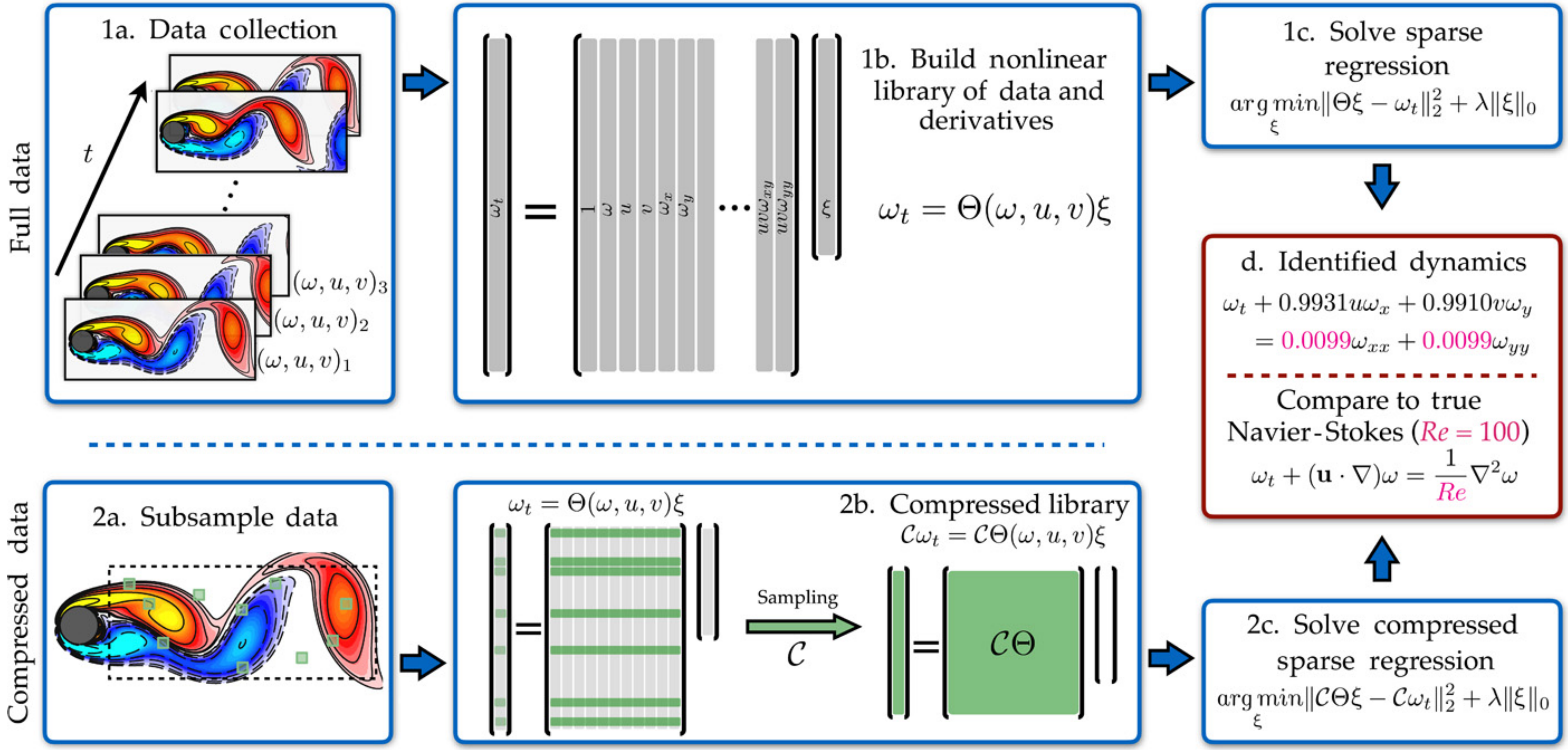
Meng *et al.*, *Comp. Meth. App. Mech. Eng.*, 2020

Machine Learning for Physics



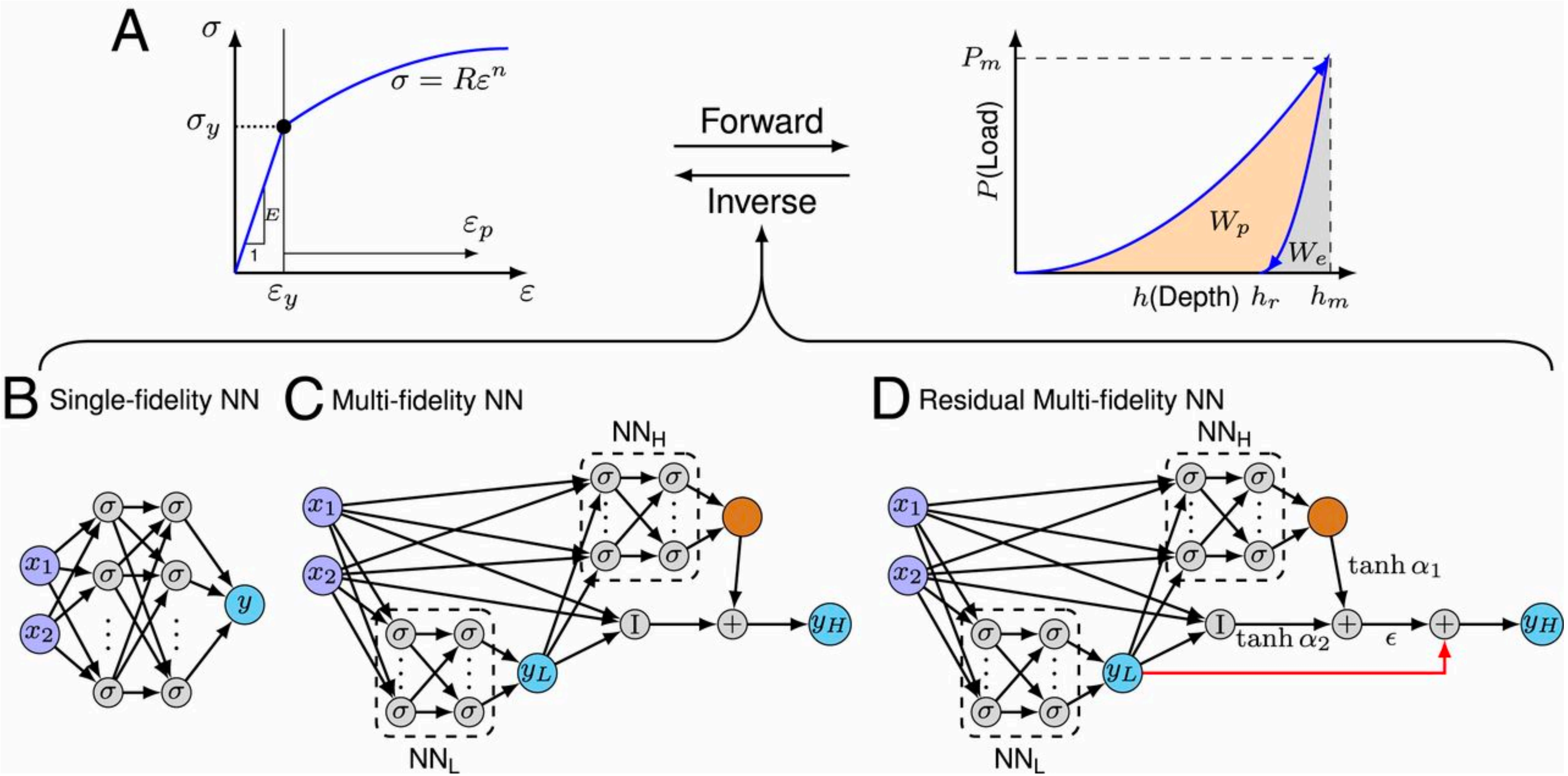
Brunton *et al.*, *PNAS*, 2016

Machine Learning for Physics



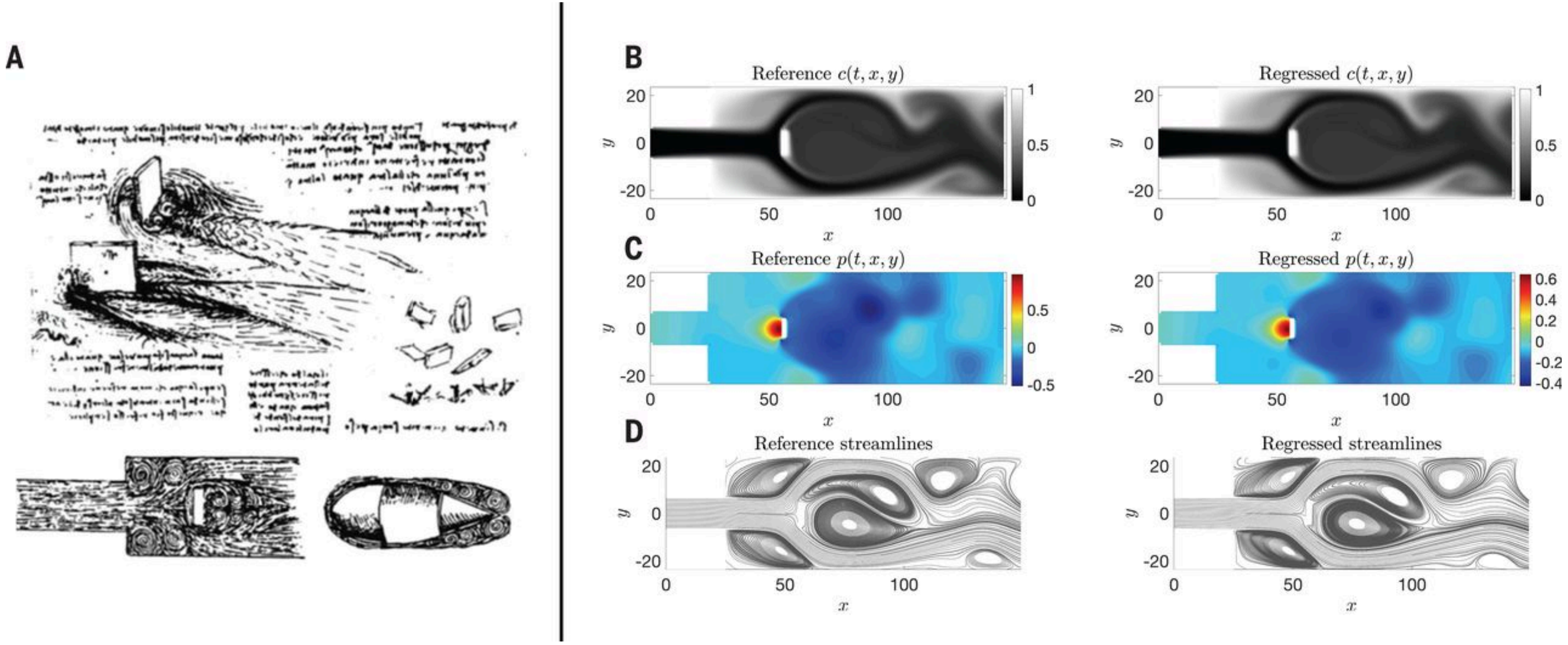
Rudy *et al*., *Sci. Adv.*, 2017

Deep Learning for Physics



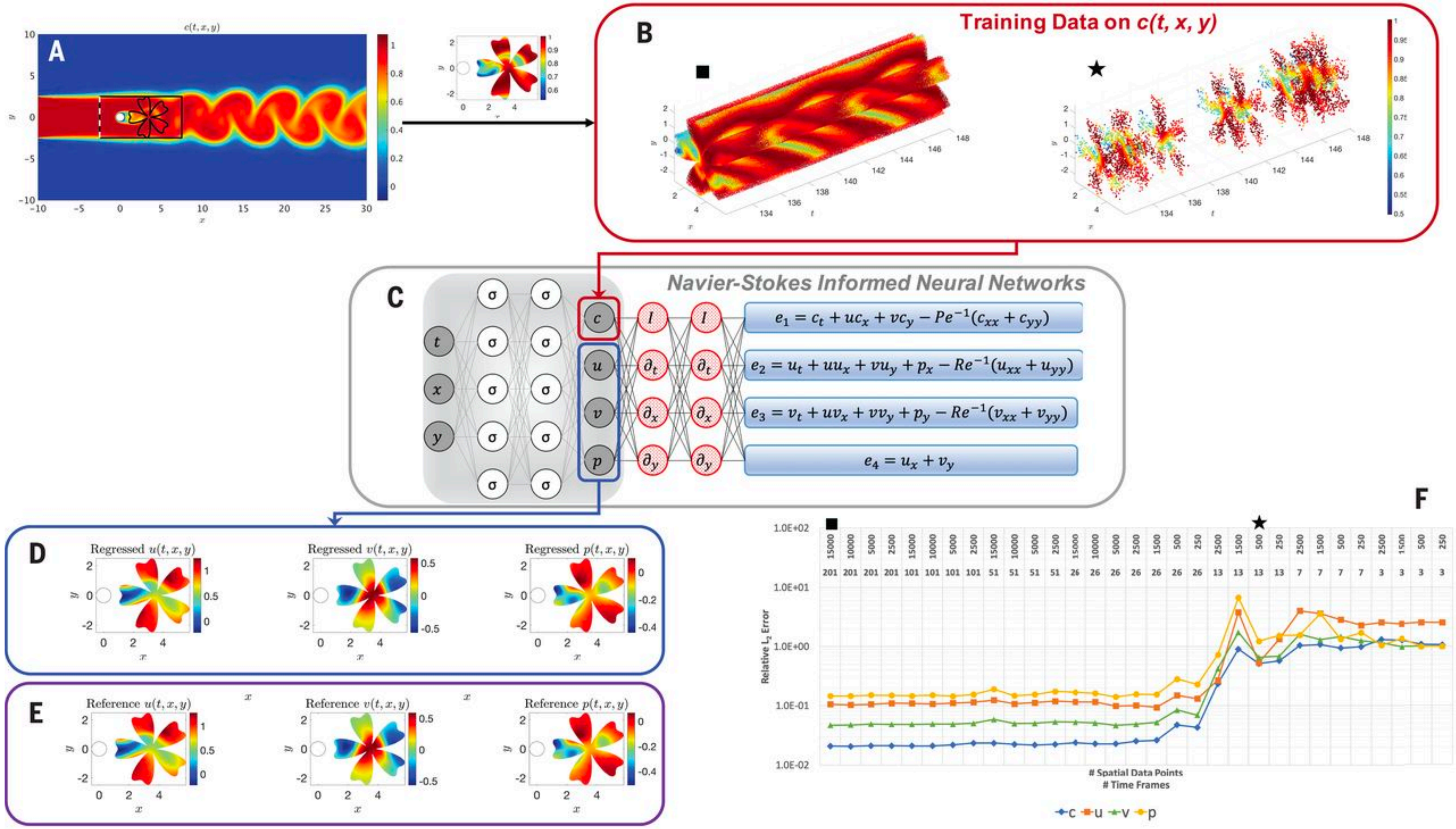
Lu *et al*., *PNAS*, 2020

Deep Learning for Physics



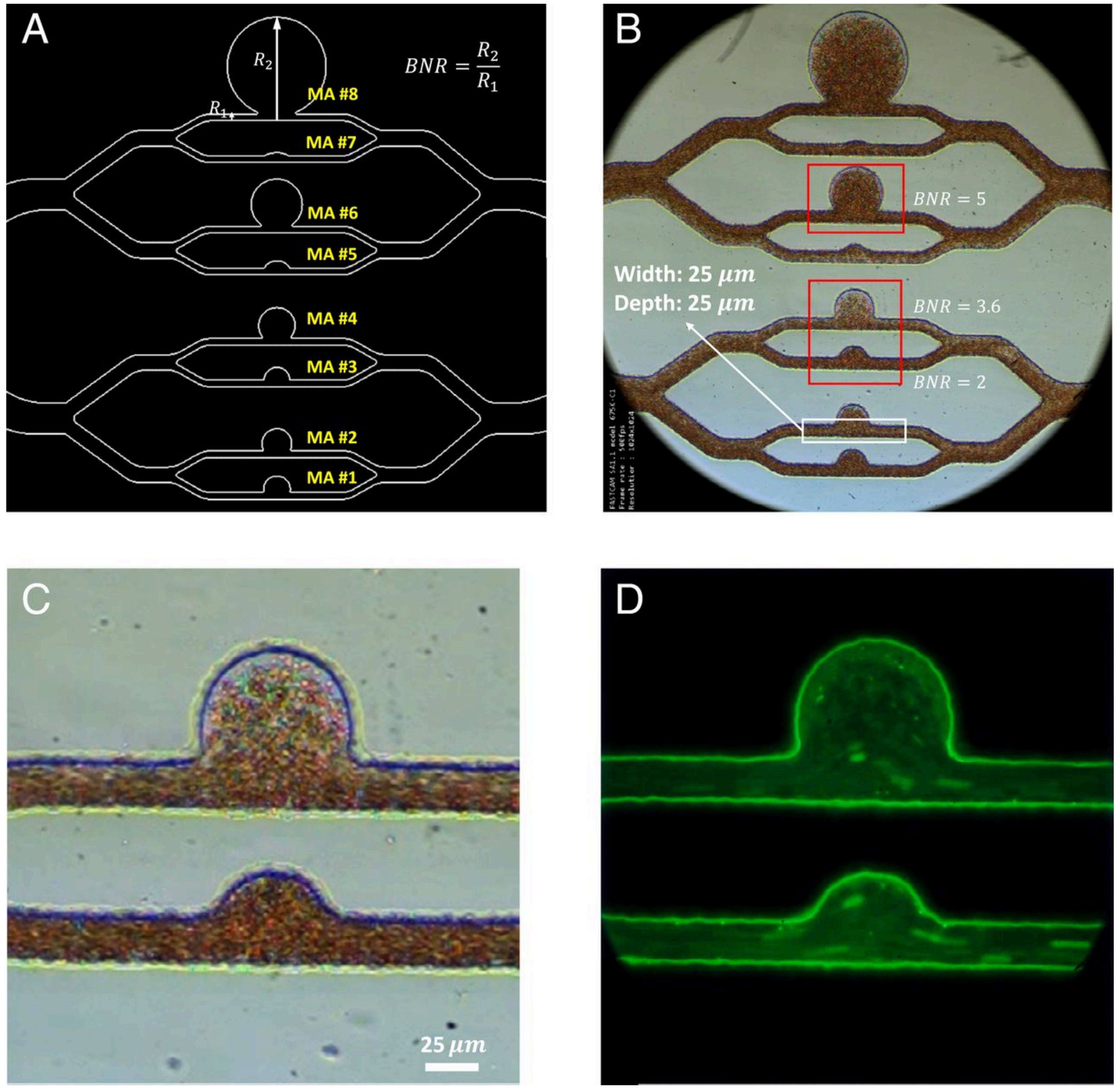
Raissi *et al.*, *Science*, 2020

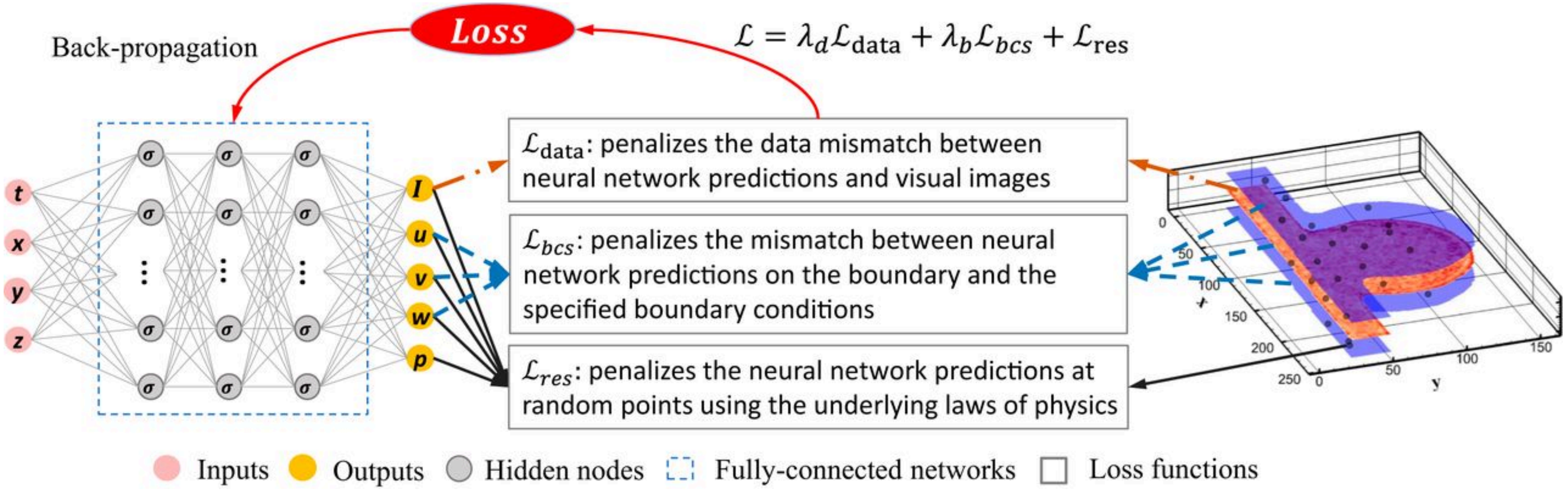
Deep Learning for Physics



Raissi *et al.*, *Science*, 2020

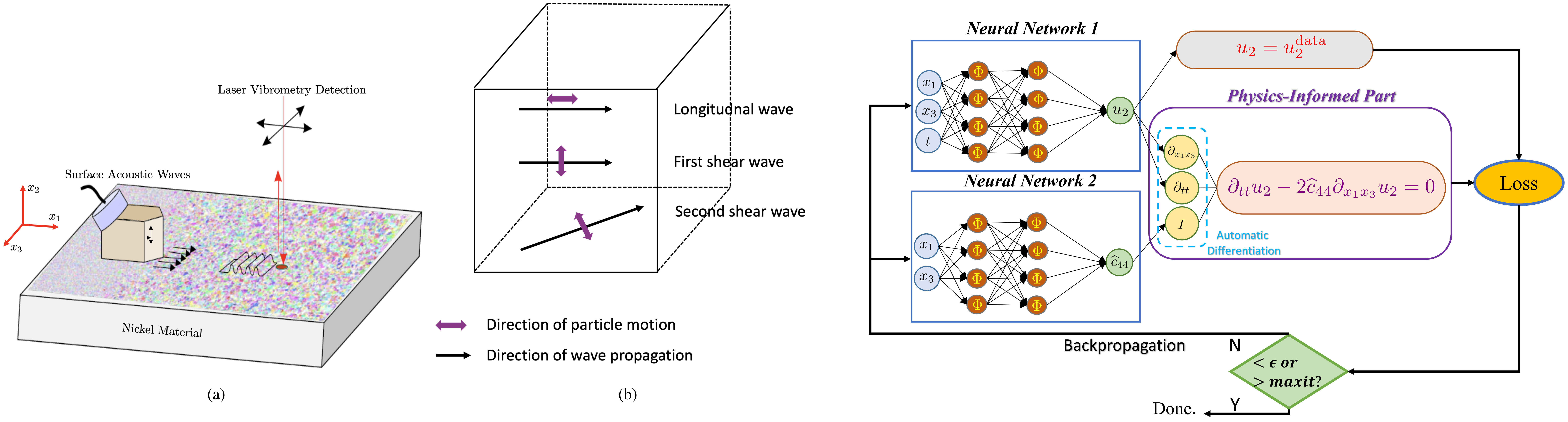
Deep Learning for Physics





Cai *et al.*, *PNAS*, 2021

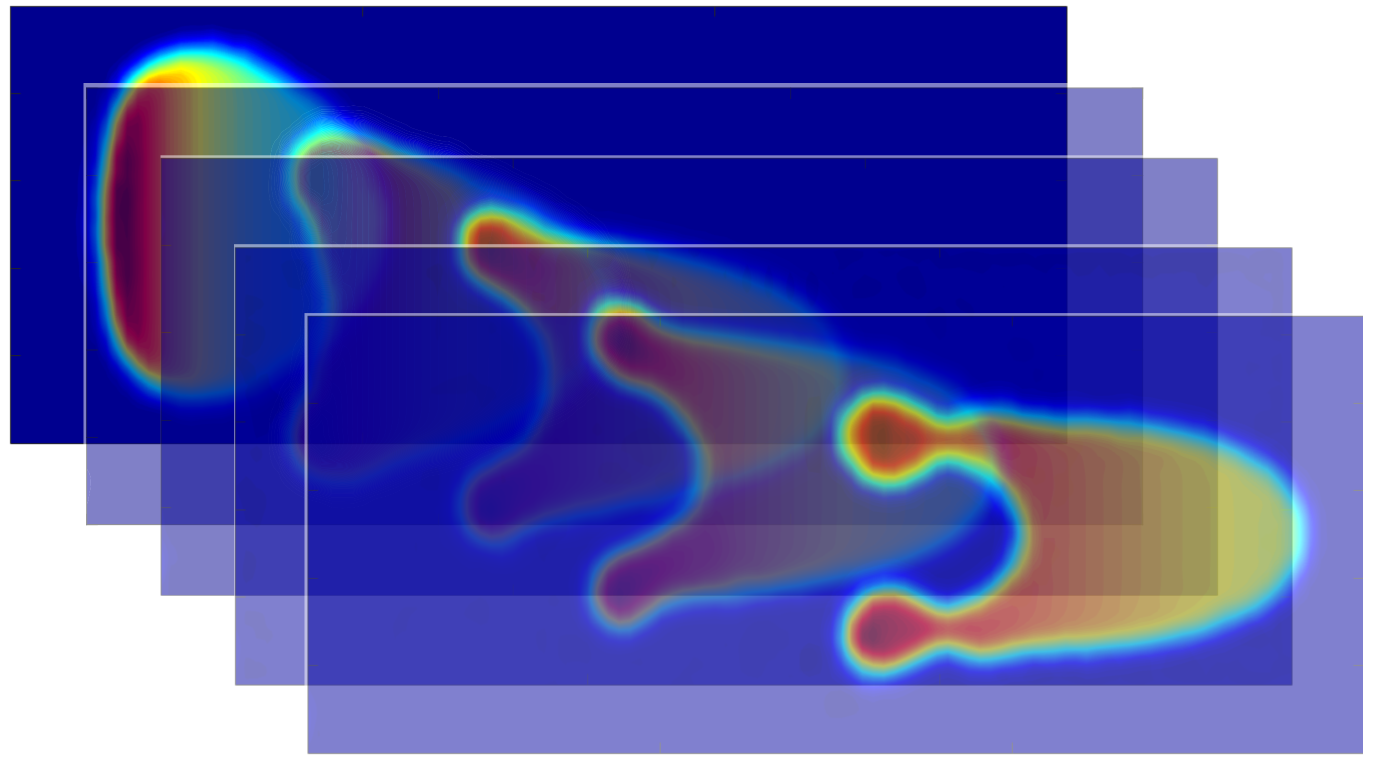
Deep Learning for Physics



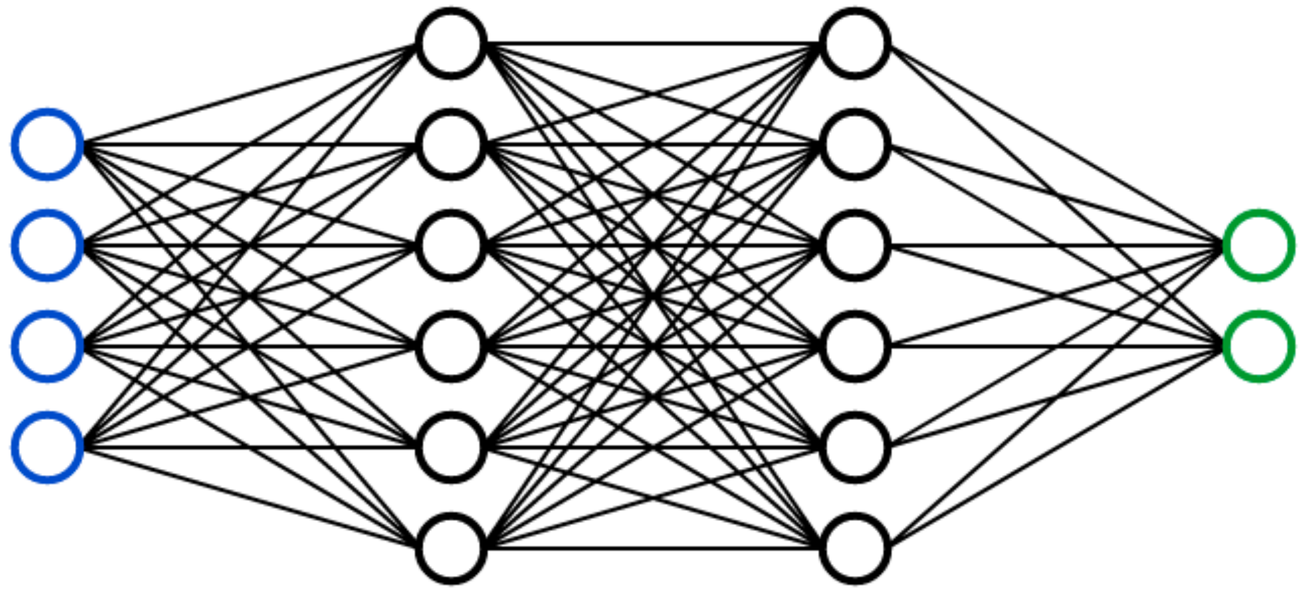
Shukla *et al.*, *Preprint*, 2021

Data-driven inference of micro-bubble dynamics with physics-informed deep learning

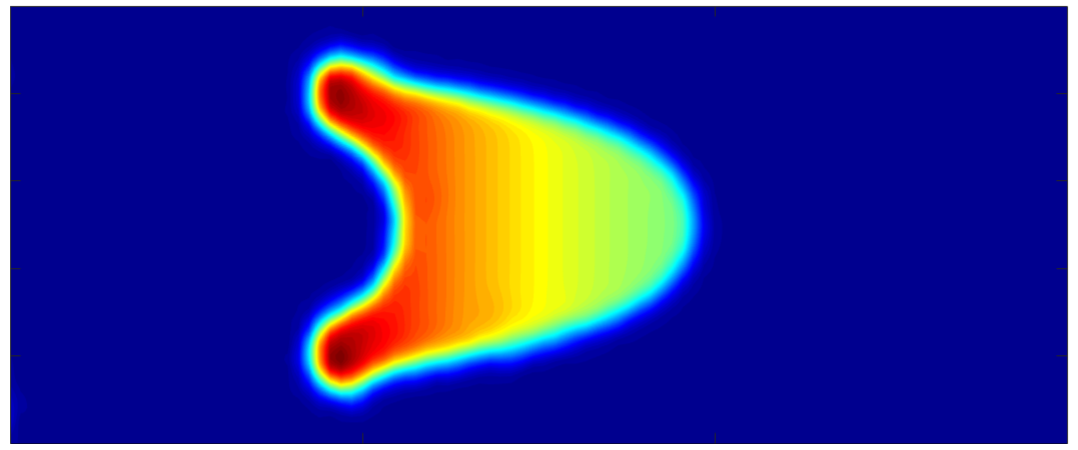
Bubble dynamics



Deep learning



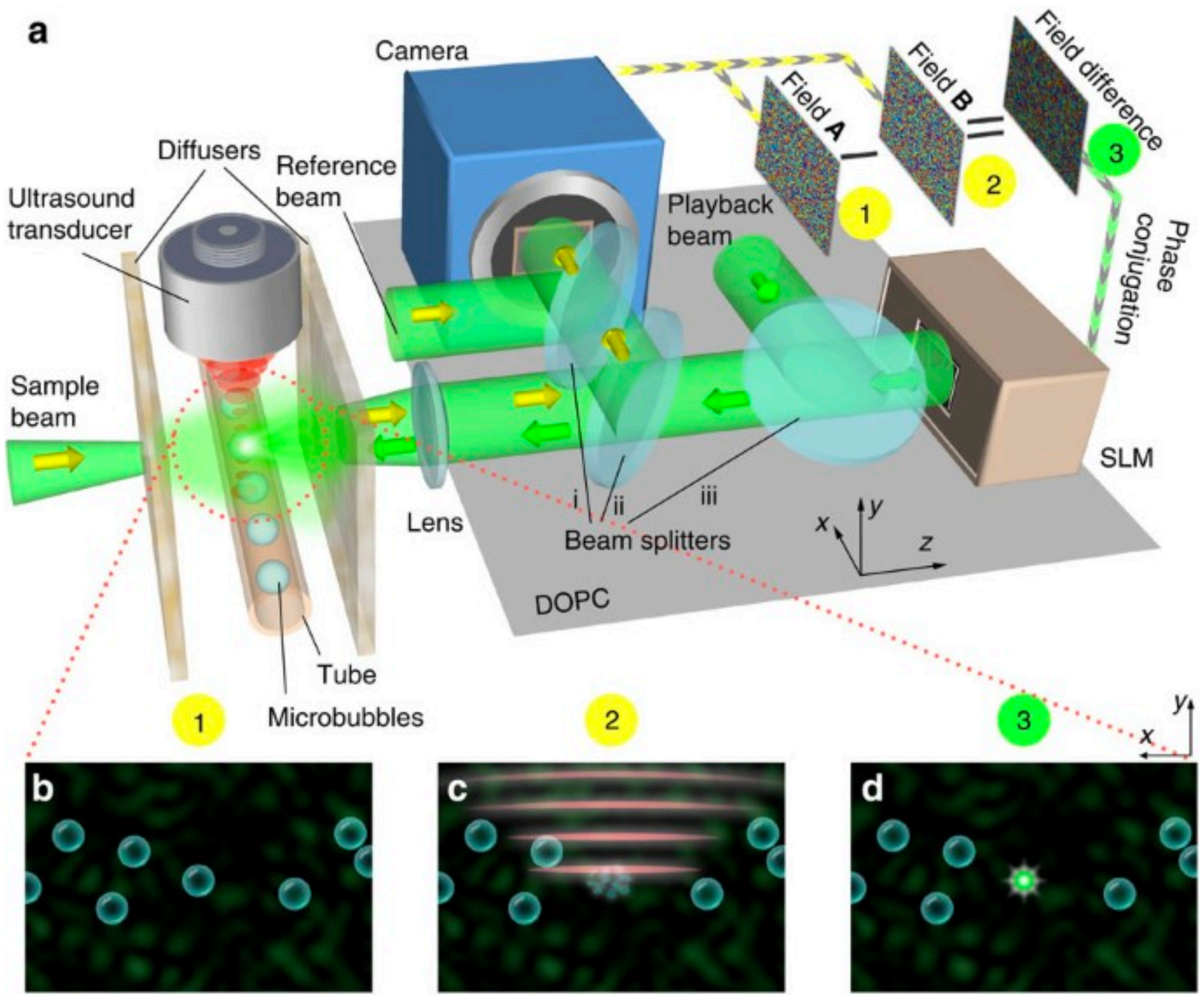
Inferred dynamics



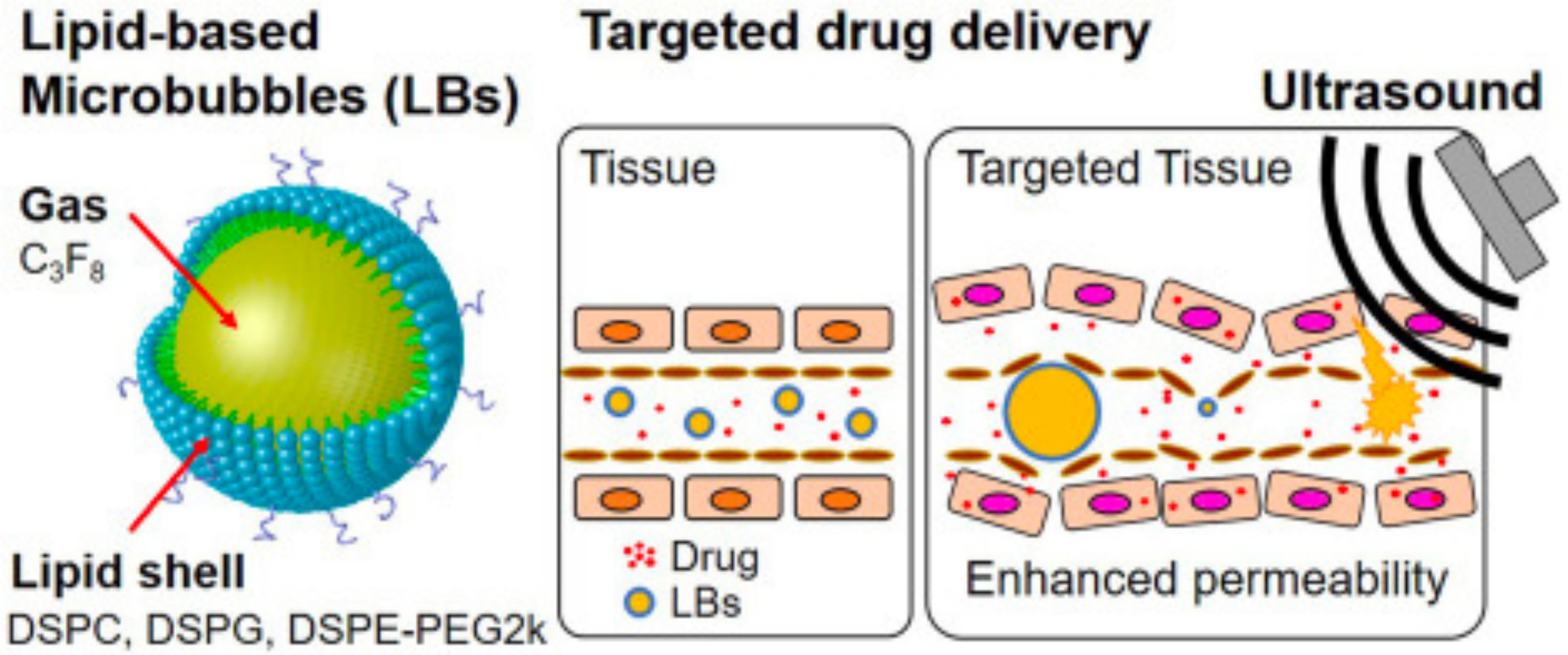
$$u_{xx} + uv_{xy} + \ldots + ? = 0$$

… or anything else?

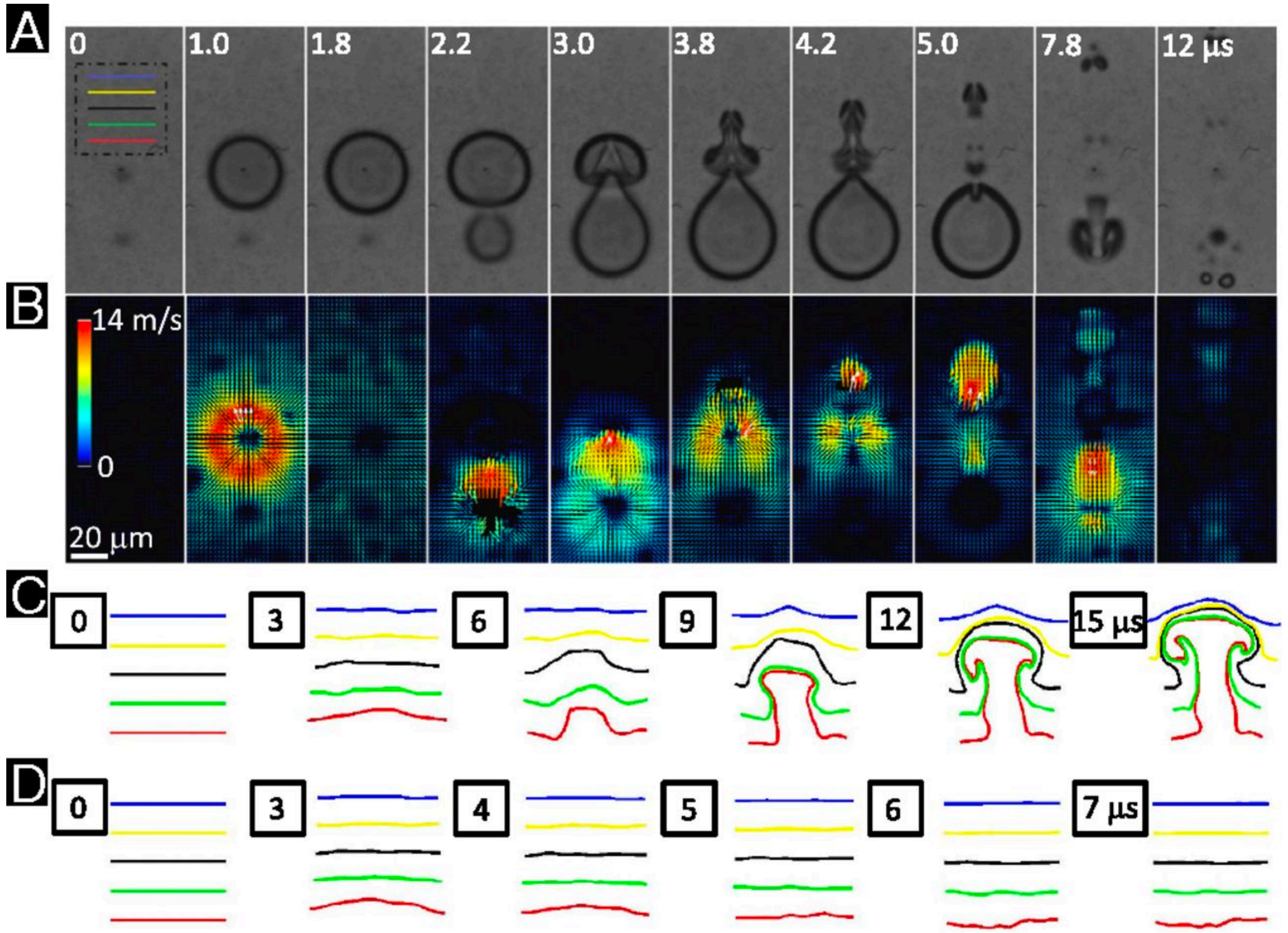
Background



Ruan *et al*., *Nat. Com.*, 2015

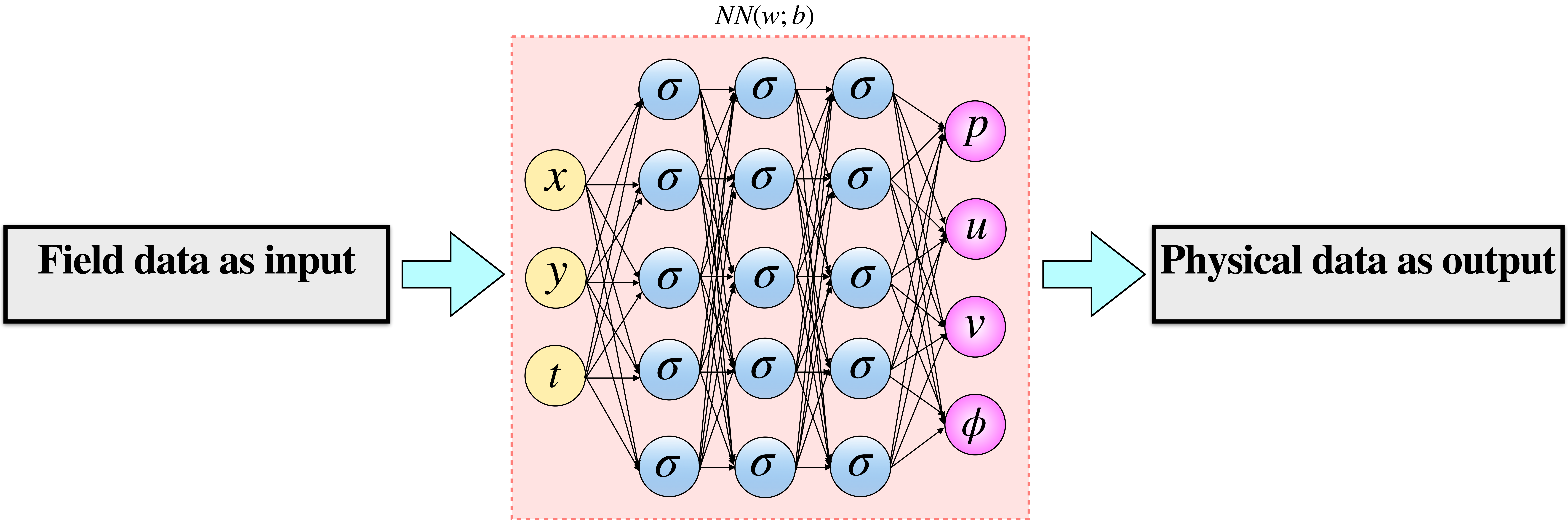


Omata *et al*., *Adv. Drug Deliv. Rev.*, 2020



Omata *et al*., *PNAS*, 2015

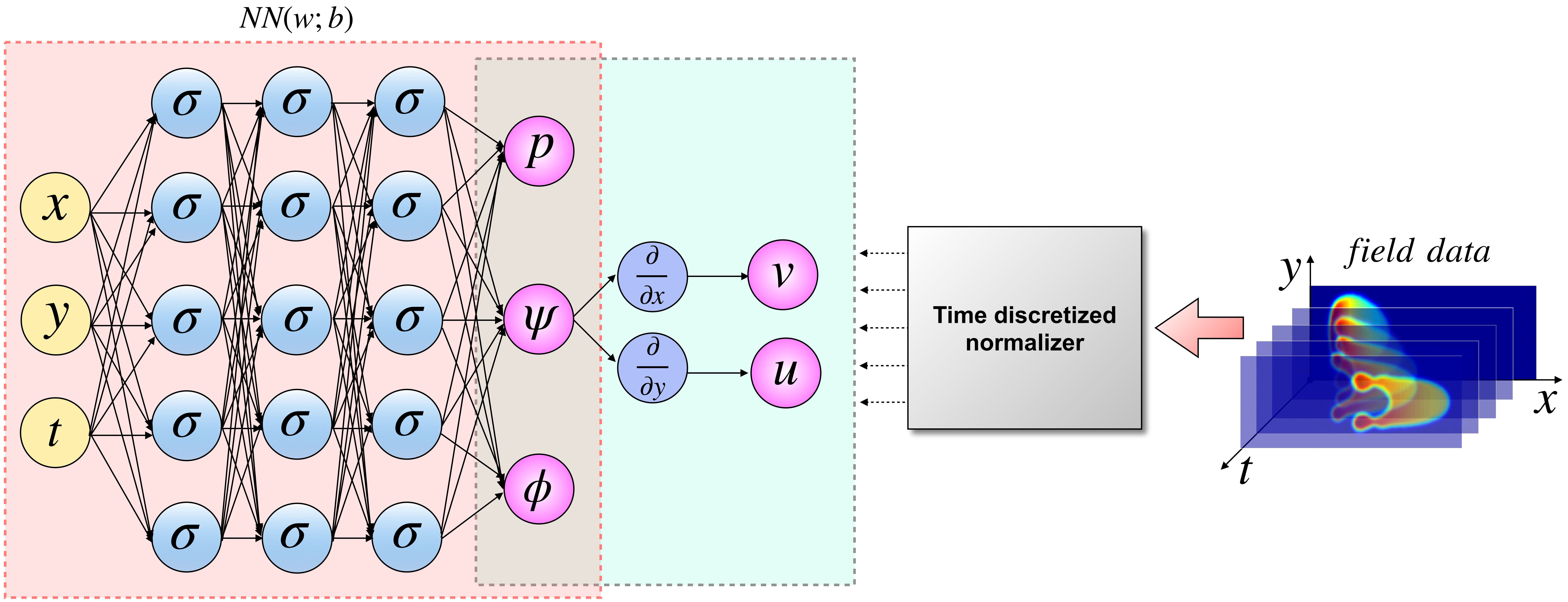
Methods



Deep Neural Networks

Methods

BubbleNet: Physics-Informed Neural Networks for general bubble dynamics



Methods

Traditional DNNs

Algorithm 1 DNN for predicting bubble dynamics

function DEEPNEURALNET($self,\ x,\ y,\ t,\ u,\ v,\ p,\ \phi,\ layers$)

  $(\hat{x},\ \hat{y},\ \hat{t},\ \hat{u},\ \hat{v},\ \hat{p},\ \hat{\phi}) = $ UPDATE$(x,\ y,\ t,\ u,\ v,\ p,\ \phi)$

  $(\widehat{weights},\ \widehat{biases},\ \widehat{layers}) = self.$INITIALIZENN$(weights,\ biases,\ layers)$

  $self.\text{Loss} = \texttt{MSE}[(u - u_{pred}) + (v - v_{pred}) + (p - p_{pred}) + (\phi - \phi_{pred})]$

  $u_{pred} = self.\text{Net}_{\text{u}}(x,\ y,\ t)$

  $v_{pred} = self.\text{Net}_{\text{v}}(x,\ y,\ t)$

  $p_{pred} = self.\text{Net}_{\text{p}}(x,\ y,\ t)$

  $\phi_{pred} = self.\text{Net}_{\phi}(x,\ y,\ t)$

  Optimization method 'L-BFGS-B' & Optimizer: Adam

  def INITIALIZENN($self,\ layers$)

    Initialize all the *weights* & *biases* for Net_{u}, Net_{v}, Net_{p}, Net_{ϕ}.

  def NEURALNET($self,\ weights,\ biases$)

    Build NN for $u,\ v,\ p,\ \phi$ with four sets of *weights* & *biases*.

  def $\{\text{Net}_{\text{u}}, \text{Net}_{\text{v}}, \text{Net}_{\text{p}}, \text{Net}_{\phi}\}$ $(self,\ x,\ y,\ t)$

    $\{u,\ v,\ p,\ \phi\} = self.$NEURALNET$(x,\ y,\ t,\ weights,\ biases)$

  def TRAIN($self,\ iterations$)

    Obtain training time & Losses; train the NN with Adam optimizer.

  def PREDICT $\{u,\ v,\ p,\ \phi\}$ $(self,\ iterations)$

    $\{u_{pred},\ v_{pred},\ p_{pred},\ \phi_{pred}\} = self.$ sess.run$(x,\ y,\ t)$

end function

Input $= \{x,\ y,\ t\}$, Output $= \{u,\ v,\ p,\ \phi\}$

Hidden layers = [30 neurons $\times$ 9 layers]

Load fields data of micro-bubble system dynamics simulation.

Set training sets $= \{x_{train},\ y_{train},\ t_{train},\ u_{train},\ v_{train},\ p_{train},\ \phi_{train},\ layers\}$ = MaxMinScaler(Simulation Data)

`model` = DEEPNEURALNET(training sets)

`model`.TRAIN(10000)

Set target prediction time as t_{pred}

Obtain $\{u_{pred},\ v_{pred},\ p_{pred},\ \phi_{pred}\}$ = `model`.PREDICT$(x,\ y,\ t)$ at t_{pred}.

Save all the data & post-processing.

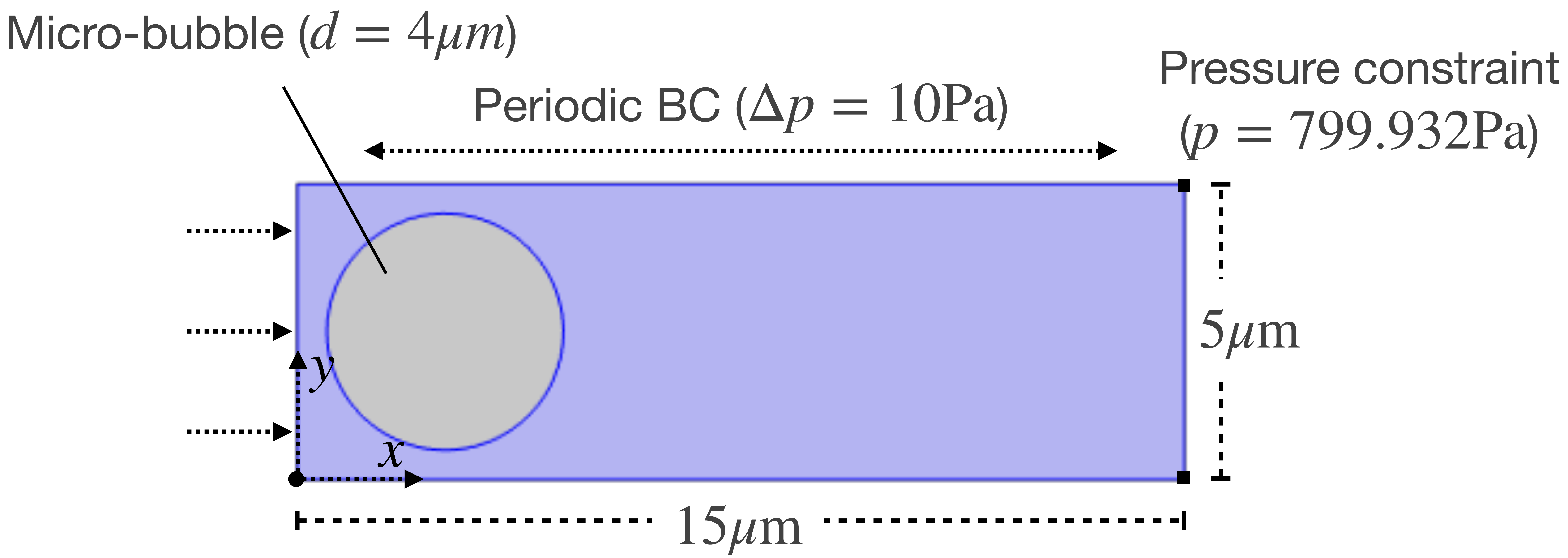
Methods

BubbleNet

Algorithm 2 BubbleNet: physics-informed neural network for bubble dynamics

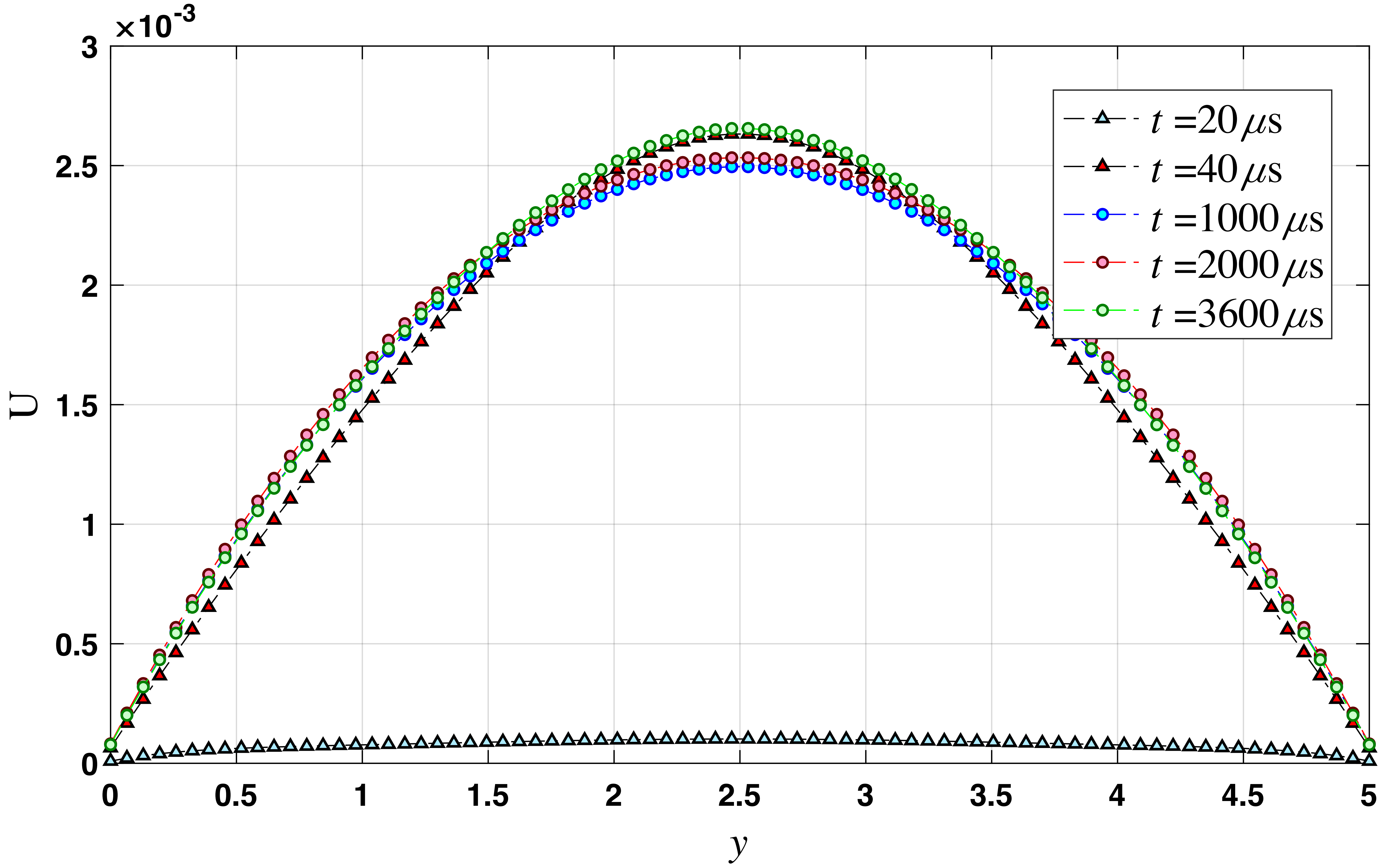
function $\textsc{BubbleNet}(self,\ x,\ y,\ t,\ u,\ v,\ p,\ \phi,\ layers)$
  $(\hat{x},\ \hat{y},\ \hat{t},\ \hat{u},\ \hat{v},\ \hat{p},\ \hat{\phi}) = \textsc{Update}(x,\ y,\ t,\ u,\ v,\ p,\ \phi)$
  $(\widehat{weights},\ \widehat{biases},\ \widehat{layers}) = self.\textsc{InitializeNN}(weights,\ biases,\ layers)$
  $self.\text{Loss} = \texttt{MSE}[(u - u_{pred}) + (v - v_{pred}) + (p - p_{pred}) + (\phi - \phi_{pred})]$
  $\{u_{pred}, v_{pred}, p_{pred}, \phi_{pred}\} = self.\{\text{Net}_\psi, \text{Net}_\text{p}, \text{Net}_\phi\}(x,\ y,\ t)$
  Optimization method 'L-BFGS-B' & Optimizer: Adam
  def $\textsc{InitializeNN}(self,\ layers)$
    Initialize all the *weights* & *biases* for Net_ψ, Net_p, Net_ϕ.
  def $\textsc{NeuralNet}(self,\ weights,\ biases)$
    Build NN for $\psi,\ p,\ \phi$ with four sets of *weights* & *biases*.
  def $\{\text{Net}_\psi, \text{Net}_\text{p}, \text{Net}_\phi\}\ (self,\ x,\ y,\ t)$
    $\{\psi,\ p,\ \phi\} = self.\textsc{NeuralNet}(x,\ y,\ t,\ weights,\ biases)$
    $u = \partial_y \psi \quad \& \quad v = -\partial_x \psi$
  def $\textsc{Train}(self,\ iterations)$
    Obtain training time & Losses; train the NN with Adam optimizer.
  def $\textsc{Predict}\ \{u,\ v,\ p,\ \phi\}\ (self,\ iterations)$
    $\{u_{pred},\ v_{pred},\ p_{pred},\ \phi_{pred}\} = self.\ \texttt{sess}.\text{run}(x,\ y,\ t)$
end function
Set training sets $= \{x_{train},\ y_{train},\ t_{train},\ u_{train},\ v_{train},\ p_{train},\ \phi_{train},\ layers\}$ $=$ TimeDiscretizedNormalization(Simulation Data, *timestep*)
$\texttt{model} = \textsc{BubbleNet}(\text{training sets})$
$\texttt{model}.\textsc{Train}(10000)$
Rest procedures same as **Algorithm 1**

Case 1: single bubble movement

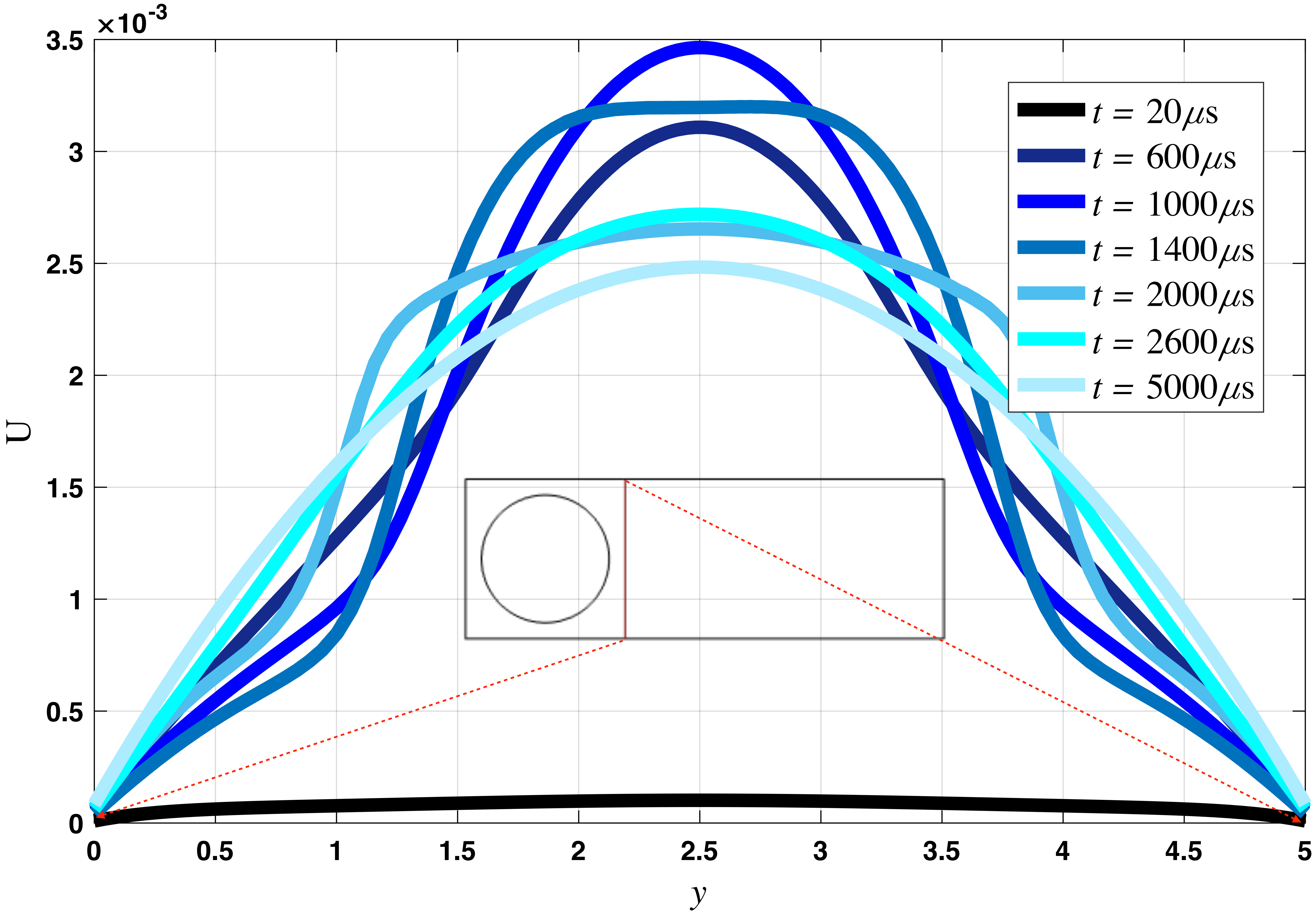


IC: $p_0 = 799.932\text{Pa}\ (6\text{mmHg})$

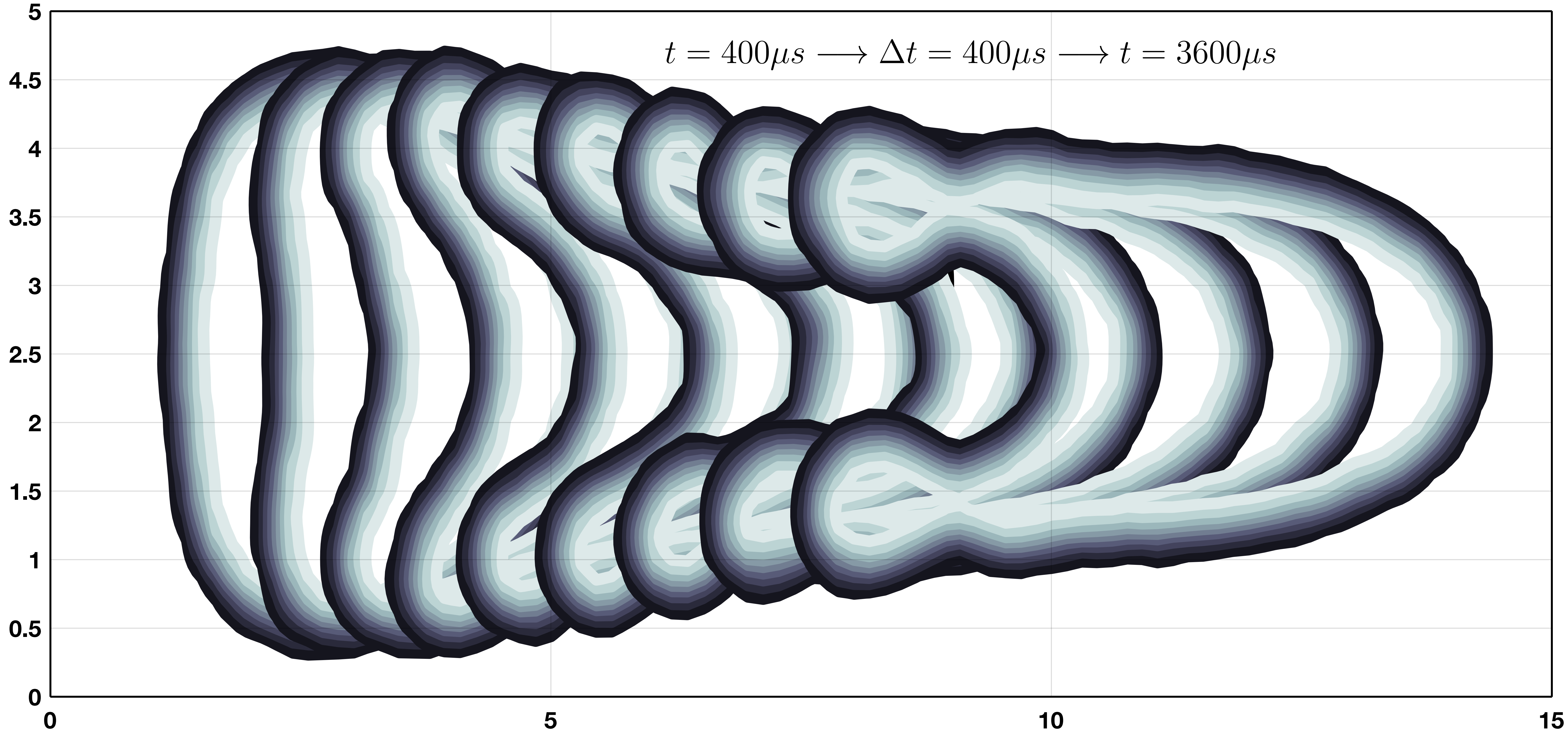
Case 1: single bubble movement



Case 1: single bubble movement



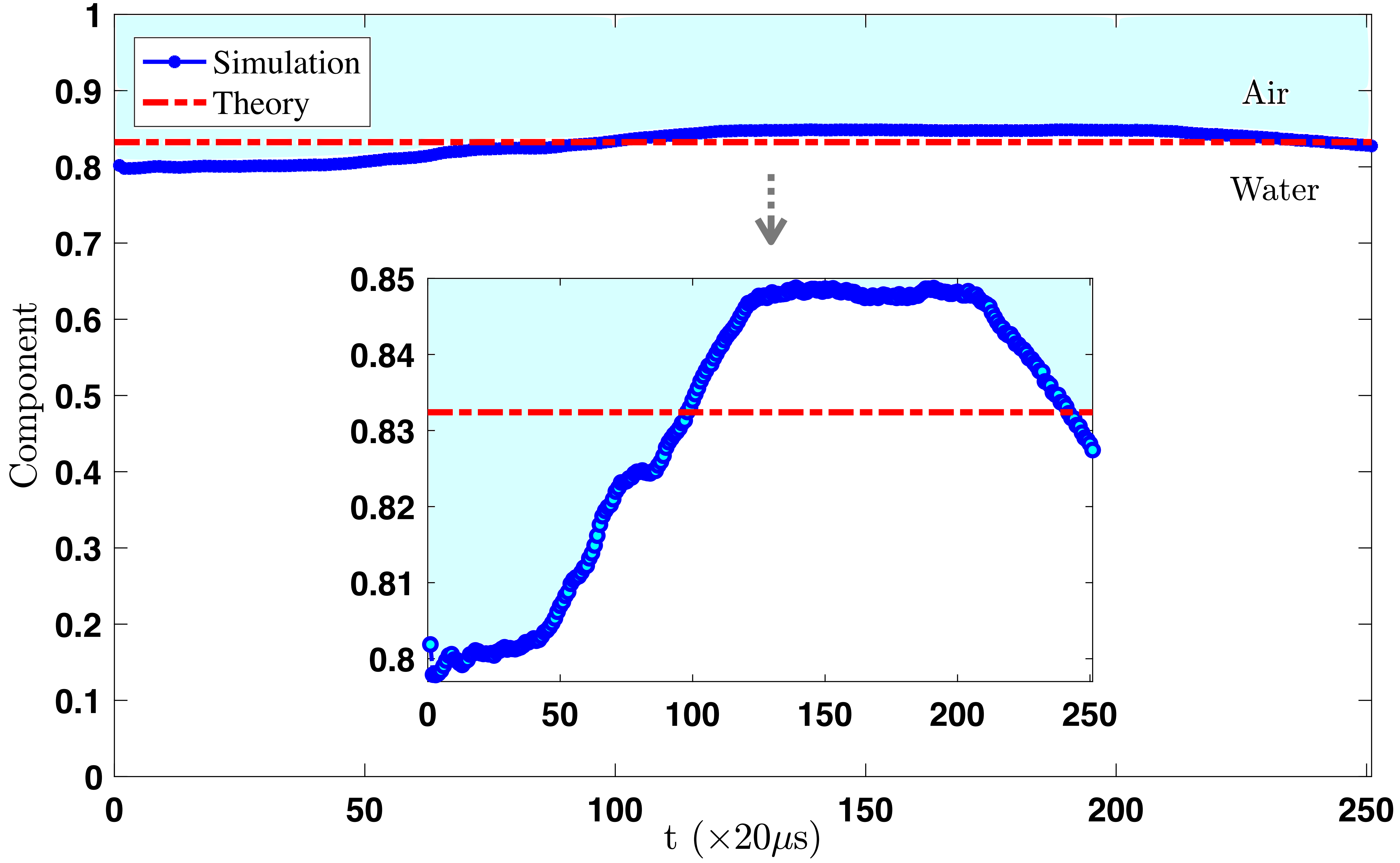
Results



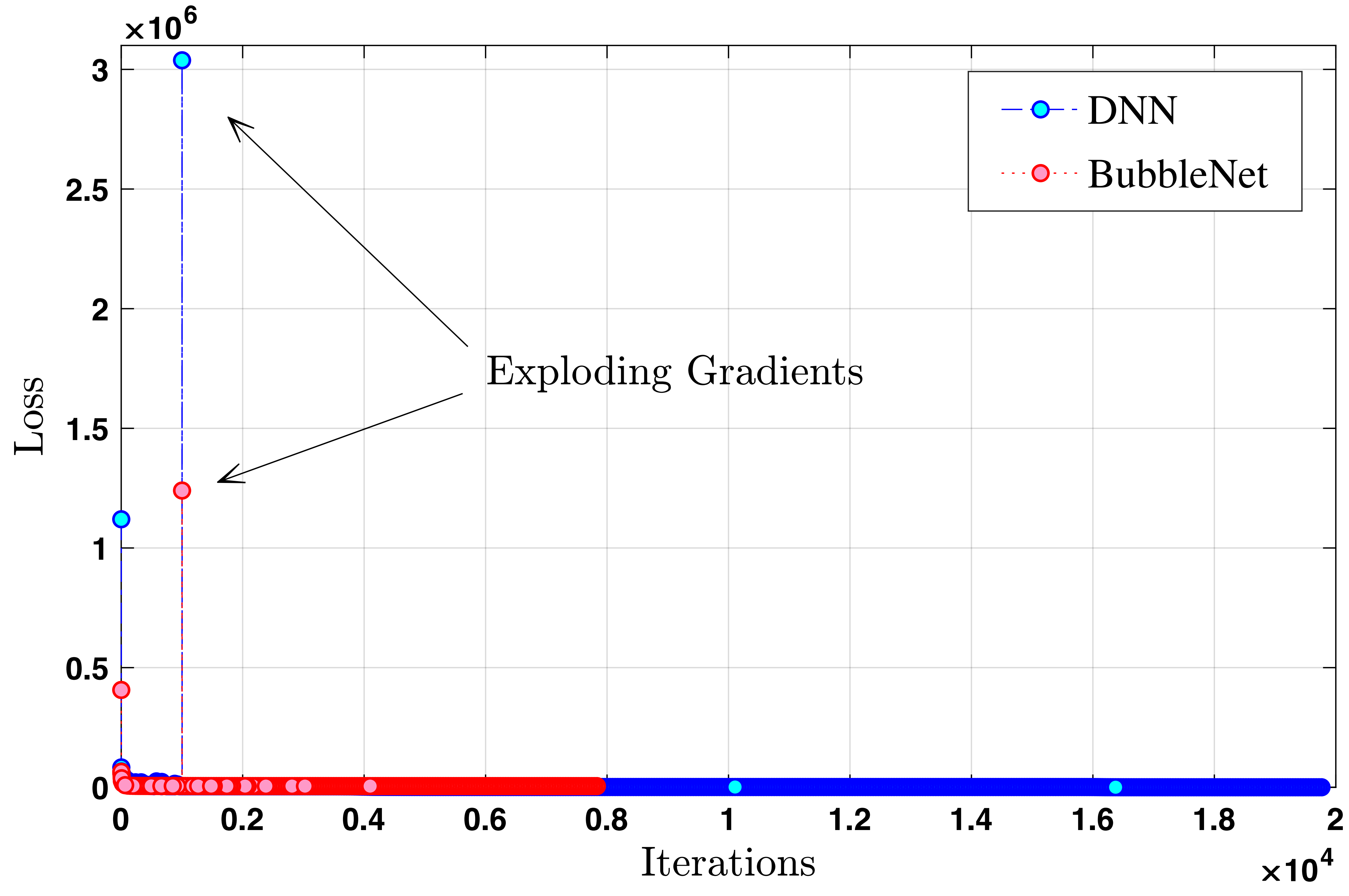
$t = 400\mu s \longrightarrow \Delta t = 400\mu s \longrightarrow t = 3600\mu s$
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
0
5
10
15

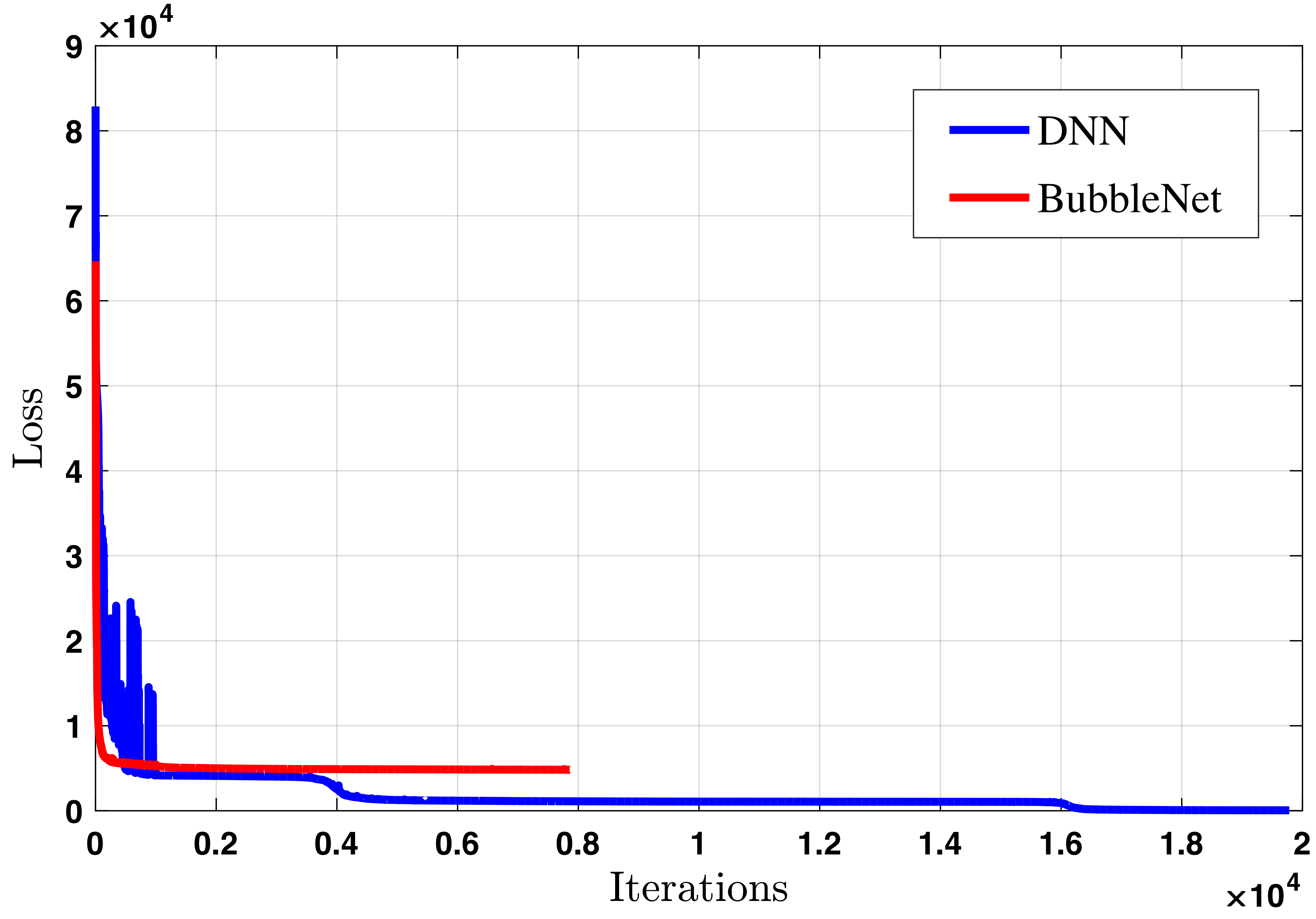
Results

The component for multiphase flow computation is estimated to satisfy general conservation laws.

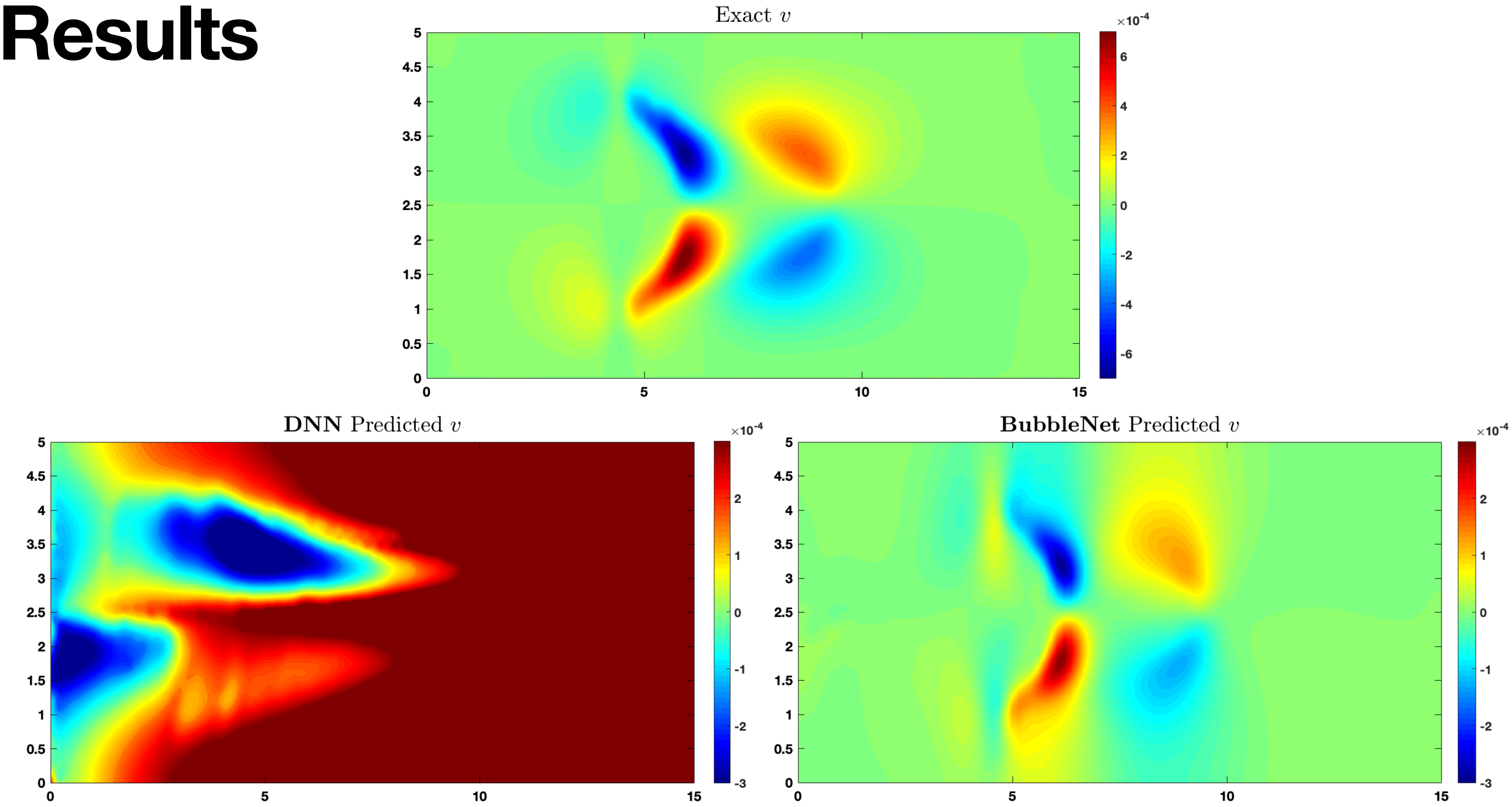


Results

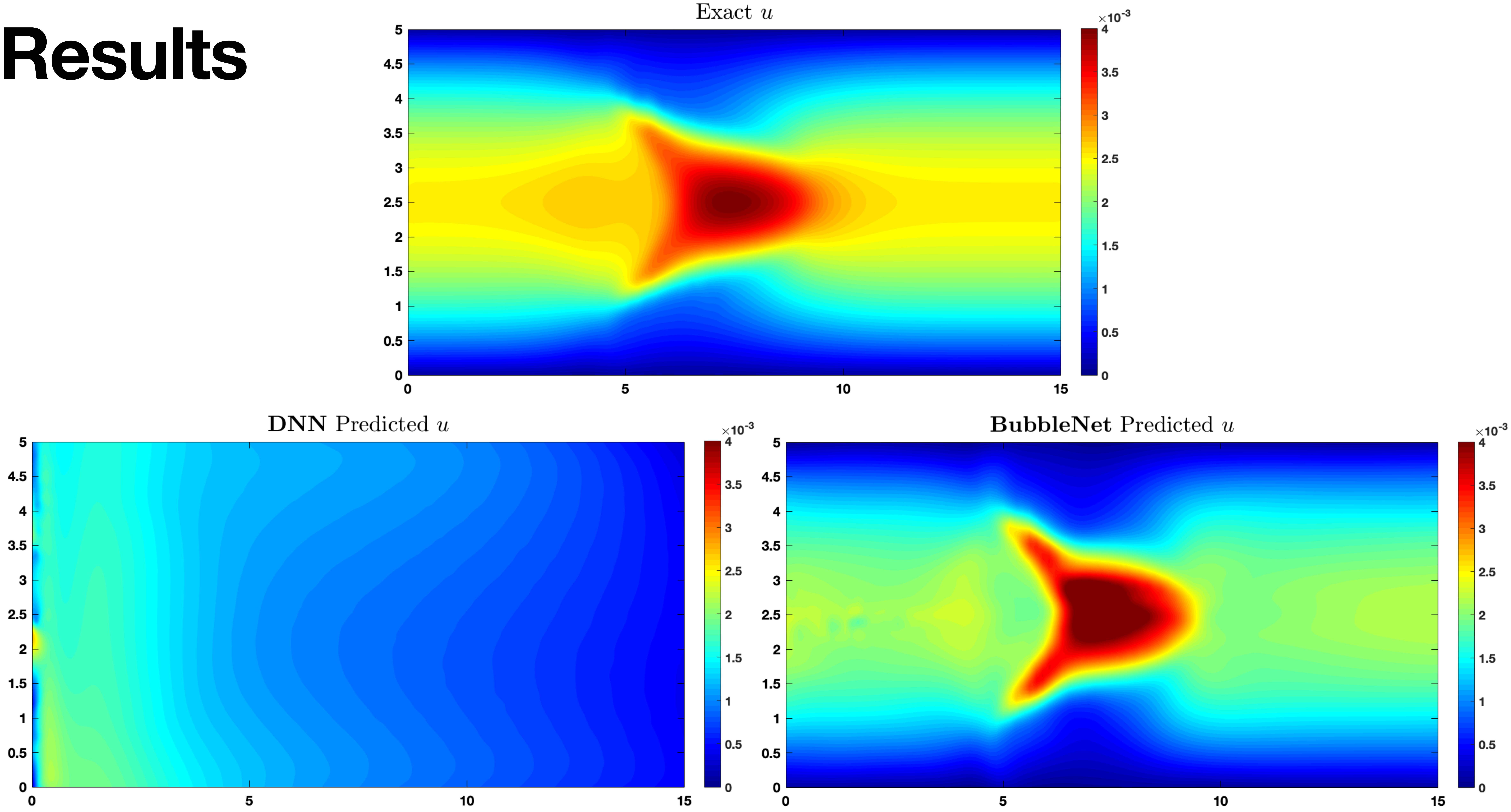

×10^6
3
2.5
2
1.5
1
0.5
0
Loss
DNN
BubbleNet
Exploding Gradients
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
Iterations
×10^4


9
×10^4
8
7
6
5
4
3
2
1
0
Loss
DNN
BubbleNet
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
Iterations
×10^4

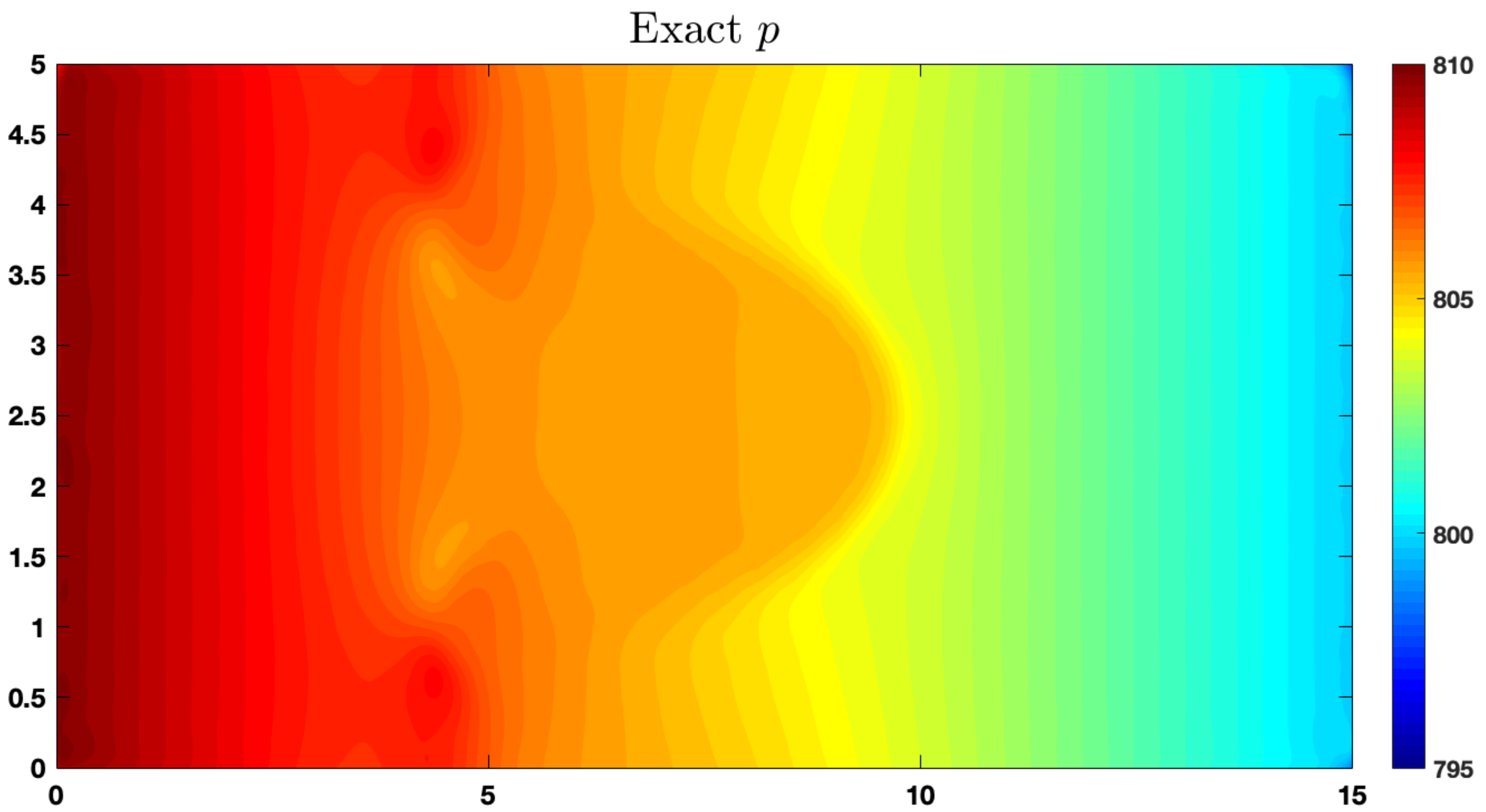
Results

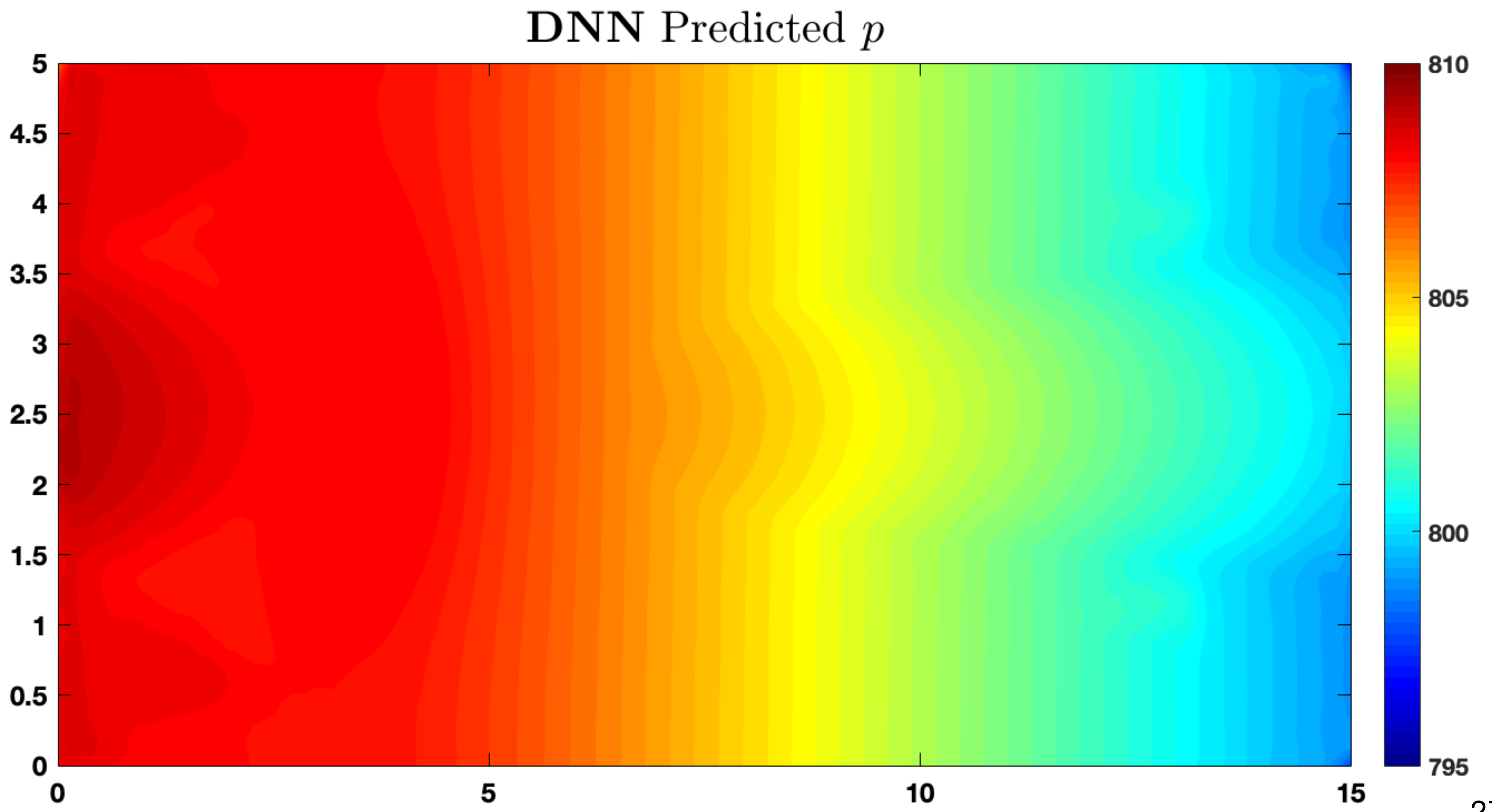

Exact v
DNN Predicted v
BubbleNet Predicted v
$\times 10^{-4}$
$\times 10^{-4}$
$\times 10^{-4}$

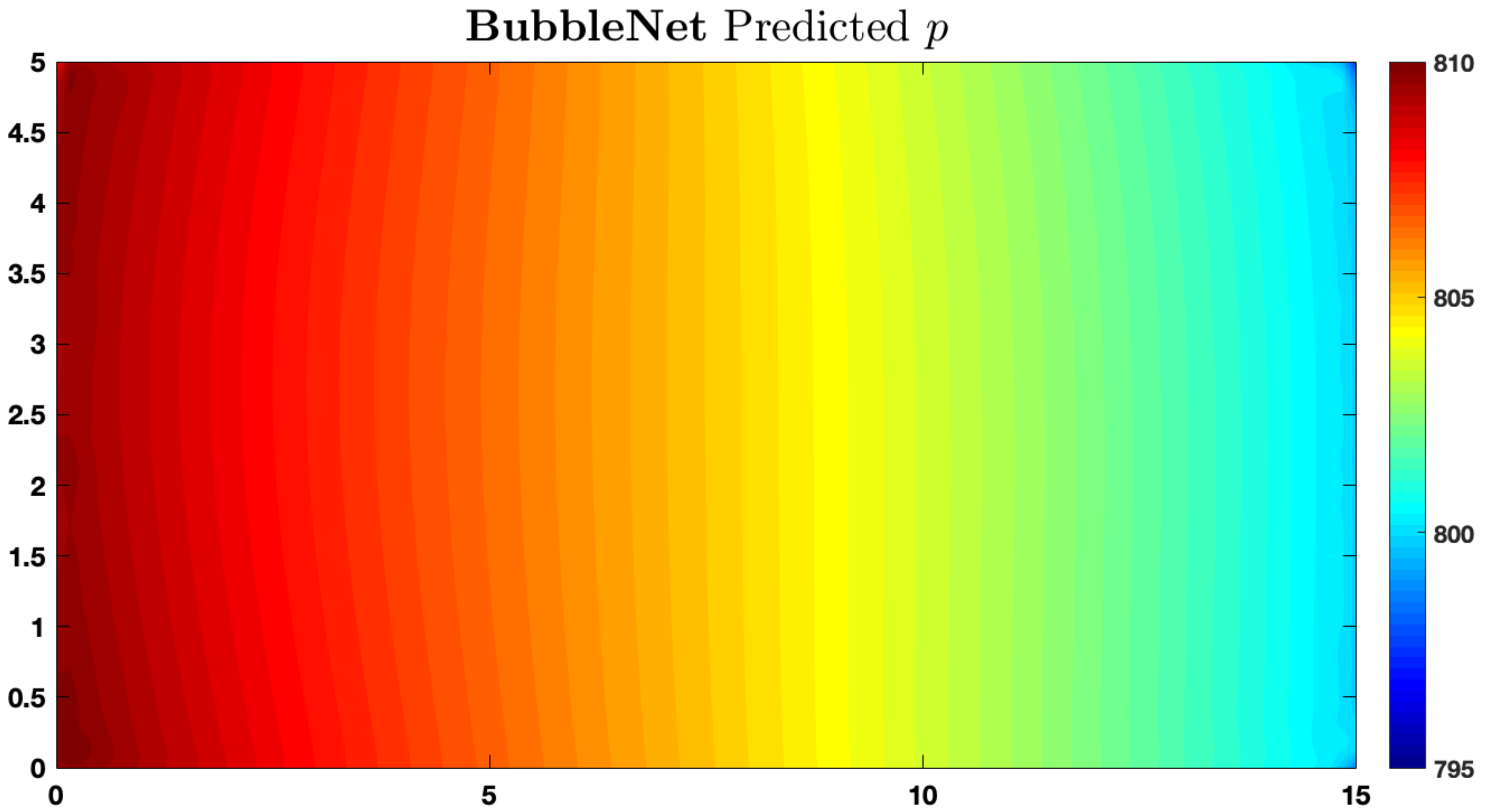
Results


Exact u
DNN Predicted u
BubbleNet Predicted u
$\times 10^{-3}$
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
10
15

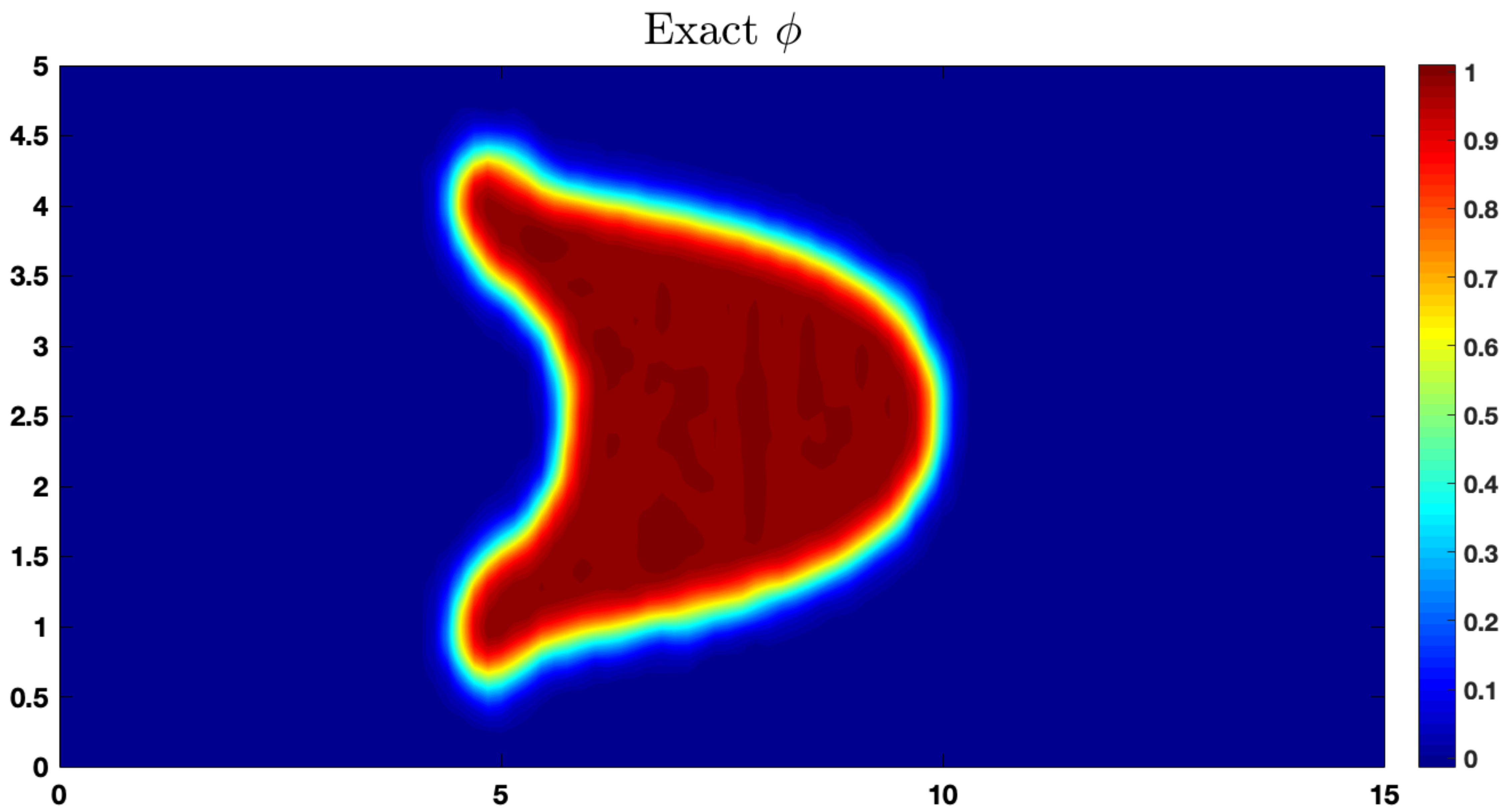
Results

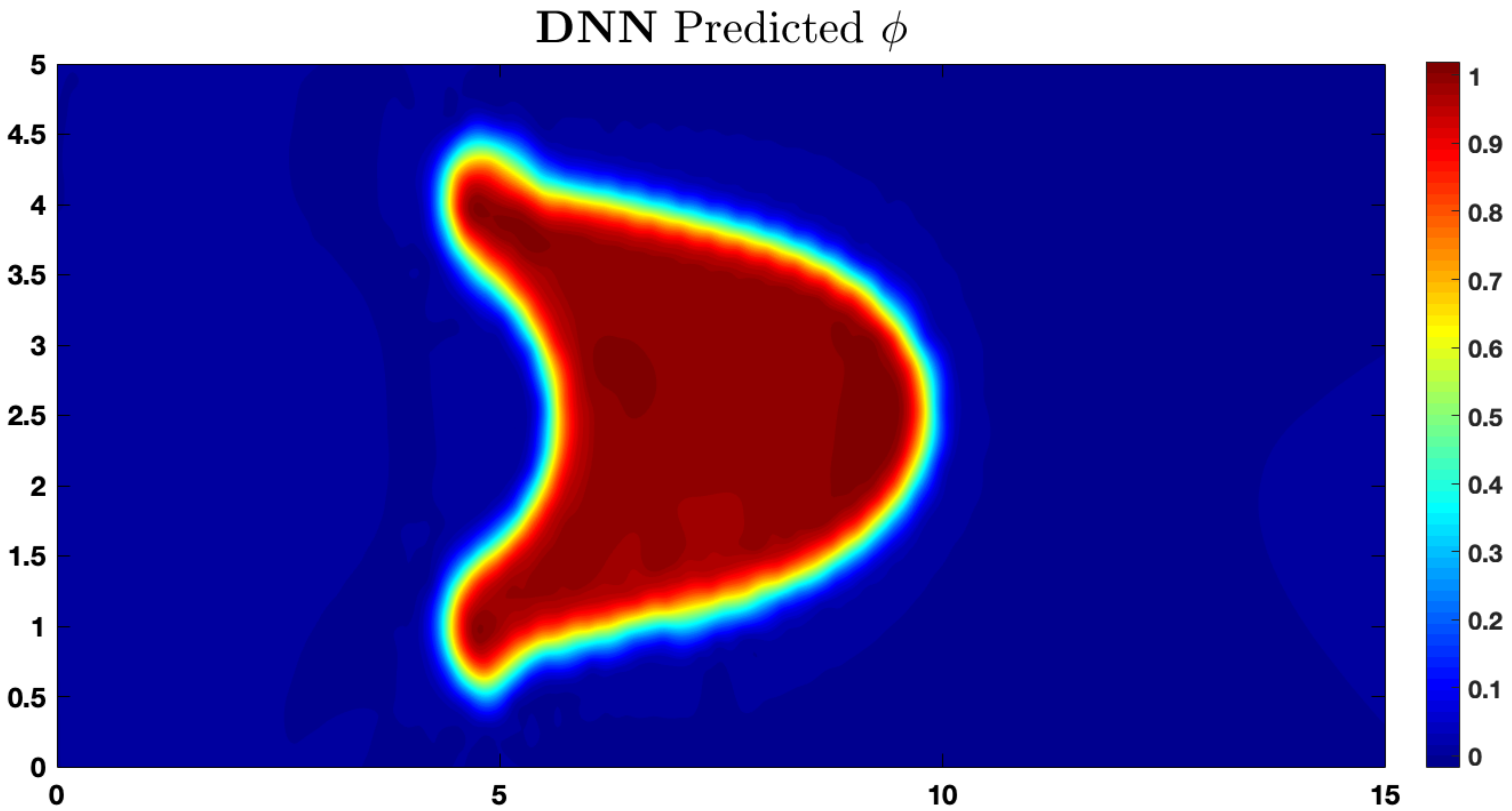

Exact p
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
0
5
10
15
810
805
800
795

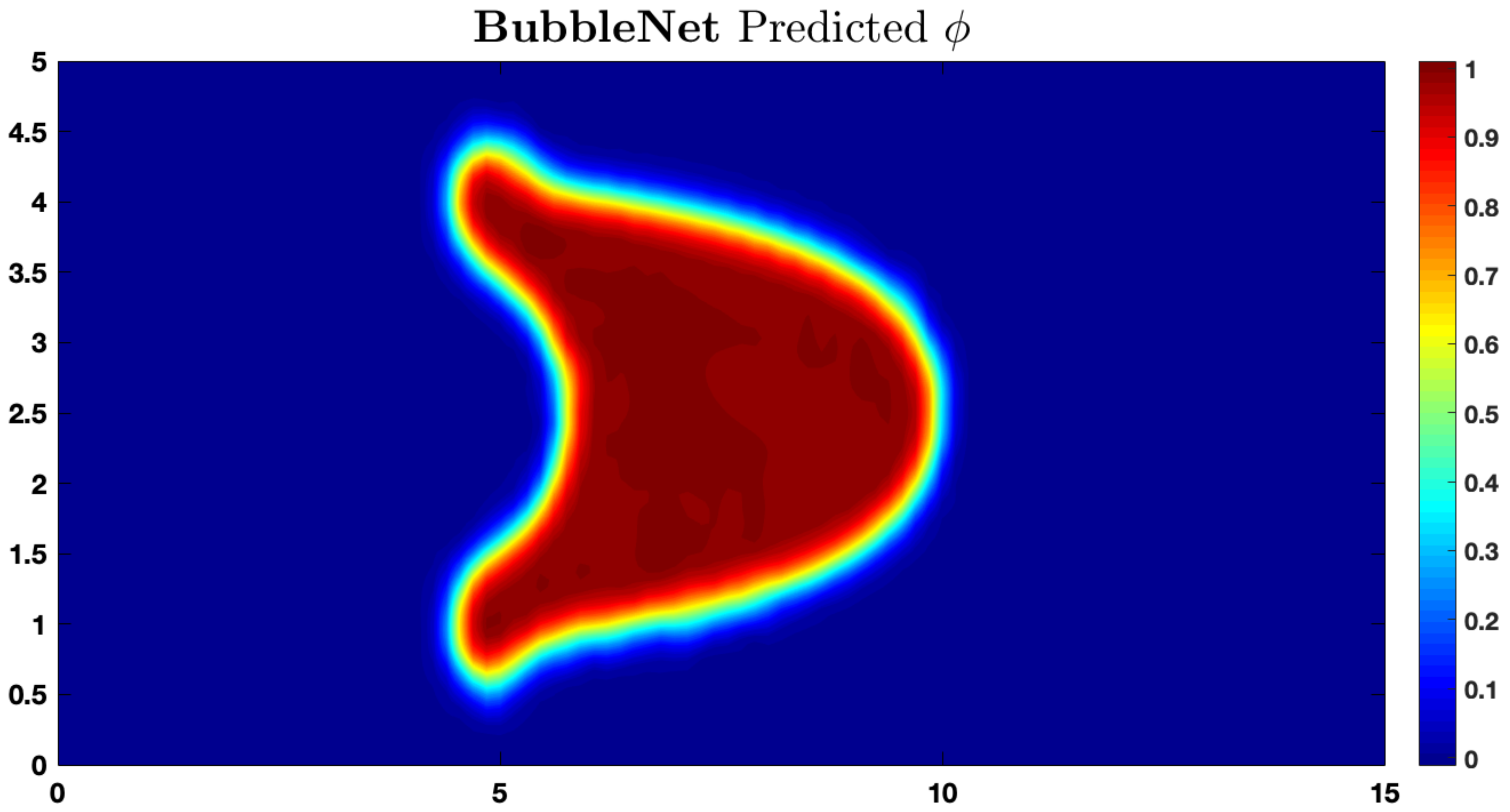

DNN Predicted p
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
0
5
10
15
810
805
800
795


BubbleNet Predicted p
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
0
5
10
15
810
805
800
795

Results


Exact ϕ
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
0
5
10
15
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0


DNN Predicted ϕ
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
0
5
10
15
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0


BubbleNet Predicted ϕ
5
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
0
5
10
15
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0

Error Analysis

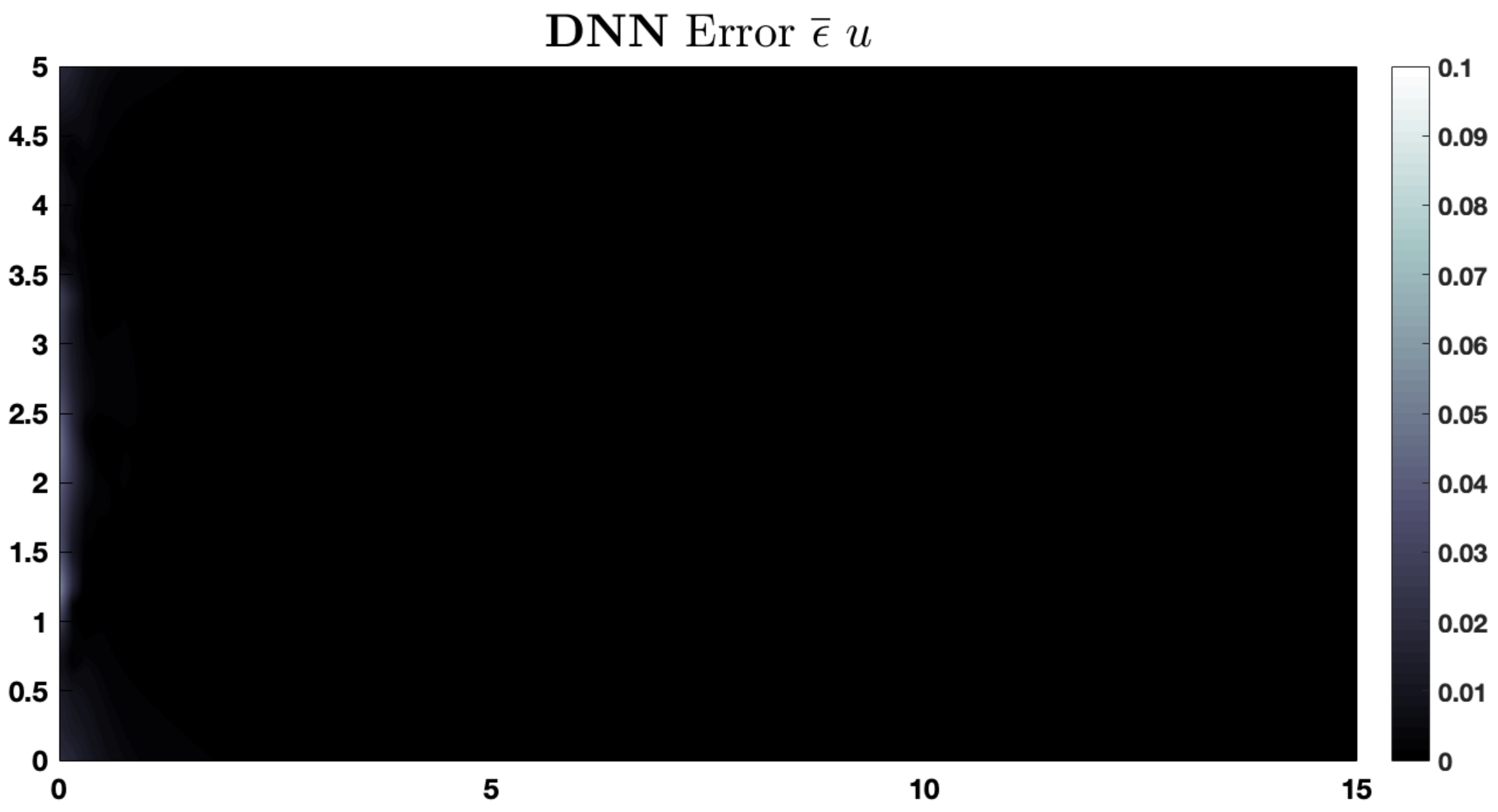
Relative error $\overline{\epsilon}$ of training can taking the form:

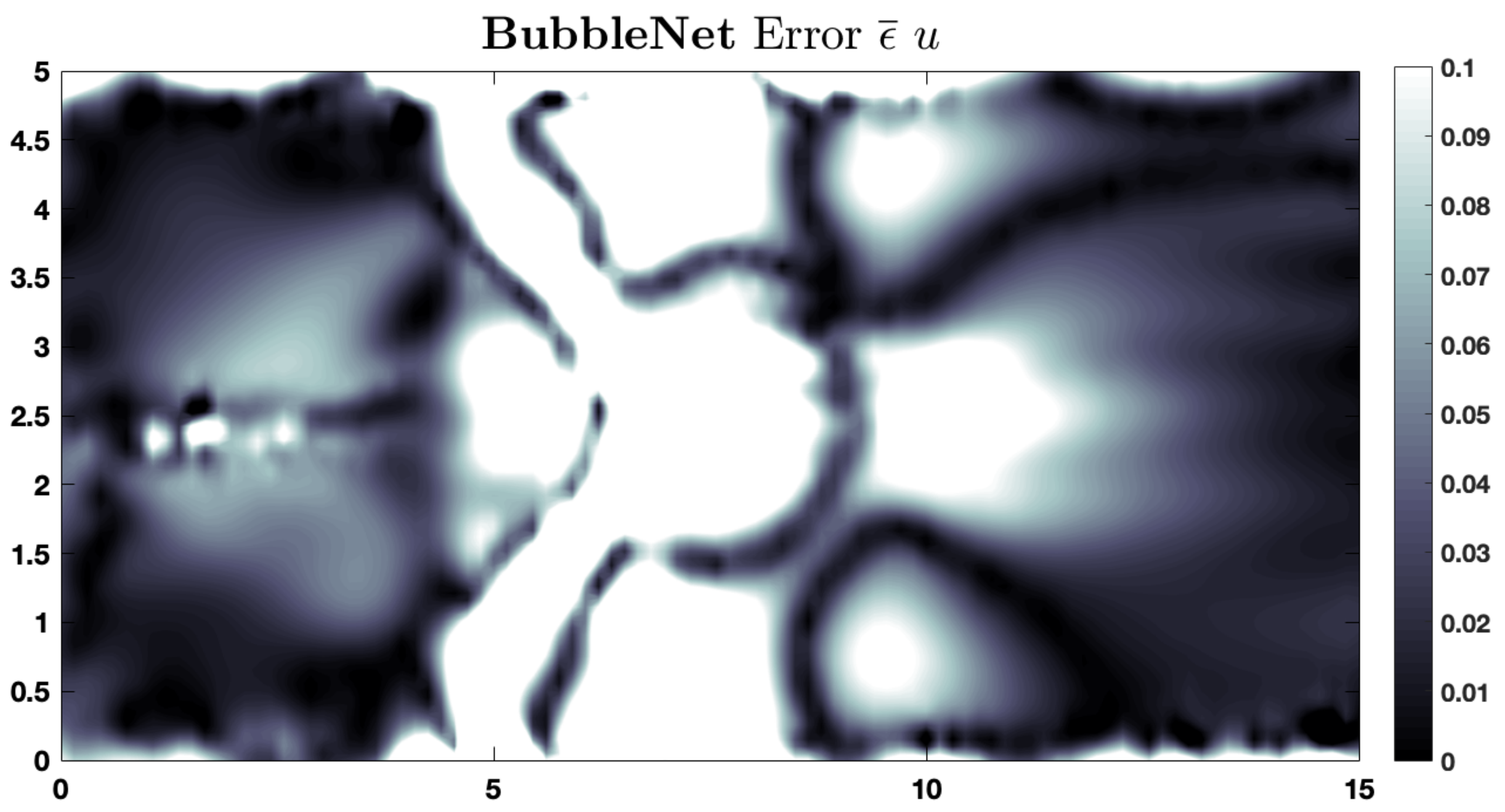
$$\overline{\epsilon}_p = \frac{|p_{NN} - p_{train}|}{|p_{train}|} \qquad \overline{\epsilon}_u = \frac{|u_{NN} - u_{train}|}{|u_{train}|} \qquad \overline{\epsilon}_v = \frac{|v_{NN} - v_{train}|}{|v_{train}|} \qquad \overline{\epsilon}_\phi = \frac{|\phi_{NN} - \phi_{train}|}{|\phi_{train}|}$$

Absolute error $|\epsilon|$ of predictions can taking the form:

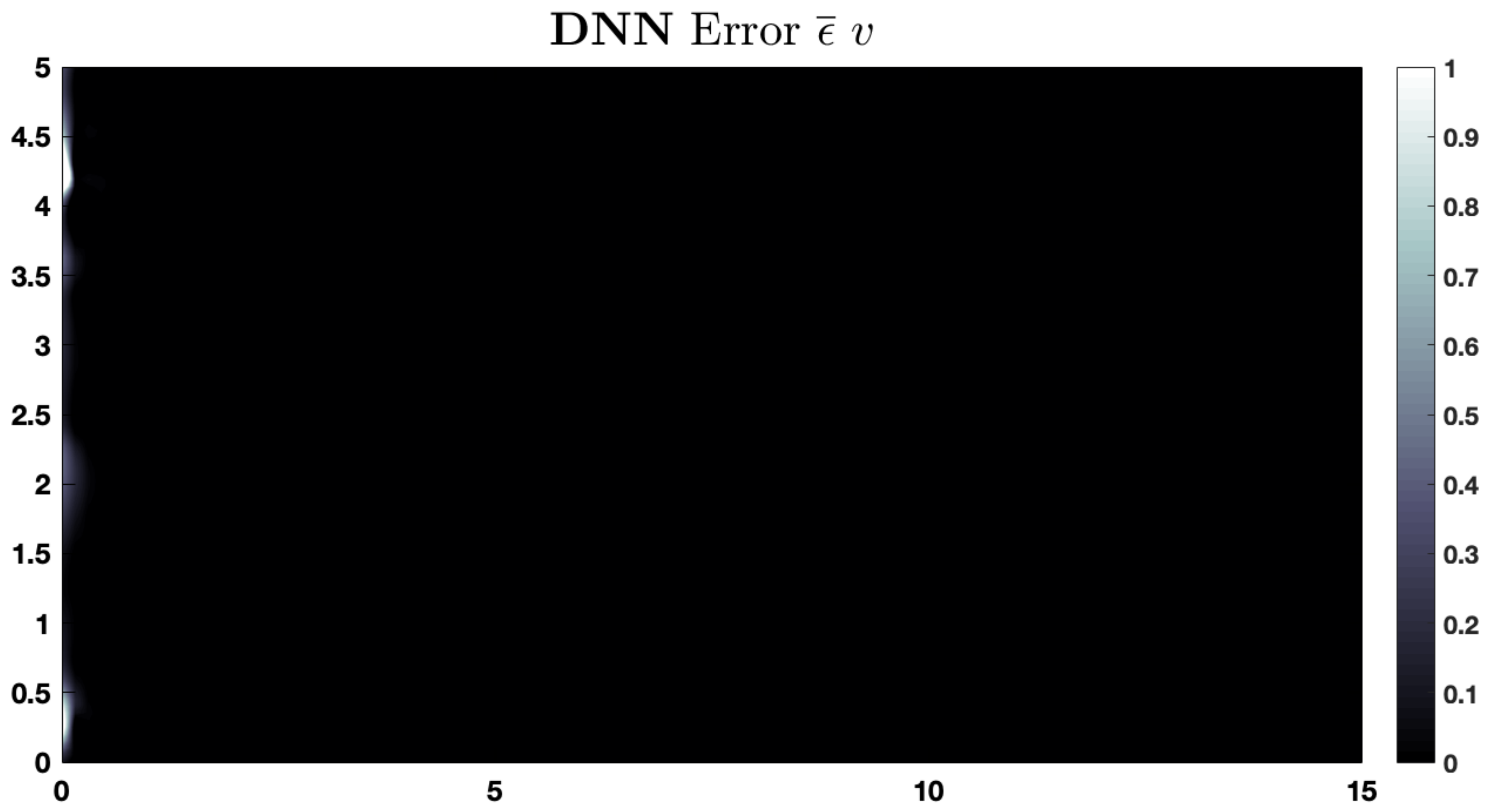
$$|\epsilon_p| = |p_{pred} - p_{exact}| \qquad |\epsilon_u| = |u_{pred} - u_{exact}| \qquad |\epsilon_v| = |v_{pred} - v_{exact}| \qquad |\epsilon_\phi| = |\phi_{pred} - \phi_{exact}|$$

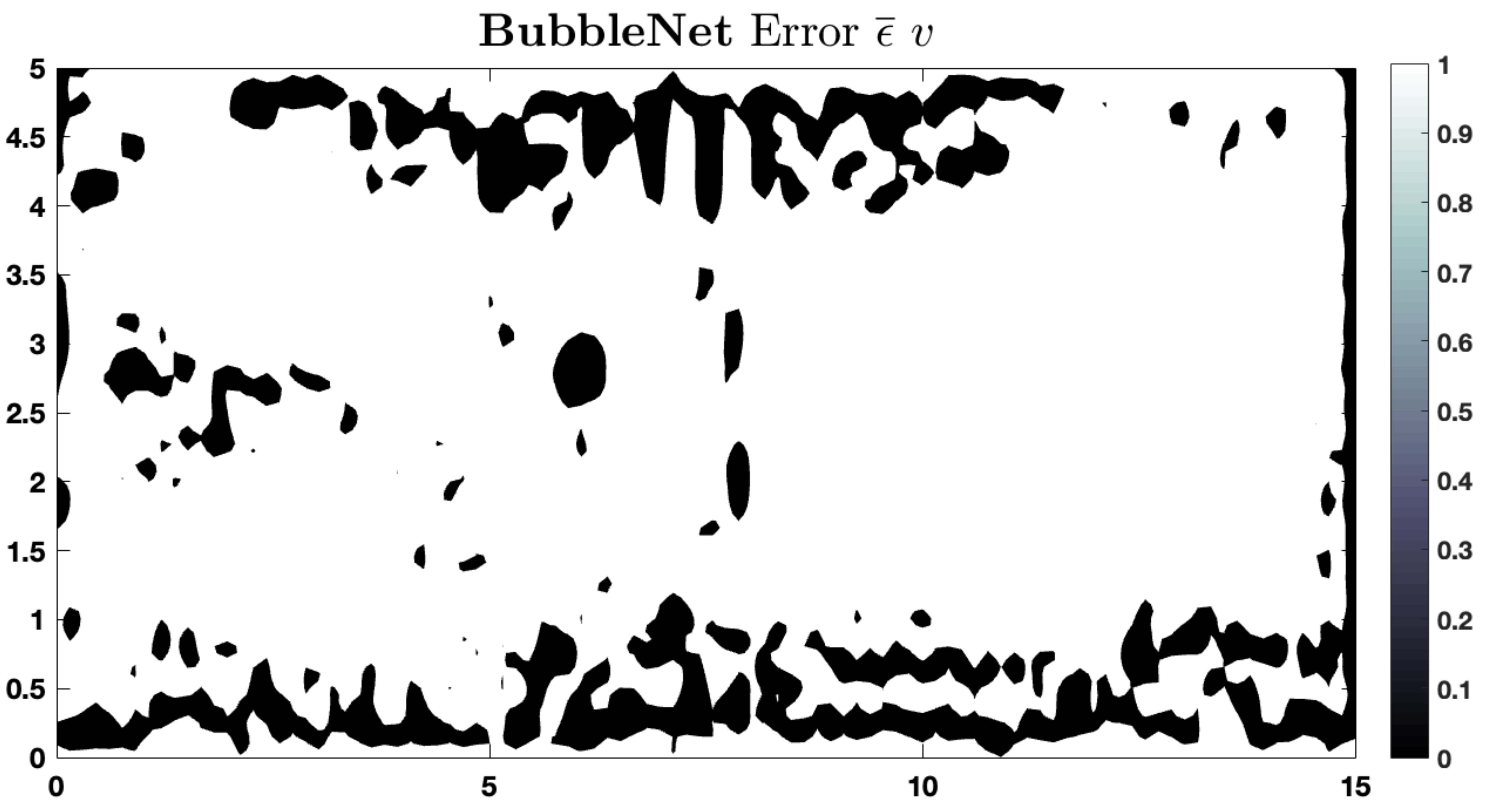
Error Analysis



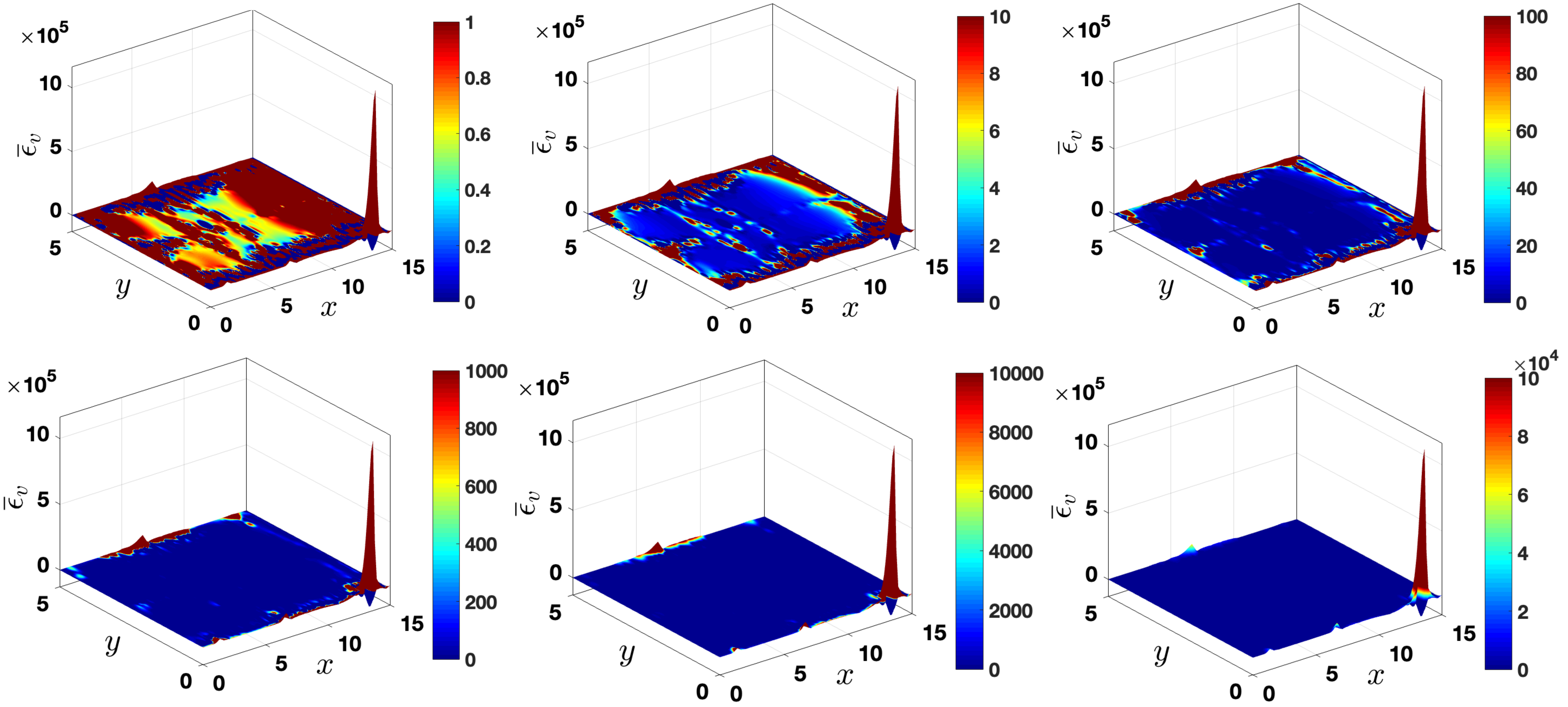


Error Analysis

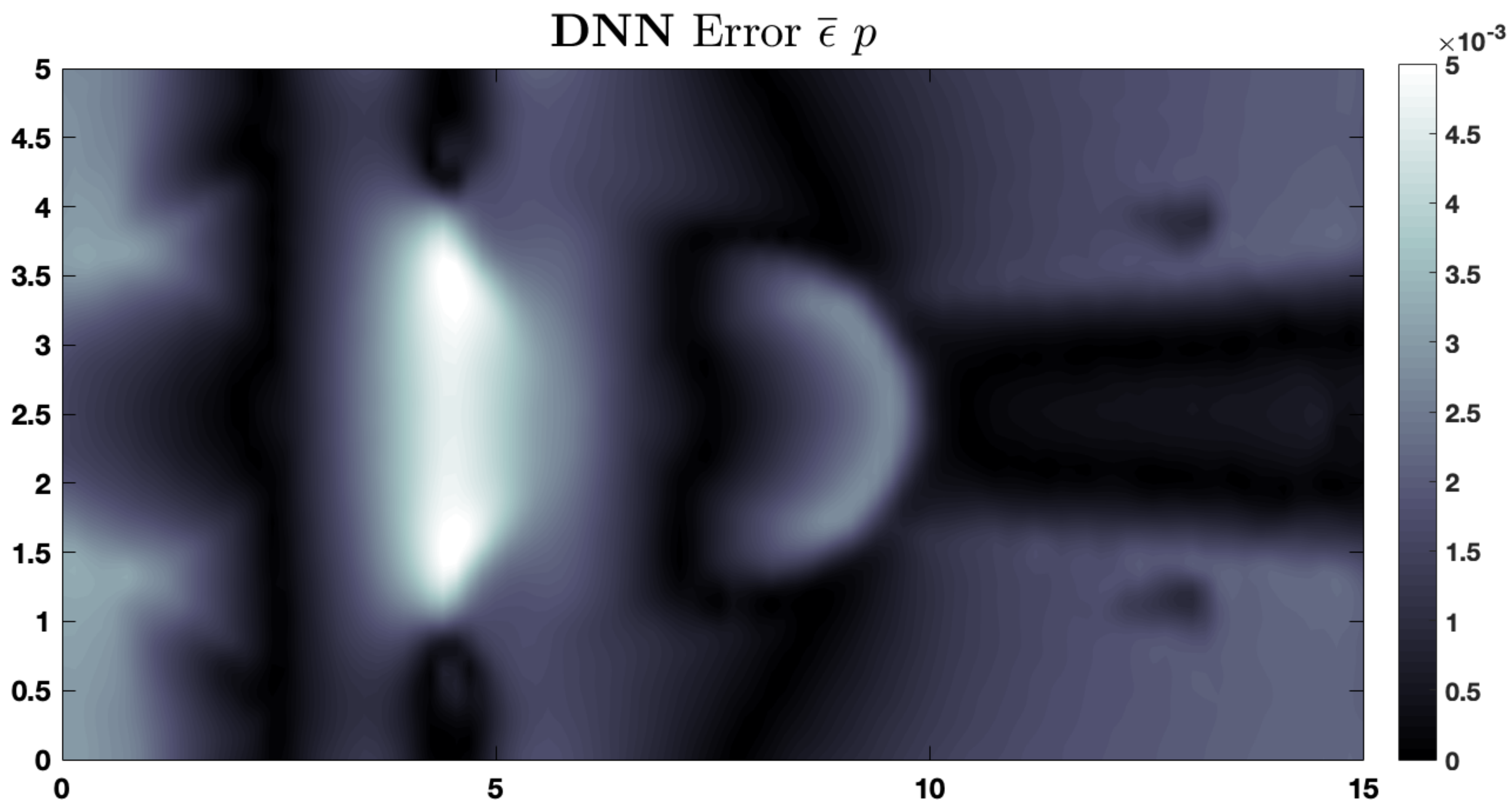


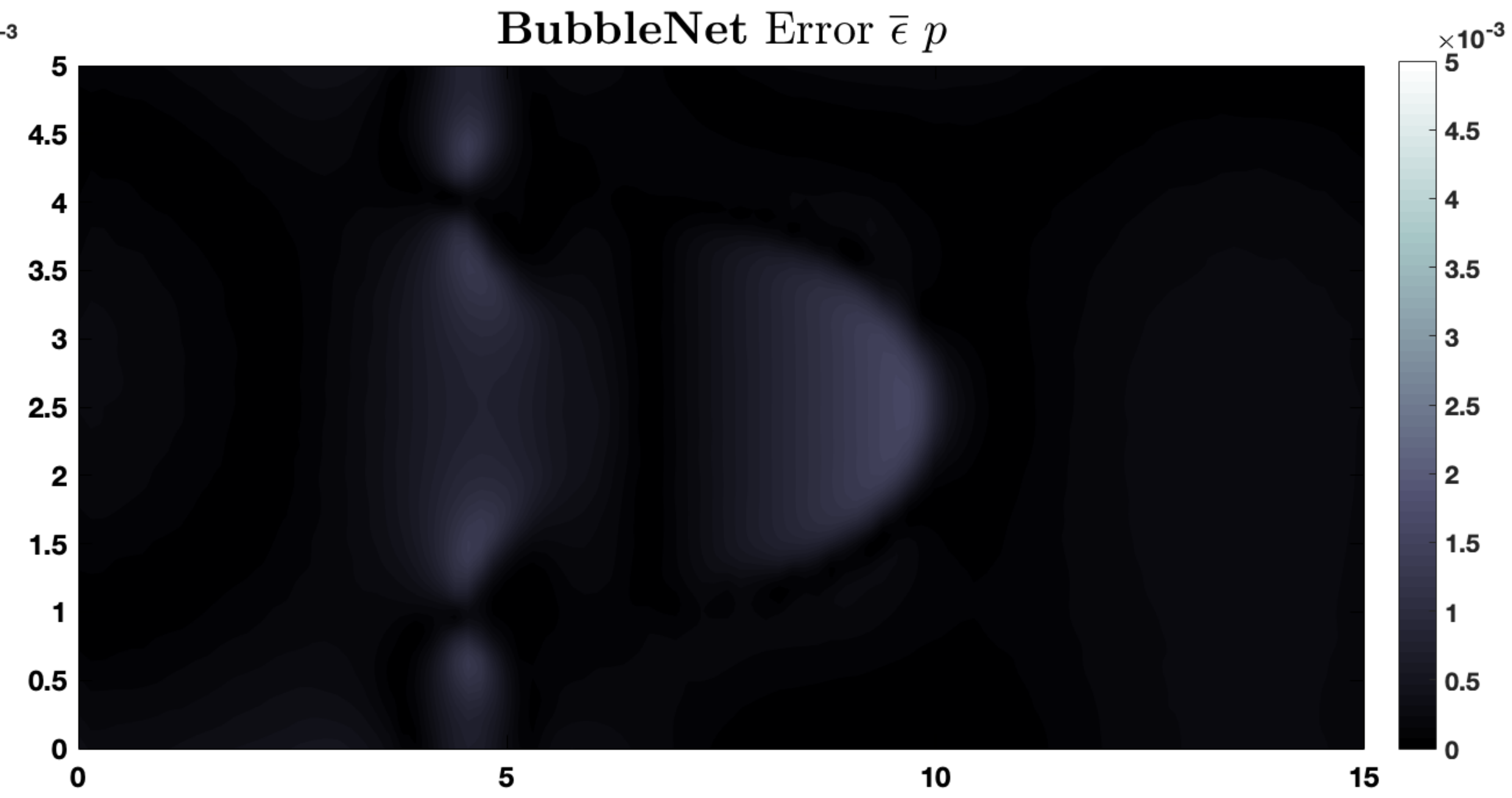


Error Analysis

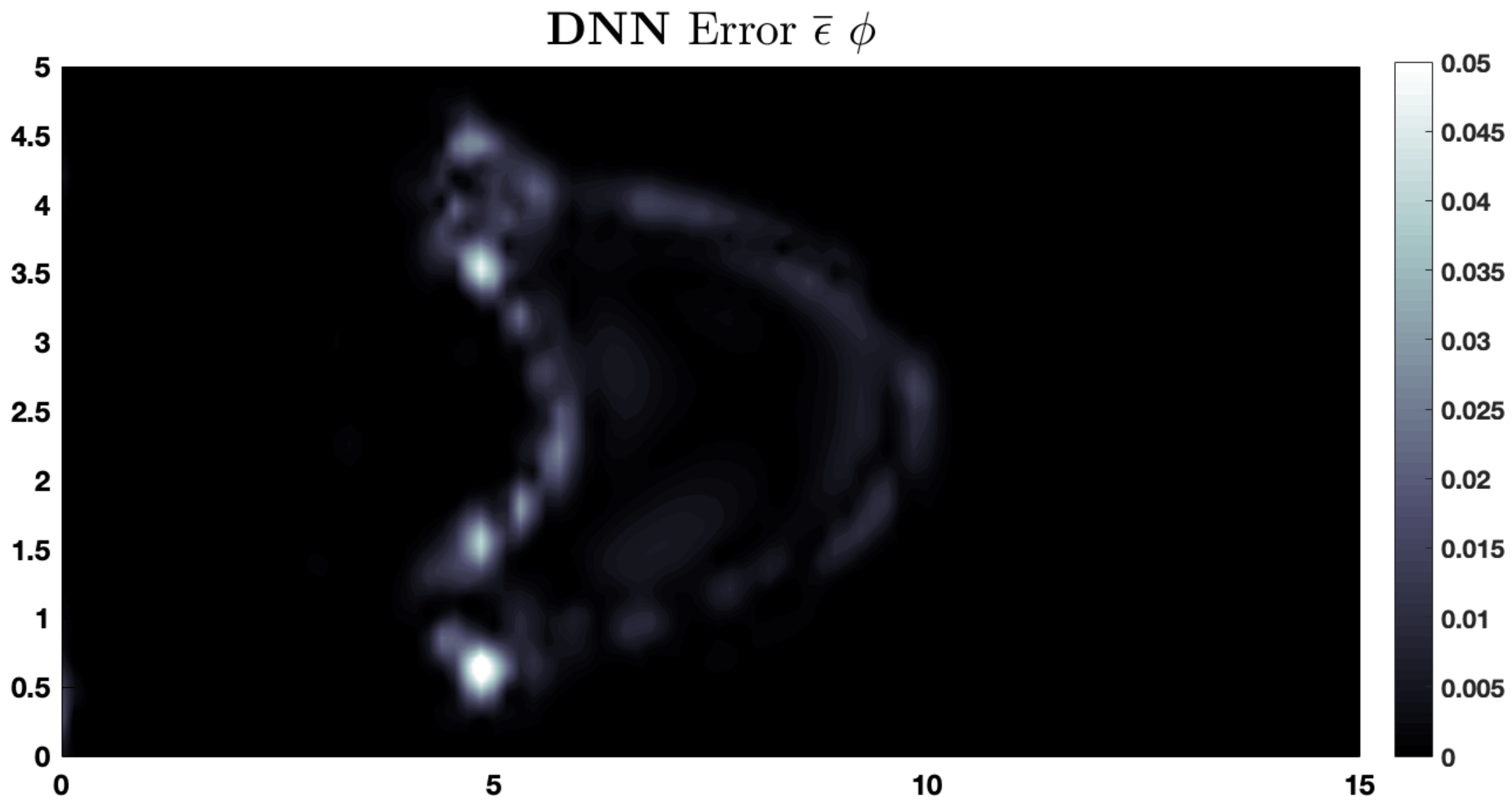


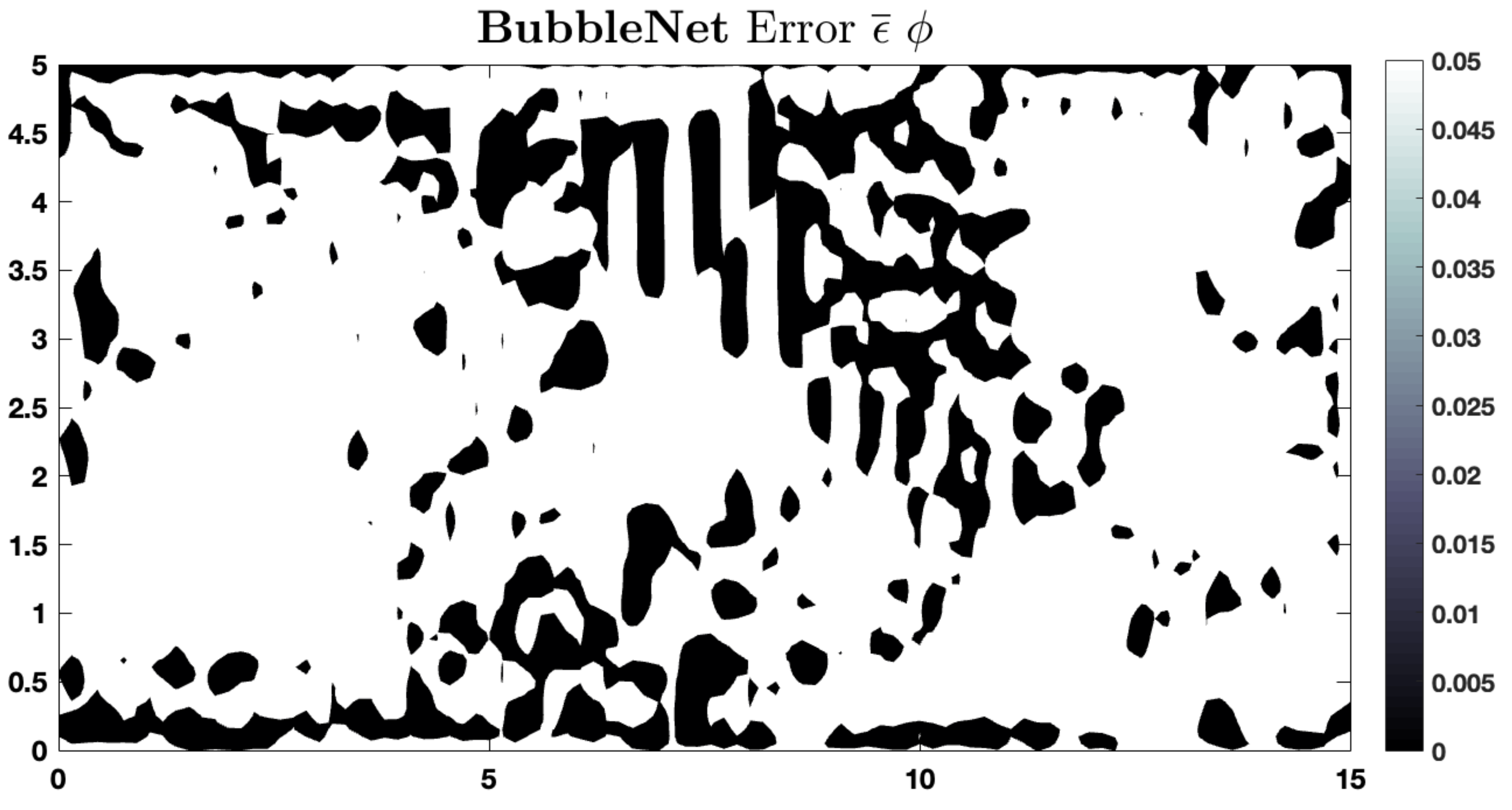
Error Analysis



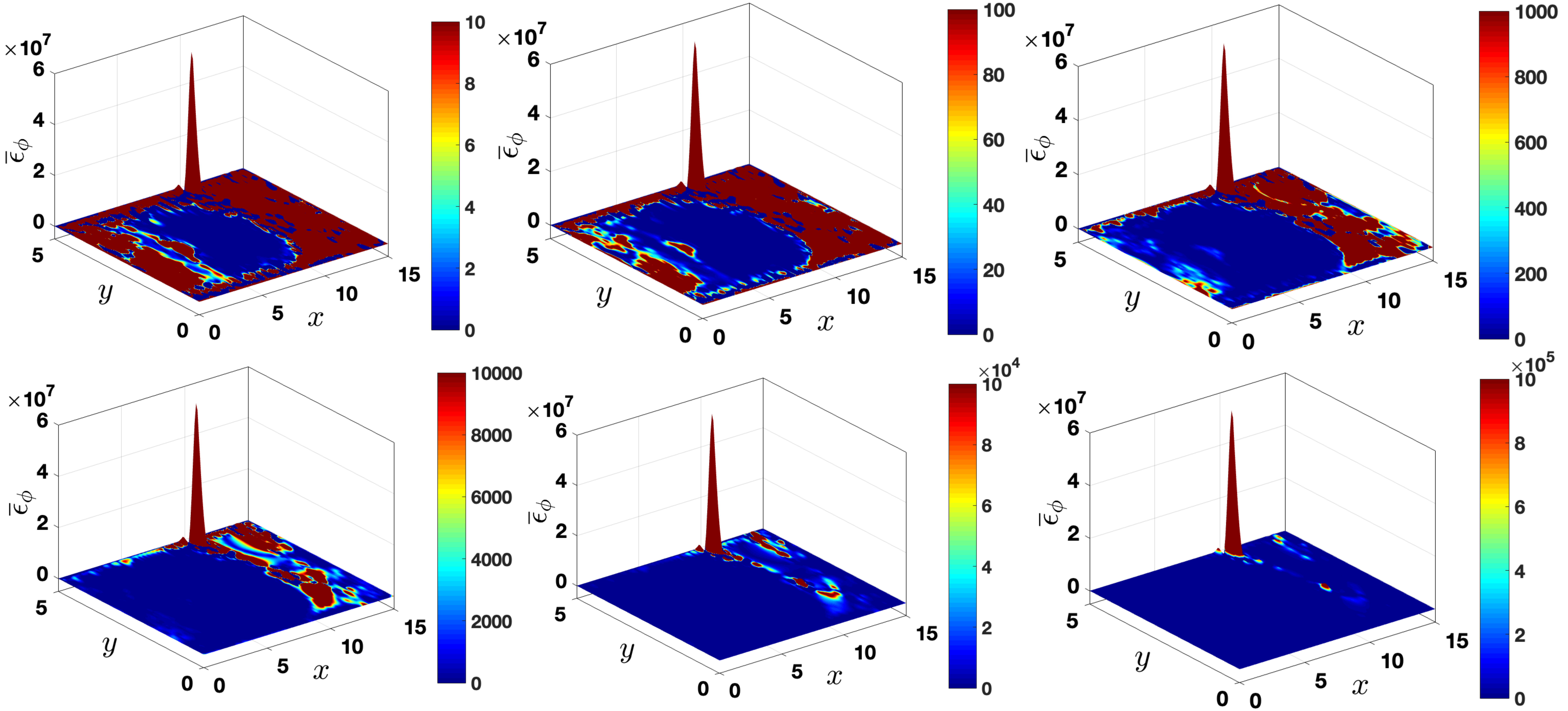


Error Analysis

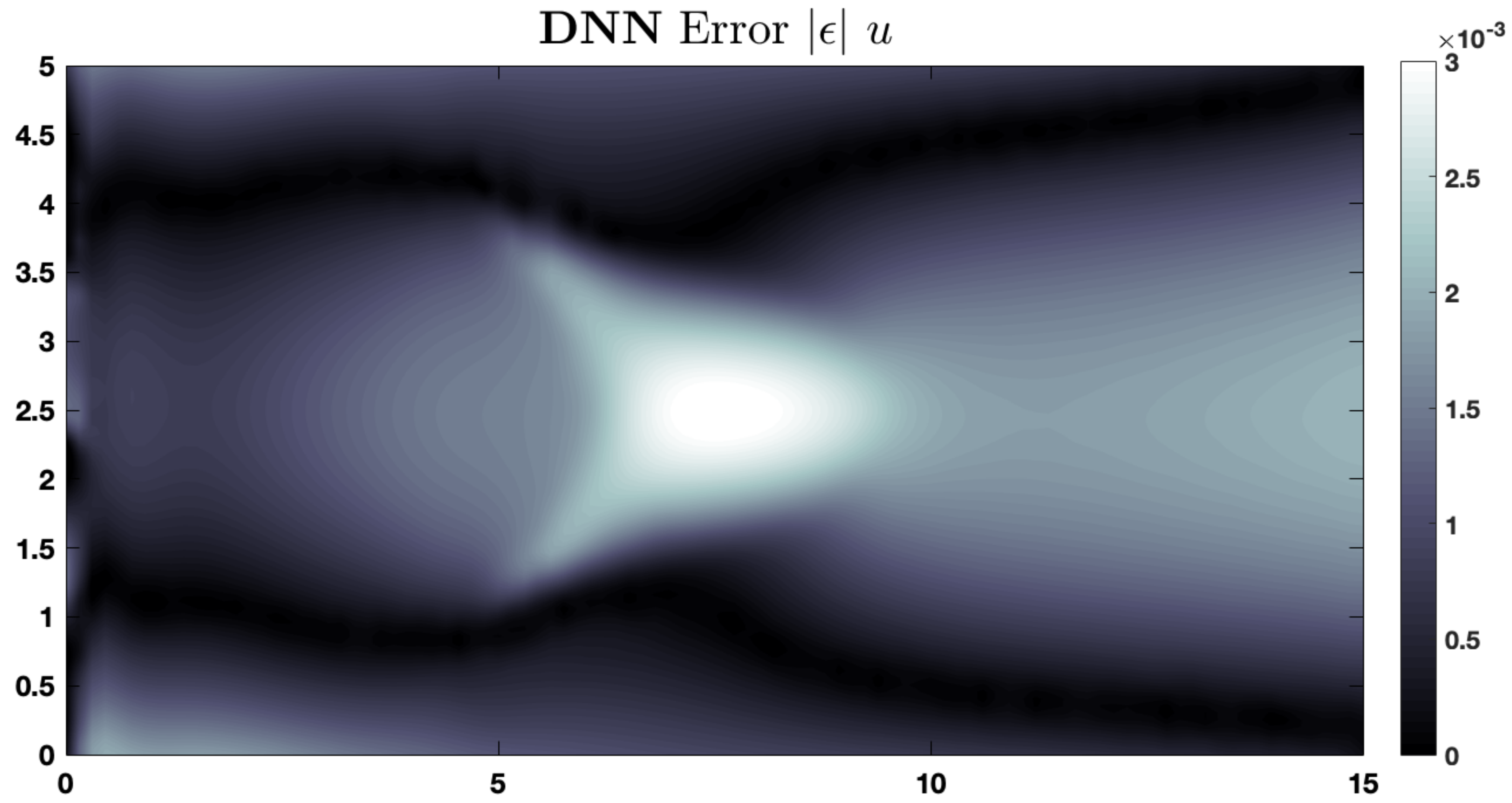


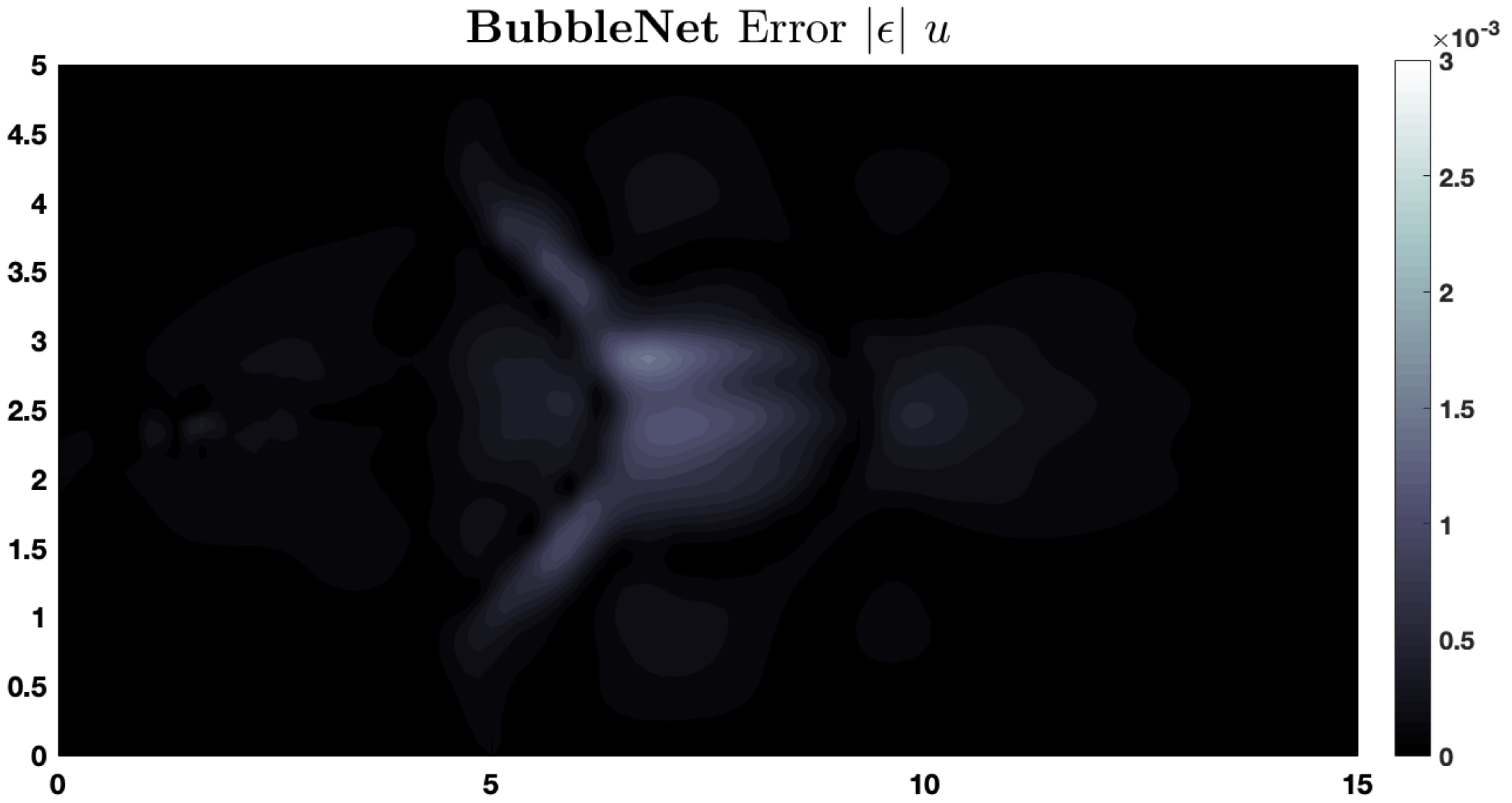


Error Analysis



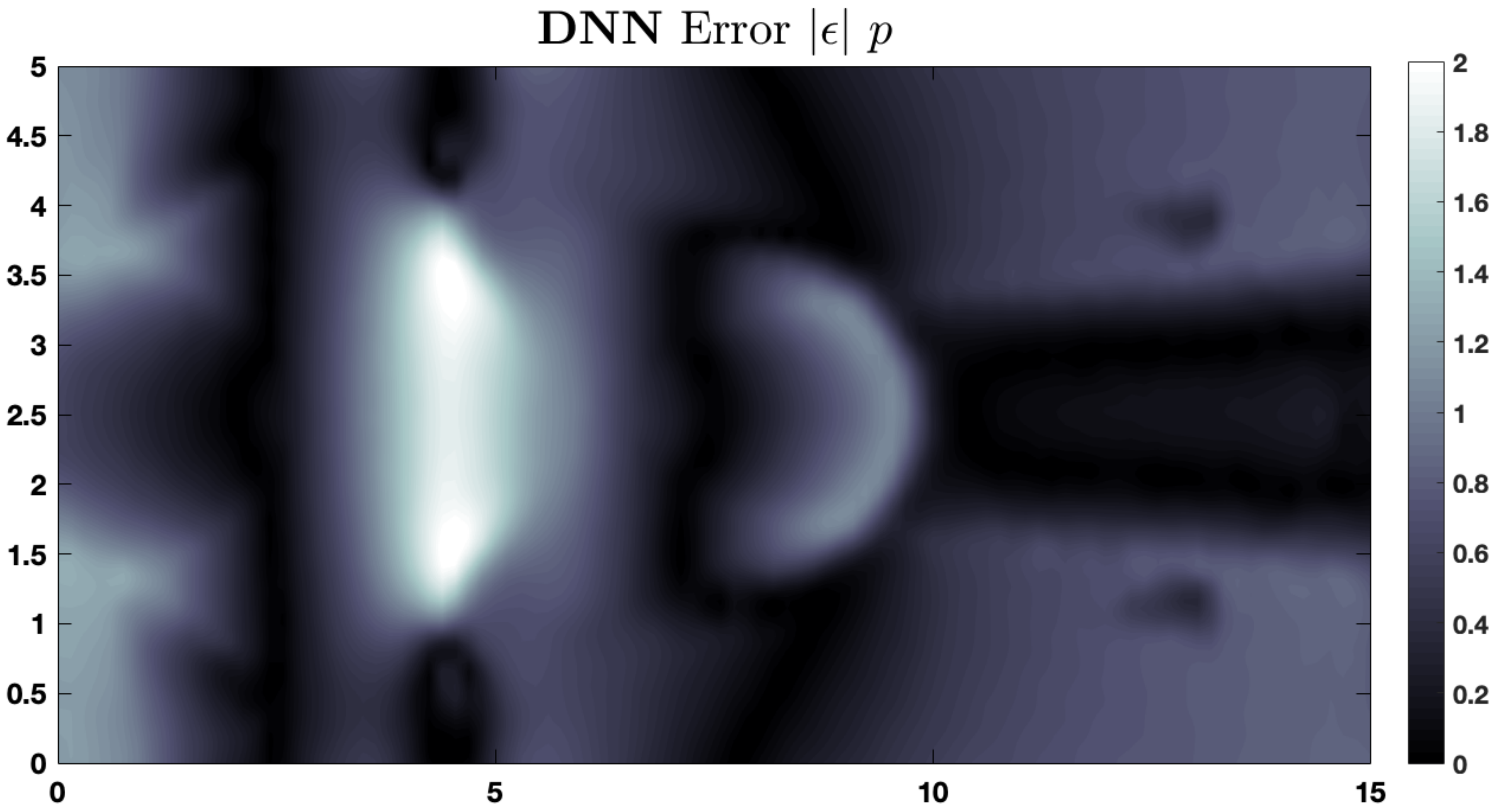
Error Analysis

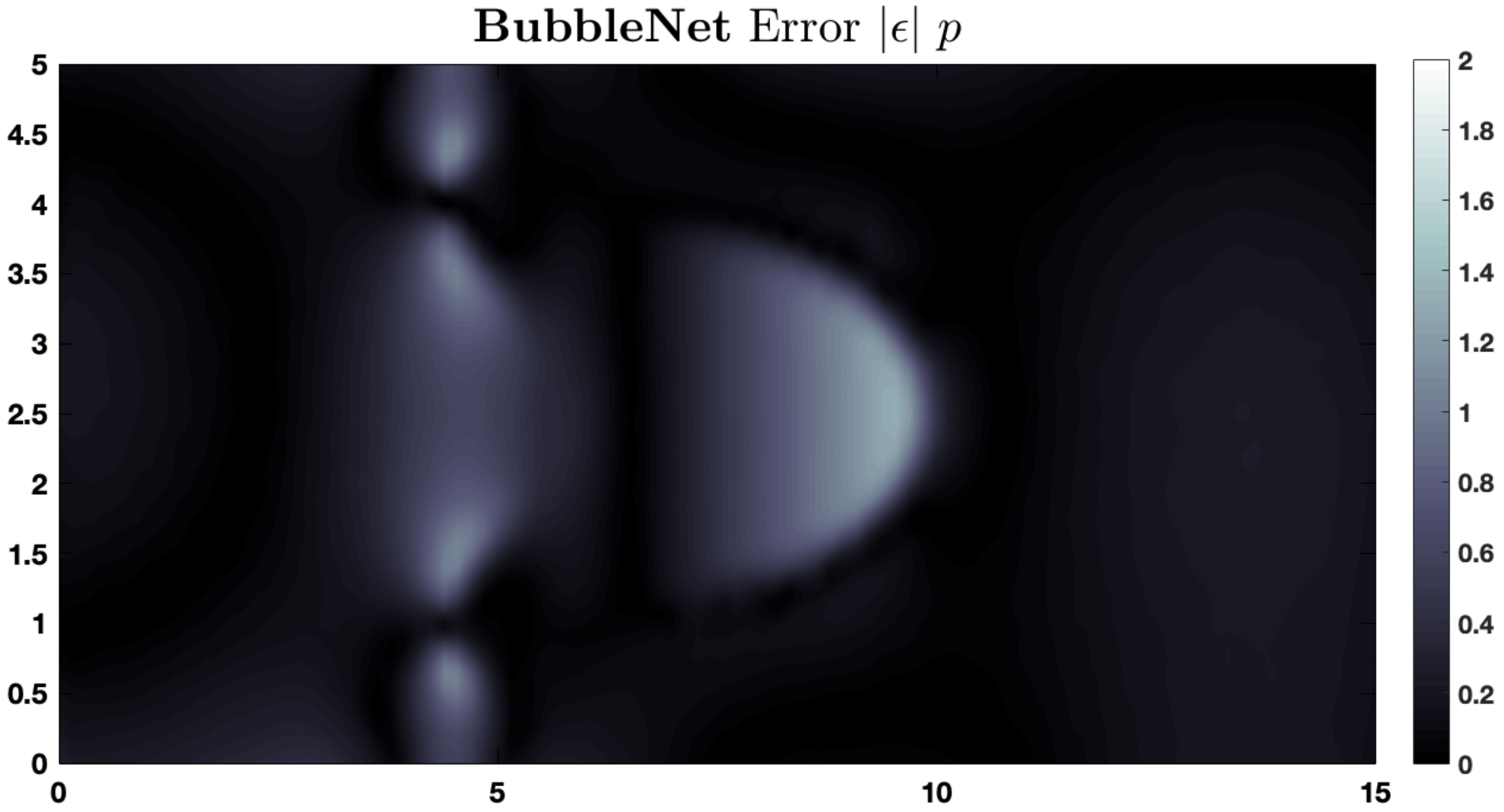




Error Analysis

Error Analysis





Error Analysis