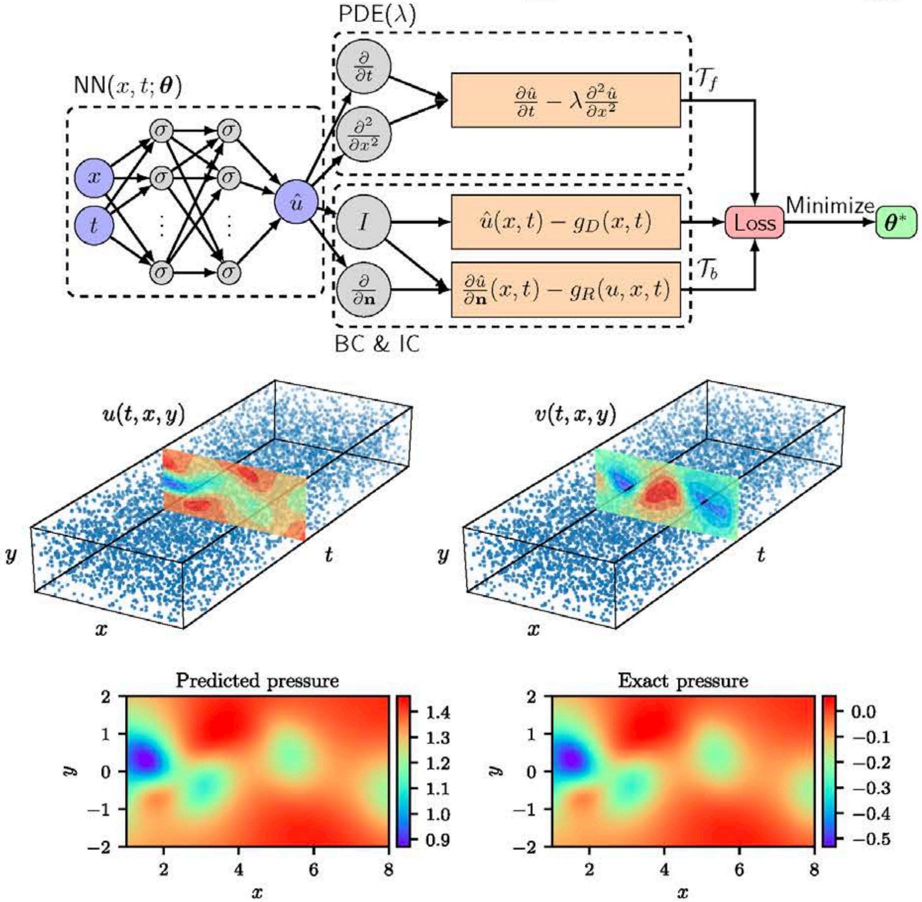
CIS522: Deep learning

Physics-informed deep learning

Paris Perdikaris
April 7, 2020

Penn
UNIVERSITY of PENNSYLVANIA

SC15-009: Recent Advances in Physics-Informed Deep Learning



Correct PDE	$u_t + (uu_x + vu_y) = -p_x + 0.01(u_{xx} + u_{yy})$ $v_t + (uv_x + vv_y) = -p_y + 0.01(v_{xx} + v_{yy})$
Identified PDE (clean data)	$u_t + 0.999(uu_x + vu_y) = -p_x + 0.01047(u_{xx} + u_{yy})$ $v_t + 0.999(uv_x + vv_y) = -p_y + 0.01047(v_{xx} + v_{yy})$
Identified PDE (1% noise)	$u_t + 0.998(uu_x + vu_y) = -p_x + 0.01057(u_{xx} + u_{yy})$ $v_t + 0.998(uv_x + vv_y) = -p_y + 0.01057(v_{xx} + v_{yy})$

Instructors:

- Paris Perdikaris (UPenn, pgp@seas.upenn.edu)
- Maziar Raissi (NVIDIA, maziar.raissi@gmail.com)

Schedule (Room 205)

Time	Lecturer	Topic
8.30-9.20am	Paris Perdikaris	Supervised learning with neural networks in Tensorflow
9.20-10.10am	Maziar Raissi	Physics-informed neural networks (Part I)
10.10-10.30am	Coffee Break	
10.30-11.20am	Paris Perdikaris	Physics-informed neural networks (Part II)
11.20-12.10pm	Maziar Raissi	Multi-step neural networks
12.10-1.00pm	Lunch Break	
1.00-1.50pm	Paris Perdikaris	PINNs on Graphs
1.50-2.20pm	Maziar Raissi	Hidden physics models
2.20-3.10pm	Paris Perdikaris	Physics-informed deep generative models
3.10-3.30pm	Coffee Break	
3.30-4.20pm	Maziar Raissi	Forward Backward Stochastic Neural Networks
4.20-5.10pm	Paris Perdikaris	Open challenges
5.10-5.30pm	Maziar Raissi	Summary and future work

https://github.com/PredictiveIntelligenceLab/USNCCM15-Short-Course-Recent-Advances-in-Physics-Informed-Deep-Learning

Motivation and open challenges

Goal: Predictive modeling, analysis and optimization of complex systems

ML ← Data — Prior knowledge → **CSE**

$$p(\theta|\mathcal{D}) \propto p(\mathcal{D}|\theta)p(\theta)$$

Challenges:

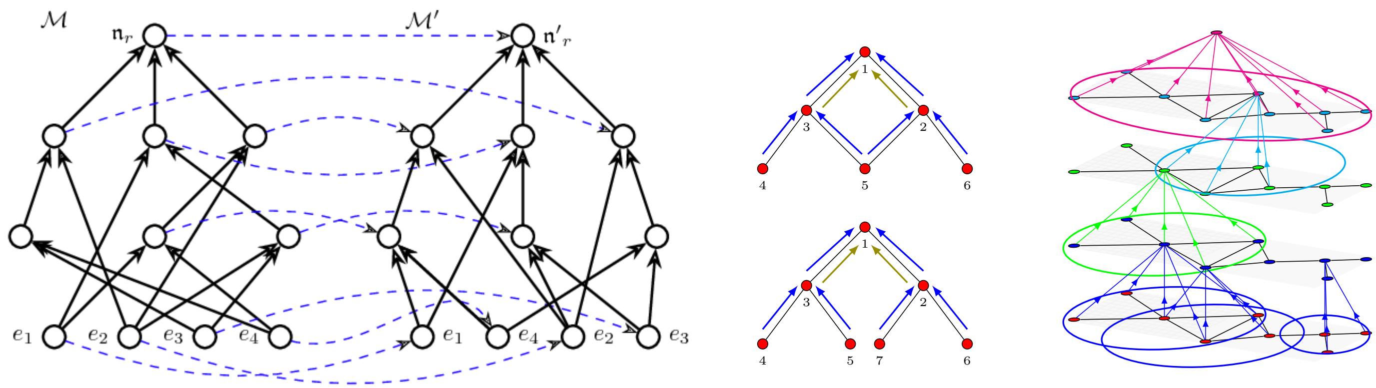
- High cost of data acquisition
- Limited and high-dimensional data
- Multiple tasks and data modalities (e.g. images, time-series, scattered measurements, etc.)
- Large parameter spaces
- Incomplete models, imperfect data (e.g., missing data, outliers, complex noise processes)
- Uncertainty quantification
- Robust design/control

Hypothesis:

- Can we bridge knowledge from scientific computing and machine learning to tackle these challenges?

Physics of AI: Two schools of thought

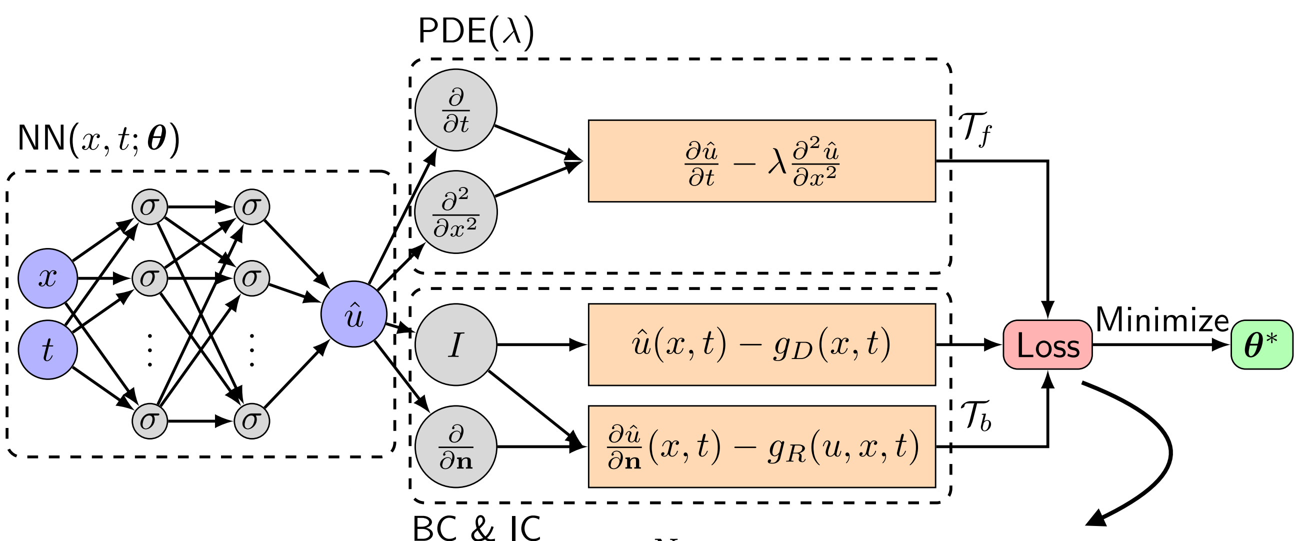
1. Physics is implicitly baked in specialized neural architectures with strong inductive biases (e.g. invariance to simple group symmetries).



**figures from Kondor, R., Son, H.T., Pan, H., Anderson, B., & Trivedi, S. (2018). Covariant compositional networks for learning graphs. arXiv preprint arXiv:1801.02144.*

2. Physics is explicitly imposed by constraining the output of conventional neural architectures with weak inductive biases.

Psichogios & Ungar, 1992
Lagaris et. al., 1998
Raissi et. al., 2019
Lu et. al., 2019
Zhu et. al., 2019

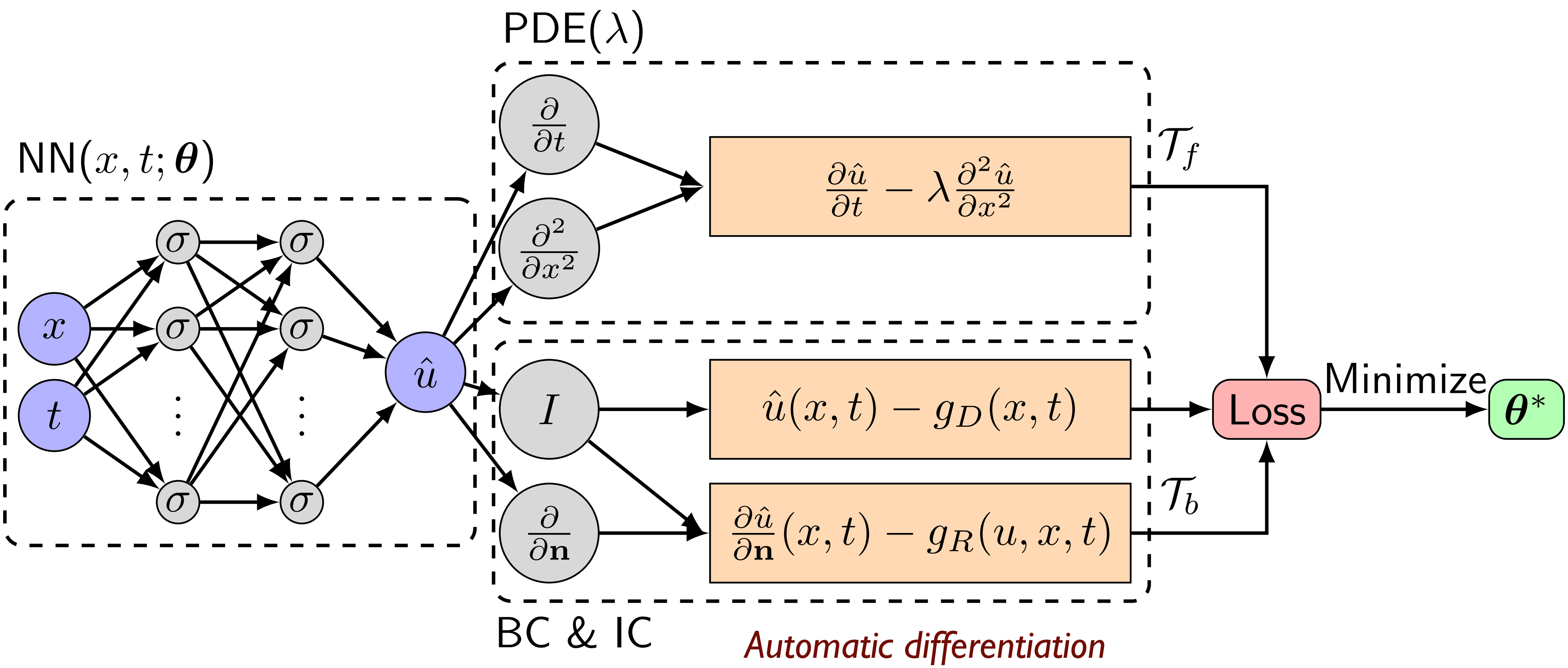


$$\mathcal{L}(\theta) := \underbrace{\frac{1}{N_u}\sum_{i=1}^{N_u}[\boldsymbol{u}_i - f_\theta(\boldsymbol{x}_i)]^2}_{\text{Data fit}} + \underbrace{\frac{1}{\lambda}\mathcal{R}[f_\theta(\boldsymbol{x})]}_{\text{Physics regularization}}$$

Physics-informed Neural Networks

$$f\left(\mathbf{x}; \frac{\partial u}{\partial x_1}, \ldots, \frac{\partial u}{\partial x_d}; \frac{\partial^2 u}{\partial x_1 \partial x_1}, \ldots, \frac{\partial^2 u}{\partial x_1 \partial x_d}; \ldots; \boldsymbol{\lambda}\right) = 0, \quad \mathbf{x} \in \Omega,$$

$$\mathcal{B}(u, \mathbf{x}) = 0 \quad \text{on} \quad \partial\Omega,$$



Psichogios, D. C., & Ungar, L. H. (1992). A hybrid neural network–first principles approach to process modeling. AIChE Journal, 38(10), 1499-1511.

Lagaris, I. E., Likas, A., & Fotiadis, D. I. (1998). Artificial neural networks for solving ordinary and partial differential equations. IEEE transactions on neural networks, 9(5), 987-1000.

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics, 378, 686-707.

Lu, L., Meng, X., Mao, Z., & Karniadakis, G. E. (2019). DeepXDE: A deep learning library for solving differential equations. arXiv preprint arXiv: 1907.04502.

General formulation of PINNs

Physics-informed neural networks (PINNs) aim at inferring a continuous latent function $\boldsymbol{u}(\boldsymbol{x}, t)$ that arises as the solution to a system of nonlinear partial differential equations (PDE) of the general form

$$\begin{aligned} &\boldsymbol{u}_t + \mathcal{N}_{\boldsymbol{x}}[\boldsymbol{u}] = 0, \quad \boldsymbol{x} \in \Omega, t \in [0, T] \\ &\boldsymbol{u}(\boldsymbol{x}, 0) = h(\boldsymbol{x}), \quad x \in \Omega \\ &\boldsymbol{u}(\boldsymbol{x}, t) = g(\boldsymbol{x}, t), \quad t \in [0, T], \quad \boldsymbol{x} \in \partial\Omega \end{aligned}$$

We proceed by approximating $\boldsymbol{u}(\boldsymbol{x}, t)$ by a deep neural network $f_\theta(\boldsymbol{x}, t)$, and define the residual of the PDE as

$$\boldsymbol{r}_\theta(\boldsymbol{x}, t) := \frac{\partial}{\partial t} f_\theta(\boldsymbol{x}, t) + \mathcal{N}_{\boldsymbol{x}}[f_\theta(\boldsymbol{x}, t)]$$

The corresponding loss function is given by

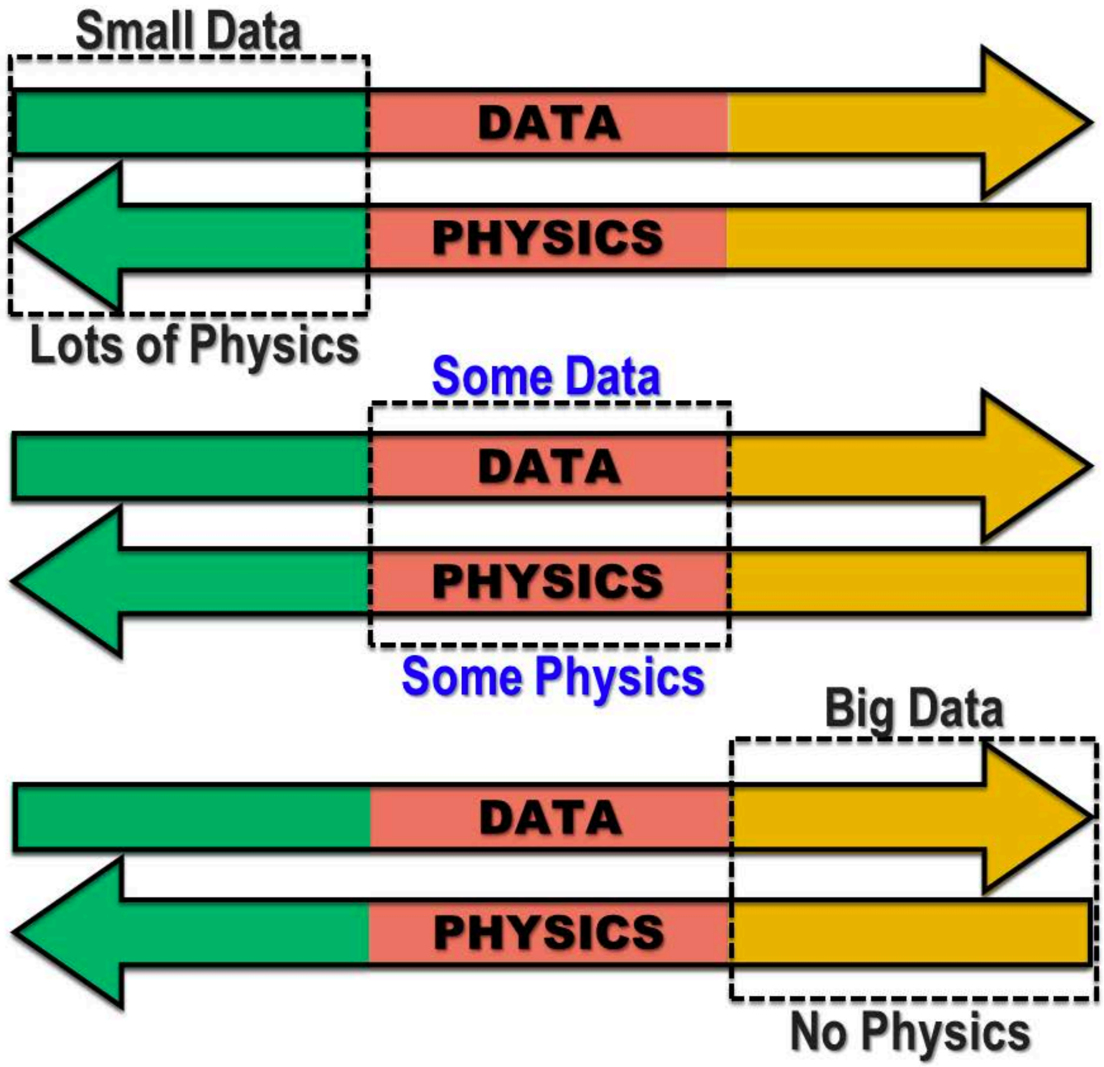
$$\mathcal{L}(\theta) := \underbrace{\mathcal{L}_u(\theta)}_{\text{Data fit}} + \underbrace{\mathcal{L}_r(\theta)}_{\text{PDE residual}} + \underbrace{\mathcal{L}_{u_0}(\theta)}_{\text{ICs fit}} + \underbrace{\mathcal{L}_{u_b}(\theta)}_{\text{BCs fit}}$$

Training via stochastic gradient descent:

$$\theta_{n+1} = \theta_n - \eta \nabla_\theta \mathcal{L}(\theta_n)$$

***all gradients are computed via automatic differentiation**

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics, 378, 686-707.


Small Data
DATA
PHYSICS
Lots of Physics
Some Data
DATA
PHYSICS
Some Physics
Big Data
DATA
PHYSICS
No Physics

Physics-informed Neural Networks

Example: Burgers' equation in 1D

$$u_t + uu_x - (0.01/\pi)u_{xx} = 0, \quad x \in [-1, 1], \quad t \in [0, 1], \tag{3}$$
$$u(0, x) = -\sin(\pi x),$$
$$u(t, -1) = u(t, 1) = 0.$$

Let us define $f(t, x)$ to be given by

$$f := u_t + uu_x - (0.01/\pi)u_{xx},$$

```
def u(t, x):
    u = neural_net(tf.concat([t,x],1), weights, biases)
    return u
```

Correspondingly, the *physics informed neural network* $f(t, x)$ takes the form

```
def f(t, x):
    u = u(t, x)
    u_t = tf.gradients(u, t)[0]
    u_x = tf.gradients(u, x)[0]
    u_xx = tf.gradients(u_x, x)[0]
    f = u_t + u*u_x - (0.01/tf.pi)*u_xx
    return f
```

Physics-informed Neural Networks

The shared parameters between the neural networks $u(t,x)$ and $f(t,x)$ can be learned by minimizing the mean squared error loss

$$MSE = MSE_u + MSE_f, \tag{4}$$

where

$$MSE_u = \frac{1}{N_u}\sum_{i=1}^{N_u} |u(t_u^i, x_u^i) - u^i|^2,$$

and

$$MSE_f = \frac{1}{N_f}\sum_{i=1}^{N_f} |f(t_f^i, x_f^i)|^2.$$

Here, $\{t_u^i, x_u^i, u^i\}_{i=1}^{N_u}$ denote the initial and boundary training data on $u(t,x)$ and $\{t_f^i, x_f^i\}_{i=1}^{N_f}$ specify the collocations points for $f(t,x)$. The loss MSE_u corresponds to the initial and boundary data while MSE_f enforces the structure imposed by equation (3) at a finite set of collocation points.

Physics-informed Neural Networks

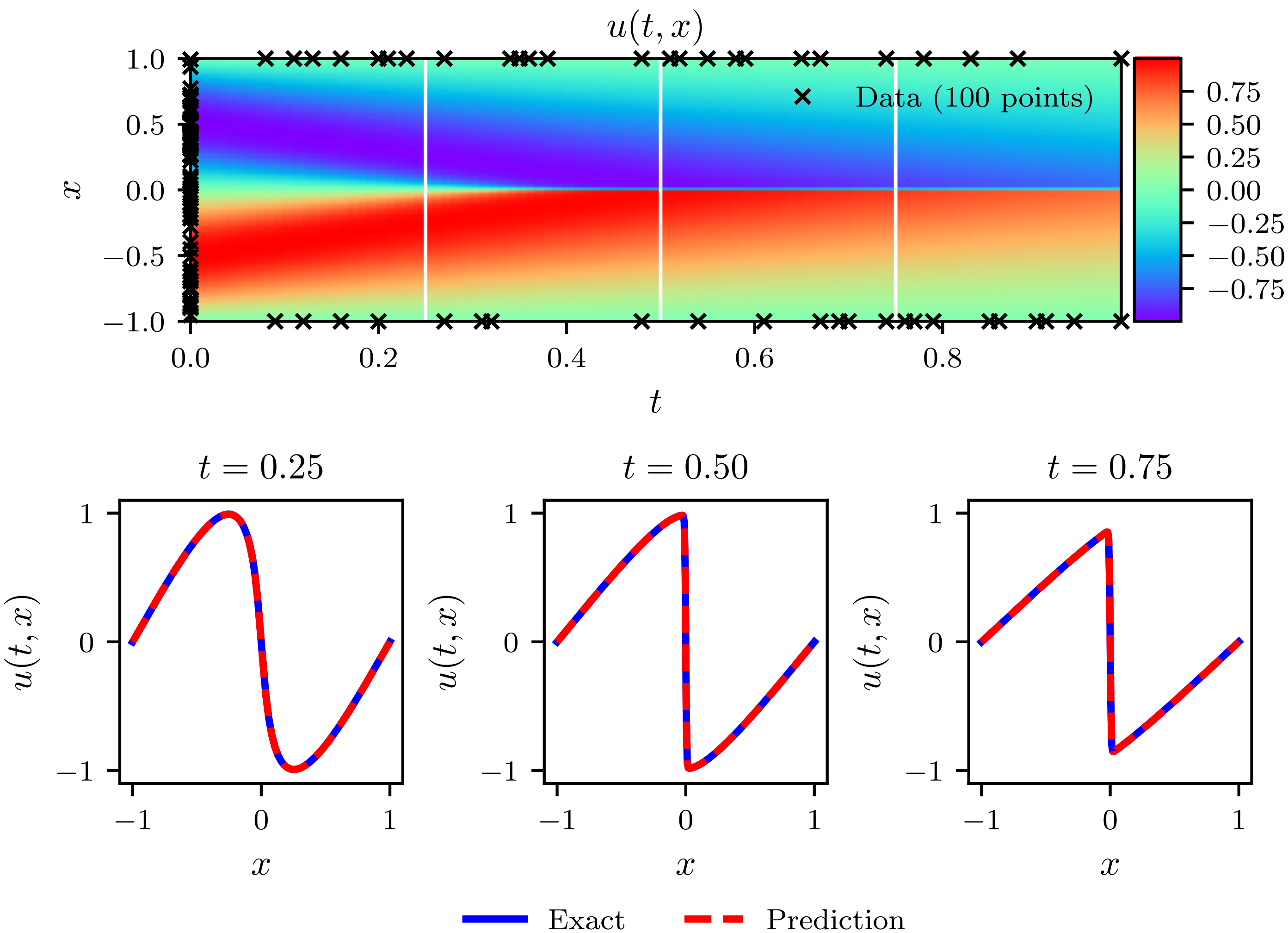
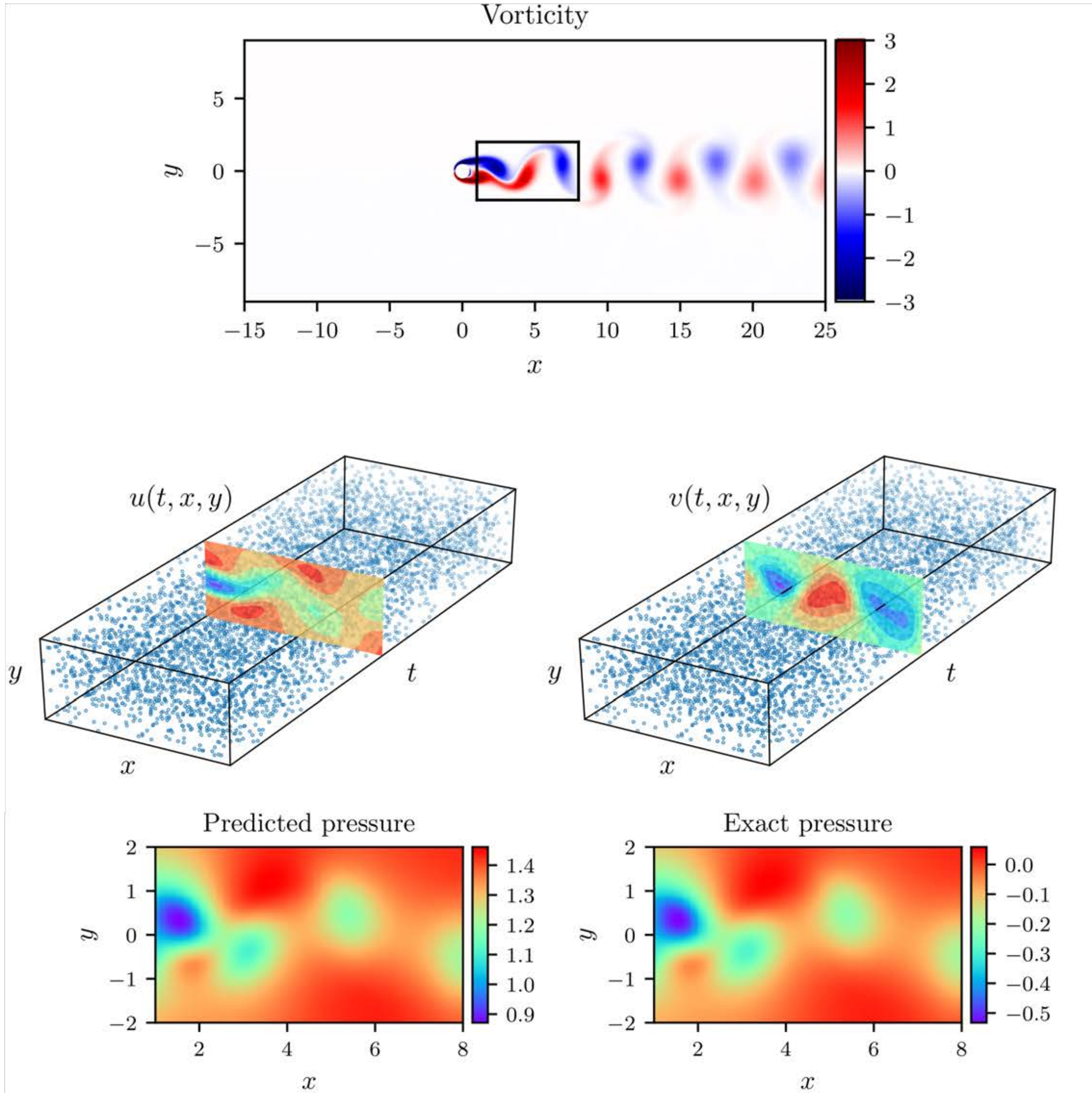


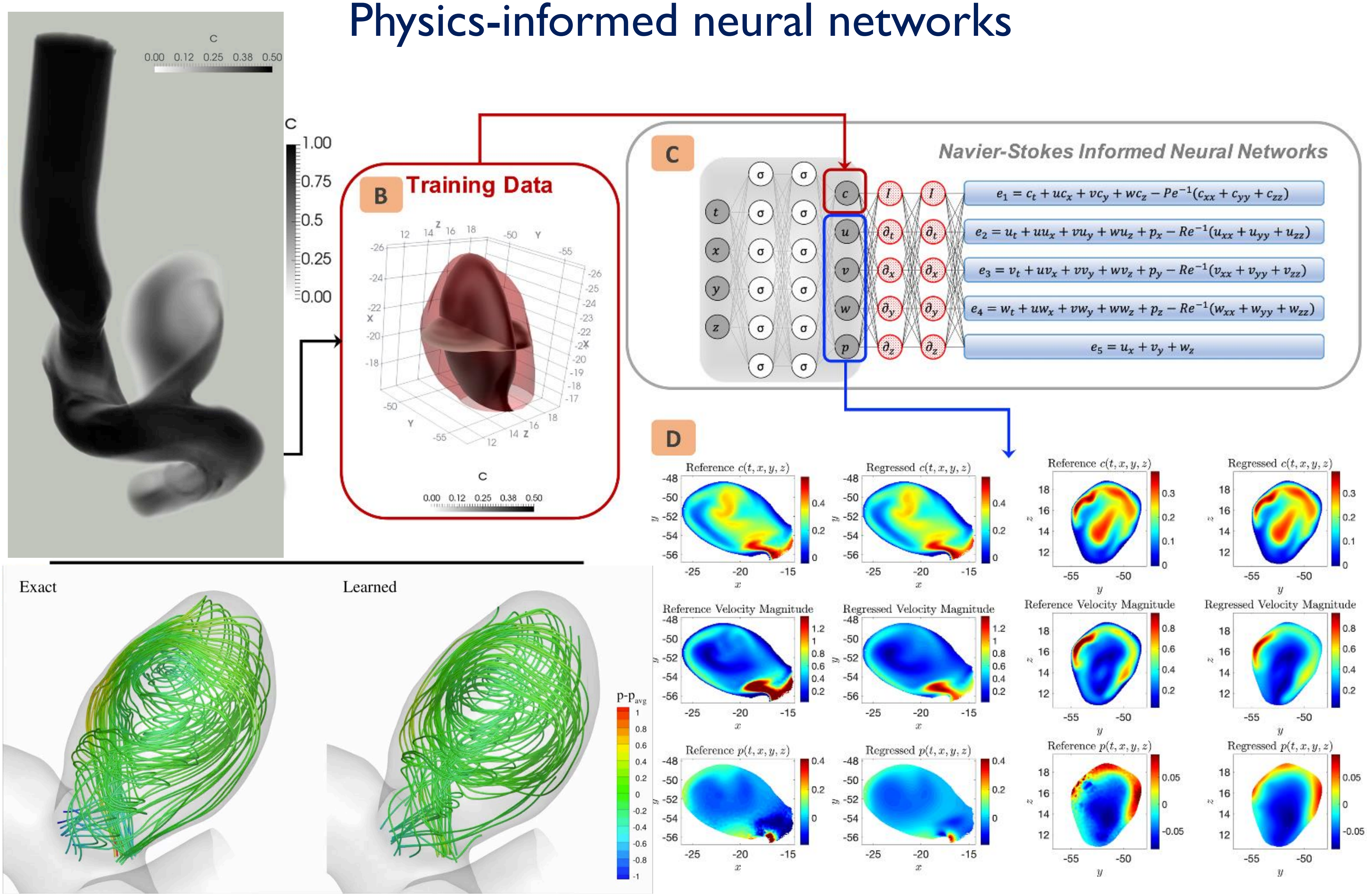
Figure 1: *Burgers' equation: Top:* Predicted solution $u(t, x)$ along with the initial and boundary training data. In addition we are using 10,000 collocation points generated using a Latin Hypercube Sampling strategy. *Bottom:* Comparison of the predicted and exact solutions corresponding to the three temporal snapshots depicted by the white vertical lines in the top panel. The relative $\mathcal{L}_2$ error for this case is $6.7 \cdot 10^{-4}$. Model training took approximately 60 seconds on a single NVIDIA Titan X GPU card.

Physics-informed Neural Networks



Correct PDE	$u_t + (uu_x + vu_y) = -p_x + 0.01(u_{xx} + u_{yy})$ $v_t + (uv_x + vv_y) = -p_y + 0.01(v_{xx} + v_{yy})$
Identified PDE (clean data)	$u_t + 0.999(uu_x + vu_y) = -p_x + 0.01047(u_{xx} + u_{yy})$ $v_t + 0.999(uv_x + vv_y) = -p_y + 0.01047(v_{xx} + v_{yy})$
Identified PDE (1% noise)	$u_t + 0.998(uu_x + vu_y) = -p_x + 0.01057(u_{xx} + u_{yy})$ $v_t + 0.998(uv_x + vv_y) = -p_y + 0.01057(v_{xx} + v_{yy})$

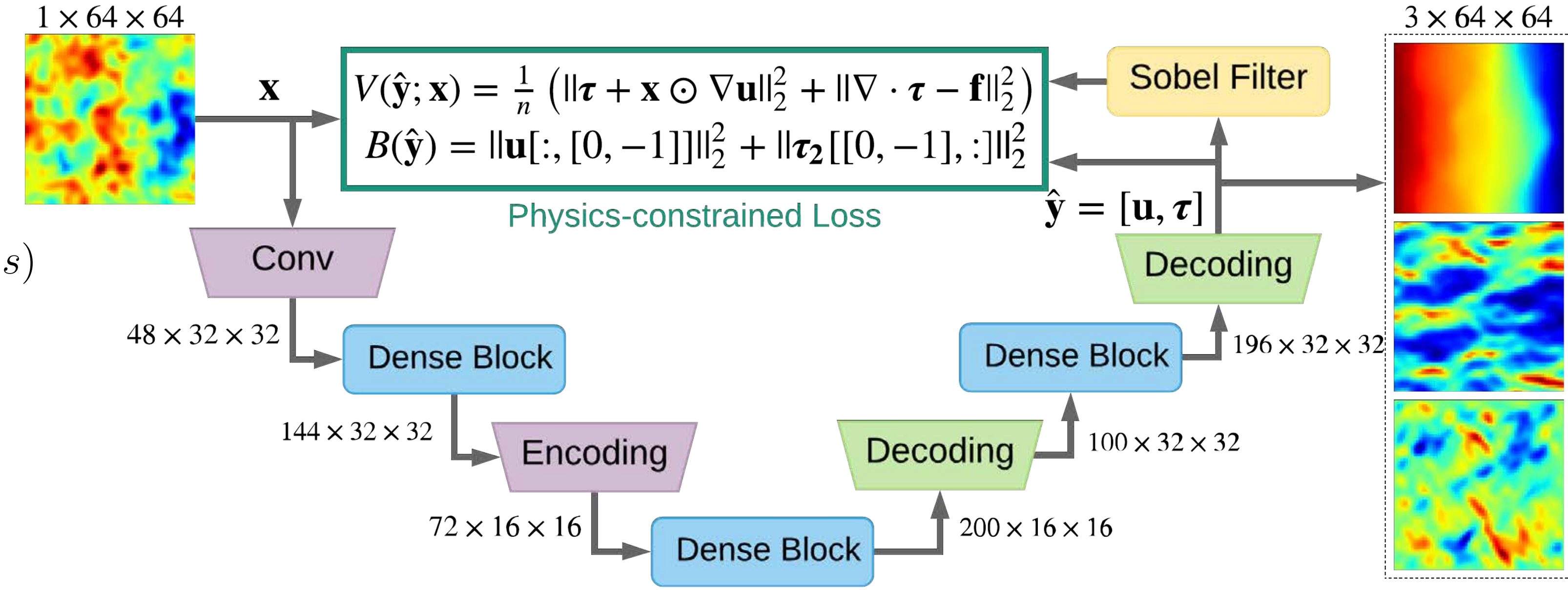
Physics-informed neural networks

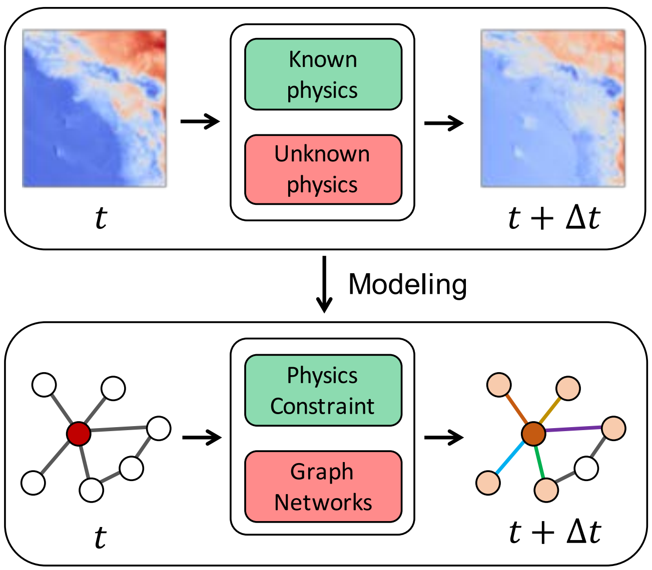


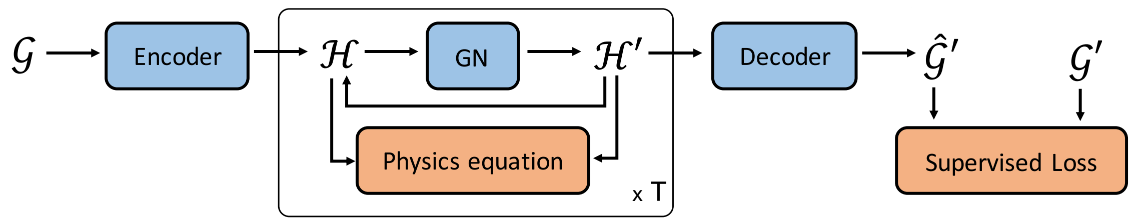
Raissi, M., Yazdani, A., & Karniadakis, G. E. (2020). Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. Science.

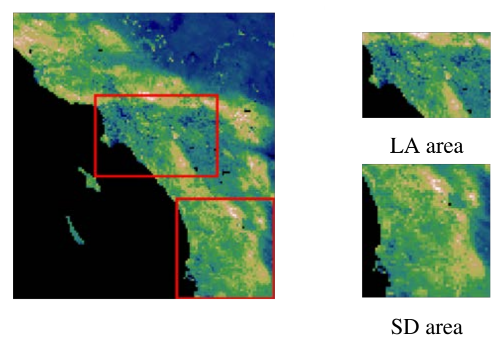
Extensions to CNNs and GCNs

$$-\nabla \cdot (K(s)\nabla u(s)) = f(s)$$
$$u(s) = u_D(s),$$
$$\nabla u(s) \cdot n = g(s),$$





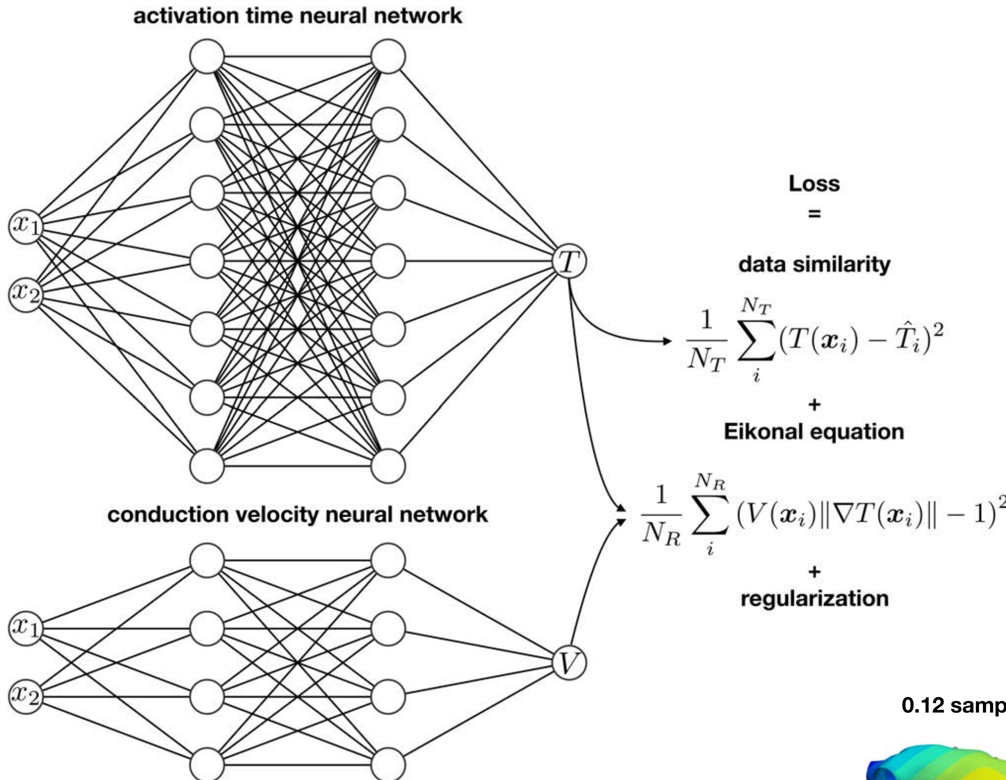




Zhu, Y., Zabaras, N., Koutsourelakis, P. S., & Perdikaris, P. (2019). Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. Journal of Computational Physics, 394, 56-81.

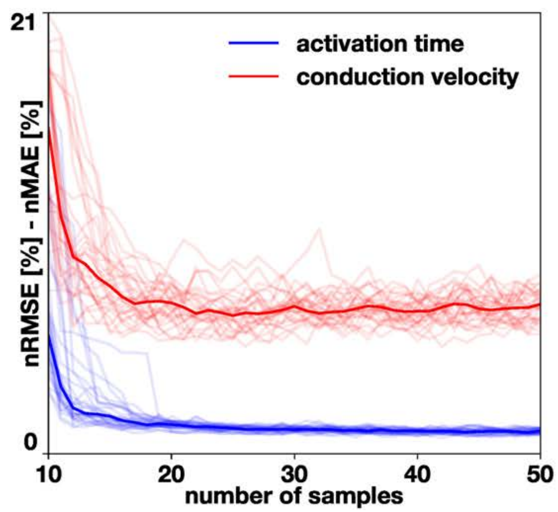
Seo, S., & Liu, Y. (2019). Differentiable physics-informed graph networks. arXiv preprint arXiv:1902.02950.

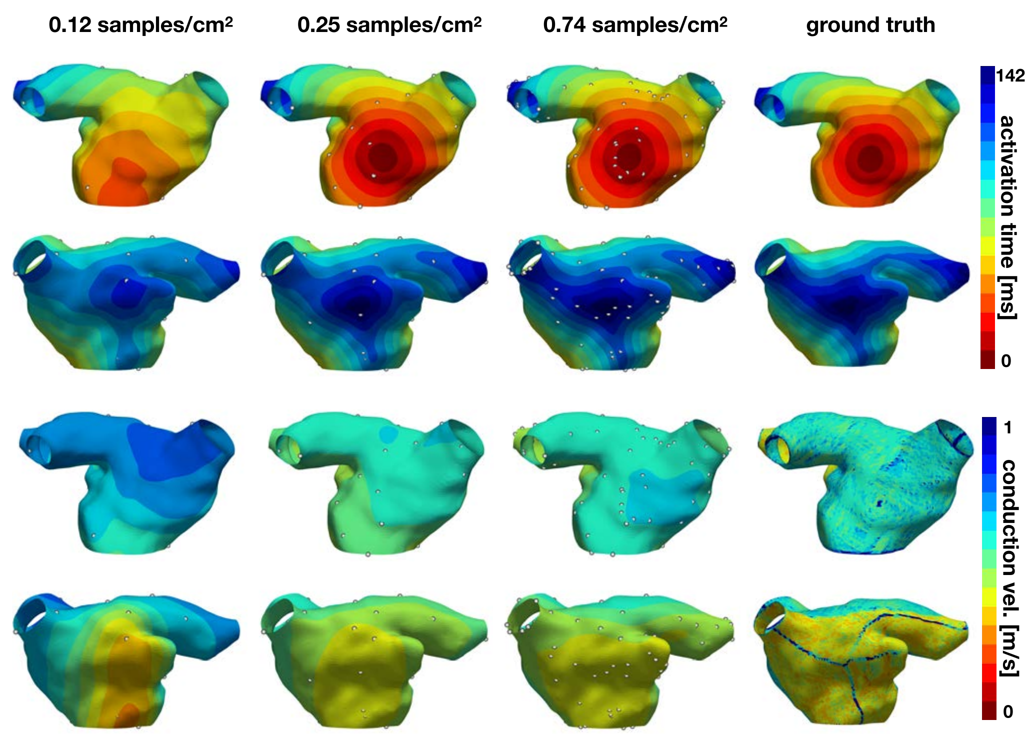
Physics-informed deep learning in cardiac electrophysiology



Algorithm 1: Active learning algorithm to iteratively identify the most efficient sampling points

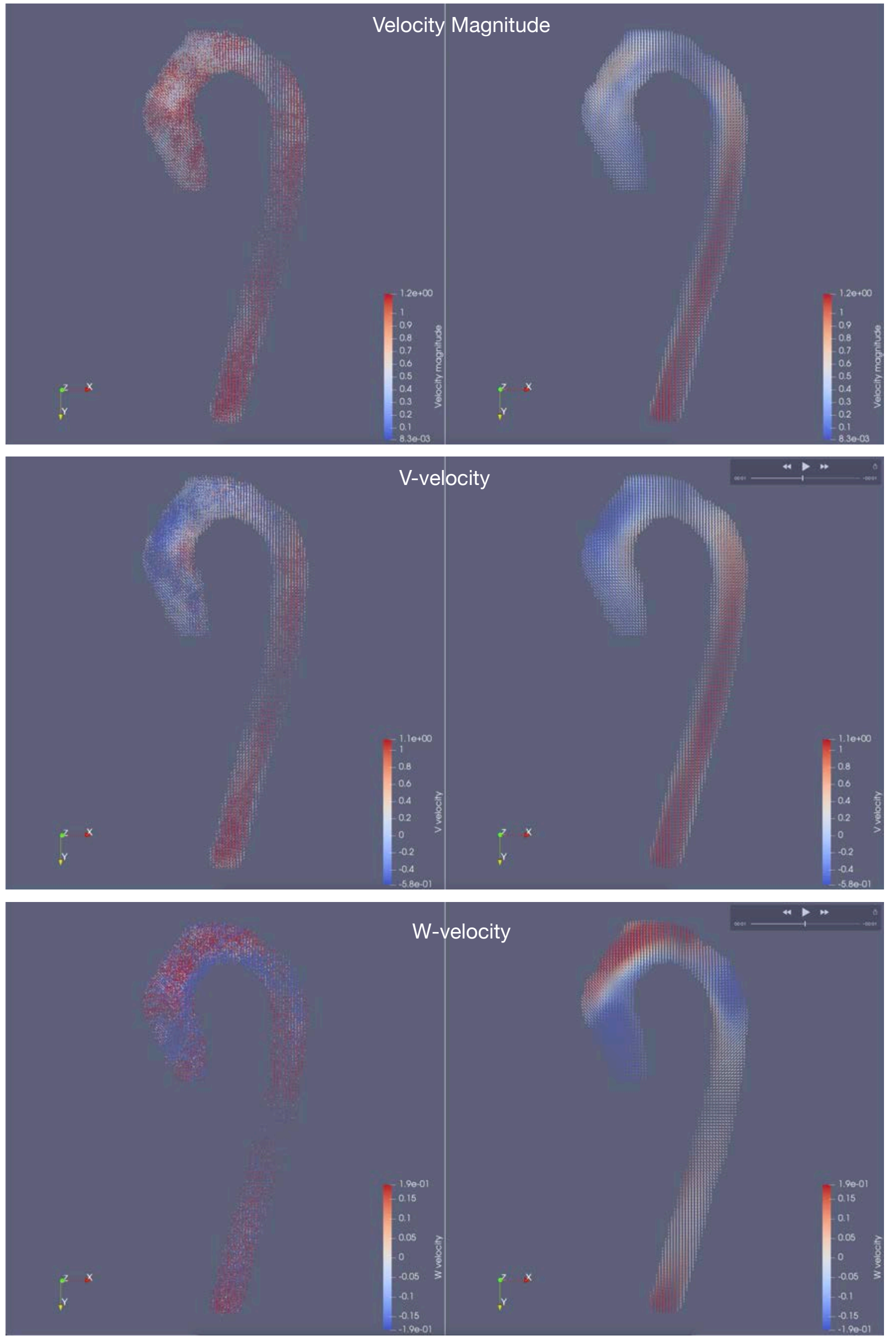
Given: number of initial samples N_{init}, number of active learning samples N_{AL}, set of candidate locations $\boldsymbol{X}_{cand}$, number of initial training iterations M_{init}, number of active learning training iterations M_{AL}, and empty sets $\boldsymbol{X}$ and $\boldsymbol{T}$ that contain locations and activations times:
Randomly select N_{init} samples from $\boldsymbol{X}_{cand}$
Remove the N_{init} samples from $\boldsymbol{X}_{cand}$ and add them to $\boldsymbol{X}$
Acquire the values of the activation times at the N_{init} locations and add them to $\boldsymbol{T}$
Initialize the model and train it using the ADAM optimizer [25] for M_{init} iterations.
for $i = \{1, N_{AL}\}$ **do**
  compute entropy $H(\boldsymbol{X}_{cand})$
  find the new location of maximum entropy:
  $\arg\max_{\boldsymbol{x}\in\boldsymbol{X}_{cand}} H(\boldsymbol{x})$
  remove $\boldsymbol{x}$ from $\boldsymbol{X}_{cand}$ and add it to $\boldsymbol{X}$
  acquire activation time at $\boldsymbol{x}$ and add it to $\boldsymbol{T}$ train the model using ADAM [25] for M_{AL} iterations.
end





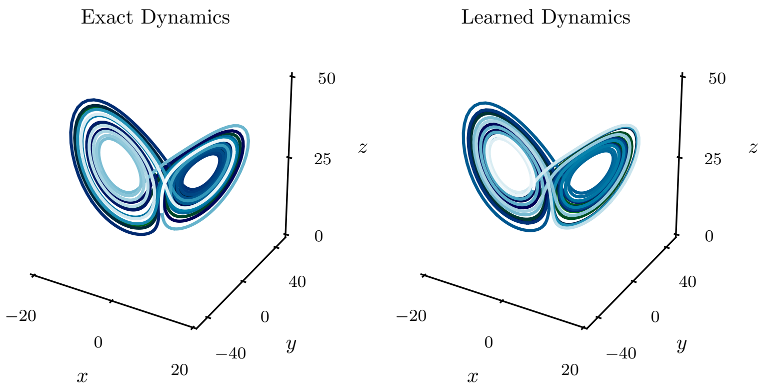
Sahli Costabal, F., Yang, Y., Perdikaris, P., Hurtado, D. E., & Kuhl, E. (2020). Physics-informed neural networks for cardiac activation mapping. Frontiers in Physics, 8, 42.

Physics-informed filtering of 4D-flow MRI



Recent advances

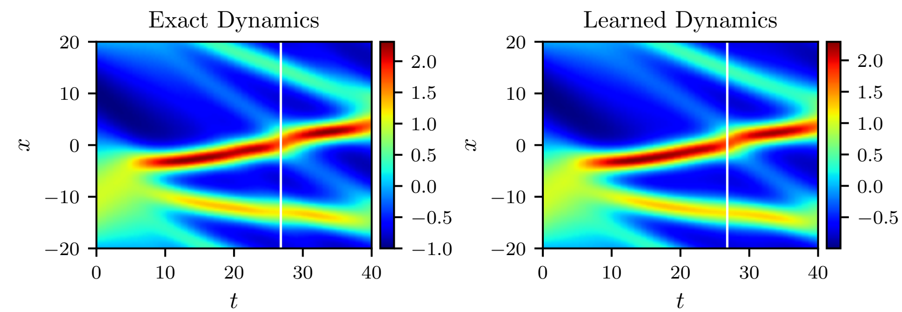
Discovery of ODEs



Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2018). Multistep Neural Networks for Data-driven Discovery of Nonlinear Dynamical Systems. arXiv preprint

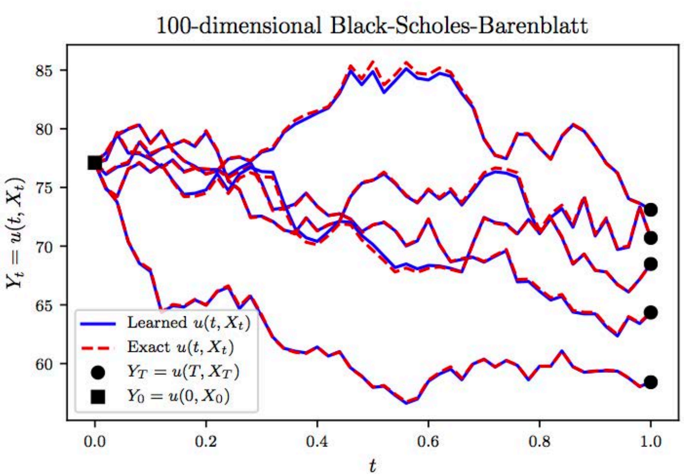
Discovery of PDEs

$$u_t = -uu_x - u_{xxx}$$



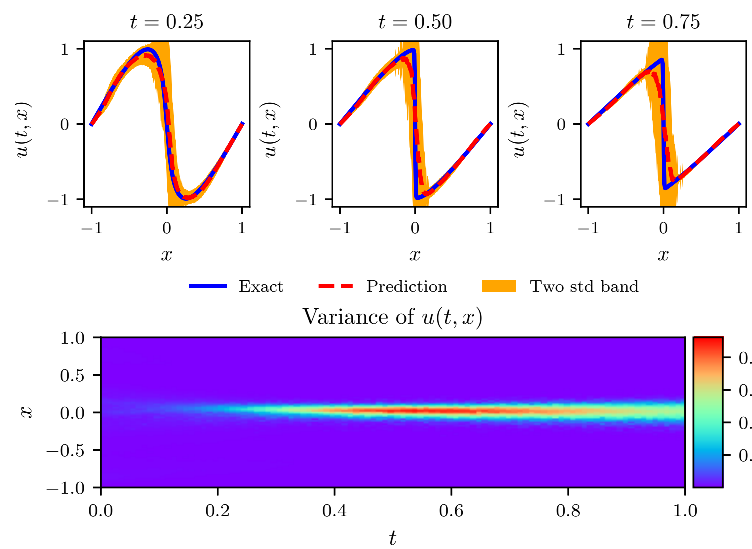
Raissi, M. (2018). Deep Hidden Physics Models: Deep Learning of Nonlinear Partial Differential Equations. arXiv preprint arXiv:1801.06637.

High-dimensional PDEs



Raissi, M. (2018). Forward-backward stochastic neural networks: Deep learning of high-dimensional partial differential equations. arXiv preprint arXiv: 1804.07010.

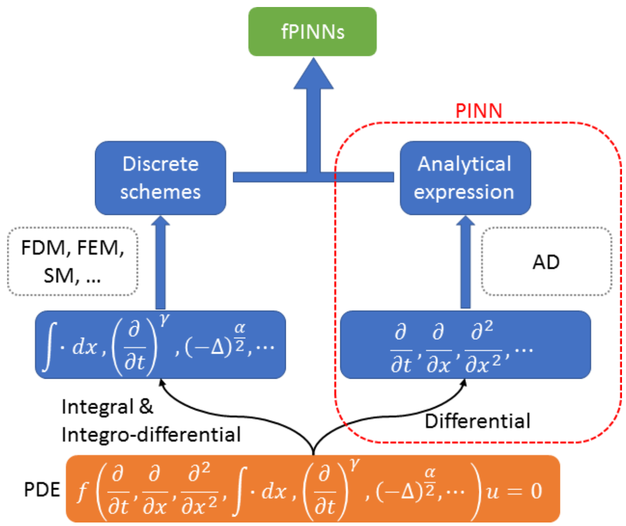
Stochastic PDEs



Yang, Y., & Perdikaris, P. (2019). Adversarial uncertainty quantification in physics-informed neural networks. Journal of Computational Physics.

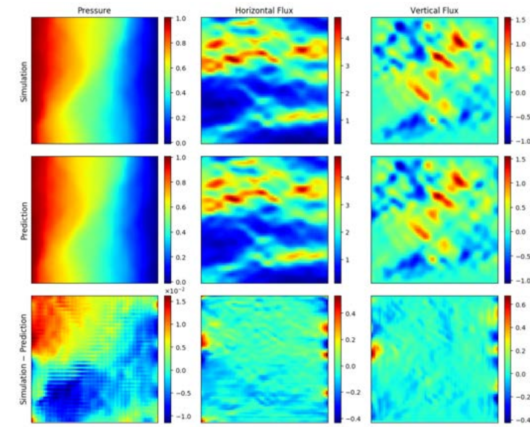
Recent advances

Fractional PDEs



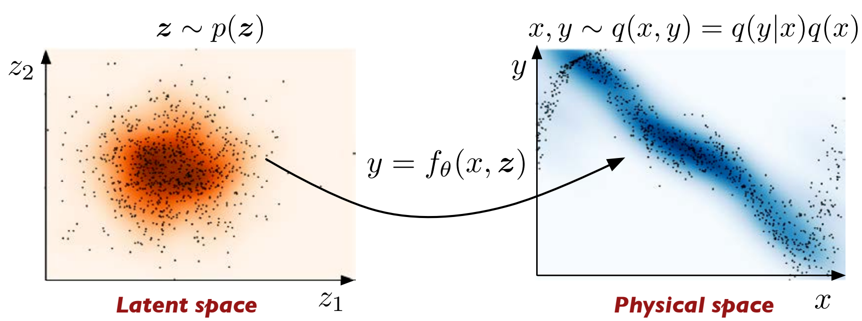
Pang, G., Lu, L., & Karniadakis, G. E. (2018). fpinns: Fractional physics-informed neural networks. arXiv preprint arXiv: 1811.08967.

Surrogate modeling & high-dimensional UQ



Zhu, Y., Zabaras, N., Koutsourelakis, P. S., & Perdikaris, P. (2019). Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. Journal of Computational Physics, 394, 56-81.

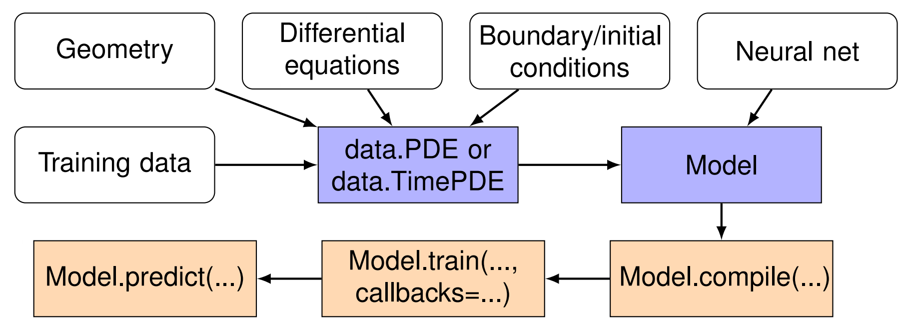
Multi-fidelity modeling for stochastic systems



$$y = f_\theta(x, \boldsymbol{z}),\ \boldsymbol{z} \sim p(\boldsymbol{z}) \ \Leftrightarrow \ y \sim p_\theta(y|x, \boldsymbol{z})$$

Conditional deep surrogate models for stochastic, high-dimensional, and multi-fidelity systems. Computational Mechanics, 1-18.

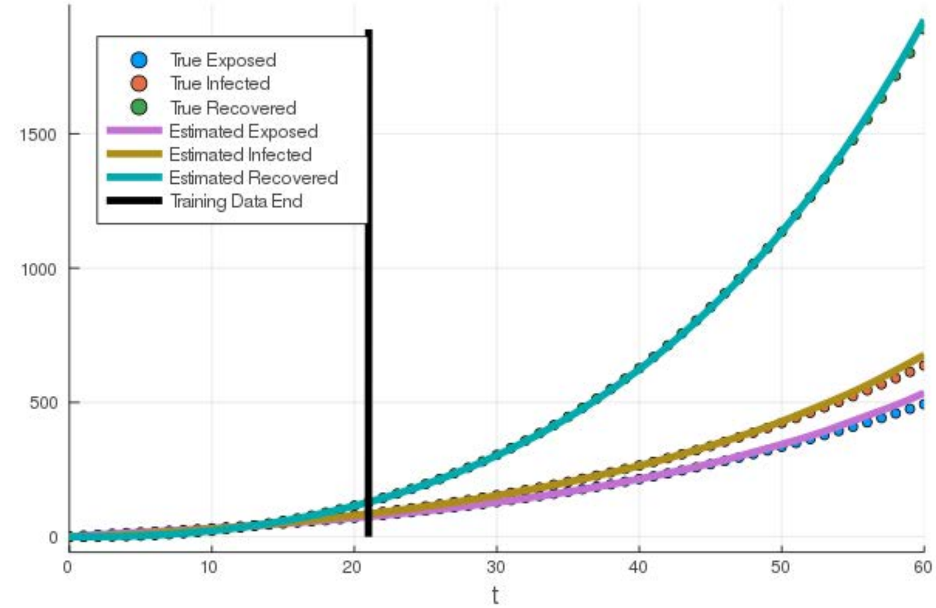
Integrated software



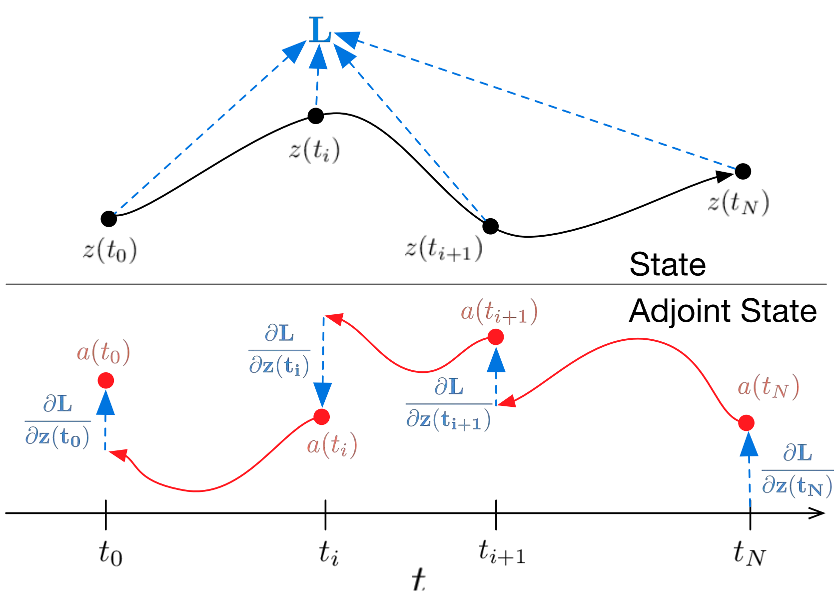
Lu, L., Meng, X., Mao, Z., & Karniadakis, G. E. (2019). DeepXDE: A deep learning library for solving differential equations. arXiv preprint arXiv:1907.04502.

Differentiable programming for scientific computing

$$
\begin{aligned}
S' &= -\frac{\beta_0 SF}{N} - \boxed{\text{Replace Unknown Portion}} - \mu S,\\
E' &= \frac{\beta_0 SF}{N} + \boxed{\text{Replace Unknown Portion}} - (\sigma + \mu)E,\\
I' &= \sigma E - (\gamma + \mu)I,\\
R' &= \gamma I - \mu R,\\
N' &= -\mu N,\\
D' &= d\,\gamma I - \lambda D, \quad \text{and}\\
C' &= \sigma E,
\end{aligned}
$$



Lin et. al. (2020). A conceptual model for the coronavirus disease 2019 (COVID-19) outbreak in Wuhan, China with individual reaction and governmental action. International journal of infectious diseases.



Algorithm 1 Reverse-mode derivative of an ODE initial value problem

Input: dynamics parameters θ, start time t_0, stop time t_1, final state $\mathbf{z}(t_1)$, loss gradient $\partial L / \partial \mathbf{z}(t_1)$

$s_0 = [\mathbf{z}(t_1), \frac{\partial L}{\partial \mathbf{z}(t_1)}, \mathbf{0}_{|\theta|}]$ ▷ Define initial augmented state

def aug_dynamics($[\mathbf{z}(t), \mathbf{a}(t), \cdot], t, \theta$): ▷ Define dynamics on augmented state

  return $[f(\mathbf{z}(t), t, \theta), -\mathbf{a}(t)^\mathsf{T} \frac{\partial f}{\partial \mathbf{z}}, -\mathbf{a}(t)^\mathsf{T} \frac{\partial f}{\partial \theta}]$ ▷ Compute vector-Jacobian products

$[\mathbf{z}(t_0), \frac{\partial L}{\partial \mathbf{z}(t_0)}, \frac{\partial L}{\partial \theta}] = \text{ODESolve}(s_0, \text{aug\_dynamics}, t_1, t_0, \theta)$ ▷ Solve reverse-time ODE

return $\frac{\partial L}{\partial \mathbf{z}(t_0)}, \frac{\partial L}{\partial \theta}$ ▷ Return gradients

Chen, T. Q., Rubanova, Y., Bettencourt, J., & Duvenaud, D. K. (2018). Neural ordinary differential equations. In Advances in neural information processing systems (pp. 6571-6583).

Rackauckas, C., Ma, Y., Martensen, J., Warner, C., Zubov, K., Supekar, R., ... & Ramadhan, A. (2020). Universal Differential Equations for Scientific Machine Learning. arXiv preprint arXiv:2001.04385.