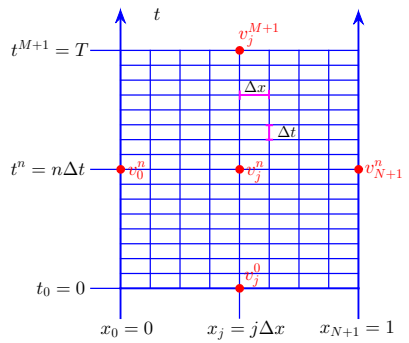
AI in the Sciences and Engineering HS 2026: Lecture 3

Siddhartha Mishra

Computational and Applied Mathematics Laboratory (CamLab)
Seminar for Applied Mathematics (SAM), D-MATH (and),
ETH AI Center (and) Swiss National AI Institute (SNAI) ,
ETH Zürich, Switzerland.

Key Aim of this Course: *Learn Physics modeled by PDEs from data using Neural Networks*

Finite Difference Schemes (FDS) for the Heat Equation



- Heat equation is $u_t = u_{xx}$, Initial conditions: $u(x, 0) = \bar{u}(x)$, Boundary conditions: $u(0, t) = u(1, t) = 0$.
- $u_t(x_j, t^n) \approx \frac{v_j^{n+1} - v_j^n}{\Delta t}$, $u_{xx} \approx \frac{v_{j+1}^n - 2v_j^n + v_{j-1}^n}{\Delta x^2}$
- Finite Difference Scheme:

$$v_j^{n+1} = (1 - 2\lambda)v_j^n + \lambda(v_{j-1}^n + v_{j+1}^n), \quad \lambda = \frac{\Delta t}{\Delta x^2}$$

Can Neural Networks be used to solve PDEs without any Data ?

Can Neural Networks be used to solve PDEs without any Data ?

- Physics-Informed Neural Networks (PINNs)

Physics Informed Neural Networks

- Variants of PINNs stem from Dissanayake, Phan-Thien, 1994.
- Also in Lagaris et al, mid 1990s.
- Reintroduced by Raissi, Perdikaris, Karniadakis, 2017.
- 10000s of papers on PINNs already.

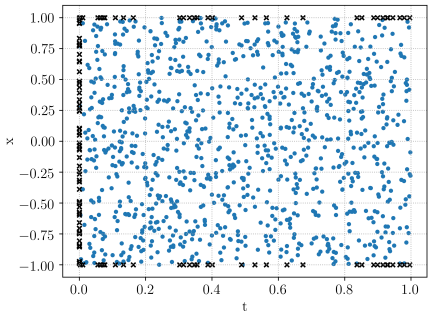
Heat Eqn: $u_t = u_{xx}$ with 0-BC and $u(x,0) = \bar{u}(x)$ IC

- Deep Neural networks : $(x,t) \mapsto u_\theta(x,t)$, $\theta \in \Theta$.
- Temporal boundary residual: $\mathcal{R}_{tb,\theta} = u_\theta(\cdot,0) - \bar{u}$
- Spatial boundary residual: $\mathcal{R}_{sb,\theta} = u_\theta|_{\partial D}$.
- Interior PDE Residual: $\mathcal{R}_{int,\theta} = \partial_t u_\theta - \partial_{xx} u_\theta$
- Evaluate PDE Residual by Automatic Differentiation
- Error (Mismatch):

$$\widehat{J}(\theta) := \int_0^T \int_D |\mathcal{R}_{int,\theta}(x,t)|^2 dxdt + \int_D |\mathcal{R}_{tb,\theta}(x)|^2 dx + \int_0^T \int_{\partial D} |\mathcal{R}_{sb,\theta}(x,t)|^2 ds(x)dt$$

PINNs for Heat Equation (Contd..)

- Replace Integrals by Quadrature
- Collocation Point Cloud: $\mathcal{S} = \mathcal{S}_{int} \cup \mathcal{S}_{tb} \cup \mathcal{S}_{sb}$ Randomly chosen.



- Replacing integrals with Quadrature Rules $\Rightarrow$
- PINNs are trained by minimizing the Loss Function:

$$J(\theta) = \frac{1}{N_{tb}} \sum_{n=1}^{N_{tb}} |\mathcal{R}_{tb,\theta}(x_n)|^2 + \frac{1}{N_{sb}} \sum_{n=1}^{N_{sb}} |\mathcal{R}_{sb,\theta}(x_n, t_n)|^2 + \frac{1}{N_{int}} \sum_{n=1}^{N_{int}} |\mathcal{R}_{int,\theta}|^2.$$

Why PINNs are great ?

- Very easy to Code !!
- A few lines in PyTorch or JAX

```
def compute_res(self, network, x_f_train):
    x_f_train.requires_grad = True
    u = network(x_f_train).reshape(-1, )
    grad_u = torch.autograd.grad(u, x_f_train, grad_outputs=torch.ones(x_f_train.shape[0], ).to(self.device), create_graph=True)[0]
    grad_u_t = grad_u[:, 0]
    grad_u_x = grad_u[:, 1]
    grad_u_xx = torch.autograd.grad(grad_u_x, x_f_train, grad_outputs=torch.ones(x_f_train.shape[0], ).to(self.device), create_graph=True)[0][:, 1

    residual = grad_u_t - self.v * grad_u_xx
    return residual
```

Results for Multi-d Heat Eqn with Random points

- PINN with Depth 4, Width 20, Interior training points 2^{16}, Boundary points 2^{15}

Dimension	Training Error	Total Error
1	2.8×10^{-5}	0.0035%
5	0.0002	0.016%
10	0.0003	0.03%
20	0.006	0.79%
50	0.006	1.5%
100	0.004	2.6%

- No Curse of dimensionality !!

PDEs in Finance (Tanios, 2021)

- Black-Scholes type PDE with Uncorrelated Noise:

Dimension	Training Error	Generalization error
20	0.0016	1.0%
50	0.0031	1.5%
100	0.0031	1.8%

- Heston option-pricing PDE

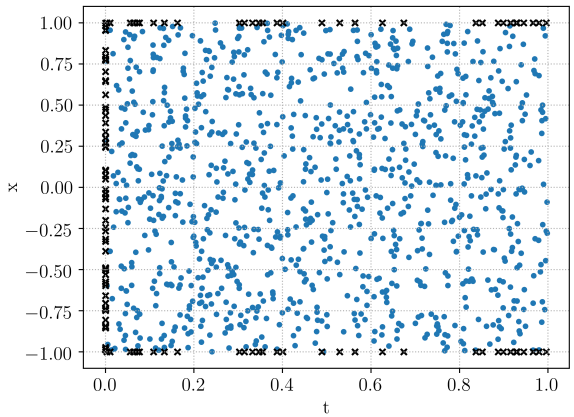
Dimension	Training Error	Total Error
20	0.0064	1.0%
50	0.0037	1.3%
100	0.0032	1.4%

Viscous scalar conservation laws

- Nonlinear hyperbolic-parabolic PDE.

$$u_t + \text{div } f(u) = \nu \Delta u \quad (x,t) \in D \times [0,T],$$
$$u(x,0) = \bar{u}(x), \quad x \in D, \quad u|_{\partial D} = 0.$$

- Training set: $\mathcal{S} = \mathcal{S}_{int} \cup \mathcal{S}_{tb} \cup \mathcal{S}_{sb}$



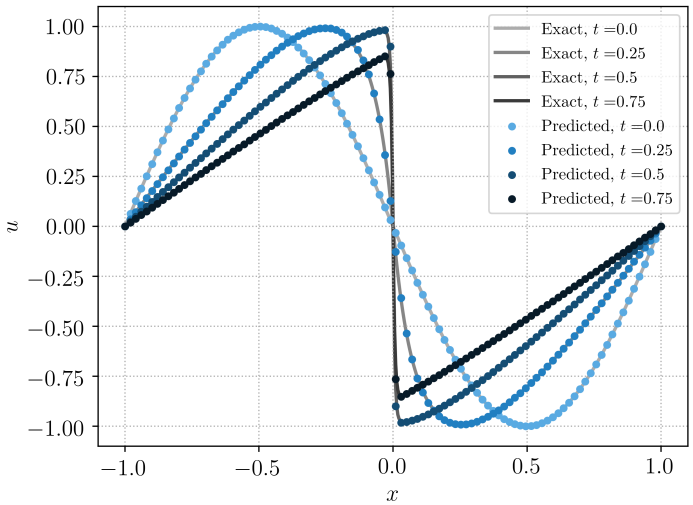
PINNs

- Deep Neural networks : $(x,t) \mapsto u_\theta(x,t)$, $\theta \in \Theta$.
- Residuals
  - Interior Residual: $\mathcal{R}_{int,\theta} = \partial_t u_\theta + \text{div } f(u_\theta) - \nu \Delta u_\theta$.
  - Temporal boundary residual: $\mathcal{R}_{tb,\theta} = u_\theta(\cdot, 0) - \bar{u}$
  - Spatial boundary residual: $\mathcal{R}_{sb,\theta} = u_\theta|_{\partial D}$.
- Loss function:

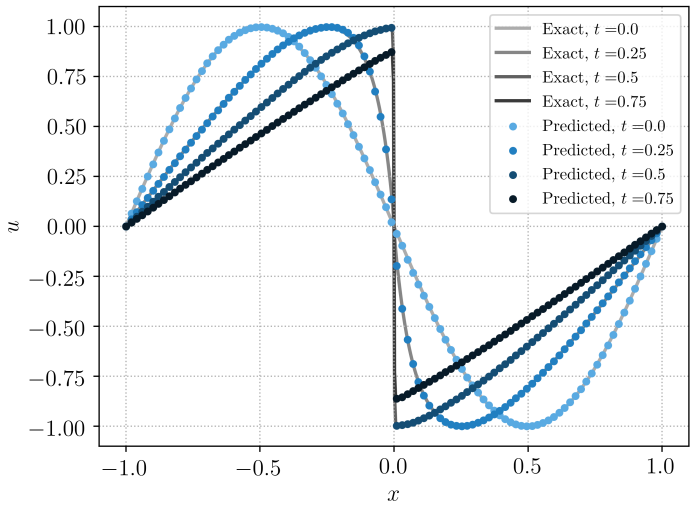
$$J = \underbrace{\sum_{n=1}^{N_{tb}} w_n^{tb} |\mathcal{R}_{tb,\theta}(x_n)|^2}_{(\mathcal{E}_T^{tb})^2} + \underbrace{\sum_{n=1}^{N_{sb}} w_n^{sb} |\mathcal{R}_{sb,\theta}|^2}_{(\mathcal{E}_T^{sb})^2} + \underbrace{\sum_{n=1}^{N_{int}} w_n^{int} |\mathcal{R}_{int,\theta}|^2}_{(\mathcal{E}_T^{int})^2}.$$

Results for 1-D Burgers'

- Sobol points, $N_{int} = 8192$, $N_{tb} = N_{sb} = 256$, Depth 8, Width 20.



$\nu = 10^{-2}$



$\nu = 10^{-3}$

Viscosity	Training Error	Total error
10^{-2}	0.0005	1.0%
10^{-3}	0.0008	1.2%

Incompressible Navier-Stokes equations

- Nonlinear PDE of form

$$\begin{aligned} \mathrm{u}_t + (\mathrm{u}\cdot\nabla)\mathrm{u} + \nabla p = \nu\Delta u + f, \quad (x,t) \in D\times[0,T], \\ \text{div } \mathrm{u} = 0, \quad (x,t)\in D\times[0,T], \\ \mathrm{u}(x,0) = \bar{\mathrm{u}}(x), \quad x\in D, \quad \mathrm{u}|_{\partial D\times(0,T)} = 0. \end{aligned}$$

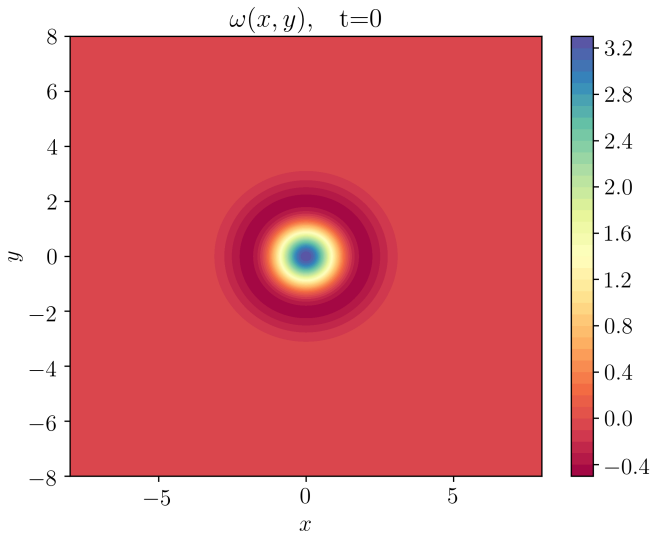
- Training set: $\mathcal{S} = \mathcal{S}_{int} \cup \mathcal{S}_{tb} \cup \mathcal{S}_{sb}$

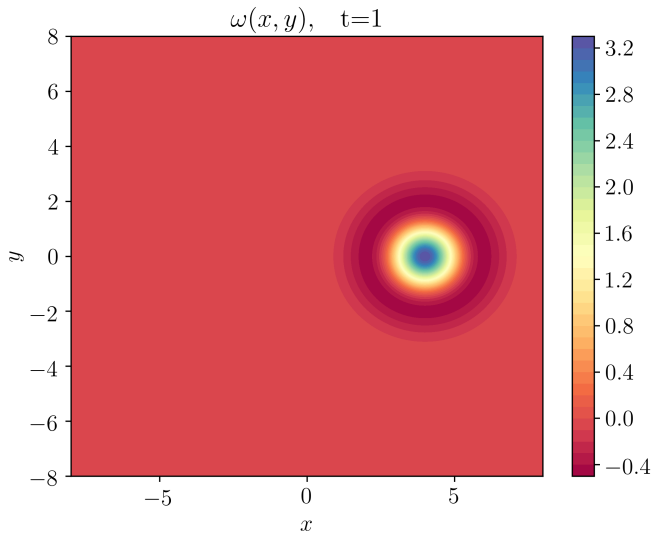
PINNs

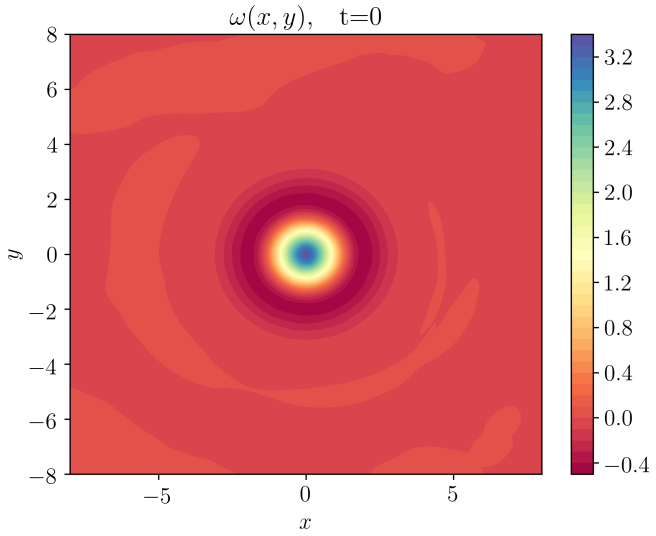
- Deep Neural networks : $(x,t) \mapsto \mathrm{u}_\theta(x,t), p_\theta(x,t),\ \theta \in \Theta$.
- Residuals
  - Velocity Residual: $\mathcal{R}_{\mathrm{u},\theta} = \partial_t \mathrm{u}_\theta + (\mathrm{u}_\theta \cdot \nabla)\mathrm{u}_\theta + \nabla p_\theta - f$
  - Divergence Residual: $\mathcal{R}_{div,\theta} := \mathrm{div}\ \mathrm{u}_\theta$
  - Temporal and Spatial boundary residuals as before.
- Loss function:

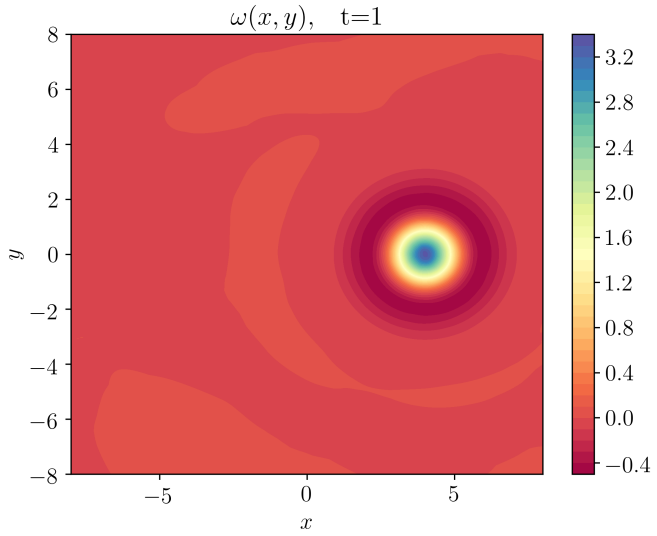
$$J(\theta) = \underbrace{\sum_{n=1}^{N_{tb}} w_n^{tb} |\mathcal{R}_{tb,\theta}(x_n)|^2}_{(\mathcal{E}_T^{tb})^2} + \underbrace{\sum_{n=1}^{N_{sb}} w_n^{sb} |\mathcal{R}_{sb,\theta}|^2}_{(\mathcal{E}_T^{sb})^2}$$
$$+ \underbrace{\sum_{n=1}^{N_{int}} w_n^{int} |\mathcal{R}_{\mathrm{u},\theta}|^2}_{(\mathcal{E}_T^{u})^2} + \underbrace{\sum_{n=1}^{N_{int}} w_n^{int} |\mathcal{R}_{div,\theta}|^2}_{(\mathcal{E}_T^{d})^2}$$

Results for 2-D Taylor vortex

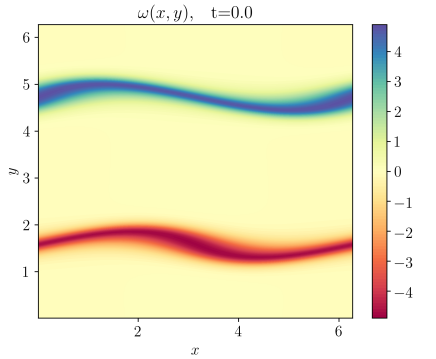


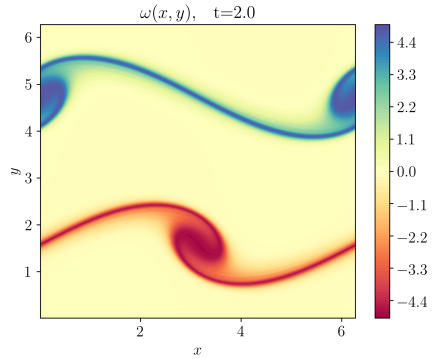


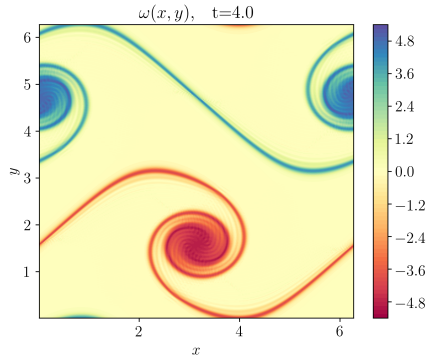


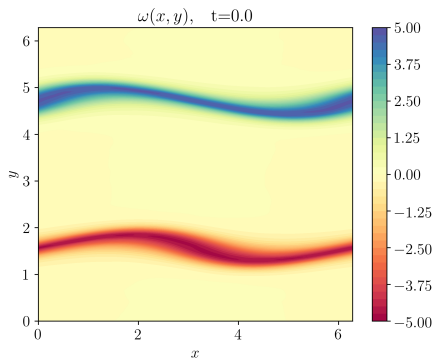


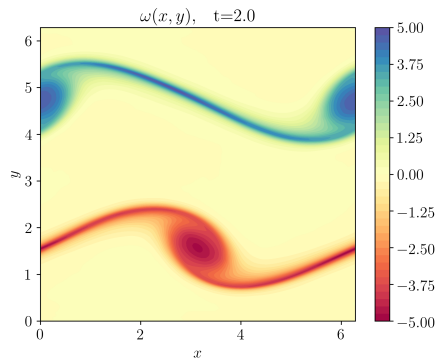
Results for 2-D Double Shear Layer

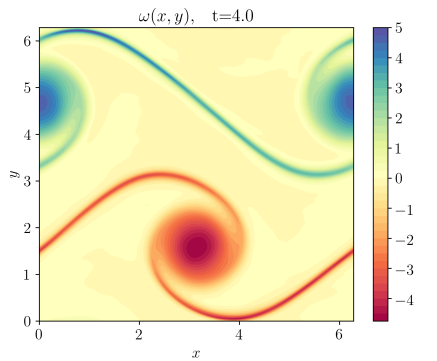






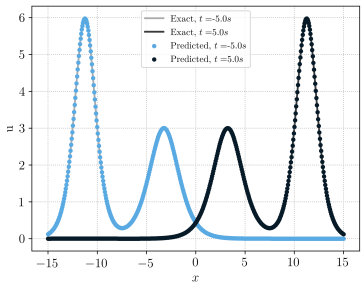




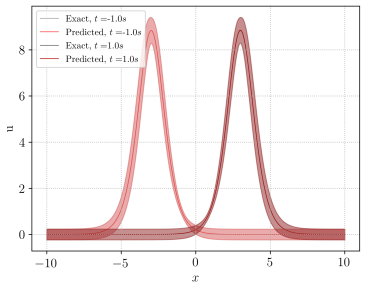


PINNs for nonlinear Dispersive PDEs

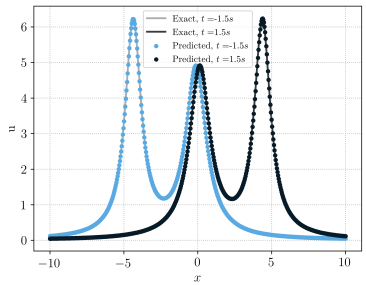
- Numex in Bai, Koley, SM, Molinaro, 2021.



KdV



6-d param KdV



Benjamin-Ono

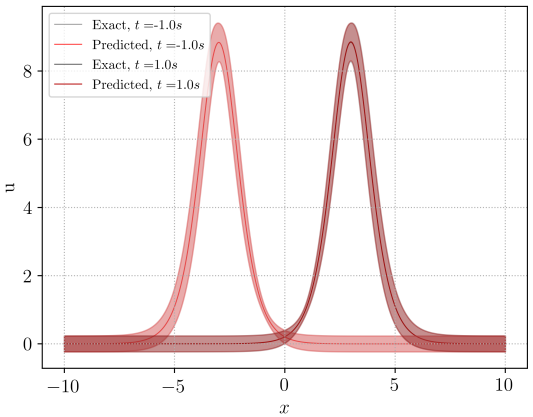
PDE	Network Size	Error
KdV	32×4	0.1%
6-D param KdV	24×8	0.5%
Benjamin-Ono	20×4	0.7%

Parametric PDEs

- Consider the KdV Eqn:

$$u_t + uu_x + u_{xxx} = 0,$$
$$u(0, x, y) = u_0(x, y).$$

- $y \in Y \subset \mathbb{R}^6$ Parametrizes Initial conditions.
- PINN: $(t, x, y) \mapsto u_\theta(t, x, y)$
- Visualizations of Mean + Variance.

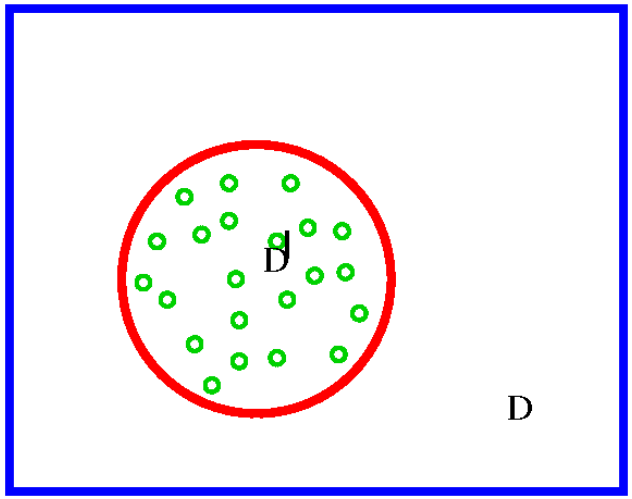


- Error of 0.5%

Role of Data in PINNs

- PINNs for PDE Forward Problem require NO labelled Data
- If small Data is available $\Rightarrow$ better empirical performance.
- Relevant for parametrized PDEs
- Situation is different for Inverse Problems for PDEs.
- Data is a part of such problems !!

Data Assimilation or Unique continuation problem



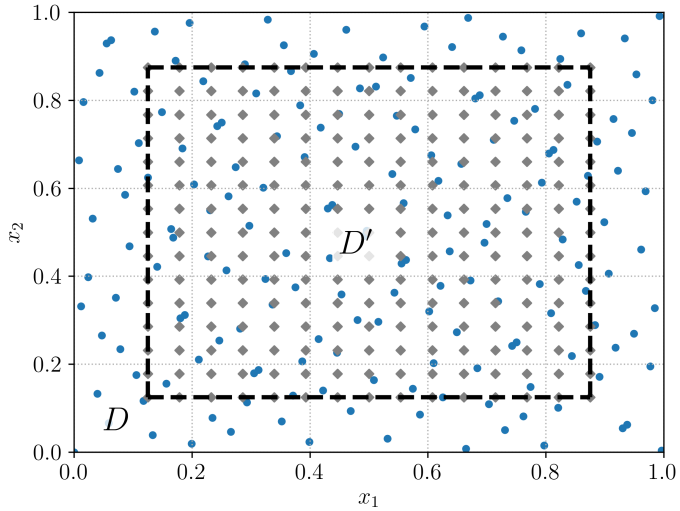
- Abstract PDE is $\mathcal{D}(\mathsf{u}) = \mathsf{f}$ in $\mathbb{D}$
- Measurements: $\mathcal{L}(\mathsf{u}) = \mathsf{g}$ in $\mathbb{D}'$
- Goal: Reconstruct u given f, g.
- $\mathsf{u}_\theta : \mathbb{D} \mapsto \mathbb{R}^m$ is a PINN
- PDE Residual: $\mathcal{R} := \mathcal{R}_\theta(y) = \mathcal{D}\left(\mathsf{u}_\theta(y)\right) - \mathsf{f}(y), \ y \in \mathbb{D}$
- Data Residual: $\mathcal{R}_d := \mathcal{R}_{d,\theta}(z) = \mathcal{L}\left(\mathsf{u}_\theta(z)\right) - \mathsf{g}(z), \ z \in \mathbb{D}'.$
- PINNs are minimizers of $\int_{\mathbb{D}} |\mathcal{R}_\theta(y)|^{p_y} \, dy + \int_{\mathbb{D}'} |\mathcal{R}_{d,\theta}(z)|^{p_z} \, dz$

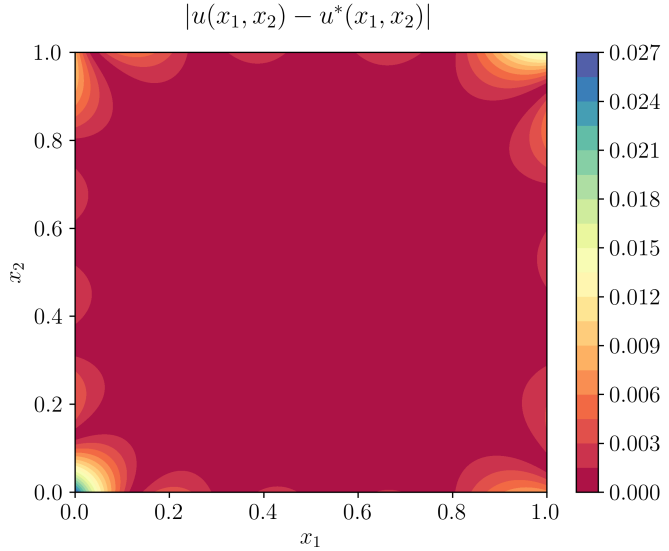
Poisson Equation

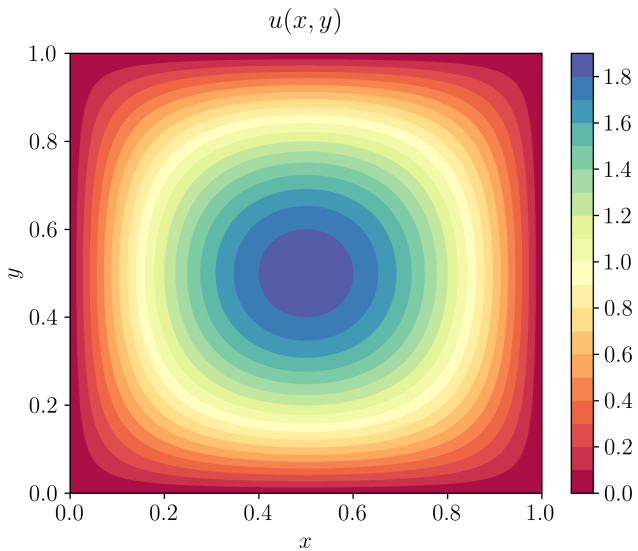
- Model Elliptic PDE with Data assimilation problem:

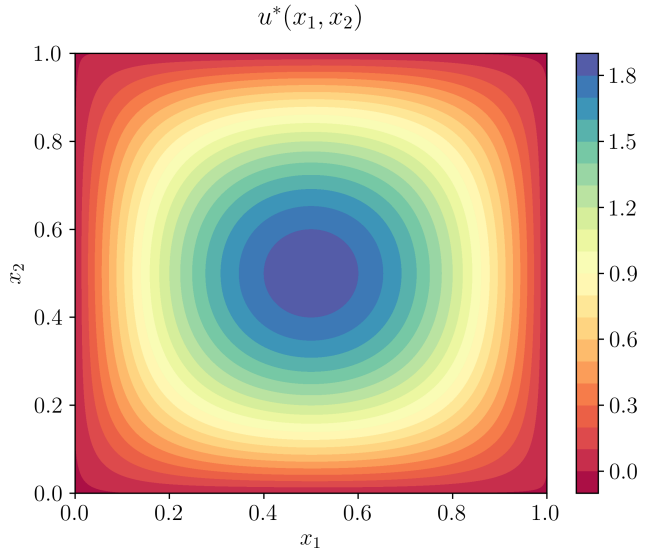
$$-\Delta u = f, \quad \text{in } \mathbb{D} \subset \mathbb{R}^d,$$
$$u|_{\mathbb{D}'} = g \quad \text{in } \mathbb{D}' \subset \mathbb{D}.$$

Results in 2-D









Results in 2-D

- No Noise: Depth 4, Width 24

N	Training Error	Generalization error
20^2	0.0008	1.1%
40^2	0.0006	1.0%
80^2	0.0005	0.9%
160^2	0.0004	0.8%

- With 1% noise:

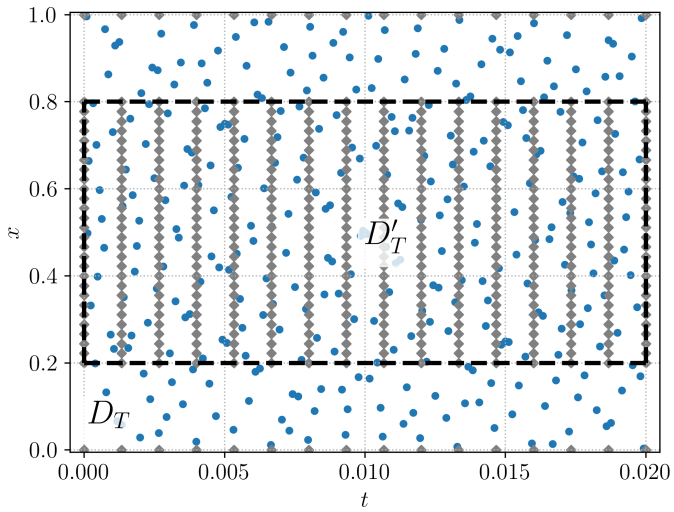
N	Training Error	Generalization error
20^2	0.012	2.3%
40^2	0.013	1.9%
80^2	0.013	1.7%
160^2	0.013	1.3%

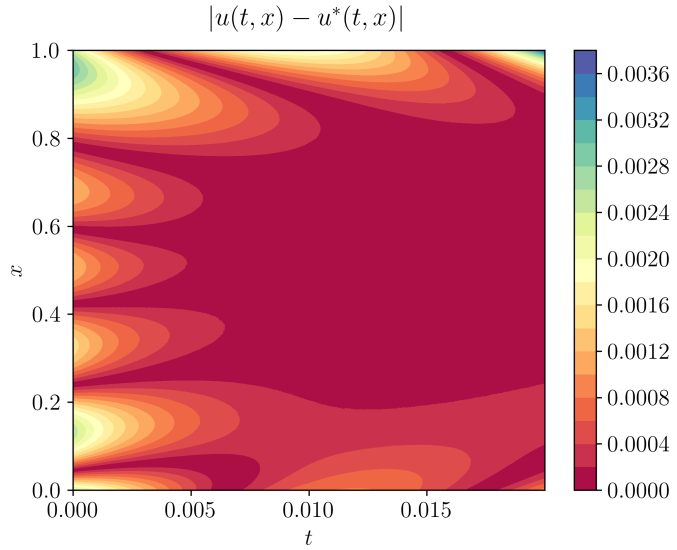
Heat Equation

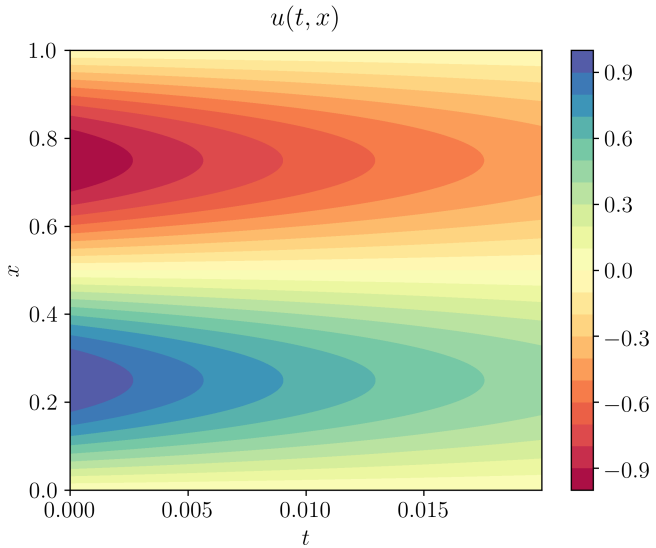
- Model Parabolic PDE with Data assimilation problem:

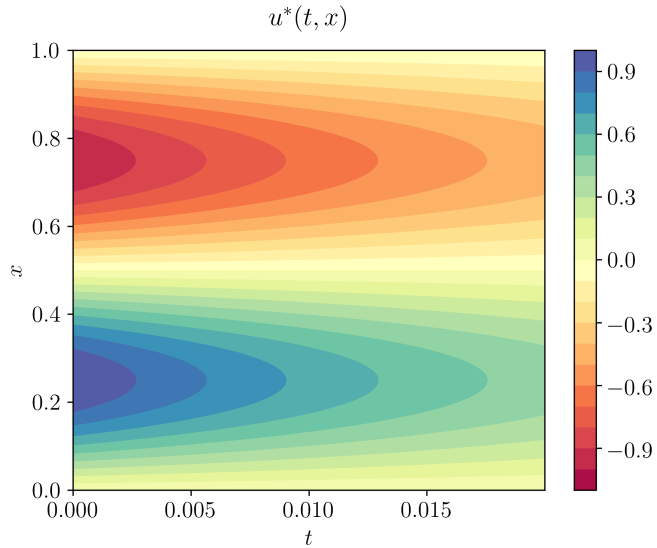
$$
\begin{aligned}
u_t - \Delta u &= f, \quad \text{in } D_T = D \times (0, T), \\
u|_{\partial D} &= 0, \\
u|_{D'_T} &= g \quad \text{in } D'_T \subset D_T
\end{aligned}
$$

Results









Results with Random points

- 1-D : Depth 4, Width 24

N	Training Error	Generalization error
16×50	0.001	0.55%
16×100	0.0009	0.52%
16×200	0.0007	0.47%

- Multi-D with $N = 2^{14}$ training points

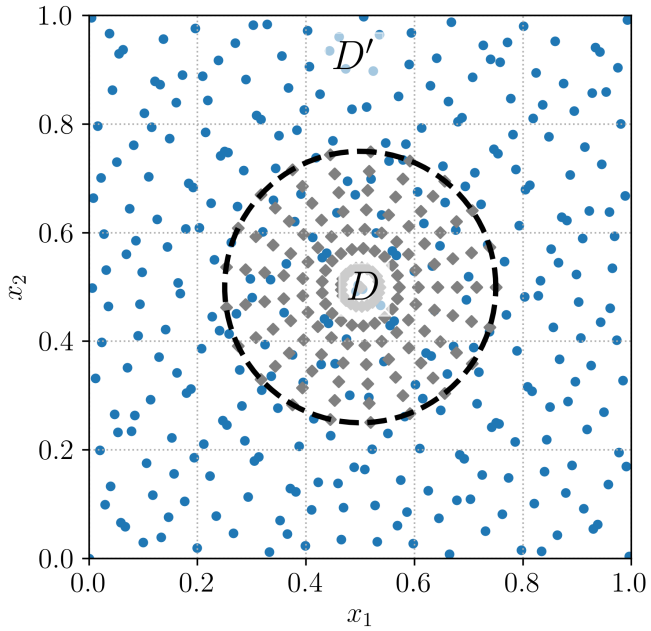
Dimension	Training Error	Generalization error
5	0.0003	0.044%
10	0.0003	0.07%
20	0.0003	0.62%
50	0.0003	1.68%
100	0.0003	2.0%

Stokes Equation

- Model Indefinite PDE with Data assimilation problem:

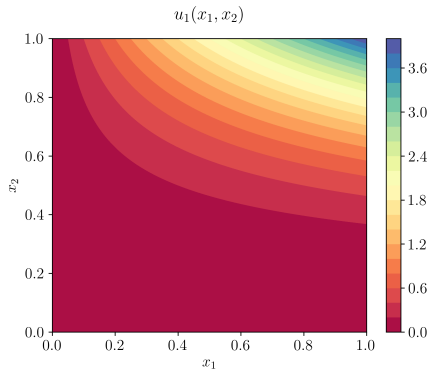
$$\Delta \mathrm{u} + \nabla p = f, \quad \text{in } \mathbb{D} \subset \mathbb{R}^d,$$
$$u|_{\mathbb{D}'} = g \quad \text{in } \mathbb{D}' \subset \mathbb{D}.$$

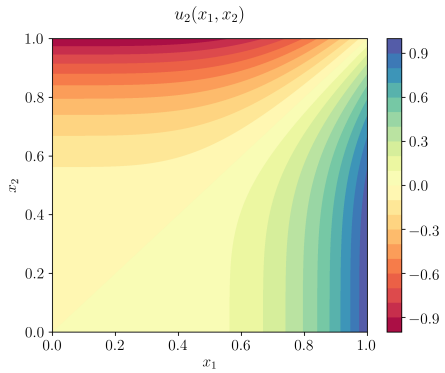
2-D Results

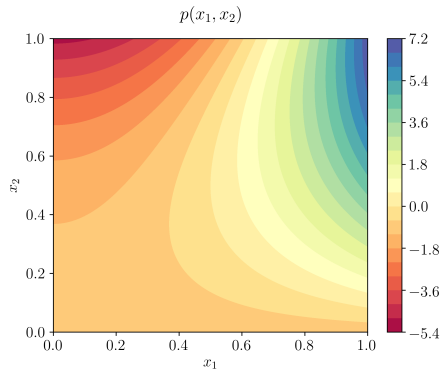


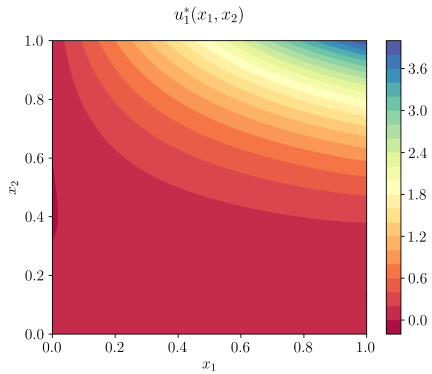
N	Training Error	$\mathcal{E}_G(\mathrm{u})$	$\mathcal{E}_G(p)$
20^2	0.0007	2.3%	5.6%
40^2	0.0004	1.7%	4.0%
80^2	0.0004	1.5%	3.5%

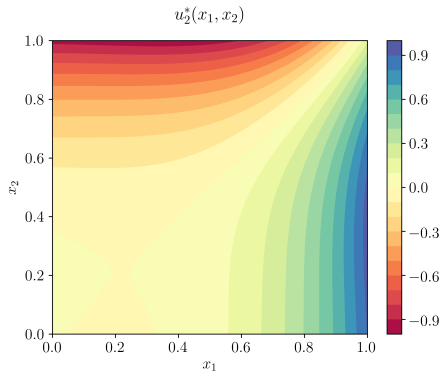
2-D Results

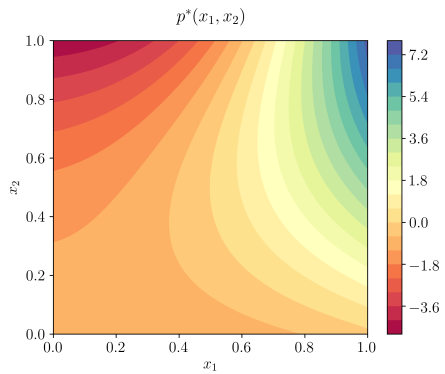

$u_1(x_1, x_2)$
x_1
x_2


$u_2(x_1, x_2)$
x_1
x_2


$p(x_1, x_2)$
x_1
x_2


$u_1^*(x_1, x_2)$
x_1
x_2


$u_2^*(x_1, x_2)$
x_1
x_2


$p^*(x_1, x_2)$
x_1
x_2

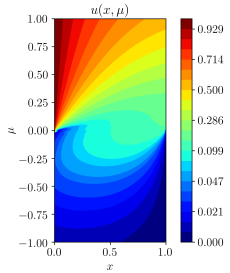
Radiative Transfer Equations

- $2d+1$-dim Integro-Differential PDE for Intensity

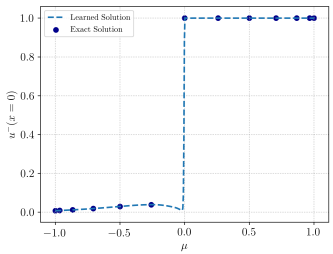
$$\frac{1}{c}u_t + \omega \cdot \nabla u + \left(k(x,\nu) + \sigma(x,\nu)\right) u - \frac{\sigma(x,\nu)}{s_d} \int_{R_+} \int_{S} \Phi(\omega,\omega',\nu,\nu') u d\omega' d\nu' = f(x,t,n,\nu).$$

- High-dimensional, non-local, hyperbolic, multiphysics
- PINNs applied and bound derived in SM, Molinaro 2021.

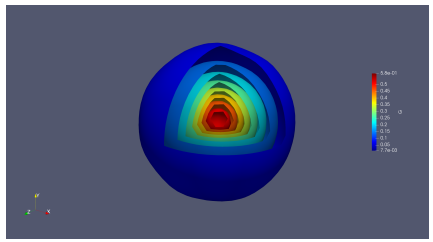
Numerical Results for Forward Problem



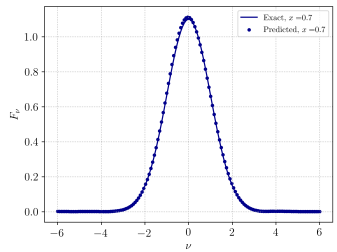
2-D, Intensity



2-D, Boundary



6-D, Inc. Radiation



6-D, Radial flux

Dimension	Network Size	Error	Training Time
2	24×8	0.3%	57 min
6	20×8	2.1%	66 min

PINNs for Inverse Problem: Parameter Identification

- For Radiative Transfer Eqns:

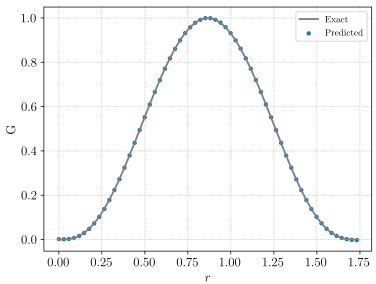
$$\frac{1}{c}u_t + \omega \cdot \nabla u + \left(k(x,\nu) + \sigma(x,\nu)\right) u - \frac{\sigma(x,\nu)}{s_d} \int_{R_+} \int_{S} \Phi(\omega,\omega',\nu,\nu') u d\omega' d\nu' = f(x,t,n,\nu).$$

- Find Absorption Coefficient k, given Incident Radiation G
- PINNs with Tikhonov Regularization :SM, Molinaro, 2020

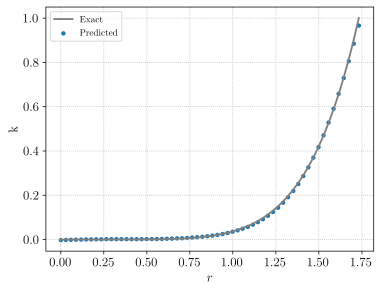
Results

- For a 5-dimensional Radiative Transfer:

M	k-Error	u-Error	Training Time
20×8	2.8%	0.7%	104 min



Incident Radiation



Absorption Coefficient.