Apprentissage statistique pour la physique

Physics Informed Neural Networks

Jean-Christophe Loiseau

Arts & Métiers

jean-christophe.loiseau@ensam.eu

loiseaujc.github.io

github.com/loiseaujc

Qui suis-je ?

- Maître de Conférences à l'Ecole Nationale Supérieure d'Arts et Métiers, Paris.
- Recherche orientée autour de trois axes principaux:
  - Transition à la turbulence.
  - Modélisation réduite et identification de systèmes.
  - Contrôle et Estimation.

Apprentissage statistique pour et par la Physique

Déroulement du cours

Cours 1 – Réseaux de neurones informés par la Physique

Lundi 6 janvier, 10h15-12h15

Résolution d'une EDP à l'aide d'un PINN
PINNS et problèmes inverses
Limites des PINNs

???

Déroulement du cours

TP/TD – Physics-Informed Neural Networks

Lundi 6 janvier, 14h00-17h00

Résolution de l'équation de Burgers
Trucs et Astuces

???

Déroulement du cours

Cours 2 – Identification de systèmes non-linéaires

Mardi 7 janvier, 09h30-10h30

Le principe de parcimonie
Sparse Identification of Nonlinear Dynamics
Illustrations

???

Déroulement du cours

TP/TD – SINDy

Mardi 7 janvier, 10h30-12h30

Optimisation convexe
`pySINDy`

???

Physics Informed Neural Networks

PINNs

- Proposed by Raissi, Perdikaris et Karniadakis in 2017 for solving PDE.

Idea Leverage the universal approximation capabilities of neural nets to represent the solution to a PDE.

Maziar Raissi

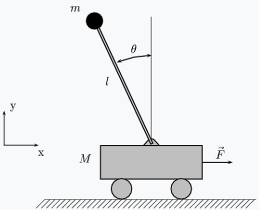
Kalman smoothing

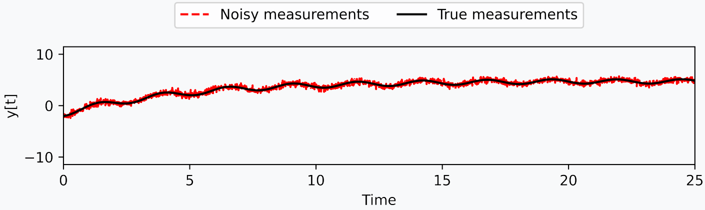
A (rapid) detour through convex optimization

$$\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t + \mathbf{w}_t$$
$$\mathbf{y}_t = \mathbf{C}\mathbf{x}_t + \mathbf{v}_t$$

State vector	$\mathbf{x} \in \mathbb{R}^n$
Transition matrix	$\mathbf{A} \in \mathbb{R}^{n \times n}, \rho(\mathbf{A}) < 1$
Measurement operator	$\mathbf{C} \in \mathbb{R}^{q \times n}$
Process noise	$\mathbf{w}_t \sim \mathcal{N}(0, \mathbf{W})$
Sensor noise	$\mathbf{v}_t \sim \mathcal{N}(0, \mathbf{V})$

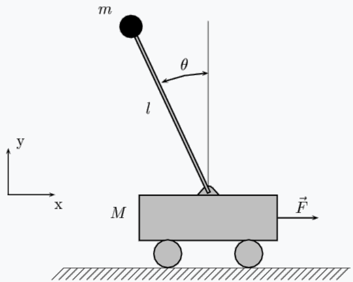
Given **A**, **C** and the sequence $\{\mathbf{y}_t\}_{t=0,\cdots,T}$, can we uniquely determine the most likely state sequence $\{\mathbf{x}_t\}_{t=0,\cdots,T}$?

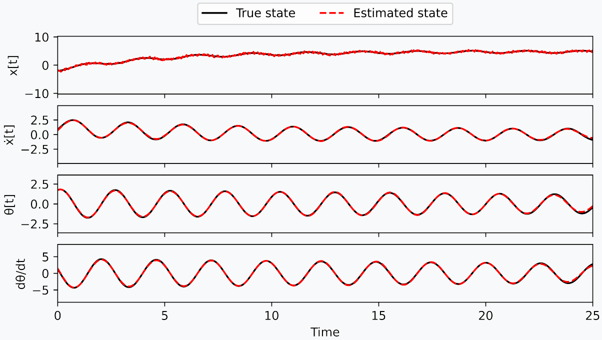




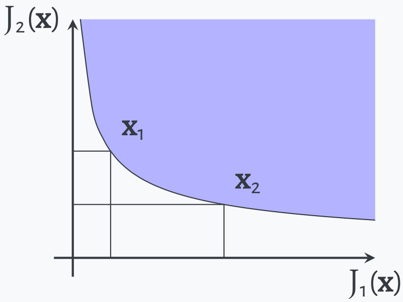
$$\text{minimize} \quad \frac{1}{2}\sum_{t=0}^{T} \overbrace{\|\mathbf{y}_t - \mathbf{C}\mathbf{x}_t\|^2_{\mathbf{V}^{-1}}}^{\text{Data fidelity}} + \frac{1}{2}\sum_{t=0}^{T-1} \underbrace{\|\mathbf{x}_{t+1} - \mathbf{A}\mathbf{x}_t\|^2_{\mathbf{W}^{-1}}}_{\text{Model fidelity}}$$

$$
\begin{bmatrix}
\mathbf{C}^*\mathbf{C}+\mathbf{A}^*\mathbf{A} & -\mathbf{A}^* & & & \\
-\mathbf{A} & \mathbf{C}^*\mathbf{C}+\mathbf{I}+\mathbf{A}^*\mathbf{A} & -\mathbf{A}^* & & \\
& \ddots & \ddots & \ddots & \\
& & -\mathbf{A} & \mathbf{C}^*\mathbf{C}+\mathbf{I}+\mathbf{A}^*\mathbf{A} & -\mathbf{A}^* \\
& & & -\mathbf{A} & \mathbf{C}^*\mathbf{C}+\mathbf{I}
\end{bmatrix}
\begin{bmatrix}
\hat{\mathbf{x}}_0 \\ \hat{\mathbf{x}}_1 \\ \vdots \\ \hat{\mathbf{x}}_{T-1} \\ \hat{\mathbf{x}}_T
\end{bmatrix}
=
\begin{bmatrix}
\mathbf{C}^*\mathbf{y}_0 \\ \mathbf{C}^*\mathbf{y}_1 \\ \cdots \\ \mathbf{C}^*\mathbf{y}_{T-1} \\ \mathbf{C}^*\mathbf{y}_T
\end{bmatrix}
$$


m
θ
l
y
x
M
F


True state
Estimated state
x[t]
ẋ[t]
θ[t]
dθ/dt
Time

- Kalman smoothing fundamentally is a *Multi-Objective* problem.
- The two objectives are competing against one another.
- Optimization problem is made tractable via *scalarization*.
- Existence of a (convex) Pareto front.

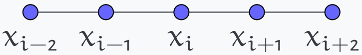


Physics-Informed Neural Nets

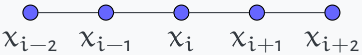
PDE and Scientific Computing

$$\begin{cases} \dfrac{\partial u}{\partial t} = \kappa \dfrac{\partial^2 u}{\partial x^2} \quad \text{for} \quad x \in]0, L[, \quad t \in [0, T] \\ u(0, t) = u(L, t) = 0 \\ u(x, 0) = \sin\left(\dfrac{\pi x}{L}\right). \end{cases}$$

Analytical solution $u(x, t) = \sin\left(\frac{\pi x}{L}\right) \exp\left(-\frac{\kappa \pi^2}{L^2} t\right)$



- Space is typically discretized using a *mesh*.
  - Structured or unstructured mesh.
- Differential operators are approximated on the discrete set of nodes.
  - Finite differences, finite volumes, finite elements, . . .
- Temporal integration based on various numerical schemes.
  - Euler, Runge-Kutta, Adam-Bashfort, . . .



First-order derivative

$$\frac{\partial u}{\partial x} = \frac{u_{i+1} - u_{i-1}}{2\Delta x} + \mathcal{O}(\Delta x^2)$$

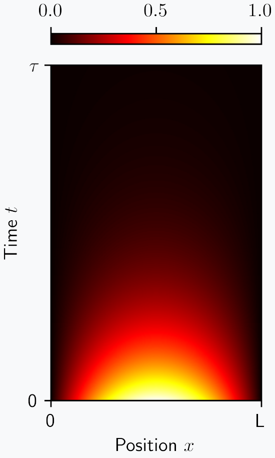
Second-order derivative

$$\frac{\partial^2 u}{\partial x^2} = \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} + \mathcal{O}(\Delta x^2)$$

Semi-discretized system

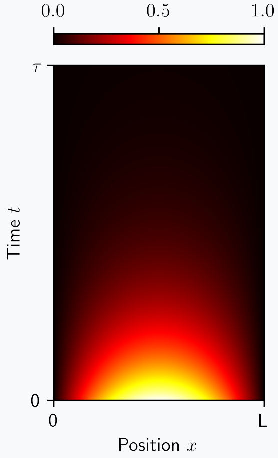
$$\frac{d}{dt}\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix} = \frac{\kappa}{\Delta x^2}\begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{bmatrix}\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix}$$

Explicit Euler	$\mathbf{u}_{t+1} = (\mathbf{I} + \Delta t \mathbf{L})\, \mathbf{u}_t$
Implicit Euler	$\mathbf{u}_{t+1} = (\mathbf{I} - \Delta t \mathbf{L})^{-1}\, \mathbf{u}_t$
Crank-Nicholson	$\mathbf{u}_{t+1} = \left(\mathbf{I} - \frac{\Delta t}{2}\mathbf{L}\right)^{-1} \left(\mathbf{I} + \frac{\Delta t}{2}\mathbf{L}\right) \mathbf{u}_t$



Pros ✔

- Proof of consistency and convergence.
- Control of the approximation error, both in time $\mathcal{O}(\Delta t^n)$ and space $\mathcal{O}(\Delta x^m)$.
- Specialized solvers amenable to high-performance computing (HPC).
- Certification of existing CFD codes for sensitive applications.



Cons ✗

- Meshing complicated geometries is more of an art than science.
- Relatively high computational cost.
- Solver specialized for a single (type of) PDE.
- Constant need for code modernization to keep up-to-date with new processor architectures.

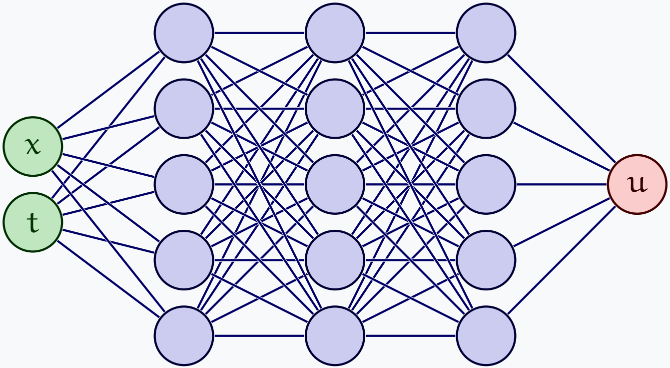
Physics-Informed Neural Nets

Back to deep learning

- Proposed by Raissi, Perdikaris et Karniadakis in 2017 for solving PDE.

Idea Leverage the universal approximation capabilities of neural nets to represent the solution to a PDE.

Maziar Raissi


x
t
u

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

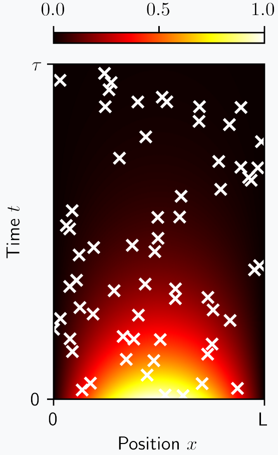
```python
def residual(x, t, kappa):
  # Computation of the time-derivative.
  u_t = tf.gradients(u, t)[0]
  # Computation of the first spatial derivative.
  u_x = tf.gradients(u, x)[0]
  # Computation of the second spatial derivative.
  u_xx = tf.gradients(u_x, x)[0]
  # Definition of the residual.
  r = u_t - kappa*u_xx
  return r
```

$$
\begin{aligned}
\text{Find} \quad & \vartheta \in \mathbb{R}^n \\
\text{subject to} \quad & \mathcal{R}(x,t) \equiv \frac{\partial u_\vartheta}{\partial t} - \kappa \frac{\partial^2 u_\vartheta}{\partial x^2} = 0 \quad \forall (x,t) \in [0,L] \times [0,T] \\
& u_\vartheta(x,0) = u_0(x) \\
& u_\vartheta(0,t) = u_\vartheta(L,t) = 0.
\end{aligned}
$$

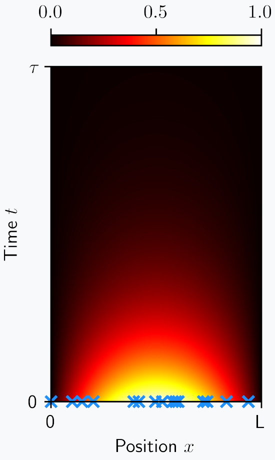
Physics	$\mathcal{L}_{\varphi}(\vartheta) = \frac{1}{T \cdot L} \int_0^T \int_0^L \lvert \mathcal{R}(x, t) \rvert^2 \, dx dt$
Initial condition	$\mathcal{L}_{IC}(\vartheta) = \frac{1}{L} \int_0^L \lvert u_\vartheta(x, 0) - u_0(x) \rvert^2 \, dx$
Boundary conditions	$\mathcal{L}_{BC}(\vartheta) = \frac{1}{T} \int_0^T \lvert u_\vartheta(0, t) \rvert^2 + \lvert u_\vartheta(L, t) \rvert^2 \, dt$

$$\underset{\vartheta \in \mathbb{R}^n}{\text{minimize}} \quad \underbrace{\alpha_1 \mathcal{L}_{IC}(\vartheta) + \alpha_2 \mathcal{L}_{BC}(\vartheta)}_{\text{Data fidelity}} + \underbrace{\alpha_3 \mathcal{L}_{\varphi}(\vartheta)}_{\text{Model fidelity}}$$

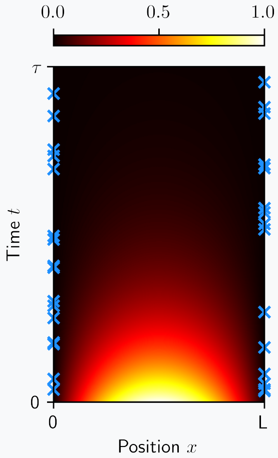
with $\alpha_1 + \alpha_2 + \alpha_3 = 1.$



$$\mathcal{L}_{\varphi}(\vartheta) = \frac{1}{T \cdot L} \int_0^T \int_0^L |\mathcal{R}(x,t)|^2 \, dx dt$$
$$\simeq \frac{1}{N_{\varphi}} \sum_{i=1}^{N_{\varphi}} |\mathcal{R}(x_i, t_i)|^2$$



$$\mathcal{L}_{IC}(\vartheta) = \frac{1}{L}\int_0^L |u_\vartheta(x,0) - u_0(x)|^2\,dx$$
$$\simeq \frac{1}{N_{IC}}\sum_{i=1}^{N_{IC}} |u_\vartheta(x_i,0) - u_0(x_i)|^2$$



$$\mathcal{L}_{BC}(\vartheta) = \frac{1}{T}\int_0^T |u_\vartheta(0,t)|^2 + |u_\vartheta(L,t)|^2 \, dt$$

$$= \frac{1}{N_{BC}} \sum_{i=1}^{N_{BC}} |u_\vartheta(0,t_i)|^2 + |u_\vartheta(L,t_i)|^2$$

Non-dimensionalization of the PDE

$$\hat{t} = \frac{t}{\tau}, \quad \hat{x} = \frac{x}{L}$$

$$\frac{\partial u}{\partial t} - \kappa \frac{\partial^2 u}{\partial x^2} = 0$$

Non-dimensionalization of the PDE

$$\hat{t} = \frac{t}{\tau}, \quad \hat{x} = \frac{x}{L}$$

$$\frac{1}{\tau}\frac{\partial u}{\partial \hat{t}} - \frac{\kappa}{L^2}\frac{\partial^2 u}{\partial \hat{x}^2} = 0$$

Non-dimensionalization of the PDE

$$\tau = \frac{L^2}{\kappa}$$

$$\frac{\kappa}{L^2}\left(\frac{\partial u}{\partial \hat{t}} - \frac{\partial^2 u}{\partial \hat{x}^2}\right) = 0$$

$$\mathcal{R}(x,t) \sim \frac{\kappa}{L^2}$$

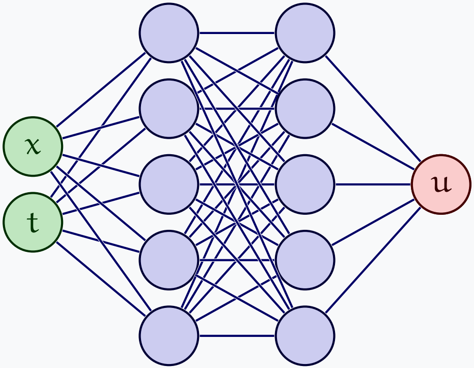
$$\text{data} \sim 1$$

Physics loss and data losses have fundamentally different scales, depending implicitly on the system's parameters via the diffusive time-scale $\tau = {}^{L^2}/_{\kappa}$.

- If $\tau \gg 1$ – Diffusion is very slow and the scale of data residuals is pre-dominant.
- If $\tau \ll 1$ – Diffusion is very fast and the scale of the physics residuals is pre-dominant.

$\implies$ Geometry of the Pareto front not only depends on α_1, α_2 and α_3 but also on τ !

Architecture	MLP
Hidden layers	2
Neurons per layer	5
Activation	sigmoid
Optimizer	ADAM
Learning rate	0.01
Epochs	10^5



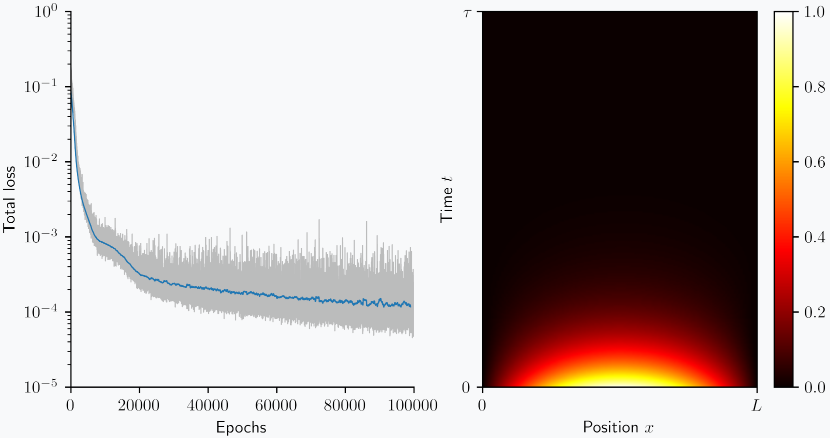


Figure : Convergence history (left) and spatio-temporal diagram of the approximate solution (right) for the parameters $(\kappa, L) = (1, 1)$.

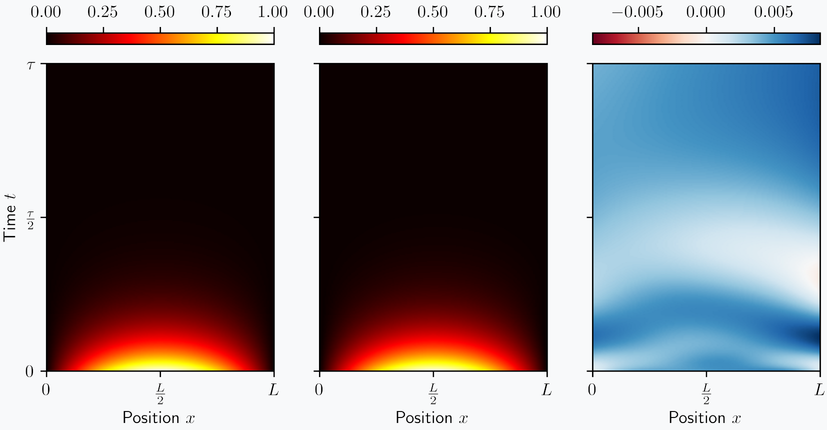
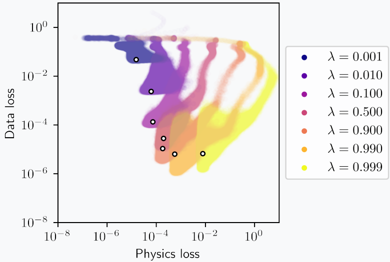
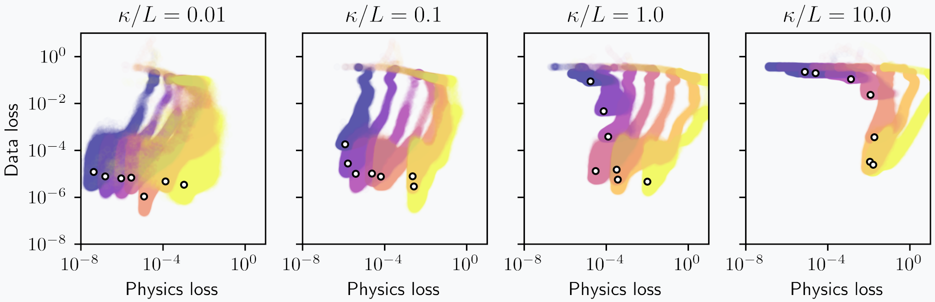


Figure : Ground truth solution (left), approximate solution (center), pointwise error (right).



- A Pareto front naturally emerges from the formulation of the problem.
- The *Physics* term is not merely a regularization.
- Need to proceed carefully with the optimization to ensure a physically acceptable solution.

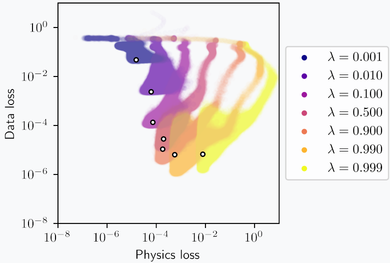


$\kappa/L = 0.01$
$\kappa/L = 0.1$
$\kappa/L = 1.0$
$\kappa/L = 10.0$
Data loss
10^{0}
10^{-2}
10^{-4}
10^{-6}
10^{-8}
Physics loss
10^{-8}
10^{-4}
10^{0}

Physics-Informed Neural Nets

Limitations

PDE and optimization

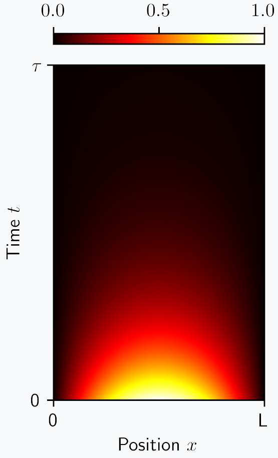


"Solving" an optimization problem

$$\neq$$

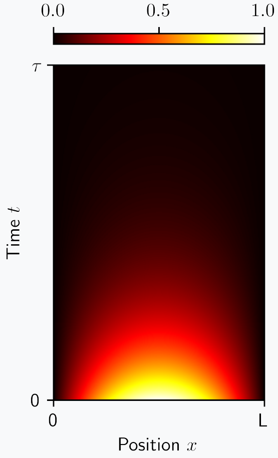
Solving a PDE.

Causality



$$
\begin{cases}
\dfrac{\partial u}{\partial t} = \kappa \dfrac{\partial^2 u}{\partial x^2} \quad \text{for} \quad x \in \,]0, L[\,, \quad t \in [0, T] \\
u(0, t) = u(L, t) = 0 \\
u(x, 0) = \sin\left(\dfrac{\pi x}{L}\right).
\end{cases}
$$

Causality

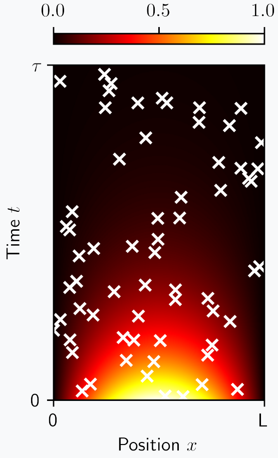


In standard scientific computing, a *time-stepping* approach is used, *i.e.*

$$\mathbf{u}_{t+1} = f(\mathbf{u}_t, \mathbf{u}_{t-1}, \cdots)$$

which naturally enforces the notion of causality.

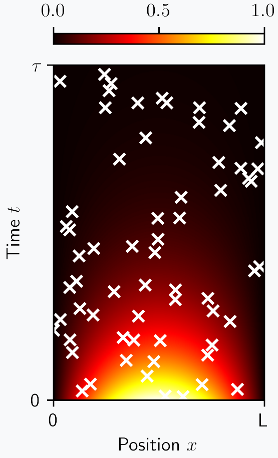
Causality



$$\mathcal{L}_\varphi(\vartheta) = \frac{1}{T \cdot L} \int_0^T \int_0^L |\mathcal{R}(x,t)|^2 \, dxdt$$

$$\simeq \frac{1}{N_\varphi} \sum_{i=1}^{N_\varphi} |\mathcal{R}(x_i, t_i)|^2$$

Causality



$$\mathcal{L}_{\varphi}(\vartheta) \simeq \frac{1}{N_{\varphi}} \sum_{i=1}^{N_{\varphi}} w_i |\mathcal{R}(x_i, t_i)|^2$$

with

$$w_i = \exp\left(-\left(\sum_{t_k \leqslant t_i} |\mathcal{R}(x_k, t_k)|^2\right)\right)$$

Convergence and consistency

Other limitations

- How to ensure conserved quantities are indeed conserved ?

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \rho \mathbf{u} = 0$$

- How to enforce existing symmetries ?

$$\sigma \cdot \frac{\partial u}{\partial t} = f(\sigma \cdot u)$$

- How to efficiently handle fundamentally multiscale problems ?

Physics-Informed Neural Nets

Conclusion

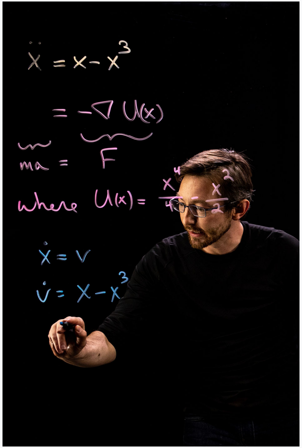
- PINNs provide an innovative way to solve ordinary, partial or stochastic differential equations.
- Some theoretical results exist about the convergence of $u_\vartheta(x, t)$ to the true solution, but these are scarce.

- Training a PINN for the sole purpose of simulation can (usually) be a bad idea.
  - For "simple" enough problems, scientific computing can go a long way!
- Although not discussed here, PINNs can on the other hand be very advantageous to solve ill-posed inverse problems !

Some open problems

- Convergence of the PINN solution to the continuous one beyond elliptic/parabolic equations?
- Rigorous *a priori* error estimates?
- Interplay between PDE type, network architecture, and optimizer.
- etc.

If you want to know more



Steve Brunton (UW, Seattle) has some very good educational resources:

- youtube.com/@Eigensteve
- *Data-driven Science and Engineering*, Cambridge Uni. Press.

Many other resources can be found online! Be curious!

Thank you for your attention!

Any questions?

loiseaujc.github.io