One-shot learning for solution operators of partial differential equations

Lu Lu, Haiyang He, Priya Kasimbeg, Rishikesh Ranade & Jay Pathak

Massachusetts Institute of Technology; *Ansys Inc.*

Email: lu_lu@mit.edu

Abstract

- Discovering governing equations of a physical system, represented by partial differential equations (PDEs), from data is a central challenge.
- Current methods require either some prior knowledge (e.g., candidate PDE terms) to discover the PDE form, or a large dataset to learn a surrogate model of the PDE solution operator.
- We propose the first learning method that only needs one PDE solution, i.e., one-shot learning.
- We first decompose the entire computational domain into small domains, where we learn a local solution operator, and then find the coupled solution via a fixed-point iteration.

Problem setup: Learning solution operators of PDEs

Consider a physical system governed by a PDE defined on a spatio-temporal domain $\Omega \subset \mathbb{R}^d$:

$$\mathcal{F}[u(\mathbf{x}); f(\mathbf{x})] = 0, \quad \mathbf{x} = (x_1, x_2, \ldots, x_d) \in \Omega$$

with suitable initial and boundary conditions. We define the solution operator as

$$\mathcal{G} : f(\mathbf{x}) \mapsto u(\mathbf{x}).$$

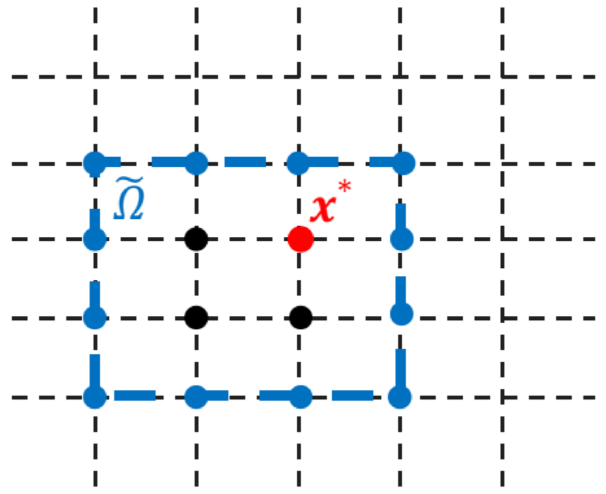
Dataset: $\mathcal{T} = \{(f_i, u_i)\}_{i=1}^{|\mathcal{T}|}$, and (f_i, u_i) is the i-th data point, where $u_i = \mathcal{G}(f_i)$ is the PDE solution for f_i.

Goal: Learn $\mathcal{G}$ from $\mathcal{T}$, such that for a new f, we can predict the corresponding solution $u = \mathcal{G}(f)$.

Extreme difficult scenario: We have only one data point for training, i.e., one-shot learning with $|\mathcal{T}| = 1$, and we let $\mathcal{T} = \{(f_\mathcal{T}, u_\mathcal{T})\}$.

- Assume we can select $f_\mathcal{T}$;
- We only predict f in a neighborhood of some f_0, where we know the solution $u_0 = \mathcal{G}(f_0)$.

Methods: One-shot learning based on locality



Idea: Consider that derivatives and PDEs are defined locally, i.e., the same PDE is satisfied in an arbitrary small domain inside Ω. We partition the entire domain Ω into many small domains, i.e., a mesh of Ω.

Learning the local solution operator via a neural network.

Consider a mesh node at the location $\mathbf{x}^*$ (the red node). If we know the solution u at the boundary of $\tilde{\Omega}$ ($\partial\tilde{\Omega}$) and f within $\tilde{\Omega}$, then $u(\mathbf{x}^*)$ is determined by the PDE. We use a neural network to represent this relationship

$$\tilde{\mathcal{G}} : \{u(\mathbf{x}) : \mathbf{x} \in \partial\tilde{\Omega}\} \cup \{f(\mathbf{x}) : \mathbf{x} \in \tilde{\Omega}\} \mapsto u(\mathbf{x}^*).$$

Training dataset:

- **"Large":** By traversing Ω for all small local domains, we can generate many input-output pairs for training.
- **"Diverse":** We choose $f_\mathcal{T}$ to be uniform random between -1 and 1 on each mesh node, i.e., $f_\mathcal{T}(\mathbf{x})$ is sampled from $U(-1, 1)$.

Prediction via a fixed-point iteration.

For a new $f = f_0 + \Delta f$, we use u_0 as the initial guess of u, and then in each iteration, we apply the trained network on the current solution as the input to get a new solution.

Initiate: $u(\mathbf{x}) \leftarrow u_0(\mathbf{x})$ for all $\mathbf{x} \in \Omega$
while *u has not converged* **do**
  for $\mathbf{x} \in \Omega$ **do**
    $\hat{u}(\mathbf{x}) \leftarrow \tilde{\mathcal{G}}$(the inputs of u and f in $\tilde{\Omega}$)
  Update: $u(\mathbf{x}) \leftarrow \hat{u}(\mathbf{x})$ for all $\mathbf{x} \in \Omega$

References

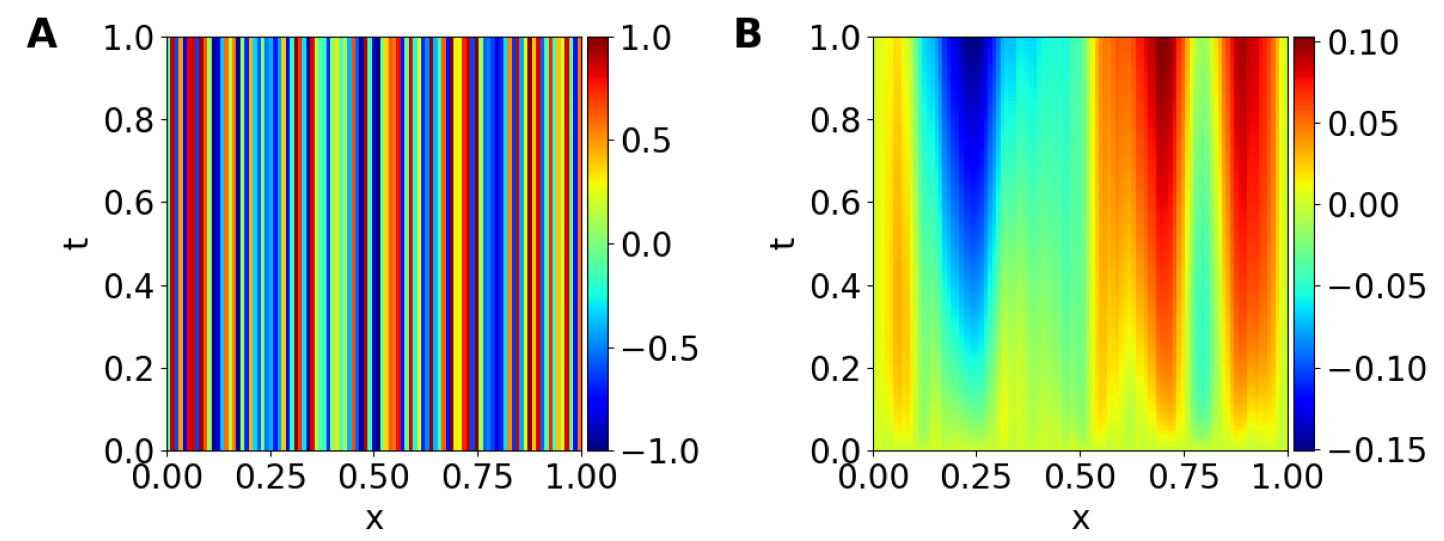
[1] L. Lu, H. He, P. Kasimbeg, R. Ranade, and J. Pathak, "One-shot learning for solution operators of partial differential equations," *arXiv preprint arXiv:2104.05512*, 2021.

Demonstration examples: Nonlinear diffusion-reaction equation

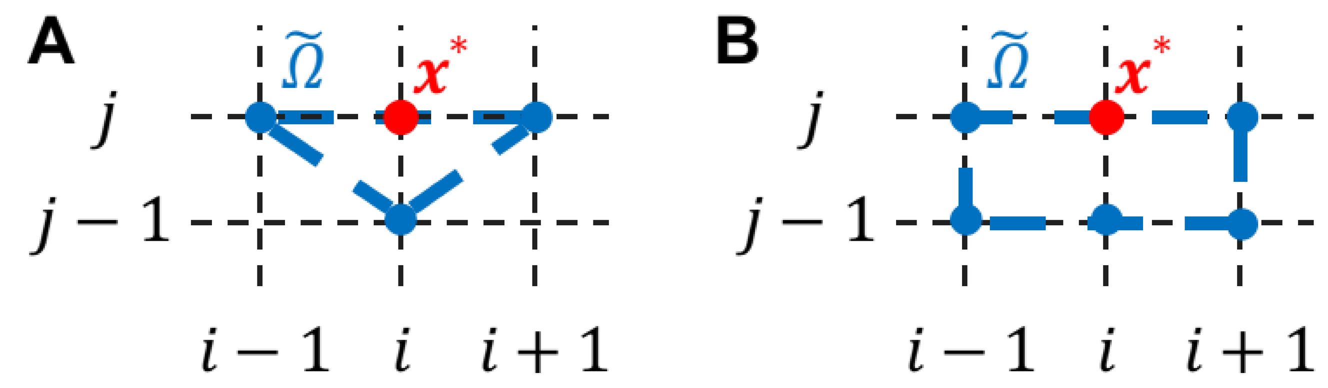
$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2} + ku^2 + f(x), \quad x \in [0, 1], t \in [0, 1],$$

with zero IC/BC. $D = 0.01$ and $k = 0.01$.

Training data for the diffusion-reaction equation:



Local domains $\tilde{\Omega}$ of the diffusion-reaction equation:



Prediction: We randomly sample Δf from a Gaussian random field (GRF): $\Delta f \sim \mathcal{GP}(0, k(x_1, x_2))$, where the covariance kernel is $k(x_1, x_2) = \sigma^2 \exp(-\|x_1 - x_2\|^2/2l^2)$.