Caltech EAS Division of Engineering and Applied Science

Caltech

Partial Differential Equations and Neural Operators

Zongyi Li
May 2022

CS159: Representation Learning for Science
Prof. Yisong Yue

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Overview

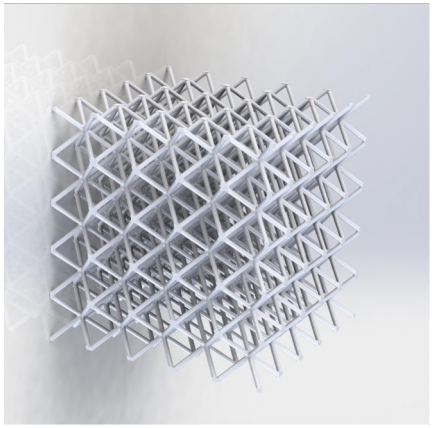
- Partial differential equations (20 min)
  - Examples
  - Numerical solvers
  - PINN
- Operator learning (30 min)
  - Neural operator
  - DeepONet
  - Fourier neural operator
- Examples and extensions (30 min)
  - Weather forecast
  - And more

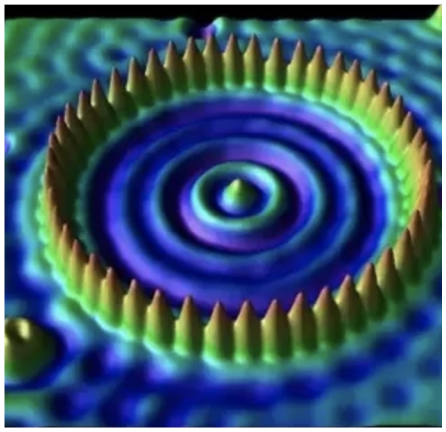
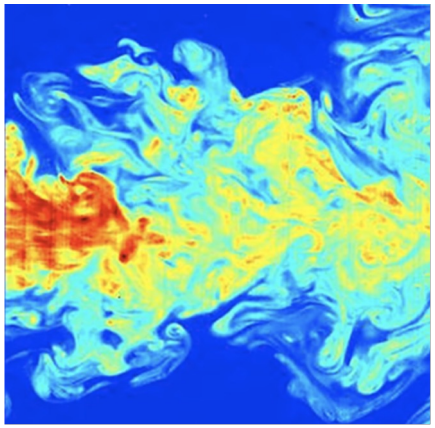
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

1. Introduction

Problems in science and engineering reduce to PDEs.

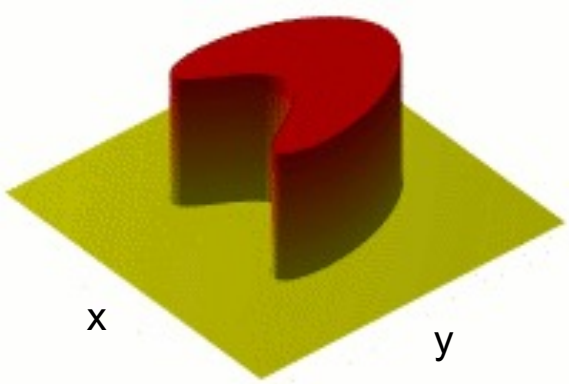


AI BOOTCAMP

Example: heat equation

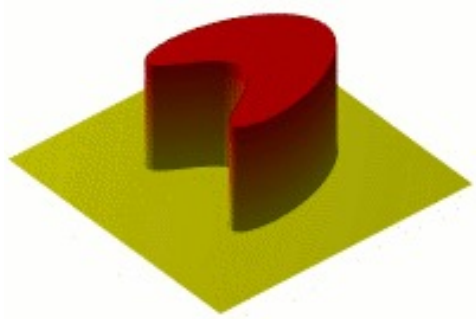
2D parabolic PDE



How long will my pan cool down?

Figure: Oleg Alexandrov

Example: heat equation



Spatial domain: $\mathrm{D} = [0,1]\times[0,1]$

Time domain: $\mathrm{T} = [0,1]$

Points: $x = (x, y) = (x_1, x_2) \in D, \quad \mathrm{t} \in T$

Solution function (Temperature) $u: T\times D \to \mathbb{R}$

Heat equation:

$$u_t = \Delta u$$

Where $\Delta u = u_{xx} + u_{yy}$

"The change in time is equal to 2nd-order difference in space."

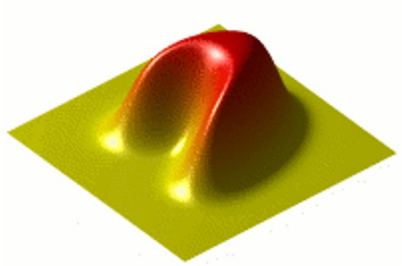
Caltech EAS Division of Engineering and Applied Science

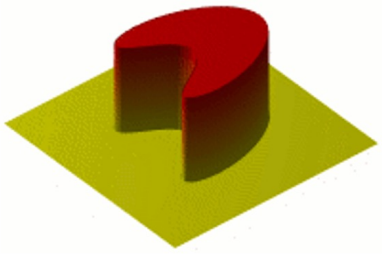
Initial value problems

Heat equation:

$$u_t = \Delta u$$

Initial value problem: given the temperature at time t=0,
What is the temperature a time t=1?
Given u(0,x), what is u(1,x)?





AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

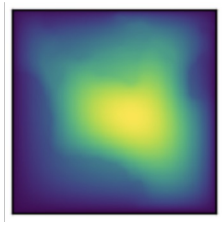
Problems with coefficients

2D elliptic PDE:

$$-\nabla \cdot (a(x)\nabla u(x)) = f(x), \quad x \in D$$

$\nabla u = (u_x, u_y)$
$\Delta u = \nabla \cdot \nabla u$

Input: a(x)



Output: u(x)

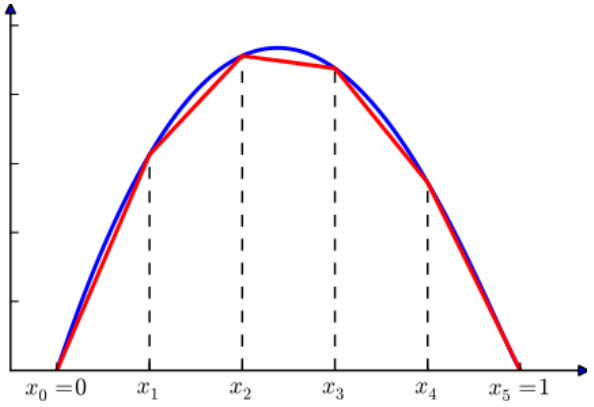
My domain has two types of media, described by a(x)
Given a(x), what is u(x)?

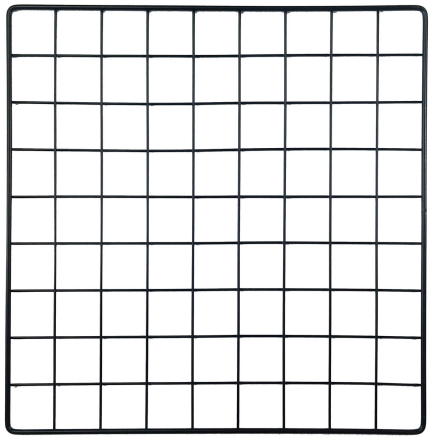
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Numerical solver

Idea: discretize the problem onto some grid
Solve a multi-variable equation of $\{u(i,j)\}, \quad i = 0, \dots, 10, j = 0, \dots, 10$
Solve a linear system Au = f





AI BOOTCAMP

Numerical solver

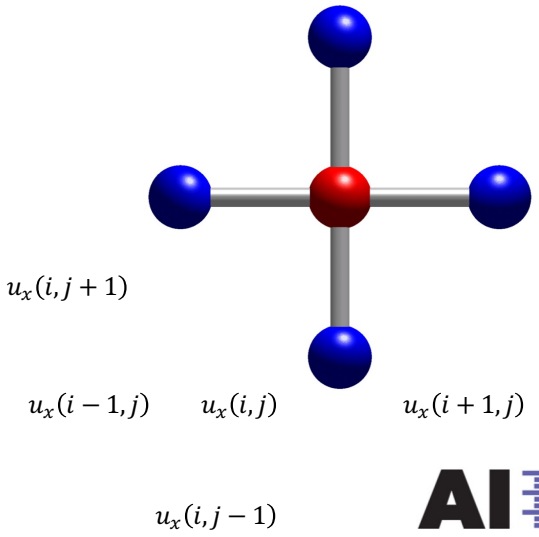
Idea: discretize the problem onto some grid
Solve a multi-variable equation of $\{u(i,j)\}$, $i = 0, \ldots, 10, j = 0, \ldots, 10$

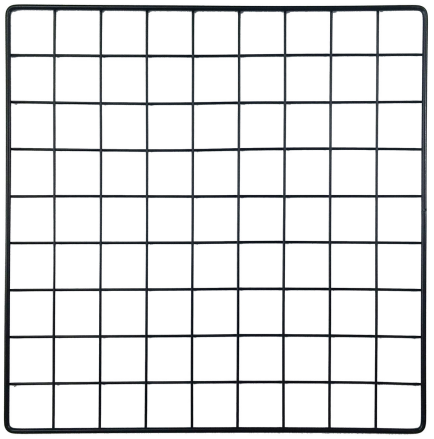
How to compute the derivative?
Finite difference method (FDM): approximate by its neighbor points

$$u_x(x) = \lim_{dx \to 0} \frac{[u(x+dx) - u(x)]}{dx}$$

$$u_x(x) \sim \frac{[u(x+dx) - u(x)]}{dx}$$

$$u_x(i,j) \sim \frac{[u(i+1,j) - u(i,j)]}{dx}$$





AI BOOTCAMP

Numerical solver

Pipeline:

1. Discretize the space
2. Write out a linear system
3. Solve the linear system

Common numerical methods:

- Finite difference methods
- Finite element methods
- Spectral methods
- Iterative methods

Trade-off: finer grids are more accurate, but also more expensive.

Caltech EAS Division of Engineering and Applied Science

Physics-informed neural networks

Heat equation:

$$u_t = \Delta u$$

Physics-informed neural networks (PINNs)

- Idea: represent function u as a neural network
- Compute the derivatives $(u_t, \Delta u)$ using the chain rule
- Solve the system using gradient descent methods

PINN: M Raissi, P Perdikaris, GE Karniadakis

AI BOOTCAMP

Recap: PINNs

Heat equation:

$$u_t = \Delta u$$

Pipeline:

- Initialize NN u(x)
- For N iterations
- Sample collocation points $\{(x, t)\}$
- Compute the derivatives $u_t(x, t),\ \Delta u(x, t)$
- Minimize $|| u_t - \Delta u ||$

PINN: M Raissi, P Perdikaris, GE Karniadakis

Recap: PINNs

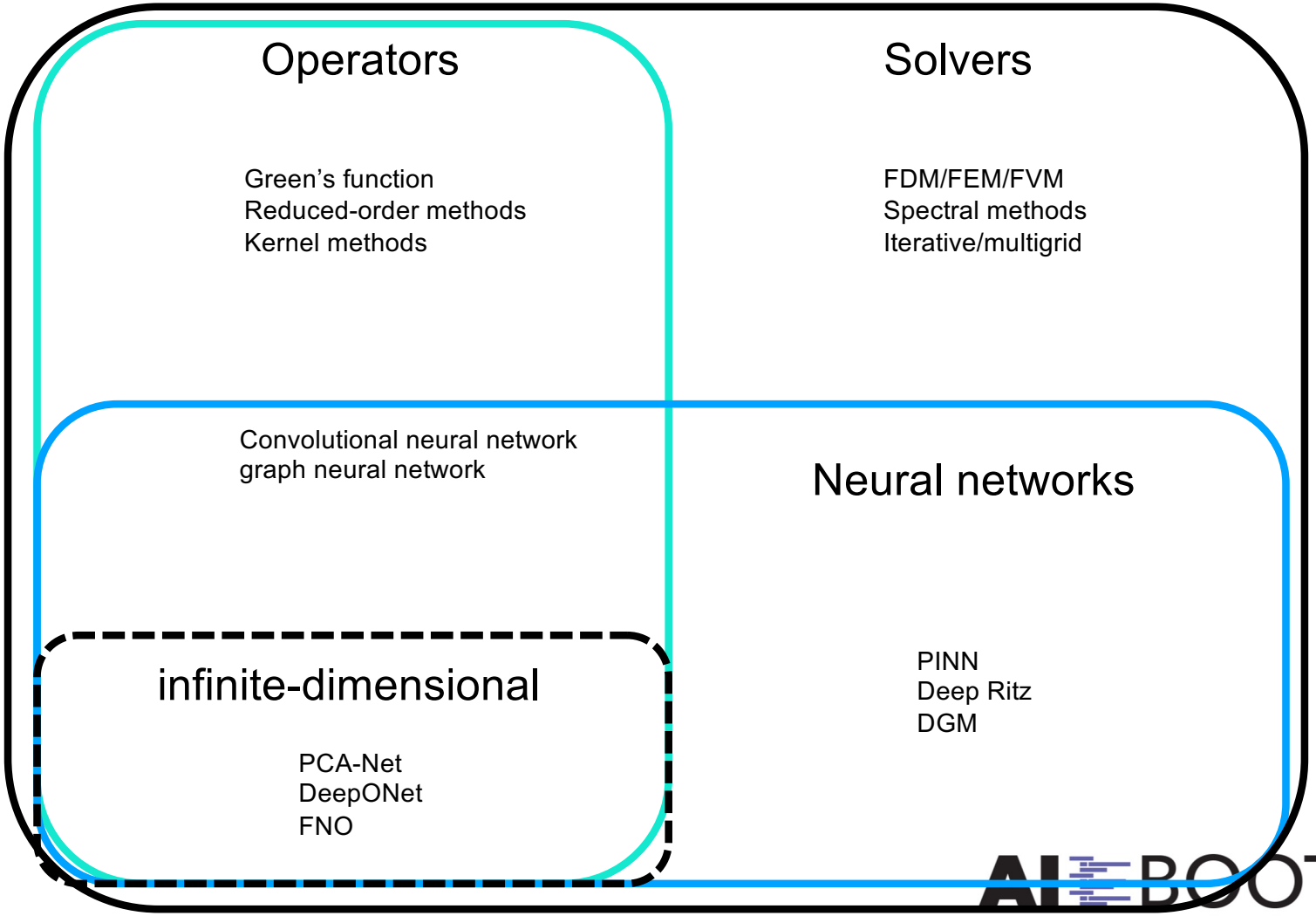
Pros:

- Simple and flexible
- No need to worry about stability conditions, etc
- Complex geometry, high-dim, inverse problem, etc

Cons:

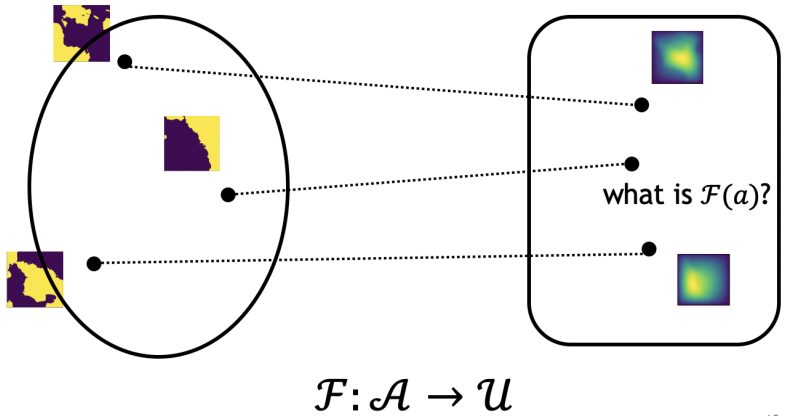
- Optimization is tricky
- Less accurate than existing solvers (FDM/FEM)
- No guarantee of (optimization)

PINN: M Raissi, P Perdikaris, GE Karniadakis


Operators
Green's function
Reduced-order methods
Kernel methods
Solvers
FDM/FEM/FVM
Spectral methods
Iterative/multigrid
Convolutional neural network
graph neural network
Neural networks
infinite-dimensional
PCA-Net
DeepONet
FNO
PINN
Deep Ritz
DGM

Caltech EAS Division of Engineering and Applied Science

2. Operator learning



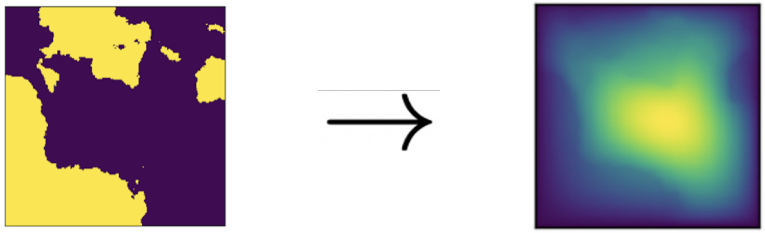
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Operator learning

Operators are map between function space.
Given a dataset of input-output pairs, find the map (operator)

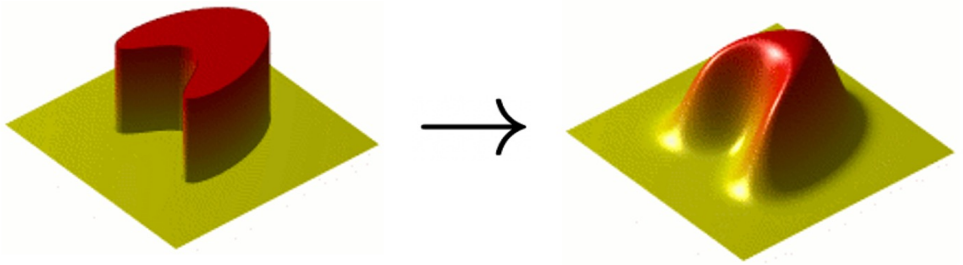
F: a->u



Input: coefficient

Output: solution

F: u0->u1



Input: initial

Output: solution

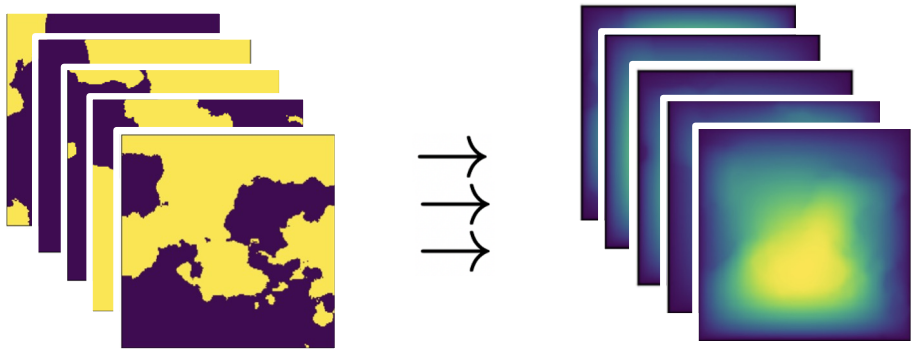
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Operator learning

Operators are map between function space.
Learn the Operators from data (coefficients & solutions pairs).

- Fix an equation
- Multiple training instances
- Learn the mapping



Input: coefficients

Output: solutions

$$\mathcal{F} : \mathcal{A} \times \Theta \to \mathcal{U}$$

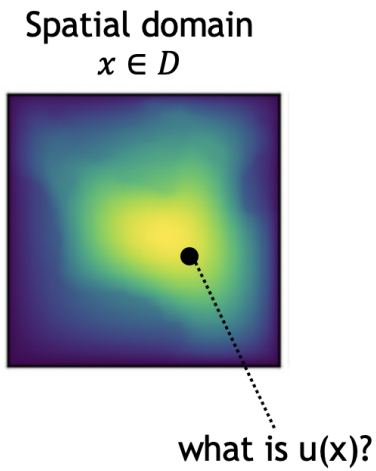
AI BOOTCAMP

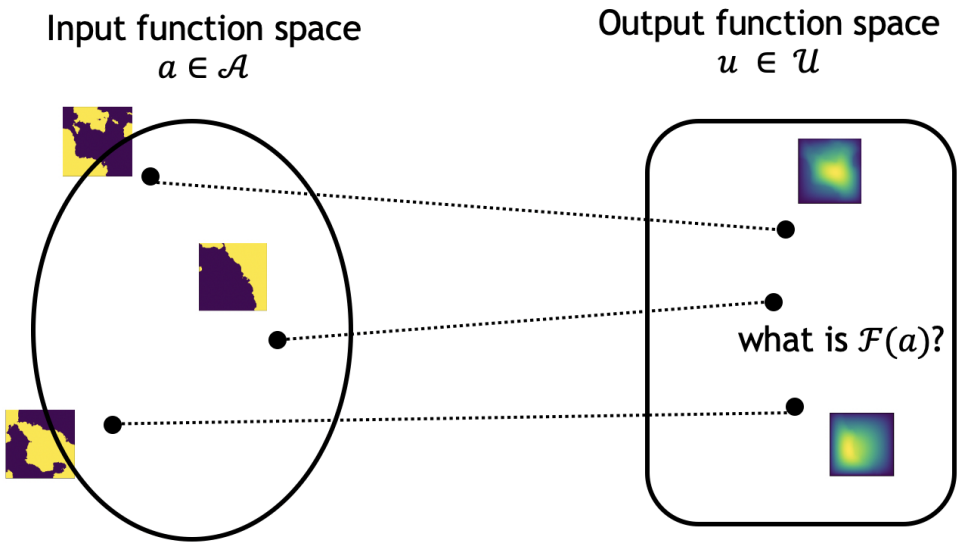
Caltech EAS Division of Engineering and Applied Science

Solve vs learn

Solving for a PDE instance u
approximate $u(x)$ in the spatial space.

learn the solution operator $\mathcal{F}$
interpolate u in the function space.





$$\mathcal{F}: \mathcal{A} \to \mathcal{U}$$

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Pipeline of operator learning

Supervised learning

Get a dataset {(a , u)} (existing solvers or experiments)

- Initialize NN, F: a->u
- For N epoch, For batches of data pairs (a, u)
- Compute the prediction F(a)
- Minimize ||F(a)-u||

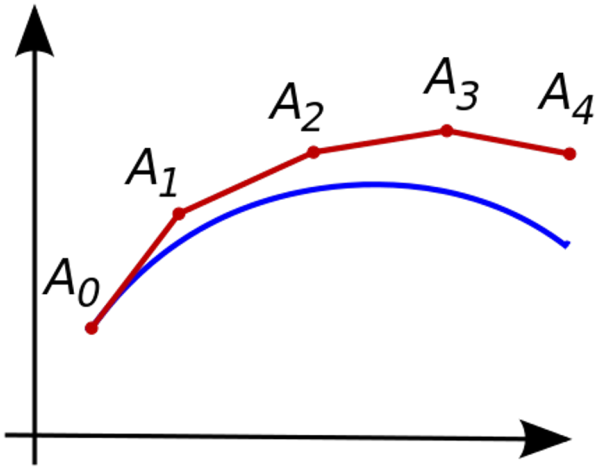
AI BOOTCAMP

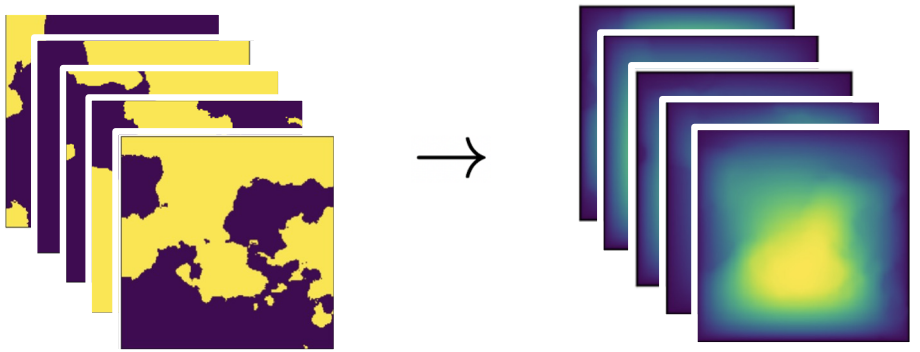
Caltech EAS Division of Engineering and Applied Science

Solve vs learn

Conventional methods:
Solve the equation
By approximation on a mesh

Data-driven methods:
Learn the trajectory
From a distribution





Input: coefficients

Output: solutions

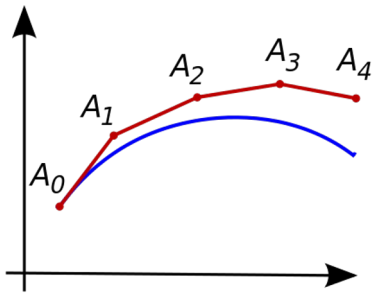
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Solve vs learn

Conventional methods:

- Solve for any parameters
- Worst case guarantees
- Consistency



Data-driven methods:

- Parameters from a distribution
- Less guaranteed
- Not “consistent”

Input: coefficients

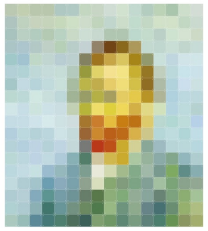
Output: solutions

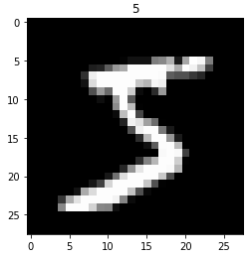
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Architecture design

- Not vector-to-vector mapping.
- But function-to-function mapping.

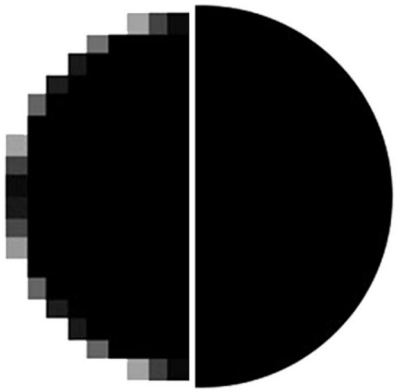


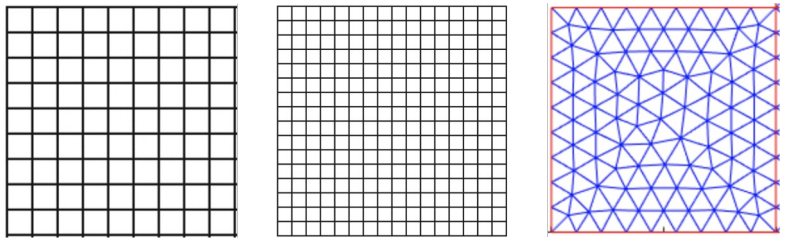


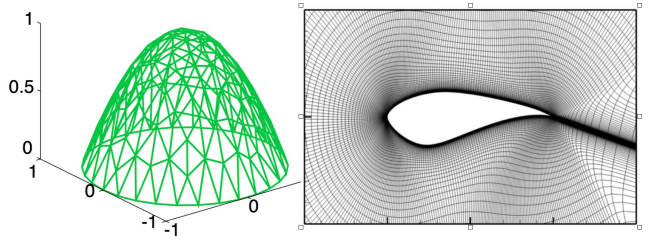
Vocabulary:
Man, woman, boy, girl, prince, princess, queen, king, monarch

	1	2	3	4	5	6	7	8	9
man	1	0	0	0	0	0	0	0	0
woman	0	1	0	0	0	0	0	0	0
boy	0	0	1	0	0	0	0	0	0
girl	0	0	0	1	0	0	0	0	0
prince	0	0	0	0	1	0	0	0	0
princess	0	0	0	0	0	1	0	0	0
queen	0	0	0	0	0	0	1	0	0
king	0	0	0	0	0	0	0	1	0
monarch	0	0	0	0	0	0	0	0	1

Discretized vector







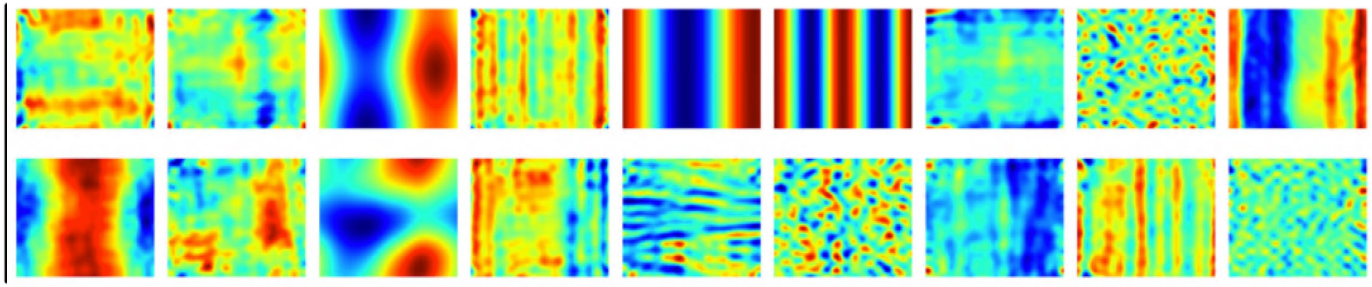
Continuous function

Caltech EAS Division of Engineering and Applied Science

CNN vs Neural operators

Key idea: represent function & operator in mesh-invariant way

Filters in CNN



Fourier Filters

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

3.1 Neural operator

$$u = (K_l \circ \sigma_l \circ \cdots \circ \sigma_1 \circ K_0)\, v$$

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu,
Kaushik Bhattacharya, Andrew Stuart, Anima Anandkumar

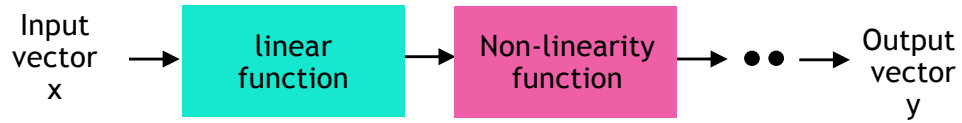
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Neural networks

f: $x \to y$

$R^n \to R^m$



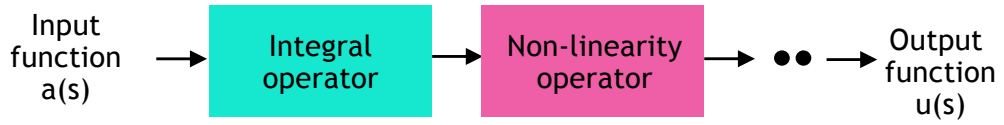
$$
\begin{aligned}
v_0 &= x \\
v_{l+1} &= \sigma\left(A_l v_l + b_l\right), \quad l = 0, \ldots, L-1 \\
y &= A_L v_L + b_L \\
\sigma : \mathbb{R} \to \mathbb{R}, \quad A_l \in \mathbb{R}^{d_{l+1} \times d_l}, \quad b_l &\in \mathbb{R}^{d_{l+1}}
\end{aligned}
$$

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Neural operators

F: a(s) → u(s)
$(R^n \to R^m) \to (R^n \to R^m)$



$$v_0(s) = P(x(s), s)$$

$$v_{l+1}(s) = \sigma\left(W_l v_l(s) + \int_D \kappa_l(s,z) v_l(z)\, dz + b_l(s)\right), \quad l = 0, \ldots, L-1$$

$$y(s) = Q(v_L(s))$$

$$P : \mathbb{R}^{m+n} \to \mathbb{R}^{d_0}, \quad W_l \in \mathbb{R}^{d_{l+1} \times d_l}, \quad \kappa_l : \mathbb{R}^{2n} \to \mathbb{R}^{d_{l+1} \times d_l}, \quad b_l : \mathbb{R}^n \to \mathbb{R}^{d_{l+1}}, \quad Q : \mathbb{R}^L \to \mathbb{R}^r$$

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Approximation bound

For any continuous operator, there exists a two-layers neural operators can approximate it.

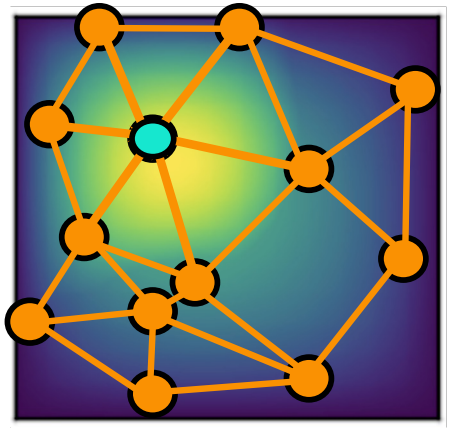
For the solution operator of Navier-Stokes equation, the number of parameters depends sub-linearly on the error

Neural Operator: Learning Maps Between Function Spaces. Kovachki et. al.
On universal approximation and error bounds for Fourier Neural Operators. Kovachki et. al.

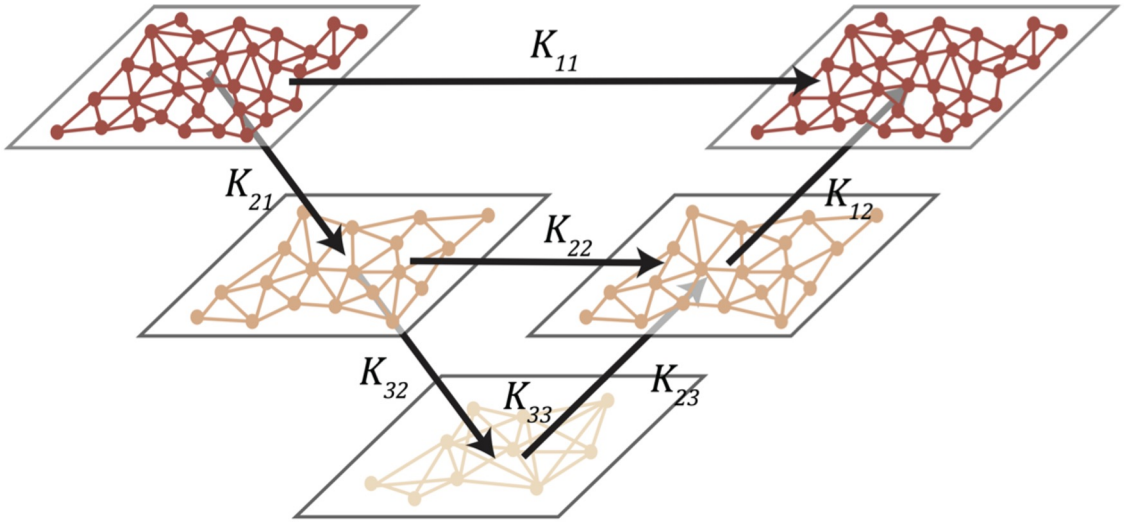
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Graph-based neural operators



GKN



MGKN

https://arxiv.org/abs/2003.03485

https://arxiv.org/abs/2006.09535 (Neurips2020)

AI BOOTCAMP

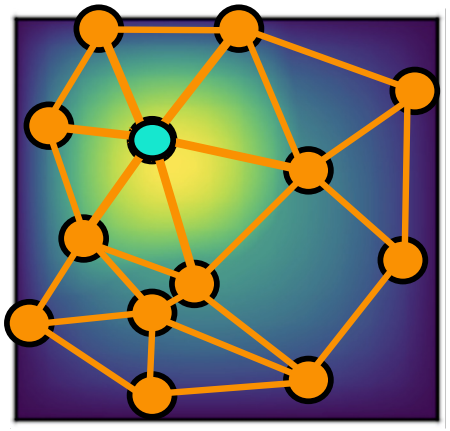
Caltech EAS Division of Engineering and Applied Science

Kernel convolution as message passing on graph

$$v_{t+1}(x) = \sigma\left(W v_t(x) + \int_D \kappa_\phi(x, y, a(x), a(y)) v_t(y)\, \nu_x(dy)\right)$$

$$v_{t+1}(x) = W v_t(x) + \sum_{y \in N(x)} \kappa_\phi\big(e(x,y)\big) v_t(y)$$

Graph neural network

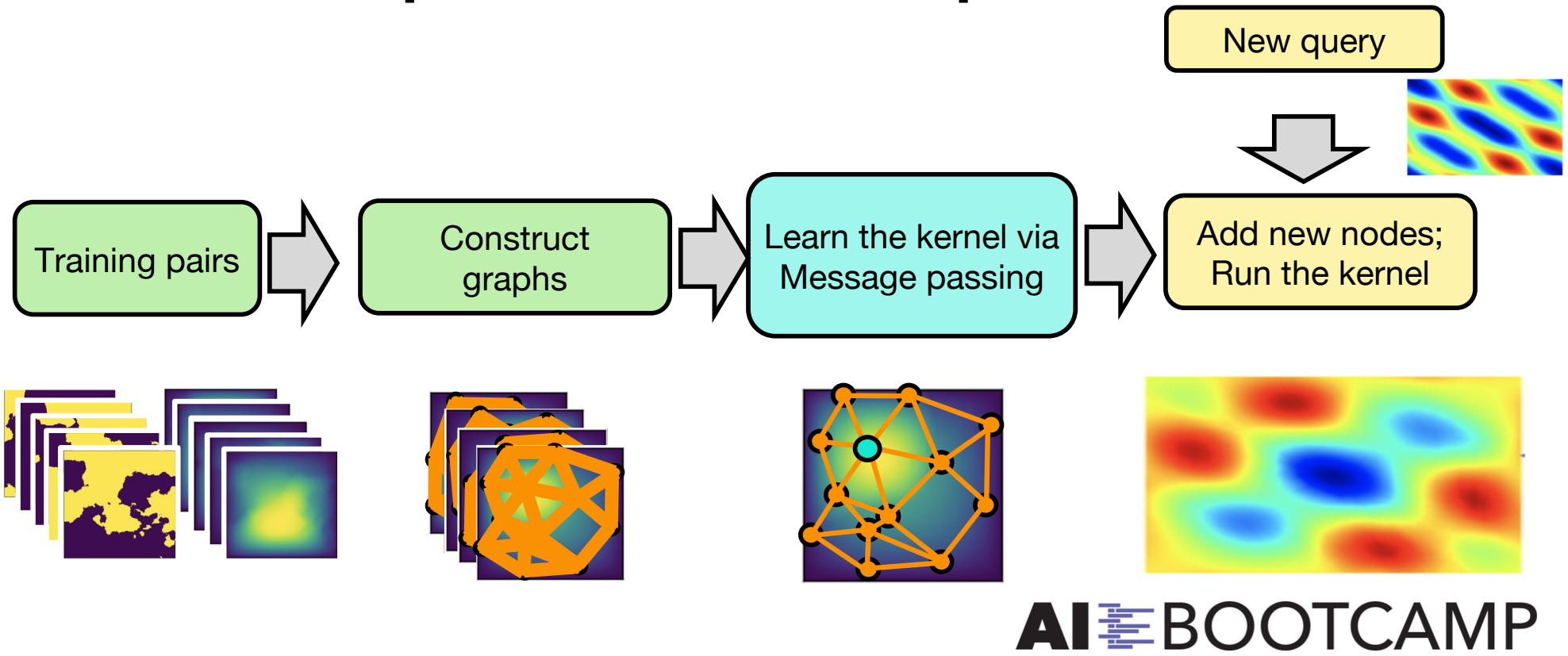


- Adjacency matrix = kernel matrix.
- Kernel integration = message passing

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

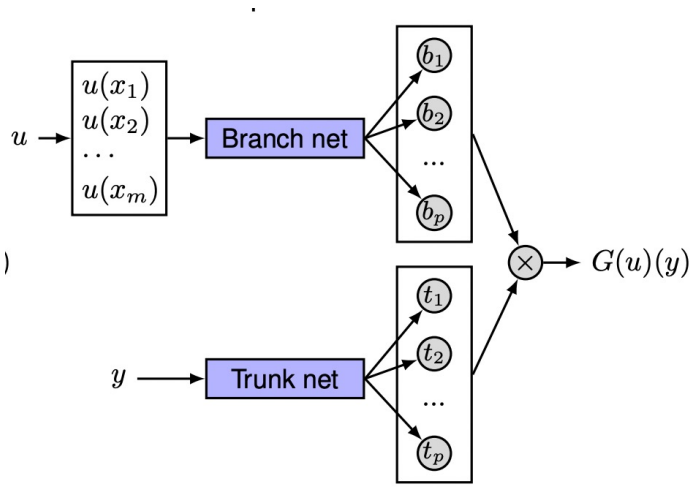
Graph Kernel Operator



AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

3.2 Deep Operator Network



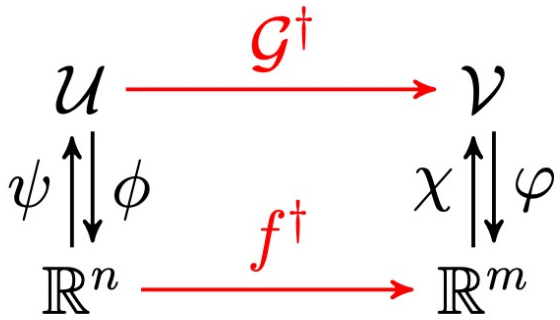
DeepONet: Lu Lu, Pengzhan Jin, George Em Karniadakis

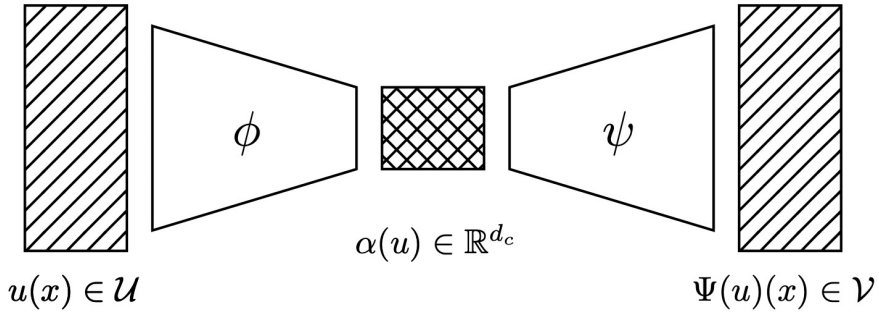
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Encoder-decoder operator

PCA-Net, DeepONet:
Define map of finite dimensional subspace with a fixed number of basis





Caltech EAS Division of Engineering and Applied Science

Parameterize infinite dimensional map by finite dimensional parameters

PCA-Net, DeepONet:
Define map of finite dimensional subspace with a fixed number of basis

Method	Encoding	Finite-dim. Mapping	Reconstruction
FEM	Galerkin projection	Numerical scheme	Finite element basis
FVM	Cell averages	Numerical scheme	Piecewise polynomial
FD	Point values	Numerical scheme	Interpolation
PCA-Net	PCA projection	Neural network	PCA basis
DeepONet	Linear encoder	Neural network	Neural network basis

NIKOLA B. KOVACHKI, SAMUEL LANTHALER, AND ANDREW M. STUART

Caltech EAS Division of Engineering and Applied Science

DeepONet

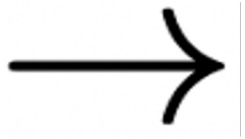
F: [a(x), y] → u(y)

$[(R^n \rightarrow R^m), R^n] \rightarrow R^m$

Input functions + query point → the solution at the query point

Input: a(x)

(y)



u(y)

DeepONet: Lu Lu, Pengzhan Jin, George Em Karniadakis

AI BOOTCAMP

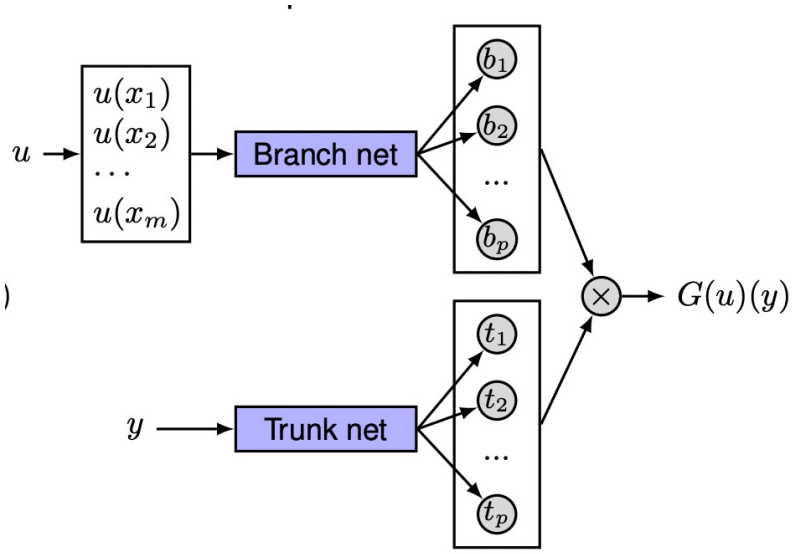
Caltech EAS Division of Engineering and Applied Science

DeepONet

Two networks:

- Branch net takes the input function
- Trunk net takes the query point

Input: a(x)



u(y)

(y)

DeepONet: Lu Lu, Pengzhan Jin, George Em Karniadakis

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

DeepONet and Neural operator

Two slightly different formulations.
Potentially one can be converted into the other.

Implementation difference:

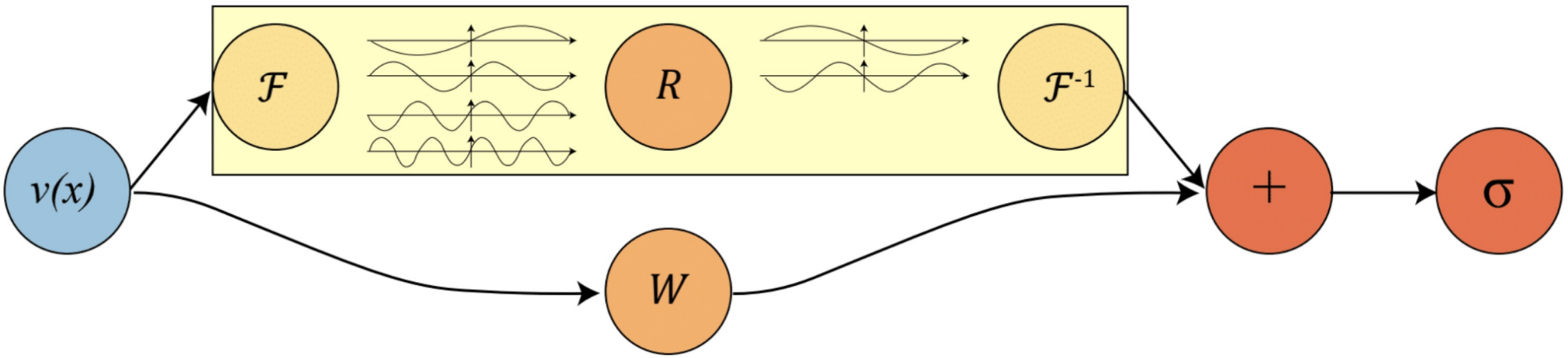
- Neural operator is easier if input and output are fully observed functions.
- DeepONet is easier when given sparse observation from sensors a(x) and query u(y).

DeepONet: Lu Lu, Pengzhan Jin, George Em Karniadakis

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

3.3 Fourier neural operator



https://arxiv.org/abs/2006.09535 (ICLR2021)

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu,
Kaushik Bhattacharya, Andrew Stuart, Anima Anandkumar

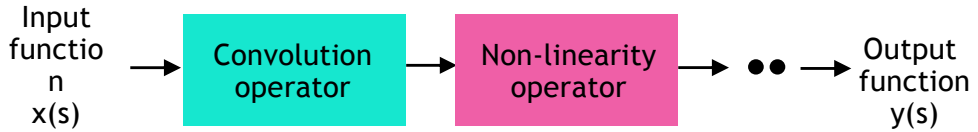
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Fourier neural operators

F: $x(s) \to y(s)$

$(R^n \to R^m) \to (R^n \to R^m)$



Architecture

$$v_{l+1}(s) = \sigma\left(W_l v_l(s) + \int_D \kappa_l(s,z) v_l(z)\, dz + b_l(s)\right)$$

Fourier space

- Assume $\kappa_l(s,z) = \kappa_l(s-z)$ and parametrize its Fourier components θ_l.
  - Convolution theorem: $\int_D \kappa_l(s-z) v_l(z)\, dz = \mathcal{F}^{-1}(\theta_l \cdot \mathcal{F}(v_l))(s) \quad \mathcal{O}(n \log n)$

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Fourier layer

Use convolution as the integral operator
and implement with Fourier transform

$$\big(\mathcal{K}(a;\phi)v_t\big)(x) := \int_D \kappa\big(x, y, a(x), a(y); \phi\big) v_t(y) \mathrm{d}y,$$

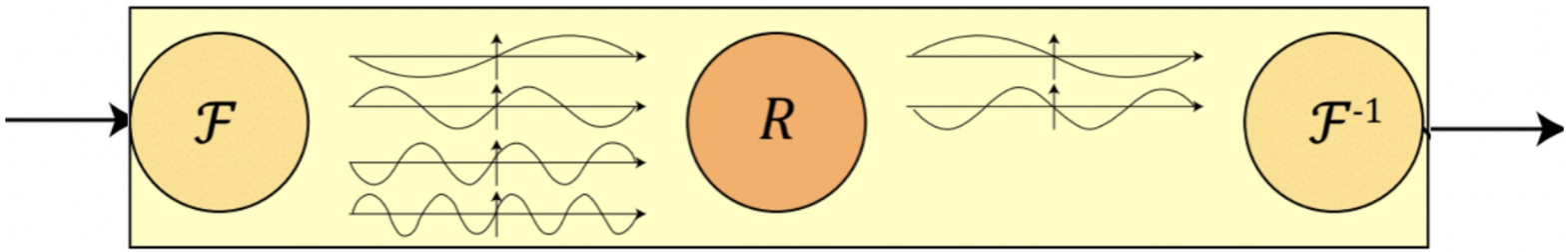
$$\big(\mathcal{K}(\phi)v_t\big)(x) = \mathcal{F}^{-1}\Big(R_\phi \cdot (\mathcal{F}v_t)\Big)(x)$$

AI BOOTCAMP

Fourier layer

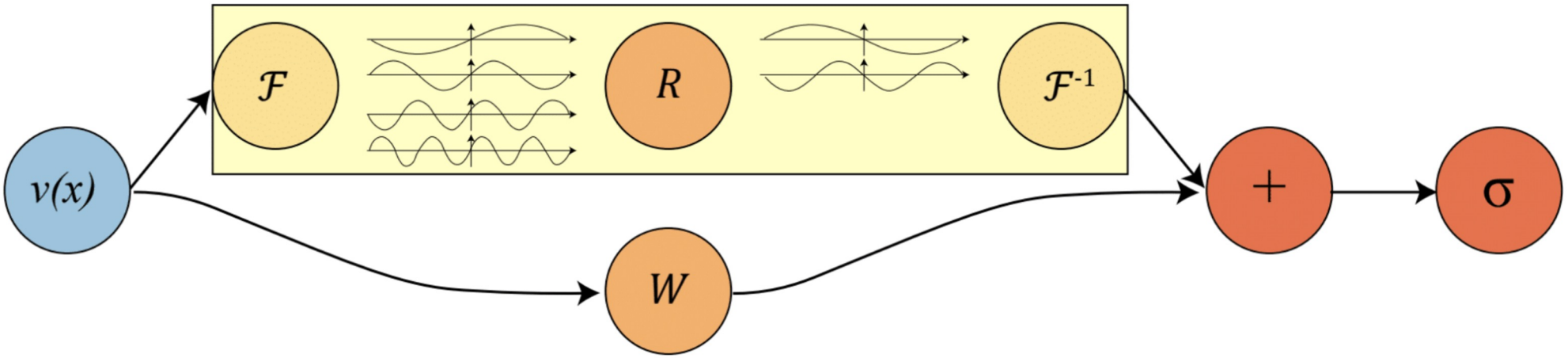
1. Fourier transform
2. Linear transform
3. Inverse Fourier transform

$$\left(\mathcal{K}(\phi)v_t\right)(x) = \mathcal{F}^{-1}\Big(R_\phi \cdot (\mathcal{F}v_t)\Big)(x)$$



Fourier layer

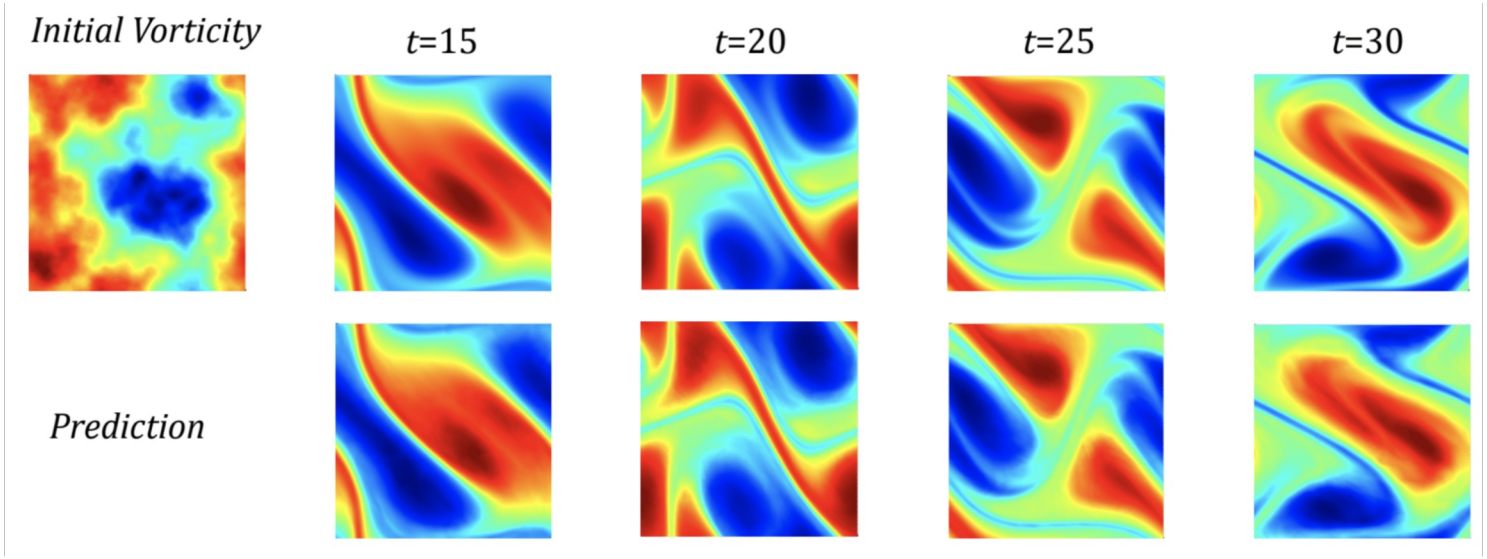
The linear transform *W* outside keep the track of the location information (x) and non-periodic boundary



$$v_{t+1}(x) = \sigma\left(W v_t(x) + \int_D \kappa_\phi(x, y, a(x), a(y)) v_t(y)\, \nu_x(dy)\right)$$

Caltech EAS Division of Engineering and Applied Science

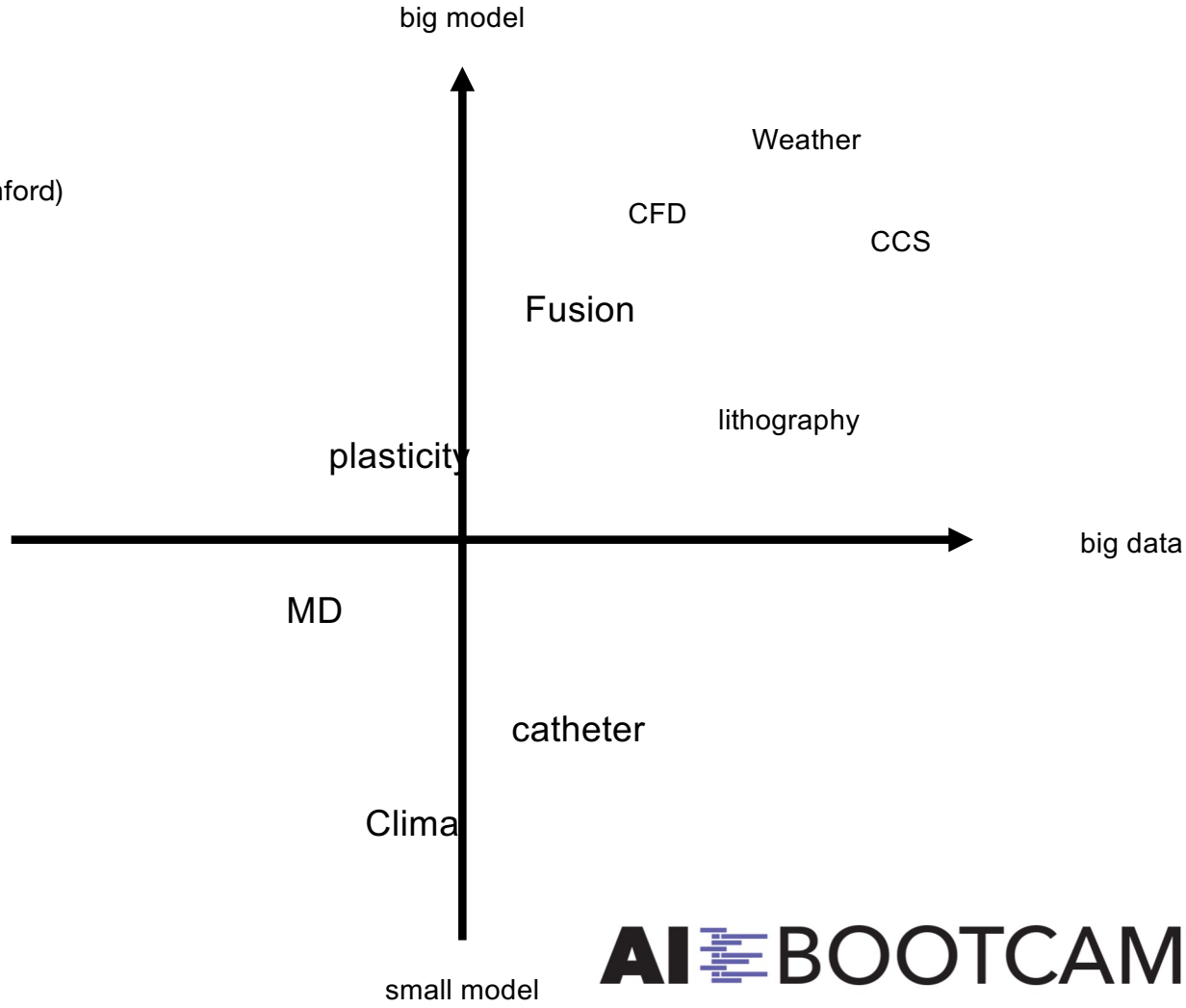
4. Experiments



AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

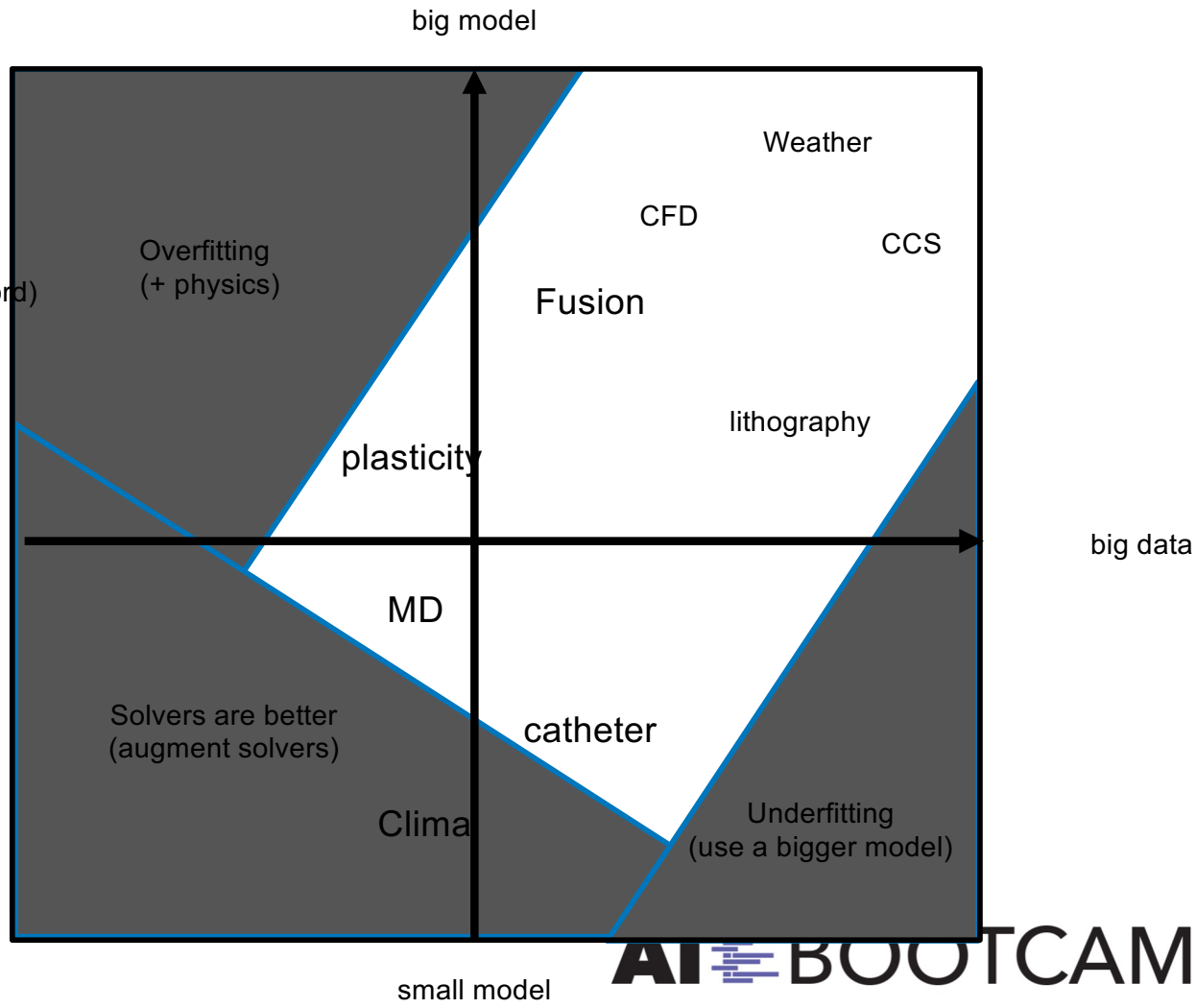
- Weather (Nvidia)
- Carbon Capture & Storage (Stanford)
- Lithography (Nvidia)
- Fluid Mechanics (Caltech)
- Solid Mechanics (Caltech)
- Molecular Dynamics (Argonne)
- Fusion (UK atomic)
- Catheter (Caltech)
- Clima Cloud (Caltech)



AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

- Weather (Nvidia)
- Carbon Capture & Storage (Stanford)
- Lithography (Nvidia)
- Fluid Mechanics (Caltech)
- Solid Mechanics (Caltech)
- Molecular Dynamics (Argonne)
- Fusion (UK atomic)
- Catheter (Caltech)
- Clima Cloud (Caltech)



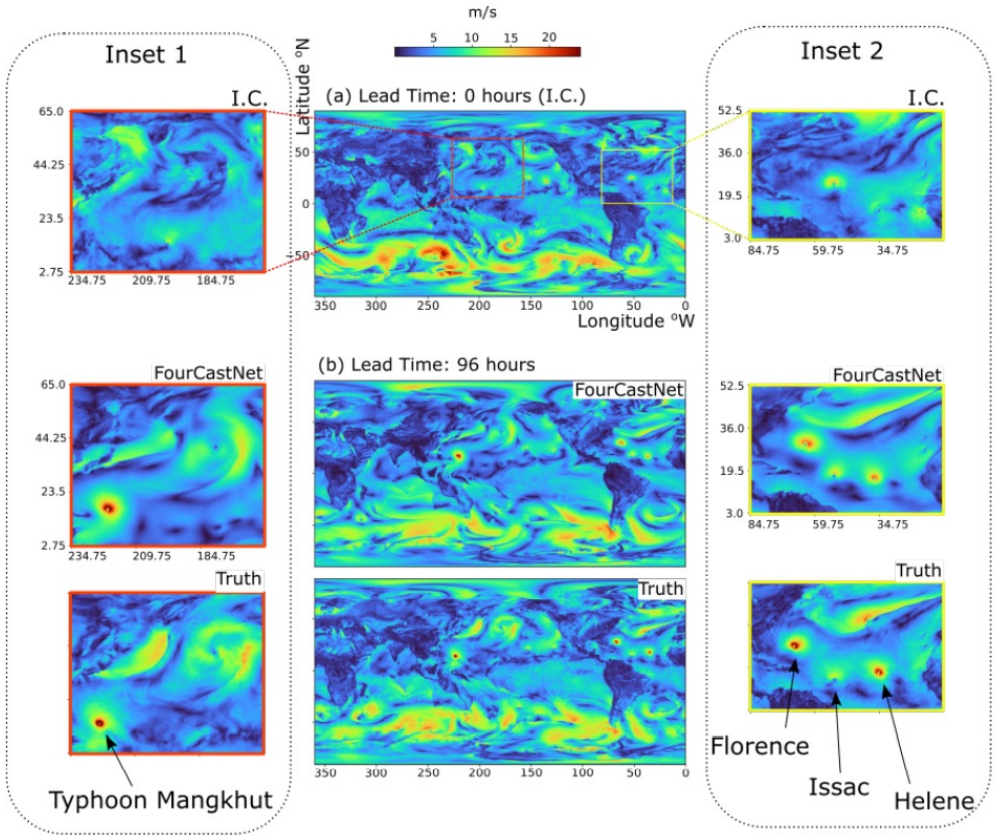
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Weather Forecast

Weather forecast: the AFNO model on the ERA5 dataset

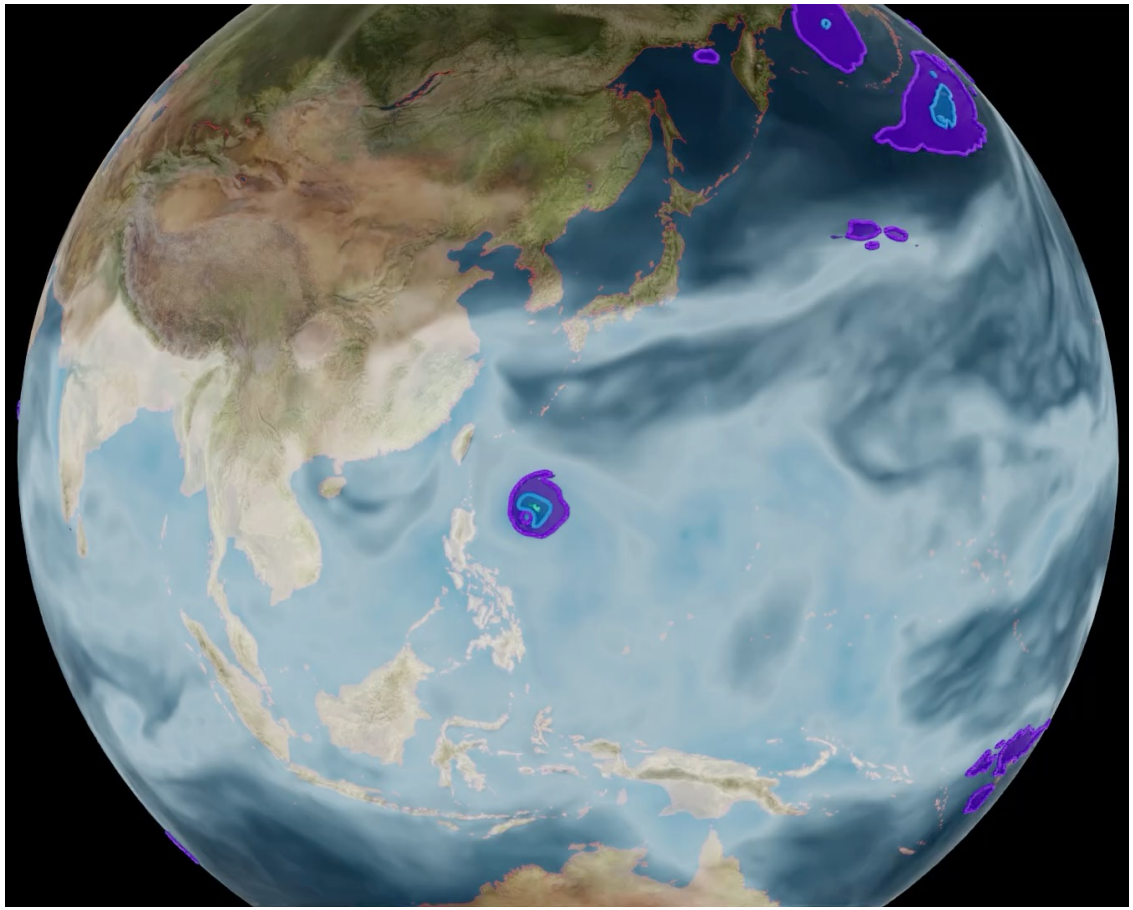
Input the current weather variables, output the weather at 6 hours later.

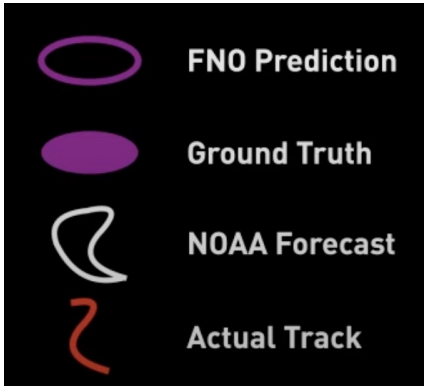


Pathak et al. (2022), FourCastNet: A Global Data-driven High-resolution Weather Model using Adaptive Fourier Neural Operators, arXiv: https://arxiv.org/abs/2202.11214

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science



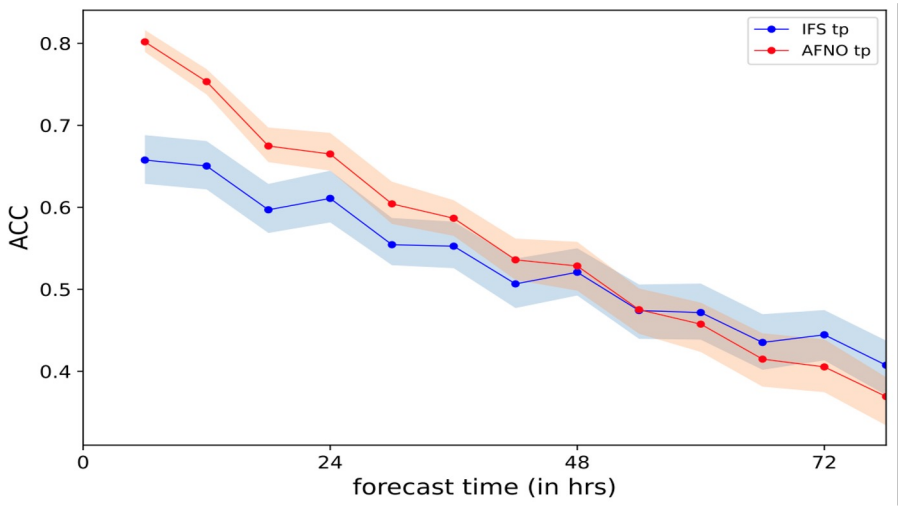


AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Weather Forecast

- Unprecedented skill on precipitation forecasts
- 6X higher resolution than other DL models
- 1000-member ensemble in a fraction of a second
- 4 to 5 orders of magnitude speedup over NWP
- 25000X smaller energy footprint

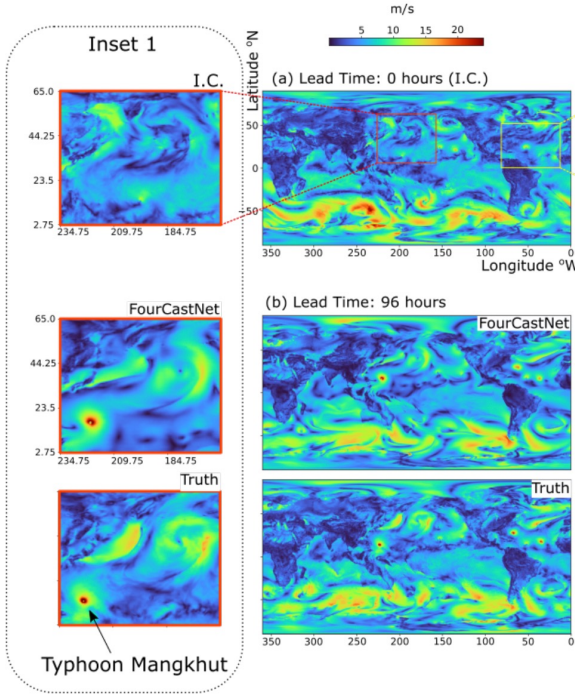


Pathak et al. (2022), FourCastNet: A Global Data-driven High-resolution Weather Model using Adaptive Fourier Neural Operators, arXiv: https://arxiv.org/abs/2202.11214

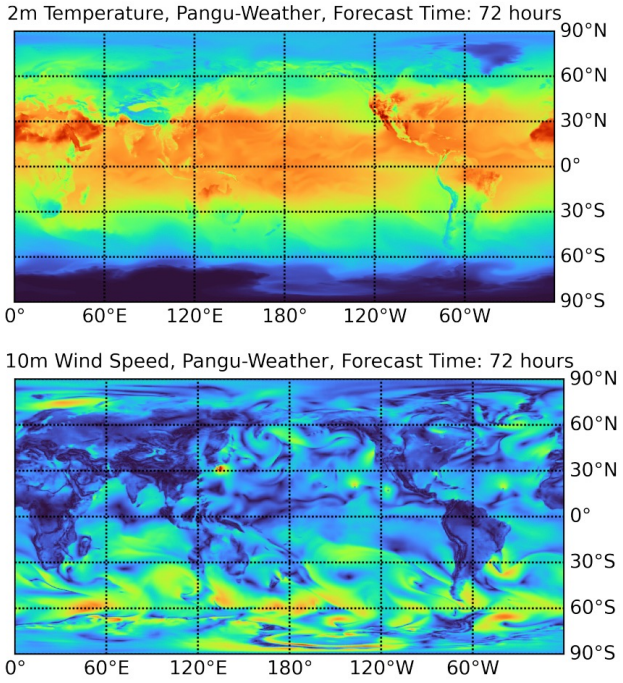
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

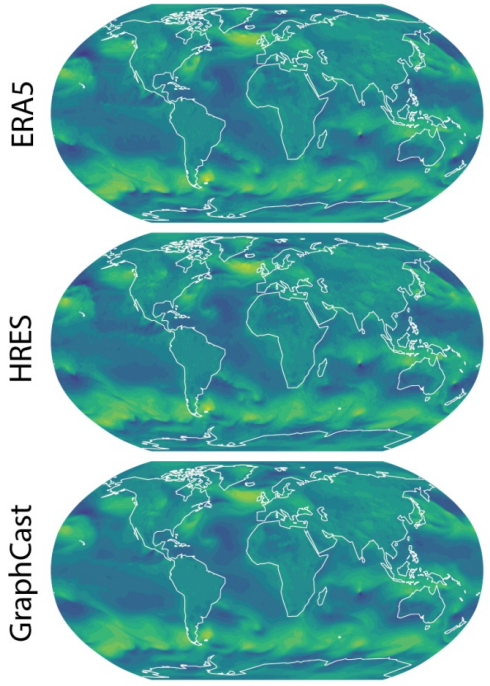
Weather Forecast



Fourcastnet [1] PSHRCMKHLAHKA
Jaideep Pathak et. al. Feb 2022



Pangu-weather [11]
Kaifeng Bi el. al. Nov 2022



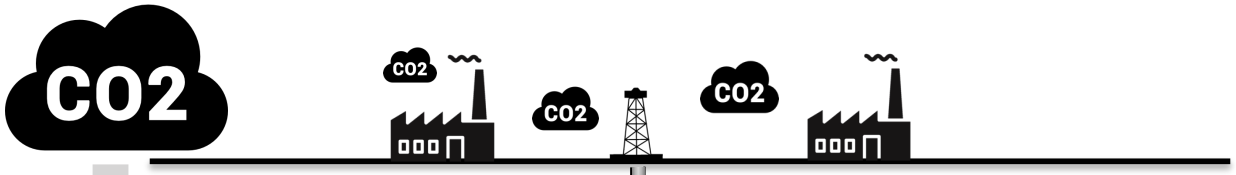
GraphCast [12]
Rem Lam el. al. Dec 2022

AI BOOTCAMP

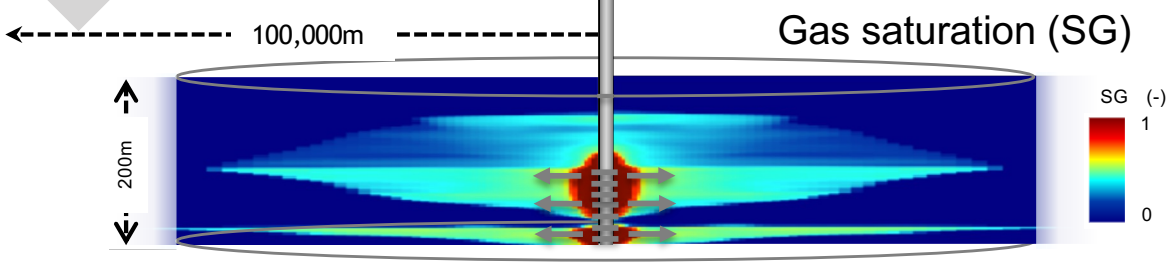
Caltech EAS Division of Engineering and Applied Science

CO_2 injection

Inject CO_2 into the earth



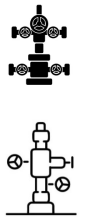
Measure CO_2 migration



FNO provides 10^5x speedup

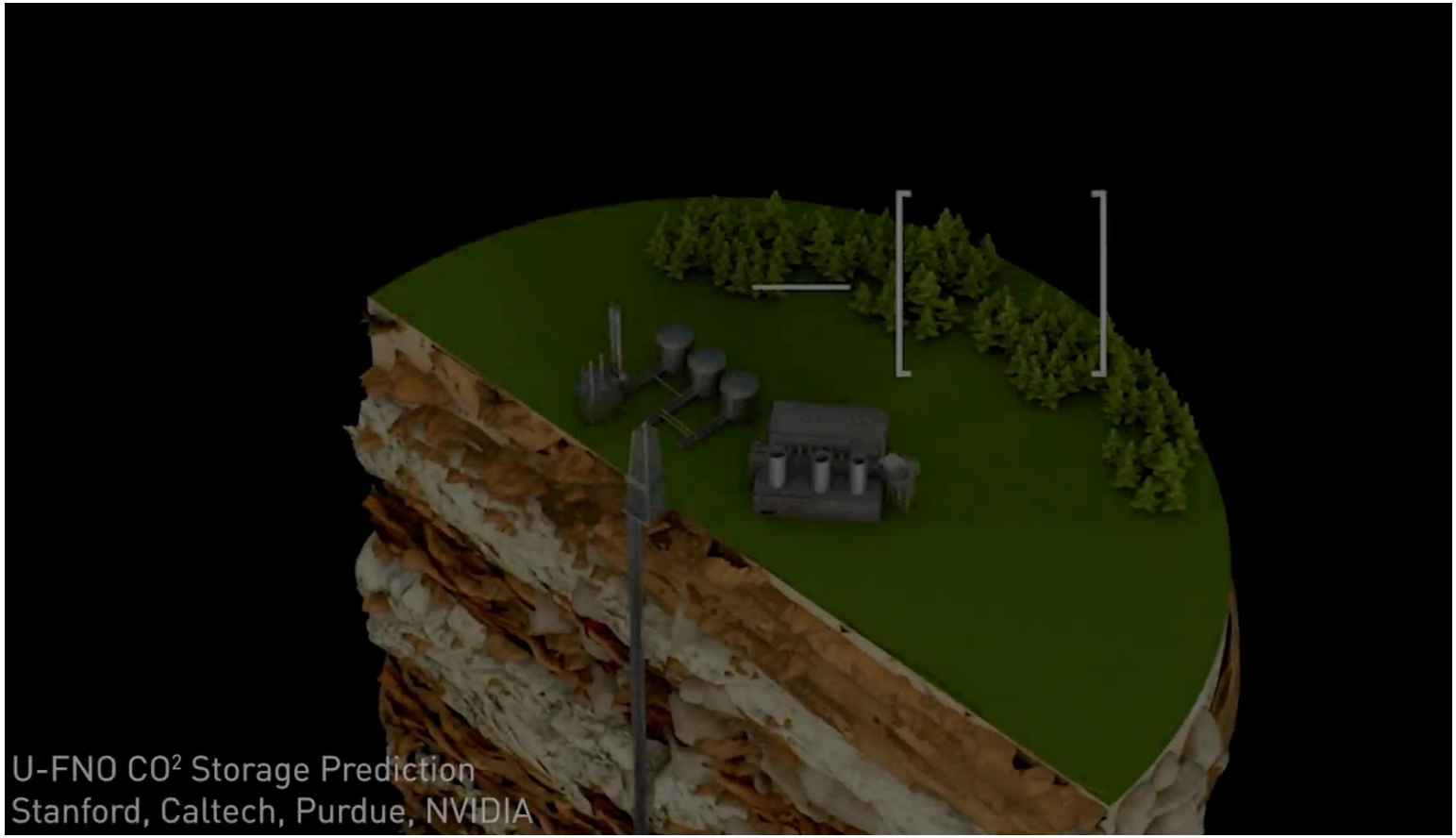
compared to a conventional Finite Difference Method solver

Alternative injection well icons



AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science



AI BOOTCAMP

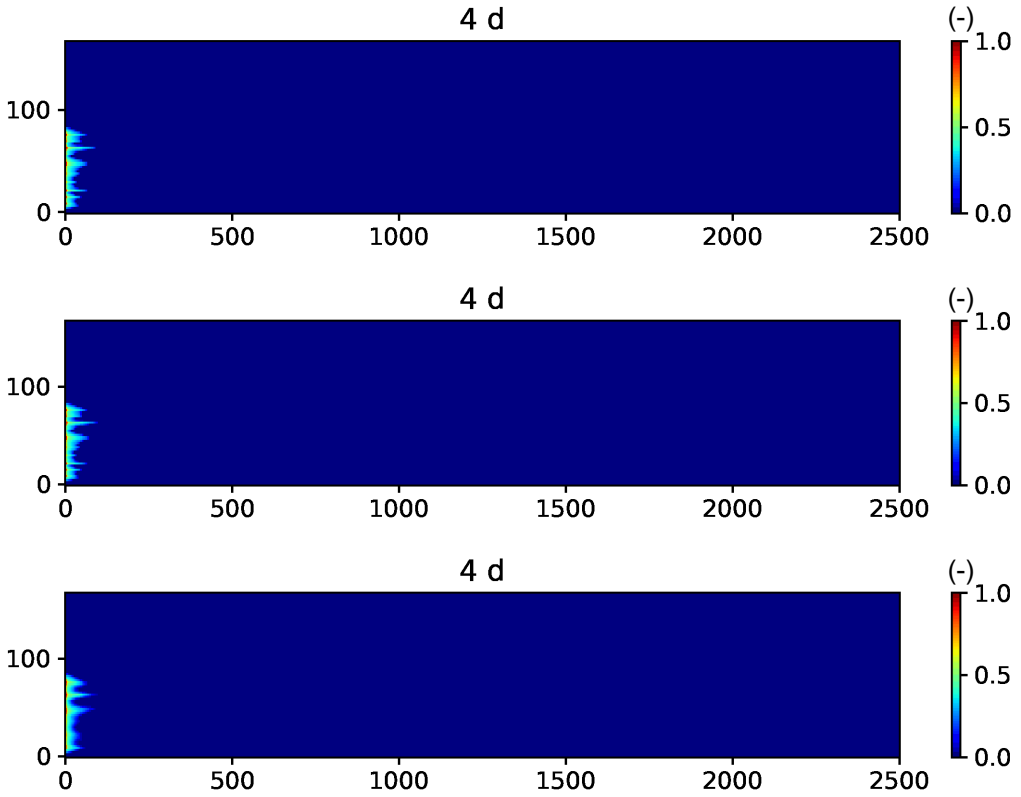
Caltech EAS Division of Engineering and Applied Science

Animation of Gas Saturation values over 30 years

Ground truth

U-FNO Prediction

CNN Prediction



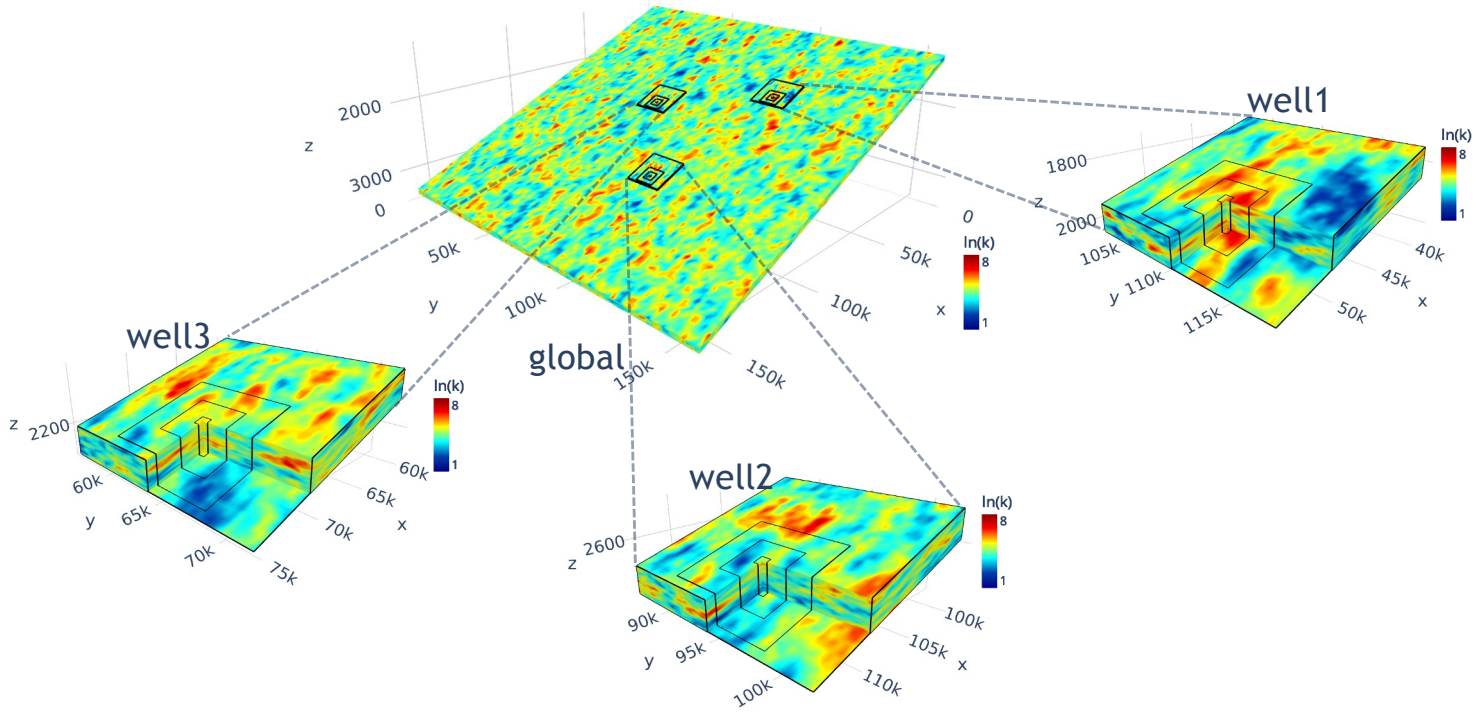
FNO is 10^5x faster than the FDM solver,
less blurry compared to CNN

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

3D+time CCS simulation with AI (FNO)

Our AI Method accelerates by 700,000 times

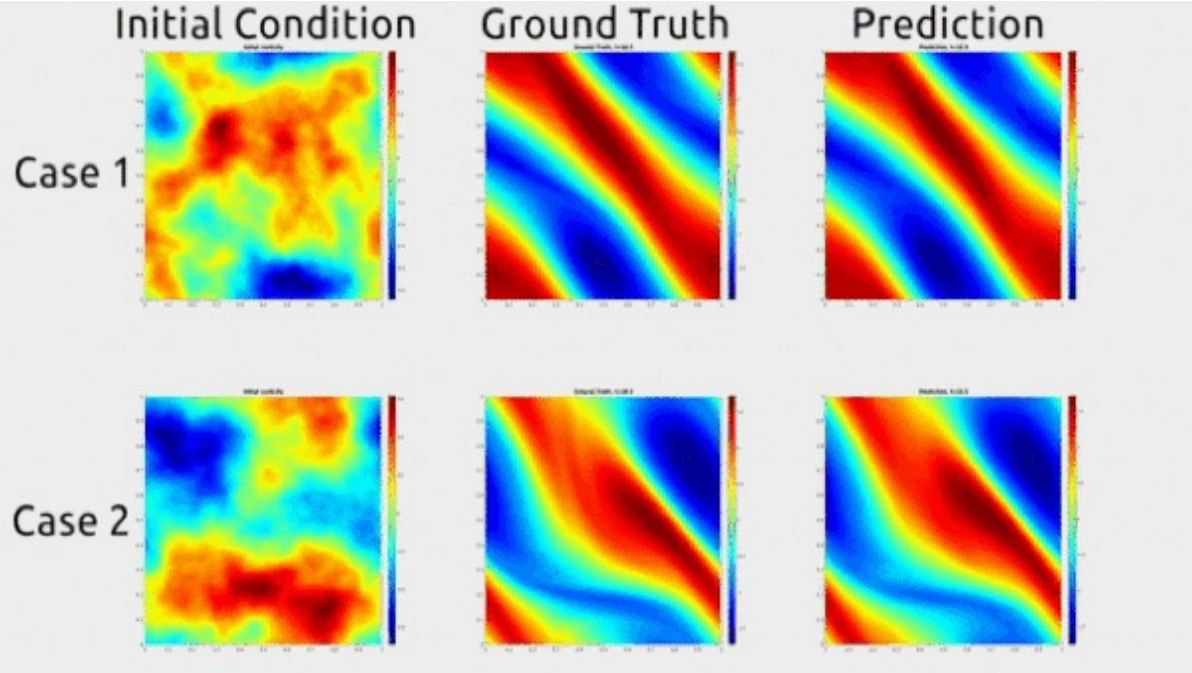


Permeability Heat Map

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

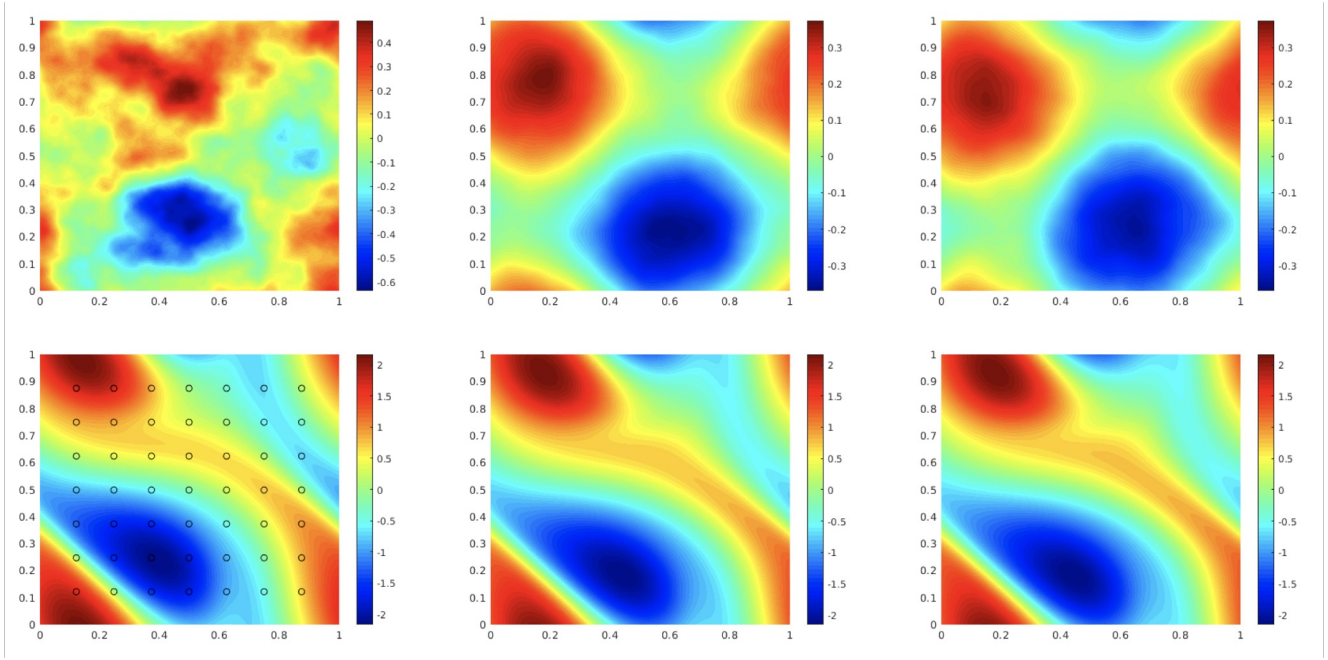
V=1e-4, zero-shot super-resolution



AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

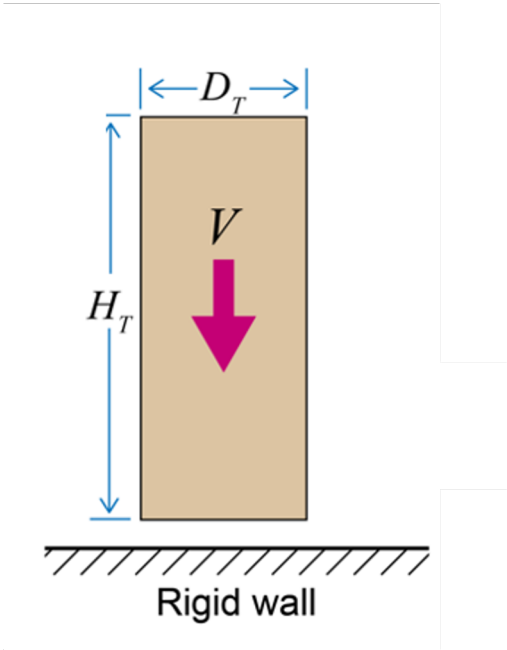
Bayesian inverse problem:



We use a MCMC method, sampling initial conditions and evaluating them with the traditional solver and Fourier operator. The Fourier operator takes **0.005s** to evaluate each initial condition, while the traditional solver takes **2.2s**.

AI BOOTCAMP

Plasticity

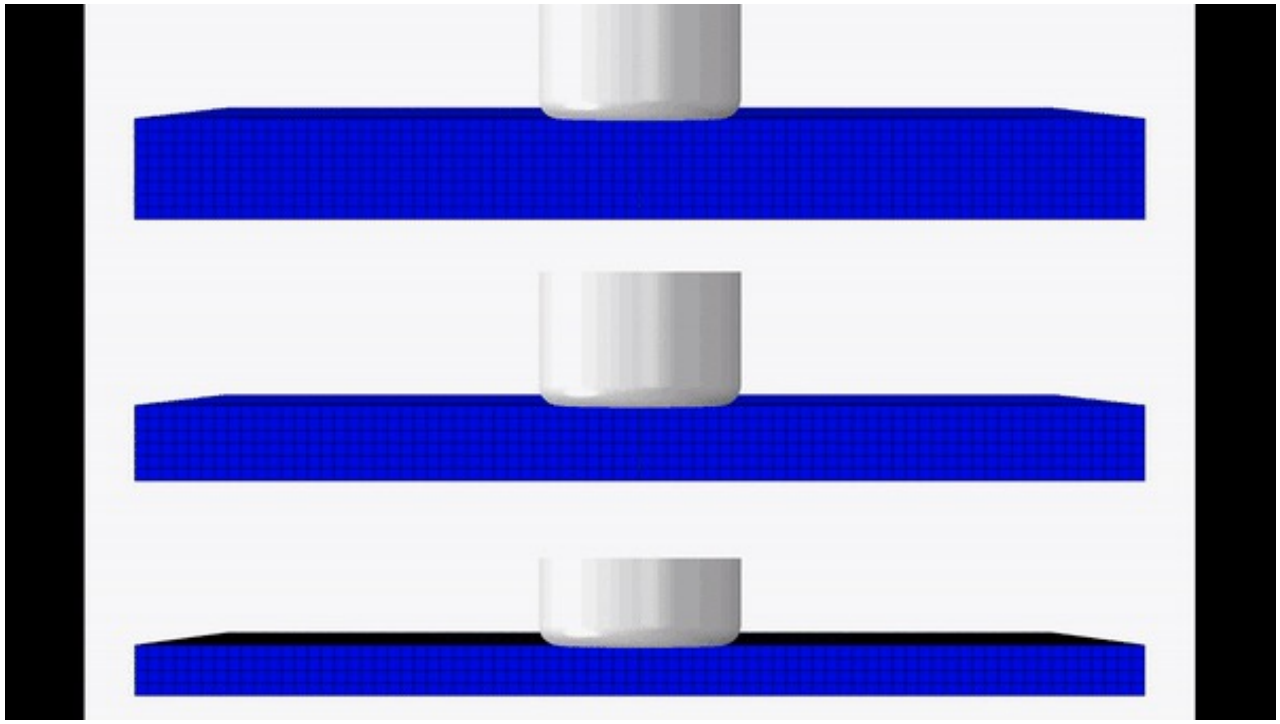


$$\nabla \cdot S^\varepsilon = \rho^\varepsilon u_{tt}^\varepsilon$$
$$K^\varepsilon = 0$$
$$u^\varepsilon(x,0) = u_0(x), \quad u_t^\varepsilon(x,0) = v_0(x), \quad \xi^\varepsilon(x,0) = \xi_0$$
$$u^\varepsilon(x,t) = u^*(x,t)$$
$$S^\varepsilon(\nabla u^\varepsilon, \xi^\varepsilon, x)n(x) = s^*(x,t)$$

Multi-scale method: use neural operator to map from strain to stress on the unit cell; update macroscale with Abaqus solver.

Caltech EAS Division of Engineering and Applied Science

Plasticity

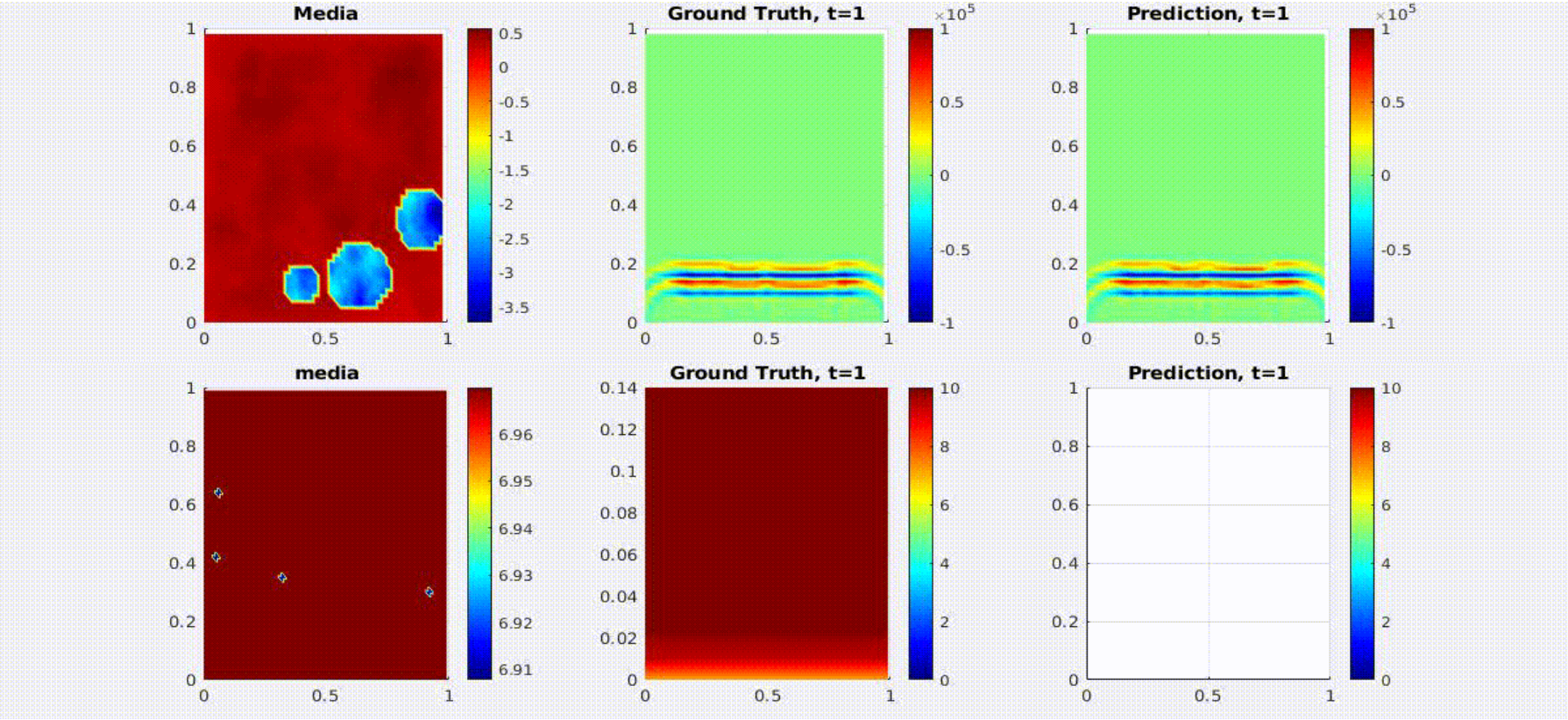


PCA-operator solves multi-scale plasticity problem (Burigede et. al.)

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

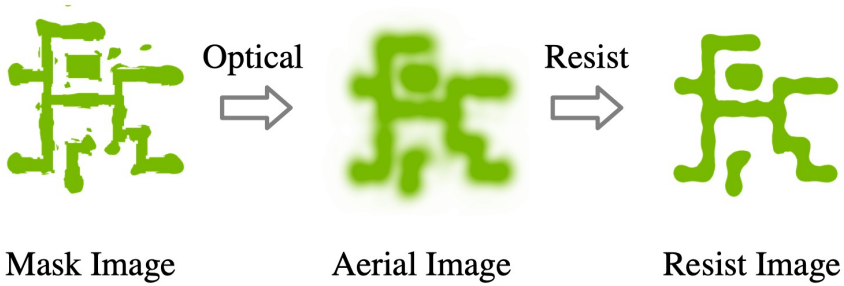
Ultrasound

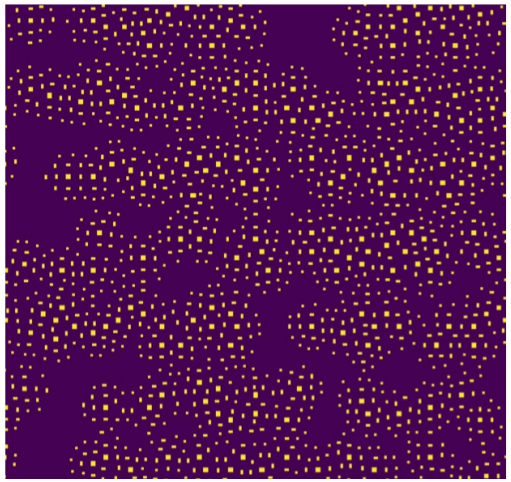


AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

lithography





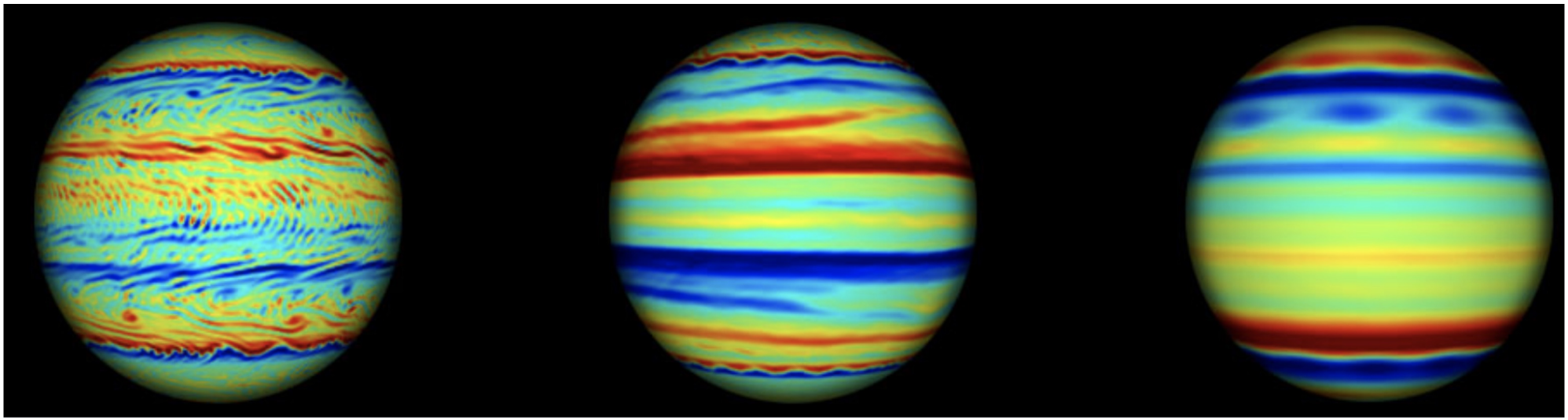
Benchmark	UNet@10epoch [25]		DAMO-DLS@10epoch [10]		DAMO-DLS@100epoch [10]		Ours@10epoch	
	mPA (%)	mIOU (%)	mPA (%)	mIOU (%)	mPA (%)	mIOU (%)	mPA (%)	mIOU (%)
ISPD-2019 (L)	99.40	98.03	98.40	97.50	99.25	98.11	**99.43**	**98.27**
ISPD-2019 (H)	99.08	97.97	-	-	-	-	**99.21**	**98.45**
ICCAD-2013 (L)	97.30	95.38	96.24	95.15	98.94	96.97	**98.98**	**97.79**
ICCAD-2013 (H)	95.16	93.04	-	-	-	-	**99.12**	**97.77**

Haoyu Yang et. al.

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

5. Extensions



AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

PINO: physics-informed neural operator

Zongyi Li*, Hongkai Zheng*, Nikola Kovachki, David Jin, Haoxuan Chen, Burigede Liu,
Kamyar Azizzadenesheli, Anima Anandkumar

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

PINO: Physics-informed neural operator

Data loss: compare prediction and ground-truth solution

Equation loss: plug prediction in PDE and compute residual

$$-\nabla \cdot (a(x)\nabla u(x)) = f(x), \quad x \in D$$
$$u(x) = 0, \quad x \in \partial D$$

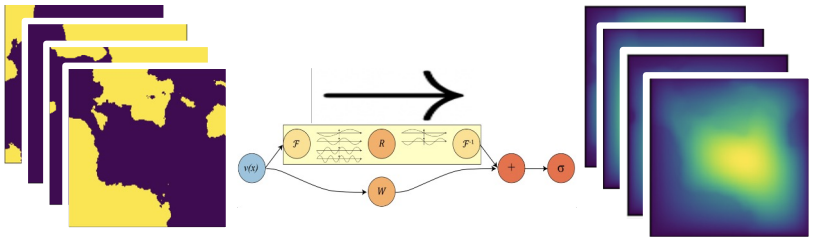
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

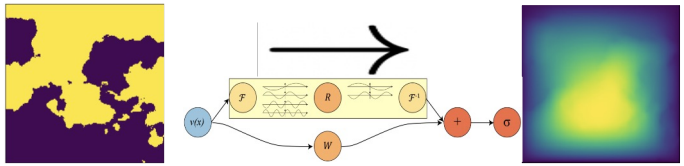
PINO: PHYSICS-INFORMED NEURAL OPERATOR

Neural operators use only the data. If the explicit form of the PDE is known, we can combine neural operators with the PINNs setting.

1. Pre-train
- operator-learning setting
- a family of PDE
- use the data loss (and equation loss)



2. test-time optimize
- solver setting (PINN)
- a specific instance
- use the equation loss

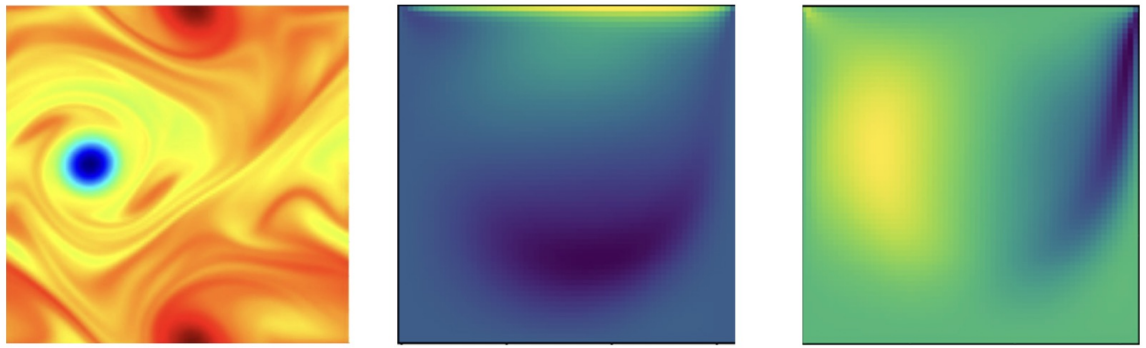


AI BOOTCAMP

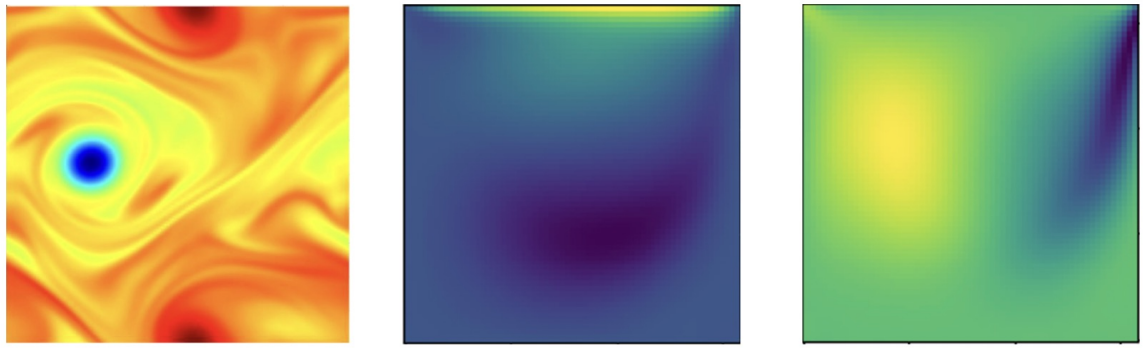
Physics-informed neural operator

Standard cases (Burgers, Darcy, KF, Cavity): no need to use any data

- Pretrain improves both the convergence speed and final error.
- ground truth



prediction

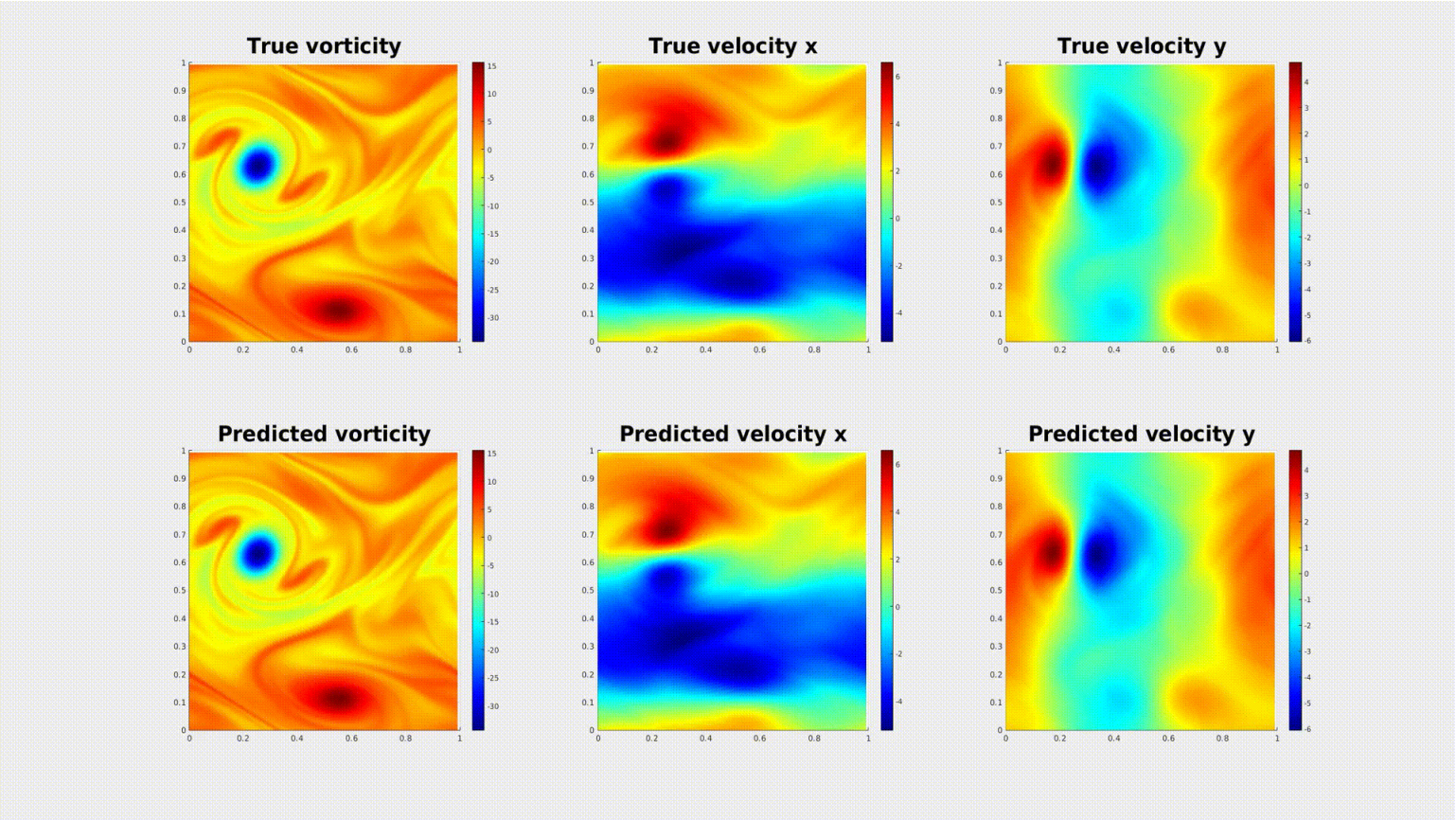


(a) KF: vorticity

(b) Cavity: velocity in x, y

Caltech EAS Division of Engineering and Applied Science

- PINO gets 2% error on Re500 [2π x 2π x 1s]
- Easily generalize from one Re to another



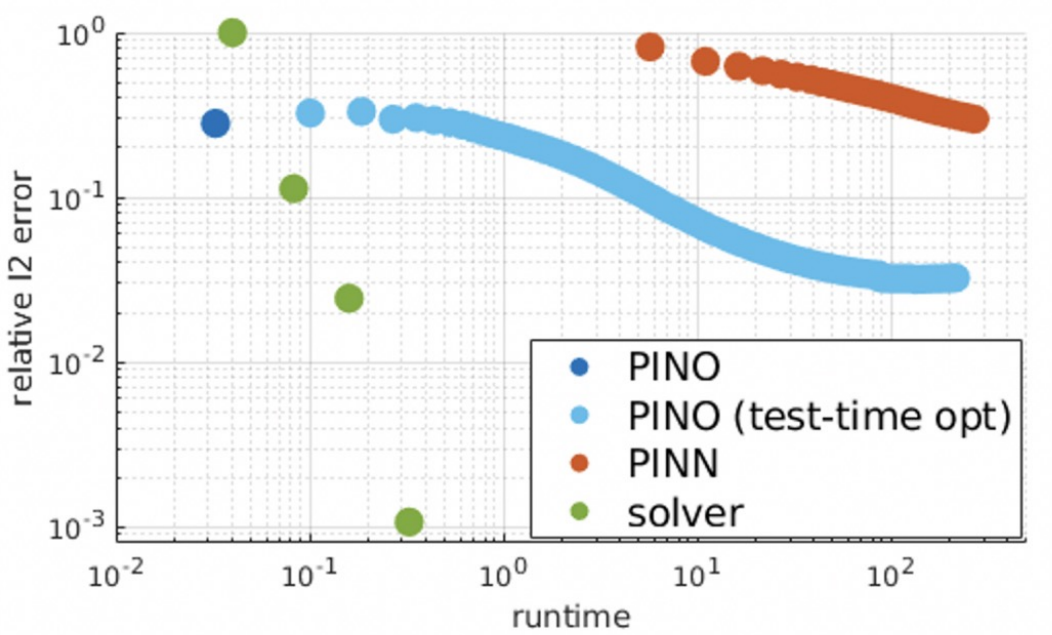
Relative error: PINO: 0.9%, PINN:18.7%

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Physics-informed neural operator

- Pretrain improves both the convergence speed and final error.
- More robust to hyperparameters



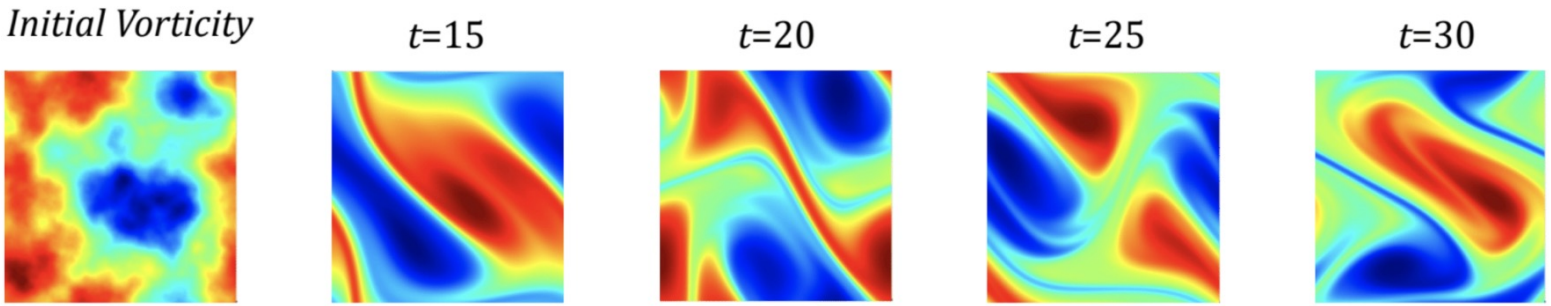
PINN: Raissi, Maziar; Perdikaris, Paris; Karniadakis, George Em

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Physics-informed neural operator

If data is available: PINO can solve very challenging scenarios such as long-temporal transient flow T=50, which is intractable to PINNs.

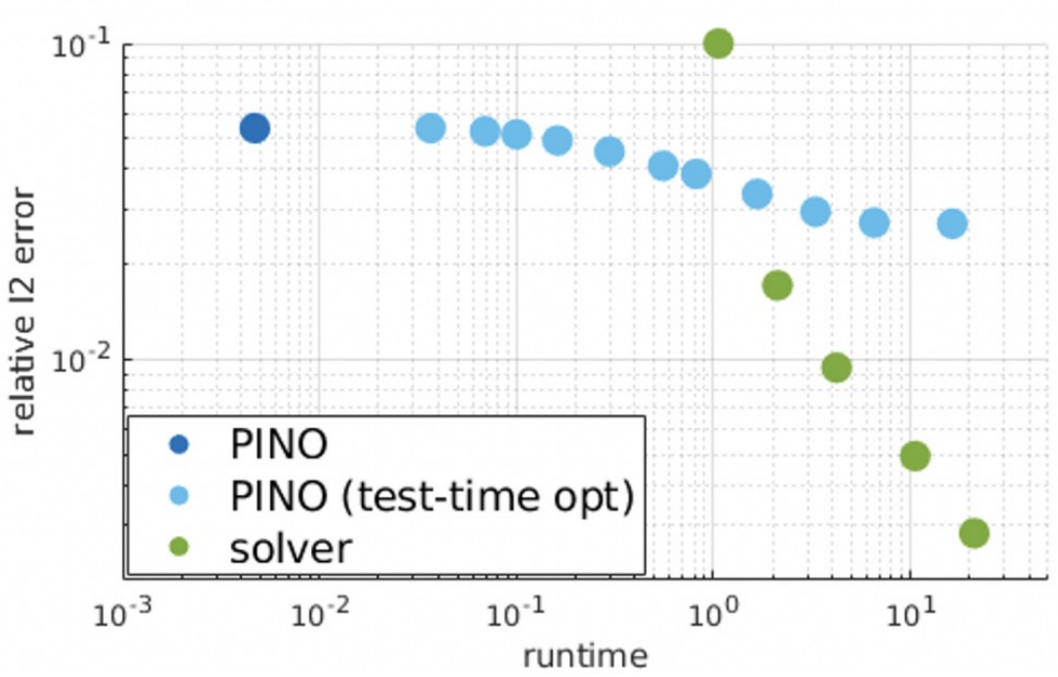


AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

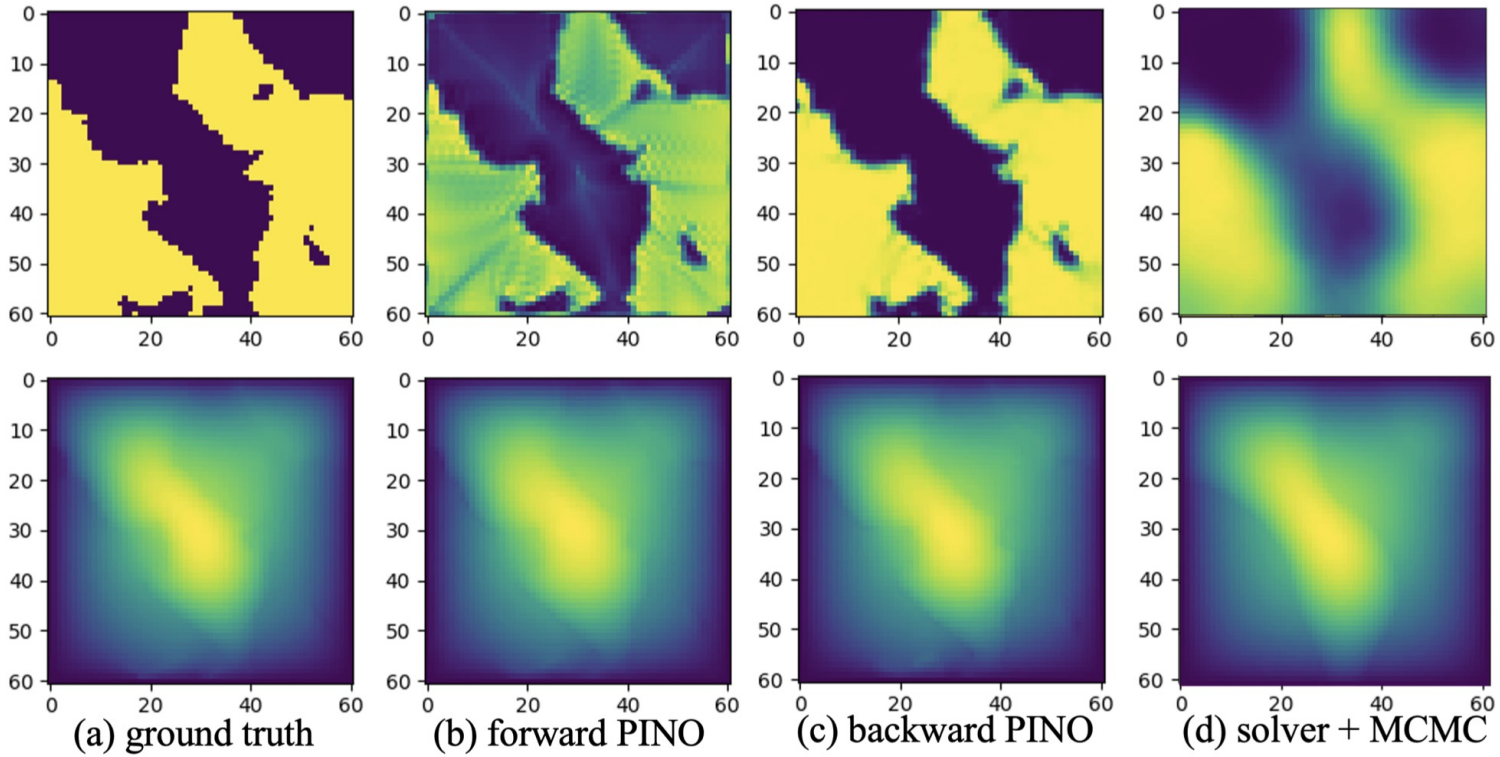
Physics-informed neural operator

Long-temporal transient flow: preserve the 400x speedup compared to the pseudo-spectral solver.



AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

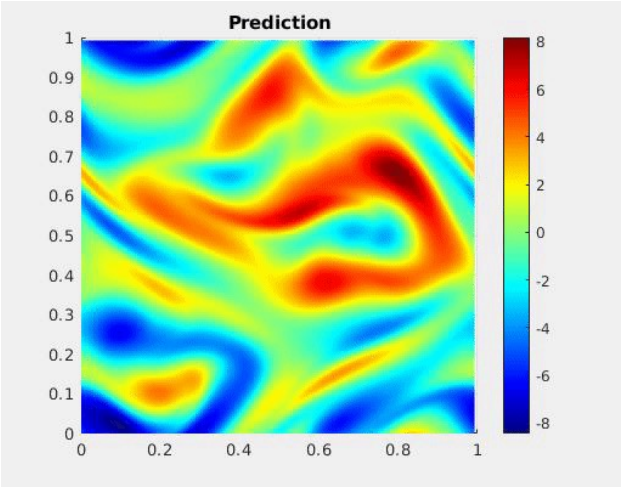


Backward PINO accurately solves inverse problem. 3000x speedup

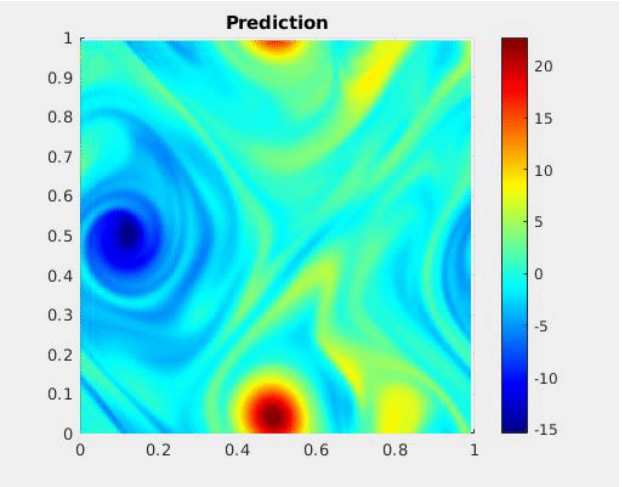
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Transfer Learning



Re100



Re500

Pre-train on Re100, fine-tune on Re500. Converge 3x faster.

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Geometric-FNO

Zongyi Li, Zhengyu Huang, Burigede Liu, Anima Anandkumar

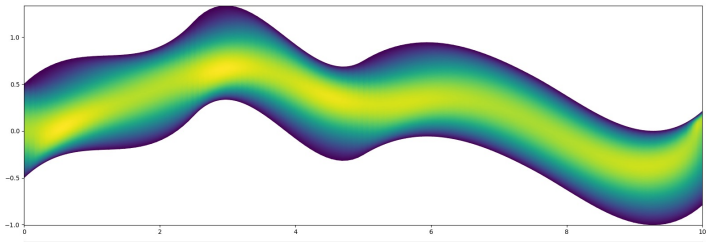
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

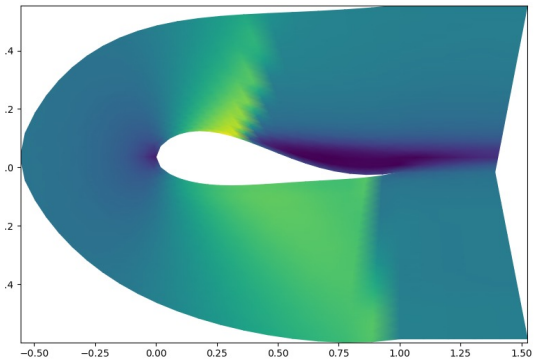
Geometric FNO

Previous FNO implementation relies on FFT. It can be only applied to rectangular domains with a uniform grid. We want to extend FNO to different geometries.

Further, we want to learn solution operators mapping from the shapes to the solution, and solve the inverse design problems.



Pipe



Airfoils

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Geometric FNO

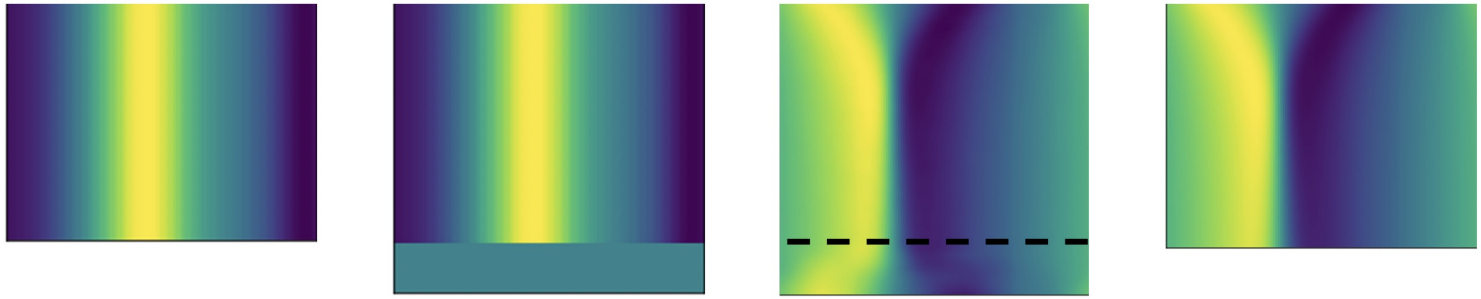
- Any boundary -> embed (Fourier continuation)
- Any mesh -> learned interpolation /DFT
- Any shape -> deform (adaptive/moving mesh)
- Any topology -> decompose (pants/handler decomposition)

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

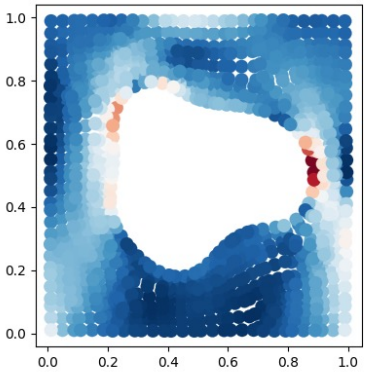
boundary

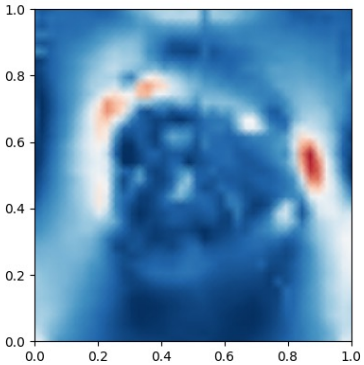
For non-periodic boundary, we can embed the domain into a larger domain with periodic condition



(a) Input (b) padded input (c) padded output (d) target output

Similar, for any obstacles and holes. we can fill in zeros.

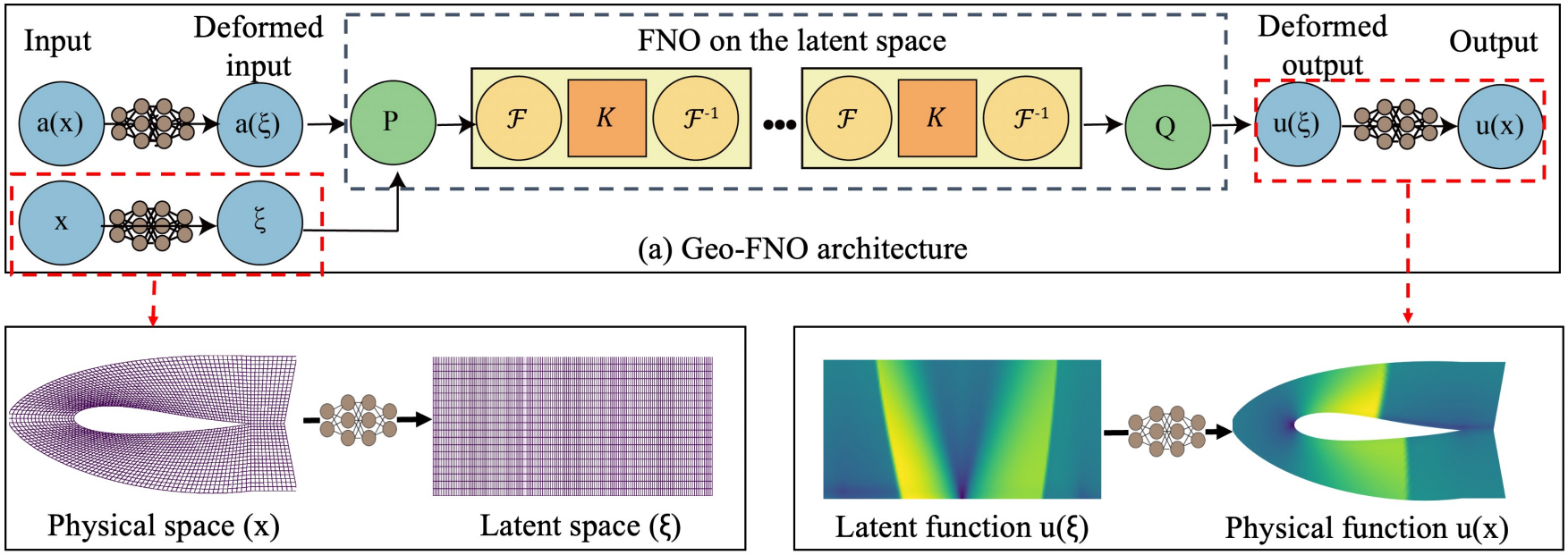




AI BOOTCAMP

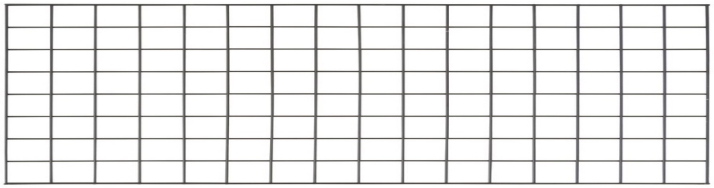
Geo-FNO

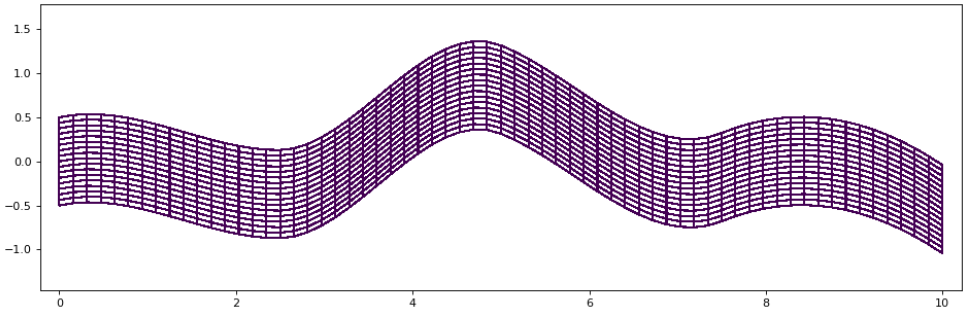
Idea: deform the irregular physical space to a uniform latent space, so the FFT can be applied in the latent space.



Caltech EAS Division of Engineering and Applied Science

Shape

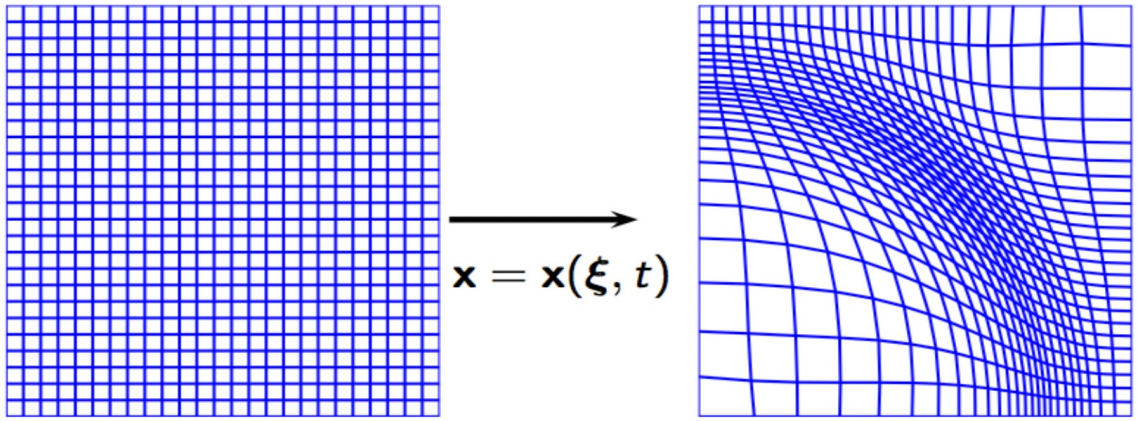




Given any input domain, we want to find a deformation (diffeomorphism)
To convert the domain into a regular one.

AI BOOTCAMP

Adaptive meshes



FFT computation mesh

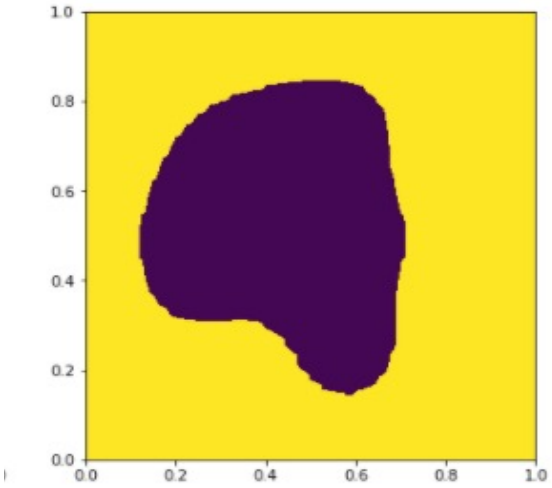
Adaptive mesh

Such deformation can be also used to construct adaptive meshes for multiscale structures

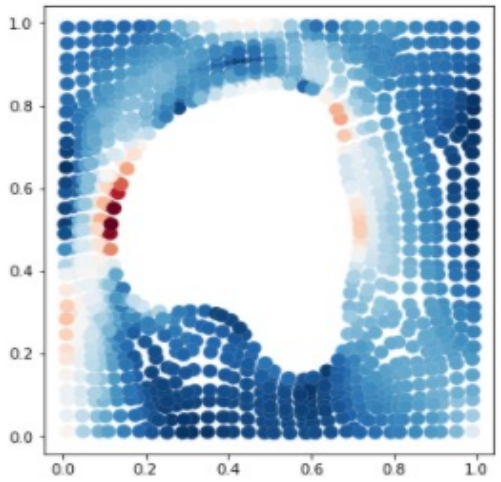
Caltech EAS Division of Engineering and Applied Science

Examples: elastic equation

A constant forcing is applied from the left. Given the shape, we want to predict the stress.
Unstructured mesh + irregular shape.



Input: shape



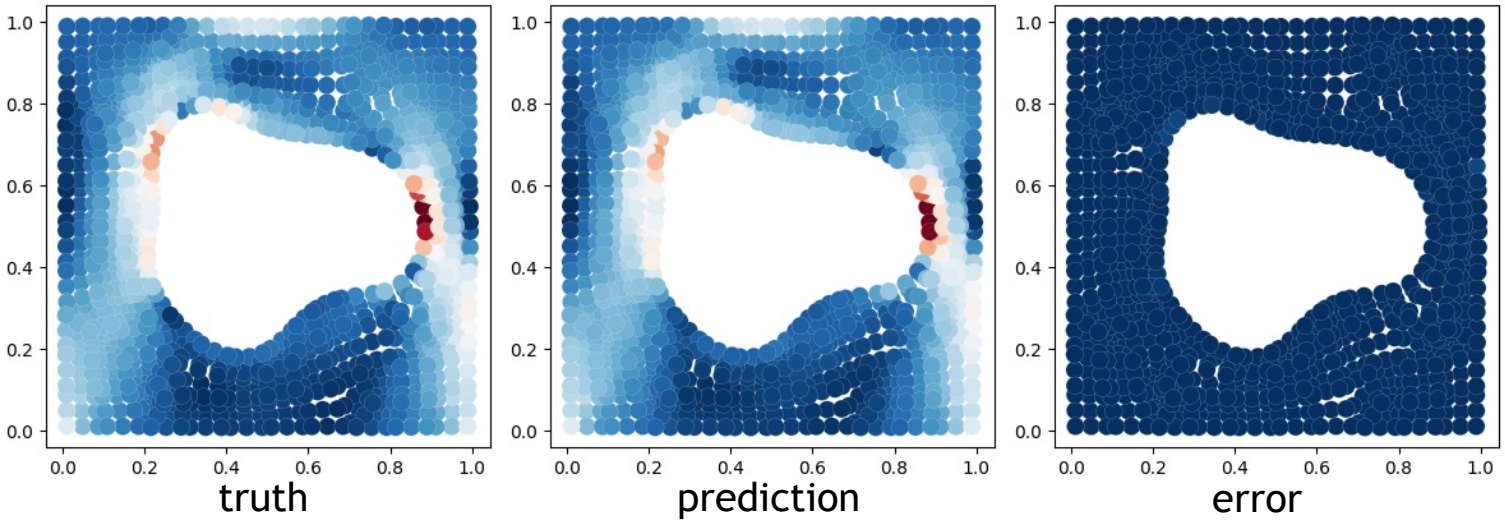
Output: stress

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Examples: elastic equation

Using dis

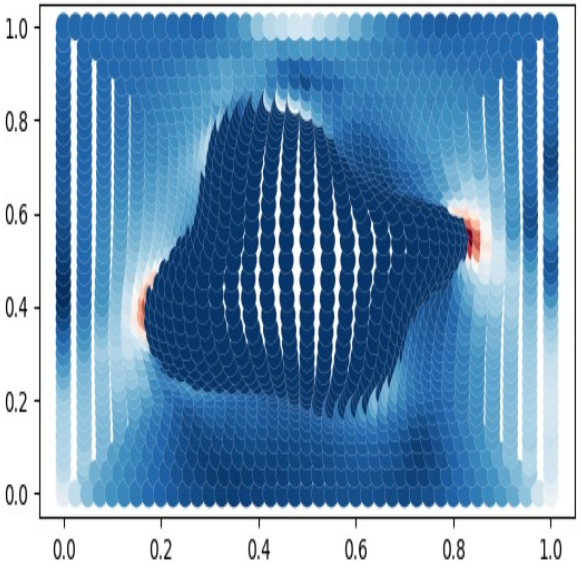


AI BOOTCAMP

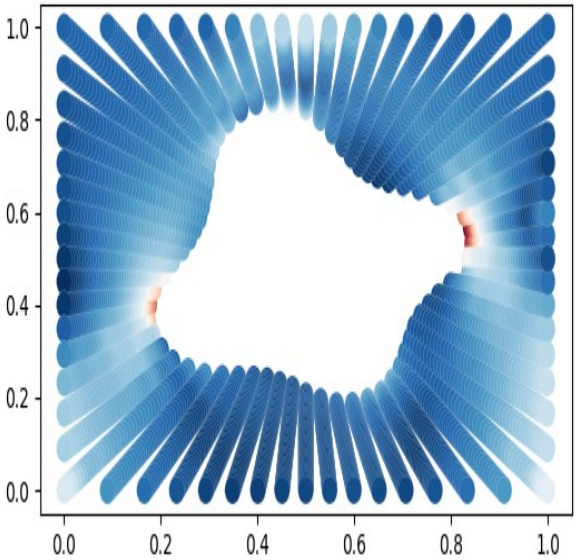
Caltech EAS Division of Engineering and Applied Science

Examples: elastic equation

Using adaptive meshes further improves the performance.



Adaptive mesh: (x,y)



Adaptive mesh: (θ,r)

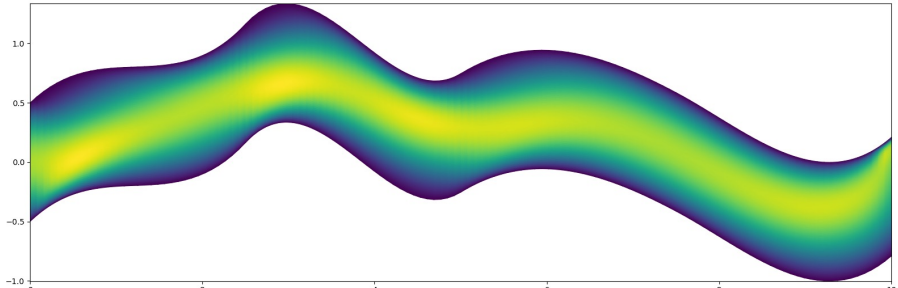
	Train Raw	Test Raw	Test DFT
Uniform	7%	10%	11%
X-Y deform	9%	10%	9%
Theta-R	6%	6%	~6%

AI BOOTCAMP

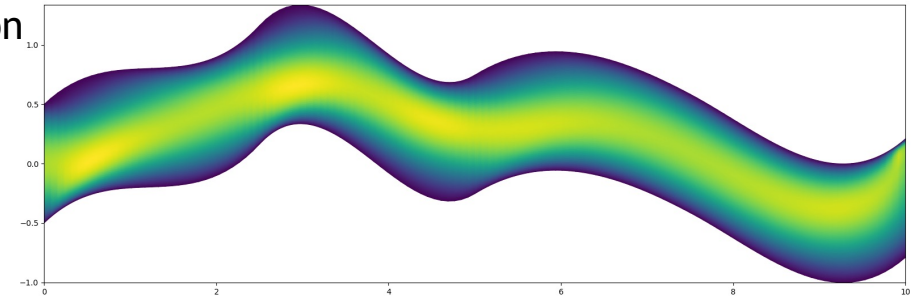
Caltech EAS Division of Engineering and Applied Science

Examples: Channel flow (pipe)

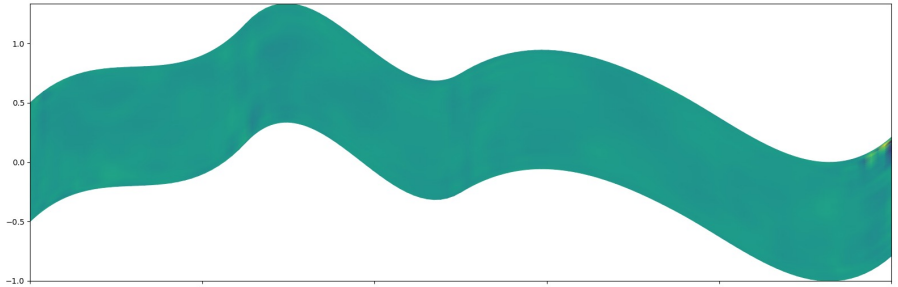
truth



prediction



error



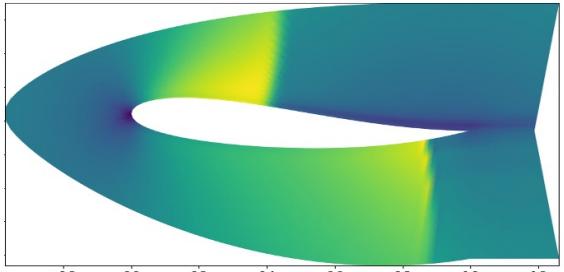
Error = 0.5%

AI BOOTCAMP

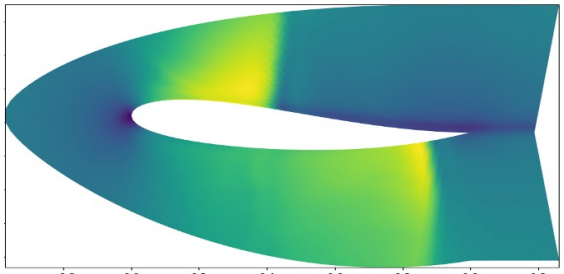
Caltech EAS Division of Engineering and Applied Science

Examples: Airfoils

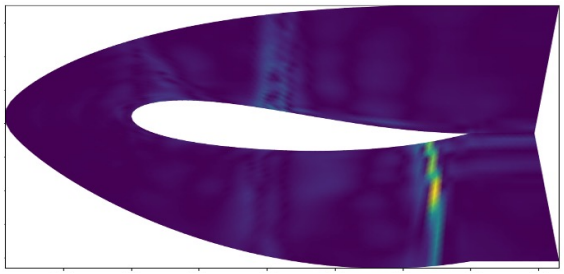
truth

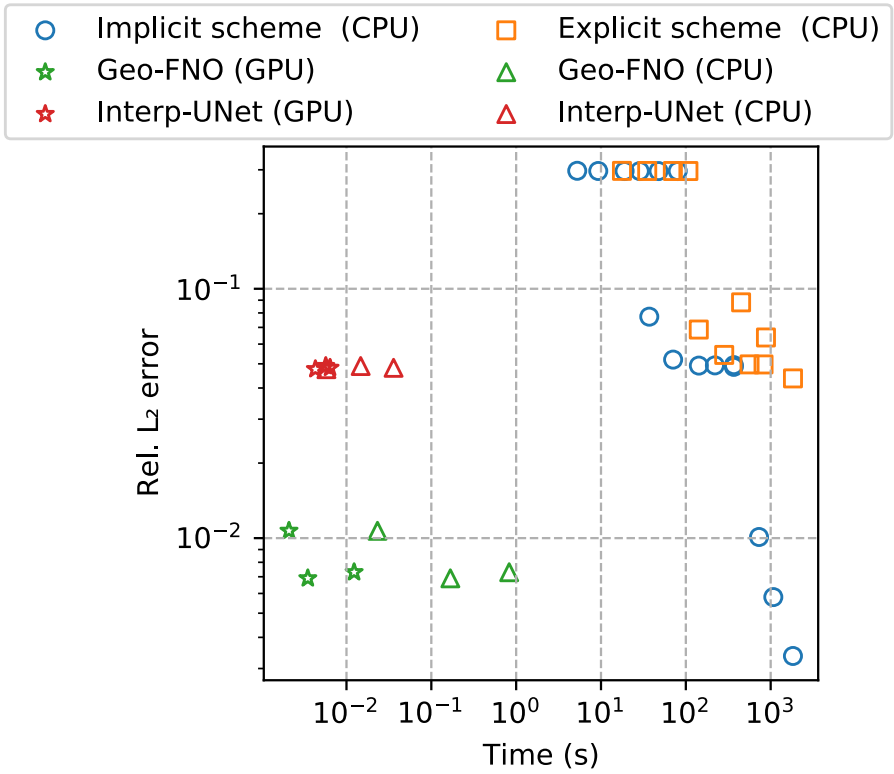


prediction



error





Error = 1.3%

10^3 speedup

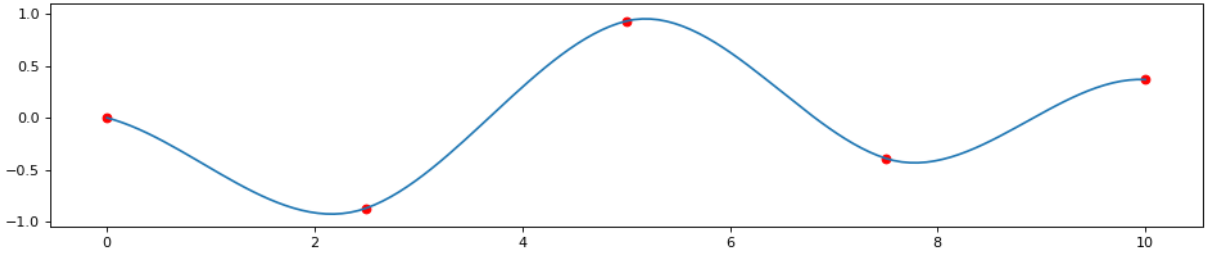
AI BOOTCAMP

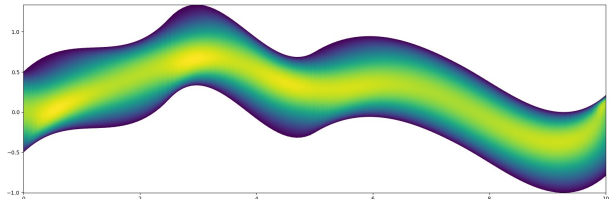
Caltech EAS Division of Engineering and Applied Science

Examples: inverse design

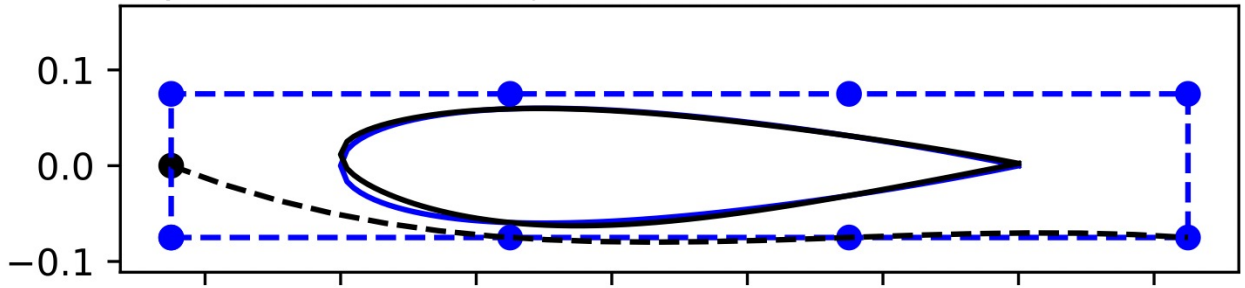
Optimize the spline nodes to achieve the design objectives.

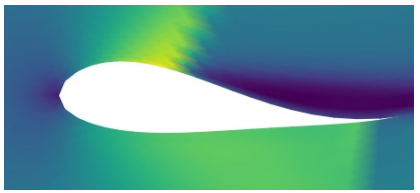
Pipe: optimize the upper (and lower) boundary





Airfoil: optimize the bounding box





AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

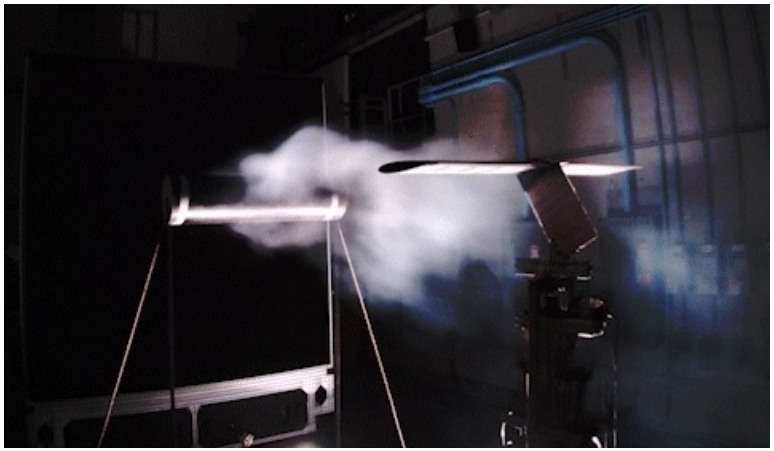
PDE-observer

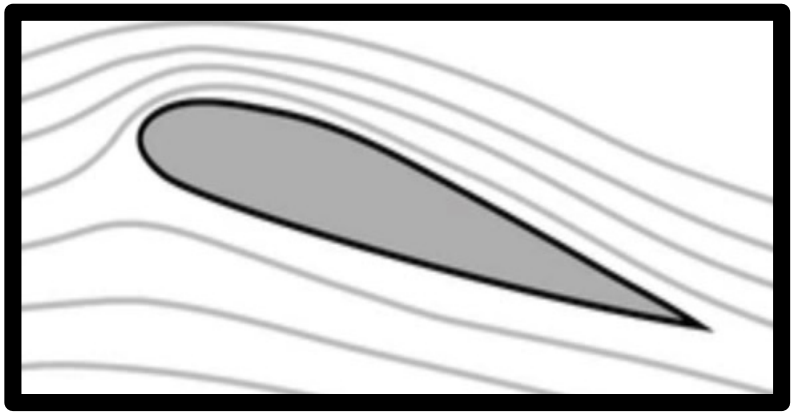
Yuanyuan Shi, Zongyi Li, Huan Yu, Drew Steeves,
Anima Anandkumar, and Miroslav Krstic

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

MOTIVATING PROBLEM IN ROBOTICS





Given sensors placed on the body of airfoil/drone (boundary),
Can we estimate the velocity field (interior) for more accurate control?

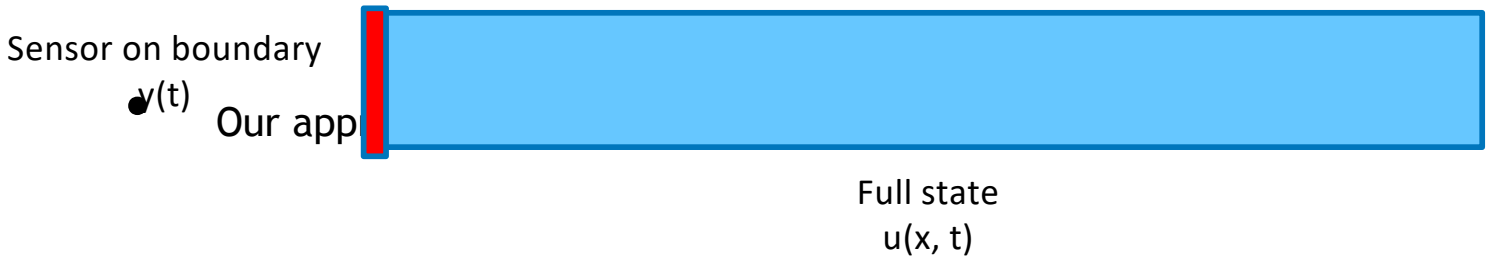
Yuanyuan Shi, Zongyi Li, Huan Yu, Drew Steeves, Anima Anandkumar, and Miroslav Krstic

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

STATE ESTIMATION THROUGH PDE OBSERVER

- Given the partial observation, estimate the state.
- PDE for state estimation: Back-stepping PDE observer.
- The estimate converges to the truth (exponentially)
- Slow and not real-time.

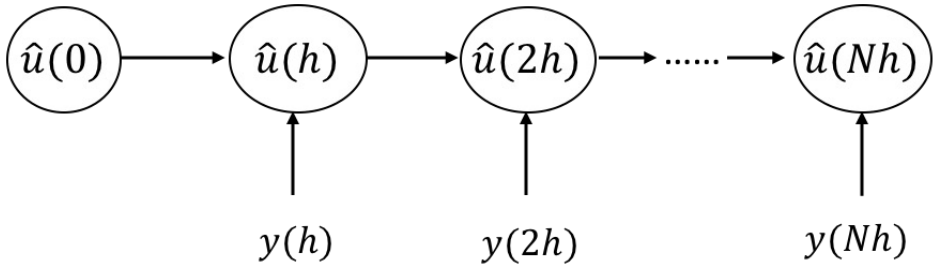


AI BOOTCAMP

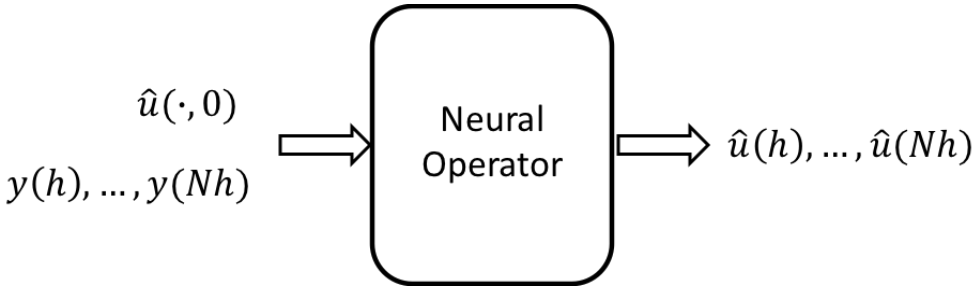
Caltech EAS Division of Engineering and Applied Science

Neural observer with FNO

Recurrent observer: make the estimation at each step recursively



Feedforward observer: make the estimate over a time interval each step.

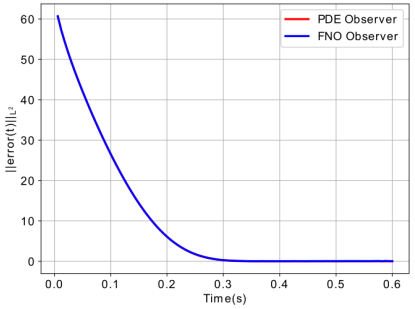


Yuanyuan Shi, Zongyi Li, Huan Yu, Drew Steeves, Anima Anandkumar, and Miroslav Krstic

AI BOOTCAMP

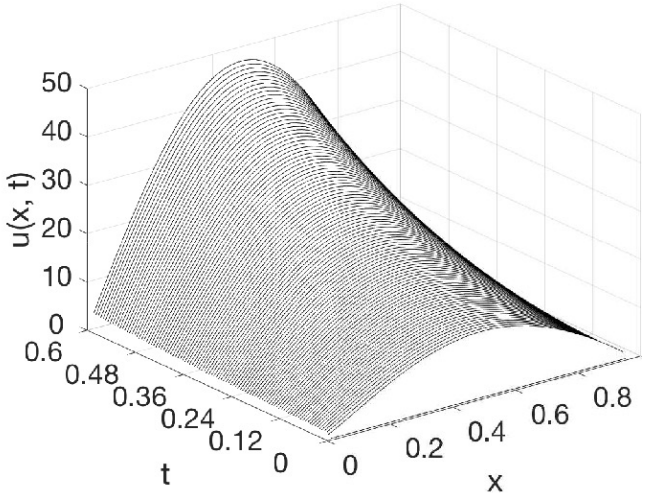
Caltech EAS Division of Engineering and Applied Science

REACTION-DIFFUSION PDE PRESCRIBED-TIME OBSERVER

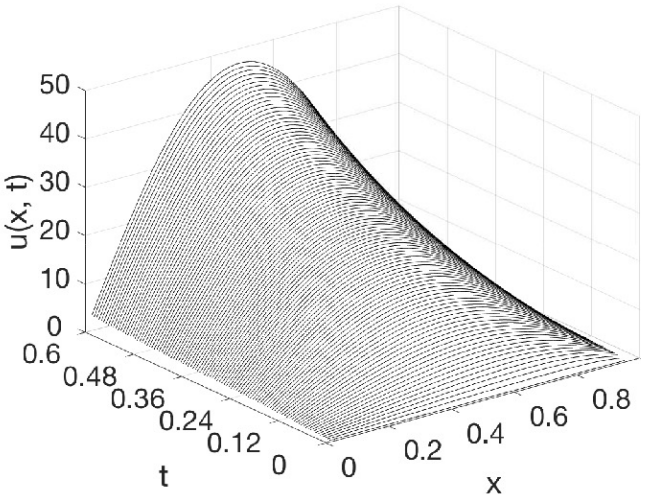


Same convergence rate

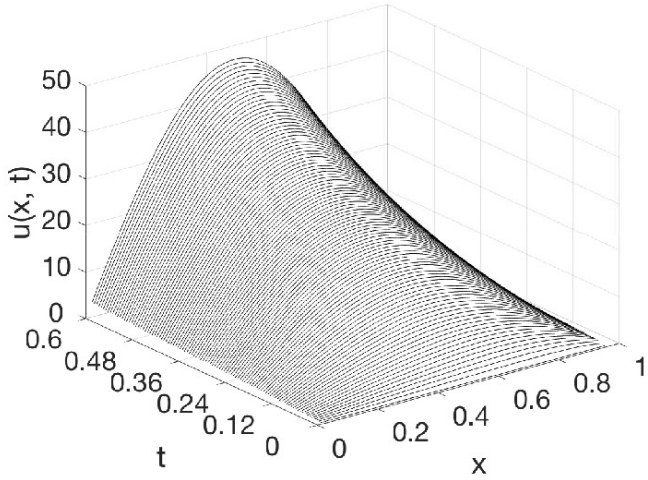
Chemical tubular reactor system:
Learn a time-dependent, fast-converging observer.



True system



Conventional PT Observer



FNO Neural Observer

2000x Speed up

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

FNO-transformer

John Guibas , Morteza Mardani , Zongyi Li ,
Andrew Tao, Anima Aanandkumar, Bryan Catanzaro

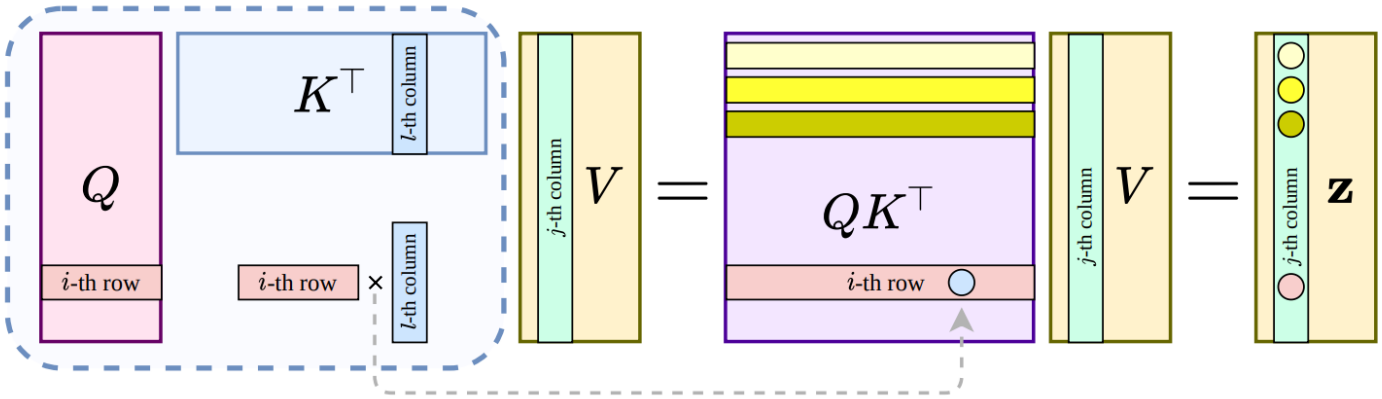
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Fourier-based Transformer

Neural operator can be viewed as a continuous generalization of Transformers

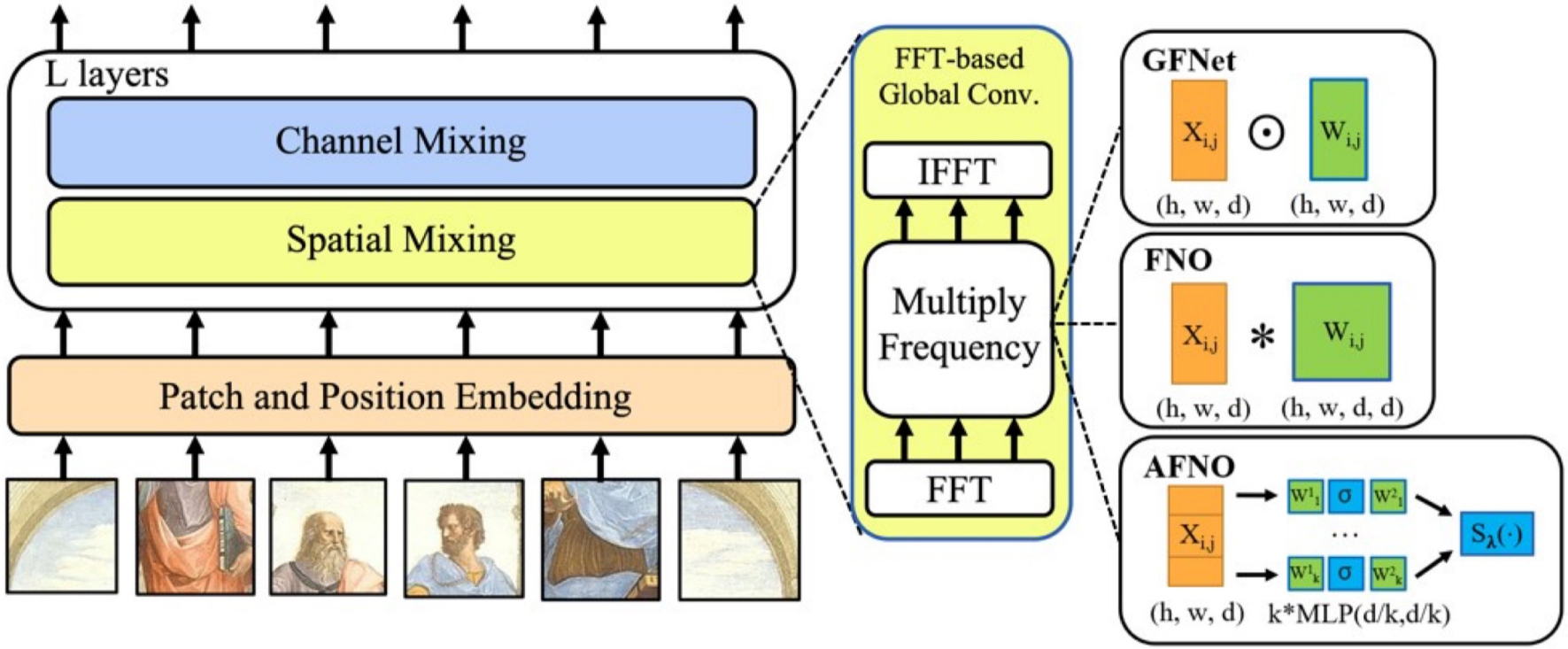
- Attention mechanism is a kernel-integration.
- Query-key can be viewed a a low-rank decomposition of the kernel.
- Replace the kernel by Fourier transform.



Shuhao Cao. Choose a Transformer: Fourier or Galerkin

AI BOOTCAMP

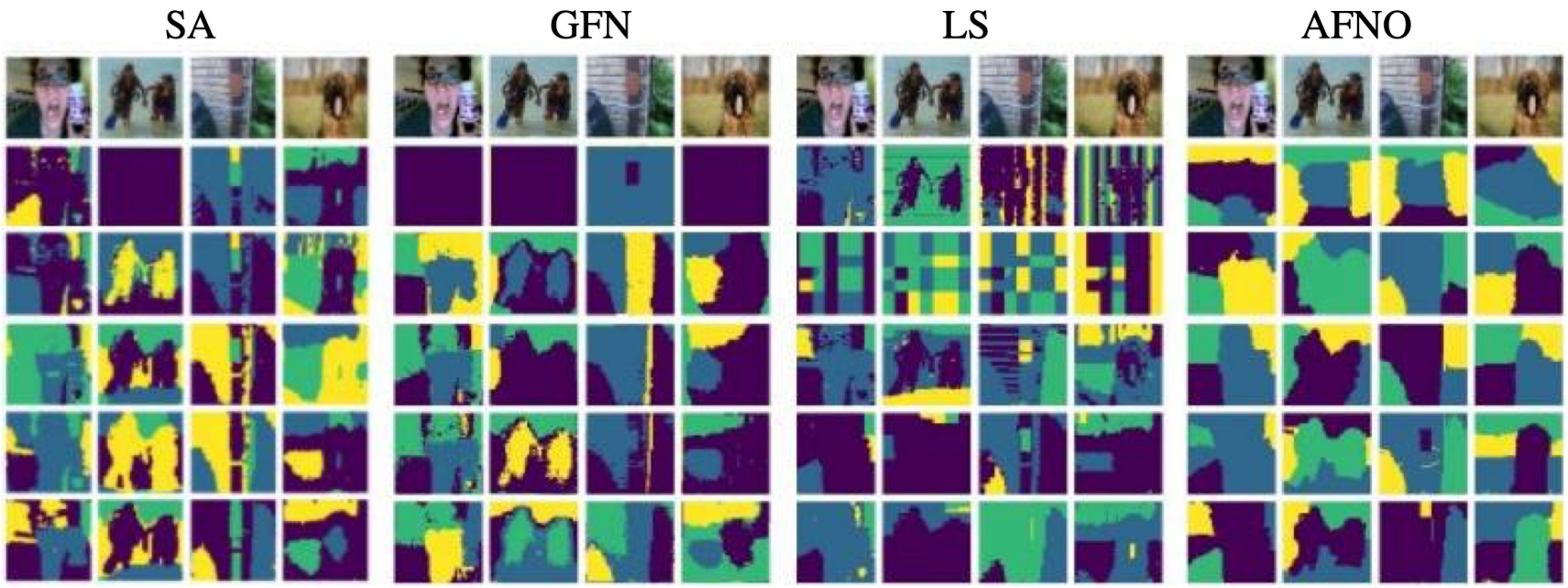
Fourier-based Transformers



Lee-Thorp et. al. FNet: Mixing Tokens with Fourier Transforms
Rao et. al. GFNet: Global Filter Networks for Image Classification

Caltech EAS Division of Engineering and Applied Science

Backbone	Mixer	Params	GFLOPs	Latency(sec)	SSIM	PSNR(dB)
ViT-B/4	Self-Attention	87M	357.2	1.2	**0.931**	**27.06**
ViT-B/4	LS	87M	274.2	1.4	0.920	26.18
ViT-B/4	GFN	87M	177.8	0.7	0.928	26.76
ViT-B/4	AFNO (ours)	87M	257.2	0.8	**0.931**	27.05



AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Chaotic system

Miguel Liu-Schiaffini, Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli,
Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, Anima Anandkumar

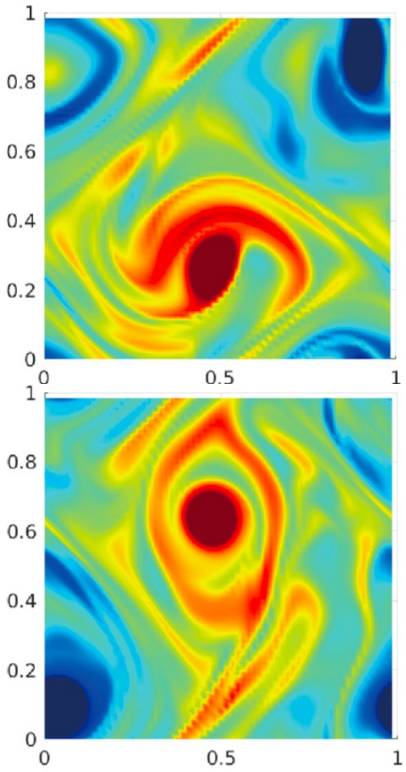
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

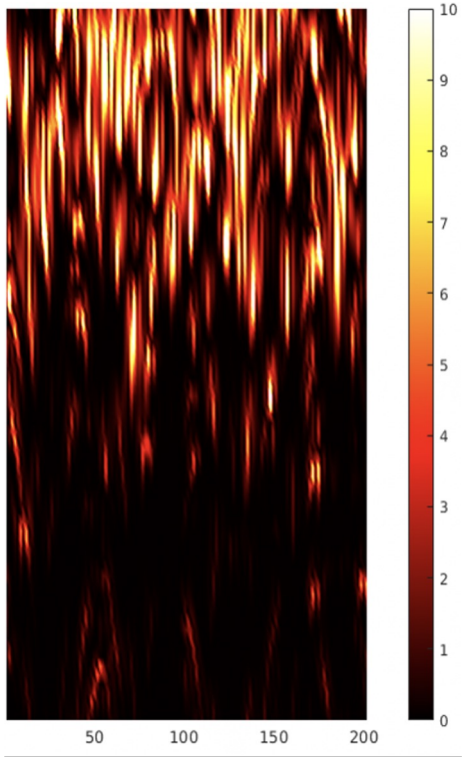
Chaotic system

Chaotic systems are intrinsically unstable. Smaller errors will accumulate and make the simulation diverge from the truth.

Can we predict long-time trajectories that, while eventually diverging from the truth, still preserve the same orbit (**attractor**) of the system and its **statistical properties**?



Navier-Stokes



Kuramoto-Sivashinsky

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Semigroup and Markov neural operator

Use Markov neural operator to model the local evolution.

$$u(t) = S_t u(0)$$

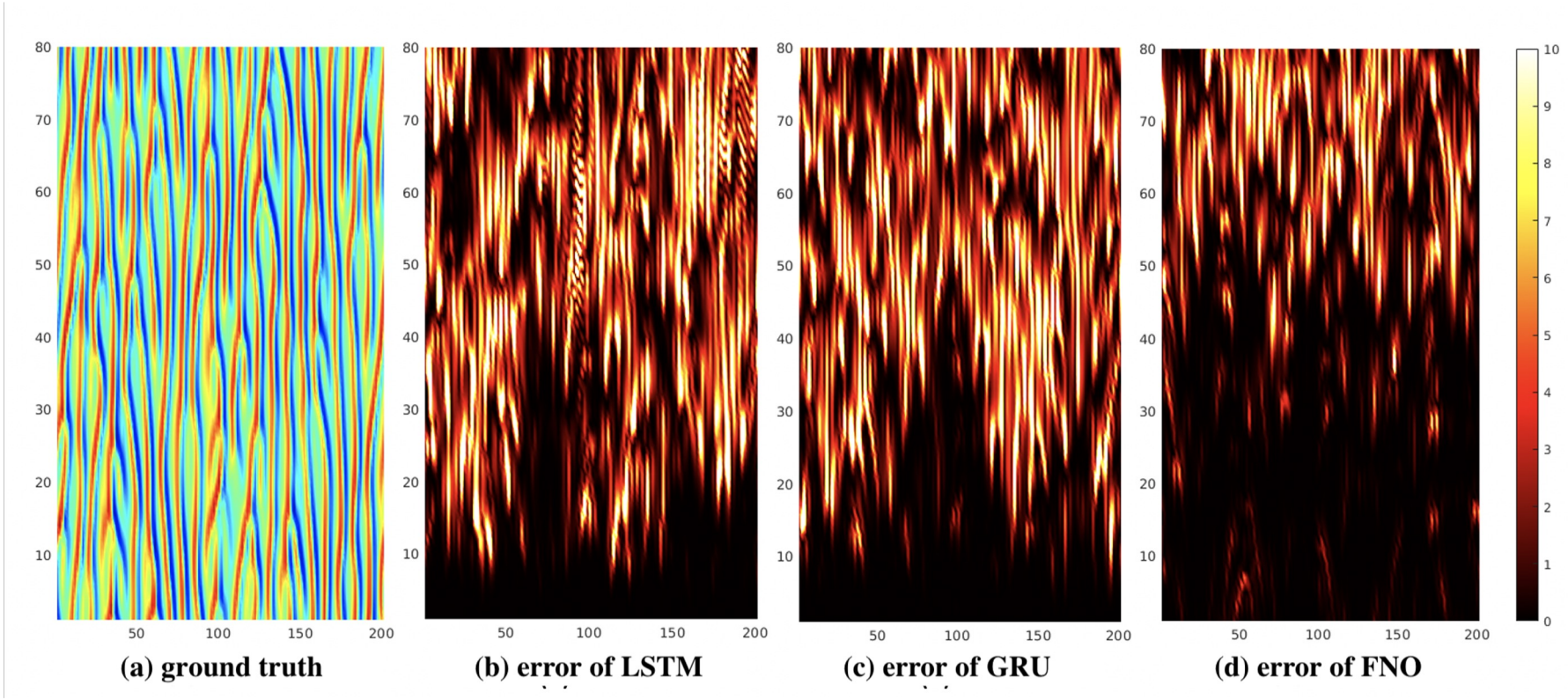
Compose the operator as a semigroup

$$u(n \cdot h) \approx \hat{S}_h^n(u_0) := \underbrace{(\hat{S}_h \circ \cdots \circ \hat{S}_h)}_{n \text{ times}}(u_0)$$

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

KS equation

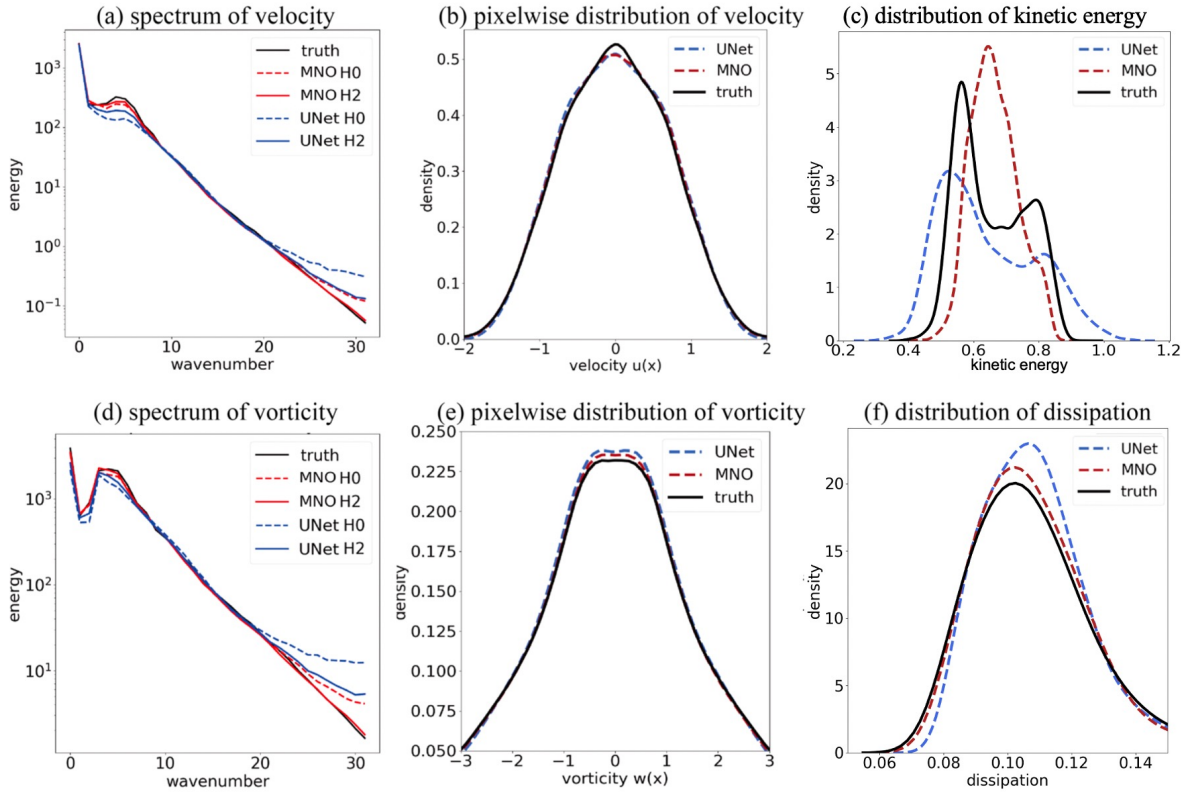


FNO captures the trajectory of chaotic systems for a longer period compared to RNNs.

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

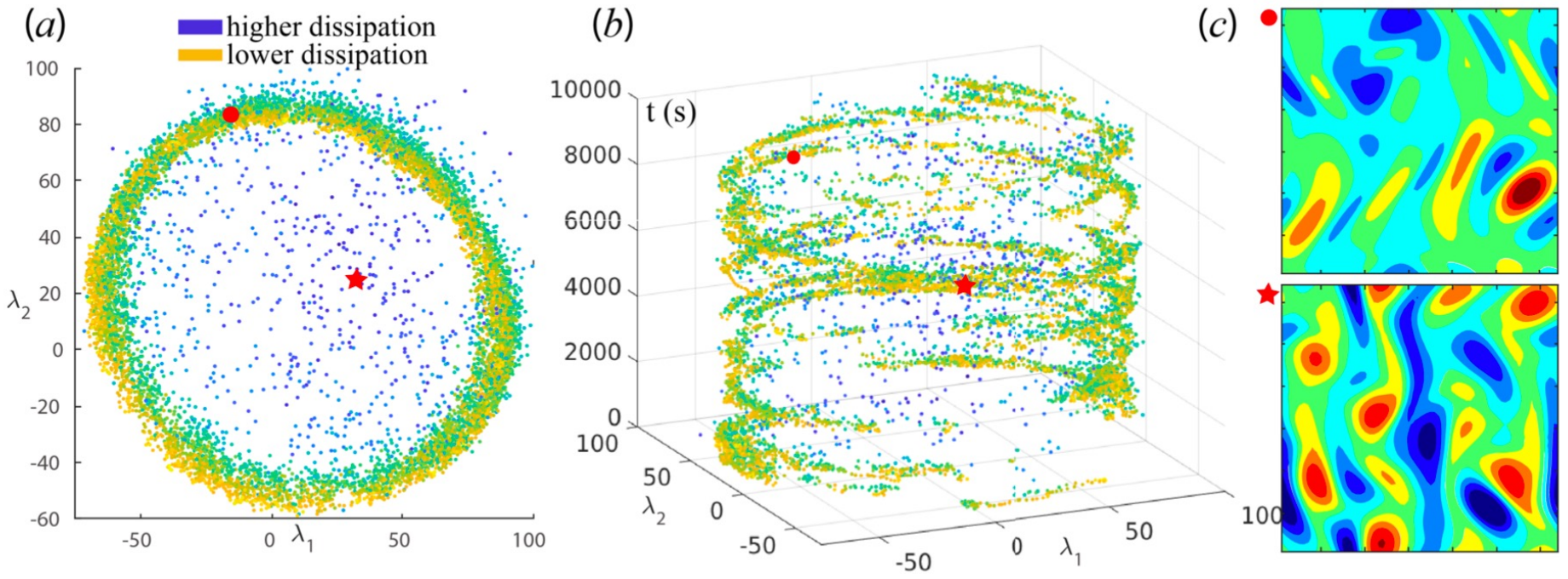
Markov neural operator captures the invariant statistics



AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Markov neural operator simulates the global attractor



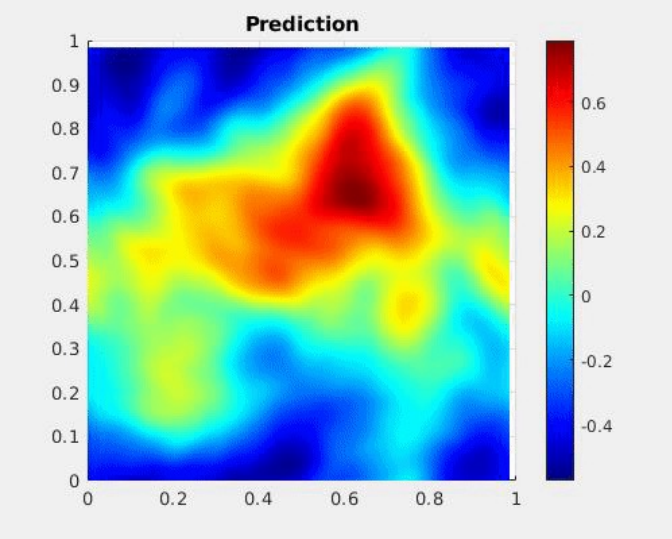
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Enforcing dissipativity: Motivations

Many chaotic systems have the **dissipative** property, which pushes the dynamics back to the attractor.

Without such dissipativity, the model is prone to blow up.



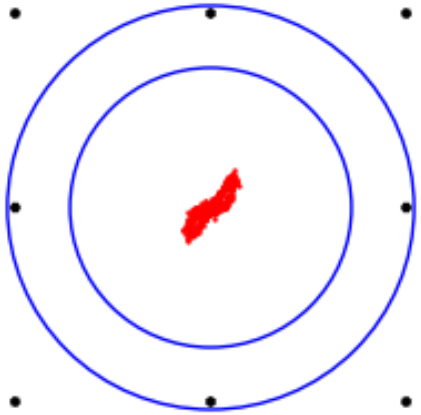
Accelerating flow

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

Methods for Enforcing dissipativity

Enforce dissipativity by augmenting virtual data points around the raw data (equivalent to adding a loss term)



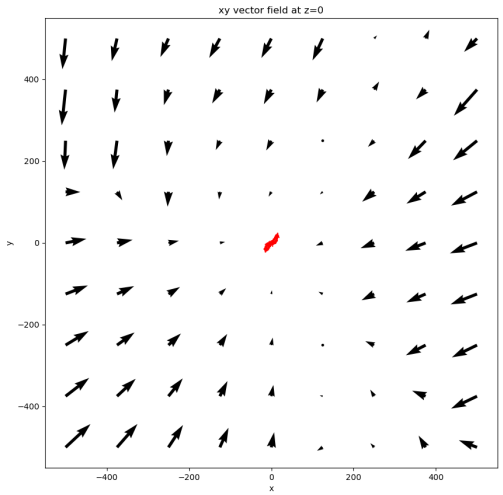
$$\mathcal{L}_{\text{data}} + \frac{\alpha}{2} \int |\Phi(x,\theta) - \lambda x|^2 \, \nu(dx)$$

Red: the raw data (attractor)
Blue shell: the region to add data points.

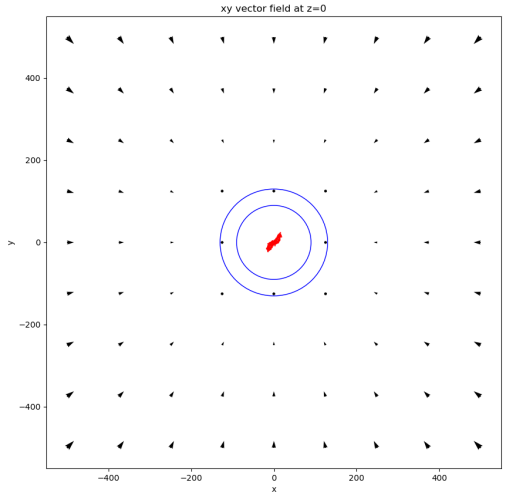
AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

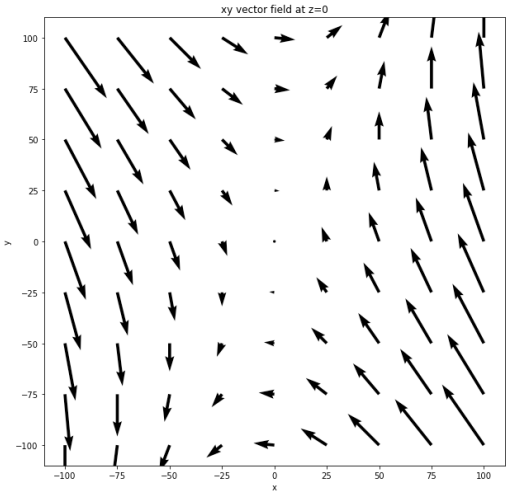
Lorenz 63 learned flow maps



Before



After



True system

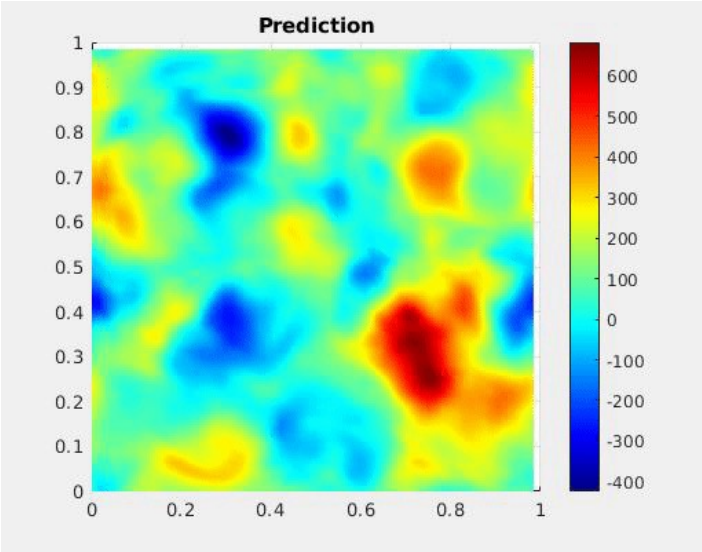
Red points are training points on attractor, blue circles represents shell where dissipativity is enforced.

AI BOOTCAMP

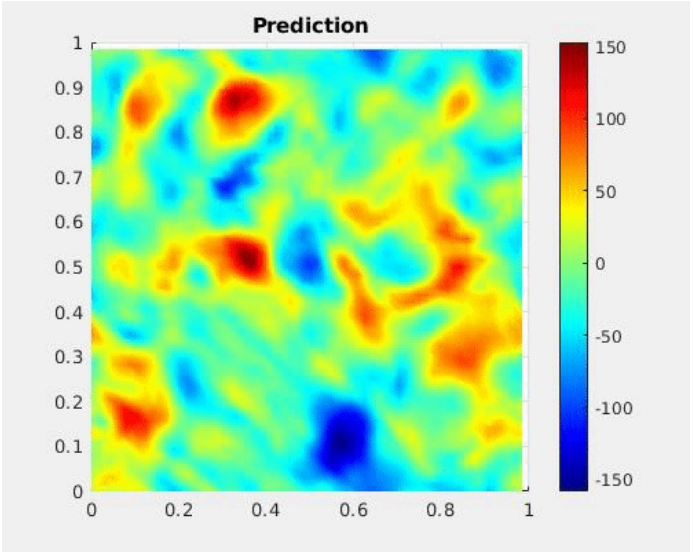
Caltech EAS Division of Engineering and Applied Science

Kolmogorov flow with enforced dissipativity

Start with an initial condition outside the attractor



Before: quickly blowing-up (10^16)



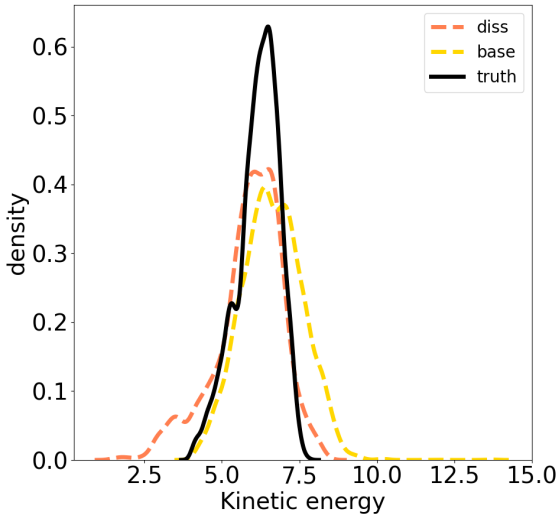
After: converge back to the attractor

AI BOOTCAMP

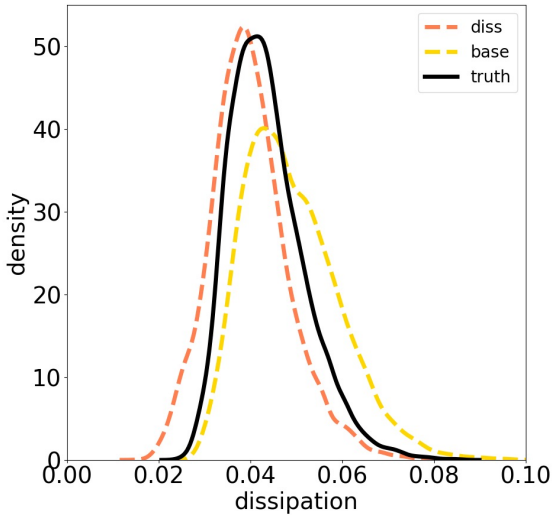
Caltech EAS Division of Engineering and Applied Science

Kolmogorov flow with enforced dissipativity

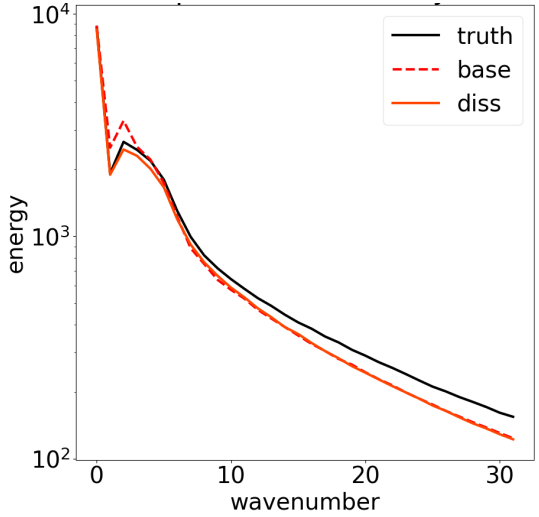
Distribution of kinetic energy



Distribution of dissipation



Spectrum of vorticity

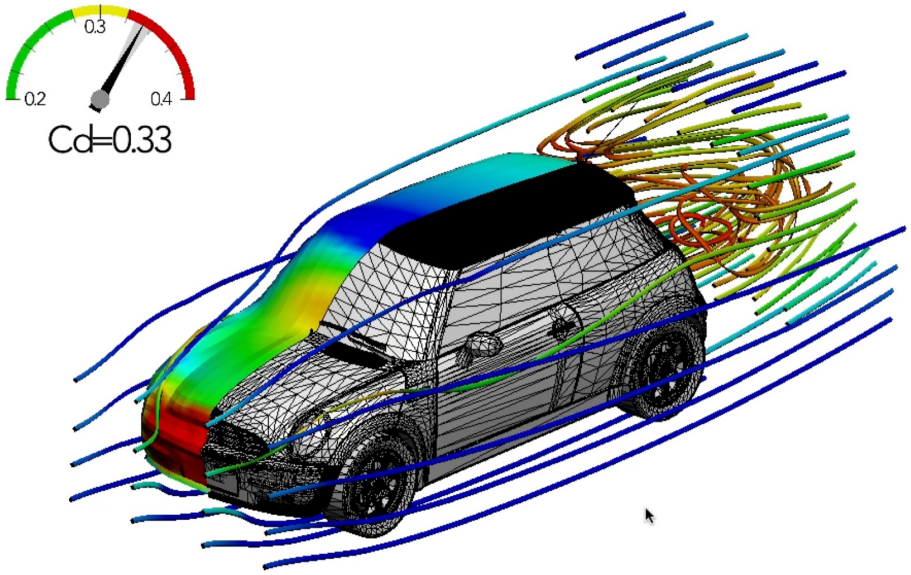


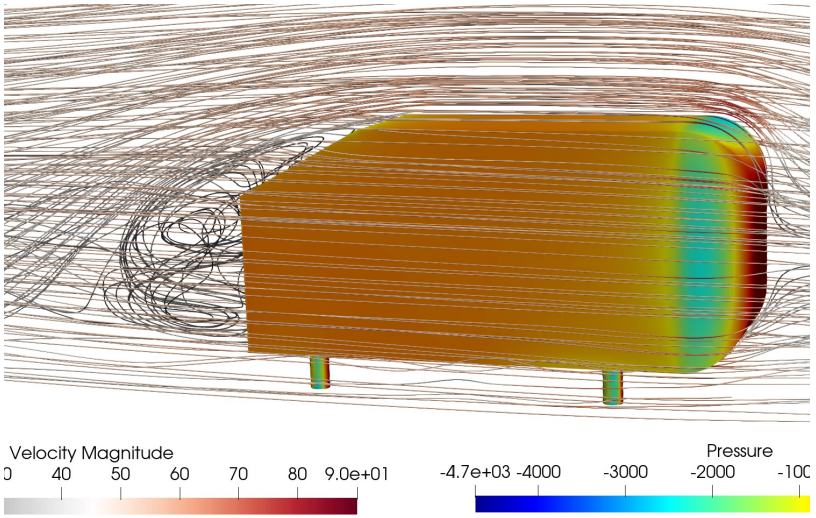
The distribution of dissipation is improved

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

3D Aerodynamics simulations are expensive





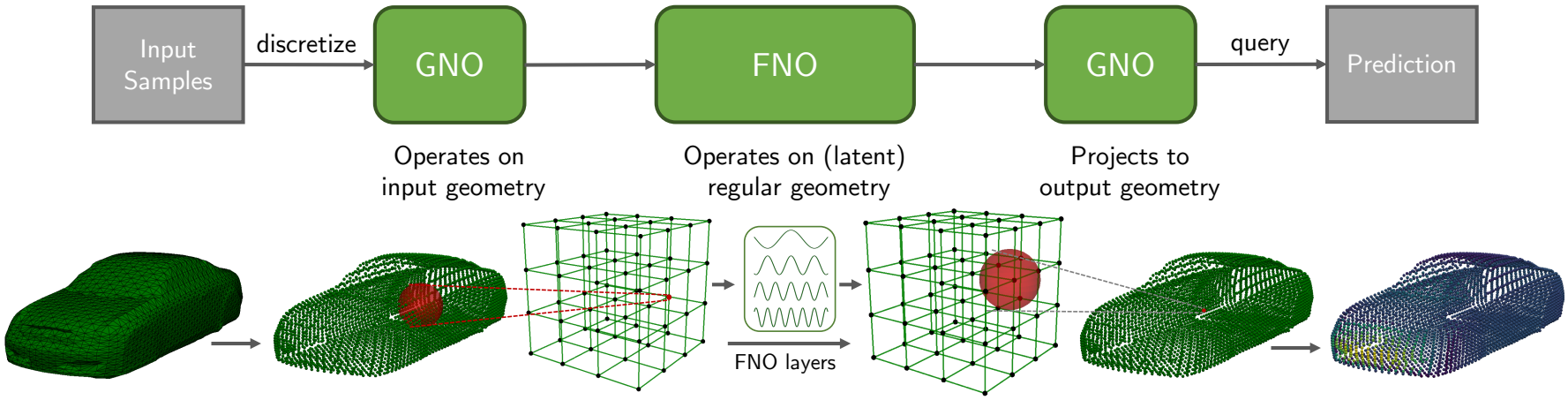
Car-CFD dataset takes 1hr on CPU
10,000x speedup

Ahmed-body dataset takes 10hr on 2 V100 GPUs
30,000x speedup

AI BOOTCAMP

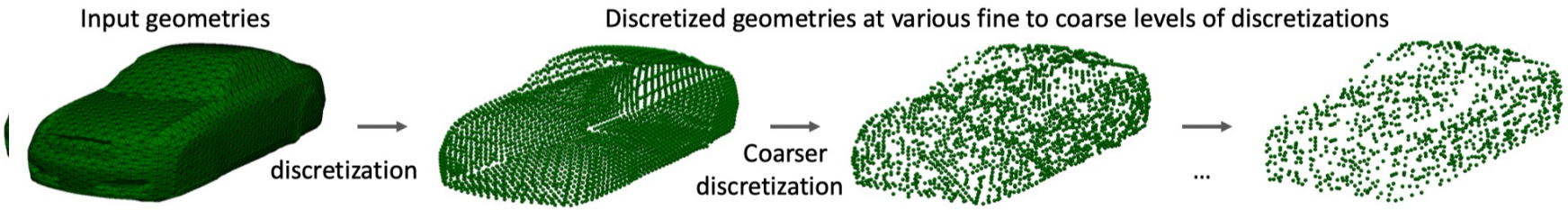
Caltech EAS Division of Engineering and Applied Science

Fourier + Graph



AI BOOTCAMP

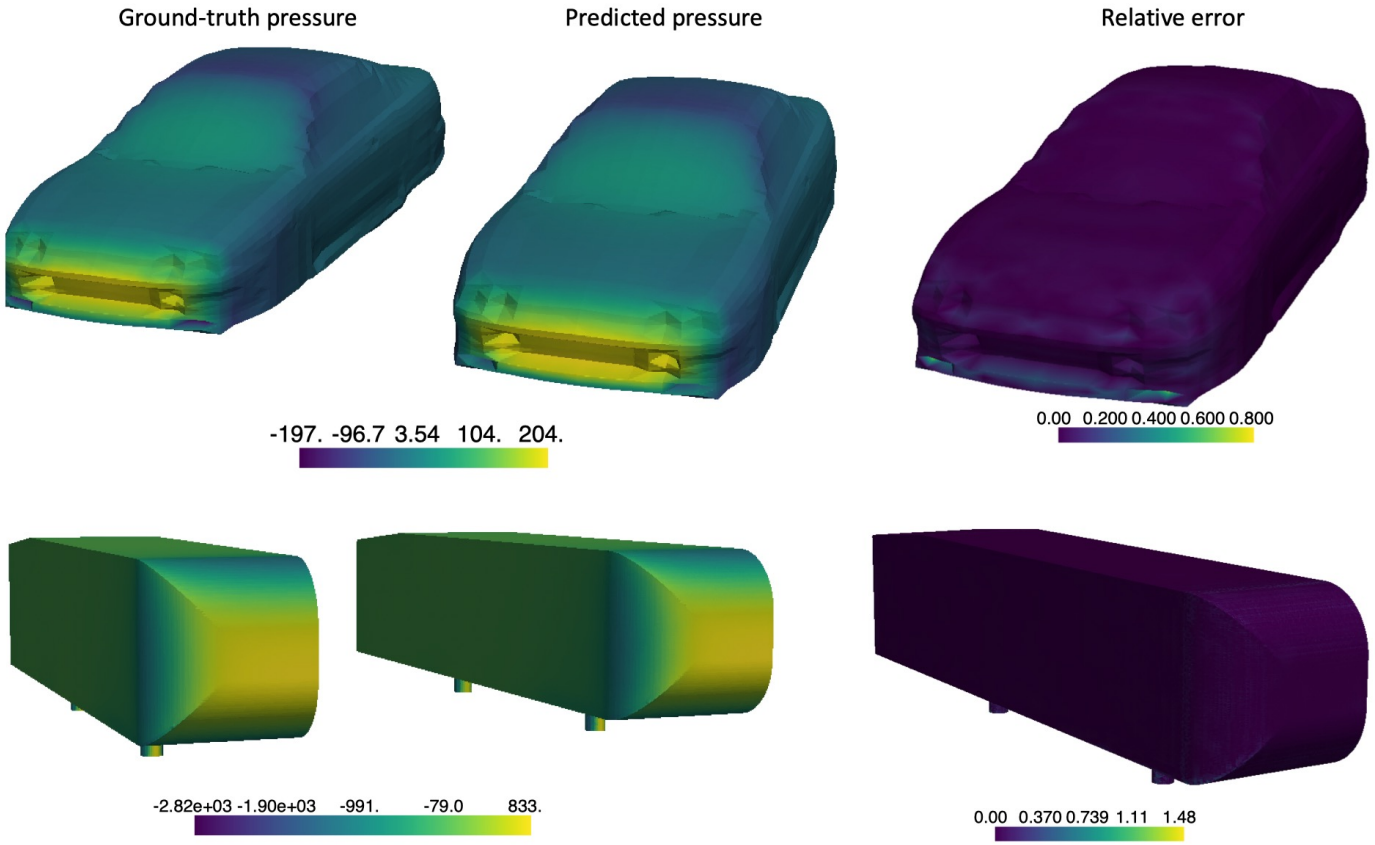
Irregular grid + Discretization convergent



Model	Range	Complexity	Irregular grid	Discretization invariant
GNN	local	$O(N\text{degree})$	✔	✗
CNN	local	$O(N)$	✗	✗
UNet	global	$O(N)$	✗	✗
Transformer	global	$O(N^2)$	✔	✔
GNO (kernel)	radius r	$O(N\text{degree})$	✔	✔
FNO (FFT)	global	$O(N \log N)$	✗	✔
GINO **[Ours]**	global	$O(N \log N + N\text{degree})$	✔	✔

Caltech EAS Division of Engineering and Applied Science

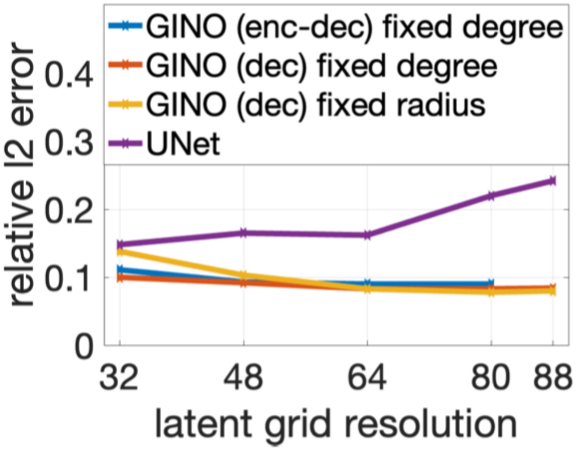
High accuracy



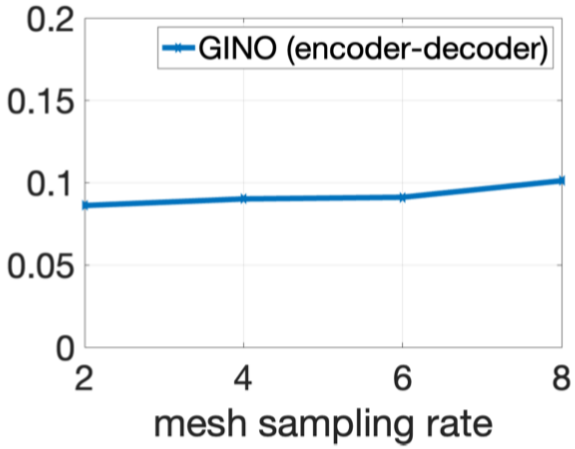
CAMP

Caltech EAS Division of Engineering and Applied Science

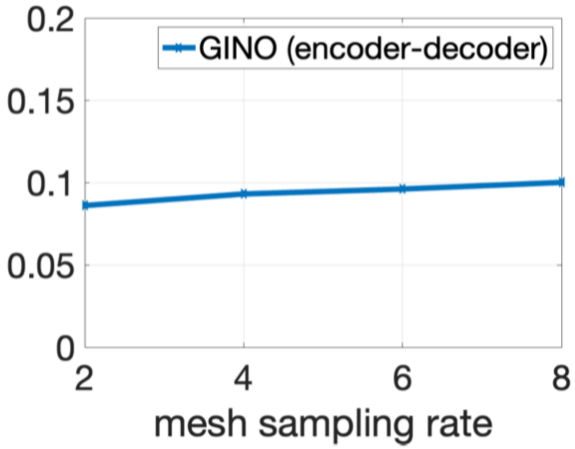
Super resolution



(a) Varying resolutions of the latent grid (same resolution for training and testing).



(b) Varying the sampling rates of the input-output mesh (same rate for training and testing).

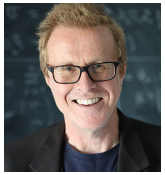


(c) Train with a low sampling rate and test on full mesh (zero-shot super-resolution).

AI BOOTCAMP

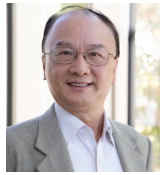
Caltech AS Division of Engineering and Applied Science

Anima Anandkumar



Andrew Stuart

Kaushik Bhattacharya



Thomas Y. Hou

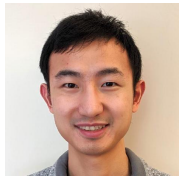
Oscar P. Bruno

Morteza Gharib

Chiara Daraio

Kamyar Azizzadenesheli

Nikola B. Kovachki



Daniel Zhengyu Huang

Jiawei Zhao

Hongkai Zheng

Sahin Lale

Yixuan Wang

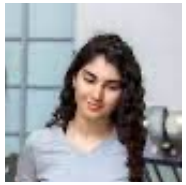
Daniel Leibovici

Miguel Liu-Schiaffini

David Jin

Derek Qin

Haydn Maust



Kimia Hasabi

Vansh Tibrewal

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

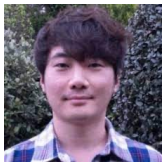
NVIDIA


Kamyar Azizzadenesheli


Nikola B. Kovachki

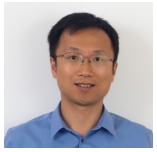

Jean Kossaifi


Boyi Li

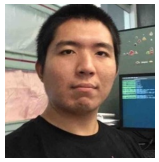

Chris Choy


Jaideep Pathak


Karthik Kashinath


Haoxing Ren


Morteza Mardani

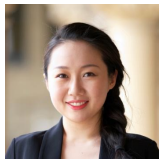

Haoyu Yang


Thorsten Kurth

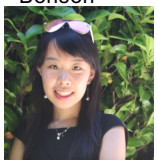

Mingjie Liu

Stanford


Sally M. Benson


Gege Wen

UC San Diego


Yuanyuan Shi


Miroslav Krstic

BERKELEY LAB


Shashank Subramanian


Peter Harrington

Argonne NATIONAL LABORATORY

Berkeley UNIVERSITY OF CALIFORNIA

UNIVERSITY OF CAMBRIDGE


Arvind Ramanathan


Austin Clyde


Sanjeev Raja


Burigede Liu

AI BOOTCAMP

Caltech EAS Division of Engineering and Applied Science

References

1. Pathak, Jaideep, et al. "Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators." *arXiv preprint arXiv:2202.11214* (2022).
2. Wen, Gege, et al. "Real-time high-resolution CO 2 geological storage prediction using nested Fourier neural operators." *Energy & Environmental Science* (2023).
3. Li, Zongyi, et al. "Neural operator: Graph kernel network for partial differential equations." *arXiv preprint arXiv:2003.03485* (2020).
4. Li, Zongyi, et al. "Fourier neural operator for parametric partial differential equations." *arXiv preprint arXiv:2010.08895* (2020).
5. Kovachki, Nikola, et al. "Neural operator: Learning maps between function spaces." *arXiv preprint arXiv:2108.08481* (2021).
6. Kovachki, Nikola, Samuel Lanthaler, and Siddhartha Mishra. "On universal approximation and error bounds for fourier neural operators." *The Journal of Machine Learning Research* 22.1 (2021): 13237-13312.
7. Lanthaler, Samuel, et al. "Nonlinear reconstruction for operator learning of pdes with discontinuities." *arXiv preprint arXiv:2210.01074* (2022).
8. De Hoop, Maarten, et al. "The cost-accuracy trade-off in operator learning with neural networks." *arXiv preprint arXiv:2203.13181* (2022).
9. Takamoto, Makoto, et al. "PDEBench: An extensive benchmark for scientific machine learning." *Advances in Neural Information Processing Systems* 35 (2022): 1596-1611.
10. Gupta, Jayesh K., and Johannes Brandstetter. "Towards Multi-spatiotemporal-scale Generalized PDE Modeling." *arXiv preprint arXiv:2209.15616* (2022).
11. Bi, Kaifeng, et al. "Pangu-Weather: A 3D High-Resolution Model for Fast and Accurate Global Weather Forecast." *arXiv preprint arXiv:2211.02556* (2022).
12. Lam, Remi, et al. "GraphCast: Learning skillful medium-range global weather forecasting." *arXiv preprint arXiv:2212.12794* (2022).
13. Tran, Alasdair, et al. "Factorized fourier neural operators." *arXiv preprint arXiv:2111.13802* (2021).
14. Li, Zongyi, et al. "Physics-informed neural operator for learning partial differential equations." *arXiv preprint arXiv:2111.03794* (2021).
15. Maust, Haydn, et al. "Fourier Continuation for Exact Derivative Computation in Physics-Informed Neural Operators." *arXiv preprint arXiv:2211.15960* (2022).
16. Liu, Burigede, et al. "A learning-based multiscale method and its application to inelastic impact problems." *Journal of the Mechanics and Physics of Solids* 158 (2022): 104668.
17. Williams, Francis, et al. "Neural fields as learnable kernels for 3d reconstruction." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022.
18. Liu, Mingjie, et al. "An Adversarial Active Sampling-based Data Augmentation Framework for Manufacturable Chip Design." *arXiv preprint arXiv:2210.15765* (2022).
19. Li, Zongyi, et al. "Fourier neural operator with learned deformations for pdes on general geometries." *arXiv preprint arXiv:2207.05209* (2022).
20. Li, Zongyi, et al. "Multipole graph neural operator for parametric partial differential equations." *Advances in Neural Information Processing Systems* 33 (2020): 6755-6766.
21. Bruno, Oscar P., Jan S. Hesthaven, and Daniel V. Leibovici. "FC-based shock-dynamics solver with neural-network localized artificial-viscosity assignment." *Journal of Computational Physics: X* 15 (2022): 100110.

AI BOOTCAMP