Learning PDE solution manifolds

Paris Perdikaris
University of Pennsylvania
Microsoft Research AI4Science
email: pgp@seas.upenn.edu

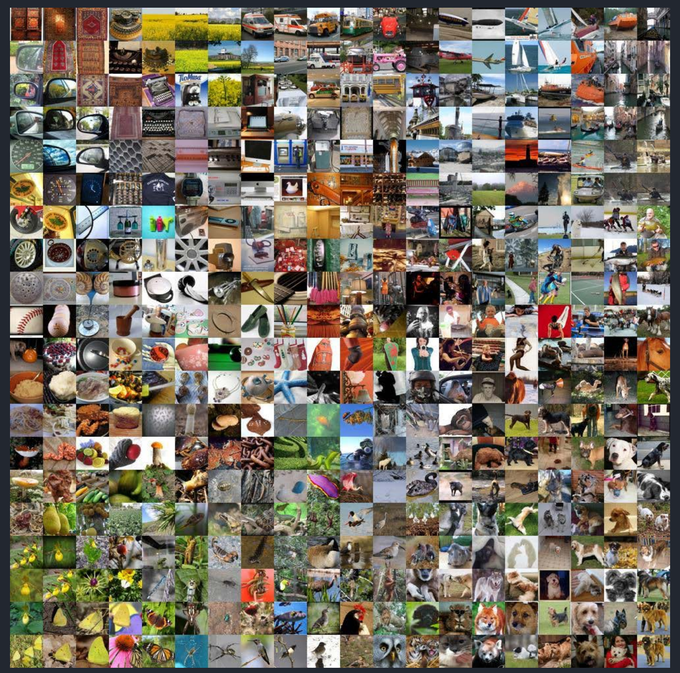
*work performed at the University of Pennsylvania

CWI Autumn School
Scientific Machine Learning and Dynamical Systems
October 12, 2023

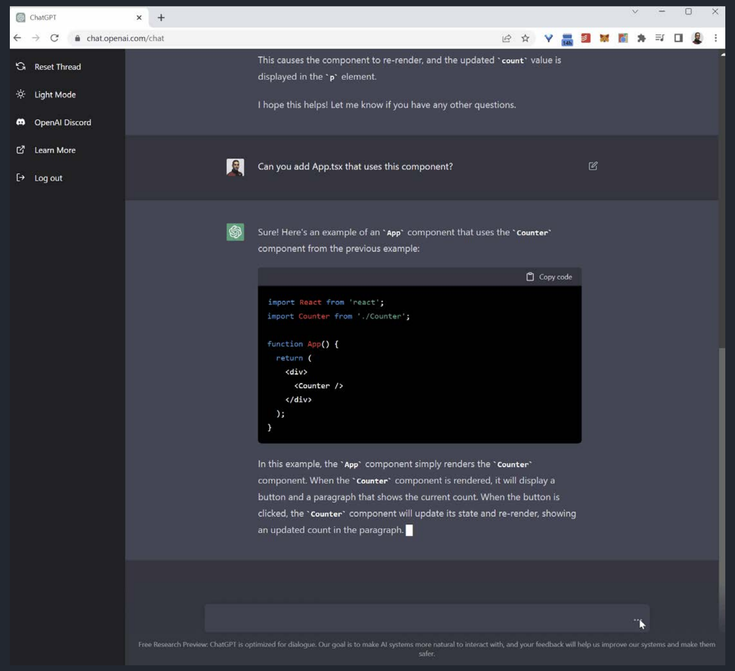
Microsoft

Finite-dimensional vs Functional Data

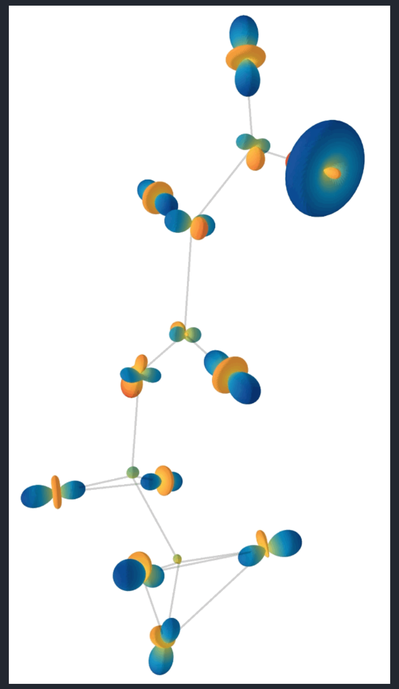
Much of machine learning research focuses on data residing in finite-dimensional vector spaces



images = vectors of pixel values



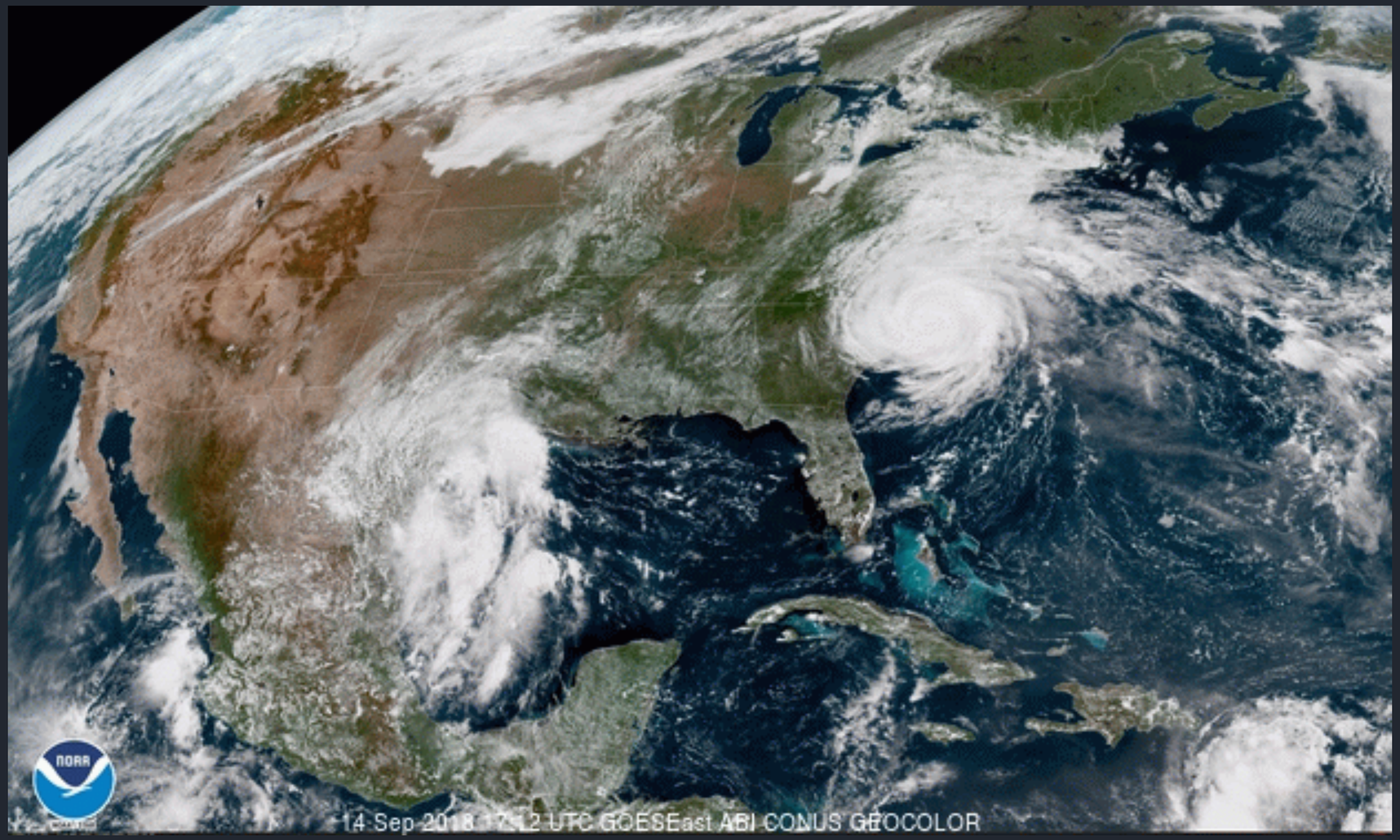
text = vectors of token embdeddings



molecules = vector-valued node/edge features

Finite-dimensional vs Functional Data

Much of science and engineering models signals in the natural world in terms of continuous fields

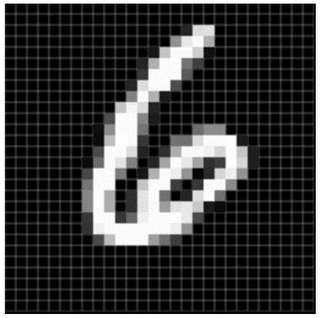


Can we build ML models that act on and return representations of functional data?

Finite-dimensional data

- Data science and machine learning methods have traditionally been applied to learn functions of finite dimensional data

$$\mathbb{R}^{784} \ni \quad \xrightarrow{f : \mathbb{R}^{784} \to \mathbb{R}^{10}} \quad \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \in \mathbb{R}^{10}$$



Functional data

- For many physical applications, we are presented with data from the world as a function over some domain.

- **Function of time**

  Trajectories from a continuous time dynamical system

$$s : [0, T] \to \mathbb{R}^d$$

- **Function of space**

  Measurements over a continuous spatial domain $D \subset \mathbb{R}^n$

$$u : D \to \mathbb{R}^d$$

Functional data

- A single data “point” is a *function* $u : A \to B$

- **Example 1**: A vector in $\mathbb{R}^n$ can be thought of as a function $\{1, \dots, n\} \to \mathbb{R}$

  These are finite dimensional objects as they can be completely characterized by finitely many numbers

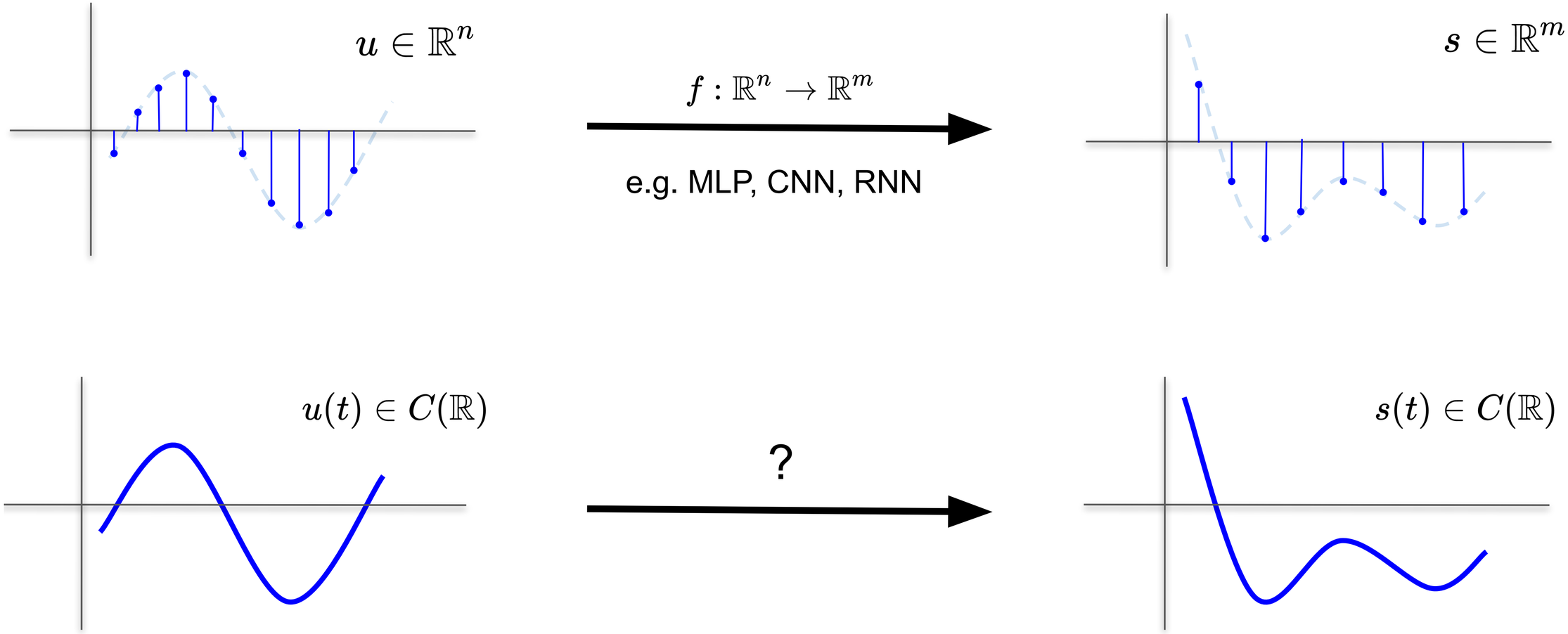
- **Example 2**: Temperature field over the earth $u : S^2 \to \mathbb{R}$

  We can’t uniquely identify every continuous function on the sphere by finitely many numbers

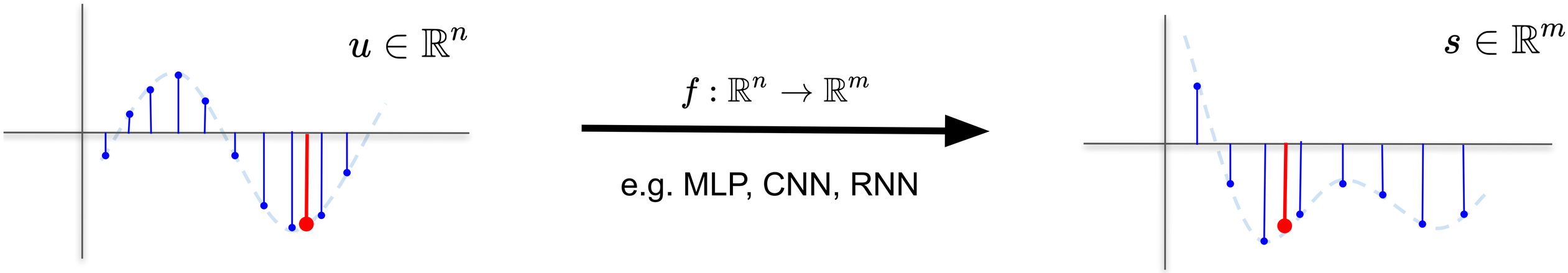
  This kind of data lives in an infinite dimensional space

Learning with functional data

- Take discrete measurements of functional data and use standard ML models on the finite dimensional discretization



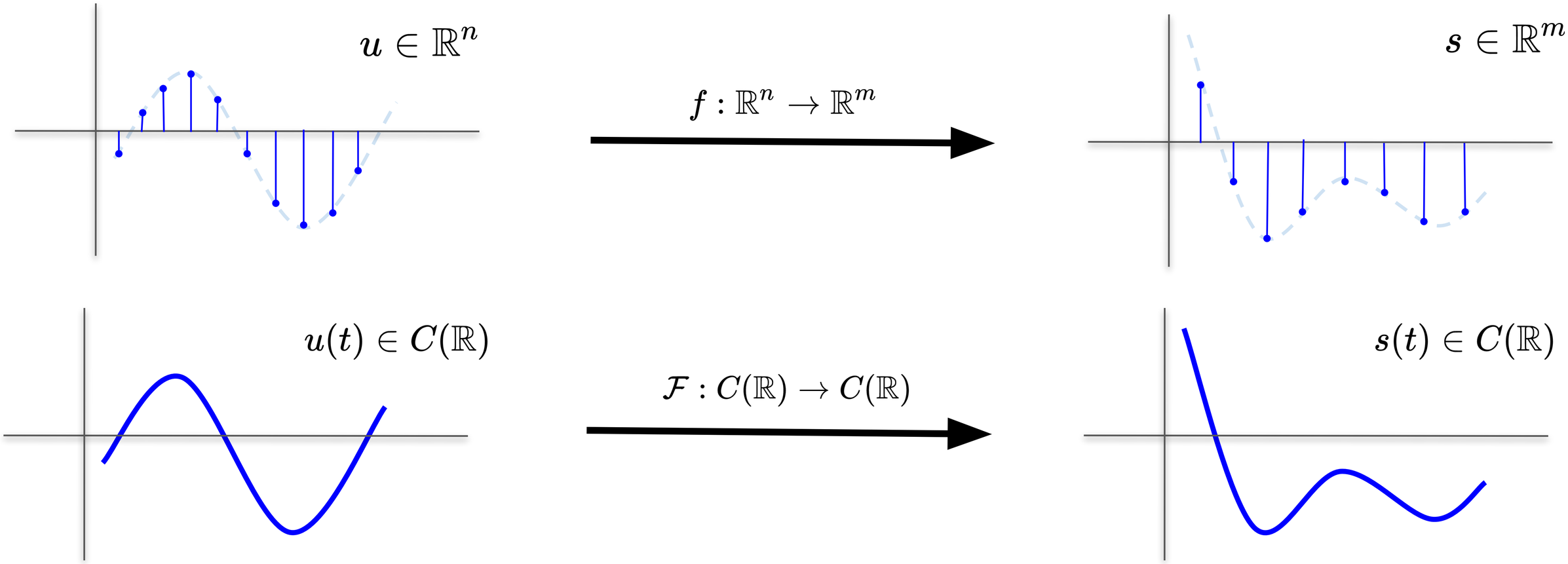
Drawbacks of the discretize-first approach



- Model is not built to accept varying numbers of measurements or new measurement locations
- We are completely constrained to our initial choice of discretization

Operator learning

Instead of learning function between discretizations…



Learn operator between function spaces directly

What could we use this for?

- ODEs with control input

$$u(x) \mapsto s(t) : \begin{cases} \dot{s} = f(s, u(x)) \\ s(0) = s_0 \end{cases}$$

- PDE forward operator

$$u(x) \mapsto s(t, y) : \begin{cases} L(s(t, y)) = f(t, y) \\ s(0, x) = u(x) \end{cases}$$

- More black box relations between functions (e.g. unknown governing PDE)

Notation

- Input functions from a domain $\mathcal{X} \subset \mathbb{R}^{d_x}$ to $\mathbb{R}^{d_u}$

$$u : \mathcal{X} \to \mathbb{R}^{d_u} \qquad u(x)$$

$$u \in C(\mathcal{X}, \mathbb{R}^{d_u})$$

"input function location"

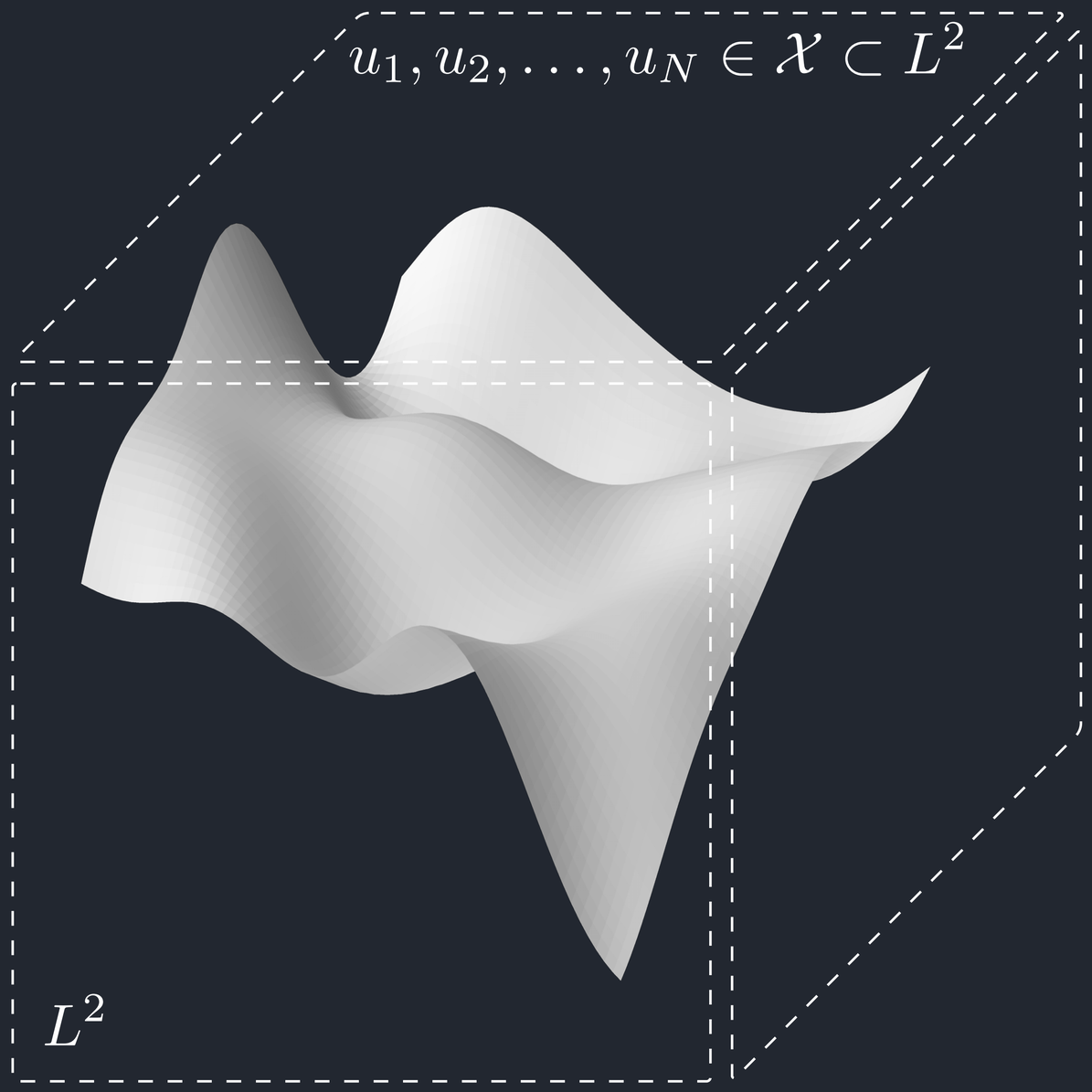
- Output functions from a domain $\mathcal{Y} \subset \mathbb{R}^{d_y}$ to $\mathbb{R}^{d_s}$

$$s : \mathcal{Y} \to \mathbb{R}^{d_s} \qquad s(y)$$

$$s \in C(\mathcal{Y}, \mathbb{R}^{d_s})$$

"query" or "query location"

The Manifold Hypothesis



Collections of functional signals in the real world actually lie on low-dimensional latent manifolds inside their ambient function space.

<u>Supervised learning in function spaces</u>

Given a data-set of N pairs of functions

$$\{(u^1, s^1), \ldots, (u^N, s^N)\}$$

learn an operator

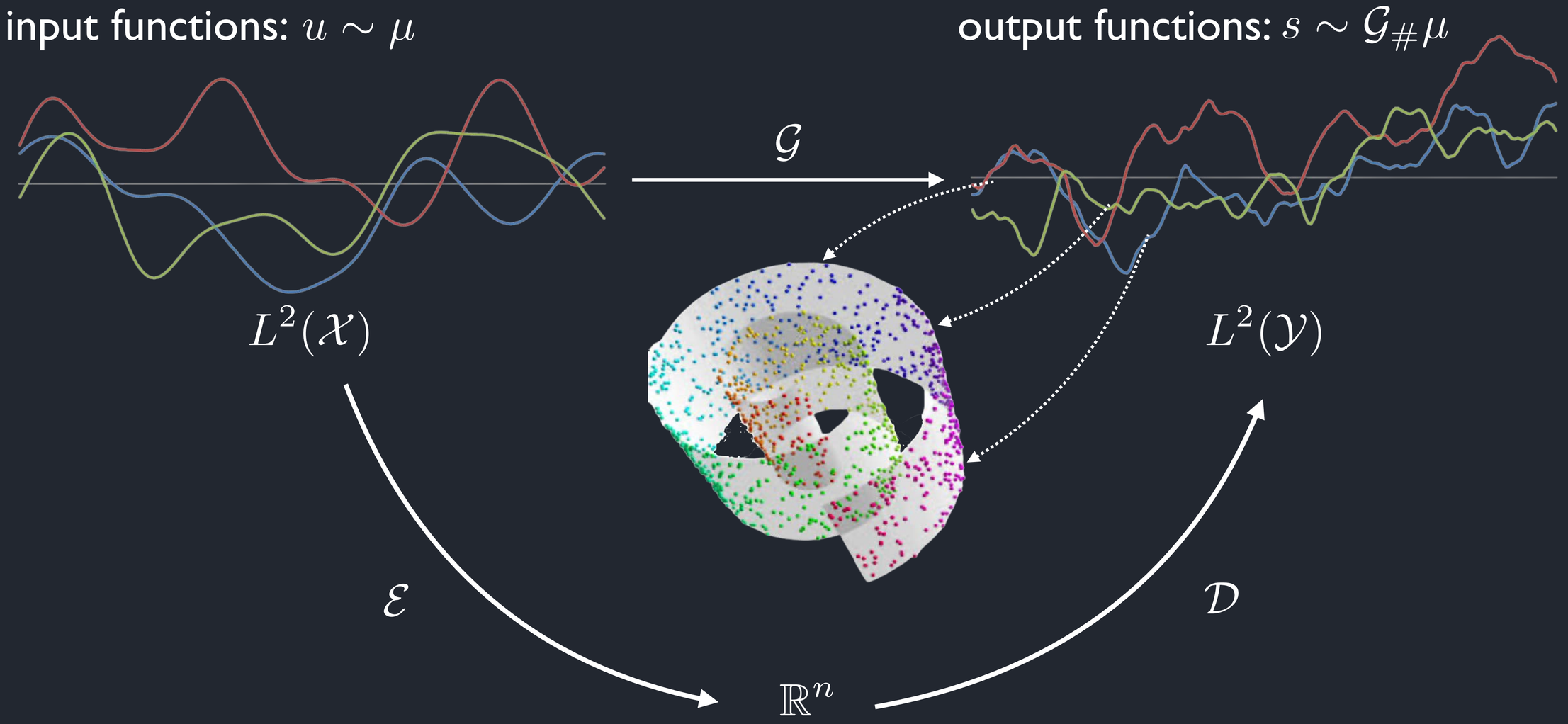
$$\mathcal{F} : u \mapsto s$$

which maps input functions to output functions

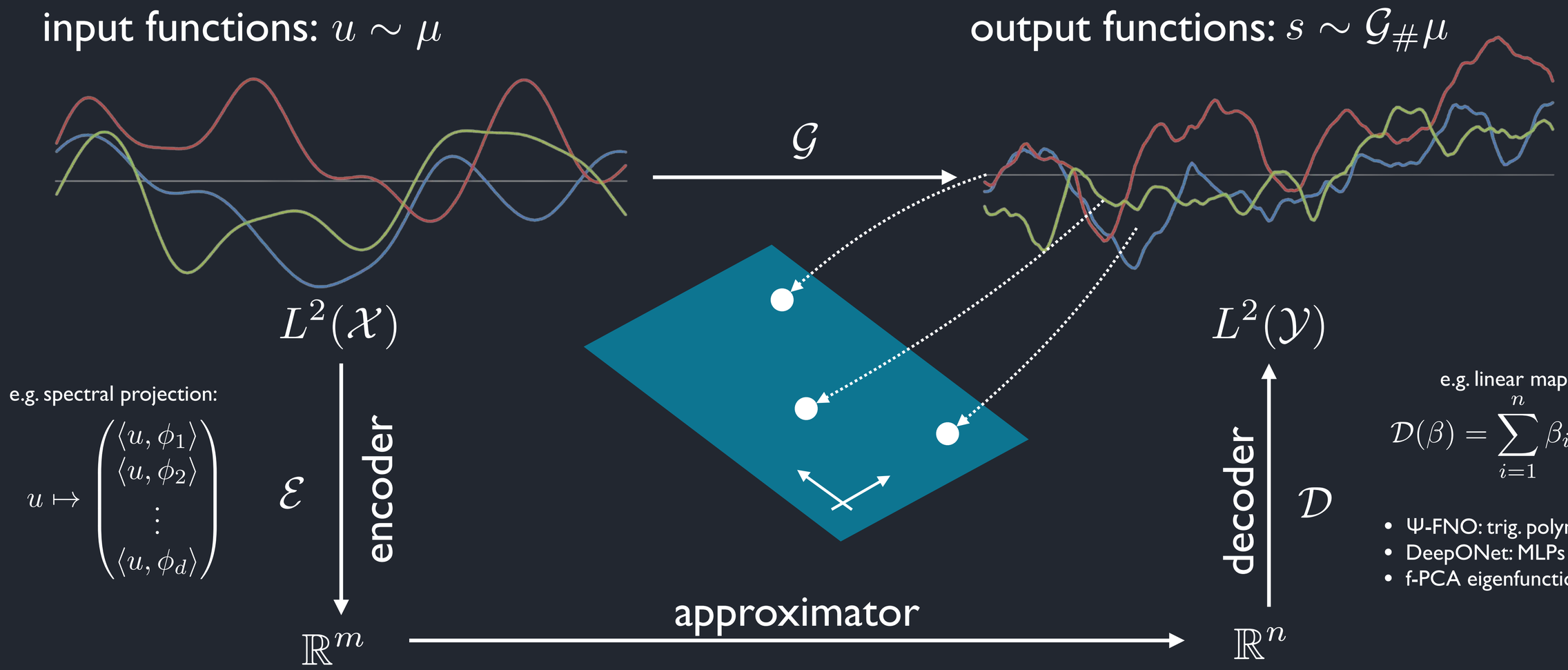
such that: $\mathcal{F}(u^i) = s^i, \ \forall i$

Cohen, A., & DeVore, R. (2015). Approximation of high-dimensional parametric PDEs. Acta Numerica, 24, 1-159

Neural Operators



Learning linear subspaces

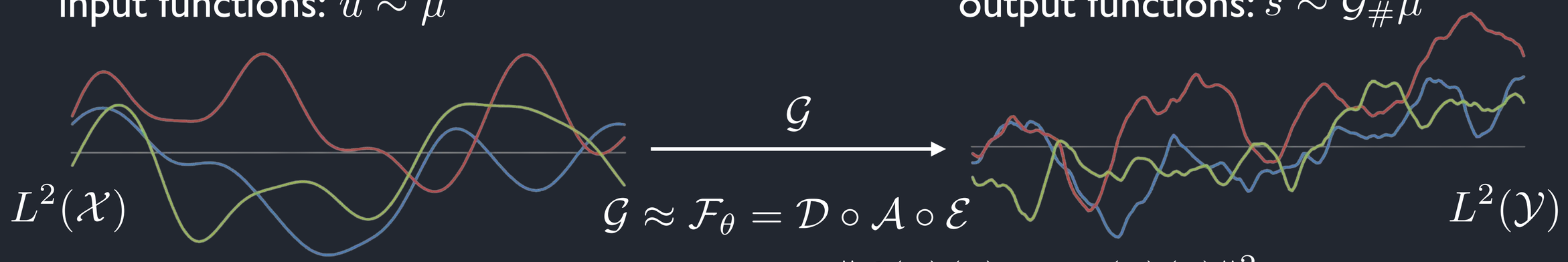


Chen, T., & Chen, H. (1995). Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. IEEE Transactions on Neural Networks.

Error bounds for linear decoders

input functions: $u \sim \mu$

output functions: $s \sim \mathcal{G}_{\#}\mu$



universal approximation: $\sup_{u \in \mathcal{U}} \sup_{y \in \mathcal{Y}} \|\mathcal{G}(u)(y) - \mathcal{F}_\theta(u)(y)\|_2^2 < \epsilon$

covariance operator on $L^2(\mathcal{Y})$

$$\Gamma : L^2(\mathcal{Y}) \to L^2(\mathcal{Y}), \quad \Gamma(f) = \int_{\mathcal{Y}} k(y, y') f(y') dy', \quad k(y, y') = \sum_{j=1}^{\infty} \sqrt{\lambda_j} \phi_j(y) \phi_j(y')$$

fundamental lower bounds for linear decoders by Lanthaler et al (2022), Kovachki et. al (2022)

$$\mathbb{E}_{u \sim \mu} \left[\|\mathcal{F}(u) - \mathcal{G}(u)\|_{L^2}^2 \right] \geq \sum_{k > n} \lambda_k \qquad \sup_{u \in \mathcal{U}} \|\mathcal{F}(u) - \mathcal{G}(u)\|_{L^2(\mathcal{Y})} \geq d_n(\mathcal{G}(\mathcal{U}))$$

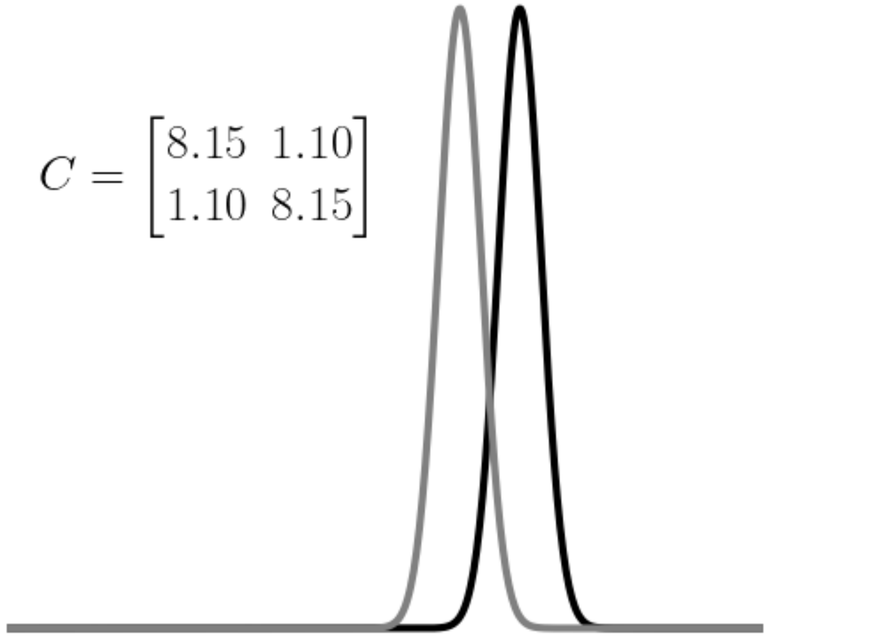
Kolmogorov n-width

Seidman, J. H., Kissas, G., Perdikaris, P., & Pappas, G. J. (2022). NOMAD: Nonlinear Manifold Decoders for Operator Learning. arXiv preprint arXiv:2206.03551. (Accepted in NeurIPS 2022)

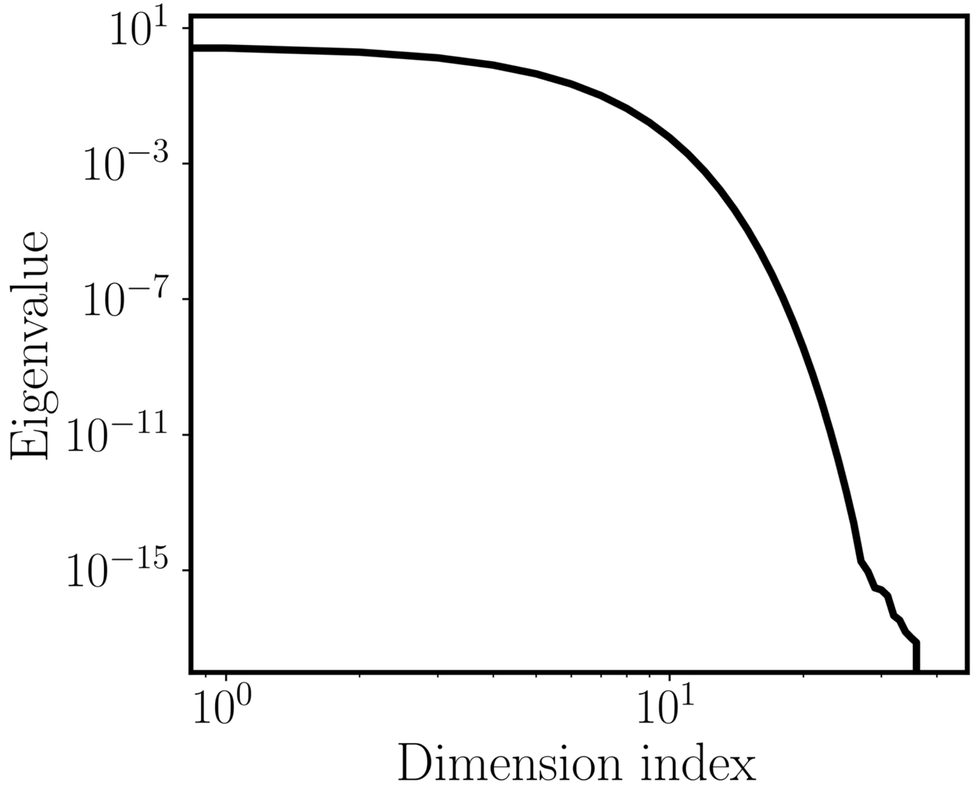
Advection-dominated PDE solutions exhibit slow spectral decay

$$\partial_t s(x,t) + \partial_x s(x,t) = 0$$

$$C = \begin{bmatrix} 8.15 & 1.10 \\ 1.10 & 8.15 \end{bmatrix}$$



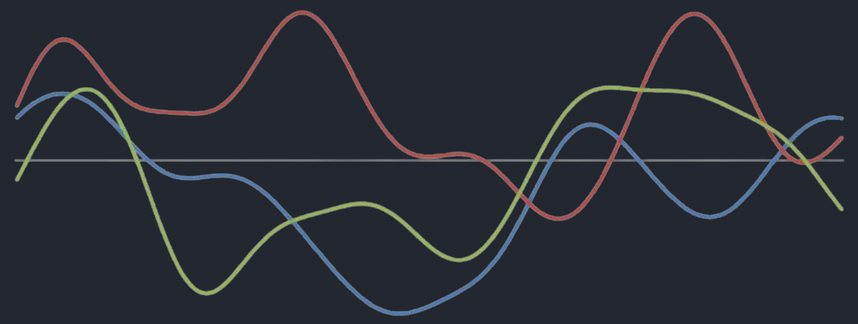
$s(x,t)$



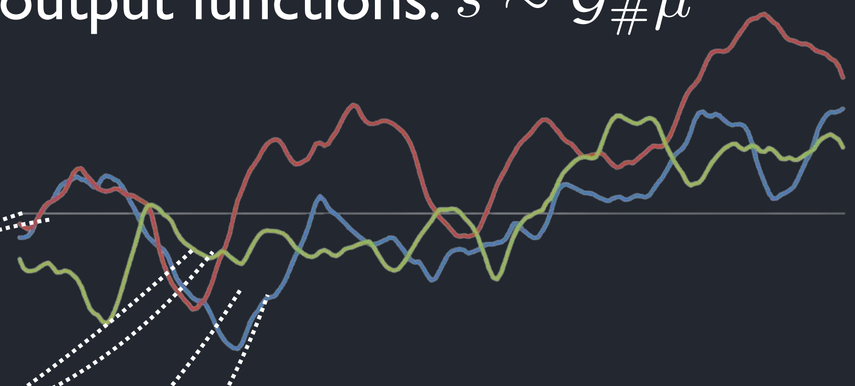
Learning nonlinear manifolds

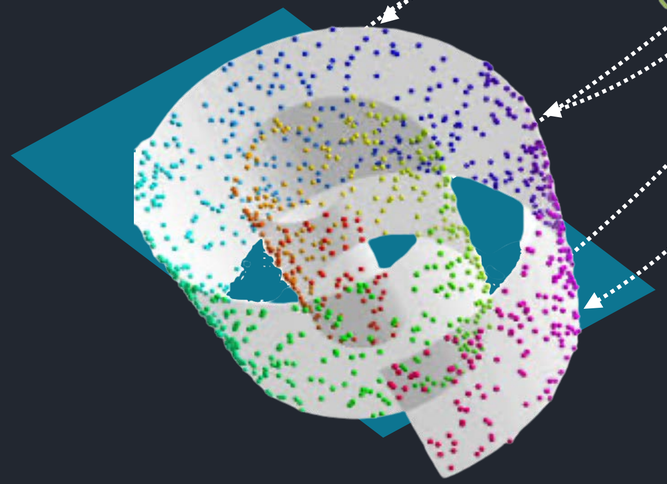
input functions: $u \sim \mu$

output functions: $s \sim \mathcal{G}_{\#}\mu$



$\mathcal{G}$





$L^2(\mathcal{X})$

$L^2(\mathcal{Y})$

e.g. spectral projection:

$$u \mapsto \begin{pmatrix} \langle u, \phi_1 \rangle \\ \langle u, \phi_2 \rangle \\ \vdots \\ \langle u, \phi_d \rangle \end{pmatrix}$$

$\mathcal{E}$ encoder

decoder $\mathcal{D}$

e.g. linear maps

$$\mathcal{D}(\beta) = \sum_{i=1}^{n} \beta_i \tau_i(\cdot)$$

$$\mathcal{D}(\beta) = f(\beta, \cdot)$$

- Ψ-FNO: trig. polynomials
- DeepONet: MLPs
- f-PCA eigenfunctions

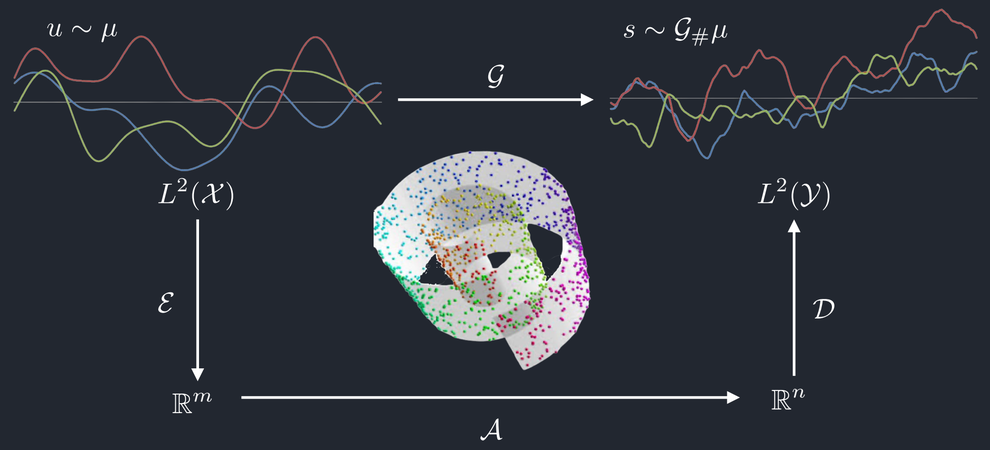
approximator

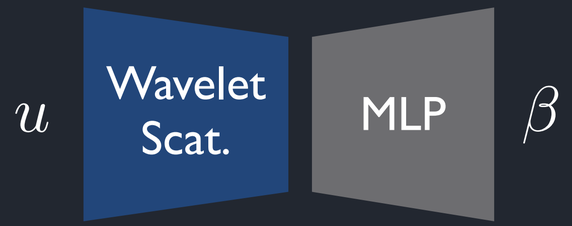
$\mathbb{R}^m$ $\mathcal{A}$ $\mathbb{R}^n$

e.g. MLPs, CNNs, GNNs, Trasformers

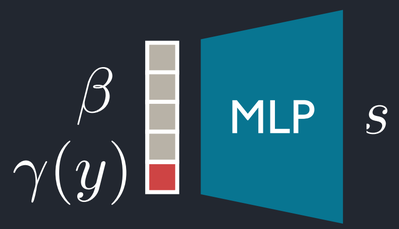
Seidman, J. H., Kissas, G., Perdikaris, P., & Pappas, G. J. (2022). NOMAD: Nonlinear Manifold Decoders for Operator Learning. arXiv preprint arXiv:2206.03551. (Accepted in NeurIPS 2022)

Nonlinear Decoder Architectures

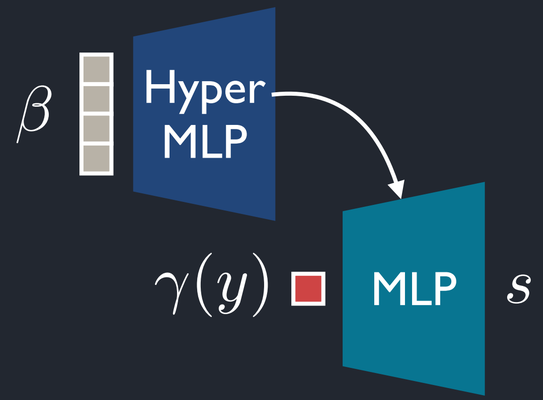




concatenation



hypernetwork



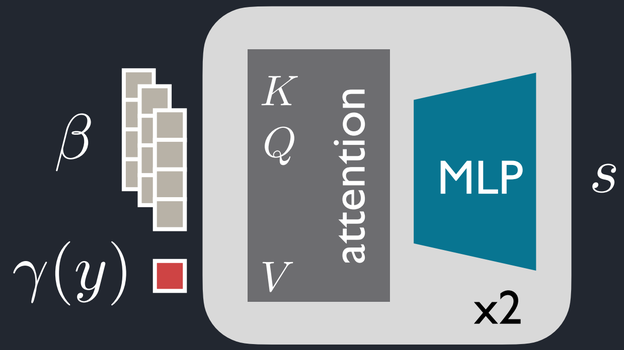
$\beta = \mathcal{A} \circ \mathcal{E}(u) \in \mathbb{R}^n$: latent coordinates

y: output function query location

$\gamma(y)$: positional encoding (optional)

$s(y) = f(\beta, y)$: target output function value

cross-attention



kernel-coupled attention

$$s(y) = \sigma\left(\int k(y,y')\ \underbrace{g(\beta,y')}_{\text{HyperMLP}}\ dy'\right) \odot \underbrace{v(\beta)}_{\text{MLP}}$$

Bruna, J., & Mallat, S. (2013). Invariant scattering convolution networks. IEEE transactions on pattern analysis and machine intelligence, 35(8), 1872-1886.

Rebain, D., Matthews, M. J., Yi, K. M., Sharma, G., Lagun, D., & Tagliasacchi, A. (2022). Attention Beats Concatenation for Conditioning Neural Fields. arXiv preprint arXiv:2209.10684.

Seidman, J. H., Kissas, G., Perdikaris, P., & Pappas, G. J. (2022). NOMAD: Nonlinear Manifold Decoders for Operator Learning. arXiv preprint arXiv:2206.03551. (Accepted in NeurIPS 2022)

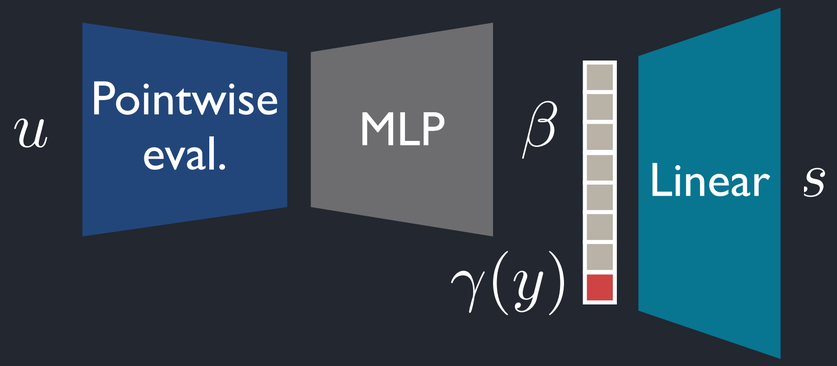
Kissas, G., Seidman, J. H., Guilhoto, L. F., Preciado, V. M., Pappas, G. J., & Perdikaris, P. (2022). Learning operators with coupled attention. Journal of Machine Learning Research, 23(215), 1-63.

Advection PDE

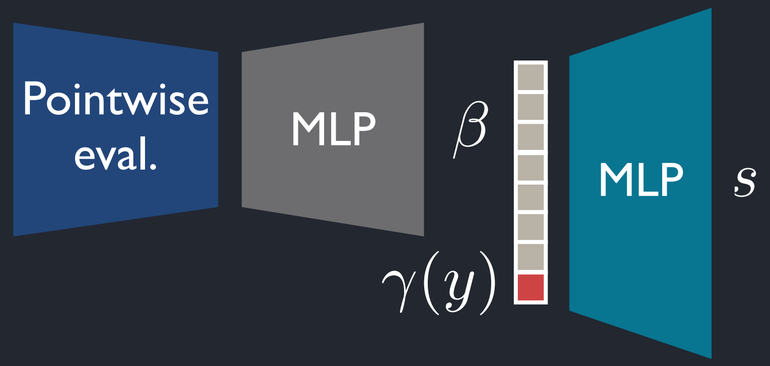
$$\partial_t s(x,t) + \partial_x s(x,t) = 0$$

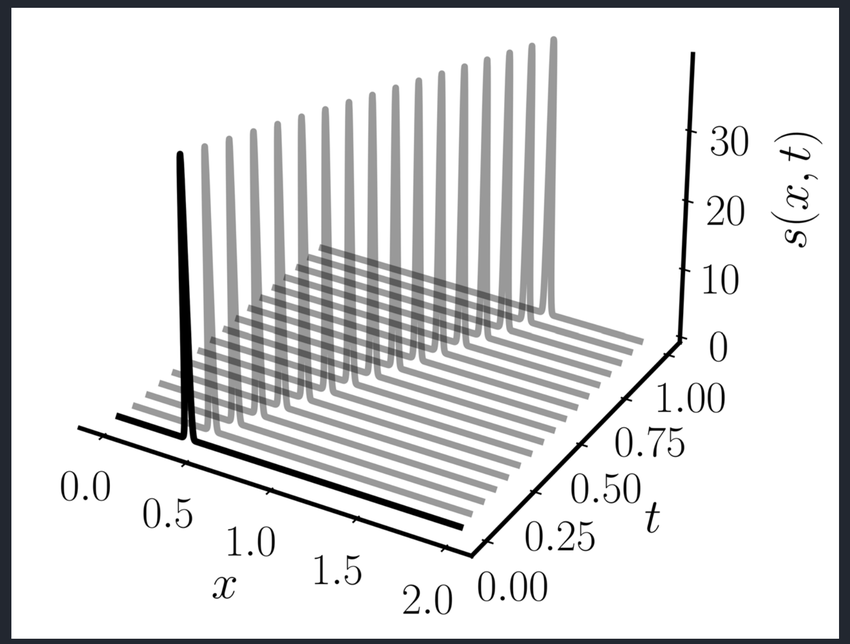
$$\mathcal{G} : s(x,0) \mapsto s(x,t)$$

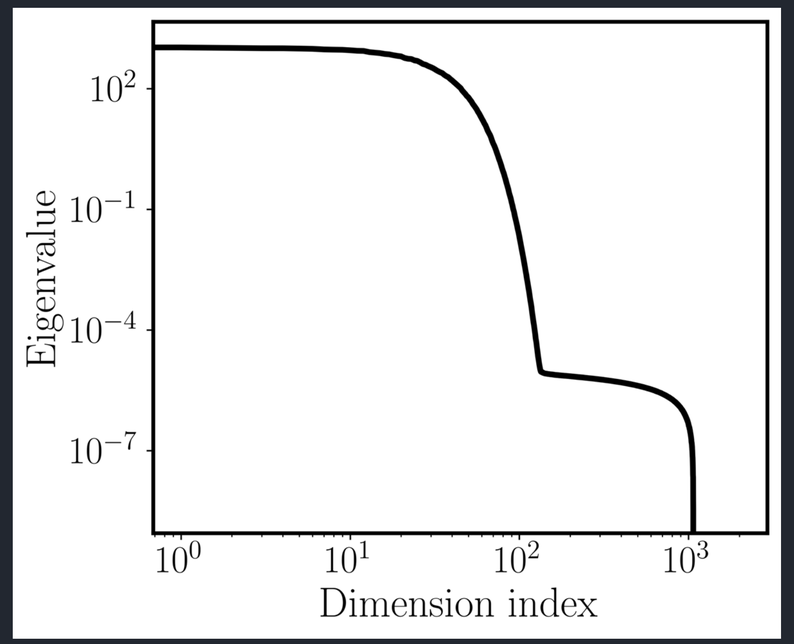
Linear decoder (DeepOnet, Lu et al. 2021)

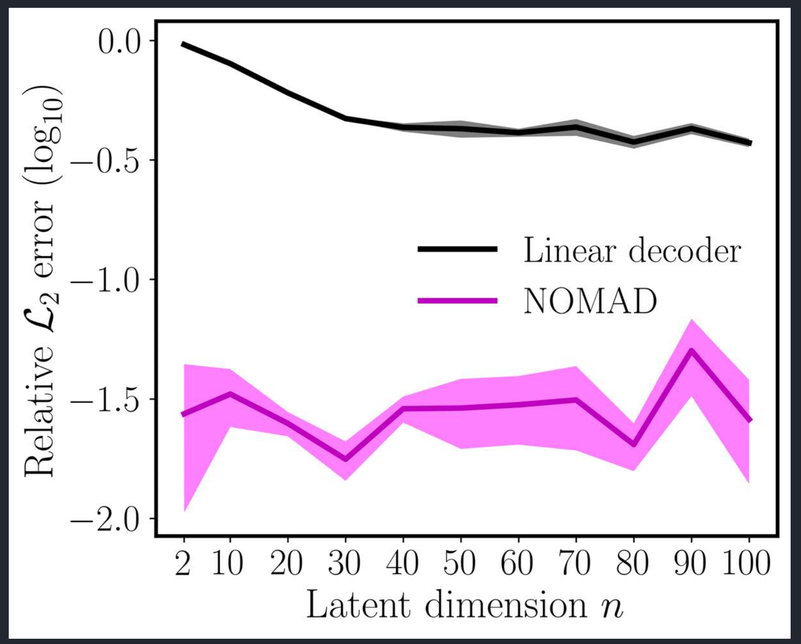


NOMAD (Seidman, et al. 2022)







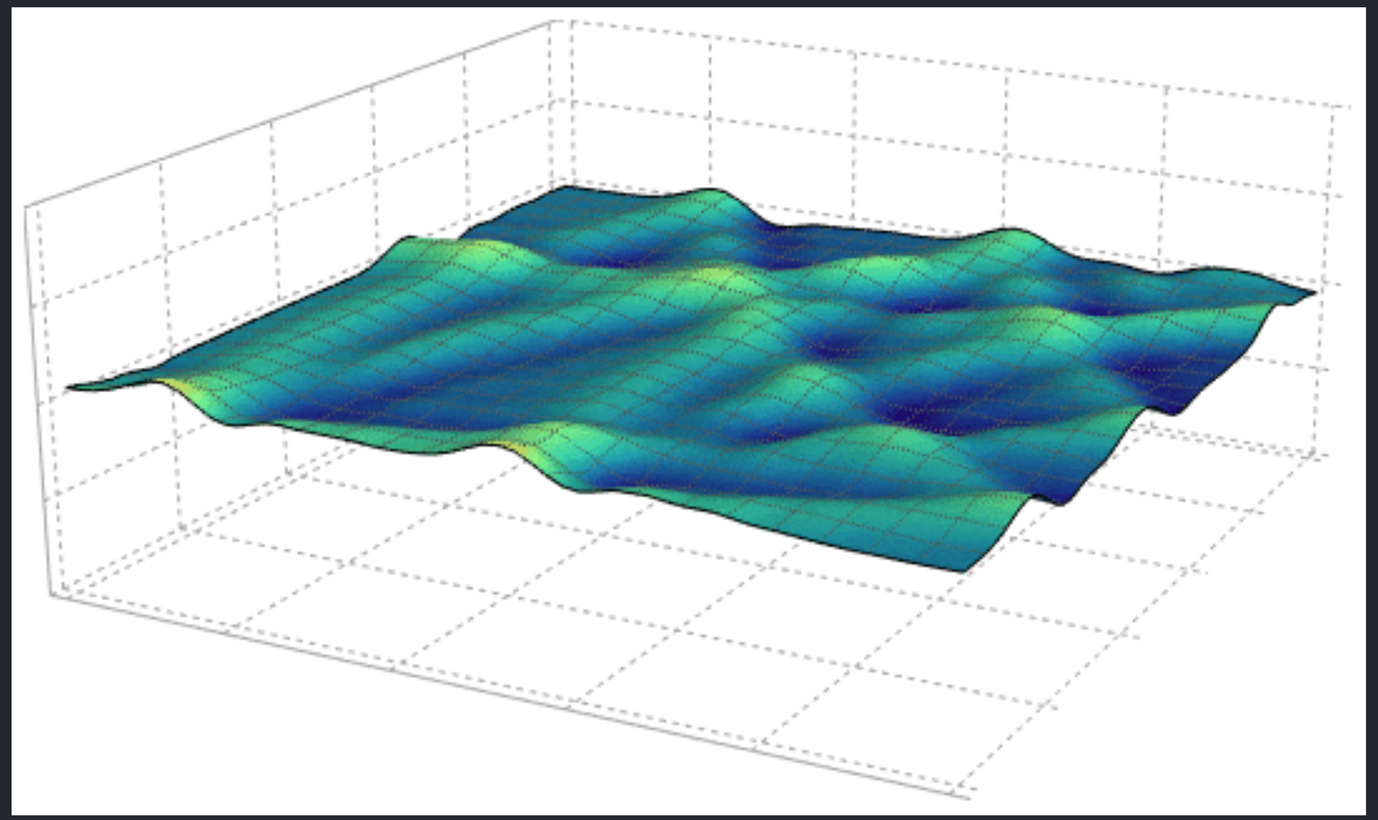


Seidman, J. H., Kissas, G., Perdikaris, P., & Pappas, G. J. (2022). NOMAD: Nonlinear Manifold Decoders for Operator Learning. arXiv preprint arXiv:2206.03551. (Accepted in NeurIPS 2022)

Nonlinear wave propagation

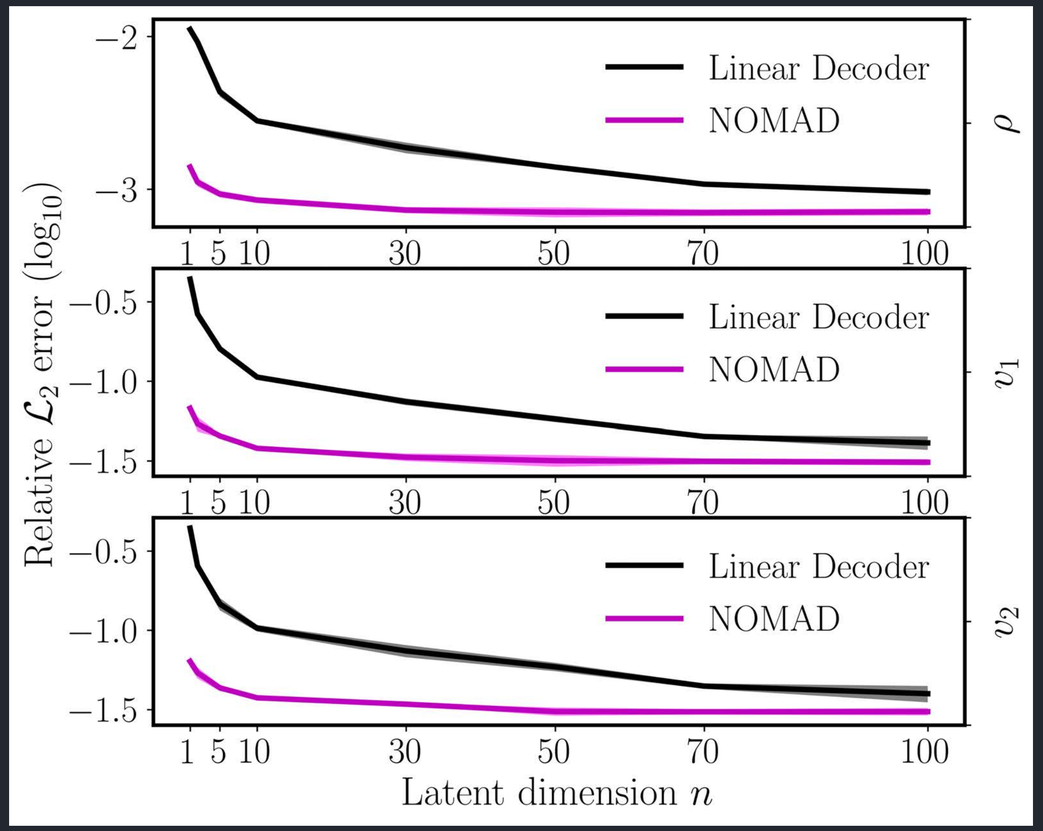
$$\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}}{\partial x} + \frac{\partial \boldsymbol{G}}{\partial y} = 0 \qquad \boldsymbol{U} = \begin{pmatrix} \rho \\ \rho v_1 \\ \rho v_2 \end{pmatrix}, \quad \boldsymbol{F} = \begin{pmatrix} \rho v_1 \\ \rho v_1^2 + \frac{1}{2} g \rho^2 \\ \rho v_1 v_2 \end{pmatrix}, \quad \boldsymbol{G} = \begin{pmatrix} \rho v_2 \\ \rho v_1 v_2 \\ \rho v_2^2 + \frac{1}{2} g \rho^2 \end{pmatrix}$$

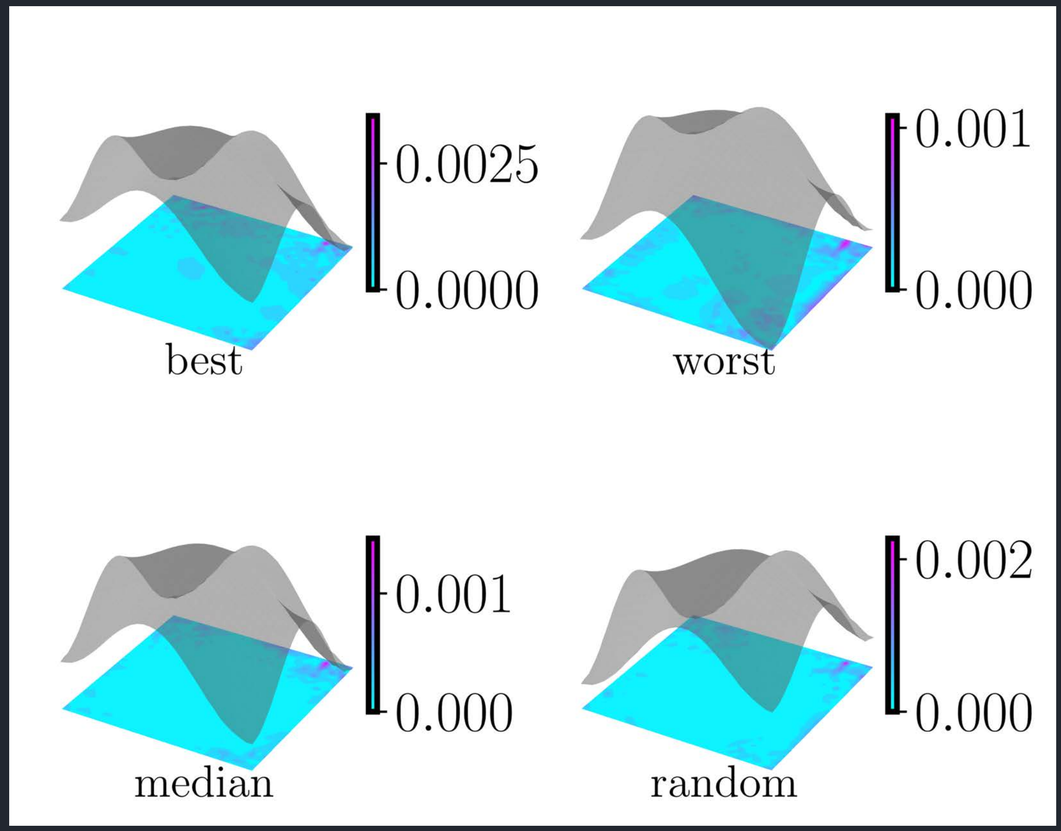
$$\mathcal{G} : \boldsymbol{U}(x, y, 0) \mapsto \boldsymbol{U}(x, y, t)$$



Seidman, J. H., Kissas, G., Perdikaris, P., & Pappas, G. J. (2022). NOMAD: Nonlinear Manifold Decoders for Operator Learning. arXiv preprint arXiv:2206.03551. (Accepted in NeurIPS 2022)

Nonlinear wave propagation





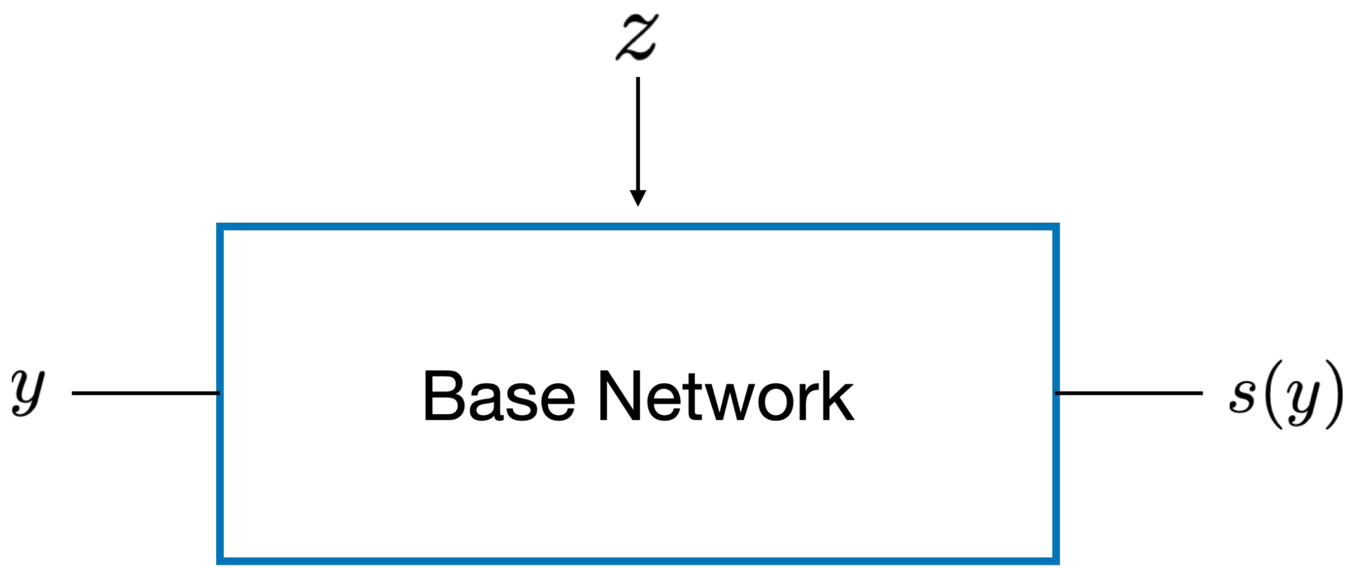
Method	ρ	v_1	v_2	worst case	d_θ	n	cost
LOCA	$\mathbf{0.040 \pm 0.015}$	2.7 ± 0.3	2.9 ± 0.4	$(\mathbf{0.1}, \mathbf{3.5}, \mathbf{4.2})$	$O(10^6)$	480	12.1
DON	0.100 ± 0.030	5.5 ± 1.2	5.9 ± 1.4	$(0.6, 11, 11)$	$O(10^6)$	480	15.4
FNO	0.140 ± 0.060	3.4 ± 1.2	3.5 ± 1.2	$(0.4, 8.9, 8.7)$	$O(10^6)$	N/A	14.0
NOMAD	0.048 ± 0.017	$\mathbf{2.0 \pm 0.4}$	$\mathbf{2.6 \pm 0.3}$	$(0.1, 5.8, 4.9)$	$\mathbf{O(10^5)}$	**20**	**5.5**

Seidman, J. H., Kissas, G., Perdikaris, P., & Pappas, G. J. (2022). NOMAD: Nonlinear Manifold Decoders for Operator Learning. arXiv preprint arXiv:2206.03551. (Accepted in NeurIPS 2022)

Conditioned neural fields

- Encode multiple signals simultaneously **without retraining** by adjusting/conditioning weights with an auxiliary variable

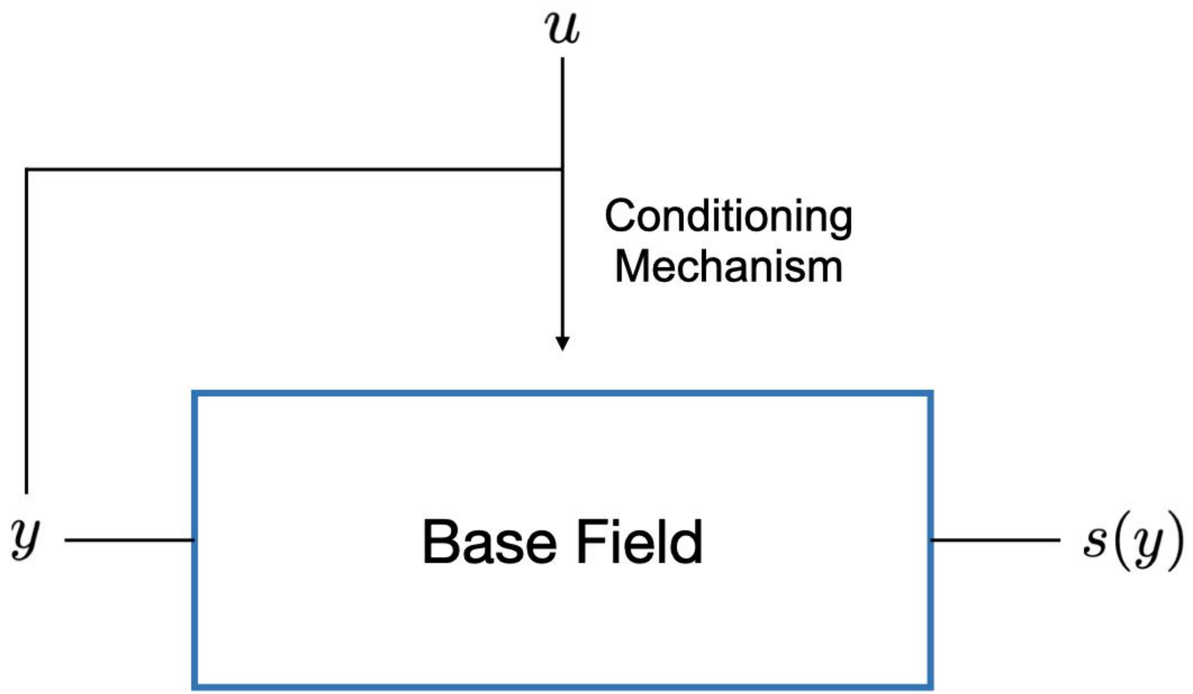
$$s(y) = f(y; z)$$



- Used in vision to incorporate additional semantic information (object class, color, texture, etc.)

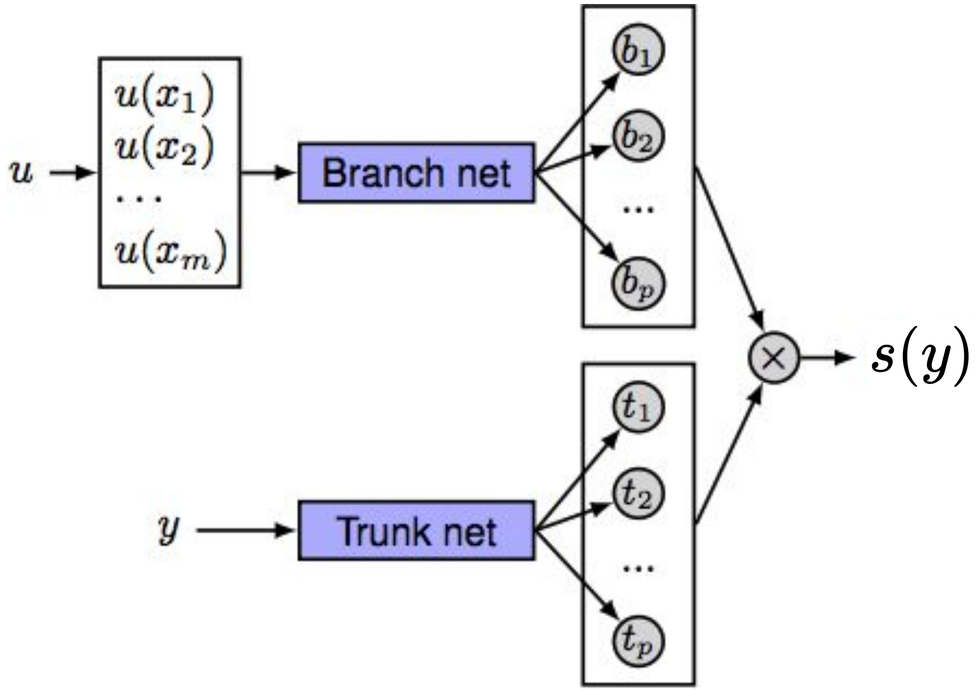
Operator learning

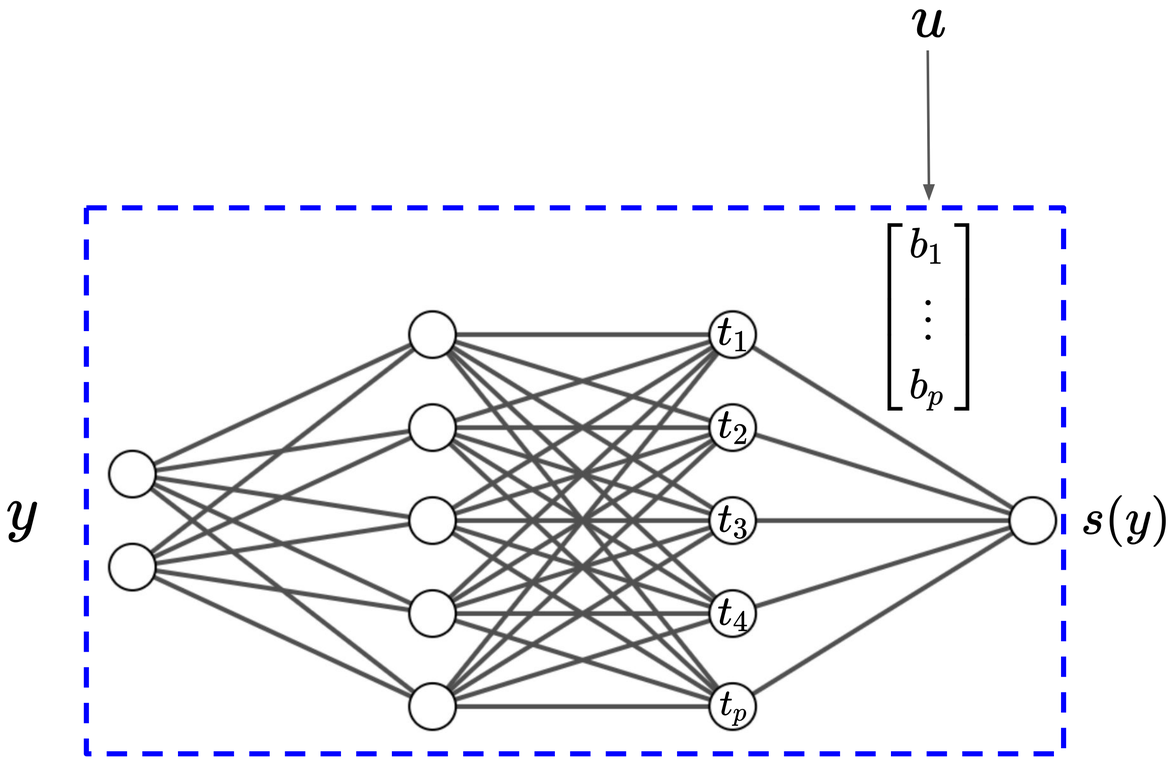
Goal is to build a queryable function ***s*** from an input function ***u***



This is what operator learning methods already do

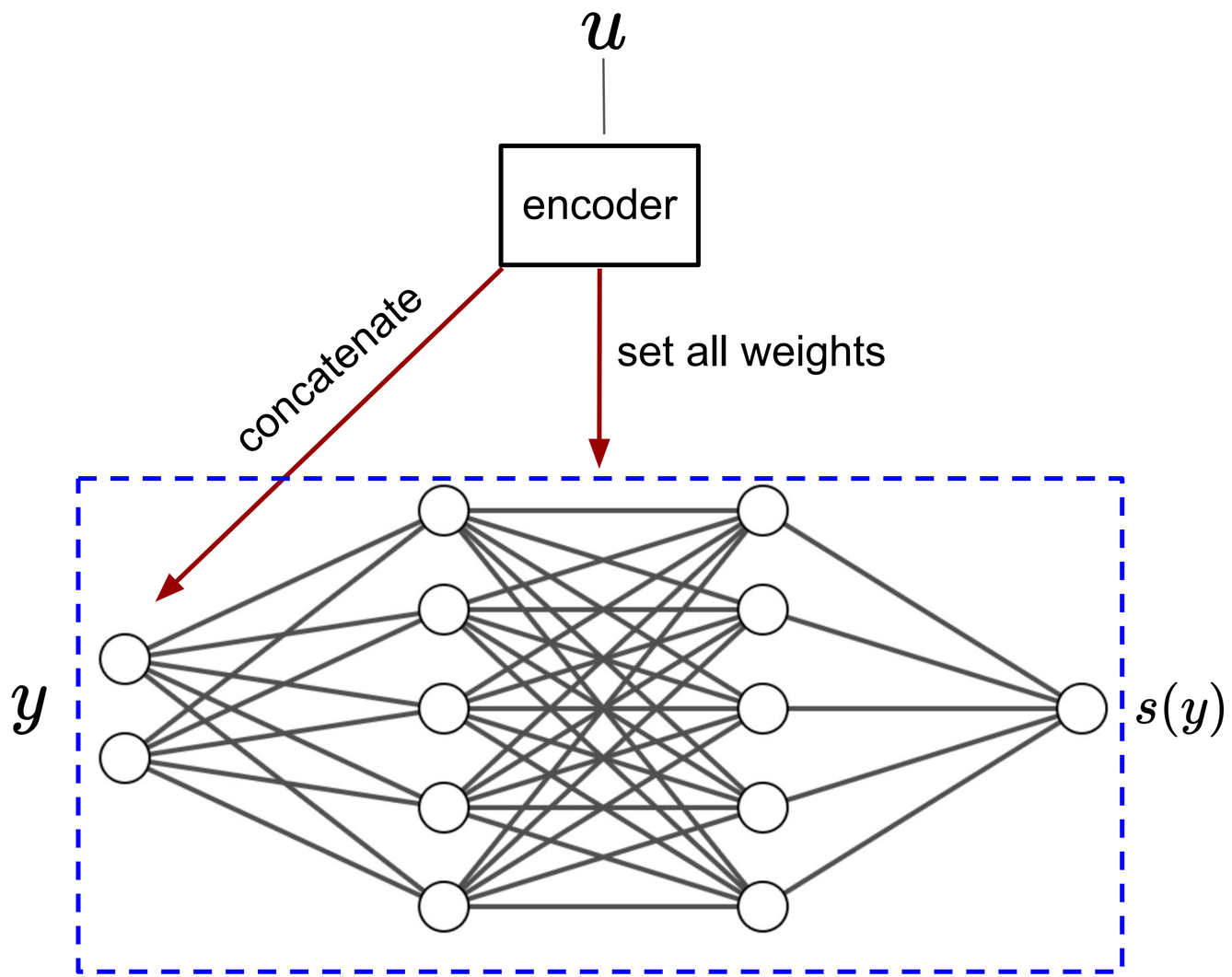
DeepONets





Lu, Lu, et al. "Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators." *Nature Machine Intelligence* 3.3 (2021): 218-229.

NOMAD



S, et al. "NOMAD: Nonlinear manifold decoders for operator learning." *Advances in Neural Information Processing Systems* 35 (2022): 5601-5613.

FNO

$$u^+(y) = \sigma\left(Wu(y) + \mathcal{F}^{-1}(K\hat{u})(y)\right)$$

$$\mathcal{F}^{-1}(K\hat{u})(y) = \sum_{j=1}^{n}(K\hat{u})_j\varphi_j(y), \quad \varphi_j(y) = e^{2\pi ijy}$$

$$\sum_{j=1}^{n}(K\hat{u})_j\varphi_j(y) = \sum_{j=1}^{n}\Re(K\hat{u})_j\cos(2\pi jy) - \sum_{i=j}^{n}\Im(K\hat{u})_j\sin(2\pi jy)$$

$$u^+(y) = \sigma\left(Wu(y) + \sum_{j=1}^{2n} z(K\hat{u})_j\gamma(y)_j\right)$$

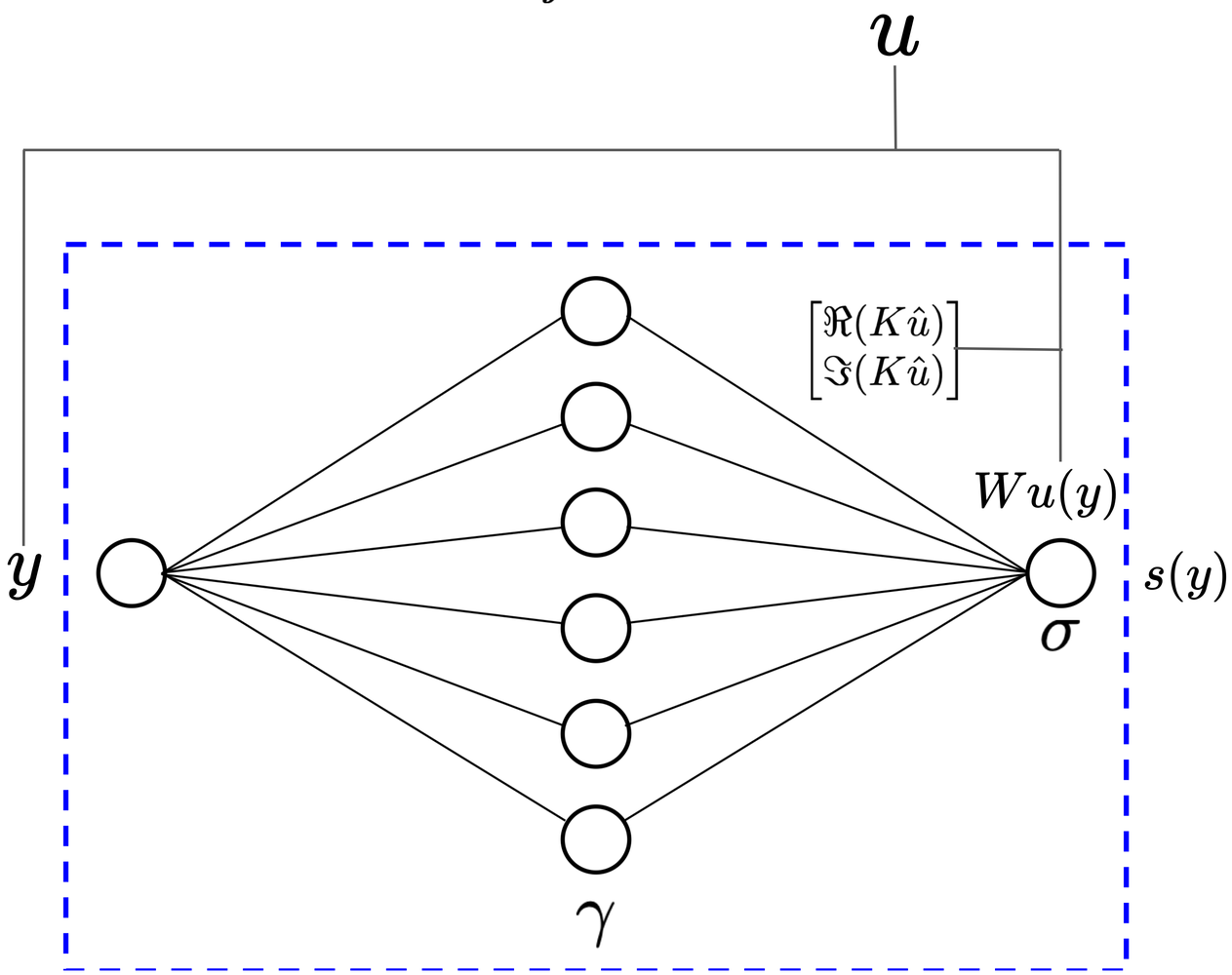
Li, Zongyi, et al. "Fourier Neural Operator for Parametric Partial Differential Equations." *International Conference on Learning Representations*. (2021)

FNO

- Global conditioning via frequency domain weights
- Local conditioning via position dependent bias
- Base field has positional encoding (Fourier features)

$$\gamma(y) = \begin{bmatrix} \cos(2\pi\langle k_1, y\rangle) \\ \vdots \\ \cos(2\pi\langle k_n, y\rangle \\ \sin(2\pi\langle k_1, y\rangle) \\ \vdots \\ \sin(2\pi\langle k_n, y\rangle) \end{bmatrix}$$

$$u^+(y) = \sigma\left(Wu(y) + \sum_{j=1}^{2n} z(K\hat{u})_j \gamma(y)_j\right)$$

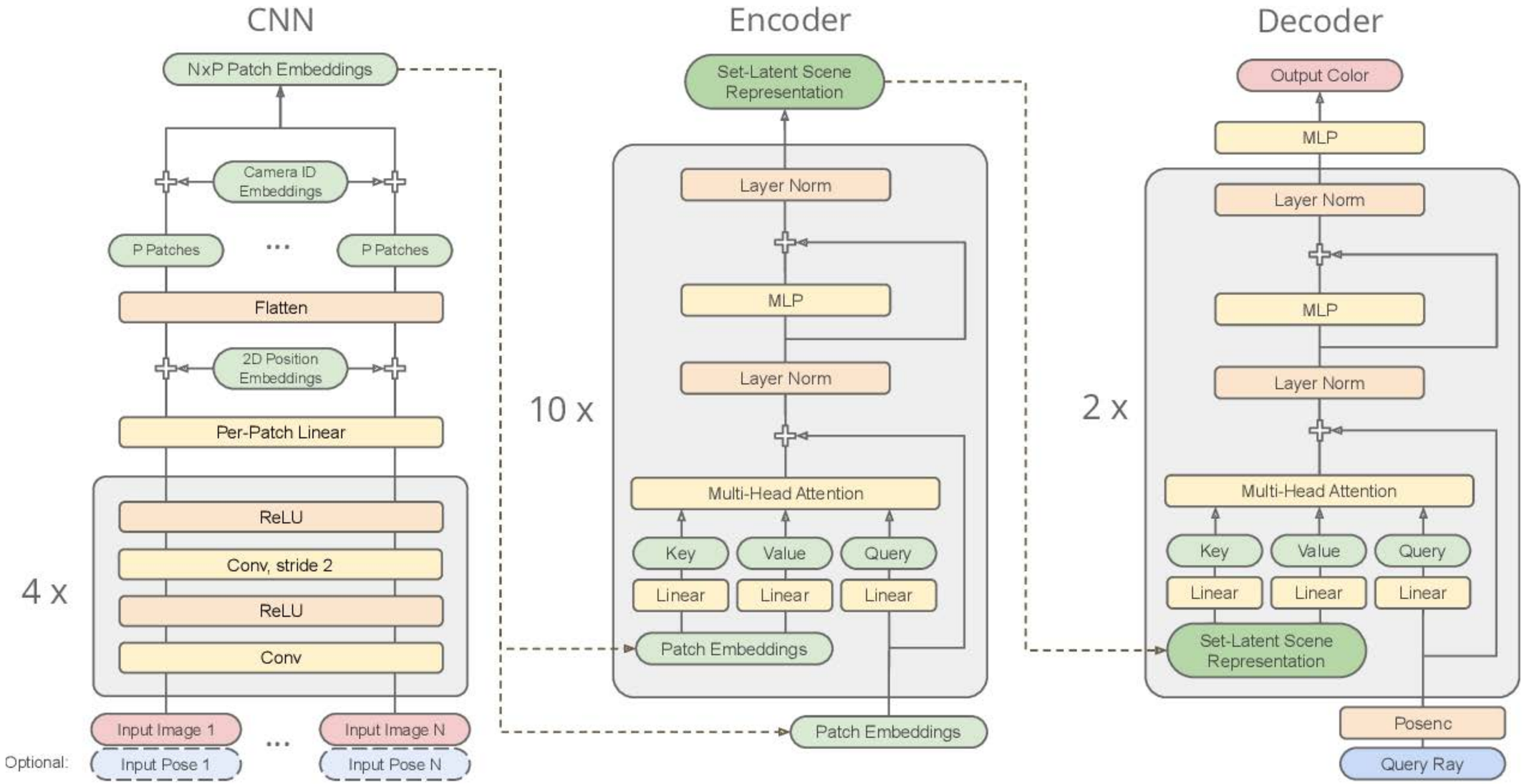


The unifying viewpoint

All operator learning architectures can be implemented as:

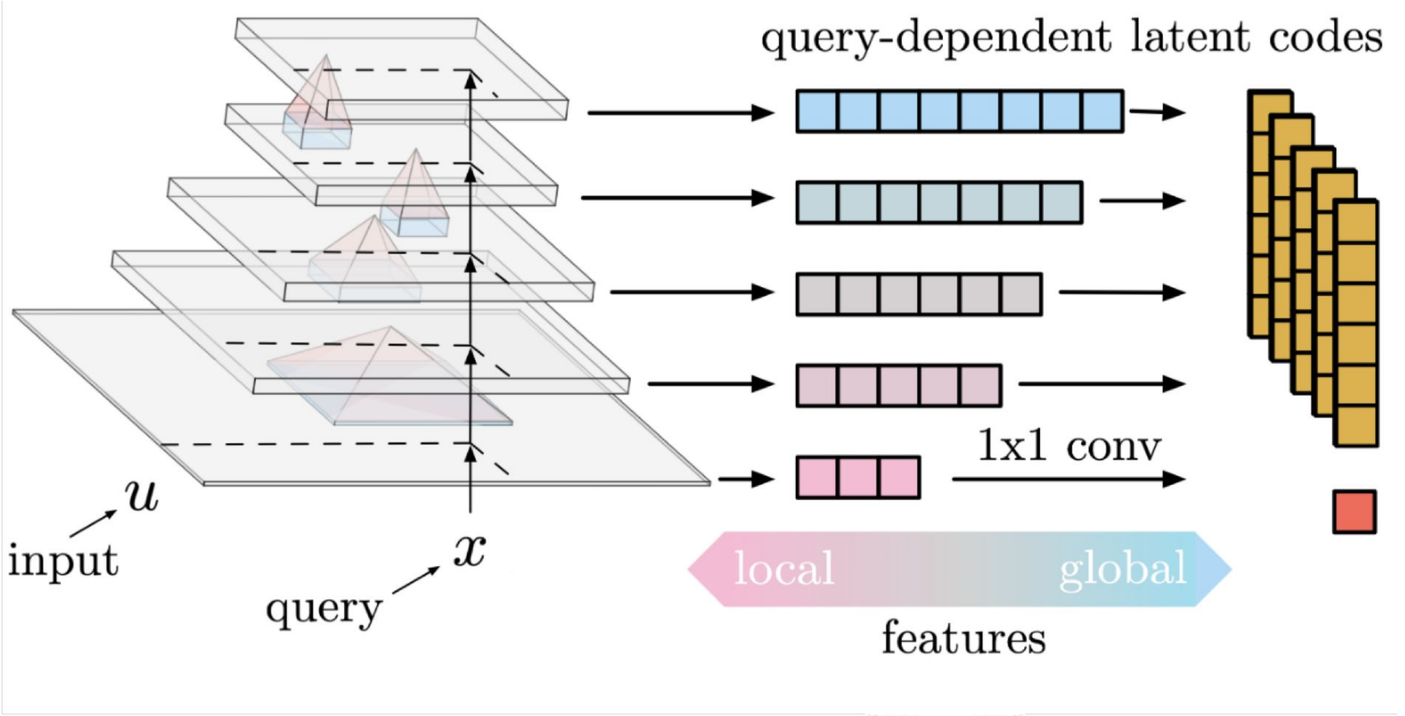
```
class CNF(nn.Module):
    @nn.compact
    def __call__(self, u, y):
        z = self.conditioner(u, y)
        s = self.basefield(z, y)
        return s
```

Why limit ourselves to DeepONets, FNOs, etc.?



Sajjadi, M. S., Meyer, H., Pot, E., Bergmann, U., Greff, K., Radwan, N., ... & Tagliasacchi, A. (2022). Scene representation transformer: Geometry-free novel view synthesis through set-latent scene representations. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 6229-6238).

Multi-scale conditioning



Pointwise information

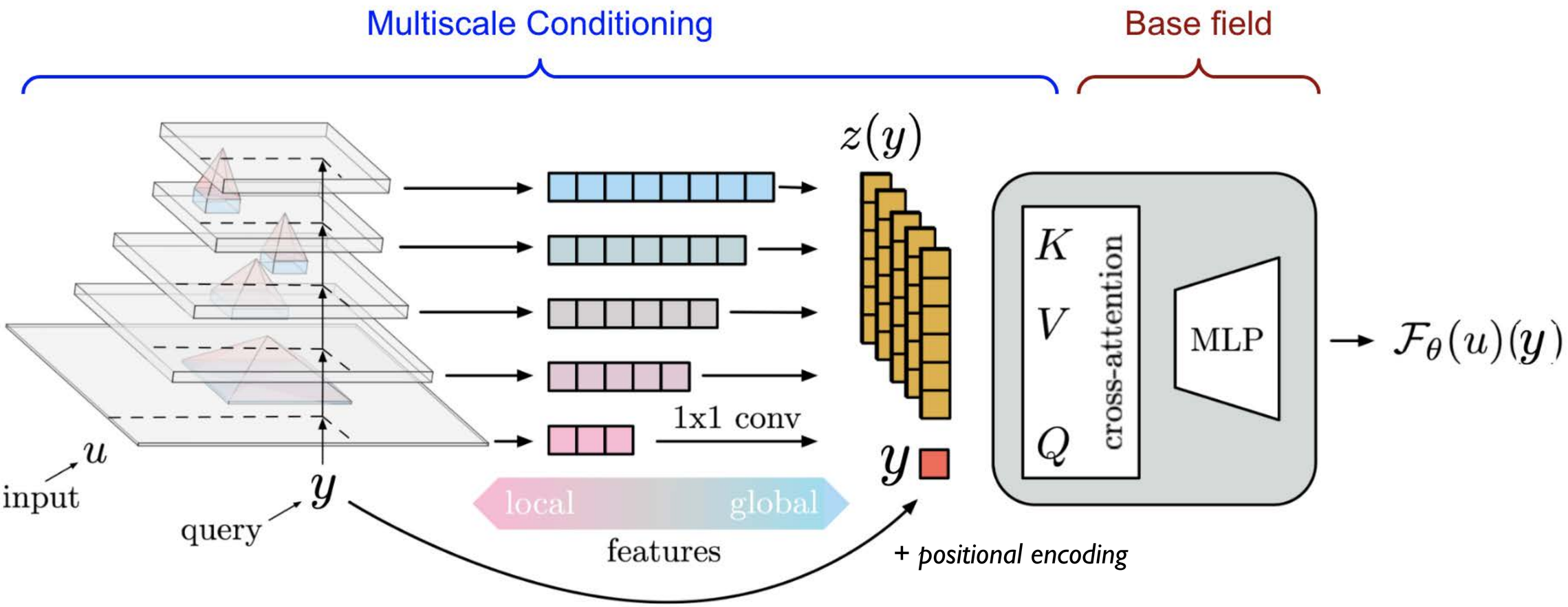
Global information

Residual connection of pointwise values (e.g. FNO)

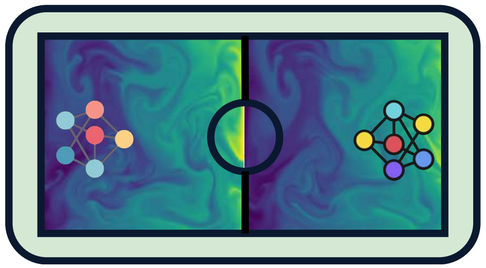
Intermediate levels of locality for conditioning?

Global latent variable (e.g. output of branch in DON)

A new multi-scale architecture



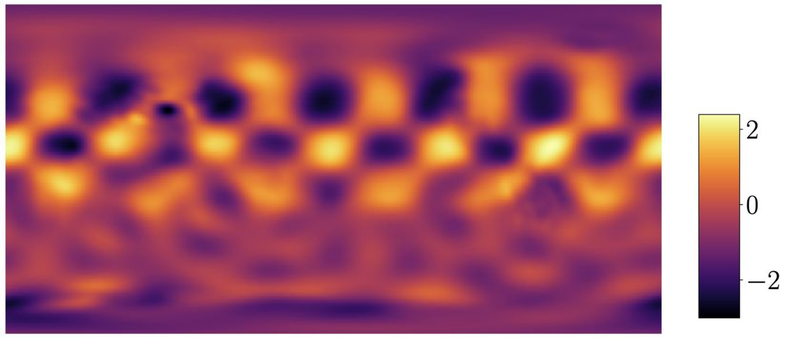
Preliminary results on challenging benchmarks



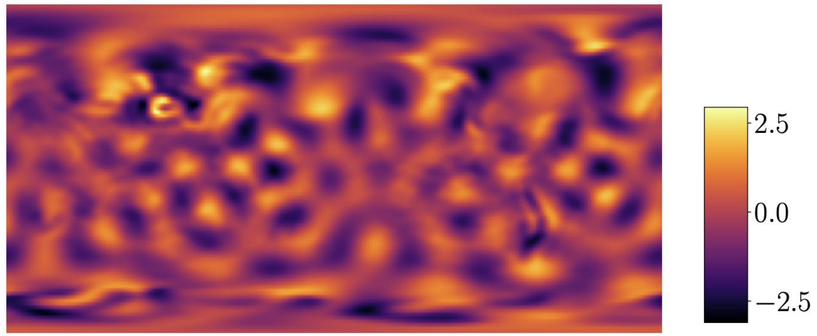
PDEArena

A modern, scalable, and easy to use PDE Surrogate Benchmarking Framework.

- 192 x 96 resolution Gaussian grid on surface of the earth
- Train on 5600 examples
- Predict pressure and vorticity 48 hours after initial condition



Pressure

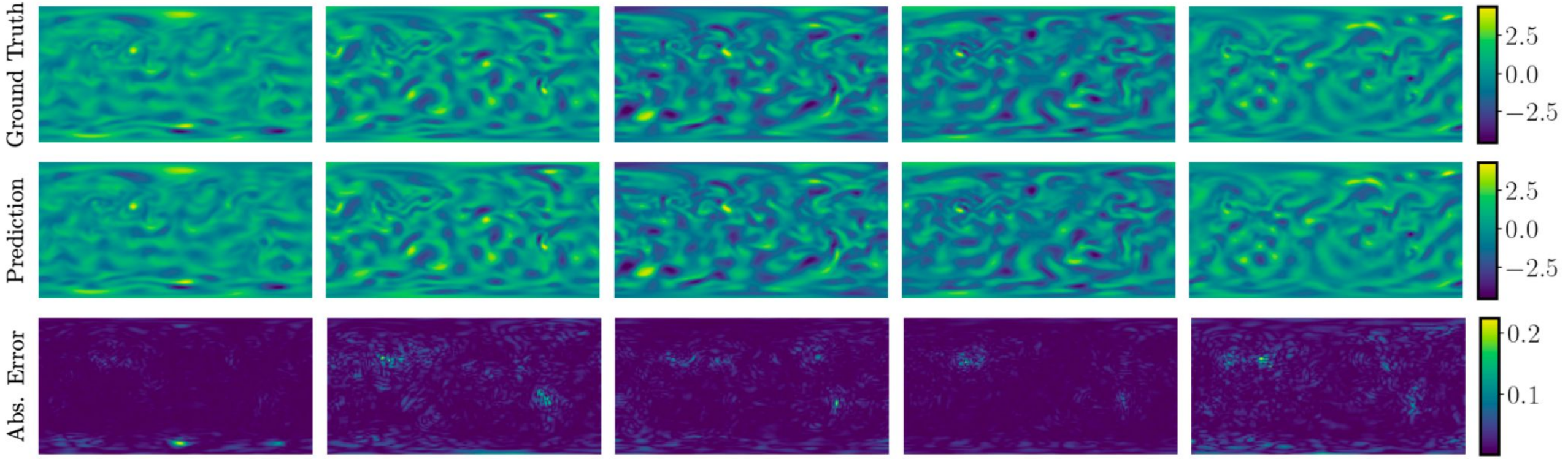


Vorticity

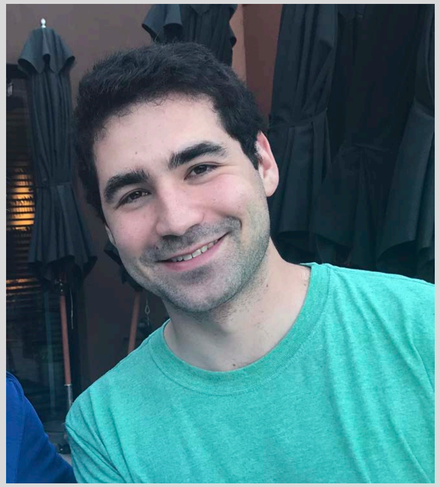
[1]https://microsoft.github.io/pdearena/
[2]https://github.com/SpeedyWeather/SpeedyWeather.jl

Preliminary results on challenging benchmarks

Model	Multi-res NF	Modern U-Net	FNO	DeepONet
Rel. L2 test error	0.0034	0.0039	0.0093	0.6223
# of trainable params	~22 million	~22 million	~67 million	~21 million



Acknowledgements



Jacob Seidman (Penn)

Sifan Wang (Penn)

- Yibo Yang (Meta)
- Georgios Kissas (Penn)
- Hanwen Wang (Penn)
- Shyam Sankaran (Penn)
- George Pappas (Penn)

Open source code & tutorials:
https://github.com/orgs/PredictiveIntelligenceLab/repositories

Thank you for your attention! Questions?

A Need for Data-Efficiency

- Current data-driven frameworks require $\mathcal{O}(10^3)$ labelled examples.
- This can be prohibitive for applications where the cost of data acquisition is high.

Data augmentation

- Multi-fidelity data-sets
- Lie groups & symmetry transformations

Inductive bias

- Equivariant layers
- Clifford layers
- Differentiable PDE solver layers

Soft constraints

- PDE residuals
- Self-supervised learning