Supervised learning in function spaces

Part I: Introduction to functional data analysis

https://github.com/PredictiveIntelligenceLab/TRIPODS_Winter_School_2022

Instructors:

- Paris Perdikaris (University of Pennsylvania, pgp@seas.upenn.edu)
- Jacob Seidman (University of Pennsylvania, seidj@sas.upenn.edu)
- Georgios Kissas (University of Pennsylvania, gkissas@seas.upenn.edu)

2022 TRIPODS Winter School & Workshop on Interplay between Artificial Intelligence (AI) and Dynamical Systems
Mathematical Institute for Data Science (MINDS)
Johns Hopkins University

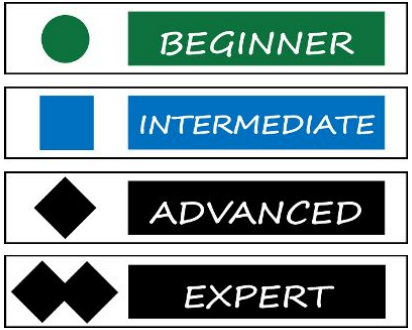
Outline of this practicum

Part I: Functional data and applications; Supervised learning in function spaces; Parametric vs non-parametric approaches; Applications highlights; Introduction to JAX.

Part II: Deep operator networks (DeepONets): Formulation, theory, implementation aspects and applications.

Part III: Fourier Neural Operators: Formulation, theory, implementation aspects and applications.

Part IV: Advanced topics: attention-based architectures; Applications to optimal control and climate modeling; Open challenges; Concluding remarks & discussion.

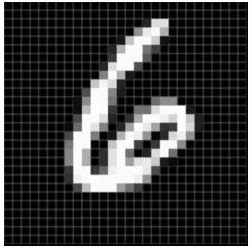


https://github.com/PredictiveIntelligenceLab/TRIPODS_Winter_School_2022

Finite Dimensional Data

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

$\mathbb{R}^{784} \ni$



$$\xrightarrow{f : \mathbb{R}^{784} \to \mathbb{R}^{10}}$$

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \in \mathbb{R}^{10}$$

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 $\{1, \ldots, n\} \to \mathbb{R}$

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

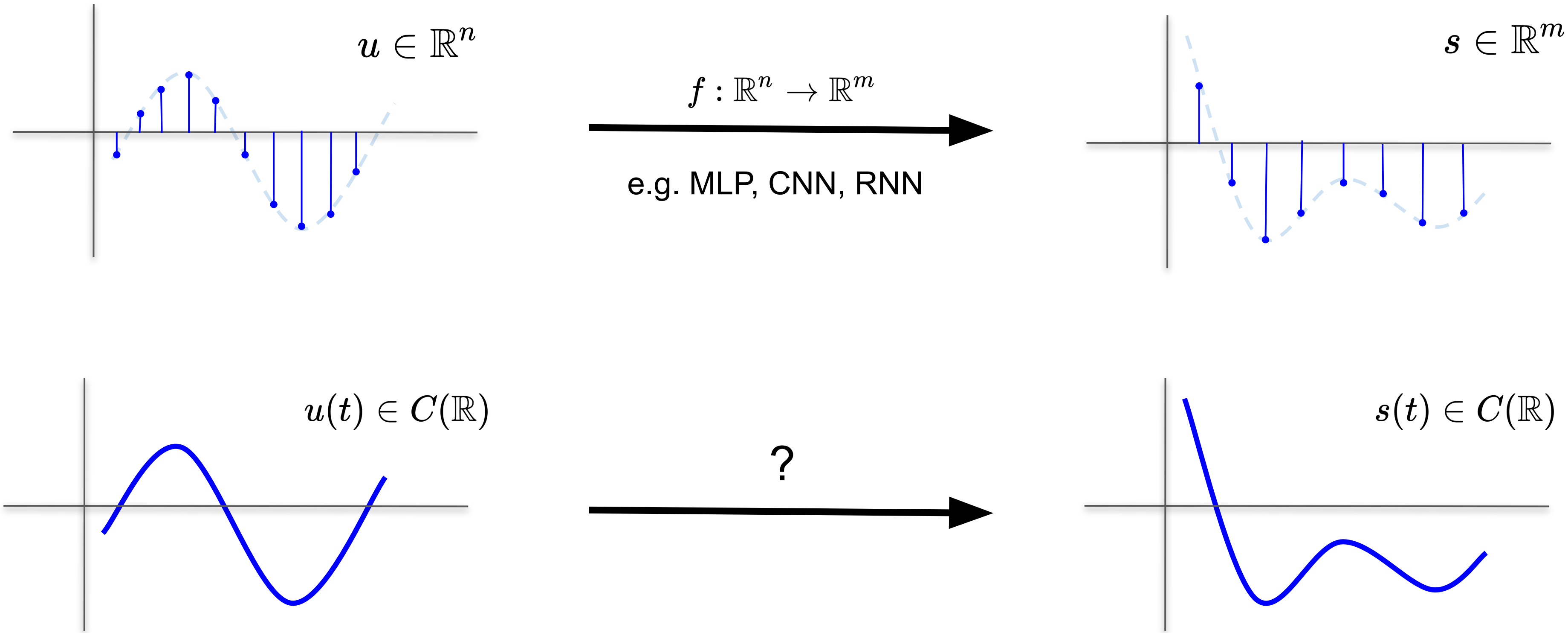
- **Example 2**: Temperature field over the earth $u : S^2 \to \mathbb{R}$

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

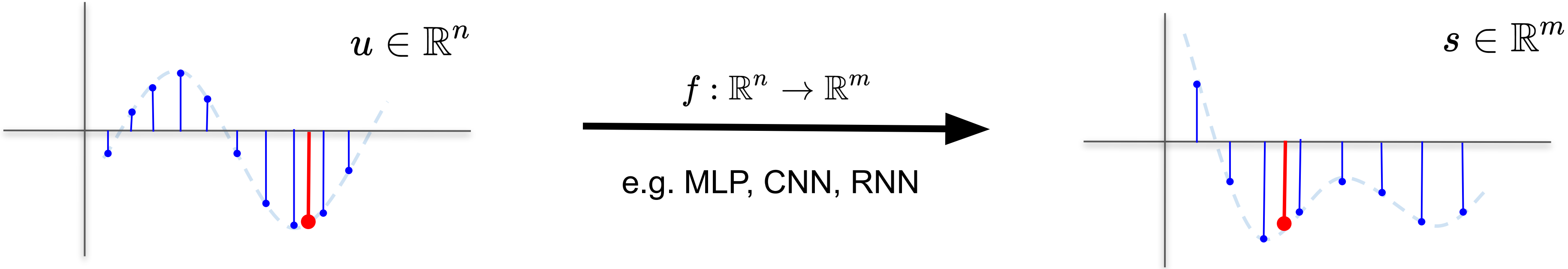
This kind of data lives in an infinite dimensional space

How to learn on function spaces?

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

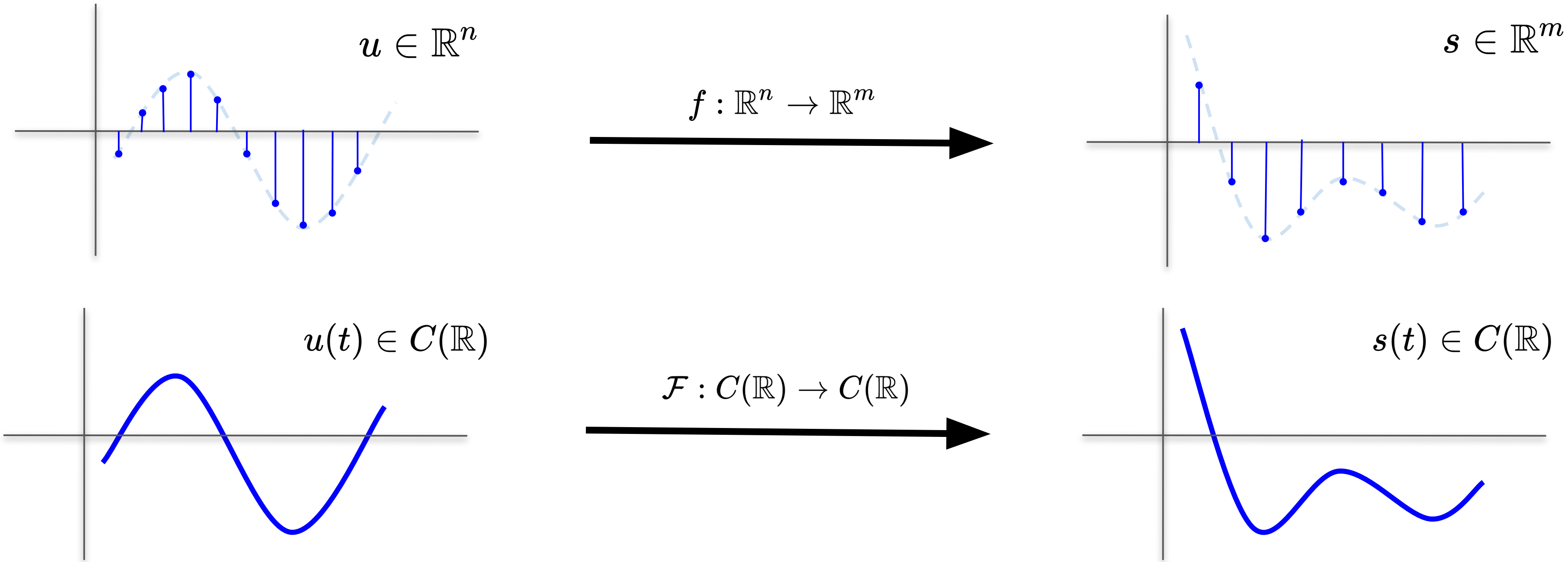


Drawbacks of Finite Dimensional Approach



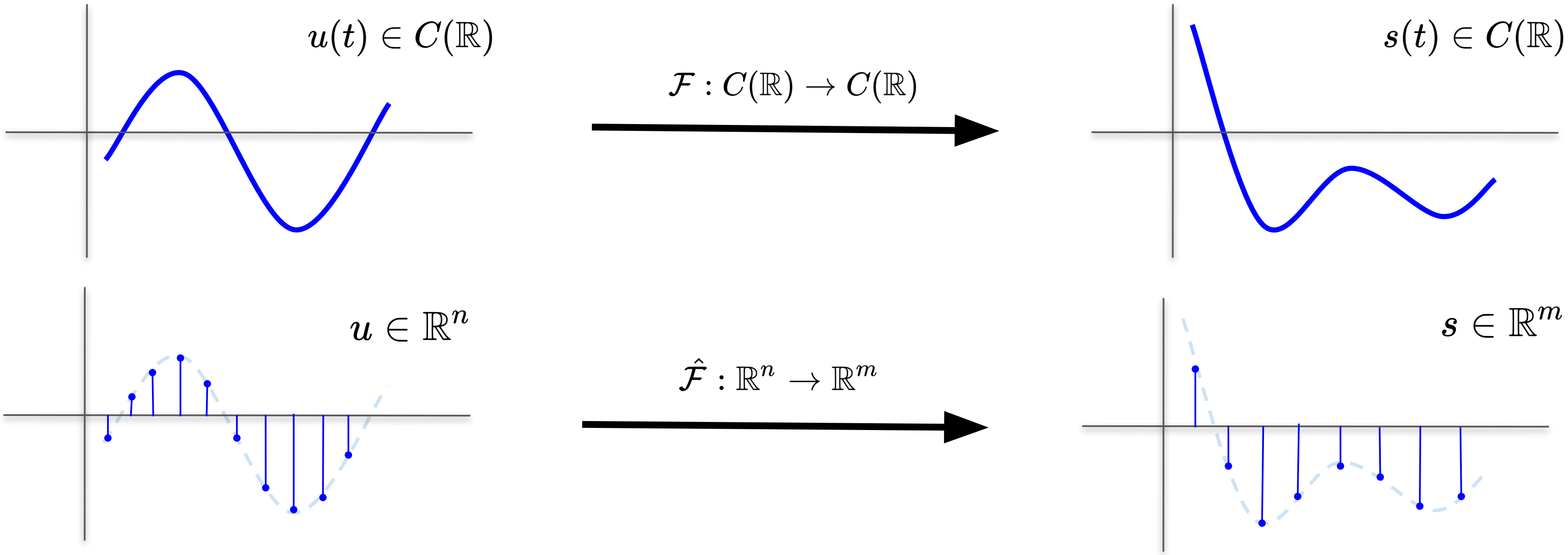
- Model is not built to accept varying numbers of measurements or new measurement locations
- We are completely constrained to our initial choice of discretization

Instead of learning function between discretizations…



Learn operator between function spaces directly

But don’t we always have to work with discrete data?



Formulating the architecture in function spaces allows different discretizations to approximate the same operator without rebuilding/retraining the model

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)

Supervised Operator Learning
Notation

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

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

$u(x)$

"input function location"

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

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

$s(y)$

"query" or "query location"

Supervised Operator Learning
Problem Formulation

- Given a dataset of N pairs of input and output functions

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

learn an operator

$$\mathcal{F} : C(\mathcal{X}, \mathbb{R}^{d_u}) \to C(\mathcal{Y}, \mathbb{R}^{d_s})$$

such that

$$\mathcal{F}(u^i) = s^i, \quad \forall i$$

Operator Learning: Kernel Methods

- RKHS methods can be extended to learning operators between arbitrary Banach spaces $\mathcal{U} \to \mathcal{S}$

$$k : \mathcal{U} \times \mathcal{U} \to \mathcal{L}(\mathcal{S}, \mathcal{S})$$

- Analogous representer theorem as in scalar/finite-dimensional case - look at operators of the form

$$\mathcal{F}(u) = \sum_{i=1}^{N} k(u^i, u)\eta^i, \quad \eta^i \in \mathcal{S}$$

Kadri, Hachem, et al. "Operator-valued kernels for learning from functional response data." *Journal of Machine Learning Research* 17.20 (2016): 1-54.

Operator Learning: Parametric Methods

- We will focus in detail on the following three recent parametric approaches

- **DeepONets (Part 2)**
  - 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.

- **Neural Operators (Part 3)**
  - Kovachki, Nikola, et al. "Neural operator: Learning maps between function spaces." *arXiv preprint arXiv:2108.08481* (2021).

- **LOCA: Learning Operators with Coupled Attention (Part 4)**
  - Kissas, Georgios, et al. "Learning Operators with Coupled Attention." *arXiv preprint arXiv:2201.01032* (2022).

NEXT UP

- Introduction to JAX
  - Basics of the language and example implementations of simple Deep Learning Models

THEN

- Some recent operator learning architectures and their JAX implementations