Supervised Learning

ADVANCED DEEP LEARNING FOR PHYSICS

Teaser Example from PBDL

- DL extremely powerful…
- … but sometimes surprisingly wrong. Solve map: $y^2 \rightarrow x$

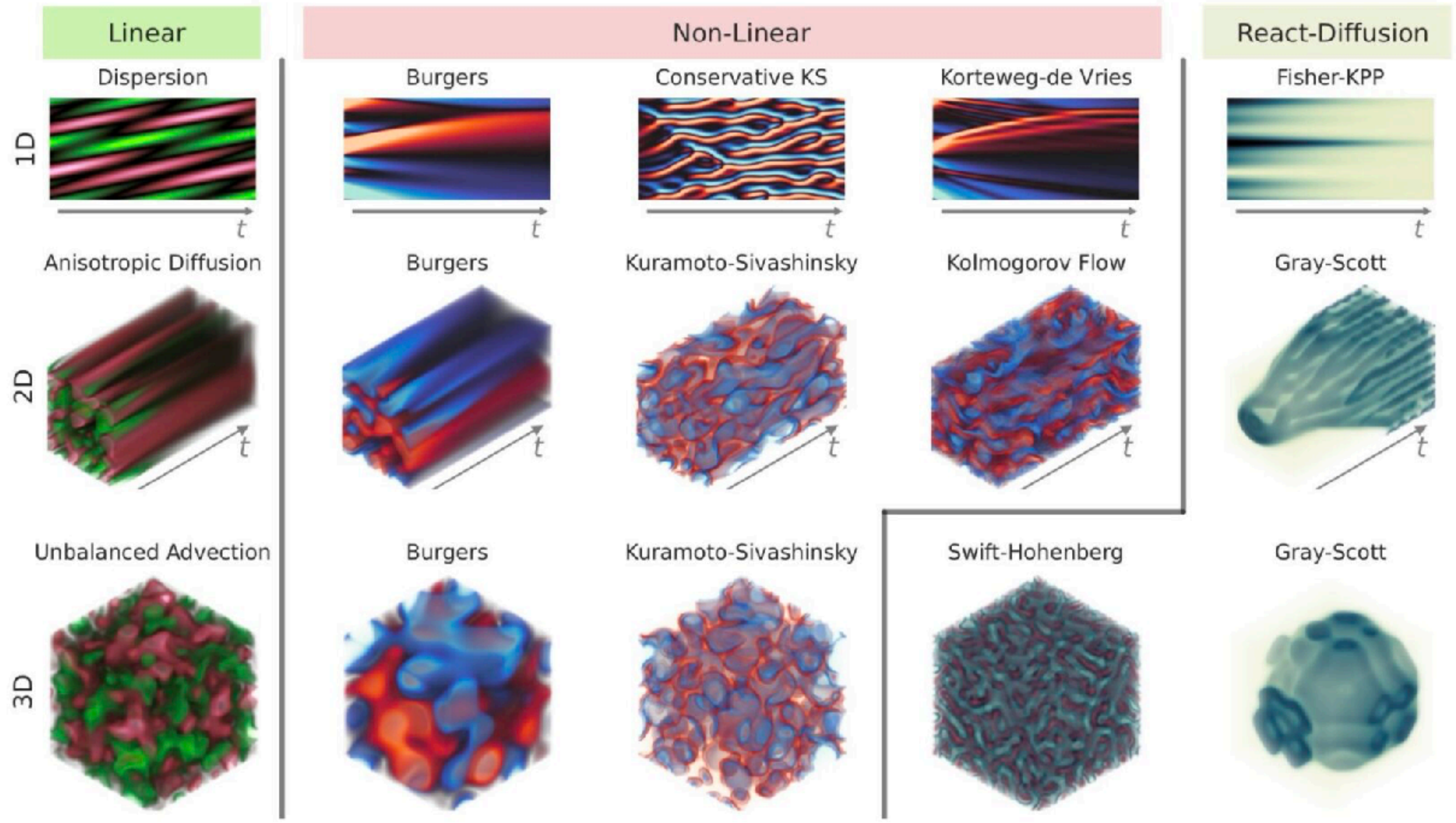
https://colab.research.google.com/github/tum-pbs/pbdl-book/blob/main/intro-teaser.ipynb

Model Equations

Model Equations

Basic PDEs

- Diffusion
- Burgers
- Navier-Stokes

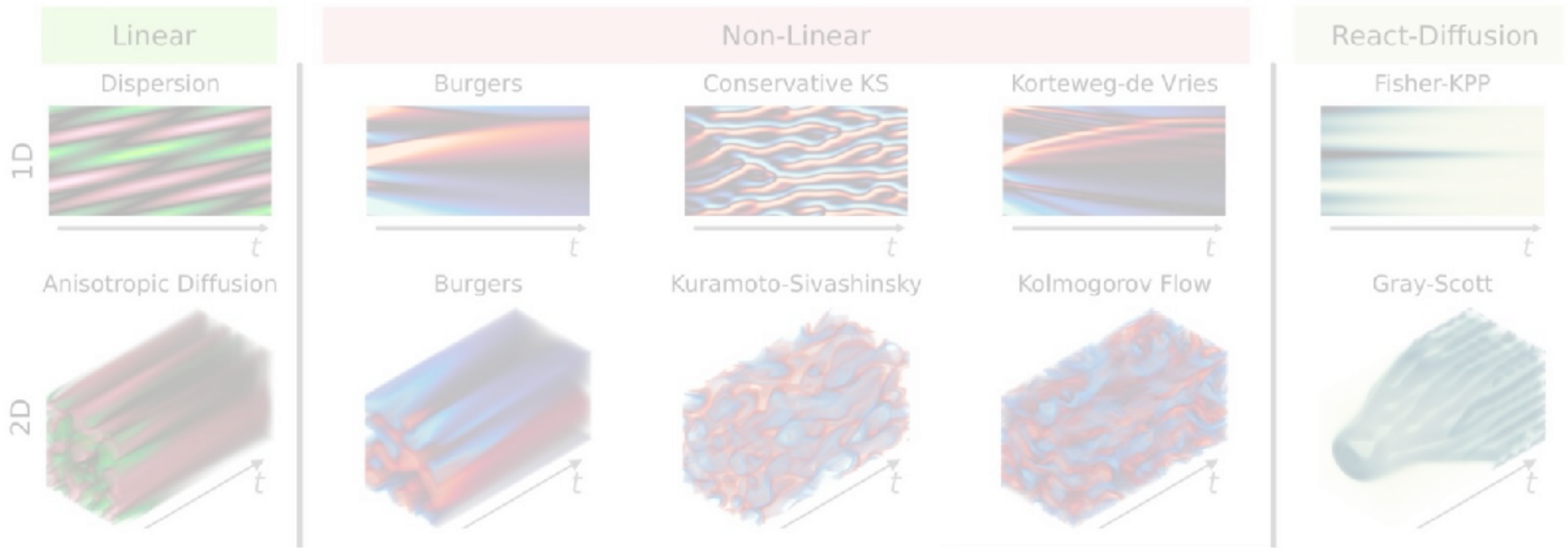


https://github.com/tum-pbs/apebench

Model Equations

Basic PDEs

- Diffusion
- Burgers
- Navier-Stokes



$$\frac{\partial u}{\partial t} - \alpha \nabla^2 u = 0$$

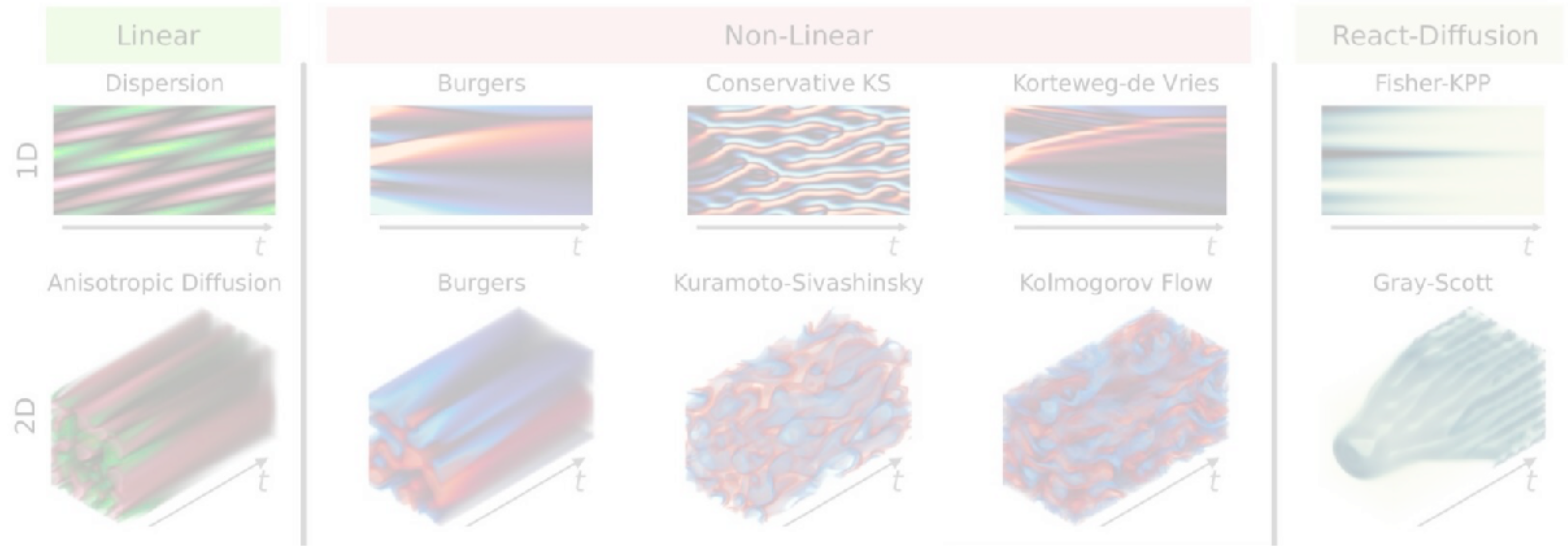
Diffusion constant α

https://github.com/tum-pbs/apebench

Model Equations

Basic PDEs

- Diffusion
- Burgers (in 2D)
- Navier-Stokes



$$\frac{\partial u_x}{\partial t} + \mathbf{u} \cdot \nabla u_x = \nu \nabla \cdot \nabla u_x$$

$$\frac{\partial u_y}{\partial t} + \mathbf{u} \cdot \nabla u_y = \nu \nabla \cdot \nabla u_y$$

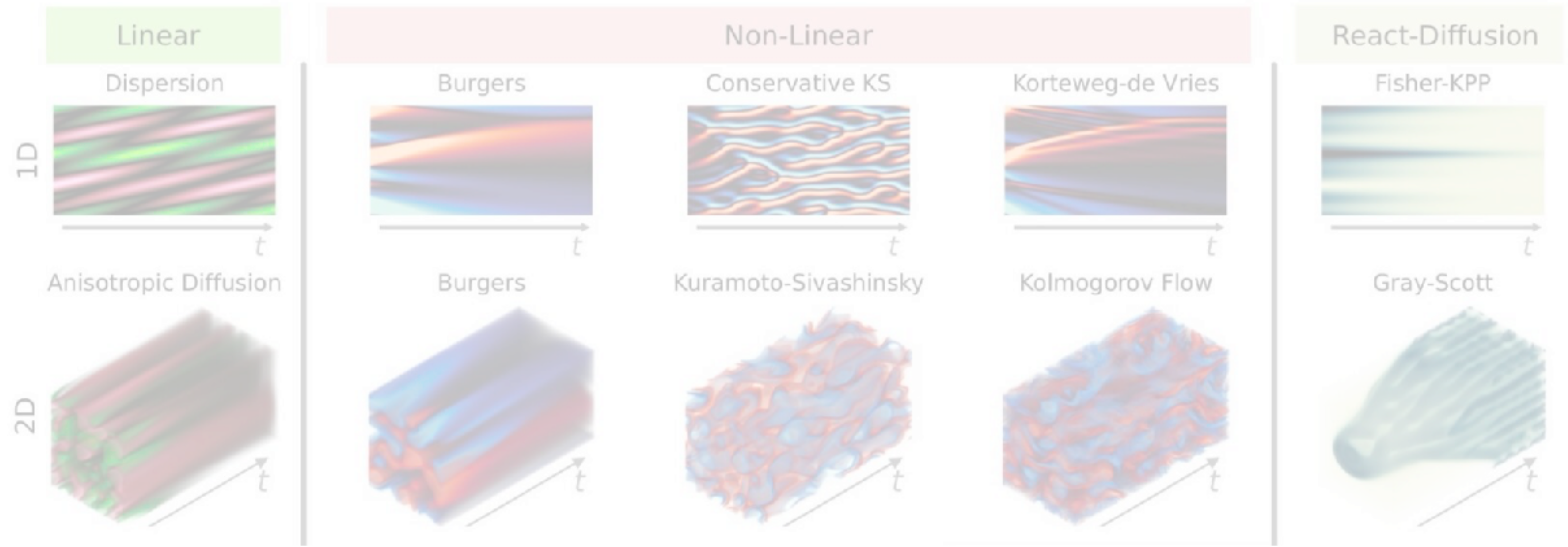
Kinematic Viscosity ν

https://github.com/tum-pbs/apebench

Model Equations

Basic PDEs

- Diffusion
- Burgers
- Navier-Stokes (2D)



$$\frac{\partial u_x}{\partial t} + \mathbf{u} \cdot \nabla u_x = -\frac{1}{\rho} \nabla p + \nu \nabla \cdot \nabla u_x + g_x$$

$$\frac{\partial u_y}{\partial t} + \mathbf{u} \cdot \nabla u_y = -\frac{1}{\rho} \nabla p + \nu \nabla \cdot \nabla u_y + g_y$$

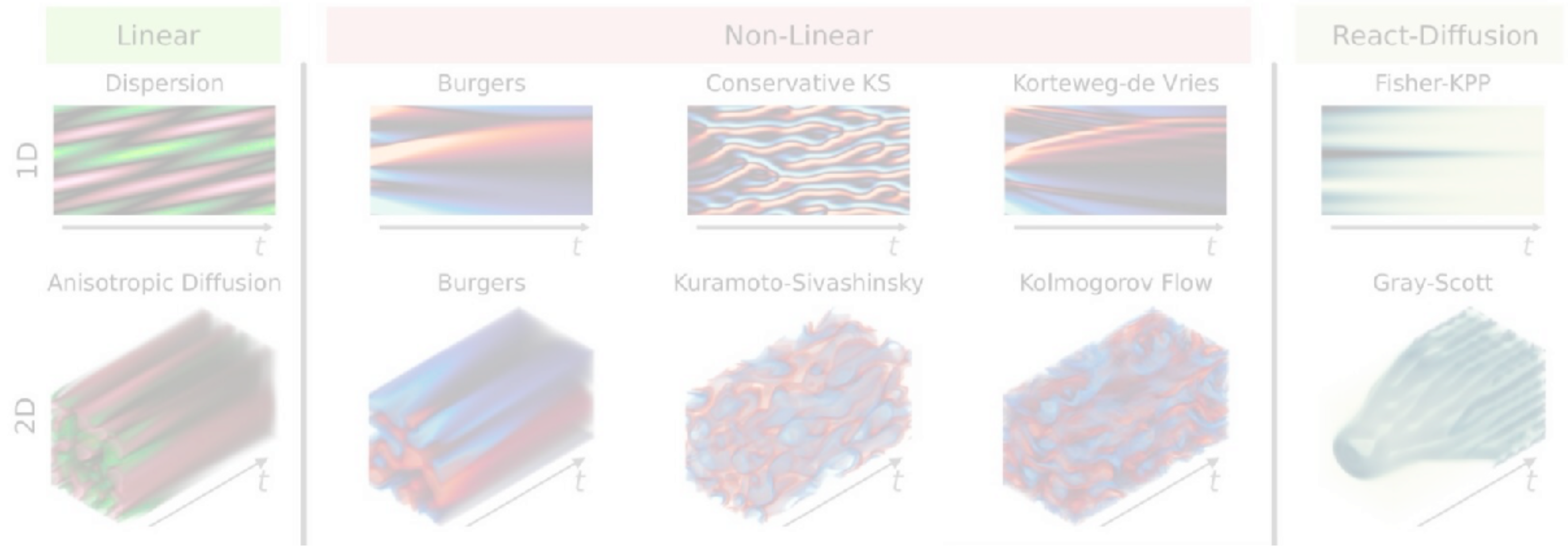
$$\text{s.t.} \quad \nabla \cdot \mathbf{u} = 0$$

https://github.com/tum-pbs/apebench

Model Equations

Basic PDEs

- Diffusion
- Burgers
- Navier-Stokes (2D)



*Distinguish **forward** and **inverse** problems*

Forward: initial & boundary conditions, solve from time t_0 to end time

Inverse: from data/observations solve for state (e.g., $\mathbf{u}(t_0)$) or parameter (e.g., viscosity ν)

Supervised Learning - The Basics

Notation

Deep Learning Basics

- Approximate unknown function $f^*(x) = y^*$
- Star super-script * denotes ground truth (often intractable)
- Find approximation $f(x)$ over training data set with (x_i, y_i^*) pairs
- Minimizing error $e(x, y)$
- In the simplest case L^2: arg min$_\theta \,|\, f(x;\theta) - y^* \,|_2^2$
- Solve non-linear minimization problem with gradient based optimizer (Adam)

Types of Machine Learning

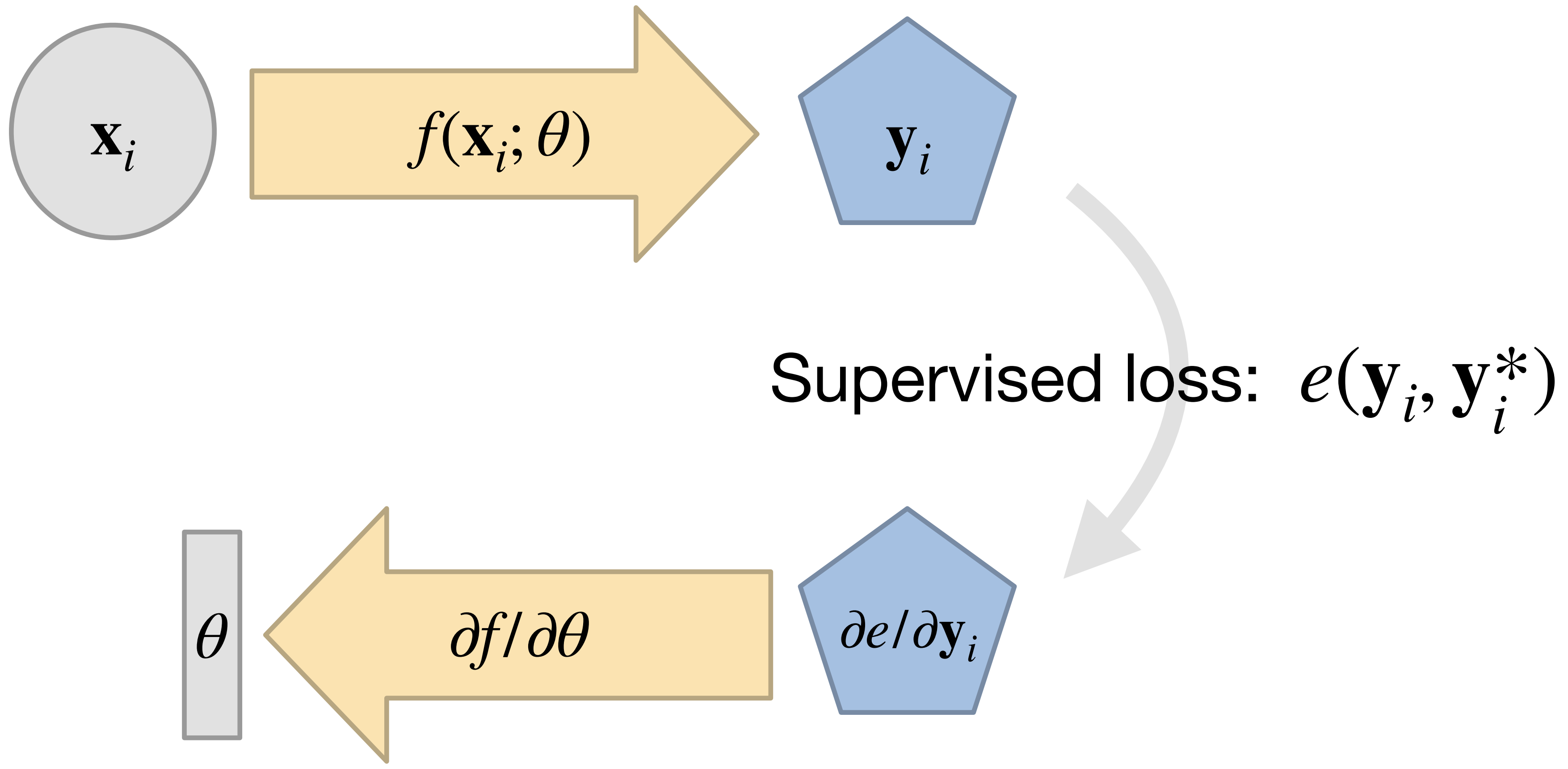
Traditional Viewpoints

- Traditional ML distinction: *classification* VS *regression*
  - In the following: regression, $f(x) = y$, with x, y continuous functions
- Later on ***physics regression*** $\mathscr{P}\left(f(x)\right) = y$ **:**

  physical model $\mathscr{P}$ combined with regression problem; typically involves highly non-linear functions that cause uneven scaling

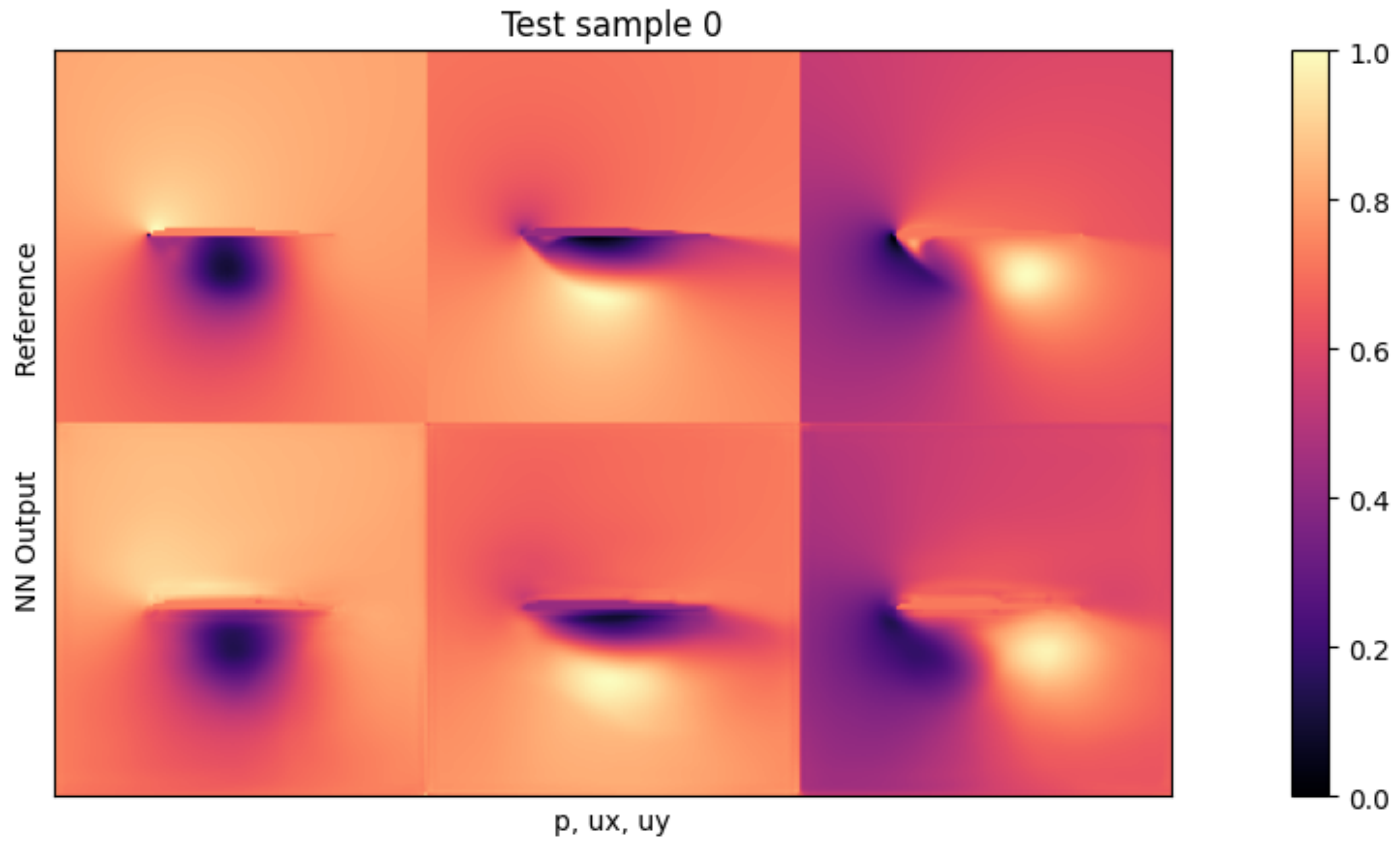
Re-cap Supervised Training

- Definition *Supervised Training* := purely data-driven, pre-computed x, y, with simple loss (e.g. L^2)
- Fully data-driven
  - Physical model not taken into account
  - Sub-optimal accuracy and generalization
- Exactly as before: *arg min*$_\theta |f(x;\theta) - y^*|_2^2$



- 😍 Beautiful from an ML perspective: *no "inductive biases" needed*
- 😱 Horrible from a computational perspective: *no existing knowledge used*

Supervised Training



https://colab.research.google.com/github/tum-pbs/pbdl-book/blob/main/supervised-airfoils.ipynb

Training Surrogate Models

Supervised Training for Time Integration

- Precompute time series data: given states over time $[u^0, u^1, \ldots, u^N]$
- Consider batches representing a single time step forward $x := u^t; \quad y* = u^{t+1}$
- Then, just like before: *arg min*$_\theta \, | f(x;\theta) - y* |_2^2$
- Given u^0 approximate any state u^i by *i* recurrent / autoregressive evaluations of $f(\,\cdot\,)$

Training Surrogate Models

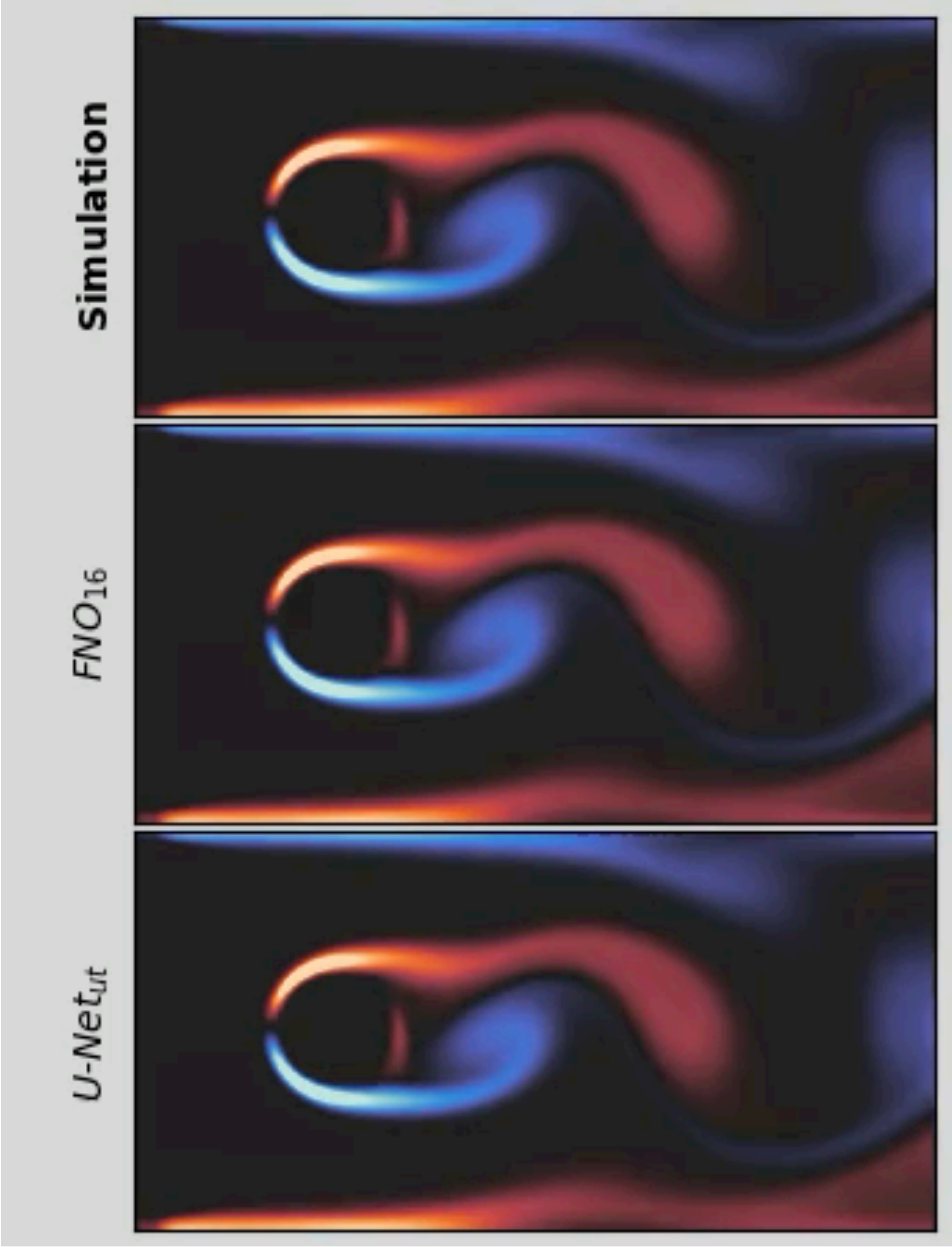
Recurrent Evaluation

- Per step approximation errors will grow; dynamical systems perspective: reference states u^* evolves on *attractor* of PDE $\mathscr{P}$, with $u^*_{t+1} = \mathscr{P}(u^*_t)$; $u^*_{t>t_0} \in A_{\mathscr{P}}$
- Attractor of NN f doesn't match the one from $\mathscr{P}$: $u_{t+1} = f(u_t)$; $A_f \neq A_{\mathscr{P}}$
- Classic "data shift" problem from ML , causes instabilities!

Training Surrogate Models

Growing Errors - Example

- Simple Navier Stokes “wake flow”
- Here: all models are quite good
- “Drift” from G.T. is very slow
- (Not shown: eventual complete blow up)



Training Surrogate Models

Outlook

- Obvious fix: include time evolution in training to improve attractor, ideally include solver
- Train with **unrolling** , more details later on…

Supervised Training

Best Practices

- *Always* start here
- *Always* start with overfitting 1 data point
- *Always* check number of NN parameters
- *Always* adjust hyper parameters at this stage
- … then slowly introduce more data and beautiful physics models

Supervised Training

Best Practices

✅ fast, reliable (builds on established DL methods)

✅ Great starting point

❌ Sub-optimal performance, accuracy and generalization.

❌ Fundamental problems in multi-modal settings

❌ Requires precomputed data (data shift problem)

Supervised Learning

ADVANCED DEEP LEARNING FOR PHYSICS

Lectures / Exercises

- Ex1: Phiflow done
- Ex2: First “real” one coming up
- Feedback session again on Monday
- Lecture slides updated just now
- Links to be fixed

Supervised Training

Best Practices

- *Always* start here
- *Always* start with overfitting 1 data point
- *Always* check number of NN parameters
- *Always* adjust hyper parameters at this stage
- … then slowly introduce more data and beautiful physics models

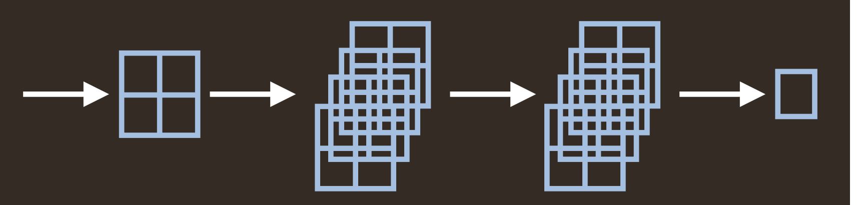
Supervised Learning

ADVANCED DEEP LEARNING FOR PHYSICS

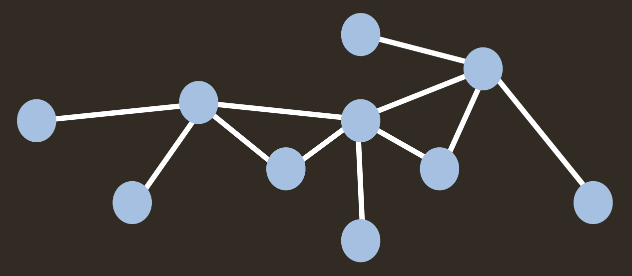
Neural Network Architectures

Overview

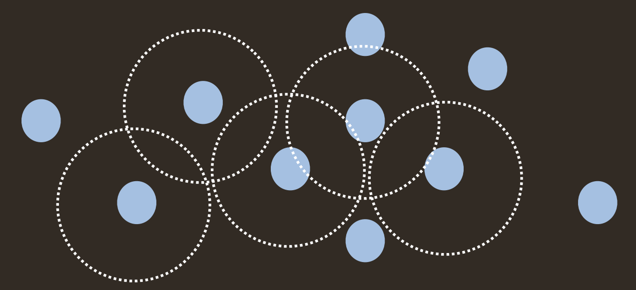
Categorization



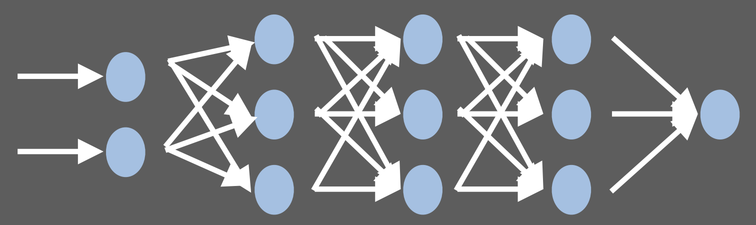
Structured (CNN)



Unstructured (GNN)



Particle-based (CConv)



Non-spatial (MLP)

- Regular spacing on a grid (*structured*)
- Irregular arrangement (*unstructured*)
- Irregular positions without connectivity (*particles*)
- [No spatial arrangement at all]

Overview

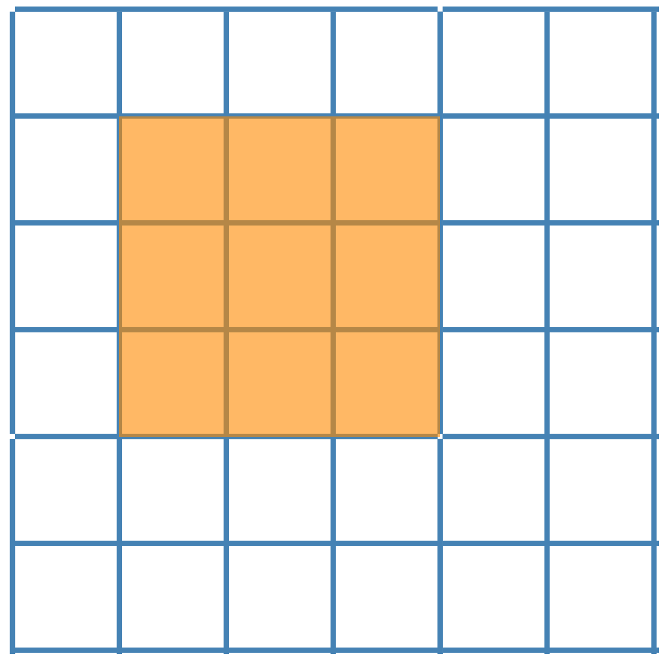
Receptive Fields 👀

- Distinguish Local vs Global interactions
- Similar to hyperbolic PDEs (e.g. waves) and parabolic/elliptic PDEs (e.g. heat)
- No surprise: capabilities of NN should match requirements of PDE… 🙄
- MLPs: trivially global, but scale badly with $\mathcal{O}(N^2)$

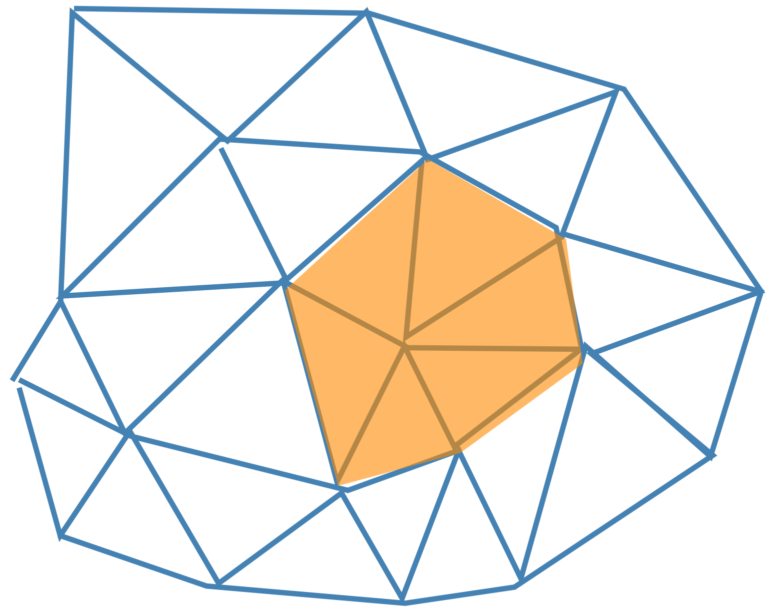
Convolutions & Message Passing

Inherently Local

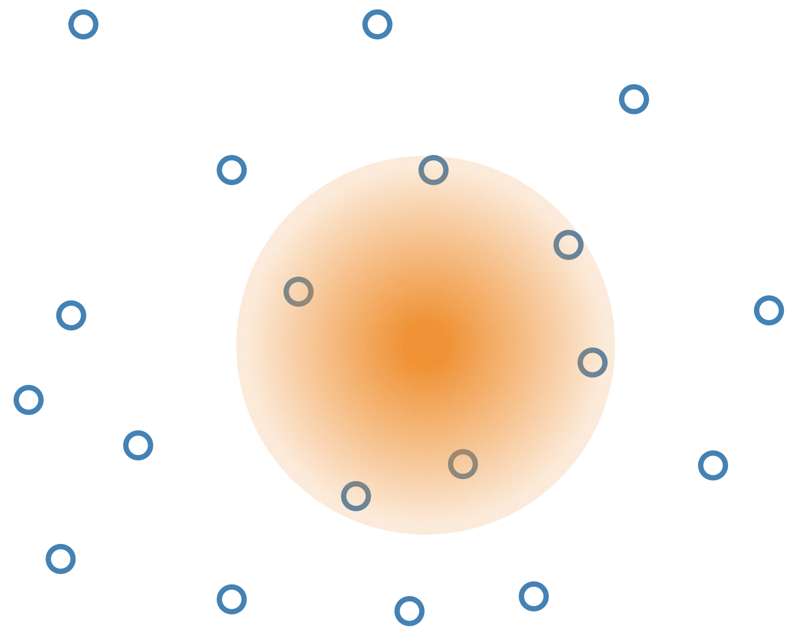
- Grids, graphs, particles
- Stencil operations



Regular convolution

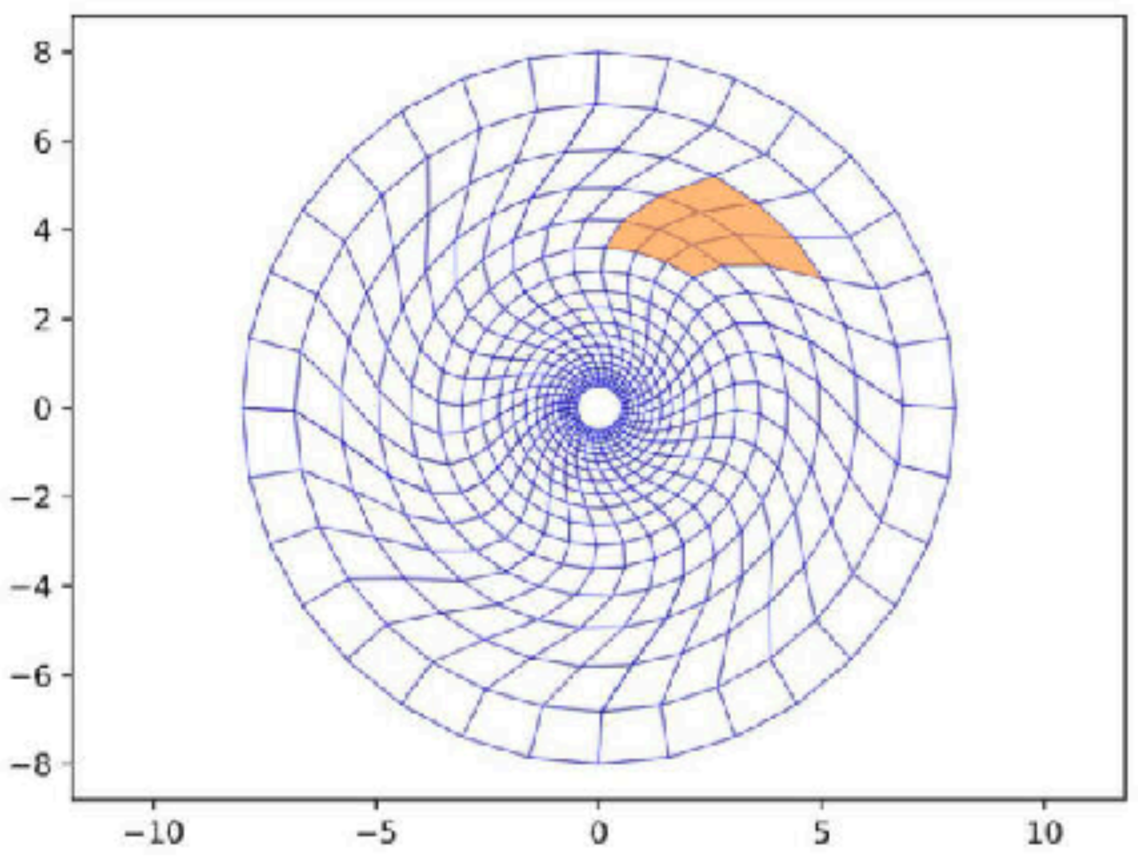


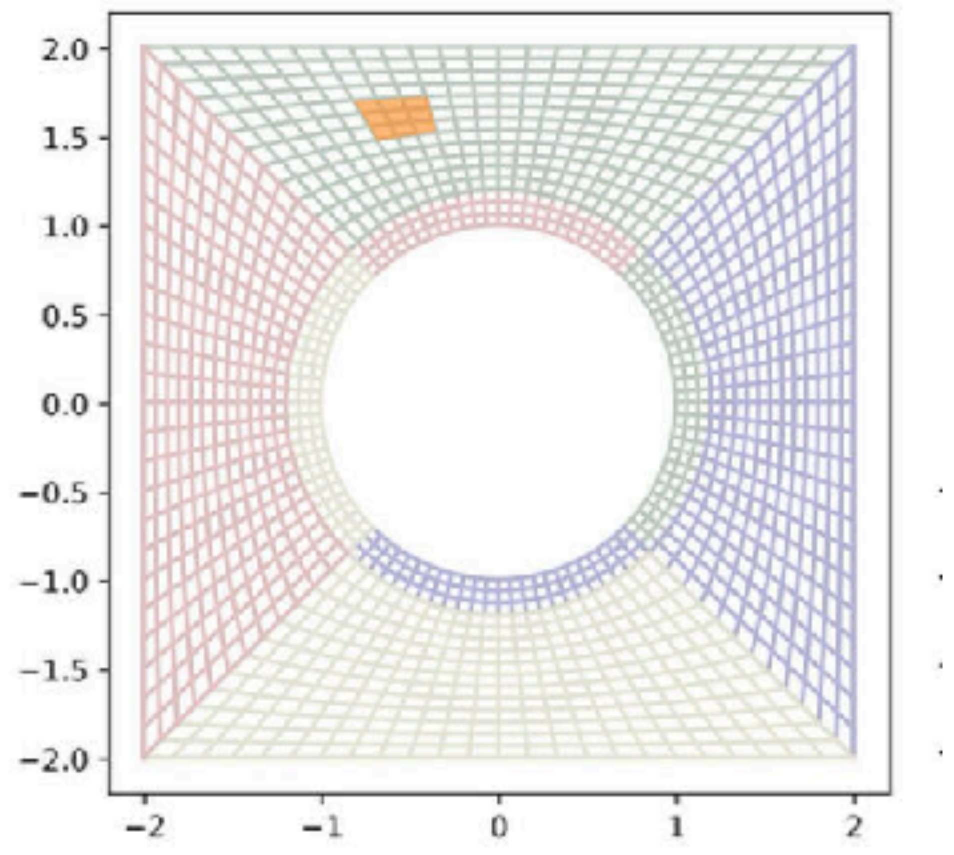
Message passing /
Graph convolution



Learned Kernel /
Continuous convolution

- Inductive bias on grids (simplifies implementation) , same concepts on graphs
- Can be non-trivial

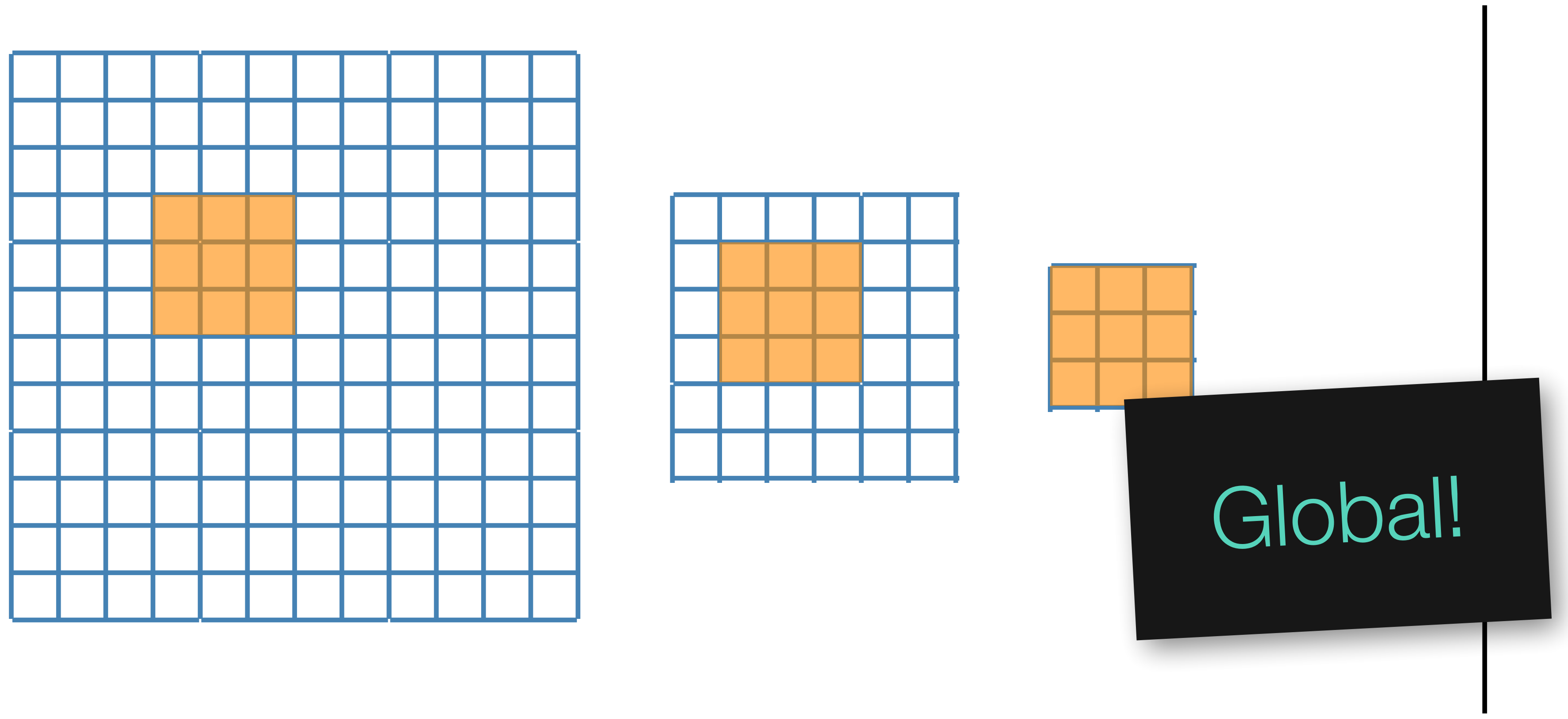




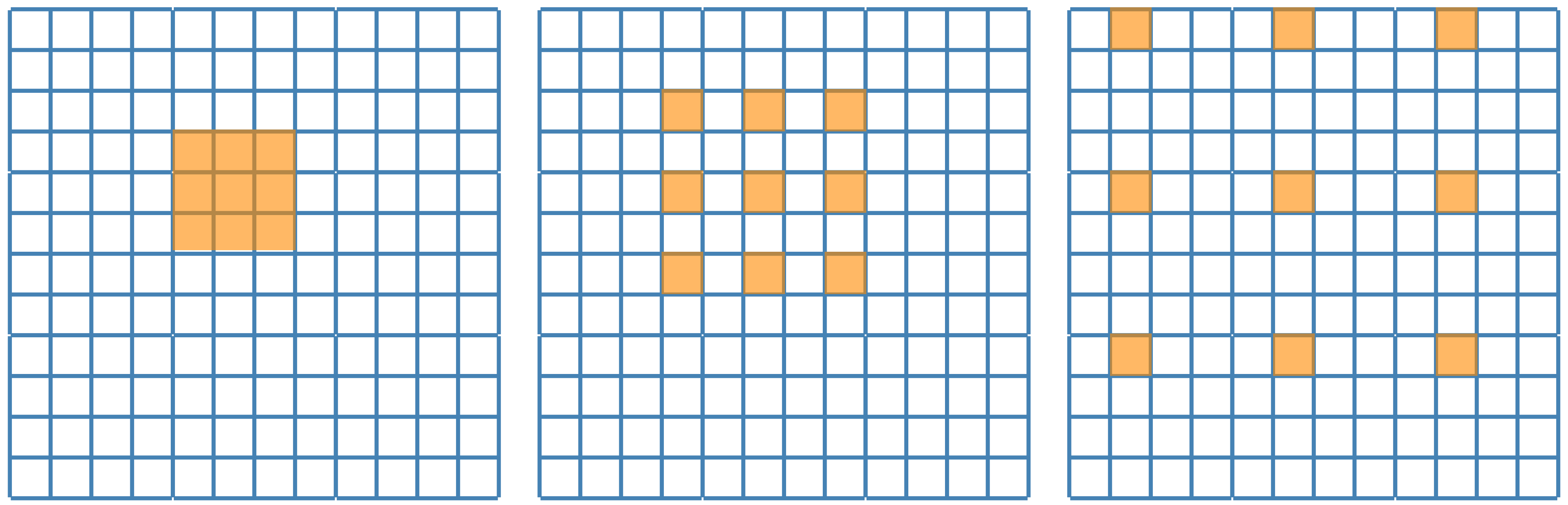
Receptive Fields

Larger Receptive Fields

- Hierarchy via **Pooling** (similar to *restriction / prolongation* of multigrid)
- Graphs require clustering step (ideally as preprocessing)
- Grid-based variant *dilation*: larger receptive field, but less aggregation



Pooling with 3x3 Convolution



Dilated Convolutions

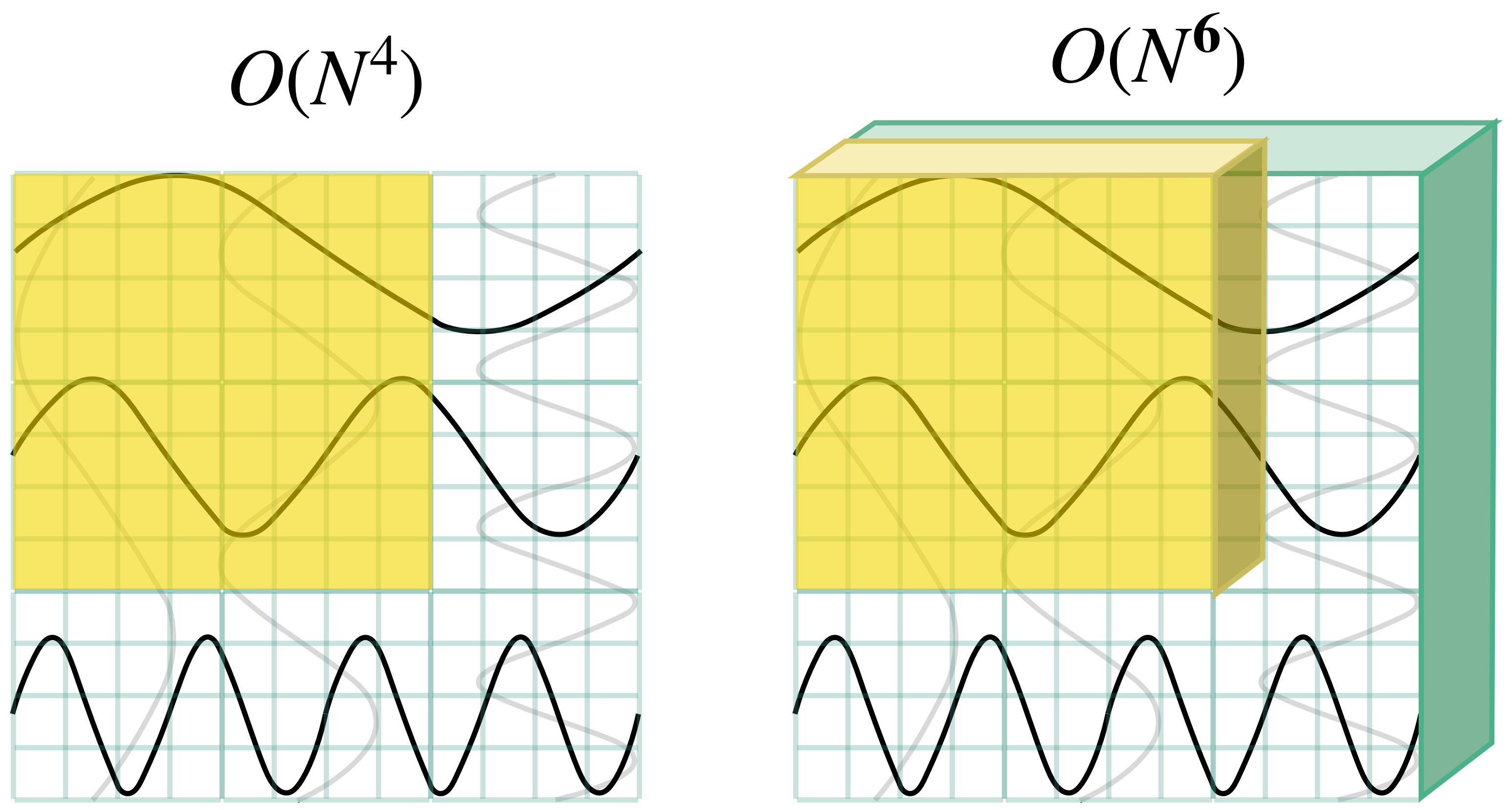
Receptive Fields

Spectral Representations

- Employ *Fourier transforms* (e.g., Fourier Neural Operators)

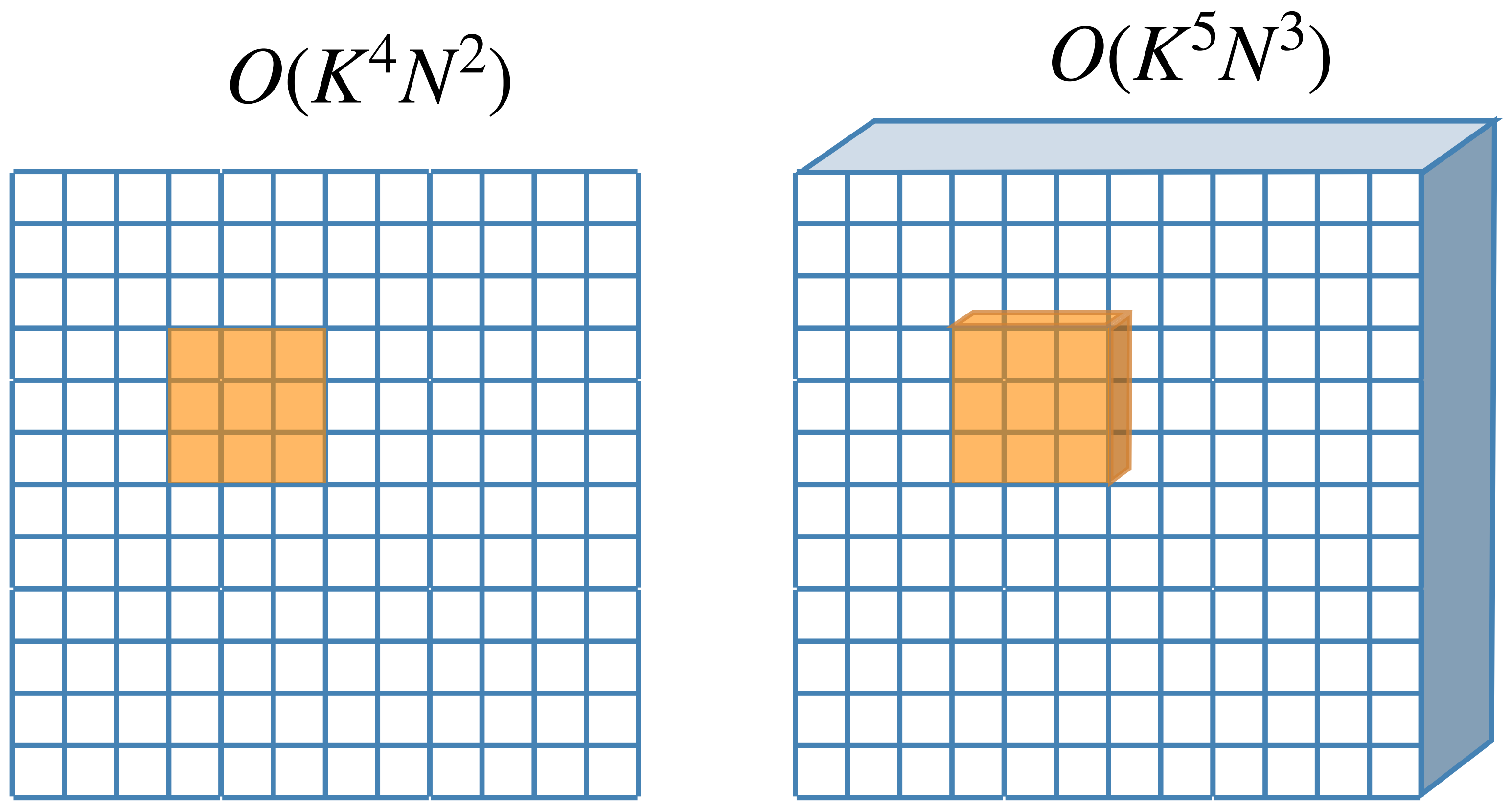
- Note on operator perspective: function transformation, “infinite dimensional”; in practice grids-based, truncated and discrete - not too different from CNN

- But: global basis functions from FFT; suboptimal scaling for 3D



MLP in Frequency Domain
(“1” feature)

CNNs:



Convolutions in Spatial Domain
(kernel size K; assuming feature dimension $O(K)$, not shown)

Receptive Fields

Summary Global vs Local

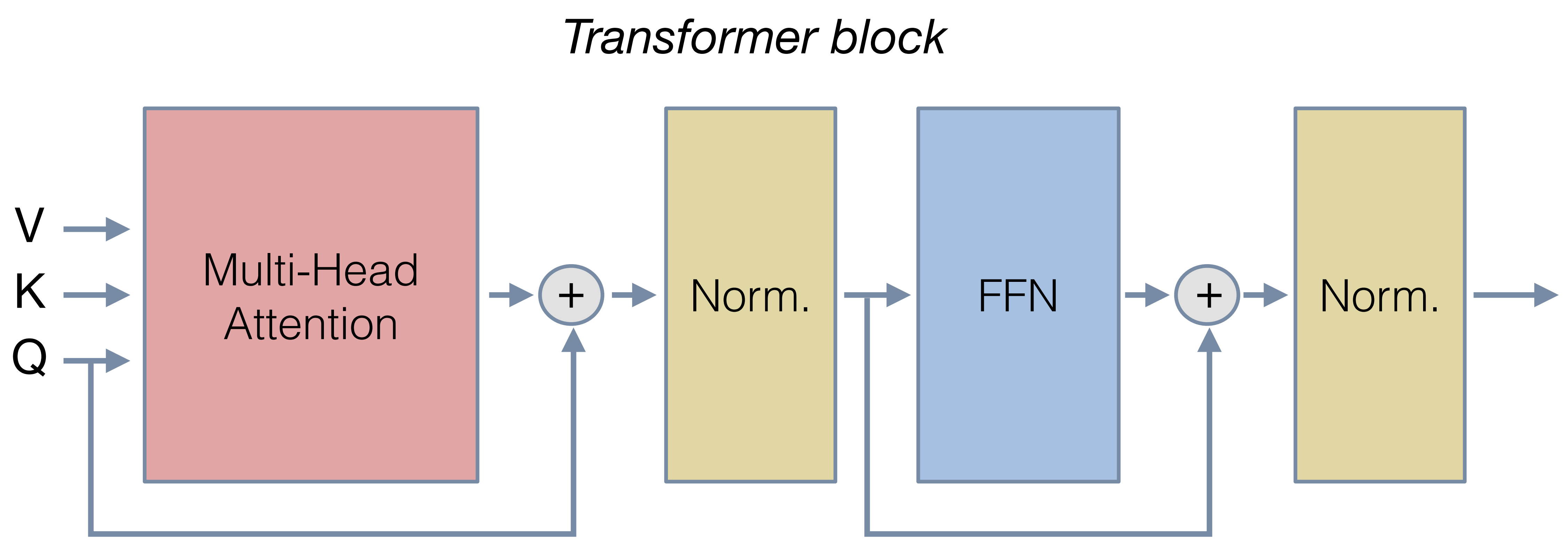
Details to follow!

	Grid	Unstructured	Points
Local	CNN , ResNet	GNN	CConv
Global			
- Hierarchy	U-Net, Dilation	Multi-scale GNN	Multi-scale CConv
- Spectral	FNO	Spectral GNN	(-)

Overview

Transformers

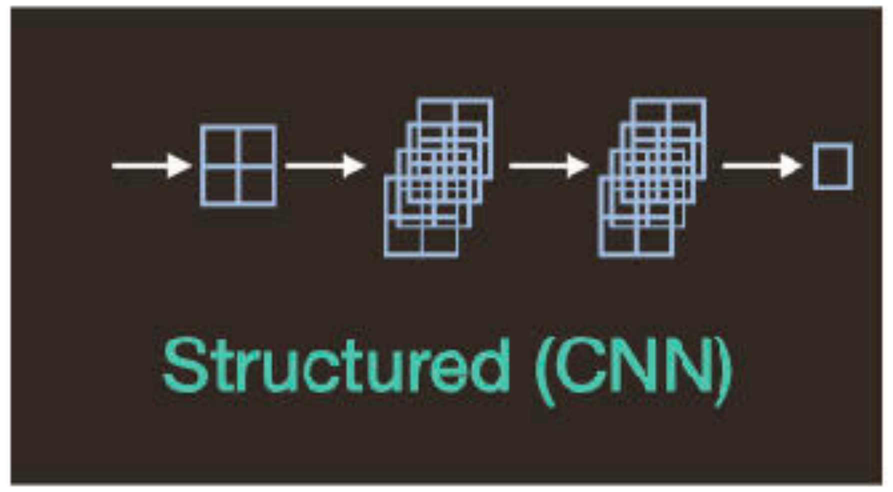
- Sequence-to-sequence architecture, token processing
- *Attention* yields “global” receptive field on token level
- Details out of scope…
- Great scaling (compute) , less ideal for memory (attention is quadratic)
- “The Future” !? 😄🤔

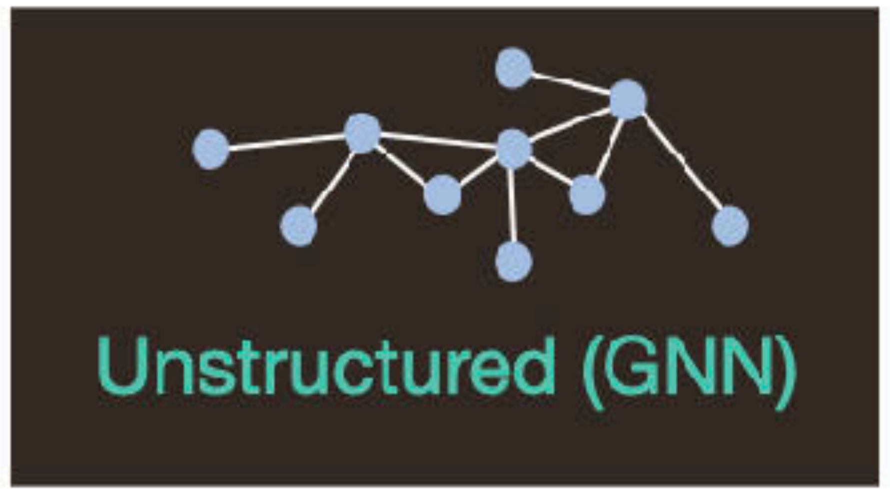


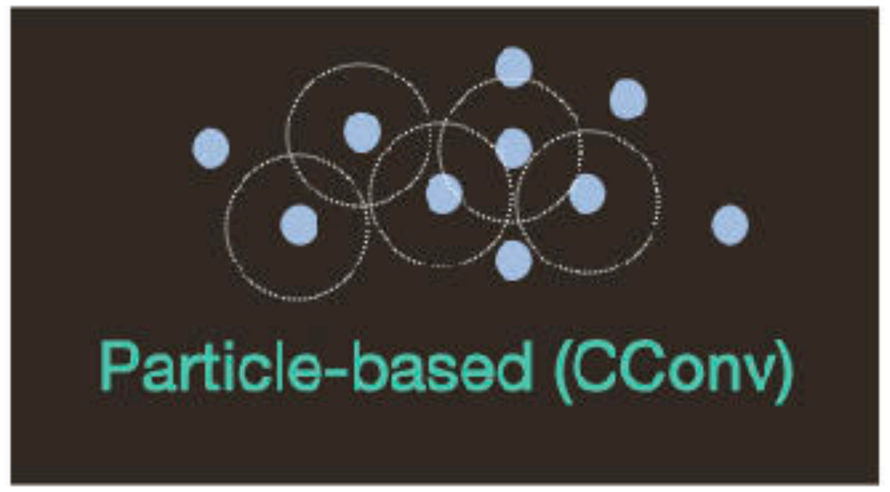
Neural Network Architectures

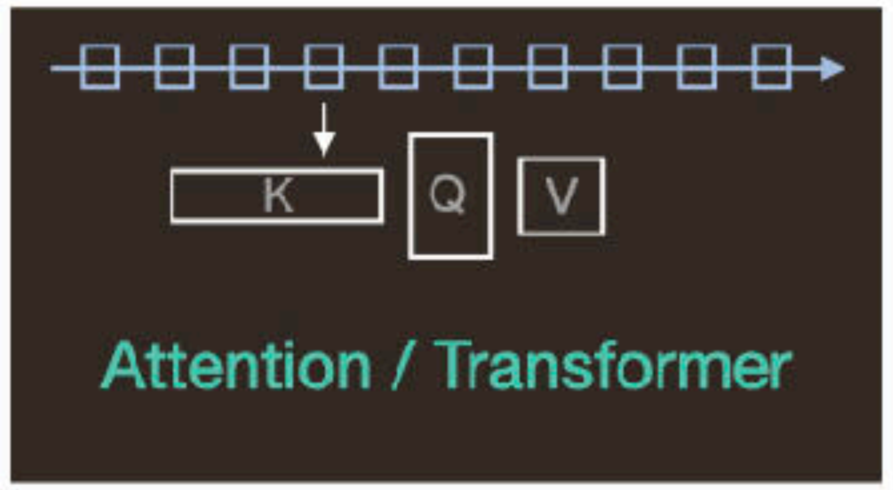
Summary

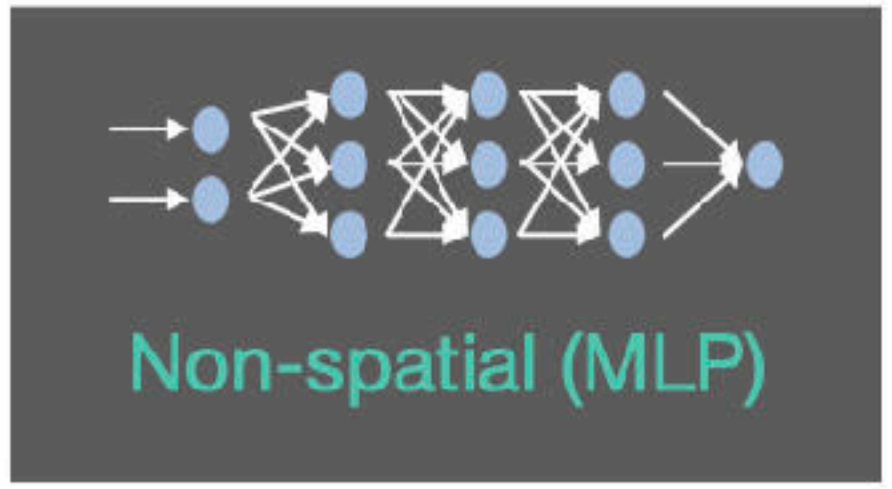
- Central consideration: local vs. global
- Grids or graphs: physics-concepts apply in the same way
- Transformers: likewise grids and graphs…











End