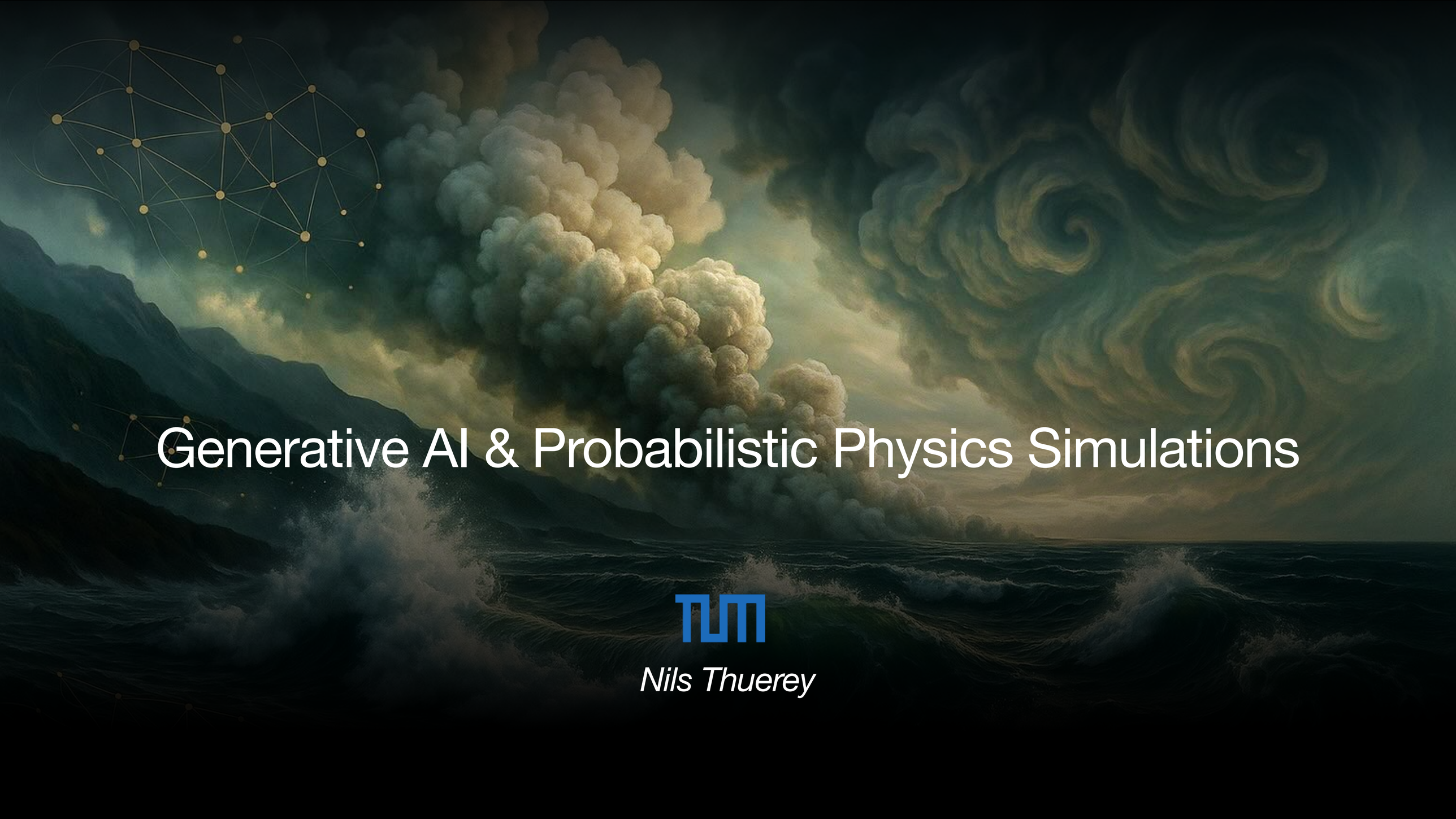


Nils Thuerey

Introduction

- Leverage AI to address challenges

Open Challenges

Current simulations are *deterministic*

The world around us is *ambiguous,*

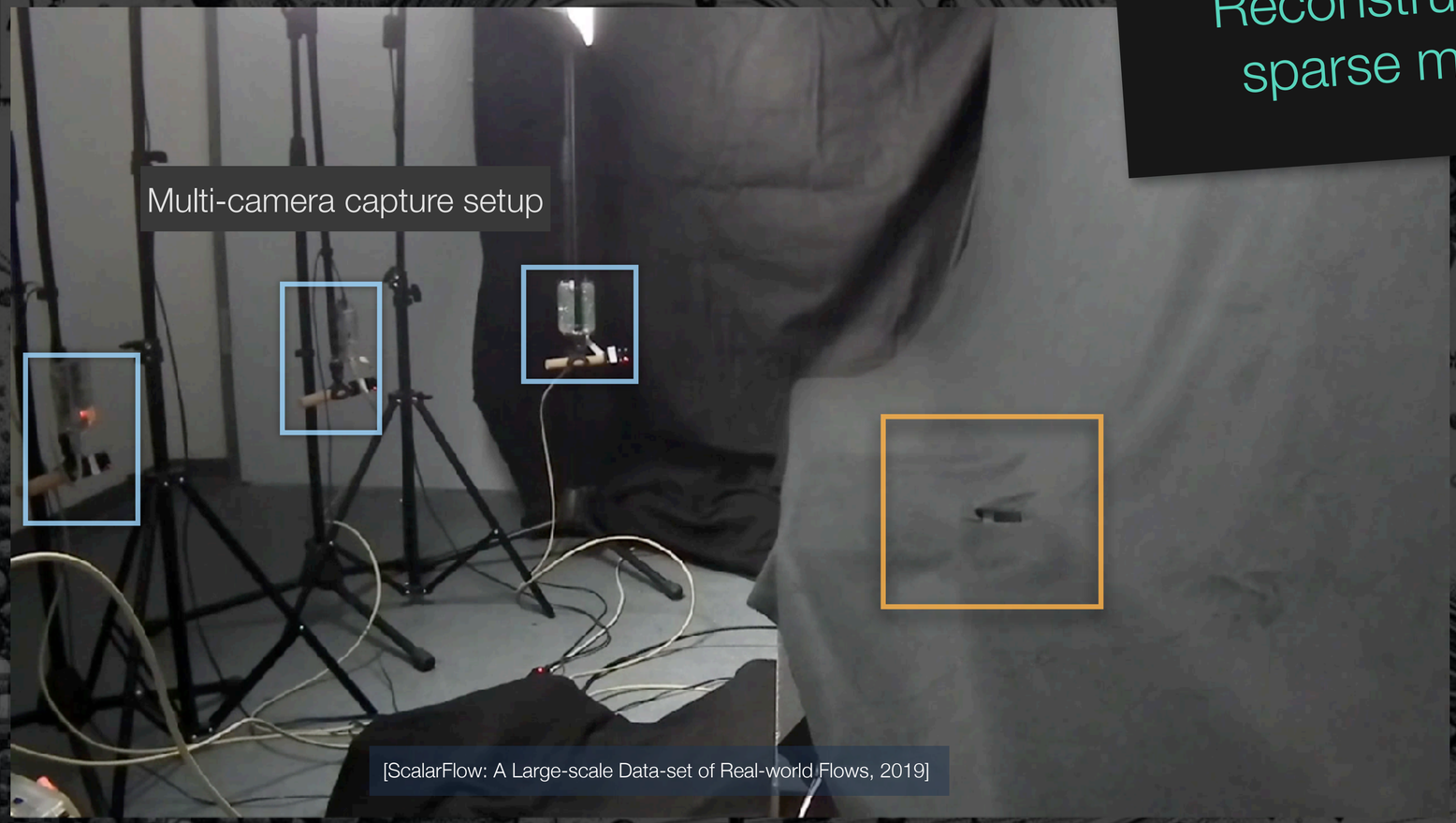
uncertain and complex

“*GenAI* beyo

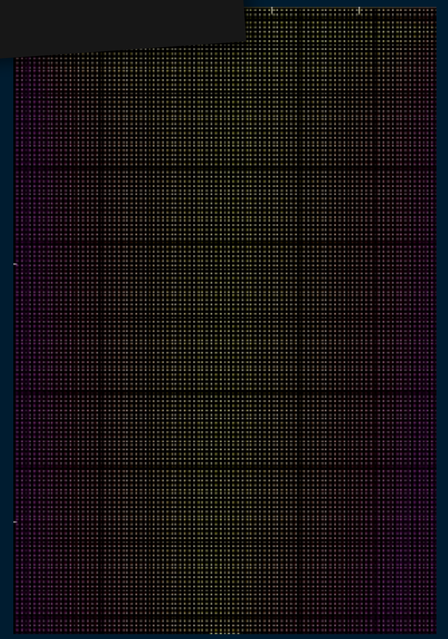
-> Improve simulations on all fronts:
accuracy, functionality, speed, …

Capturing Air Motions

Reconstruct nature from sparse measurements



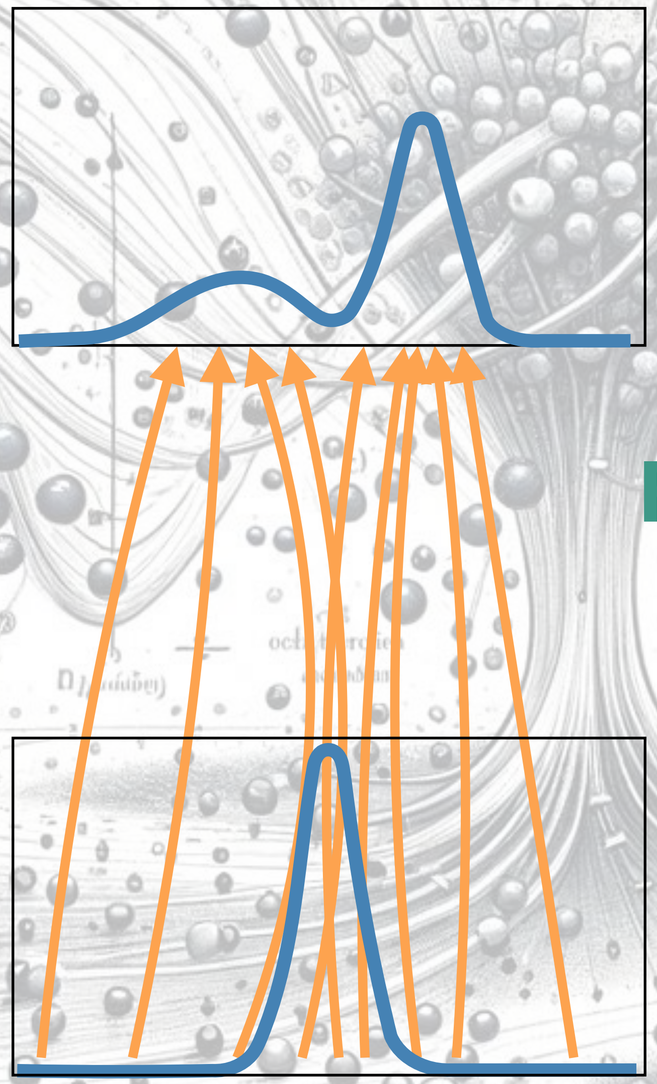
[ScalarFlow: A Large-scale Data-set of Real-world Flows, 2019]



Reconstructed volumetric motion

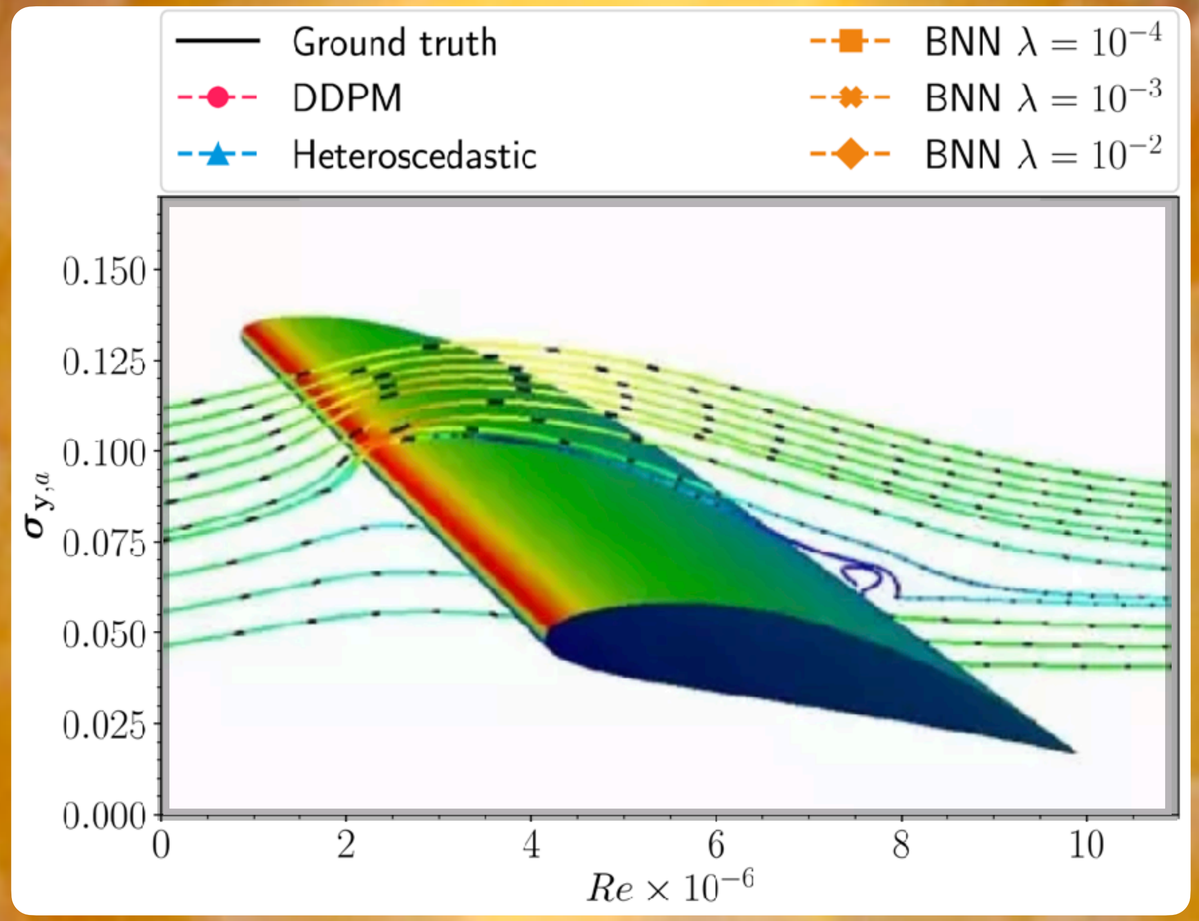
Generative Models

Target distribution



Source distribution

Fundamental question in ML!



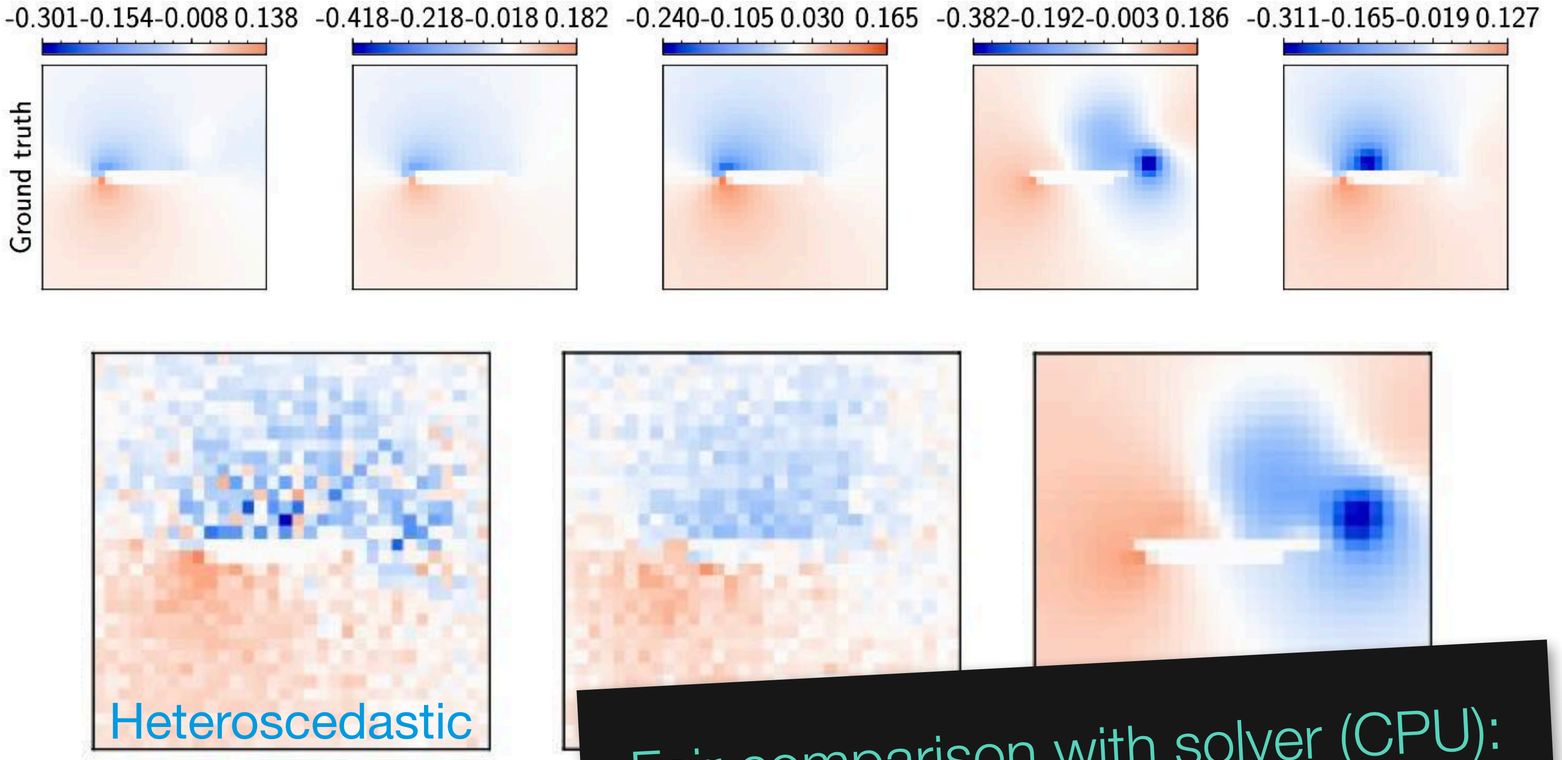
Bayesian NN
Ground Truth
Diffusion Model

Heteroscedastic

Known Ground Truth Distribution

Turbulent NS case with varying Reynolds number:

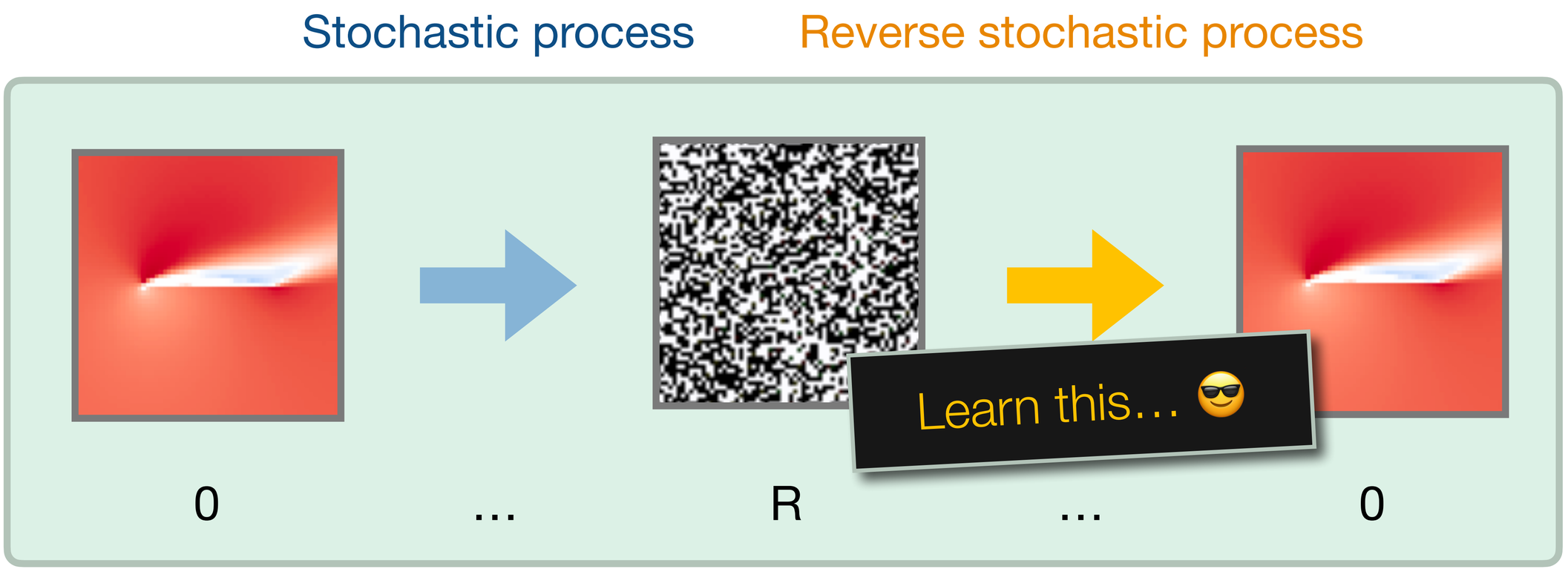
- *Heteroscedastic*
- *Bayesian NNs*
- *DDPM*



Fair comparison with solver (CPU):
DDPM ca. **9.5x** faster

Liu et. al: Uncertainty Prediction of Airfoil Flow S

Diffusion Models (“DDPM”)

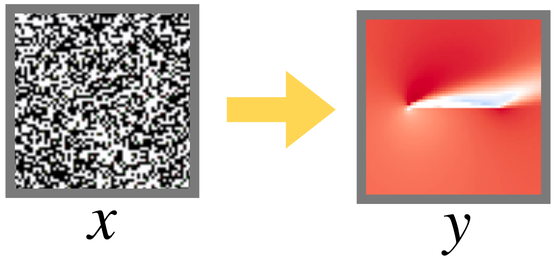


Generative Models

- Everything is a **conditional probability** $p(y \mid x)$
- Examples:
  - Steady state flow: $p(\text{steady state flow} \mid \text{inflow, Re})$
  - Flight conditions: $p(\text{stalling} \mid \text{flow conditions})$

Diffusion Models

- **Output** y from **input** (or parameters) x



- We'd like to know the ***posterior*** distribution: $p(y \mid x)$
- According to Bayes: $p(y \mid x) = \dfrac{p(x \mid y)\ p(y)}{\int p(x \mid y')\ p(y')\ dy'}$

Likelihood

Evidence

- Train $p(y \mid x) = f_\theta(x)$ with $x \sim p(x \mid y)$ being easy to sample
- Denoising diffusion provides **stable, supervised learning objective**

Diffusion Models

- Deterministic / supervised learning: quantity of interest $\bar{y} = f(x)$, approximated by ***NN*** with weights θ: $y = f_\theta(x) \approx \bar{y}$, optimize θ via $\arg\min_\theta |\bar{y} - f_\theta(x)|_2$

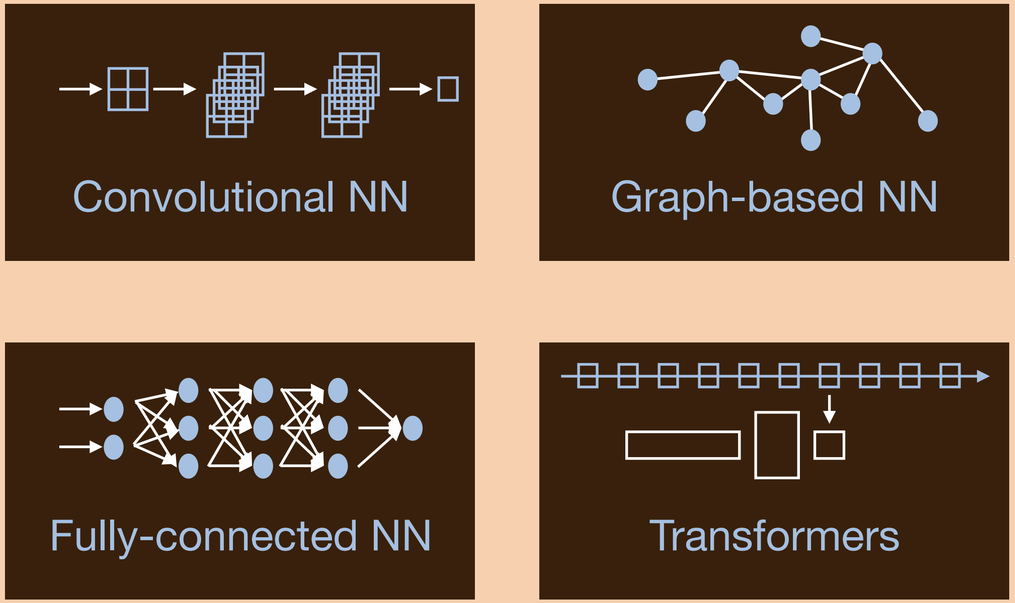
- *Denoising diffusion probabilistic models* (DDPM
  $t \in 0, \ldots, T$ with increasing Gaussian noise:

  ➡ Forward: $y_{r+1} = \sqrt{\alpha_r} y_r + \sqrt{1-\alpha_r}\ \epsilon$ with r

- Learn reverse process $y_{r-1} = \beta_y y_r + \beta_f\ f_\theta(y_r)$ -

  $\beta_\circ$ factors determined by noising schedule

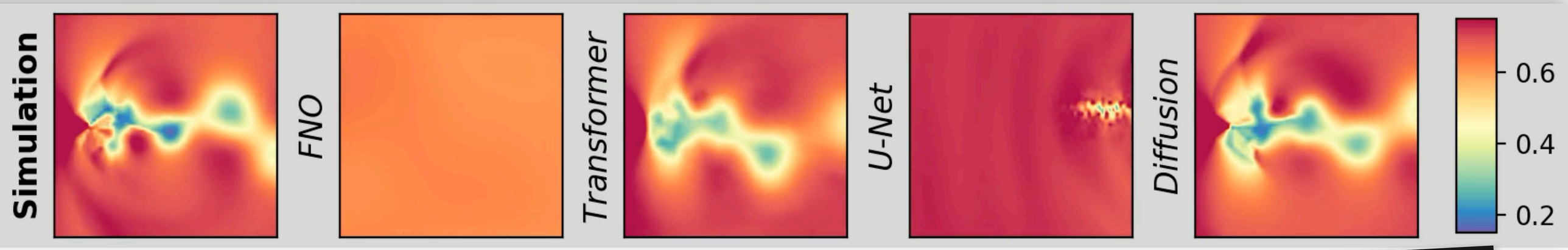
Agnostic to architectures…



Temporal Prediction

$p(\text{next state} \mid \text{previous state})$

- Learned time step $x^{i+1} = f_\theta(x^i)$

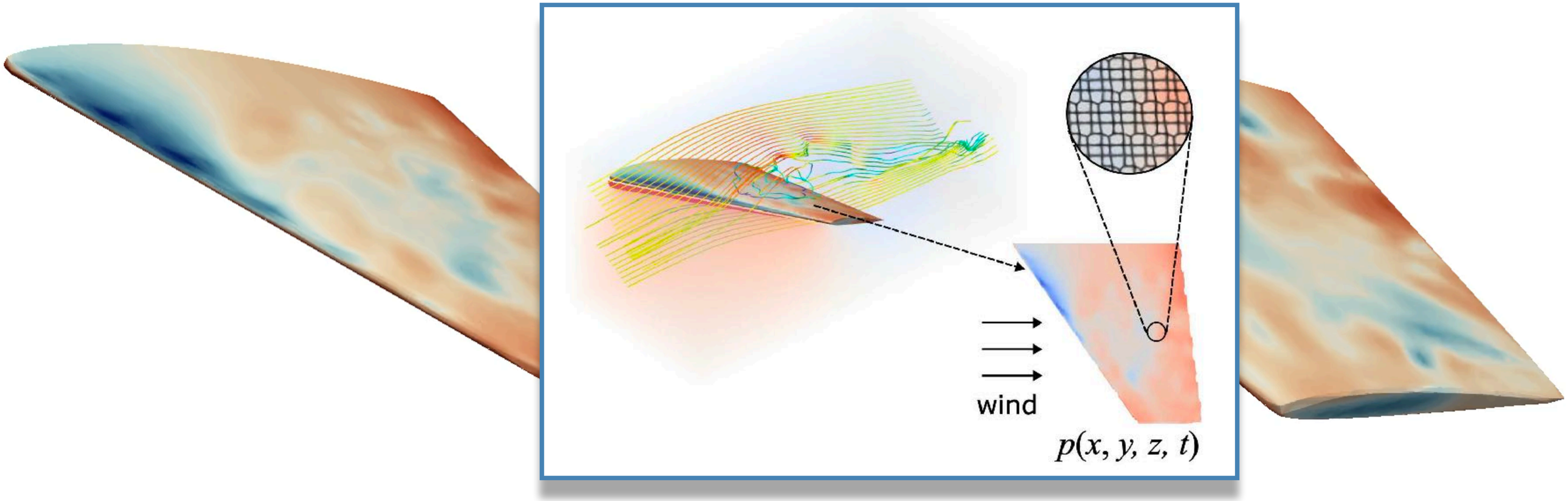


"Unconditionally stable"
(200.000+ steps)

Kohl *et. al*: Turbulent Flow Simulation using Autoregressive Conditional Diffusion Models

Flow Prediction for Unstructured Meshes

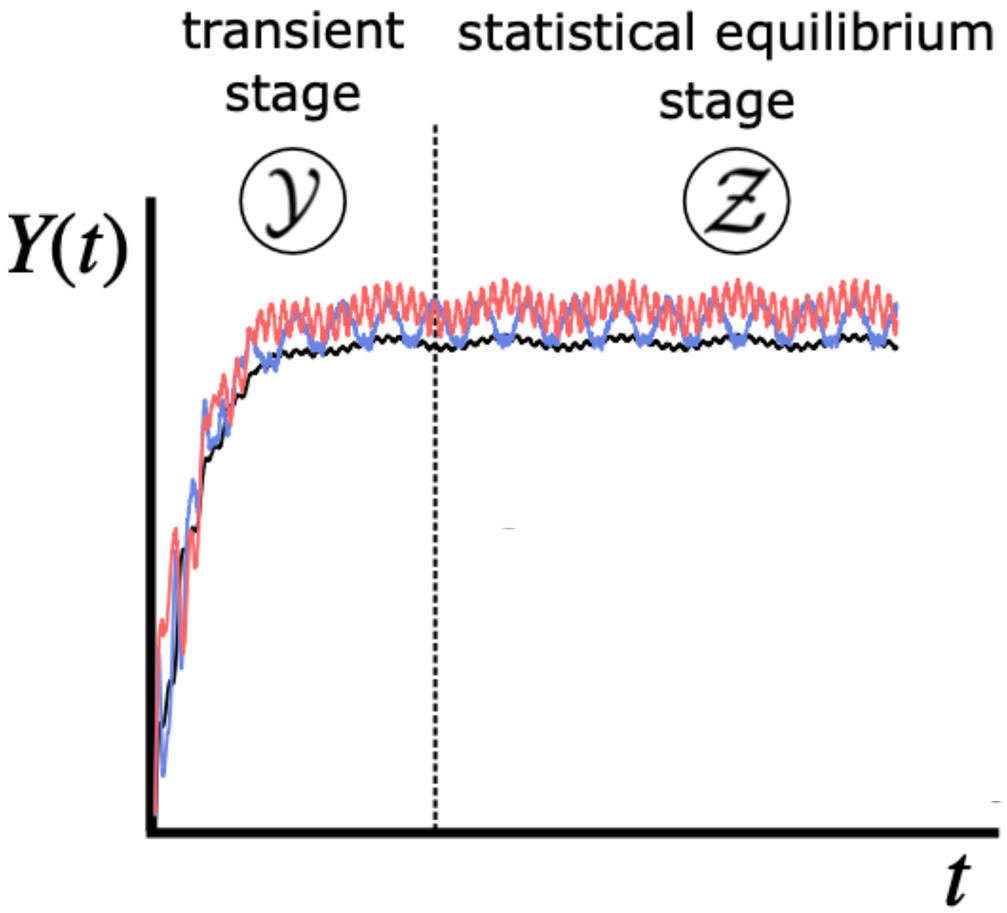
Targeting complex unsteady dynamics (no single mean solution)



Lino et. al: Learning Distributions of Complex Fluid Simulations with Diffusion Graph Networks

Flow Prediction for Unstructured Meshes

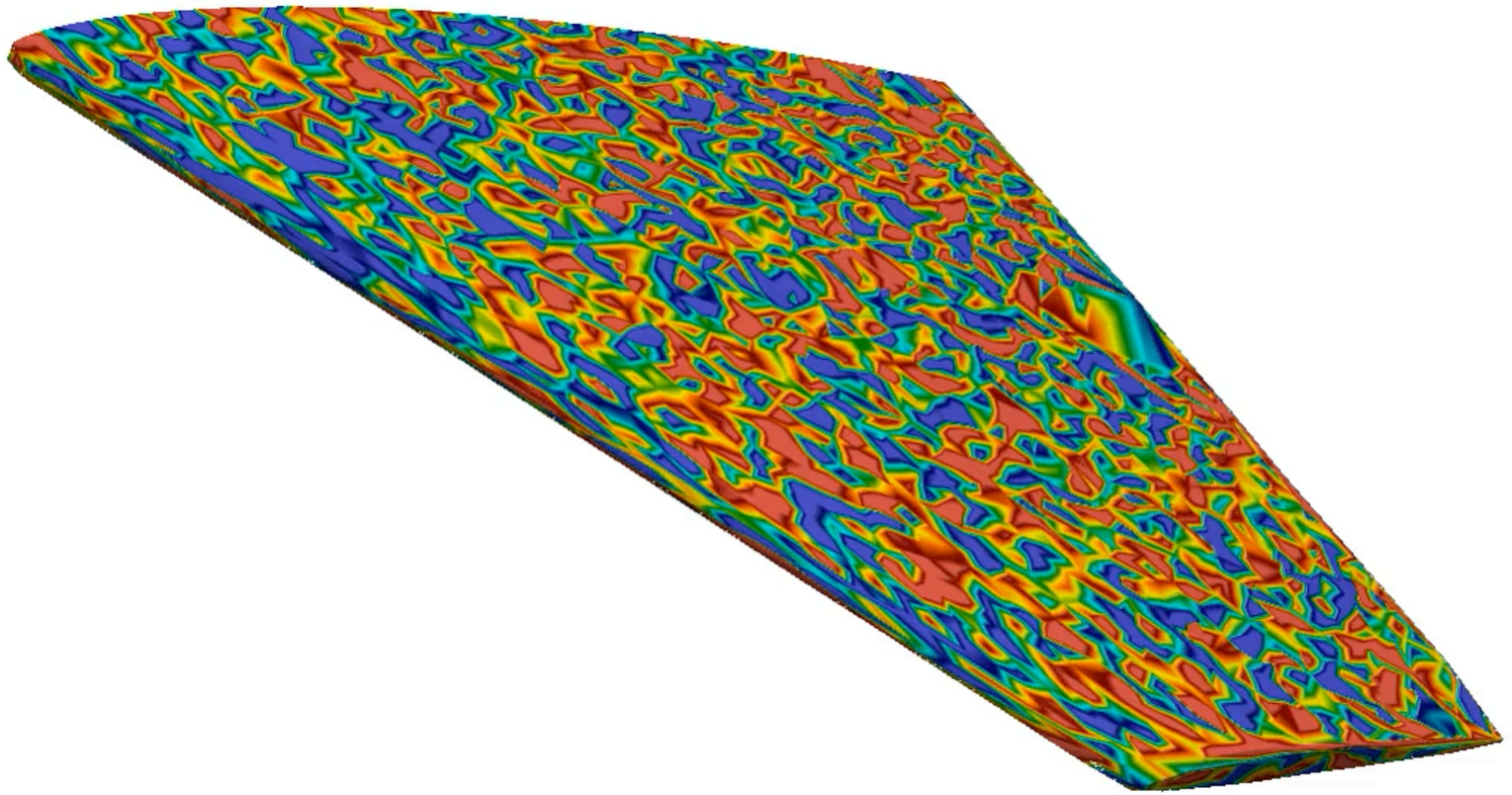
$p(\text{surface pressure} \mid \text{angle of attack, Re})$



Lino et. al: Learning Distributions of Complex Fluid Simulations with Diffusion Graph Networks

Denoising in Action

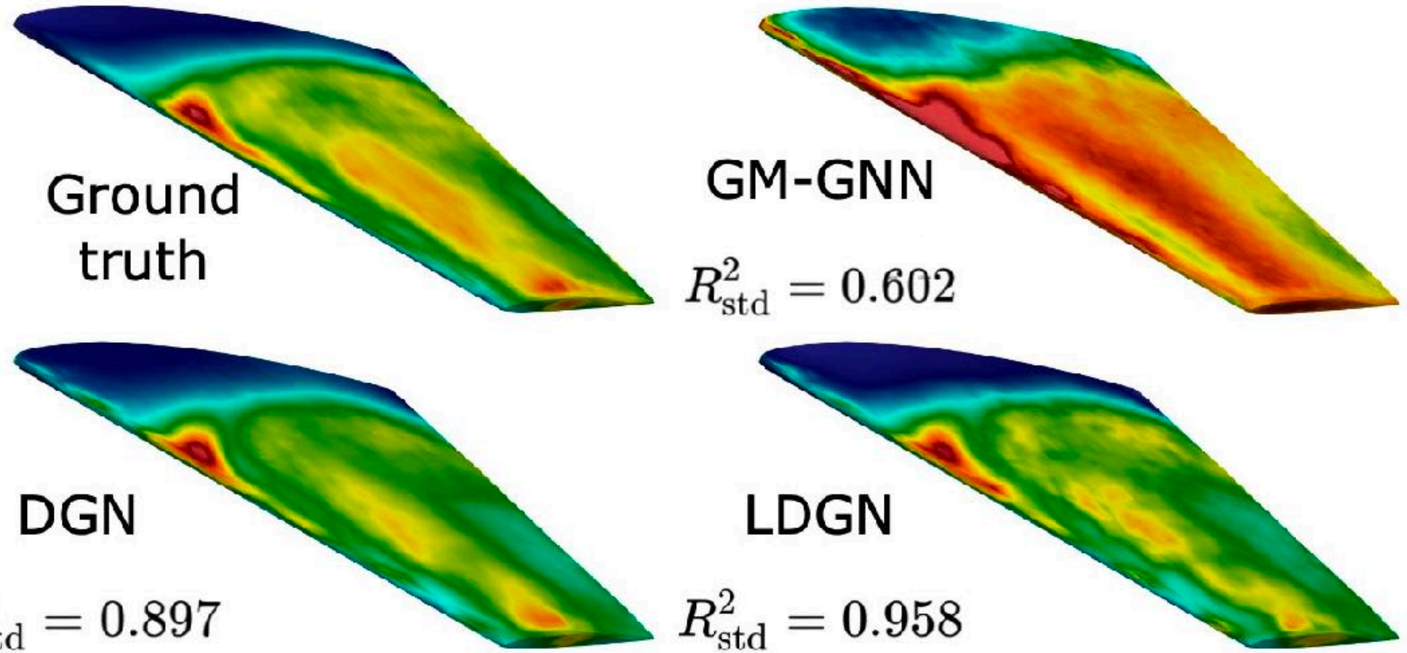
$p(\text{surface pressure} \mid \text{angle of attack, Re})$



Lino et. al: Learning Distributions of Complex Fluid Simulations with Diffusion Graph Networks

Flow Prediction for Unstructured Meshes

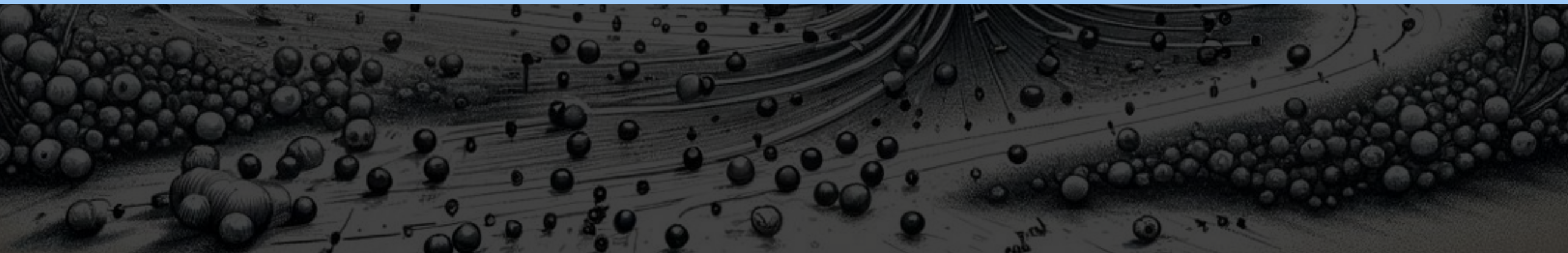
Excellent Distributional Accuracy:
(Inferred std.-dev. on test case)



Model	**CPU** s/sample	**CPU** min/distribution (speedup)	**GPU** s/sample	**GPU** min/distribution (speedup)
DGN	6.81	340 (×8.8)	0.59	19 (×152)
LDGN	0.98	49 (×61)	0.20	2.43 (×1235)

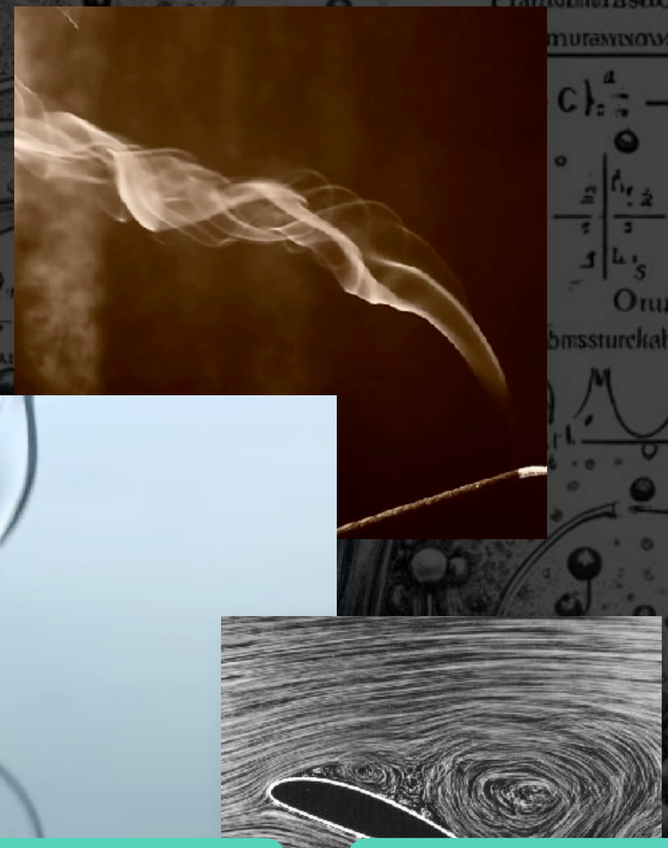
Lino et. al: Learning Distributions of Complex Fluid Simulations with Diffusion Graph Networks

Including Physics Knowledge



Inverse Problems

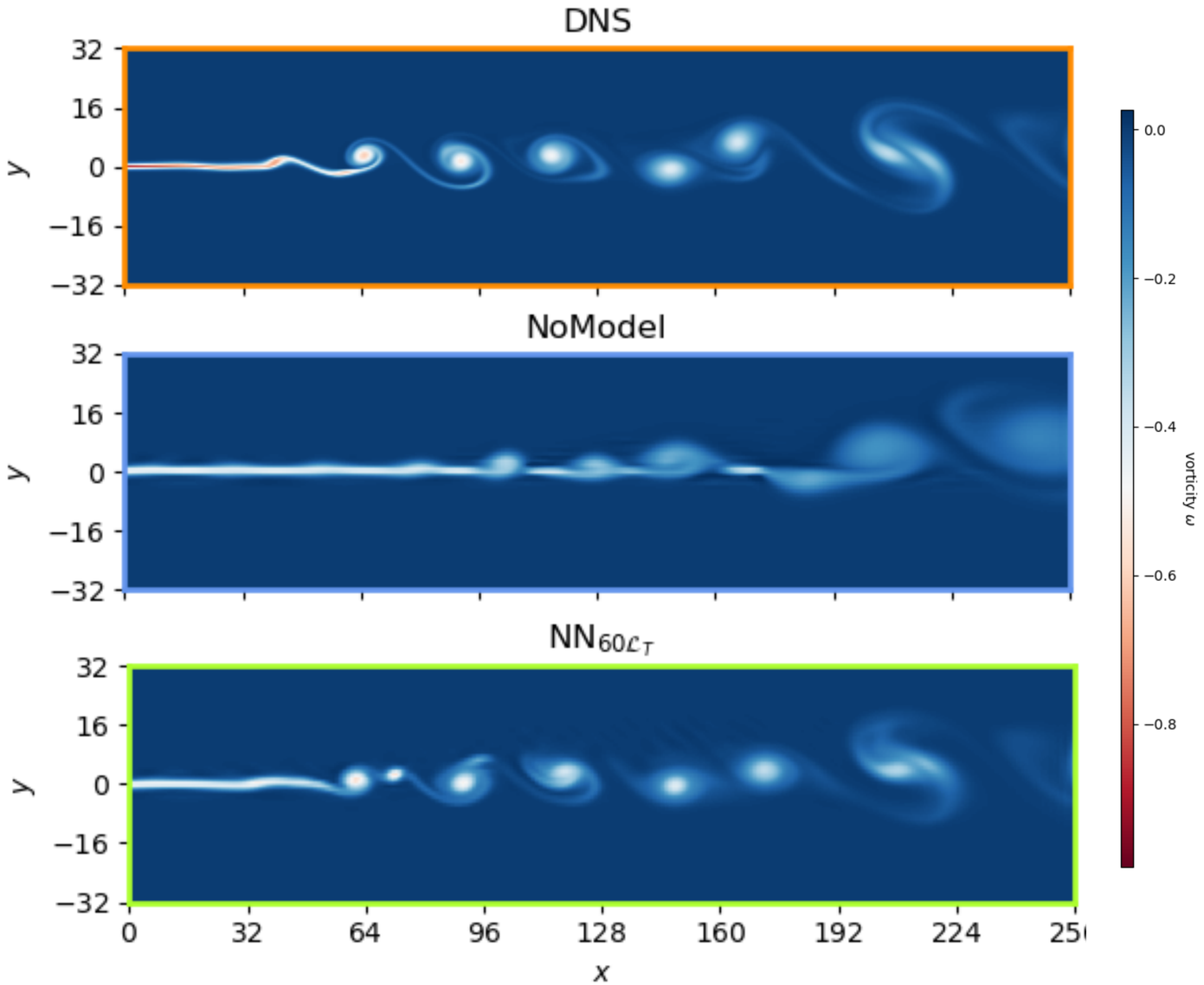
- Leverage **differentiability**
- Trivial for neural networks…
- Numerical methods: for solver $\mathcal{P}(\mathbf{x})$ make sure that it can compute $\partial\mathcal{P}/\partial\mathbf{x}$
- D. Kahnemann: *Thinking fast & slow*



System 1 / NN Component	System 1 / NN Component	System 1 / NN Component	System 1 / NN Component	System 1 / NN Component	System 1 / NN Component
System 2 / Simulation	System 2 / Simulation	System 2 / Simulation	System 2 / Simulation	System 2 / Simulation	System 2 / Simulation

Turbulence: Spatial Mixing Layer

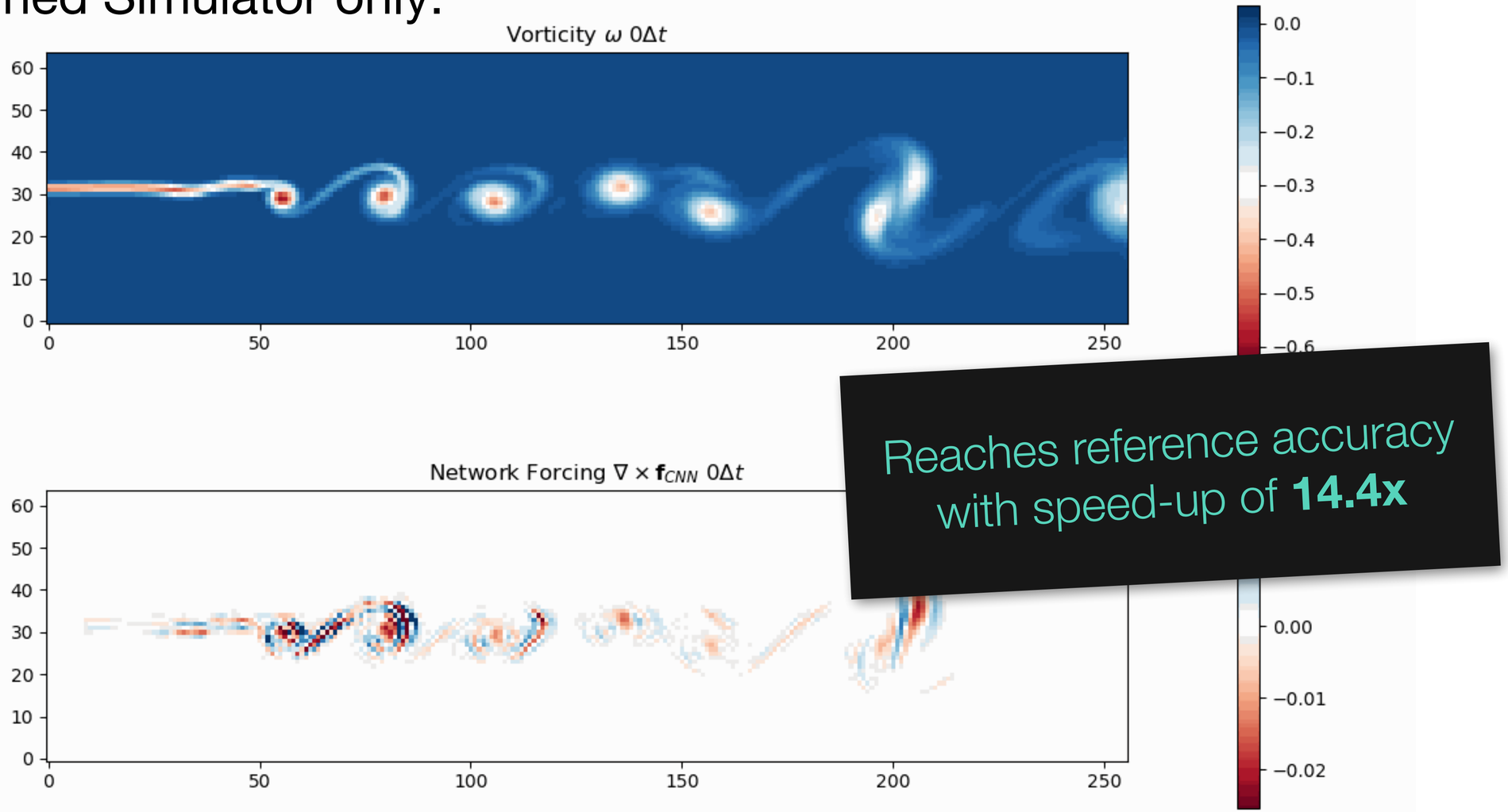
- Expensive reference (DNS) shown at top
- Start with fast, approximate simulator
- Improve solution via NN



List et. al: Learned Turbulence Modelling with Differentiable Fluid Solvers

Turbulence: Spatial Mixing Layer

Learned Simulator only:



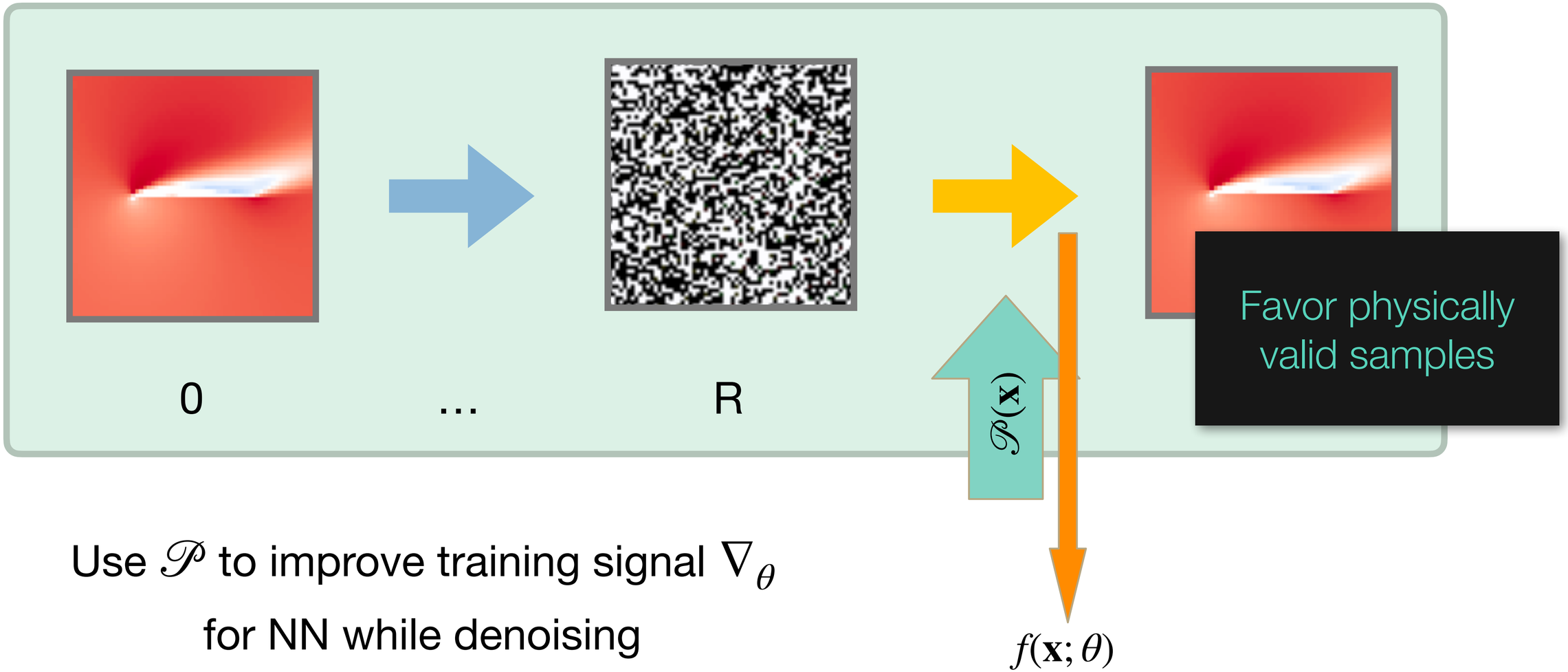
Reaches reference accuracy with speed-up of **14.4x**

List et. al: Learned Turbulence Modelling with Differentiable Fluid Solvers

Physics Constraints for Diffusion Models

- Main options: invoke $\mathscr{P}$ at **training or inference time**
- **1) Inference time:** train diffusion model (DM) as usual; follow physics gradient when de-noising
- **2) Training time:** add physics gradient at training time; inference unmodified

Physics Constraints for Diffusion Models

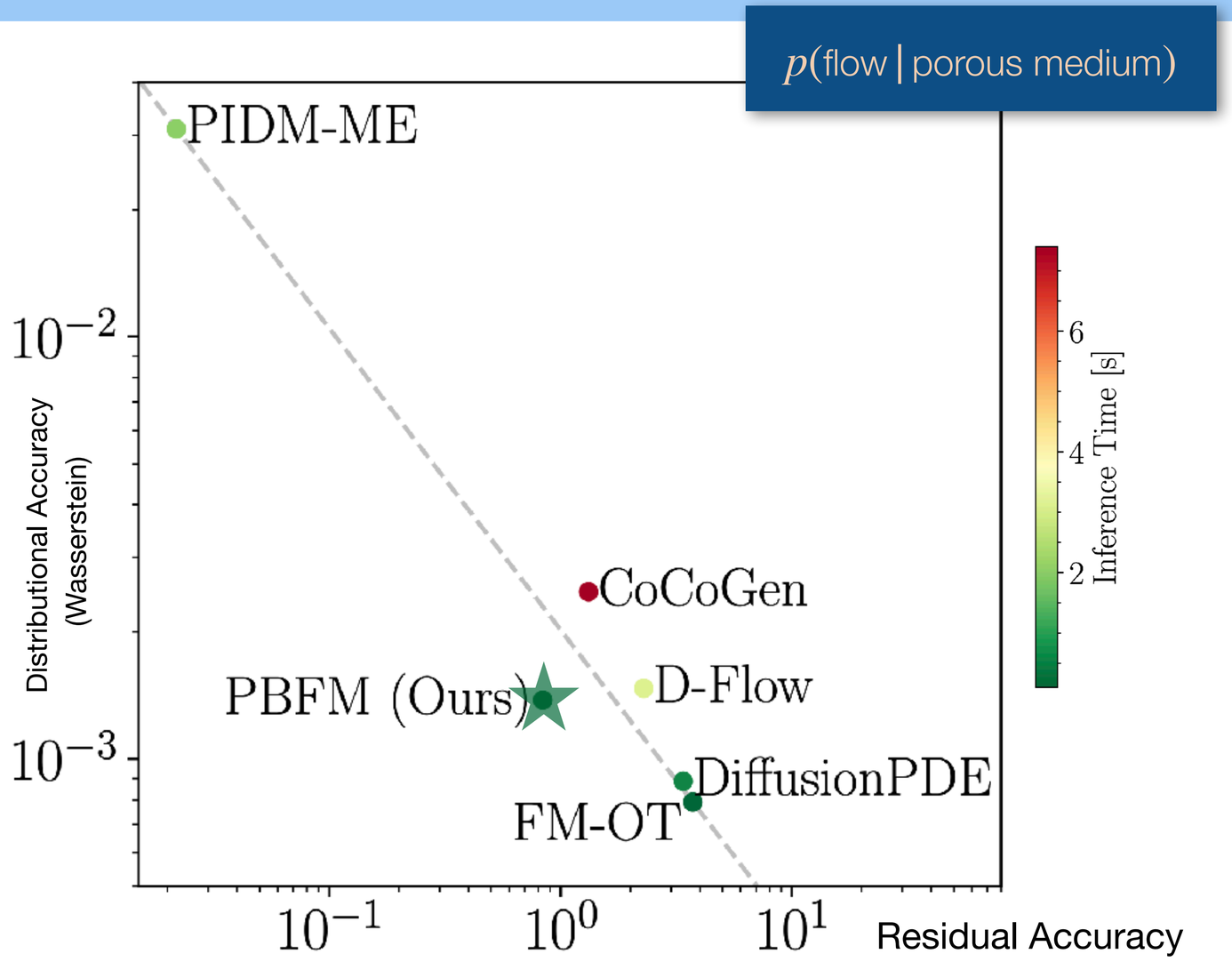


Use $\mathscr{P}$ to improve training signal ∇_θ

for NN while denoising

Physics-Constraints in Action - Training Vs Inference

- Standard test case: Darcy flow (porous medium)
- Inference time: **CoCoGen** (red) slower, larger distributional errors
- Training time: **PBMF** (green) successfully improves residual
- Main conclusion: differentiable solvers also improve probabilistic learning!

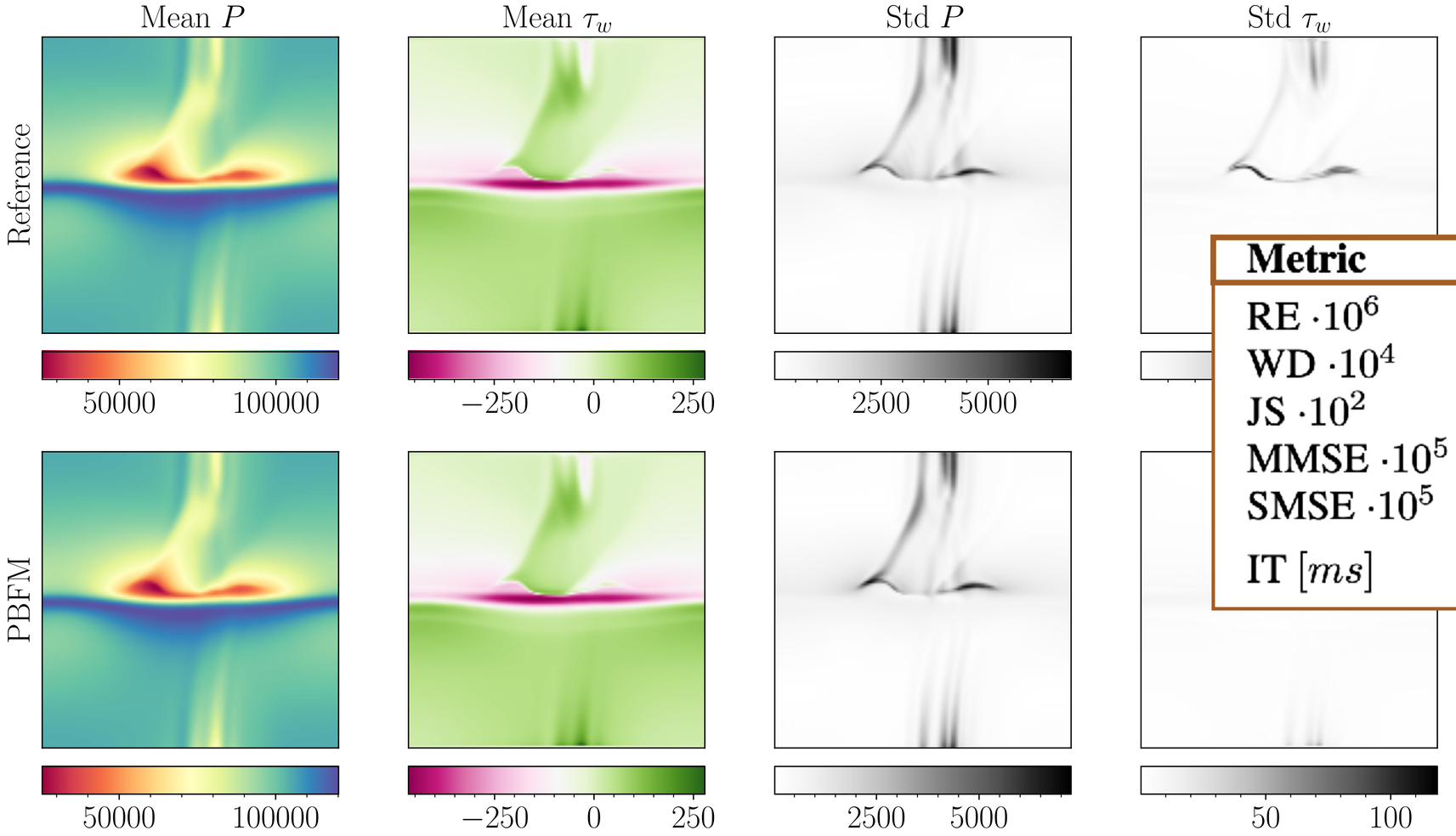


Baldan et. al: Flow Matching meets PDEs: A Unified Framework for Physics-Constrained Generation

Physics-Constraints in Action - Dynamic Stall

- Case with industrial relevance: dynamic stall of rotor blades

$p(\text{surface p} \mid \text{flow condition})$

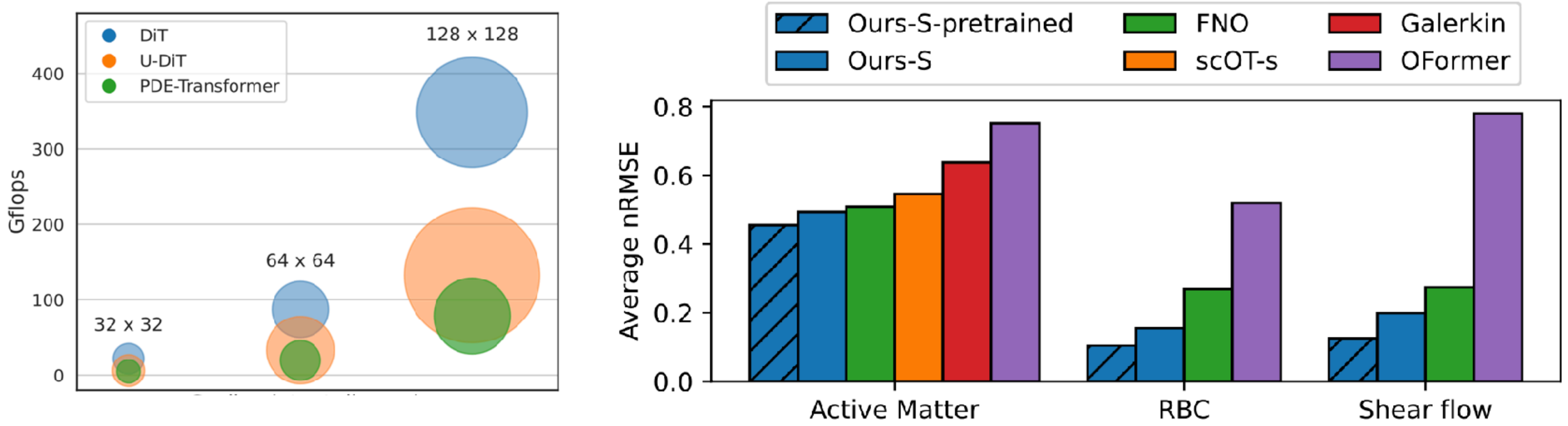


Metric	PBFM	OT-FM	DiffusionPDE
RE $\cdot 10^6$	**0.339**	<u>11.02</u>	12.20
WD $\cdot 10^4$	**1.814**	2.707	<u>2.509</u>
JS $\cdot 10^2$	**0.680**	<u>0.983</u>	1.029
MMSE $\cdot 10^5$	**1.490**	2.791	<u>2.626</u>
SMSE $\cdot 10^5$	**0.874**	1.458	<u>1.236</u>
IT $[ms]$	<u>60.47</u>	**59.75**	171.7

Baldan et. al: Flow Matching meets PDEs: A Unified Framework for Physics-Constrained Generation

Scalability via PDE-Transformer

- Improved scaling, especially important for 3D (up to 512^3)
- Very good generalization to new tasks

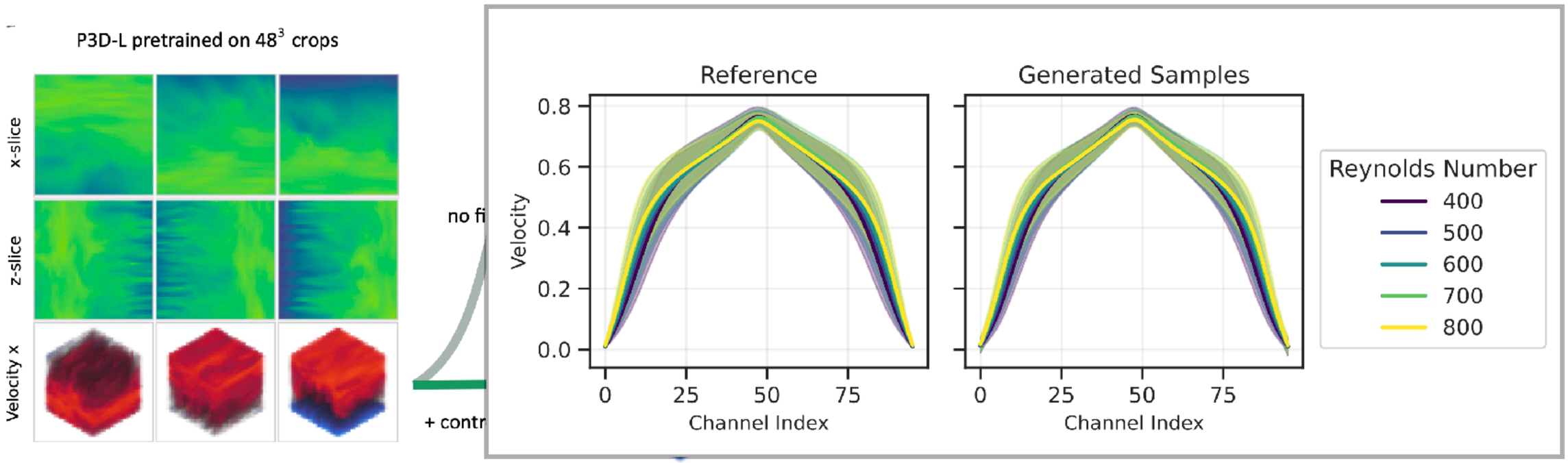


Holzschuh et. al: PDE-Transformer: Efficient and Versatile Transformers for Physics Simulations

Scalability via PDE-Transformer

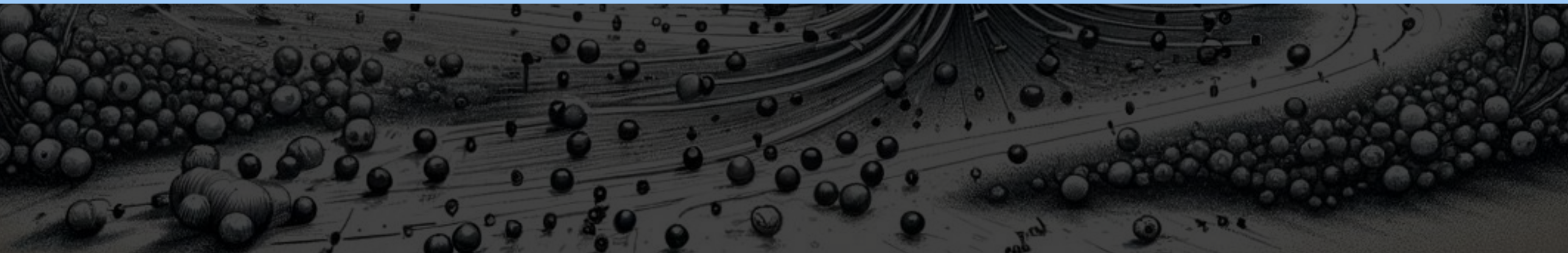
$p(\text{vel,p} \mid \text{Re})$

- Pre-train 3D regions with 48^3 , synchronize latent codes via controller
- Produces samples from equilibrium regime very efficiently



Holzschuh et. al: PDE-Transformer: Efficient and Versatile Transformers for Physics Simulations

Conclusions



Conclusions & Outlook

- ▶ *Diffusion models* can learn complex distributions
- ▶ … provide wide ranging & new applications
- ▶ … profit from human understanding.

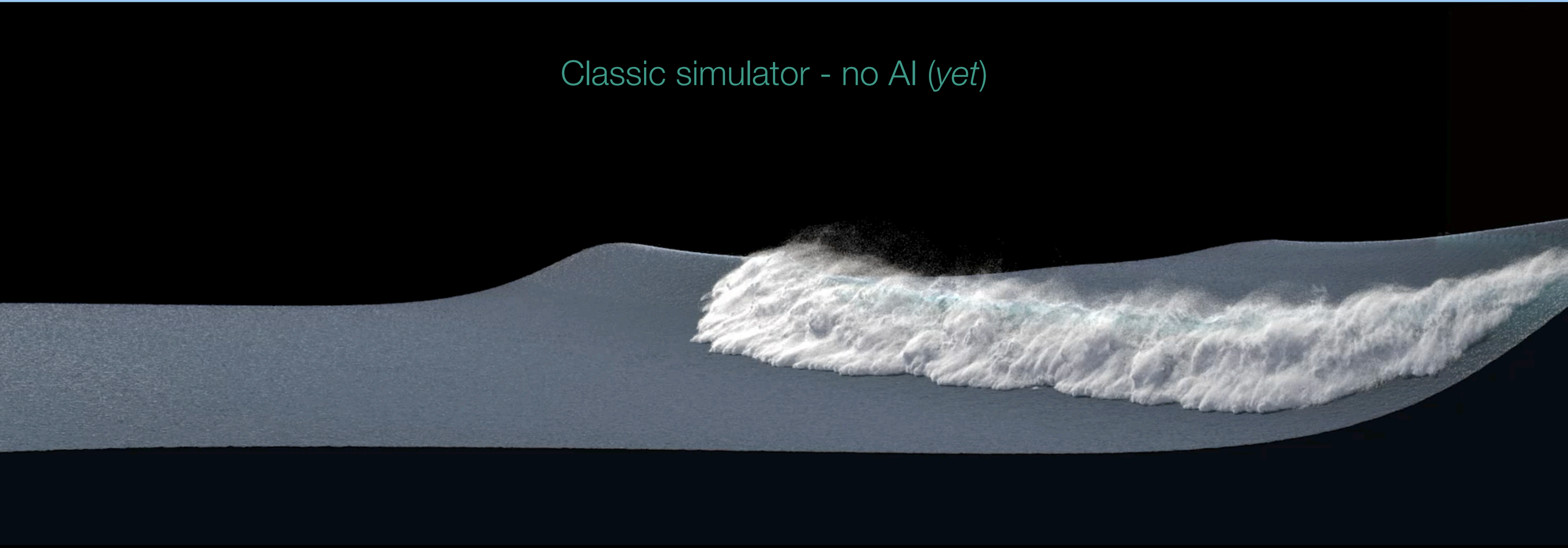
Posterior Distributions

Temporal Predictions

Incorporate Physics

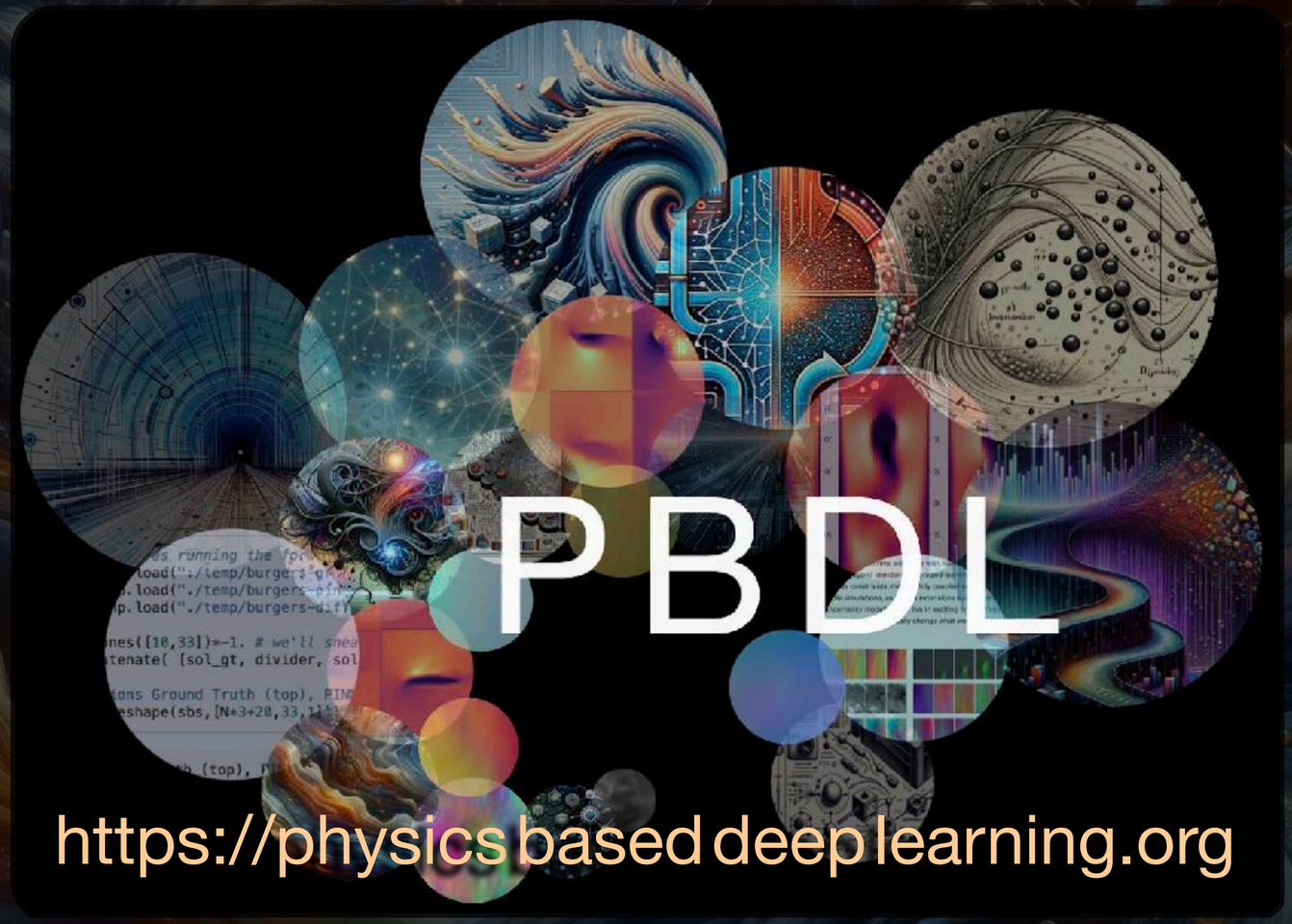
Conclusions & Outlook

Classic simulator - no AI (*yet*)



Braun et. al: Adaptive Phase-Field-FLIP for Very Large Scale Two-Phase Fluid Simulation

Thanks for Listening!



Work done by: Q. Liu, G. Kohl, B. List, L. Chen, B. Holzschuh, B. Braun