Physics-informed Reduced Order Modeling of Time-dependent PDEs via Differentiable Solvers

Nima Hosseini*, Hesam Salehipour,
Adrian Butscher, and Nigel Morris

*Computer Science PhD Student at Western University
Research conducted during internship at Autodesk

Accelerated (Real-time?) Simulation

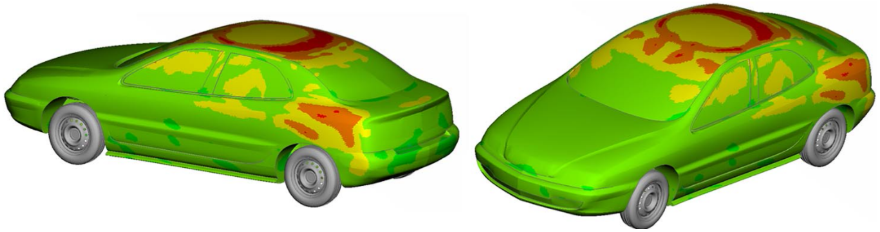
Many-query problems rely on exploring the solution manifold of governing PDEs

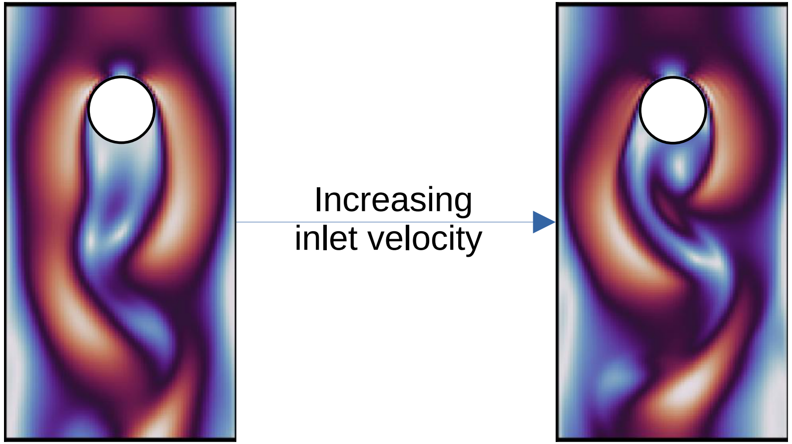
- Design optimization
- Optimal control
- Inverse problems

Accelerated (Real-time?) Simulation

Many-query problems rely on exploring the solution manifold of governing PDEs

- Design optimization
- Optimal control
- Inverse problems

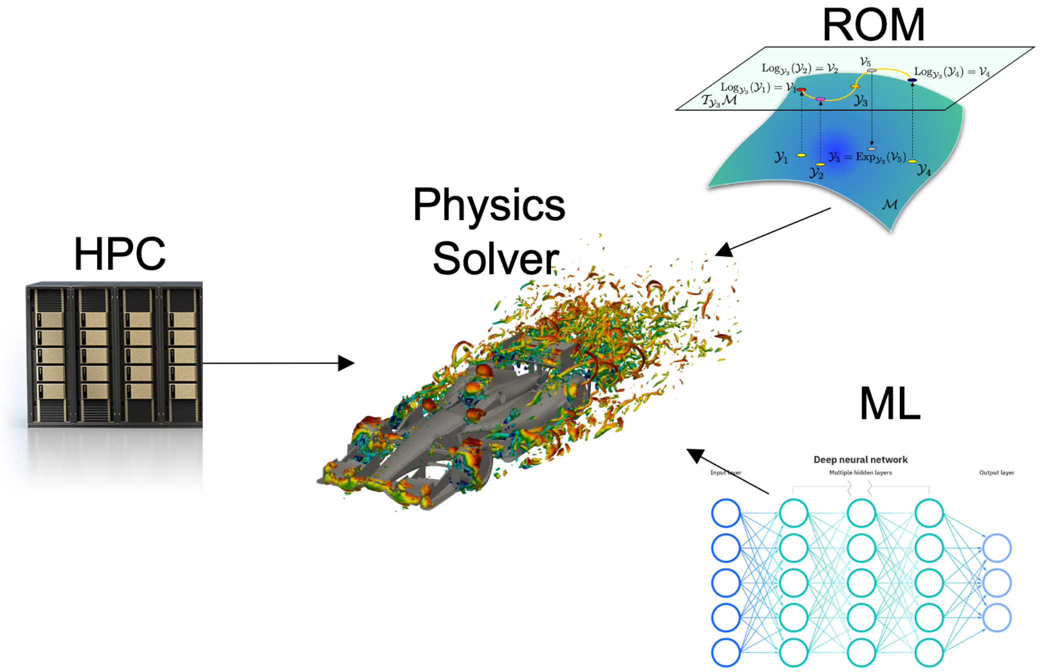




Accelerated (Real-time?) Simulation

Many-query problems rely on exploring the solution manifold of governing PDEs

- Design optimization
- Optimal control
- Inverse problems



Problem Statement

For time-dependent parameterized PDE:

$$\dot{u} = \mathcal{N}(u; \beta), \quad u(t, x) : \mathcal{T} \times \Omega \to \mathbb{R}^m,$$

Find:

1. A reduced latent space with latent coordinates $\alpha_t \in \mathbb{R}^k$, corresponding to solution states u_t in the physical space at time t.

Problem Statement

For time-dependent parameterized PDE:

$$\dot{u} = \mathcal{N}(u; \beta), \quad u(t, x) : \mathcal{T} \times \Omega \to \mathbb{R}^m,$$

Find:

1. A reduced latent space with latent coordinates $\alpha_t \in \mathbb{R}^k$, corresponding to solution states u_t in the physical space at time t.
2. A mapping D_θ from latent space to physical space and vice-versa

Problem Statement

For time-dependent parameterized PDE:

$$\dot{u} = \mathcal{N}(u; \beta), \quad u(t, x) : \mathcal{T} \times \Omega \to \mathbb{R}^m,$$

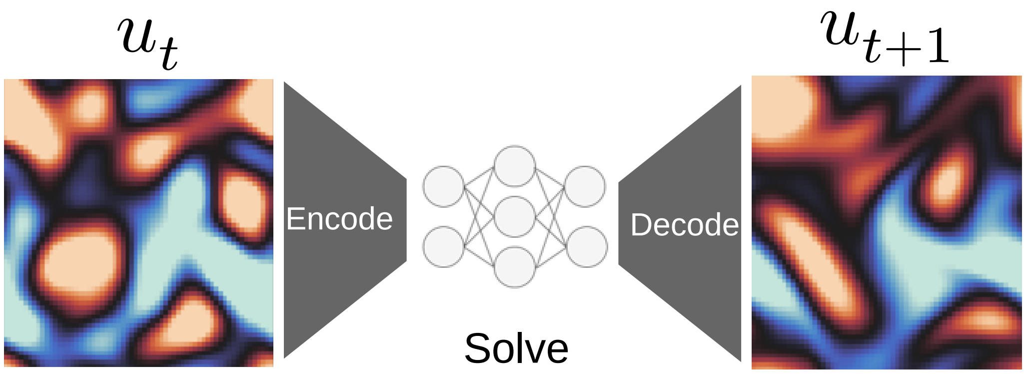
Find:

1. A reduced latent space with latent coordinates $\alpha_t \in \mathbb{R}^k$, corresponding to solution states u_t in the physical space at time t.
2. A mapping D_θ from latent space to physical space and vice-versa
3. A mapping from α_t to α_{t+1} that solves the PDE in the latent space

Problem Statement

For time-dependent parameterized PDE:

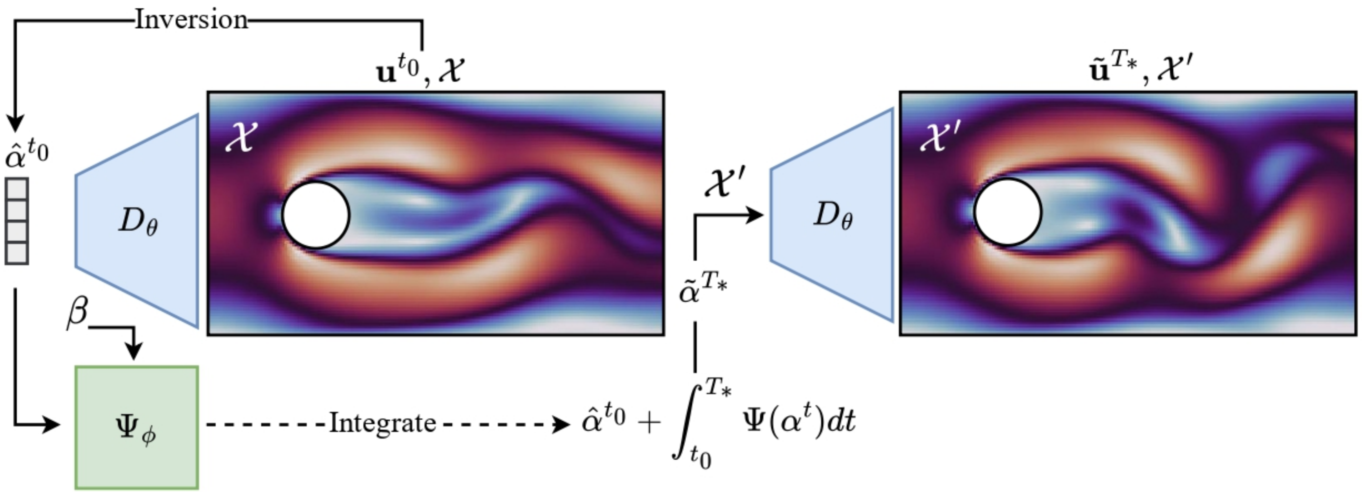
$$\dot{u} = \mathcal{N}(u; \beta), \quad u(t, x) : \mathcal{T} \times \Omega \to \mathbb{R}^m,$$



Introducing $\Phi\!-\!\mathrm{ROM}$

$\Phi\!-\!$ROM uses the same architecture as DINo[1]:

- Mesh-free and continuous decoding and encoding using a Neural Field $D(\alpha, \mathcal{X}) = \mathbf{u}$,
- Latent space time evolution using Neural-ODE-inspired dynamics network, $\Psi(\alpha) = d\alpha/dt$.

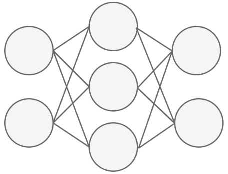


AUTODESK Research

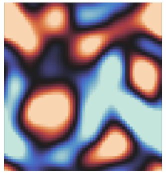
[1] "Continuous PDE Dynamics Forecasting with Implicit Neural Representations," Y. Yin, et al.

An emerging “physics-informed” method

u_t →



→ $\hat{u}_{t+1}$



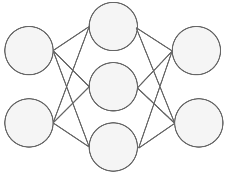
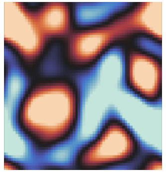
Data-Driven

$$Loss = \text{MSE}(\hat{u}_{t+1} - u_{t+1})$$

Fails to generalize to unseen physical parameters, forecast beyond the training time horizon, etc.

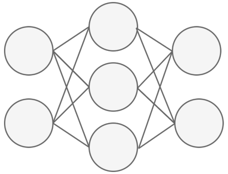
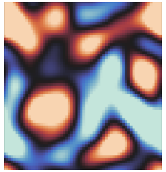
An emerging “physics-informed” method

$\hat{u}_{t+1}$

u_t →  → 

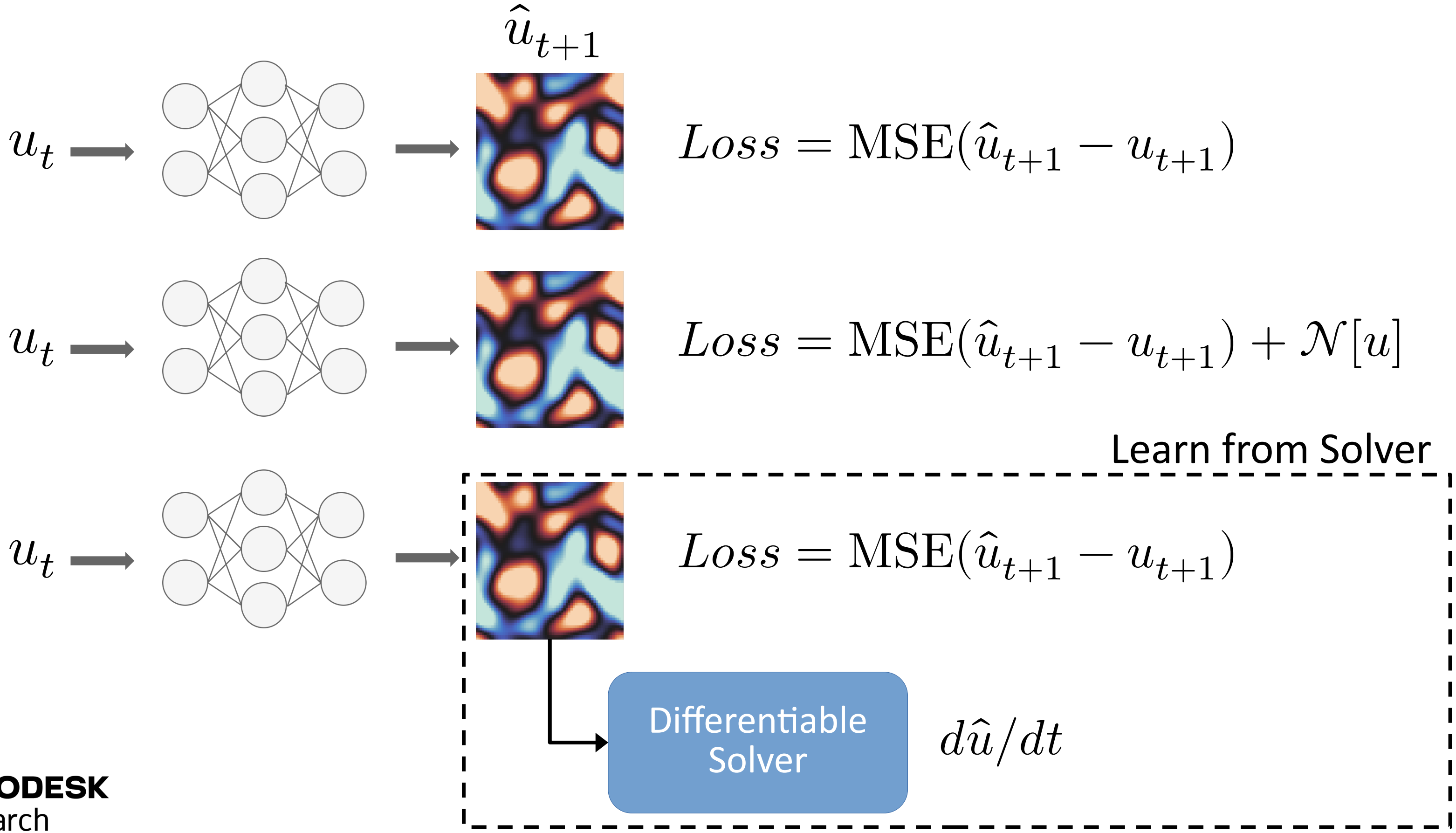
$$Loss = \mathrm{MSE}(\hat{u}_{t+1} - u_{t+1})$$

Learn from PDE

u_t →  → 

$$Loss = \mathrm{MSE}(\hat{u}_{t+1} - u_{t+1}) + \mathcal{N}[u]$$

An emerging "physics-informed" method



Training Φ−ROM

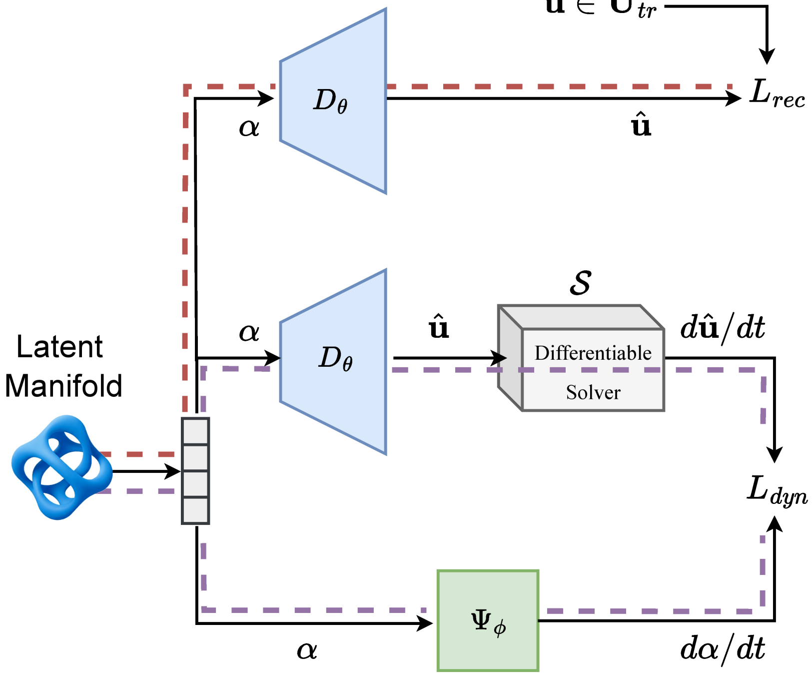
Φ−ROM learns the latent dynamics $d\alpha/dt$ directly from a differentiable PDE solver $\mathcal{S}$.

- The dynamics loss L_{dyn} trains the dynamics network using the true physical dynamics given by the solver, i.e. $d\hat{u}/dt = \mathcal{S}\left[\hat{u}\right]$.

$$J_D(\alpha)\frac{d\alpha}{dt} = \frac{d\hat{u}}{dt} \quad \Rightarrow \quad \Psi(\alpha) = J_D(\alpha)^{\dagger}\frac{d\hat{u}}{dt}$$

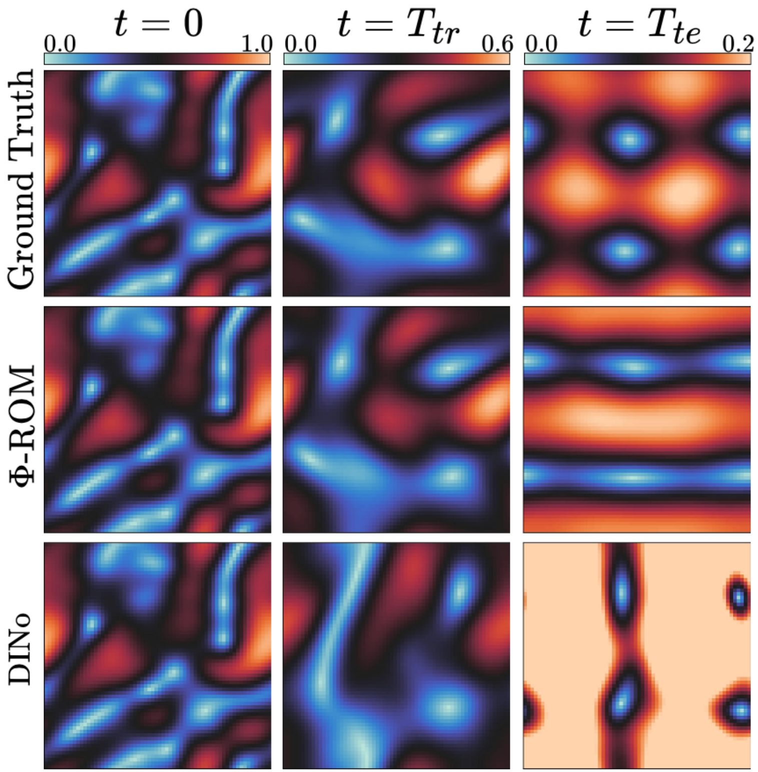
Where $d\alpha/dt = \Psi(\alpha)$.

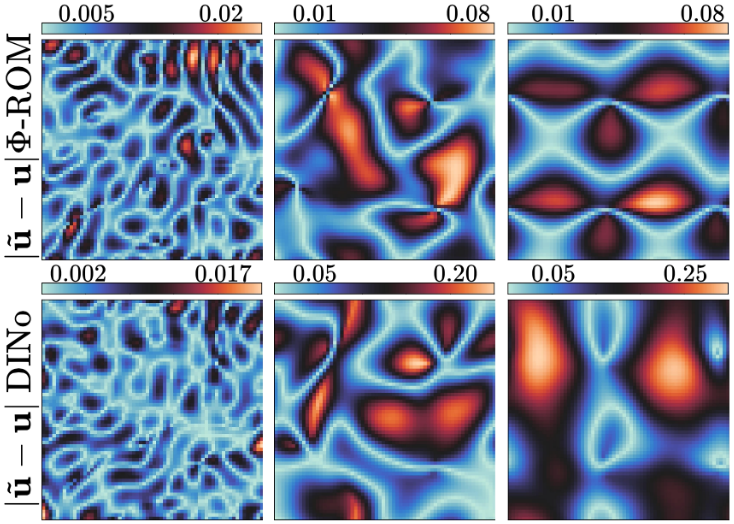
- The latent manifold is regularized by the PDE solver.



AUTODESK
Research

Results: Temporal Stability



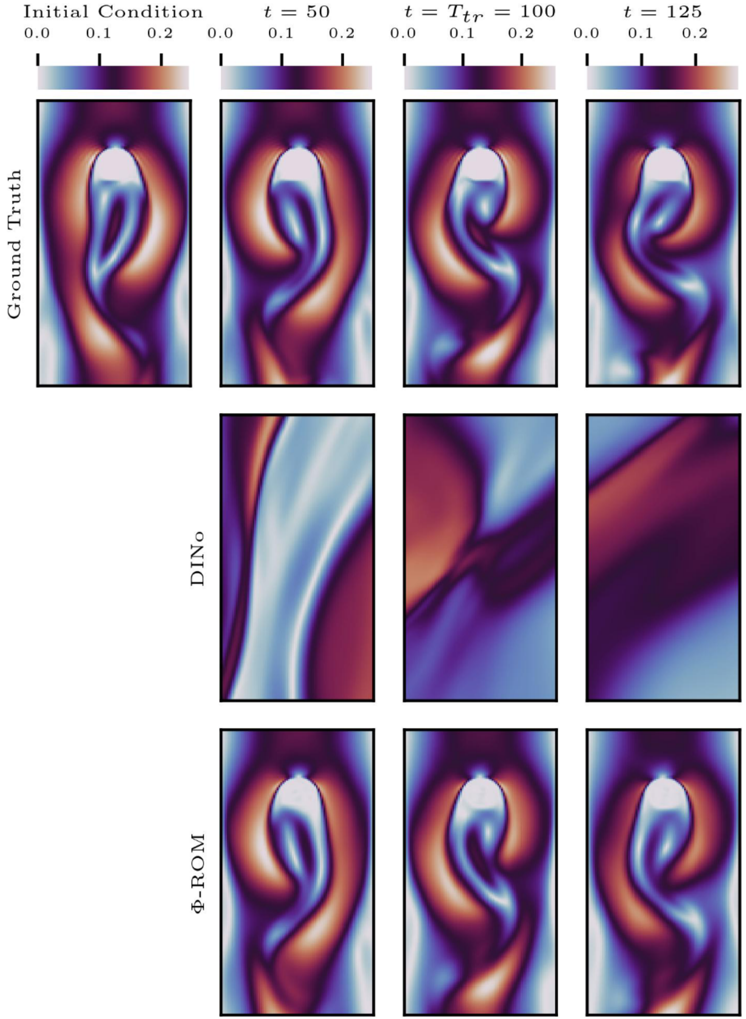


AUTODESK
Research

Results: Parameter Extrapolation

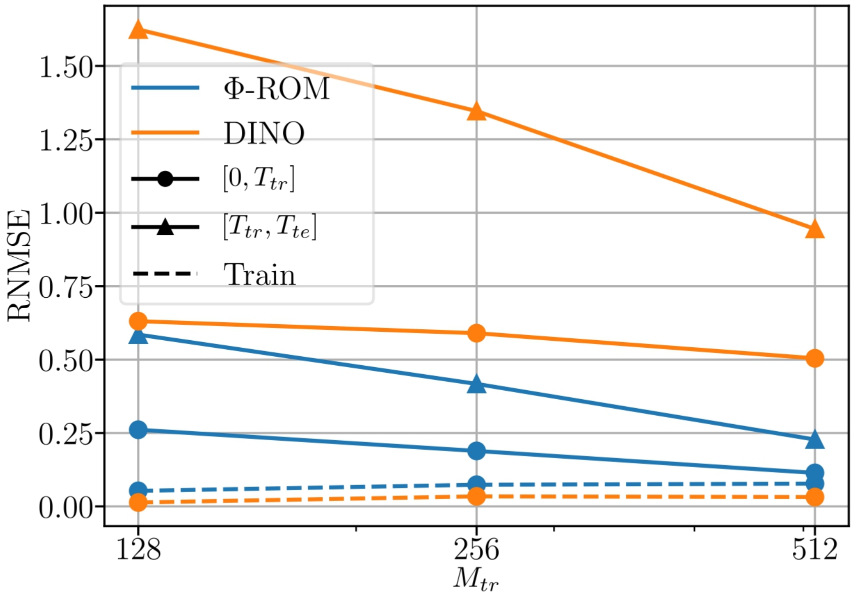
Re extrapolation is improved by about 4 times

Time	$[0, T_{tr}]$		$[T_{tr}, T_{te}]$	
β	β_{tr}	β_{te}	β_{tr}	β_{te}
	$\mathcal{X}_{tr} = \mathcal{X}_{\mathcal{S}} = \mathcal{X}_{te}$			
Φ-ROM	0.049	**0.115**	0.116	**0.180**
DINo	**0.011**	0.457	**0.108**	0.566



AUTODESK
Research

Results: Data Efficiency

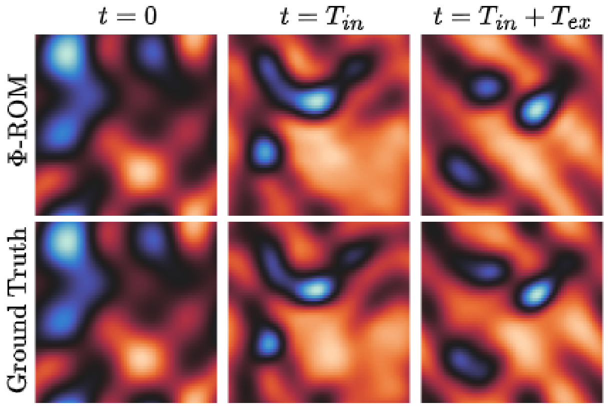


Summary and Conclusions

- **Physics-informed** training:
  - Differentiable solver as a building block of ML

- Φ-ROM:
  - Generalizes better to unseen initial conditions or parameters
  - Has improved temporal stability
  - Extrapolates beyond training time horizon

- More in our paper:
  - Robust to various physical phenomena (e.g. shocks)
  - Works nicely with sparse observations
  - Mesh-free
  - Continuous in space and time



AUTODESK
Research

Thank you!

AUTODESK
Research