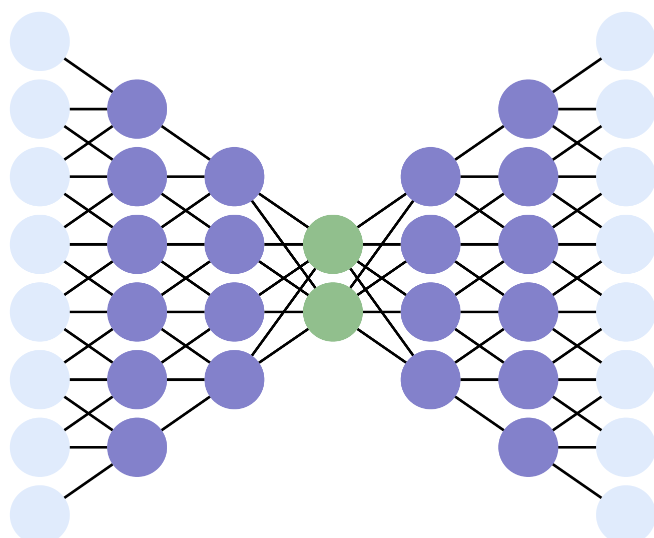
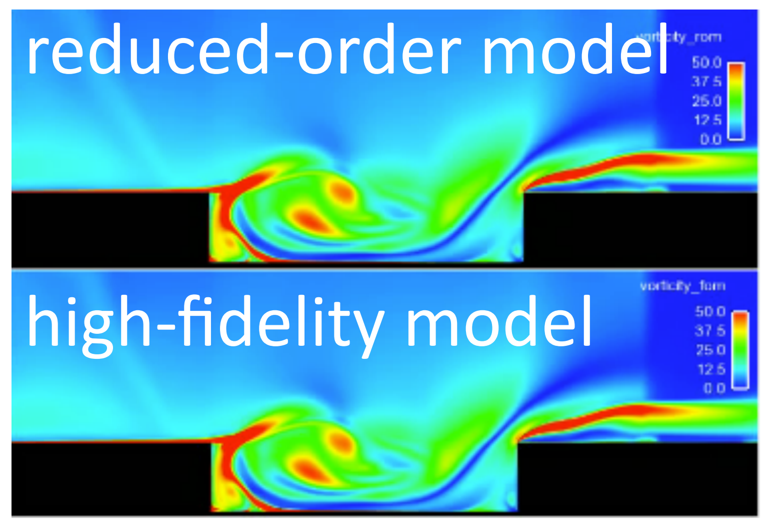
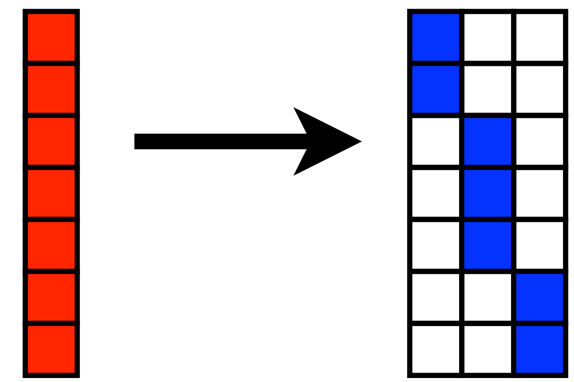
# Nonlinear model reduction

Using machine learning to enable rapid simulation of extreme-scale physics models





*Kookjin Lee*

*Eric Parish*

**Kevin Carlberg**

University of Washington

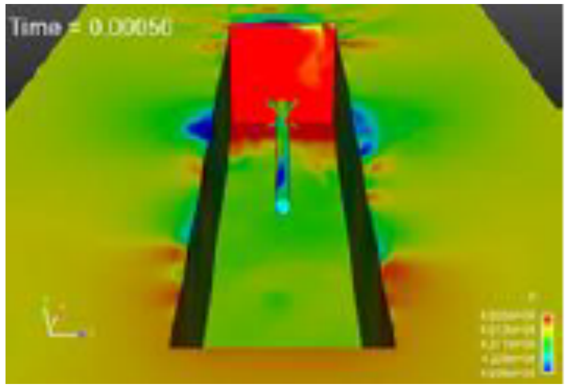
*Exuberance of Machine Learning in Transport Phenomena*

Dallas, Texas

February 10, 2020

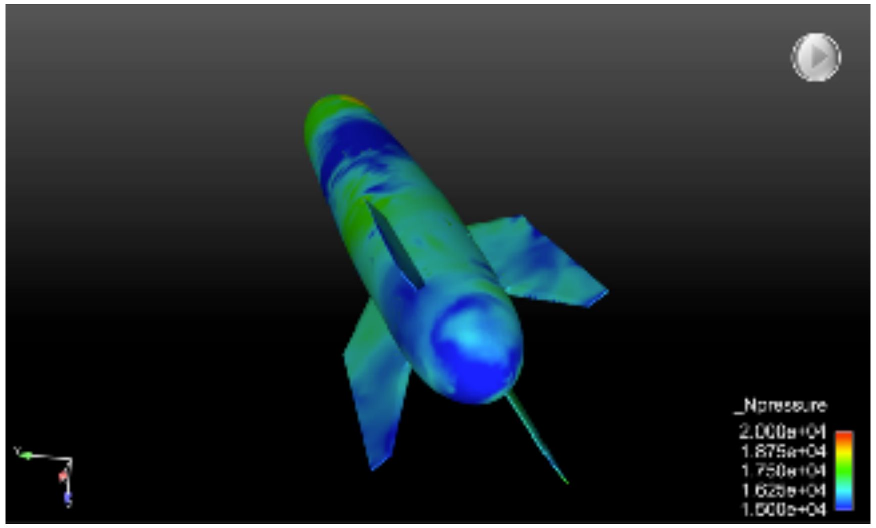
# High-fidelity simulation

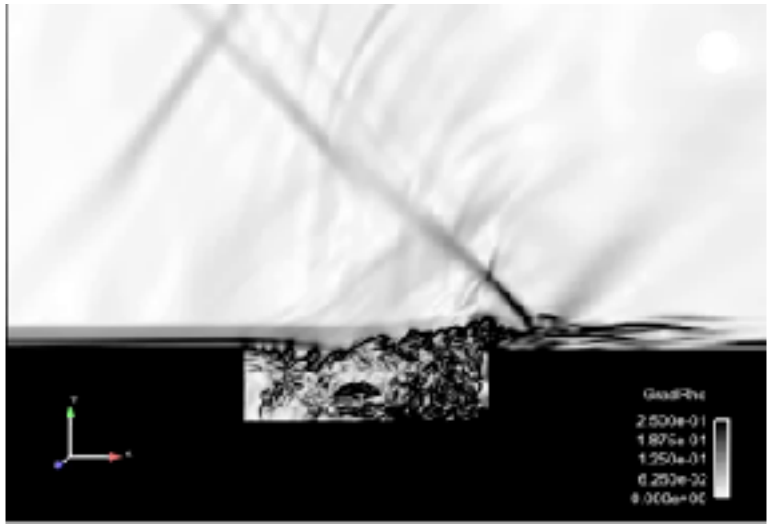
+ Indispensable in science and engineering
- Extreme-scale models required for high fidelity



# High-fidelity simulation

+ Indispensable in science and engineering
- Extreme-scale models required for high fidelity





+ *High fidelity*: matches wind-tunnel experiments to within 5%
- *Extreme scale*: 100 million cells, 200,000 time steps
- *High simulation costs*: 6 weeks, 5000 cores

**computational barrier**

# Many-query problems

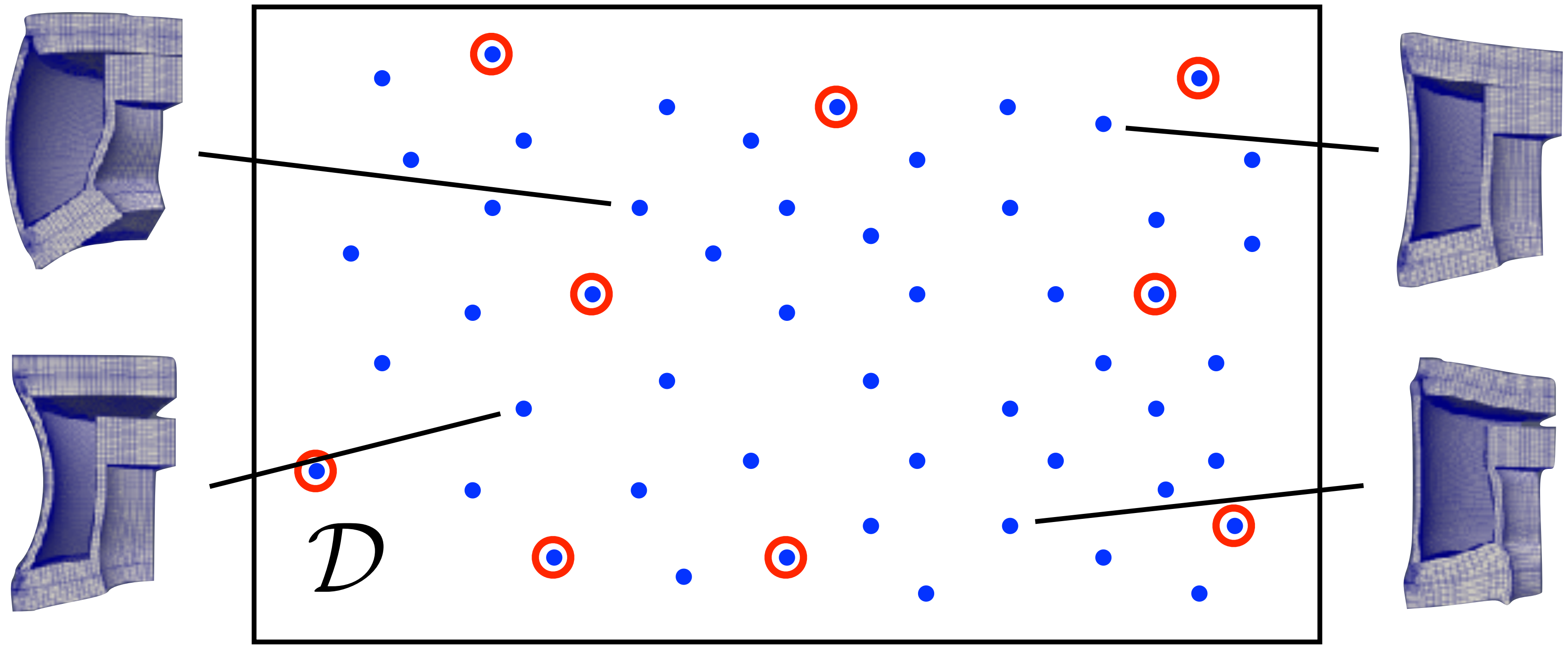
- uncertainty propagation
- Bayesian inference
- stochastic optimization

***Goal: break computational barrier***

# **Approach**: exploit simulation data

$$\text{ODE:} \quad \frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}; t, \boldsymbol{\mu}), \quad \mathbf{x}(0, \boldsymbol{\mu}) = \mathbf{x}_0(\boldsymbol{\mu}), \quad t \in [0, T_{\text{final}}], \quad \boldsymbol{\mu} \in \mathcal{D}$$

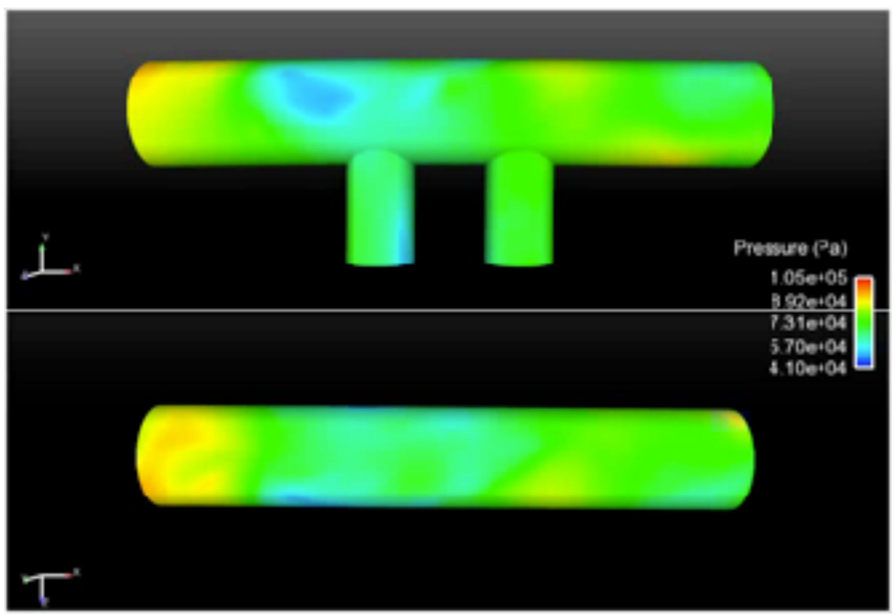
***Many-query problem**: solve ODE for* $\boldsymbol{\mu} \in \mathcal{D}_{\text{query}}$

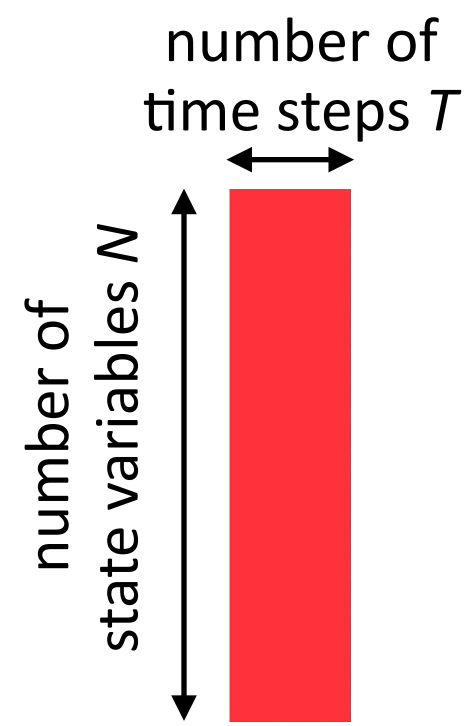


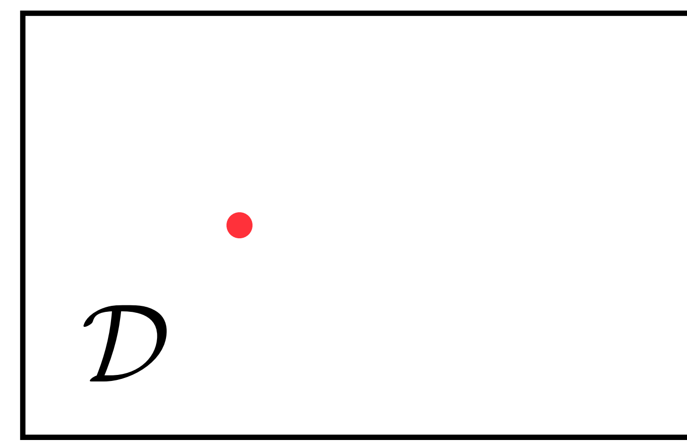

***Idea**: exploit simulation data collected at a few points*

1. *Training:* Solve ODE for $\mu \in \mathcal{D}_{\text{training}}$ and collect simulation data
2. *Machine learning:* Identify structure in data
3. *Reduction:* Reduce the cost of solving ODE for $\mu \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$

$$\text{ODE:} \quad \frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}; t, \boldsymbol{\mu})$$

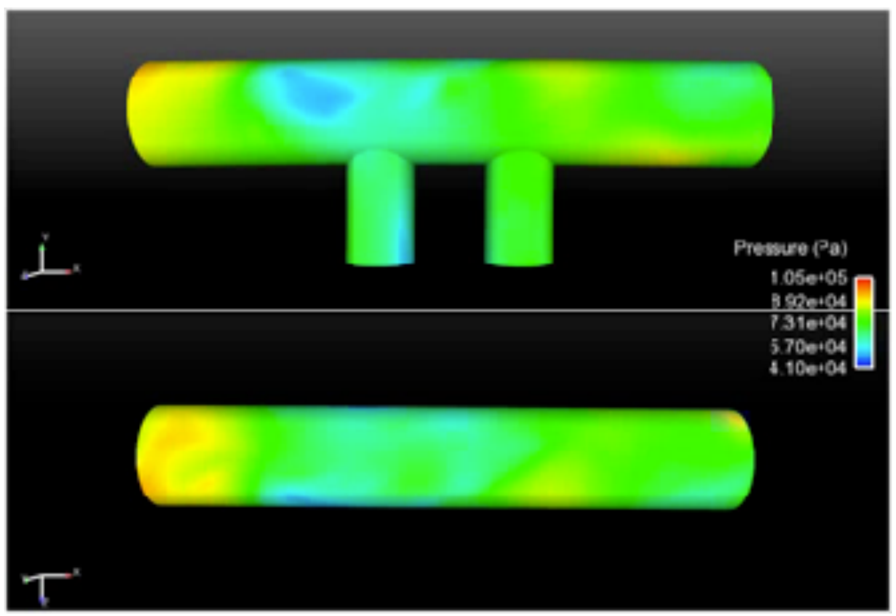


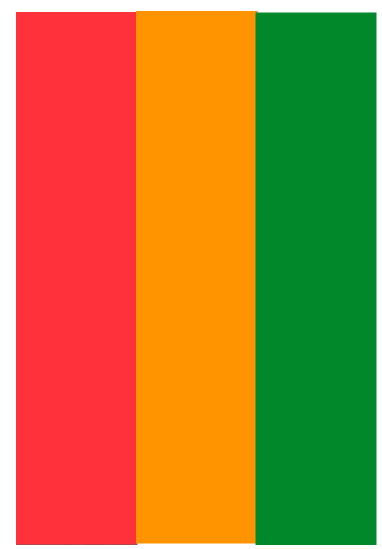



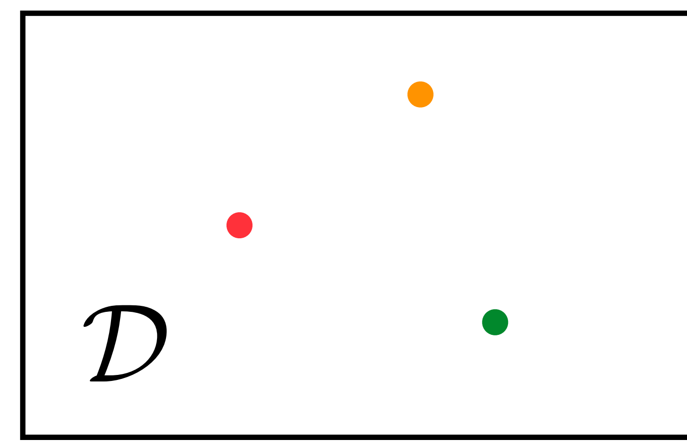



1. *Training:* Solve ODE for $\mu \in \mathcal{D}_{\text{training}}$ and collect simulation data
2. *Machine learning:* Identify structure in data
3. *Reduction:* Reduce the cost of solving ODE for $\mu \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$

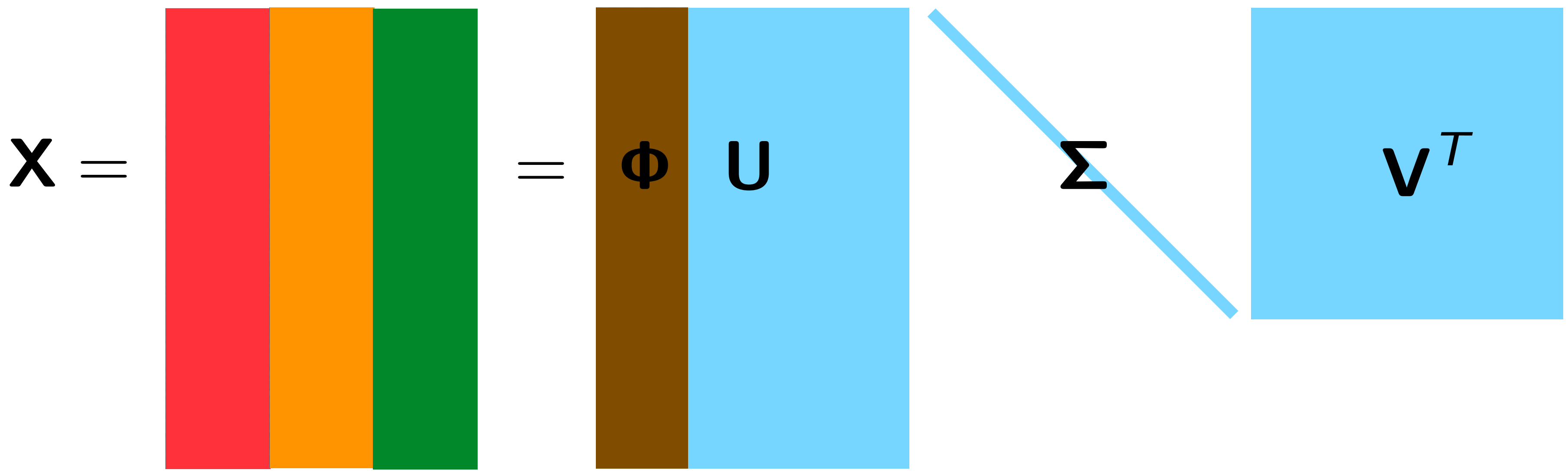
$$\text{ODE:} \qquad \frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}; t, \mu)$$




$\mathbf{X} =$ 



1. *Training:* Solve ODE for $\mu \in \mathcal{D}_{\text{training}}$ and collect simulation data
2. *Machine learning:* Identify structure in data
3. *Reduction:* Reduce the cost of solving ODE for $\mu \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$




$\boldsymbol{\Phi}$ *columns are principal components of the spatial simulation data*

1. *Training:* Solve ODE for $\mu \in \mathcal{D}_{\text{training}}$ and collect simulation data
2. *Machine learning:* Identify structure in data
3. *Reduction:* Reduce the cost of solving ODE for $\boldsymbol{\mu} \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$

$$\mathbf{x}(t) \approx \tilde{\mathbf{x}}(t) = \boldsymbol{\Phi}\,\hat{\mathbf{x}}(t)$$

*ODE*

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}; t)$$

$$\mathbf{r}\left(\frac{d\mathbf{x}}{dt}, \mathbf{x}, t\right) = \mathbf{0}$$

*residual minimization* →

*Galerkin ODE*

$$\frac{d\hat{\mathbf{x}}}{dt} = \boldsymbol{\Phi}^T\mathbf{f}(\boldsymbol{\Phi}\hat{\mathbf{x}}, t)$$

$$\boldsymbol{\Phi}\frac{d\hat{\mathbf{x}}}{dt}(\boldsymbol{\Phi}\hat{\mathbf{x}}, t) = \underset{\mathbf{v}\in\text{range}(\boldsymbol{\Phi})}{\arg\min}\, \|\mathbf{r}(\mathbf{v}, \boldsymbol{\Phi}\hat{\mathbf{x}}, t)\|_2$$

*time discretization* ↓

*time discretization* ↓

*LSPG OΔE*

[C., Bou-Mosleh, Farhat, 2011]

$$\boldsymbol{\Phi}\hat{\mathbf{x}}^n = \underset{\mathbf{v}\in\text{range}(\boldsymbol{\Phi})}{\arg\min}\, \|\mathbf{r}^n(\mathbf{v})\|_2$$
$$n = 1, \ldots, T$$

← *residual minimization*

*OΔE*

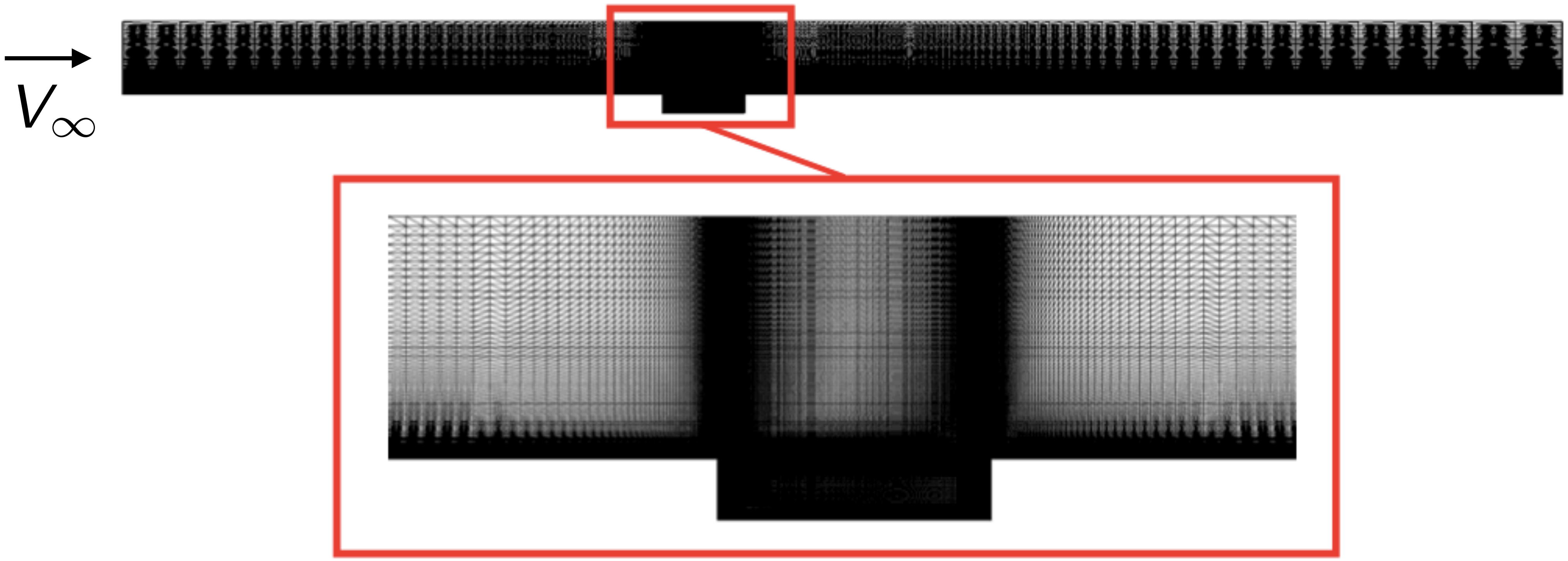
$$\mathbf{r}^n(\mathbf{x}^n) = 0$$
$$n = 1, \ldots, T$$

*Galerkin OΔE*

$$\boldsymbol{\Phi}^T\mathbf{r}^n(\boldsymbol{\Phi}\hat{\mathbf{x}}^n) = 0$$
$$n = 1, \ldots, T$$

- ODE residual: $\mathbf{r}(\mathbf{v}, \mathbf{x}, t) := \mathbf{v} - \mathbf{f}(\mathbf{x}, t)$
- OΔE residual: $\mathbf{r}^n(\mathbf{w}) := \alpha_0\mathbf{w} - \Delta t\beta_0\mathbf{f}(\mathbf{w}, t^n) + \sum_{j=1}^{k}\alpha_j\mathbf{x}^{n-j} - \Delta t\sum_{j=1}^{k}\beta_j\mathbf{f}(\mathbf{x}^{n-j}, t^{n-j})$
- Other residual-minimizing ROMs [LeGresley and Alonso, 2000; Bui-Thanh et al., 2008; Bui-Thanh et al., 2008; Constantine and Wang, 2012; Choi and C.; 2019; **Parish and C., 2019**]

# Captive carry




- Unsteady Navier–Stokes
- Re = 6.3 x $10^6$
- $M_\infty = 0.6$

**Spatial discretization**

- 2nd-order finite volume
- DES turbulence model
- $1.2 \times 10^6$ degrees of freedom

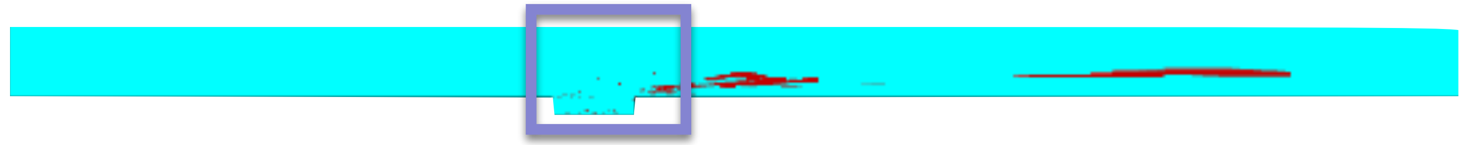
**Temporal discretization**

- 2nd-order BDF
- Verified time step $\Delta t = 1.5 \times 10^{-3}$
- $8.3 \times 10^3$ time instances

# LSPG ROM with sample mesh [C., Farhat, Cortial, Amsallem, 2013]

$$\boldsymbol{\Phi}\hat{\mathbf{x}}^{n} = \underset{\mathbf{v}\in\mathrm{range}(\boldsymbol{\Phi})}{\arg\min} \left\|\mathbf{r}^{n}(\mathbf{v})\right\|_{\Theta}$$

sample mesh

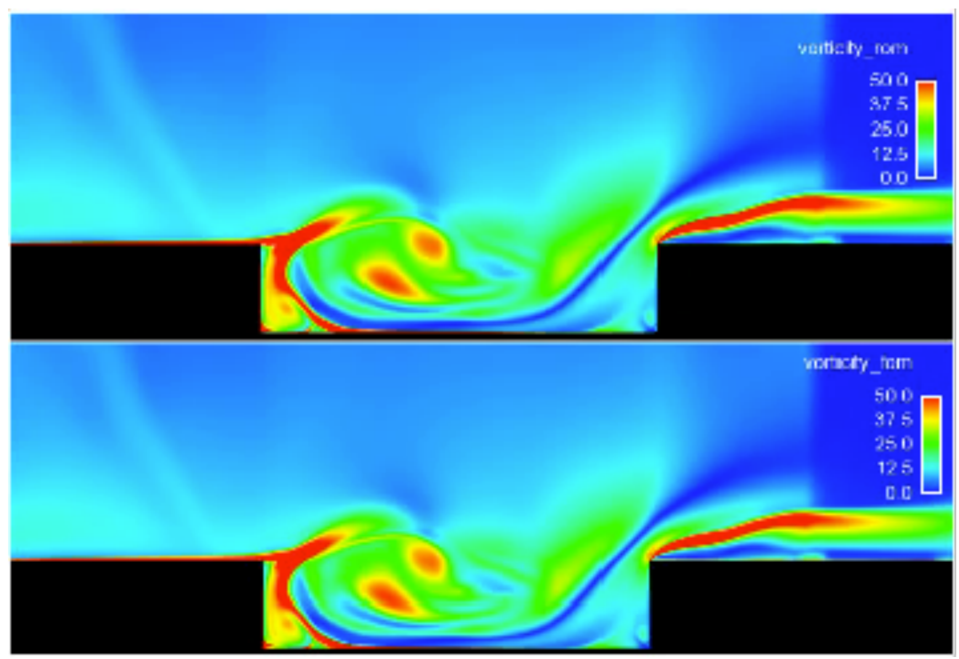


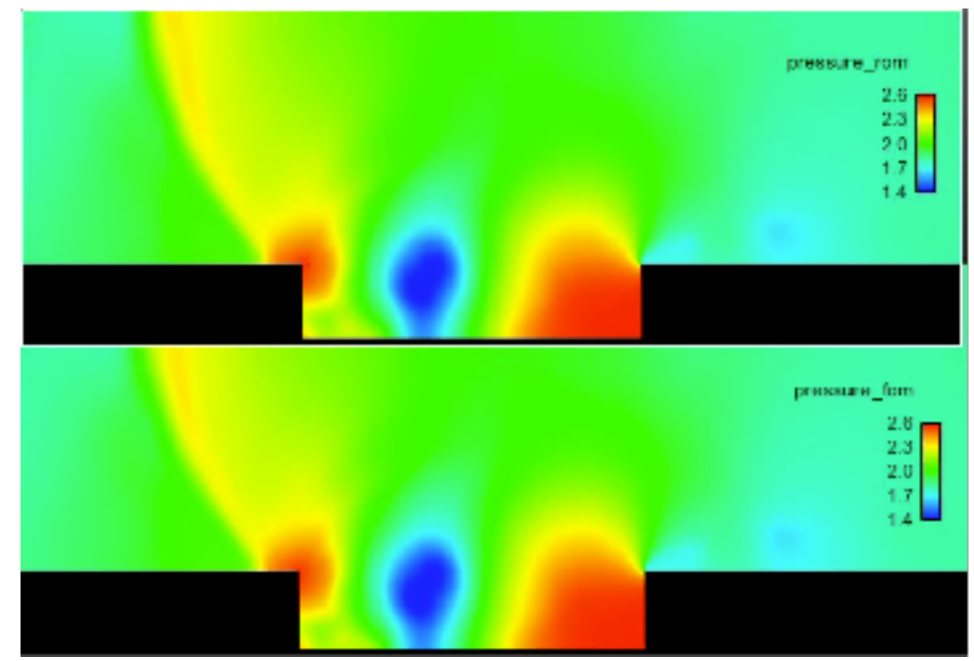
+ *HPC on a laptop*

*vorticity field* | *pressure field*

LSPG ROM
32 min, 2 cores

high-fidelity
5 hours, 48 cores




+ 229x savings in core–hours
+ < 1% error in time-averaged drag

***… so why doesn't everyone use ROMs?***

# Outstanding challenges in model reduction

## 1) Linear-subspace assumption is strong

$$\mathbf{x}(t) \approx \tilde{\mathbf{x}}(t) = \boldsymbol{\Phi}\, \hat{\mathbf{x}}(t)$$

- Lee and C. “Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders.” J Comp Phys, 404:108973, 2020.

## 2) Important physical properties not satisfied

*Galerkin*

$$\boldsymbol{\Phi}\frac{d\hat{\mathbf{x}}}{dt}(\mathbf{x}, t) = \underset{\mathbf{v}\in\text{range}(\boldsymbol{\Phi})}{\text{argmin}} \|\mathbf{r}(\mathbf{v}, \mathbf{x}; t)\|_2$$

*LSPG*

$$\boldsymbol{\Phi}\hat{\mathbf{x}}^n = \underset{\mathbf{v}\in\text{range}(\boldsymbol{\Phi})}{\arg\min} \|\mathbf{r}^n(\mathbf{v})\|_2$$

- C., Choi, and Sargsyan. “Conservative model reduction for finite-volume models.” J Comp Phys, 371:280–314, 2018.
- Lee and C. “Deep conservation: A latent dynamics model for exact satisfaction of physical conservation laws .” arXiv e-print 1909.09754, 2019.

## 3) Error analysis difficult

- Freno and C. “Machine-learning error models for approximate solutions to parameterized systems of nonlinear equations.” CMAME, 348:250–296, 2019.
- Parish and C. “Time-series machine-learning error models for approximate solutions to parameterized dynamical systems.” arXiv e-print, (1907.11822).

# Outstanding challenges in model reduction

## 1) Linear-subspace assumption is strong

$$\mathbf{x}(t) \approx \tilde{\mathbf{x}}(t) = \boldsymbol{\Phi}\,\hat{\mathbf{x}}(t)$$

- Lee and C. "Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders." J Comp Phys, 404:108973, 2020.

## 2) Important physical properties not guaranteed

*Galerkin*

$$\boldsymbol{\Phi}\frac{d\hat{\mathbf{x}}}{dt}(\mathbf{x}, t) = \underset{\mathbf{v}\in\text{range}(\boldsymbol{\Phi})}{\text{argmin}} \|\mathbf{r}(\mathbf{v}, \mathbf{x}; t)\|_2$$

*LSPG*

$$\boldsymbol{\Phi}\hat{\mathbf{x}}^n = \underset{\mathbf{v}\in\text{range}(\boldsymbol{\Phi})}{\arg\min} \|\mathbf{r}^n(\mathbf{v})\|_2$$

- C., Choi, and Sargsyan. "Conservative model reduction for finite-volume models." J Comp Phys, 371:280–314, 2018.
- Lee and C. "Deep conservation: A latent dynamics model for exact satisfaction of physical conservation laws ." arXiv e-print 1909.09754, 2019.

## 3) Error analysis difficult

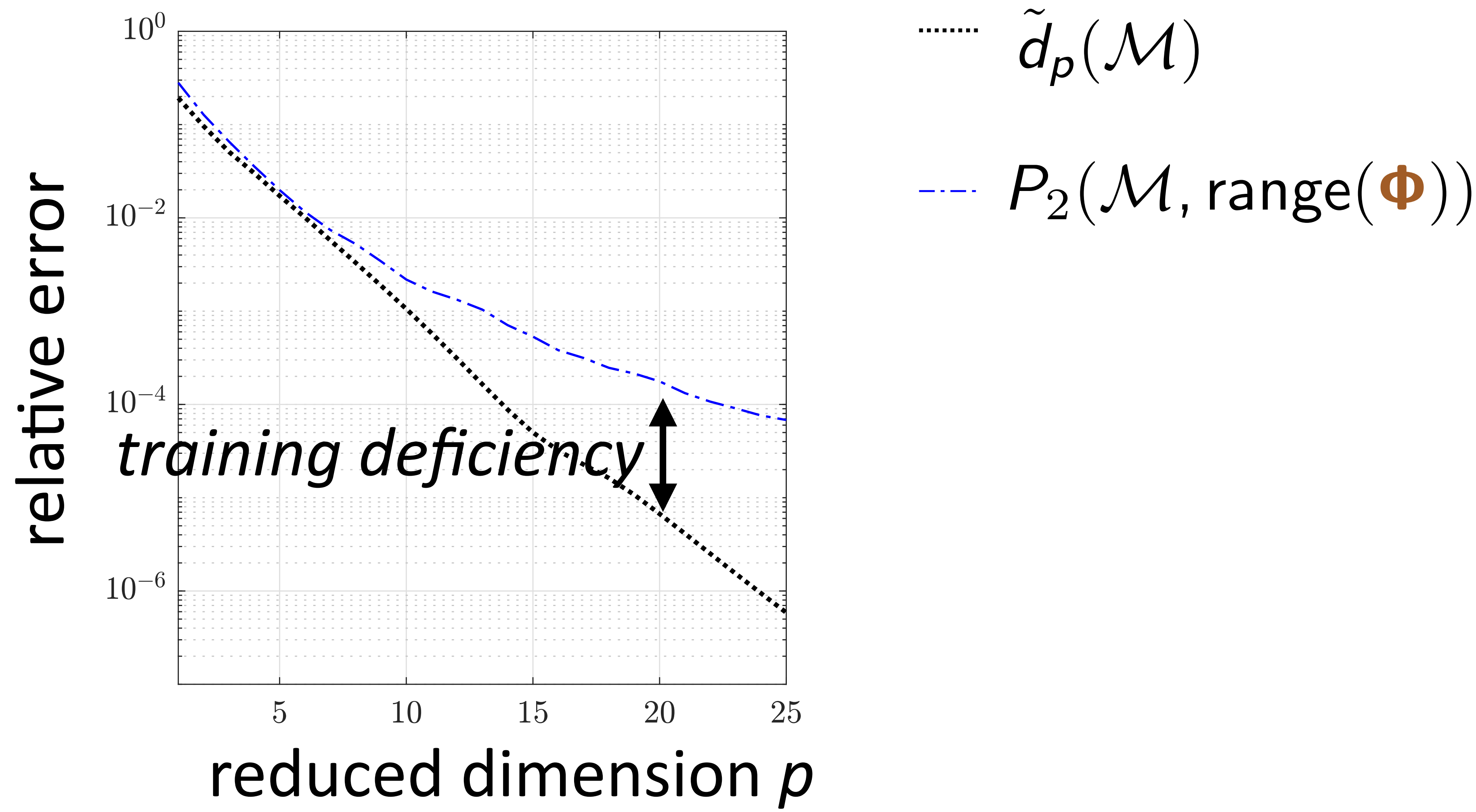
- Freno and C. "Machine-learning error models for approximate solutions to parameterized systems of nonlinear equations." CMAME, 348:250–296, 2019.
- Parish and C. "Time-series machine-learning error models for approximate solutions to parameterized dynamical systems." arXiv e-print, (1907.11822).

# Kolmogorov-width limitation of linear subspaces

- $\mathcal{M} := \{\mathbf{x}(t, \boldsymbol{\mu}) \mid t \in [0, T_{\text{final}}],\ \boldsymbol{\mu} \in \mathcal{D}\}$: solution manifold
- $\mathcal{S}_p$ : set of all $p$-dimensional linear subspaces
- $d_p(\mathcal{M}) := \inf_{\mathcal{S} \in \mathcal{S}_p} P_\infty(\mathcal{M}, \mathcal{S})$, $P_\infty(\mathcal{M}, \mathcal{S}) := \sup_{\mathbf{x} \in \mathcal{M}} \inf_{\mathbf{y} \in \mathcal{S}} \|\mathbf{x} - \mathbf{y}\|$

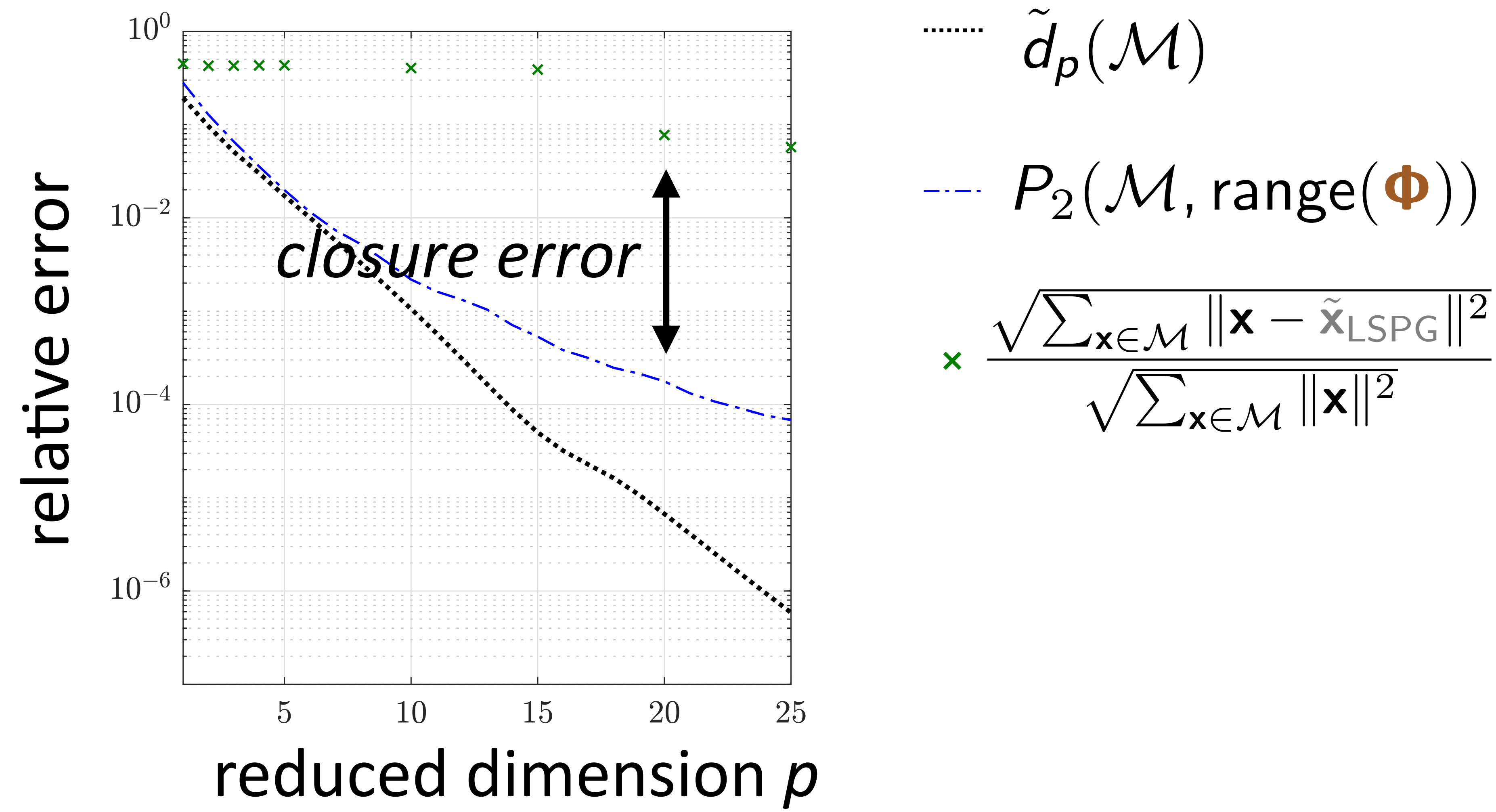
# Kolmogorov-width limitation of linear subspaces

- $\mathcal{M} := \{\mathbf{x}(t, \boldsymbol{\mu}) \mid t \in [0, T_{\text{final}}],\ \boldsymbol{\mu} \in \mathcal{D}\}$: solution manifold
- $\mathcal{S}_p$ : set of all $p$-dimensional linear subspaces
- $\tilde{d}_p(\mathcal{M}) := \inf_{\mathcal{S} \in \mathcal{S}_p} P_2(\mathcal{M}, \mathcal{S})$ , $P_2(\mathcal{M}, \mathcal{S}) := \sqrt{\sum_{\mathbf{x} \in \mathcal{M}} \inf_{\mathbf{y} \in \mathcal{S}} \|\mathbf{x} - \mathbf{y}\|^2} / \sqrt{\sum_{\mathbf{x} \in \mathcal{M}} \|\mathbf{x}\|^2}$



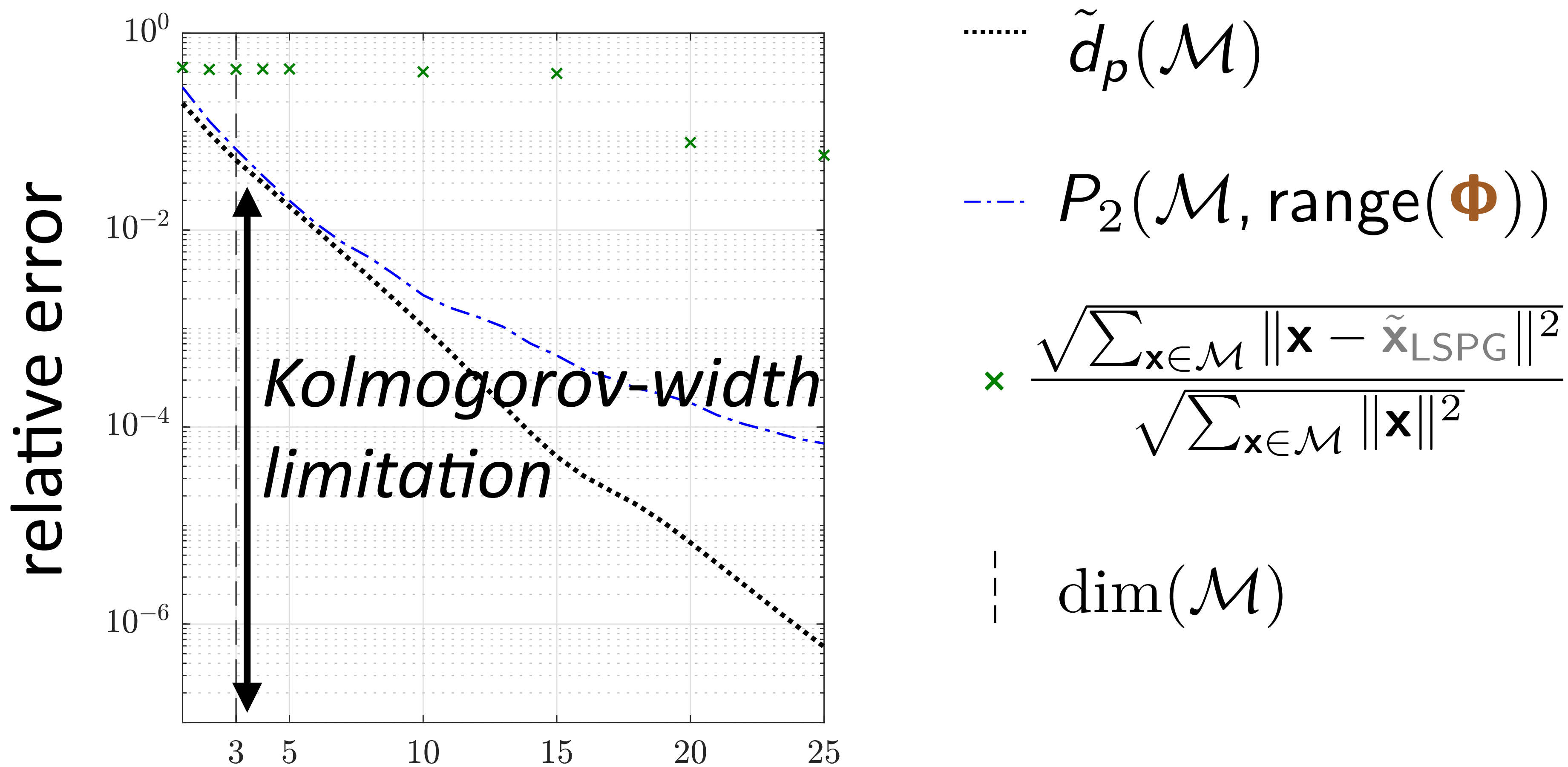
# Kolmogorov-width limitation of linear subspaces

- $\mathcal{M} := \{\mathbf{x}(t,\boldsymbol{\mu}) \mid t \in [0, T_{\text{final}}],\ \boldsymbol{\mu} \in \mathcal{D}\}$: solution manifold
- $\mathcal{S}_p$ : set of all $p$-dimensional linear subspaces
- $\tilde{d}_p(\mathcal{M}) := \inf\limits_{\mathcal{S} \in \mathcal{S}_p} P_2(\mathcal{M},\mathcal{S})$ , $P_2(\mathcal{M},\mathcal{S}) := \sqrt{\sum\limits_{\mathbf{x}\in\mathcal{M}} \inf\limits_{\mathbf{y}\in\mathcal{S}} \|\mathbf{x}-\mathbf{y}\|^2} \Big/ \sqrt{\sum\limits_{\mathbf{x}\in\mathcal{M}} \|\mathbf{x}\|^2}$



# Kolmogorov-width limitation of linear subspaces

- $\mathcal{M} := \{\mathbf{x}(t, \boldsymbol{\mu}) \mid t \in [0, T_{\text{final}}], \boldsymbol{\mu} \in \mathcal{D}\}$: solution manifold
- $\mathcal{S}_p$ : set of all $p$-dimensional linear subspaces
- $\tilde{d}_p(\mathcal{M}) := \inf_{\mathcal{S} \in \mathcal{S}_p} P_2(\mathcal{M}, \mathcal{S})$ , $P_2(\mathcal{M}, \mathcal{S}) := \sqrt{\sum_{\mathbf{x} \in \mathcal{M}} \inf_{\mathbf{y} \in \mathcal{S}} \|\mathbf{x} - \mathbf{y}\|^2} / \sqrt{\sum_{\mathbf{x} \in \mathcal{M}} \|\mathbf{x}\|^2}$




- Kolmogorov-width limitation: significant error for $p = \dim(\mathcal{M})$

***Goal**: overcome limitation via projection onto a nonlinear manifold*

# Overcoming Kolmogorov-width limitation

**Transform/update the linear subspace**

[Ohlberger and Rave, 2013; Iollo and Lombardi, 2014; Gerbeau and Lombardi, 2014; Peherstorfer and Willcox, 2015; Welper, 2017; Mojgani and Balajewicz, 2017; Reiss et al., 2018; Zimmermann et al., 2018; Peherstorfer, 2018; Rim and Mandli, 2018; Rim and Mandli, 2018; Nair and Balajewicz, 2019; Cagniart et al., 2019]

+ Can work much better than a fixed basis
- Some require problem-specific knowledge or characteristics
- Do not consider manifolds of general nonlinear structure

# Overcoming Kolmogorov-width limitation

**Transform/update the linear subspace**

[Ohlberger and Rave, 2013; Iollo and Lombardi, 2014; Gerbeau and Lombardi, 2014; Peherstorfer and Willcox, 2015; Welper, 2017; Mojgani and Balajewicz, 2017; Reiss et al., 2018; Zimmermann et al., 2018; Peherstorfer, 2018; Rim and Mandli, 2018; Rim and Mandli, 2018; Nair and Balajewicz, 2019; Cagniart et al., 2019]

+ Can work much better than a fixed basis
- Some require problem-specific knowledge or characteristics
- Do not consider manifolds of general nonlinear structure

***A priori* construction of local linear subspaces**

[Dihlmann et al., 2011; Drohmann et al., 2011; Amsallem, Zahr, Farhat, 2012; Peherstorfer et el., 2014; Taddei et al., 2015]

+ Tailored bases for local regions of space/time domain, state space
- Do not consider manifolds of general nonlinear structure

# Overcoming Kolmogorov-width limitation

## Transform/update the linear subspace

[Ohlberger and Rave, 2013; Iollo and Lombardi, 2014; Gerbeau and Lombardi, 2014; Peherstorfer and Willcox, 2015; Welper, 2017; Mojgani and Balajewicz, 2017; Reiss et al., 2018; Zimmermann et al., 2018; Peherstorfer, 2018; Rim and Mandli, 2018; Rim and Mandli, 2018; Nair and Balajewicz, 2019; Cagniart et al., 2019]

+ Can work much better than a fixed basis
- Some require problem-specific knowledge or characteristics
- Do not consider manifolds of general nonlinear structure

## *A priori* construction of local linear subspaces

[Dihlmann et al., 2011; Drohmann et al., 2011; Amsallem, Zahr, Farhat, 2012; Peherstorfer et el., 2014; Taddei et al., 2015]

+ Tailored bases for local regions of space/time domain, state space
- Do not consider manifolds of general nonlinear structure

## Model reduction on nonlinear manifolds [Gu, 2011; Kashima, 2016; Hartman and Mestha, 2017]

- Kinematically inconsistent [Kashima, 2016; Hartman and Mestha, 2017]
- Limited to piecewise linear manifolds [Gu, 2011]
- Solutions lack optimality [Gu, 2011; Kashima, 2016; Hartman and Mestha, 2017]

# Goals

**Overcome shortcomings of existing methods**

+ Enable manifolds with general nonlinear structure
+ Kinematically consistent
+ Satisfy optimality property

***Manifold Galerkin and LSPG projection***

**Practical nonlinear-manifold construction**

+ No problem-specific knowledge required
+ Use same training data as POD

***Deep convolutional autoencoders***

# Goals

**Overcome shortcomings of existing methods**

+ Enable manifolds with general nonlinear structure
+ Kinematically consistent
+ Satisfy optimality property

***Manifold Galerkin and LSPG projection***

**Practical nonlinear-manifold construction**

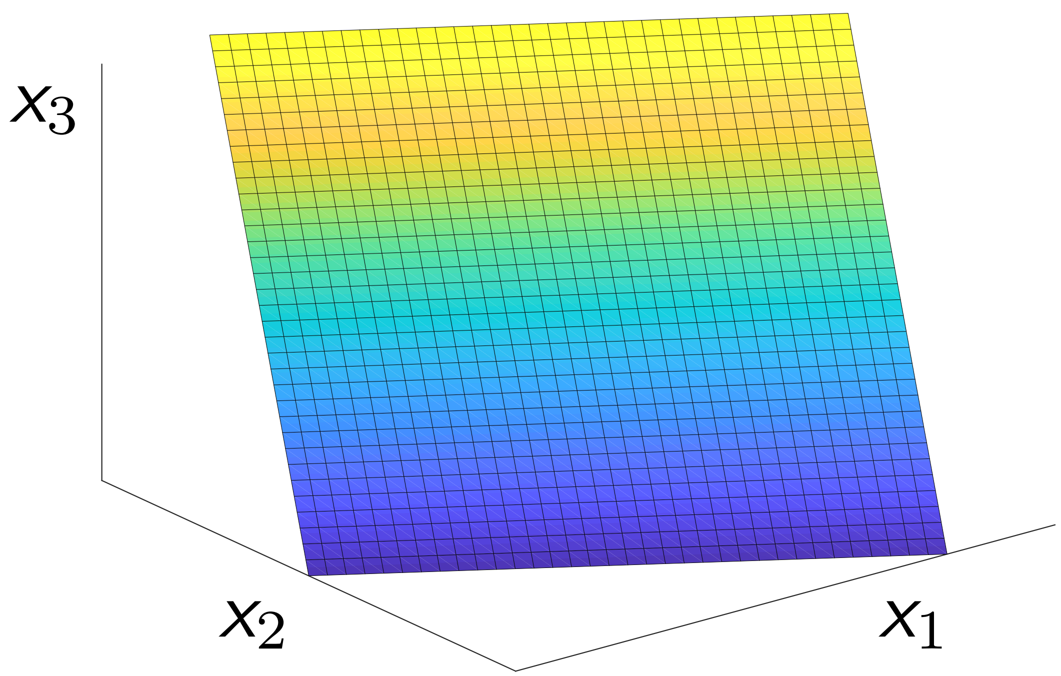
+ No problem-specific knowledge required
+ Use same training data as POD

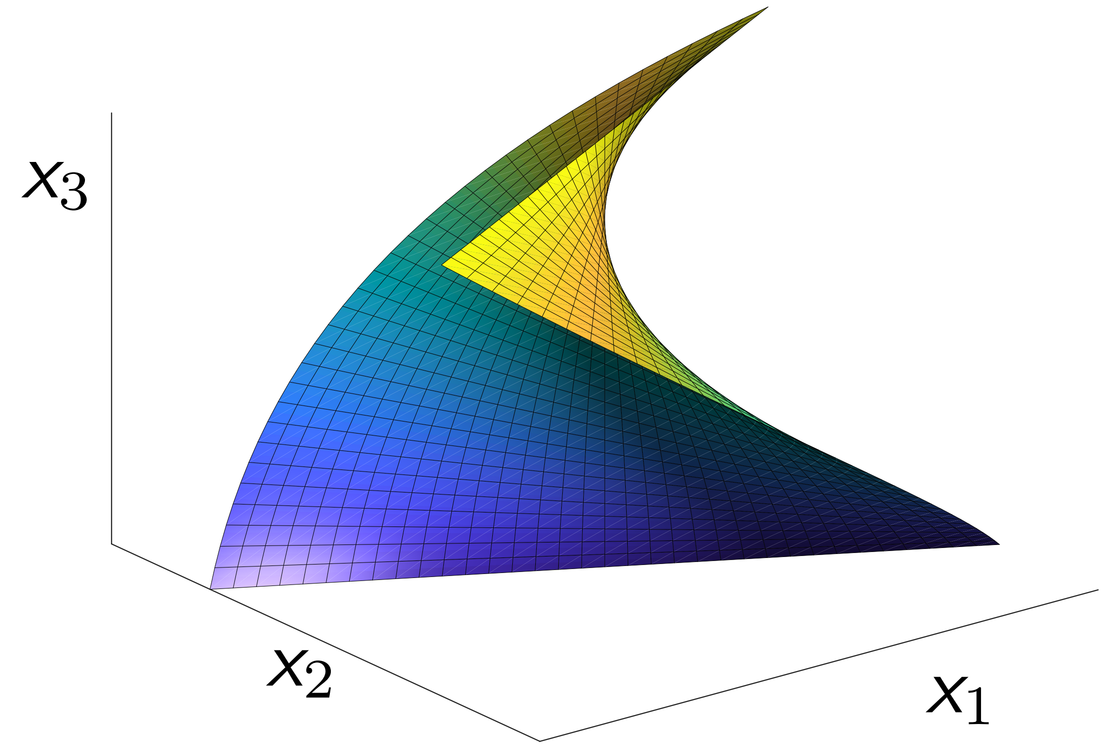
***Deep convolutional autoencoders***

# Nonlinear trial manifold

| | ***Linear trial subspace*** | ***Nonlinear trial manifold*** |
|---|---|---|
| | $\mathrm{range}(\mathbf{\Phi}) := \{\mathbf{\Phi}\hat{\mathbf{x}} \mid \hat{\mathbf{x}} \in \mathbb{R}^p\}$ | $\mathcal{S} := \{\mathbf{g}(\hat{\mathbf{x}}) \mid \hat{\mathbf{x}} \in \mathbb{R}^p\}$ |

*example*
*N=3*
*p=2*




*state*

$\mathbf{x}(t) \approx \tilde{\mathbf{x}}(t) = \mathbf{\Phi}\,\hat{\mathbf{x}}(t) \in \mathrm{range}(\mathbf{\Phi})$

$\mathbf{x}(t) \approx \tilde{\mathbf{x}}(t) = \mathbf{g}(\hat{\mathbf{x}}(t)) \in \mathcal{S}$

+ Manifold has general structure

*velocity*

$$\frac{d\mathbf{x}}{dt} \approx \frac{d\tilde{\mathbf{x}}}{dt} = \mathbf{\Phi}\frac{d\hat{\mathbf{x}}}{dt} \in \mathrm{range}(\mathbf{\Phi})$$

$$\frac{d\mathbf{x}}{dt} \approx \frac{d\tilde{\mathbf{x}}}{dt} = \nabla\mathbf{g}(\hat{\mathbf{x}})\frac{d\hat{\mathbf{x}}}{dt} \in T_{\hat{\mathbf{x}}}\mathcal{S}$$

+ Kinematically consistent

1. *Training:* Solve ODE for $\boldsymbol{\mu} \in \mathcal{D}_{\text{training}}$ and collect simulation data
2. *Machine learning:* Identify structure in data
3. *Reduction:* Reduce the cost of solving ODE for $\boldsymbol{\mu} \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$

| | ***Linear-subspace ROM*** | ***Nonlinear-manifold ROM*** |
|---|---|---|
| | Given $\boldsymbol{\Phi}$ | Given $\mathbf{g}(\hat{\mathbf{x}})$ |
| *Galerkin* | $\frac{d\hat{\mathbf{x}}}{dt} = \underset{\hat{\mathbf{v}} \in \mathbb{R}^p}{\operatorname{argmin}} \\|\mathbf{r}(\boldsymbol{\Phi}\hat{\mathbf{v}}, \boldsymbol{\Phi}\hat{\mathbf{x}}; t)\\|_2$ $\Updownarrow$ $\frac{d\hat{\mathbf{x}}}{dt} = \boldsymbol{\Phi}^T \mathbf{f}(\boldsymbol{\Phi}\hat{\mathbf{x}}; t)$ | $\frac{d\hat{\mathbf{x}}}{dt} = \underset{\hat{\mathbf{v}} \in \mathbb{R}^p}{\operatorname{argmin}} \\|\mathbf{r}(\nabla \mathbf{g}(\hat{\mathbf{x}})\hat{\mathbf{v}}, \mathbf{g}(\hat{\mathbf{x}}); t)\\|_2$ $\Updownarrow$ $\frac{d\hat{\mathbf{x}}}{dt} = \nabla \mathbf{g}(\hat{\mathbf{x}})^+ \mathbf{f}(\mathbf{g}(\hat{\mathbf{x}}); t)$ |
| *LSPG* | $\hat{\mathbf{x}}^n = \underset{\hat{\mathbf{v}} \in \mathbb{R}^p}{\operatorname{argmin}} \\|\mathbf{r}^n(\boldsymbol{\Phi}\hat{\mathbf{v}})\\|_2$ | $\hat{\mathbf{x}}^n = \underset{\hat{\mathbf{v}} \in \mathbb{R}^p}{\operatorname{argmin}} \\|\mathbf{r}^n(\mathbf{g}(\hat{\mathbf{v}}))\\|_2$ |

\+ Satisfy residual minimization

**Theorem** [Lee, C., 2020]

Manifold Galerkin and manifold LSPG are equivalent if
1. the nonlinear trial manifold $\mathcal{S}$ is twice continuously differentiable,
2. $\|\hat{\mathbf{x}}^{n-j} - \hat{\mathbf{x}}^n\| = O(\Delta t)$ for $n = 1, \ldots, T$ and $j = 1, \ldots, k$, and
3. the limit $\Delta t \to 0$ is taken.

# Error bound

**Theorem** [Lee, C., 2020]

If the following conditions hold:

1. $\mathbf{f}(\cdot; t)$ is Lipschitz continuous with Lipschitz constant $\kappa$
2. $\Delta t$ is small enough such that $0 < h := |\alpha_0| - |\beta_0|\kappa\Delta t$, then

$$\|\mathbf{x}^n - \mathbf{g}(\hat{\mathbf{x}}_{\mathrm{G}}^n)\|_2 \leq \frac{1}{h}\|\mathbf{r}_{\mathrm{G}}^n(\mathbf{g}(\hat{\mathbf{x}}_{\mathrm{G}}))\|_2 + \frac{1}{h}\sum_{\ell=1}^{k}|\gamma_\ell|\|\mathbf{x}^{n-\ell} - \mathbf{g}(\hat{\mathbf{x}}_{\mathrm{G}})\|_2$$

$$\|\mathbf{x}^n - \mathbf{g}(\hat{\mathbf{x}}_{\mathrm{LSPG}}^n)\|_2 \leq \frac{1}{h}\min_{\hat{\mathbf{v}}}\|\mathbf{r}_{\mathrm{LSPG}}^n(\mathbf{g}(\hat{\mathbf{v}}))\|_2 + \frac{1}{h}\sum_{\ell=1}^{k}|\gamma_\ell|\|\mathbf{x}^{n-\ell} - \mathbf{g}(\hat{\mathbf{x}}_{\mathrm{LSPG}})\|_2$$

\+ Manifold LSPG sequentially minimizes the error bound

***How to construct manifold*** $\mathcal{S} := \{\mathbf{g}(\hat{\mathbf{x}}) \mid \hat{\mathbf{x}} \in \mathbb{R}^p\}$ ***from training data?***

# Goals

**Overcome shortcomings of existing methods**

+ Enable manifolds with general nonlinear structure
+ Kinematically consistent
+ Satisfy optimality property

***Manifold Galerkin and LSPG projection***

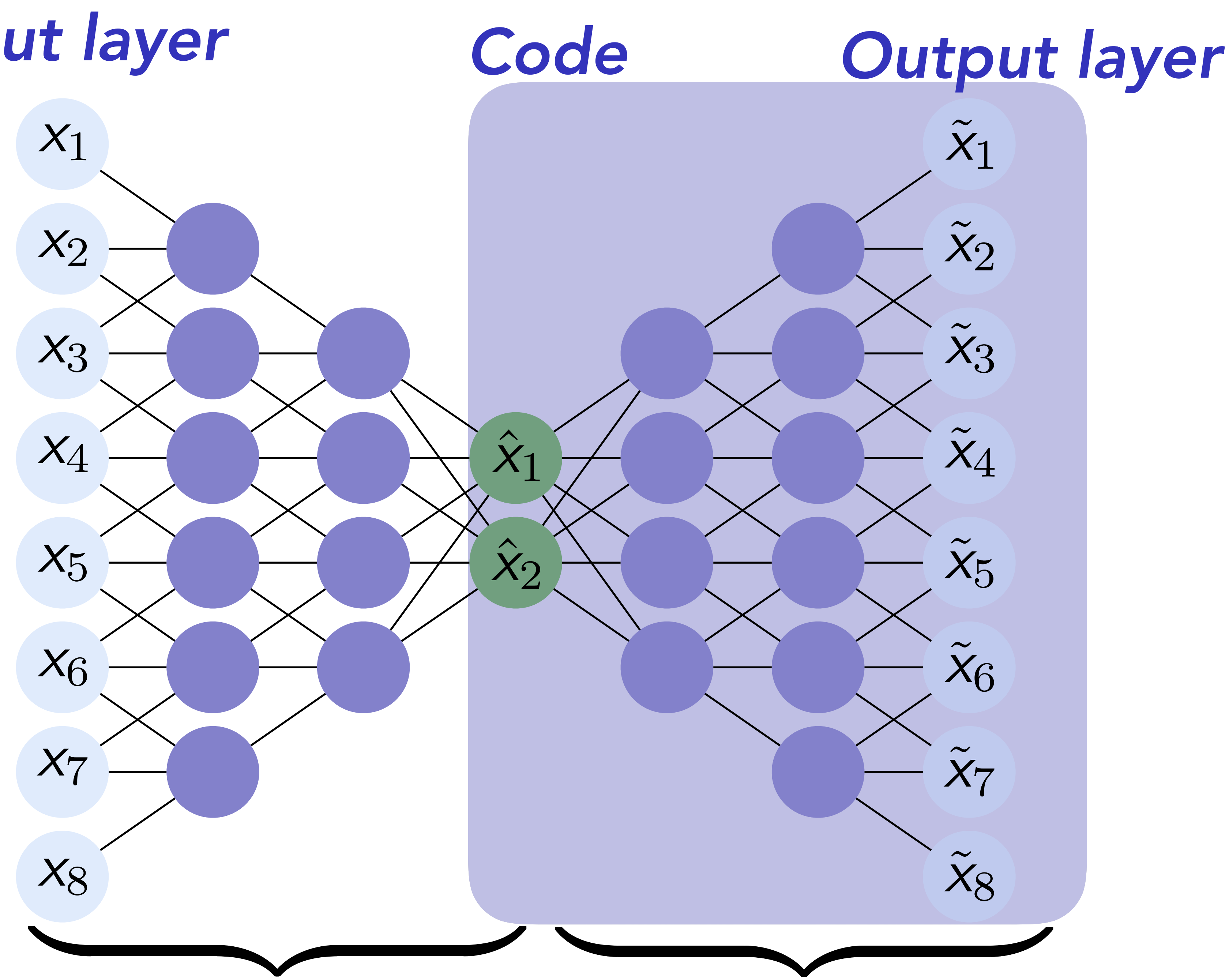
**Practical nonlinear-manifold construction**

+ No problem-specific knowledge required
+ Use same training data as POD

***Deep convolutional autoencoders***

$$\mathcal{S} := \{\mathbf{g}(\hat{\mathbf{x}}) \mid \hat{\mathbf{x}} \in \mathbb{R}^p\}$$

# Deep autoencoders

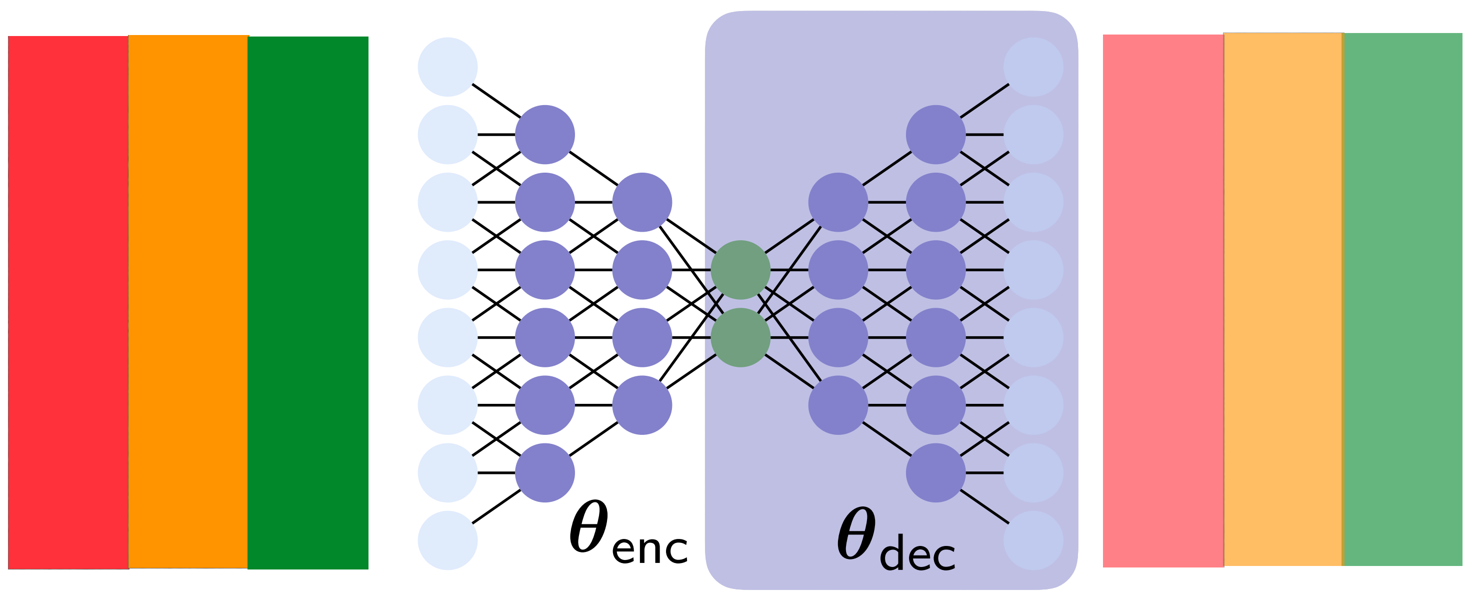



***Encoder*** $\mathbf{h}_{\mathrm{enc}}(\cdot;\boldsymbol{\theta}_{\mathrm{enc}})$ ***Decoder*** $\mathbf{h}_{\mathrm{dec}}(\cdot;\boldsymbol{\theta}_{\mathrm{dec}})$

$$\tilde{\mathbf{x}} = \mathbf{h}_{\mathrm{dec}}(\cdot;\boldsymbol{\theta}_{\mathrm{dec}}) \circ \mathbf{h}_{\mathrm{enc}}(\mathbf{x};\boldsymbol{\theta}_{\mathrm{enc}})$$

+ If $\tilde{\mathbf{x}} \approx \mathbf{x}$ for $\boldsymbol{\theta}^{\star}_{\mathrm{dec}}$, then $\mathbf{g} = \mathbf{h}_{\mathrm{dec}}(\cdot;\boldsymbol{\theta}^{\star}_{\mathrm{dec}})$ is accurate manifold parameterization

1. *Training:* Solve ODE for $\mu \in \mathcal{D}_{\text{training}}$ and collect simulation data
2. *Machine learning:* Identify structure in data
3. *Reduction:* Reduce the cost of solving ODE for $\mu \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$

$\mathbf{X} =$  $= \tilde{\mathbf{X}}(\boldsymbol{\theta})$



- Compute $\boldsymbol{\theta}^\star$ by approximately solving $\underset{\boldsymbol{\theta}}{\text{minimize}} \|\mathbf{X} - \tilde{\mathbf{X}}(\boldsymbol{\theta})\|_F$
- Define nonlinear trial manifold by setting $\mathbf{g} = \mathbf{h}_{\text{dec}}(\cdot; \boldsymbol{\theta}^\star_{\text{dec}})$

\+ Same snapshot data, no specialized problem knowledge

1. *Training:* Solve ODE for $\mu \in \mathcal{D}_{\text{training}}$ and collect simulation data
2. *Machine learning:* Identify structure in data
3. *Reduction:* Reduce the cost of solving ODE for $\boldsymbol{\mu} \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$

| | Subspace ROM | Manifold ROM |
|---|---|---|
| | Given $\boldsymbol{\Phi}$ | Given $\mathbf{g}(\hat{\mathbf{x}})$ |
| Galerkin | $\frac{d\hat{\mathbf{x}}}{dt} = \underset{\hat{\mathbf{v}} \in \mathbb{R}^p}{\operatorname{argmin}} \\|\mathbf{r}(\boldsymbol{\Phi}\hat{\mathbf{v}}, \boldsymbol{\Phi}\hat{\mathbf{x}}; t)\\|_2$ $\Updownarrow$ $\frac{d\hat{\mathbf{x}}}{dt} = \boldsymbol{\Phi}^T \mathbf{f}(\boldsymbol{\Phi}\hat{\mathbf{x}}; t)$ | $\frac{d\hat{\mathbf{x}}}{dt} = \underset{\hat{\mathbf{v}} \in \mathbb{R}^p}{\operatorname{argmin}} \\|\mathbf{r}(\nabla \mathbf{g}(\hat{\mathbf{x}})\hat{\mathbf{v}}, \mathbf{g}(\hat{\mathbf{x}}); t)\\|_2$ $\Updownarrow$ $\frac{d\hat{\mathbf{x}}}{dt} = \nabla \mathbf{g}(\hat{\mathbf{x}})^+ \mathbf{f}(\mathbf{g}(\hat{\mathbf{x}}); t)$ |
| LSPG | $\hat{\mathbf{x}}^n = \underset{\hat{\mathbf{v}} \in \mathbb{R}^p}{\operatorname{argmin}} \\|\mathbf{r}^n(\boldsymbol{\Phi}\hat{\mathbf{v}})\\|_2$ | $\hat{\mathbf{x}}^n = \underset{\hat{\mathbf{v}} \in \mathbb{R}^p}{\operatorname{argmin}} \\|\mathbf{r}^n(\mathbf{g}(\hat{\mathbf{v}}))\\|_2$ |

\+ Satisfy residual minimization

\+ Predictions directly integrate deep learning with computational physics

# Numerical results

## *1D Burgers' equation*

$$\frac{\partial w(x,t;\boldsymbol{\mu})}{\partial t} + \frac{\partial f(w(x,t;\boldsymbol{\mu}))}{\partial x} = 0.02e^{\alpha x}$$

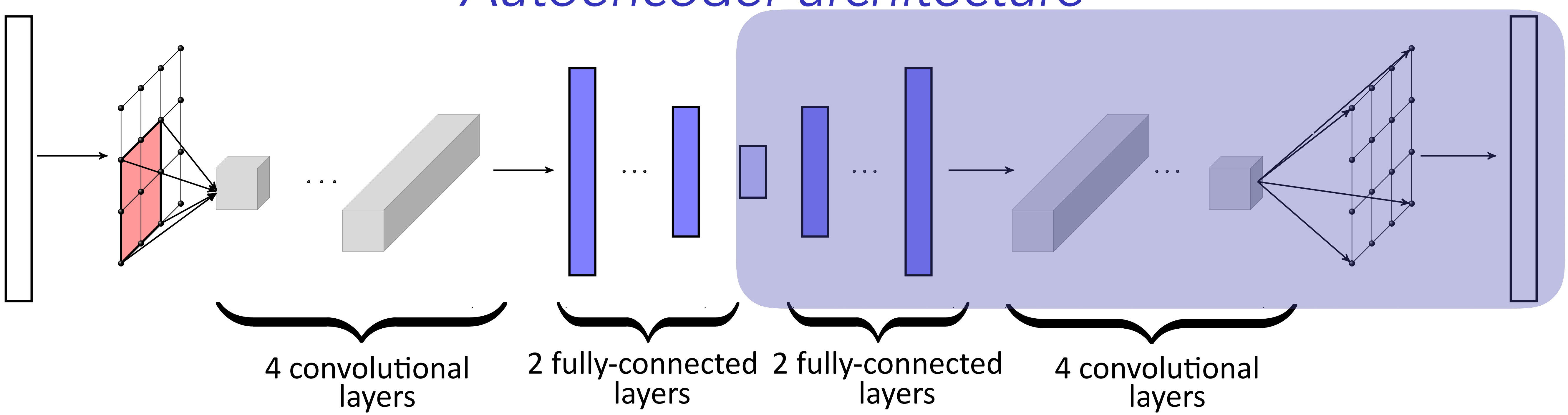
- $\boldsymbol{\mu}$: $\alpha$, inlet boundary condition
- *Spatial discretization*: finite volume
- *Time integrator:* backward Euler

## *2D reacting flow*

$$\frac{\partial \mathbf{w}(\vec{x},t;\boldsymbol{\mu})}{\partial t} = \nabla \cdot (\kappa \nabla \mathbf{w}(\vec{x},t;\boldsymbol{\mu})) - \mathbf{v} \cdot \nabla \mathbf{w}(\vec{x},t;\boldsymbol{\mu}) + \mathbf{q}(\mathbf{w}(\vec{x},t;\boldsymbol{\mu});\boldsymbol{\mu})$$

- $\boldsymbol{\mu}$: two terms in reaction
- *Spatial discretization*: finite difference
- *Time integrator*: BDF2

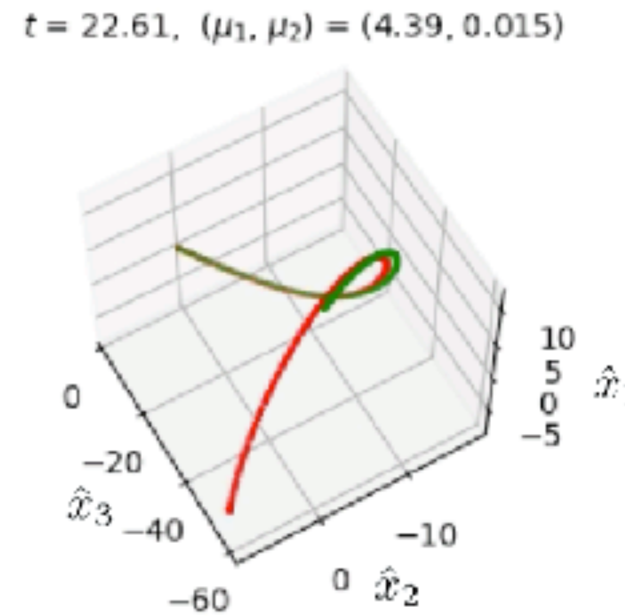
## *Autoencoder architecture*



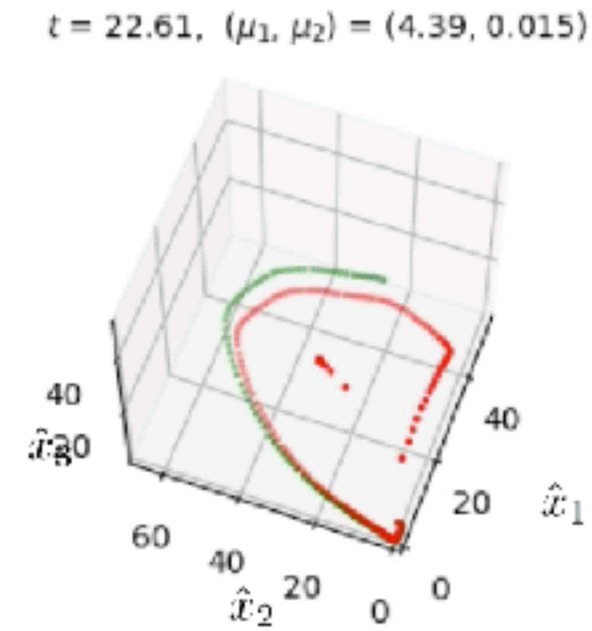
# Manifold interpretation: Burgers' equation

***FOM***

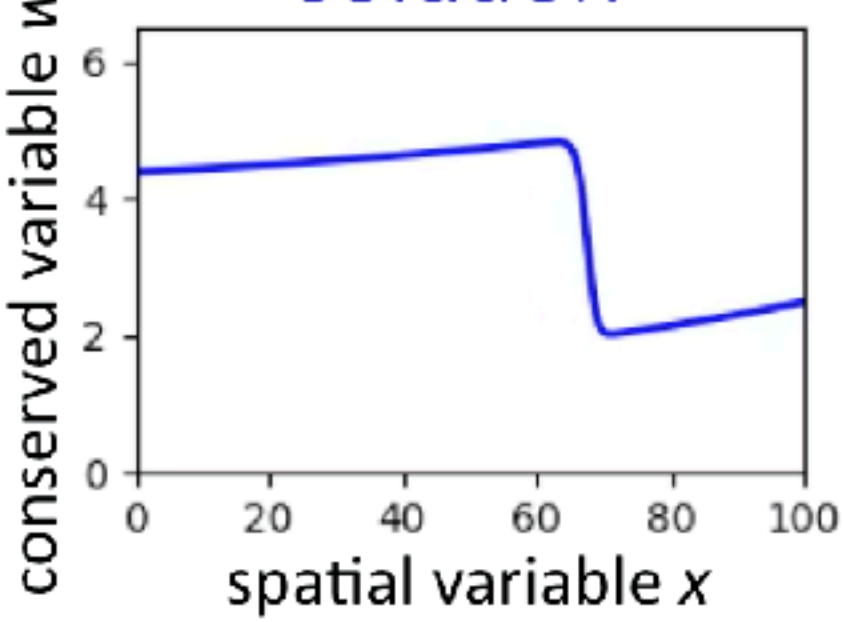
***POD, p=3*** *projection*



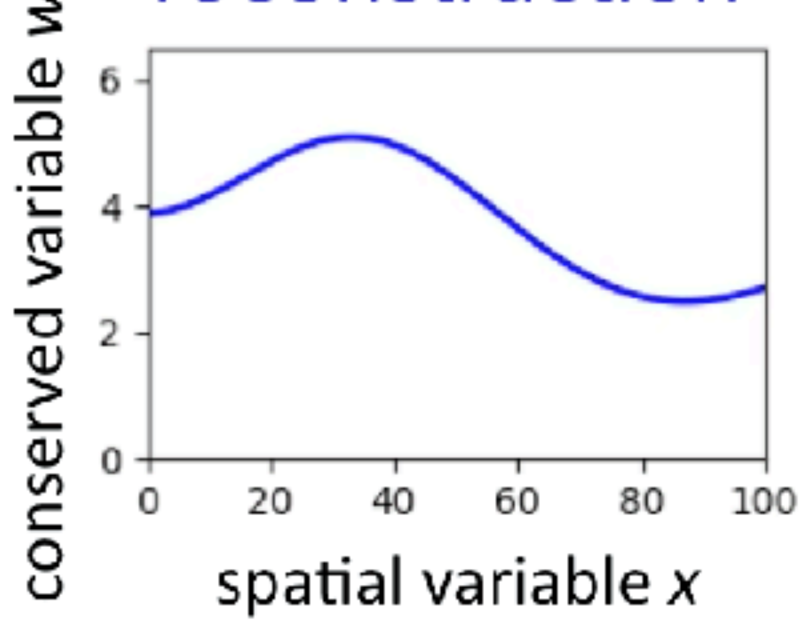

***Autoencoder, p=3*** *projection*



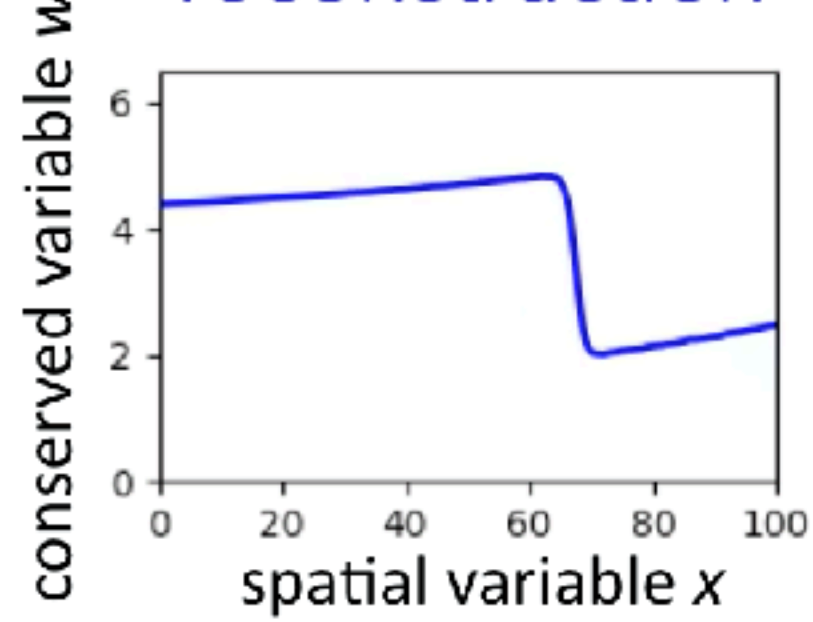

*solution*




*reconstruction*




*reconstruction*




+ Projection error onto 3-dimensional manifold nearly perfect

# Manifold LSPG outperforms optimal linear subspace

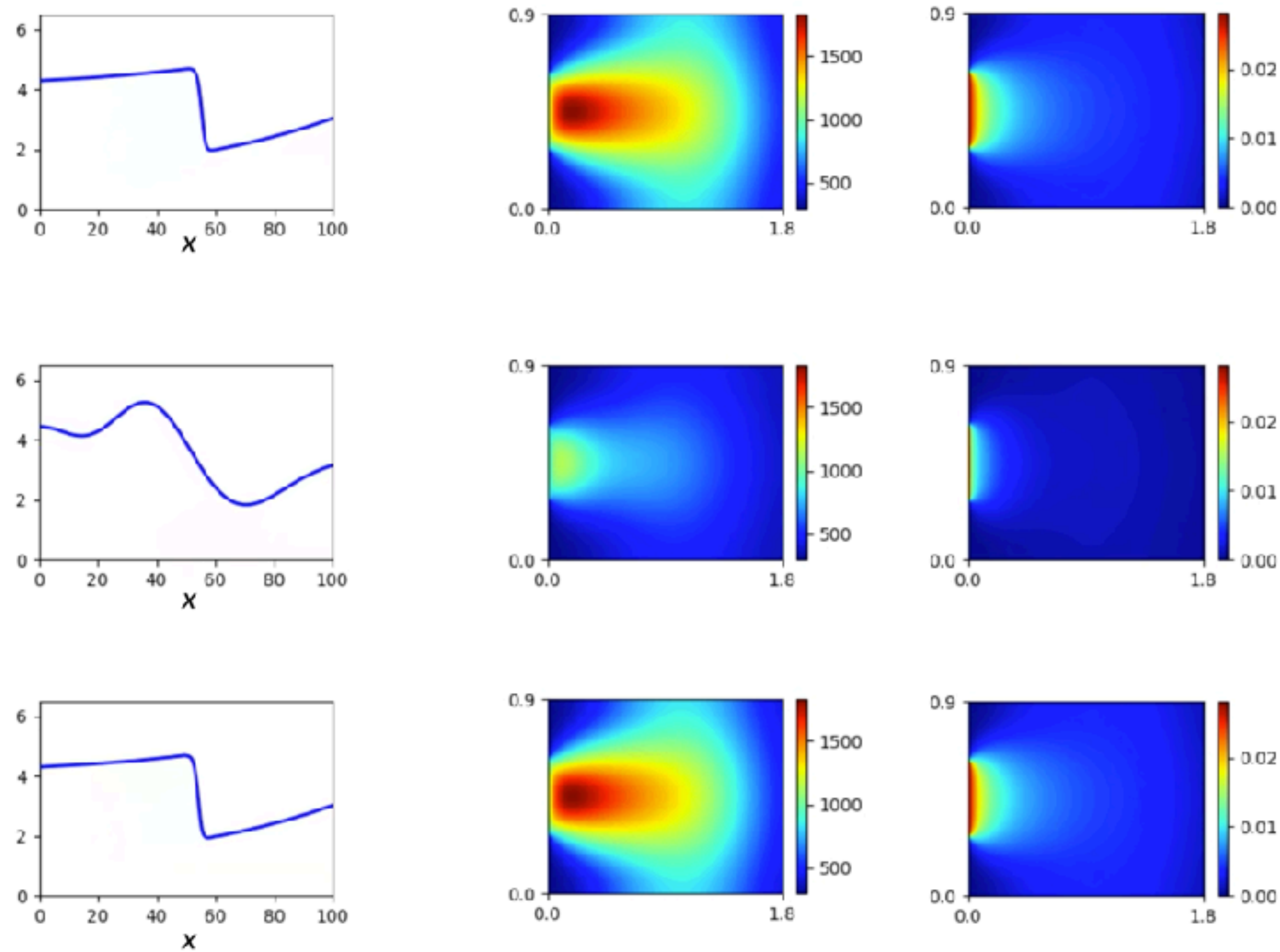
*1D Burgers' equation* — *2D reacting flow*

conserved variable — temperature — $H_2$ fraction

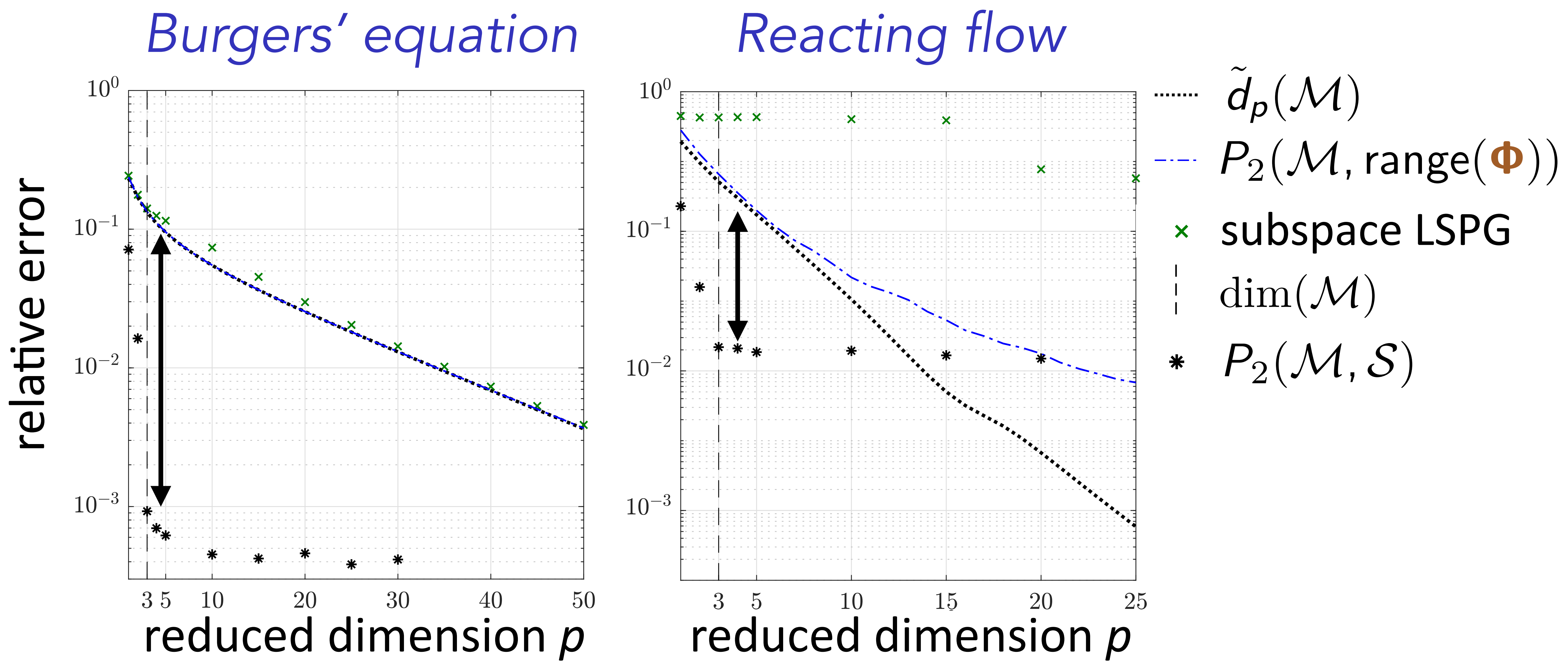
*high-fidelity model*

*POD-LSPG p=5*

*Manifold LSPG p=5*

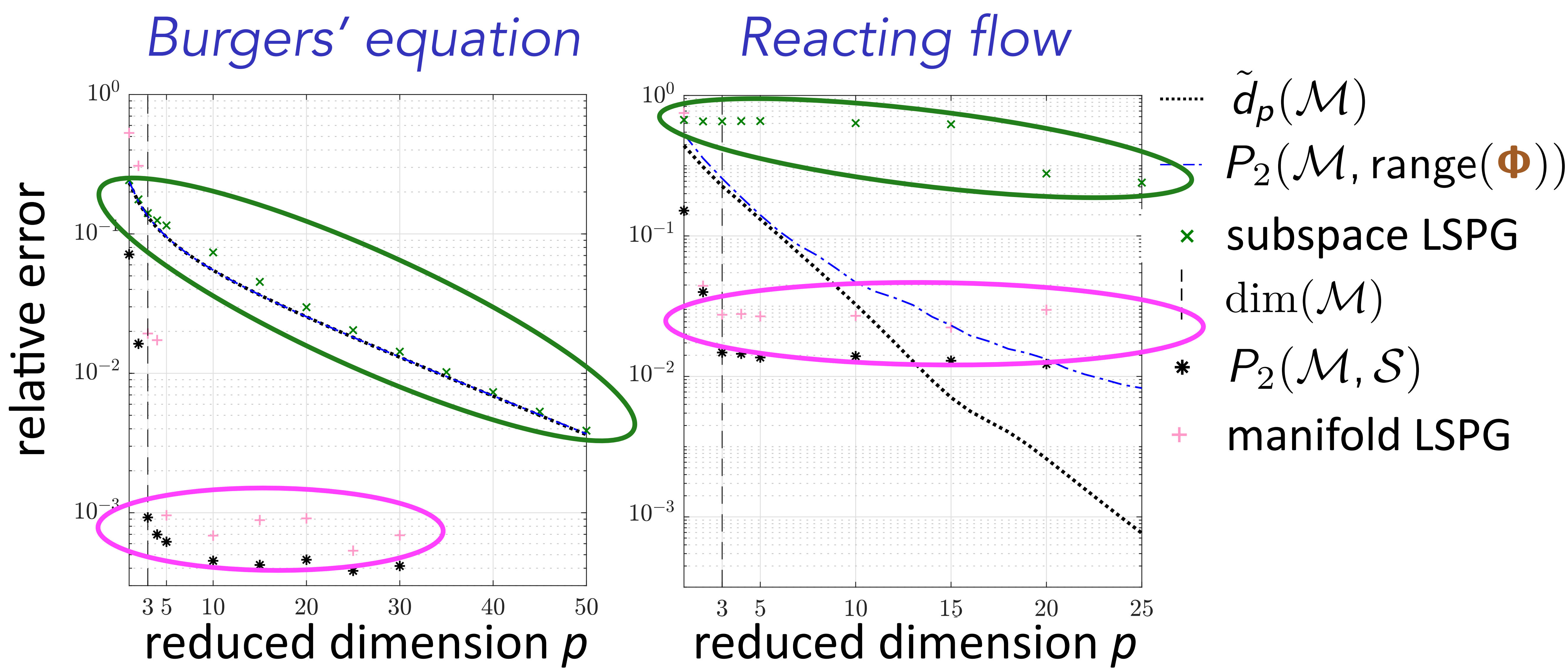


# Method improves generalization performance



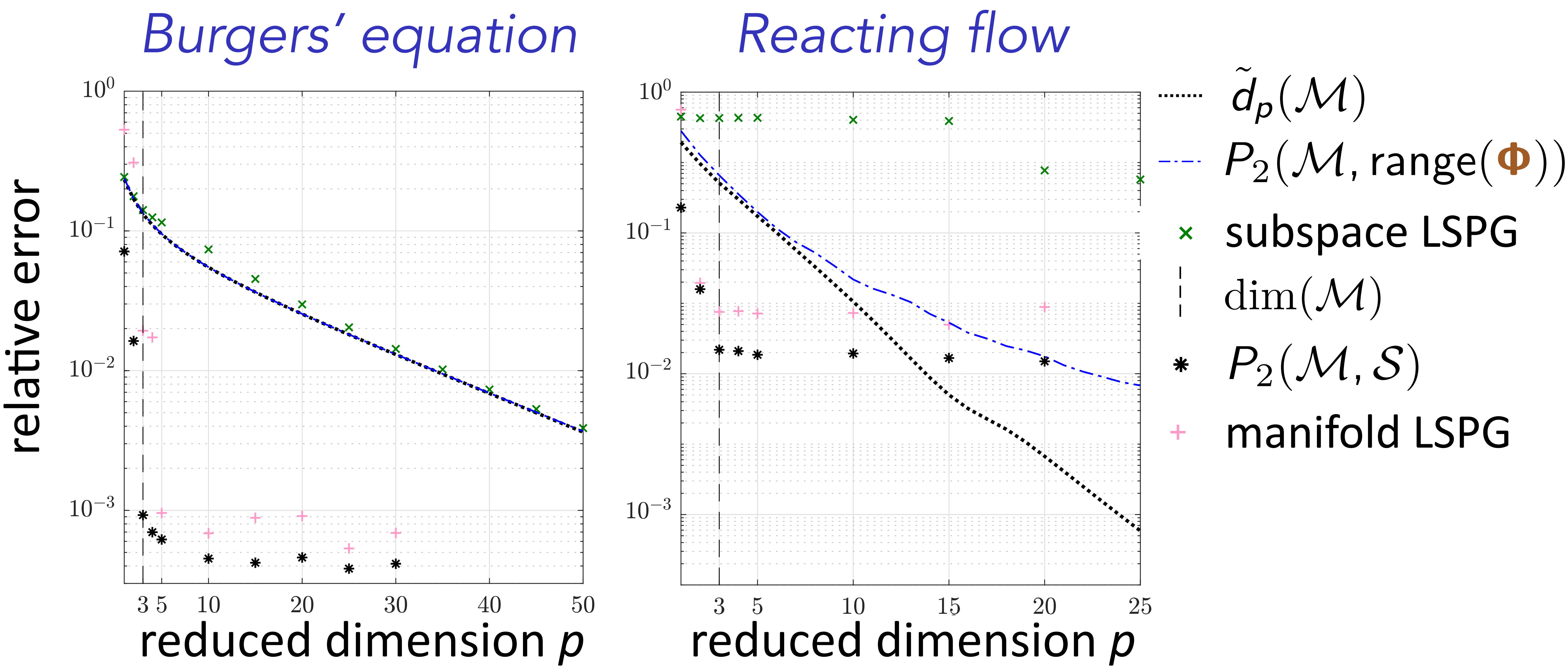

+ Autoencoder manifold significantly better than optimal linear subspace

# Method improves generalization performance



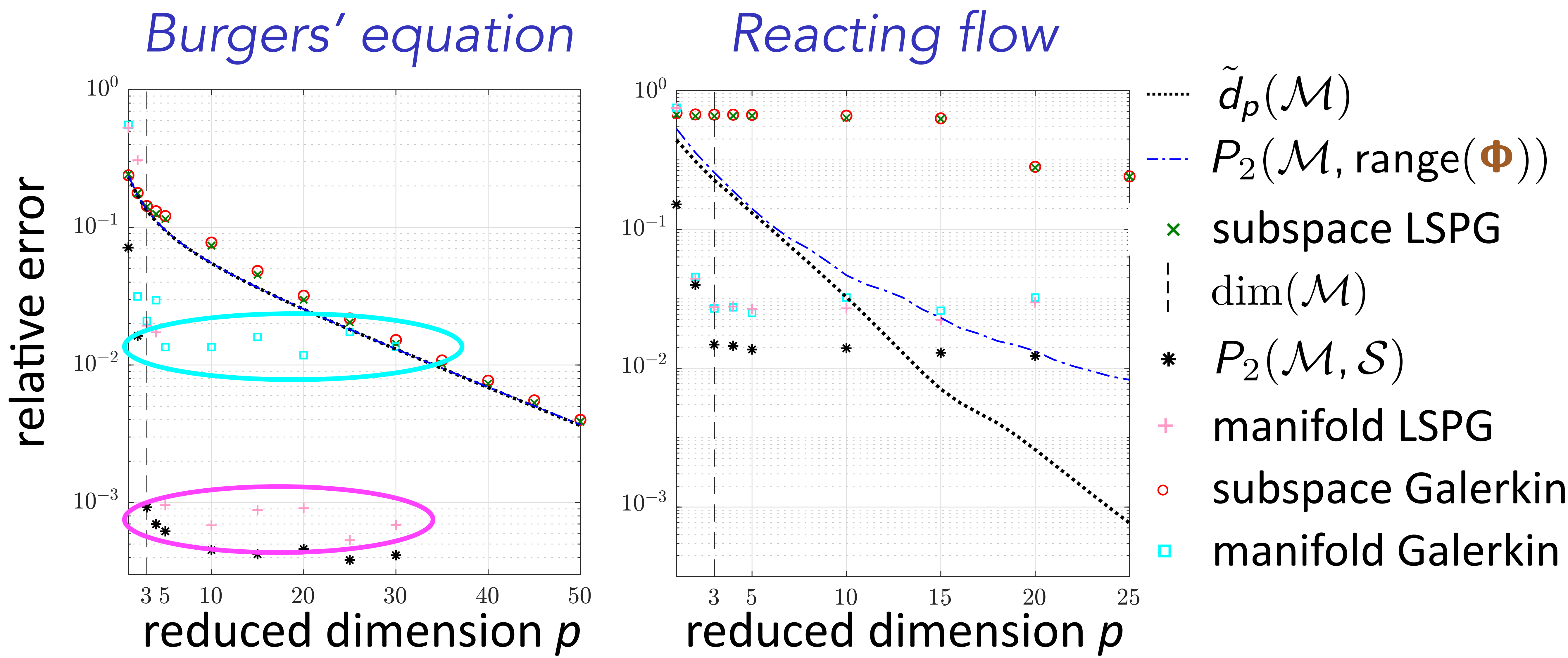

+ Autoencoder manifold significantly better than optimal linear subspace
+ Manifold LSPG orders-of-magnitude more accurate than subspace LSPG

# Method improves generalization performance




+ Autoencoder manifold significantly better than optimal linear subspace
+ Manifold LSPG orders-of-magnitude more accurate than subspace LSPG
+ Method breaks Kolmogorov-width barrier

# Method improves generalization performance




+ Autoencoder manifold significantly better than optimal linear subspace
+ Manifold LSPG orders-of-magnitude more accurate than subspace LSPG
+ Method breaks Kolmogorov-width barrier
+ Manifold LSPG outperforms manifold Galerkin on 1D Burgers' equation

# Outstanding challenges in model reduction

## 1) Linear-subspace assumption is strong

$$\mathbf{x}(t) \approx \tilde{\mathbf{x}}(t) = \boldsymbol{\Phi}\,\hat{\mathbf{x}}(t)$$

- Lee and C. "Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders." J Comp Phys, 404:108973, 2020.

## 2) Important physical properties not satisfied

*Galerkin*

$$\boldsymbol{\Phi}\frac{d\hat{\mathbf{x}}}{dt}(\mathbf{x}, t) = \underset{\mathbf{v}\in \mathrm{range}(\boldsymbol{\Phi})}{\mathrm{argmin}}\ \|\mathbf{r}(\mathbf{v}, \mathbf{x}; t)\|_2$$

*LSPG*

$$\boldsymbol{\Phi}\hat{\mathbf{x}}^n = \underset{\mathbf{v}\in \mathrm{range}(\boldsymbol{\Phi})}{\arg\min}\ \|\mathbf{r}^n(\mathbf{v})\|_2$$

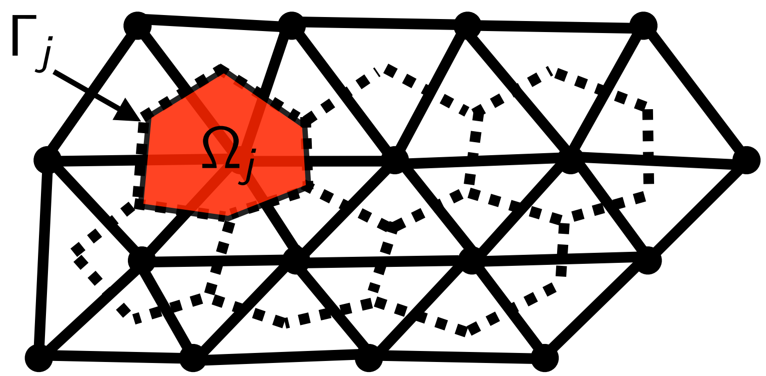
- C., Choi, and Sargsyan. "Conservative model reduction for finite-volume models." J Comp Phys, 371:280–314, 2018.
- Lee and C. "Deep conservation: A latent dynamics model for exact satisfaction of physical conservation laws ." arXiv e-print 1909.09754, 2019.

## 3) Error analysis difficult

- Freno and C. "Machine-learning error models for approximate solutions to parameterized systems of nonlinear equations." CMAME, 348:250–296, 2019.
- Parish and C. "Time-series machine-learning error models for approximate solutions to parameterized dynamical systems." arXiv e-print, (1907.11822).

# Finite-volume method

$$\boxed{\text{ODE:} \quad \frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}; t)}$$




$$x_{\mathcal{I}(i,j)}(t) = \frac{1}{|\Omega_j|} \int_{\Omega_j} u_i(\vec{x}, t)\, \mathrm{d}\vec{x}$$

- average value of conserved variable $i$ over control volume $j$

$$f_{\mathcal{I}(i,j)}(\mathbf{x}, t) = -\frac{1}{|\Omega_j|} \int_{\Gamma_j} \underbrace{\mathbf{g}_i(\mathbf{x}; \vec{x}, t)}_{\text{flux}} \cdot \mathbf{n}_j(\vec{x})\, \mathrm{d}\vec{s}(\vec{x}) + \frac{1}{|\Omega_j|} \int_{\Omega_j} \underbrace{s_i(\mathbf{x}; \vec{x}, t)}_{\text{source}}\, \mathrm{d}\vec{x}$$

- flux and source of conserved variable $i$ within control volume $j$

$$r_{\mathcal{I}(i,j)} = \frac{dx_{\mathcal{I}(i,j)}}{dt}(t) - f_{\mathcal{I}(i,j)}(\mathbf{x}, t)$$

- **rate of conservation violation** of variable $i$ in control volume $j$

$$\boxed{\text{O}\Delta\text{E:} \quad \mathbf{r}^n(\mathbf{x}^n) = 0, \;\; n = 1, \ldots, N}$$

$$r^n_{\mathcal{I}(i,j)} = x_{\mathcal{I}(i,j)}(t^{n+1}) - x_{\mathcal{I}(i,j)}(t^n) + \int_{t^n}^{t^{n+1}} f_{\mathcal{I}(i,j)}(\mathbf{x}, t)dt$$

- **conservation violation** of variable $i$ in control volume $j$ over time step $n$

***Conservation is the intrinsic structure enforced by finite-volume methods***

# Conservative manifold model reduction

## *Manifold Galerkin*

$$\underset{\hat{\mathbf{v}}\in\mathbb{R}^p}{\text{minimize}}\ \|\mathbf{r}(\nabla\mathbf{g}(\hat{\mathbf{x}})\hat{\mathbf{v}};\mathbf{g}(\hat{\mathbf{x}});t)\|_2$$

- Minimize conservation-violation rates

## *Manifold LSPG*

$$\hat{\mathbf{x}}^n = \underset{\hat{\mathbf{v}}\in\mathbb{R}^p}{\text{argmin}}\|\mathbf{r}^n(\mathbf{g}(\hat{\mathbf{v}}))\|_2$$

- Minimize conservation violations over time step *n*

- Neither enforces conservation!

## *Conservative manifold Galerkin*

$$\underset{\hat{\mathbf{v}}\in\mathbb{R}^p}{\text{minimize}}\ \|\mathbf{r}(\nabla\mathbf{g}(\hat{\mathbf{x}})\hat{\mathbf{v}};\mathbf{g}(\hat{\mathbf{x}});t)\|_2$$

$$\text{subject to }\ \mathbf{C}\mathbf{r}(\nabla\mathbf{g}(\hat{\mathbf{x}})\hat{\mathbf{v}};\mathbf{g}(\hat{\mathbf{x}});t) = \mathbf{0}$$

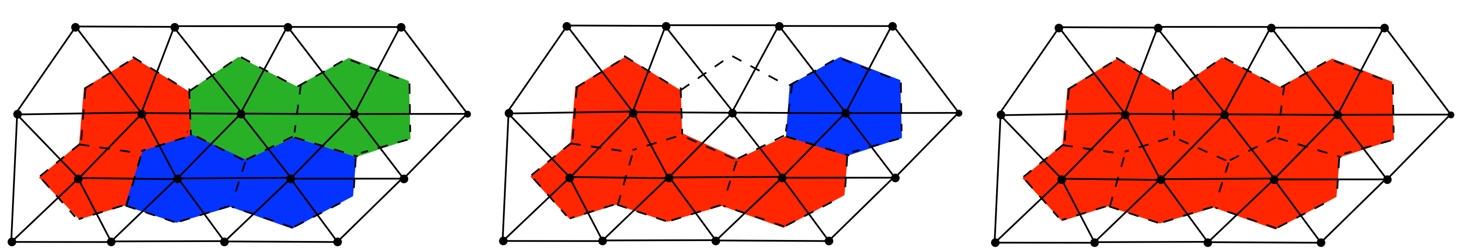
- Minimize conservation-violation rates subject to zero conservation-violation rates **over subdomains**

## *Conservative manifold LSPG*

$$\underset{\hat{\mathbf{v}}\in\mathbb{R}^p}{\text{minimize}}\ \|\mathbf{r}^n(\mathbf{g}(\hat{\mathbf{v}}))\|_2$$

$$\text{subject to }\ \mathbf{C}\mathbf{r}^n(\mathbf{g}(\hat{\mathbf{v}})) = \mathbf{0}$$

- Minimize conservation violations over time step *n* subject to zero conservation violations over time step ***n* over subdomains**



+ Conservation enforced over prescribed subdomains

# Discrete-time error bound (linear subspaces)

**Lemma:** local conserved-quantity error bounds [C., Choi, Sargsyan, 2018]

The error in the conserved quantities computed with either conservative Galerkin or conservative LSPG can be bounded as:

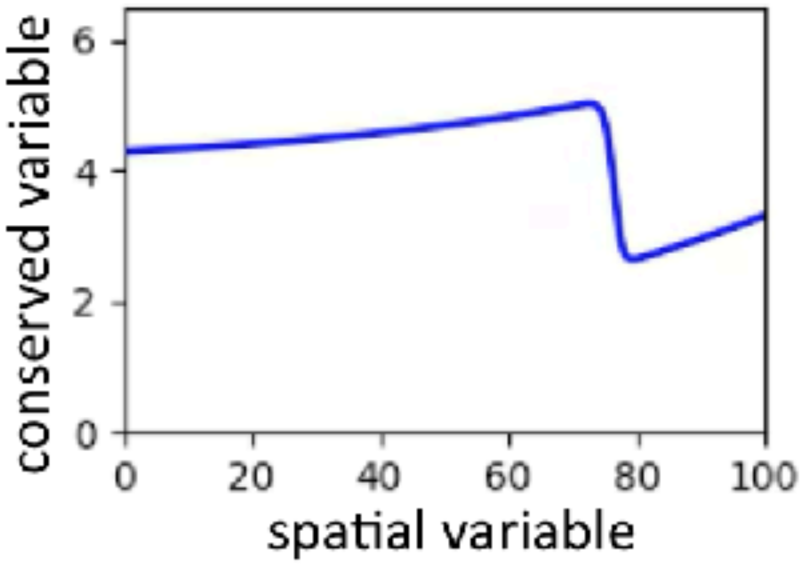
$$\|\bar{\mathbf{C}}(\mathbf{x}^n - \boldsymbol{\Phi}\hat{\mathbf{x}}^n)\|_2 \leq \sum_{\ell=0}^{k} \frac{|\beta_\ell^n|\Delta t}{|\alpha_0^n|}\|\bar{\mathbf{C}}\mathbf{f}(\mathbf{x}^{n-\ell}) - \bar{\mathbf{C}}\mathbf{f}(\boldsymbol{\Phi}\hat{\mathbf{x}}^{n-\ell})\|_2$$

$$+ \sum_{\ell=1}^{k} \frac{|\alpha_\ell^n|}{|\alpha_0^n|}\|\bar{\mathbf{C}}(\mathbf{x}^{n-\ell} - \boldsymbol{\Phi}\hat{\mathbf{x}}^{n-\ell})\|_2$$

- Error depends only on velocity error on *decomposed mesh*

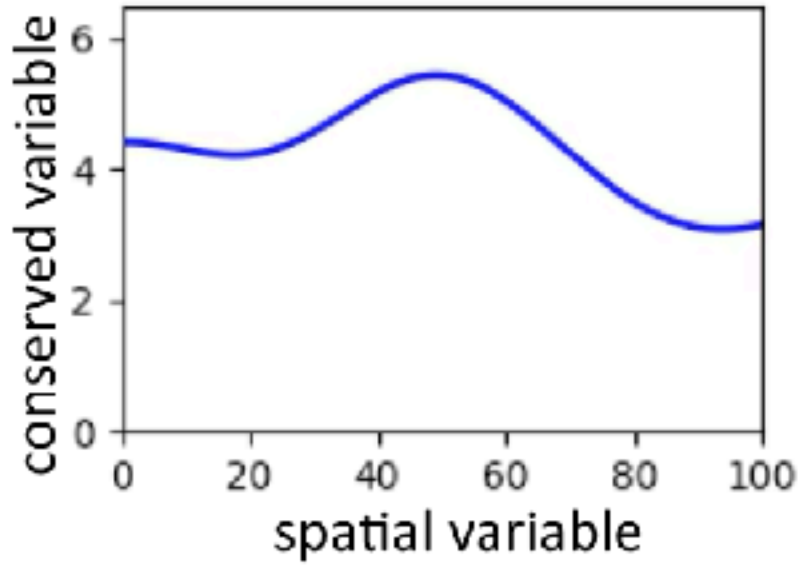
+ No source, global conservation: error due to flux error along boundary!

# High-fidelity model



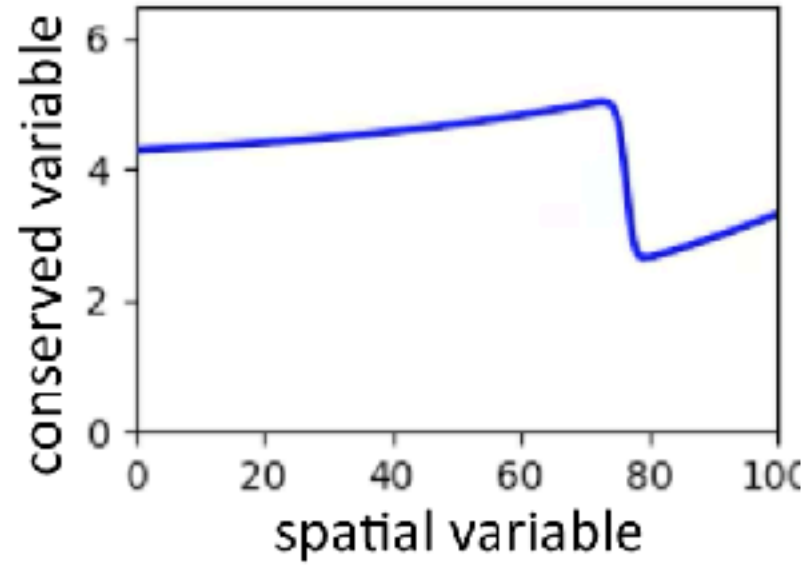

# Reduced-order models

## POD subspace



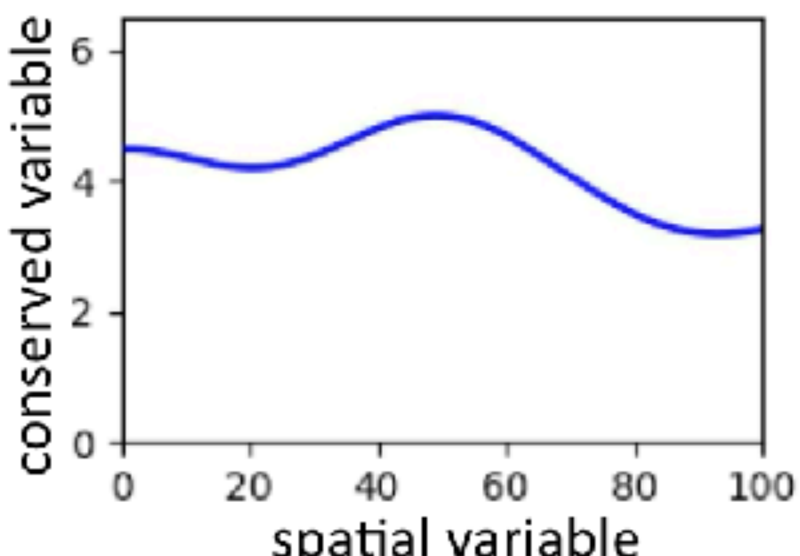

**Solution error: 13%**
**Conservation violation: 16%**

## Autoencoder manifold



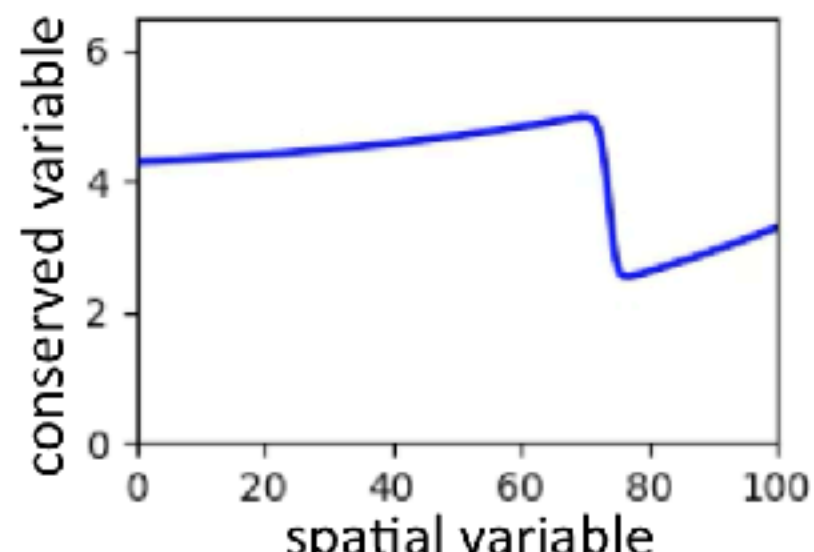

**Solution error: 0.5%**
**Conservation violation: 1%**

## POD subspace with conservation constraints




**Solution error: 12%**
**Conservation violation: <0.001%**

## Autoencoder manifold with conservation constraints




**Solution error: 0.2%**
**Conservation violation: <0.001%**

# Outlook

*Conservative manifold Galerkin*

$$\underset{\hat{\mathbf{v}}\in\mathbb{R}^p}{\text{minimize}}\ \|\mathbf{r}(\nabla\mathbf{g}(\hat{\mathbf{x}})\hat{\mathbf{v}};\mathbf{g}(\hat{\mathbf{x}});t)\|_2$$

$$\text{subject to}\ \mathbf{C}\mathbf{r}(\nabla\mathbf{g}(\hat{\mathbf{x}})\hat{\mathbf{v}};\mathbf{g}(\hat{\mathbf{x}});t) = \mathbf{0}$$

*Conservative manifold LSPG*

$$\underset{\hat{\mathbf{v}}\in\mathbb{R}^p}{\text{minimize}}\ \|\mathbf{r}^n(\mathbf{g}(\hat{\mathbf{v}}))\|_2$$

$$\text{subject to}\ \mathbf{C}\mathbf{r}^n(\mathbf{g}(\hat{\mathbf{v}})) = \mathbf{0}$$

**Interpretation**

- Integrates computational physics with deep learning
- *Projection-based latent dynamics model* that enforces conservation
- Nearly all existing methods are *data-driven latent dynamics models*

  [Böhmer et al., 2015; Goroshin et al., 2015; Watter et al., 2015; Karl et al., 2017; Takeishi et al., 2017; Banijamali et al., 2018; Lesort et al., 2018; Lusch et al., 2018; Morton et al., 2018 Otto and Rowley, 2019]

**Gradient computation**

- Backpropagation used to compute decoder Jacobian $\nabla\mathbf{g}(\hat{\mathbf{x}})$
- Quasi-Newton solvers directly call TensorFlow

**Ongoing work**

- *Hyper-reduction*: "easy" because convolutional layers preserve sparsity
- Integration in large-scale code underway in Pressio

# Shortcomings of state-of-the-art ROMs

## 1) Linear-subspace assumption is strong

$$\mathbf{x}(t) \approx \tilde{\mathbf{x}}(t) = \boldsymbol{\Phi}\,\hat{\mathbf{x}}(t)$$

- Lee and C. "Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders." J Comp Phys, 404:108973, 2020.

## 2) Important physical properties not guaranteed

*Galerkin*

$$\boldsymbol{\Phi}\frac{d\hat{\mathbf{x}}}{dt}(\mathbf{x}, t) = \underset{\mathbf{v}\in\mathrm{range}(\boldsymbol{\Phi})}{\mathrm{argmin}} \|\mathbf{r}(\mathbf{v}, \mathbf{x}; t)\|_2$$

*LSPG*

$$\boldsymbol{\Phi}\hat{\mathbf{x}}^n = \underset{\mathbf{v}\in\mathrm{range}(\boldsymbol{\Phi})}{\arg\min} \|\mathbf{r}^n(\mathbf{v})\|_2$$

- C., Choi, and Sargsyan. "Conservative model reduction for finite-volume models." J Comp Phys, 371:280–314, 2018.
- Lee and C. "Deep conservation: A latent dynamics model for exact satisfaction of physical conservation laws ." arXiv e-print 1909.09754, 2019.

## 3) Error analysis difficult

- Freno and C. "Machine-learning error models for approximate solutions to parameterized systems of nonlinear equations." CMAME, 348:250–296, 2019.
- Parish and C. "Time-series machine-learning error models for approximate solutions to parameterized dynamical systems." arXiv e-print, (1907.11822).

# Discrete-time error bound

**Theorem:** error bound for BDF integrators [C., Barone, Antil, 2017]

If the following conditions hold:

1. $\mathbf{f}(\cdot;t)$ is Lipschitz continuous with Lipschitz constant $\kappa$
2. The time step $\Delta t$ is small enough such that $0 < h := |\alpha_0| - |\beta_0|\kappa\Delta t$,

$$\|\mathbf{x}^n - \mathbf{g}(\hat{\mathbf{x}}_{\mathrm{G}}^n)\|_2 \leq \frac{1}{h}\|\mathbf{r}_{\mathrm{G}}^n(\mathbf{g}(\hat{\mathbf{x}}_{\mathrm{G}}))\|_2 + \frac{1}{h}\sum_{\ell=1}^{k}|\gamma_\ell|\|\mathbf{x}^{n-\ell} - \mathbf{g}(\hat{\mathbf{x}}_{\mathrm{G}})\|_2$$

$$\|\mathbf{x}^n - \mathbf{g}(\hat{\mathbf{x}}_{\mathrm{LSPG}}^n)\|_2 \leq \frac{1}{h}\min_{\hat{\mathbf{v}}}\|\mathbf{r}_{\mathrm{LSPG}}^n(\mathbf{g}(\hat{\mathbf{v}}))\|_2 + \frac{1}{h}\sum_{\ell=1}^{k}|\gamma_\ell|\|\mathbf{x}^{n-\ell} - \mathbf{g}(\hat{\mathbf{x}}_{\mathrm{LSPG}})\|_2$$

***Can we use these error bounds for error estimation?***

# Discrete-time error bound

**Theorem:** error bound for BDF integrators [C., Barone, Antil, 2017]

If the following conditions hold:

1. $\mathbf{f}(\cdot; t)$ is Lipschitz continuous with Lipschitz constant $\kappa$
2. The time step $\Delta t$ is small enough such that $0 < h := |\alpha_0| - |\beta_0|\kappa\Delta t$,

$$\|\mathbf{x}^n - \mathbf{g}(\hat{\mathbf{x}}_{\mathrm{G}}^n)\|_2 \leq \frac{\gamma_1(\gamma_2)^n \exp(\gamma_3 t^n)}{\gamma_4 + \gamma_5\Delta t} \max_{j\in\{1,\ldots,N\}} \|\mathbf{r}_{\mathrm{LSPG}}^j(\mathbf{g}(\hat{\mathbf{x}}_{\mathrm{G}}^j))\|_2$$

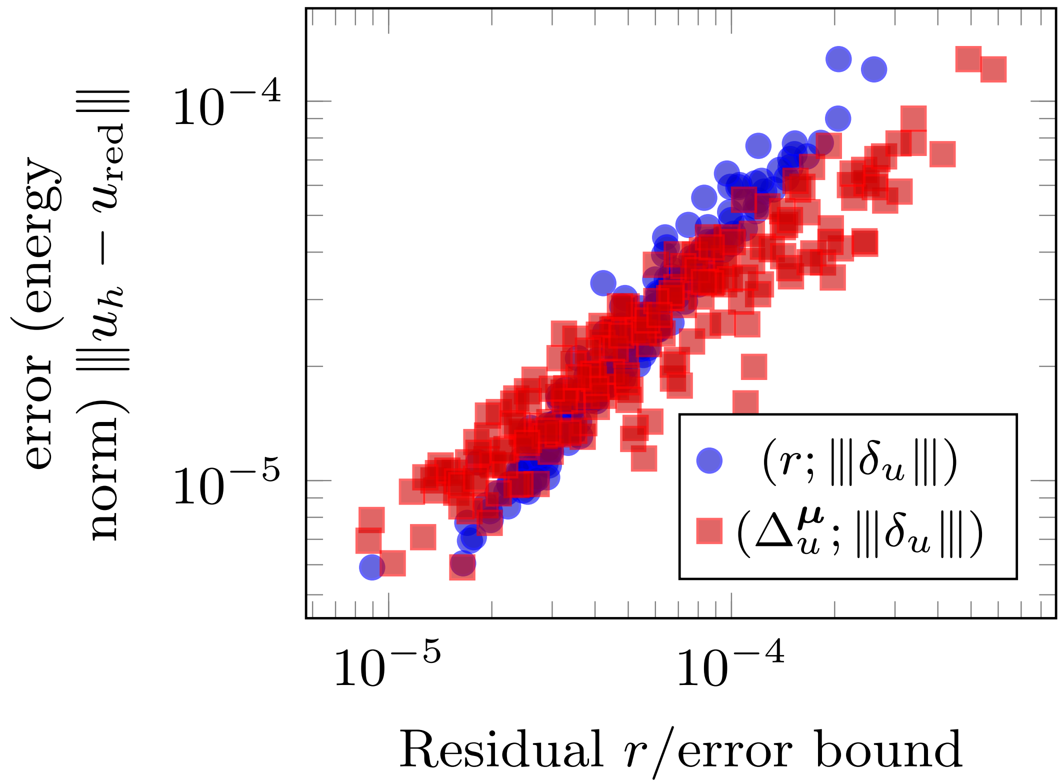
$$\|\mathbf{x}^n - \mathbf{g}(\hat{\mathbf{x}}_{\mathrm{LSPG}}^n)\|_2 \leq \frac{\gamma_1(\gamma_2)^n \exp(\gamma_3 t^n)}{\gamma_4 + \gamma_5\Delta t} \max_{j\in\{1,\ldots,N\}} \min_{\hat{\mathbf{v}}} \|\mathbf{r}_{\mathrm{LSPG}}^j(\mathbf{g}(\hat{\mathbf{v}}))\|_2$$

***Can we use these error bounds for error estimation?***

- grow exponentially in time
- deterministic: not amenable to uncertainty quantification

# Main idea

- **Observation**: ROMs generate quantities that are informative of the error




- **ML perspective**: these are good features for predicting the error

***Idea:*** *Apply* ***machine learning regression*** *to generate a mapping from residual-based quantities to a random variable for the error*

***Machine-learning error models*** [Freno and C., 2019; Parish and C., 2019]

# Machine-learning error models: formulation

***What attributes does the ROM error have?***

$$\|\mathbf{x}^n - \mathbf{g}(\hat{\mathbf{x}}^n_{\mathrm{LSPG}})\|_2 \leq \frac{\gamma_1(\gamma_2)^n \exp(\gamma_3 t^n)}{\gamma_4 + \gamma_5 \Delta t} \max_{j\in\{1,\ldots,T\}} \min_{\hat{\mathbf{v}}} \|\mathbf{r}^j_{\mathrm{LSPG}}(\mathbf{g}(\hat{\mathbf{v}}))\|_2$$

1. Dependence on non-local quantities in time
2. Dependence on the residual

***Regression model***

$$\hat{\delta}^n(\boldsymbol{\mu}) = \underbrace{\hat{\delta}^n_f(\boldsymbol{\mu})}_{\text{deterministic}} + \underbrace{\hat{\delta}^n_\epsilon(\boldsymbol{\mu})}_{\text{stochastic}}$$

- regression function: $\hat{\delta}^n_f(\boldsymbol{\mu}) = \hat{f}(\boldsymbol{\rho}^n(\boldsymbol{\mu}), \boldsymbol{h}^{n-1}(\boldsymbol{\mu}), \hat{\delta}^{n-1}_f(\boldsymbol{\mu}))$

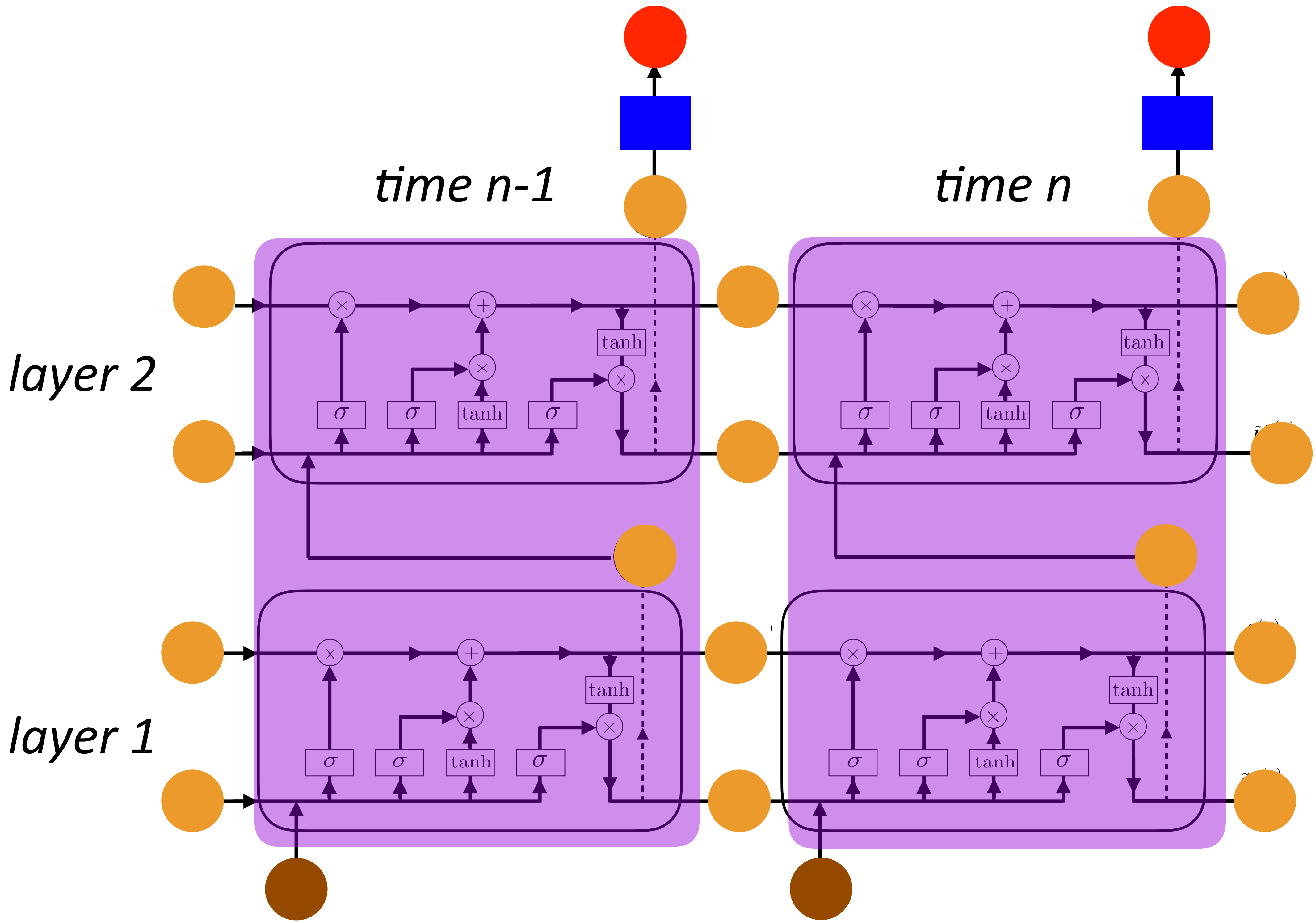
$$\boldsymbol{h}^n(\boldsymbol{\mu}) = \mathrm{g}(\boldsymbol{\rho}^n(\boldsymbol{\mu}), \boldsymbol{h}^{n-1}(\boldsymbol{\mu}), \hat{\delta}^{n-1}_f(\boldsymbol{\mu}))$$

  + latent variables $\boldsymbol{h}^n(\boldsymbol{\mu})$: enable capturing non-local dependencies
  + features $\boldsymbol{\rho}^n(\boldsymbol{\mu})$: residual-based (and cheaply computable)
  + general formulation encompasses ARX, LARX, RNN, LSTM, GRU

# **Example**: long short-term memory (LSTM)

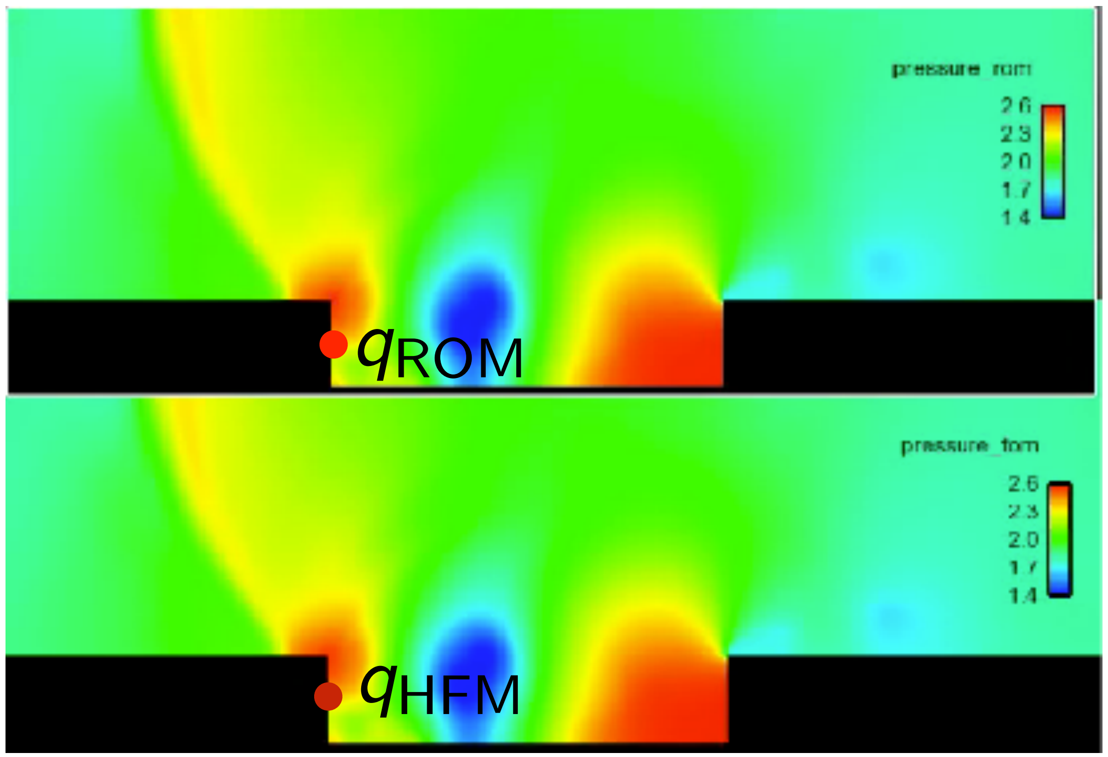
$$\hat{\delta}_f^n(\boldsymbol{\mu}) = \hat{f}(\boldsymbol{\rho}^n(\boldsymbol{\mu}), \boldsymbol{h}^{n-1}(\boldsymbol{\mu}))$$

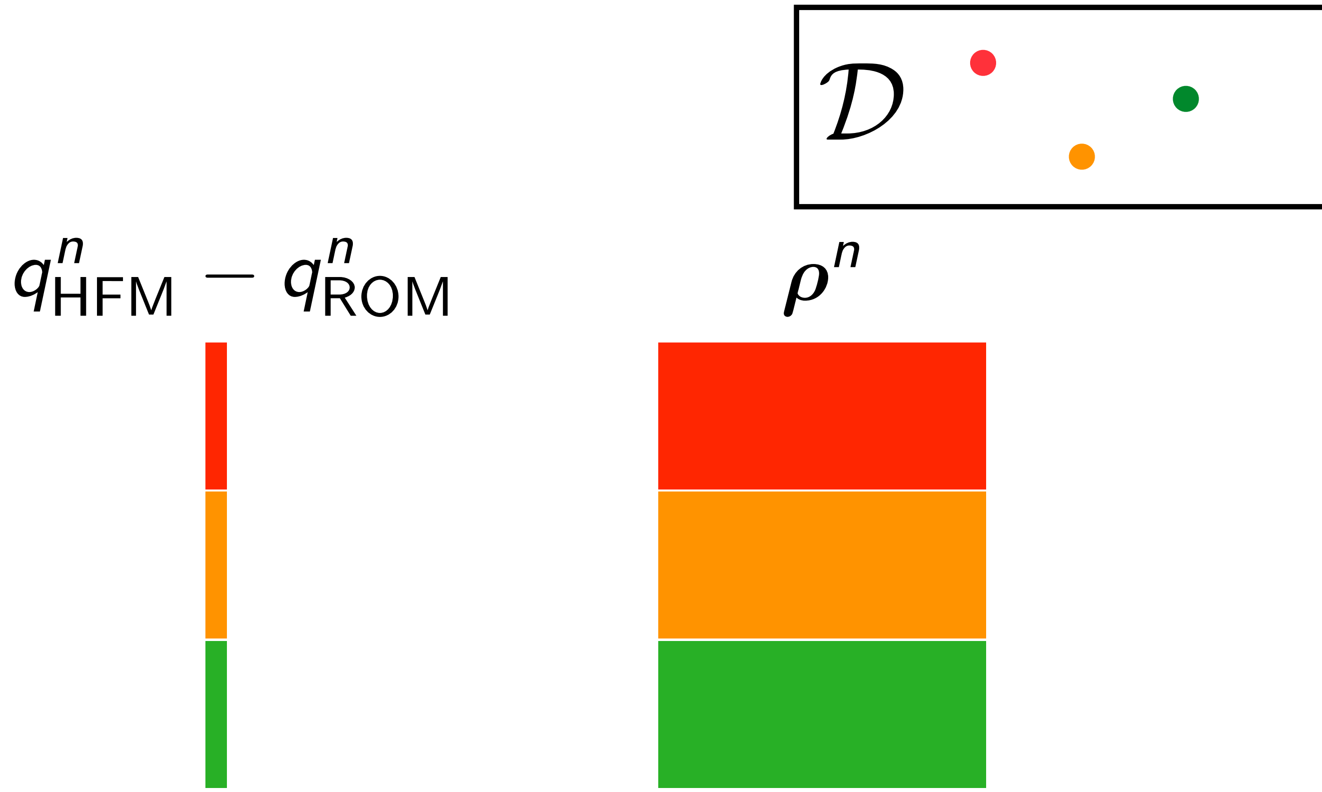
$$\boldsymbol{h}^n(\boldsymbol{\mu}) = \mathbf{g}(\boldsymbol{\rho}^n(\boldsymbol{\mu}), \boldsymbol{h}^{n-1}(\boldsymbol{\mu}))$$



# Training and machine learning: error modeling

1. *Training:* Solve high-fidelity and reduced-order models for $\mu \in \mathcal{D}_{\text{training}}$
2. *Machine learning:* Construct regression model
3. *Reduction:* predict reduced-order-model error for $\mu \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$



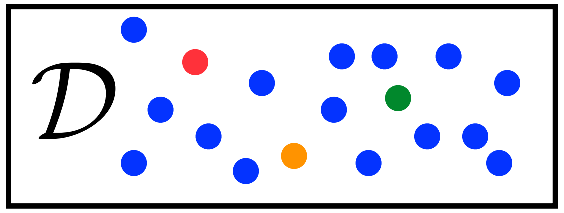




- randomly divide data into (1) training data and (2) testing data
- construct regression function $\hat{\delta}^n_f$ via cross validation on **training data**
- construct noise model $\hat{\delta}^n_\epsilon$ from sample variance on **test data**

# Reduction

1. *Training:* Solve high-fidelity and reduced-order models for $\boldsymbol{\mu} \in \mathcal{D}_{\text{training}}$
2. *Machine learning:* Construct regression model
3. *Reduction:* predict reduced-order-model error for $\boldsymbol{\mu} \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$

*inputs* $\boldsymbol{\mu}$ → *reduced-order model* → *outputs* $q_{\text{ROM}}^n$, $n = 1, \ldots, T$




↓

*features* $\boldsymbol{\rho}^n$, $n = 1, \ldots, T$

↓

*regression model*

$$\hat{\delta}^n(\boldsymbol{\mu}) = \hat{\delta}_f^n(\boldsymbol{\mu}) + \hat{\delta}_\epsilon^n(\boldsymbol{\mu})$$

→ *machine learning error model* $\hat{\delta}^n$ $n = 1, \ldots, T$

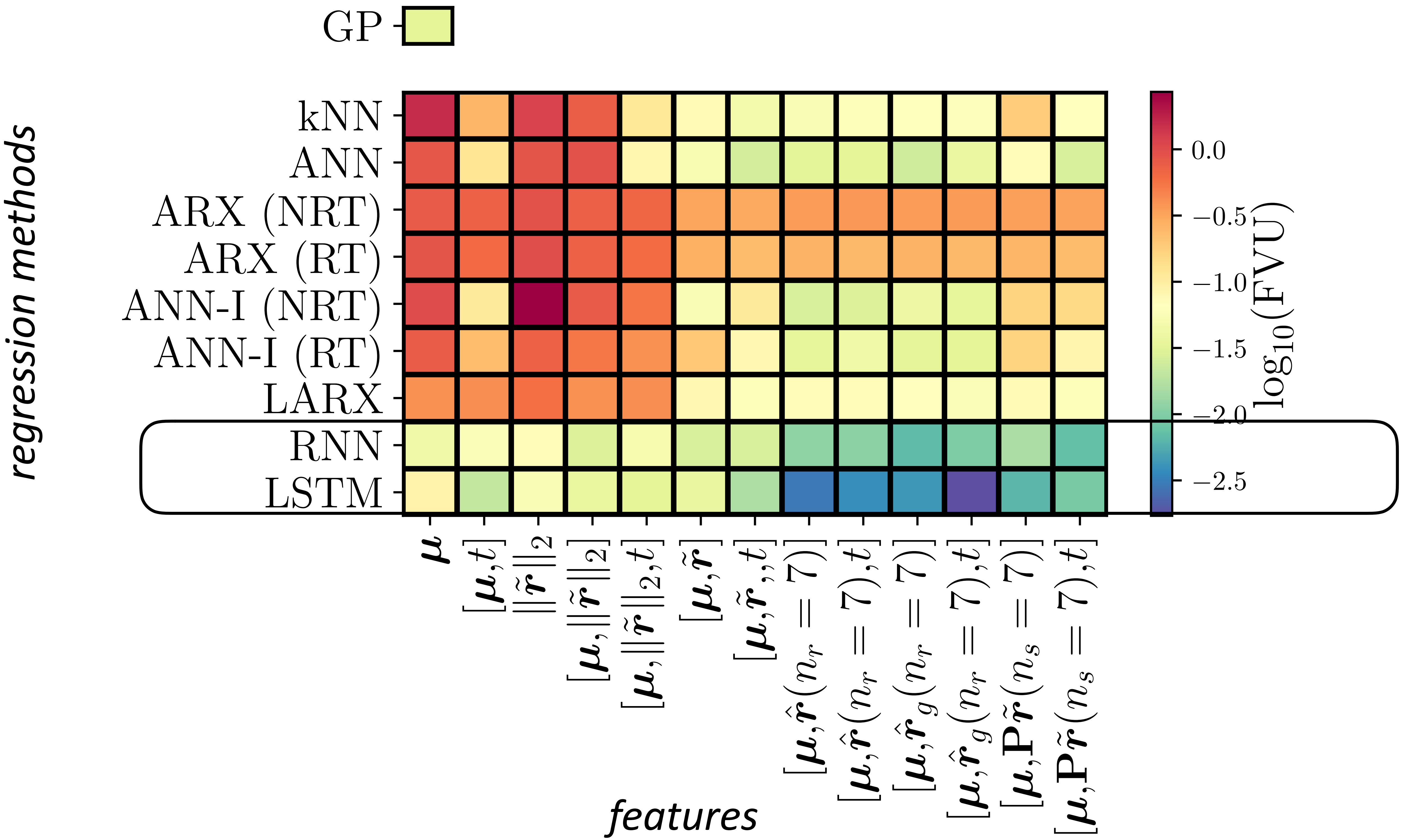
$$\boldsymbol{h}^n(\boldsymbol{\mu}) = \mathbf{g}(\boldsymbol{\rho}^n(\boldsymbol{\mu}), \boldsymbol{h}^{n-1}(\boldsymbol{\mu}), \hat{\delta}_f^{n-1}(\boldsymbol{\mu}))$$

- *Latent dynamics learning*

$$\hat{\delta}_f^n(\boldsymbol{\mu}) = \hat{f}(\boldsymbol{\rho}^n(\boldsymbol{\mu}), \boldsymbol{h}^{n-1}(\boldsymbol{\mu}), \hat{\delta}_f^{n-1}(\boldsymbol{\mu}))$$

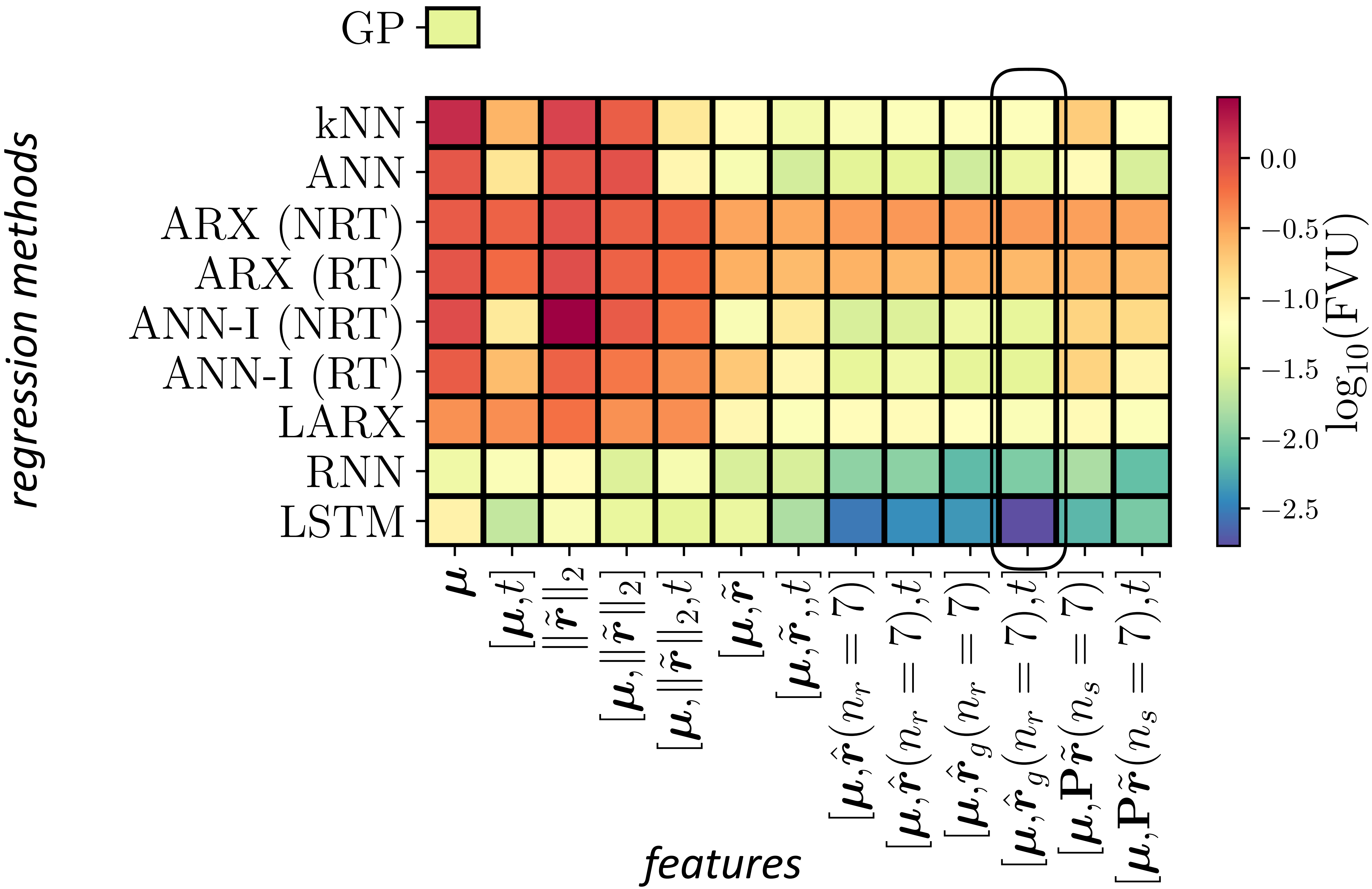
$$\boxed{\underbrace{\tilde{q}_{HFM}^n(\boldsymbol{\mu})}_{\text{stochastic}} = \underbrace{q_{\text{ROM}}^n(\boldsymbol{\mu})}_{\text{deterministic}} + \underbrace{\hat{\delta}^n(\boldsymbol{\mu})}_{\text{stochastic}}}$$

# **Application**: Advection–diffusion equation



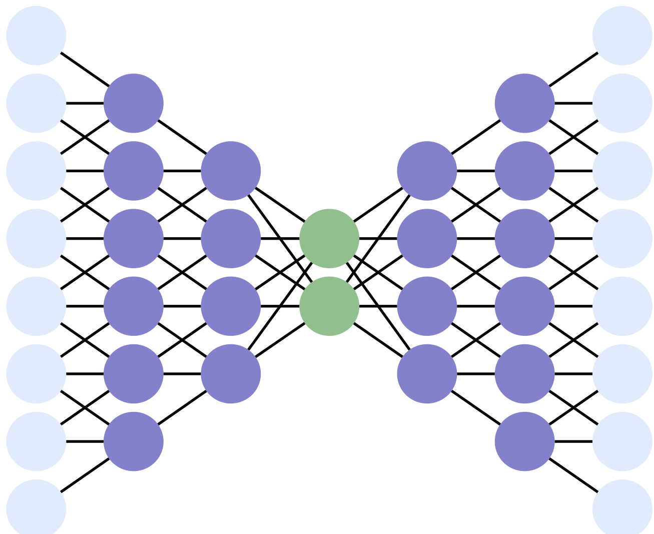

- *regression methods:* classical RNN and LSTM most accurate

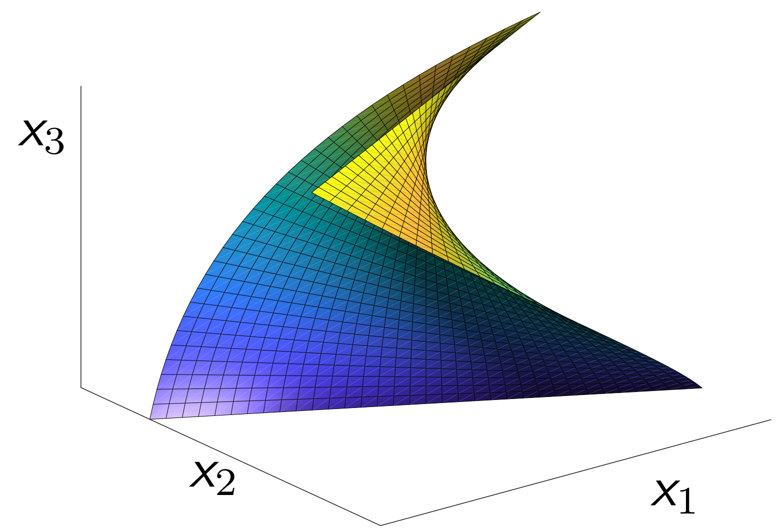
# **Application**: Advection–diffusion equation




+ *regression methods:* classical RNN and LSTM most accurate
+ *features:* only 7 residual samples needed for good accuracy

# Questions?

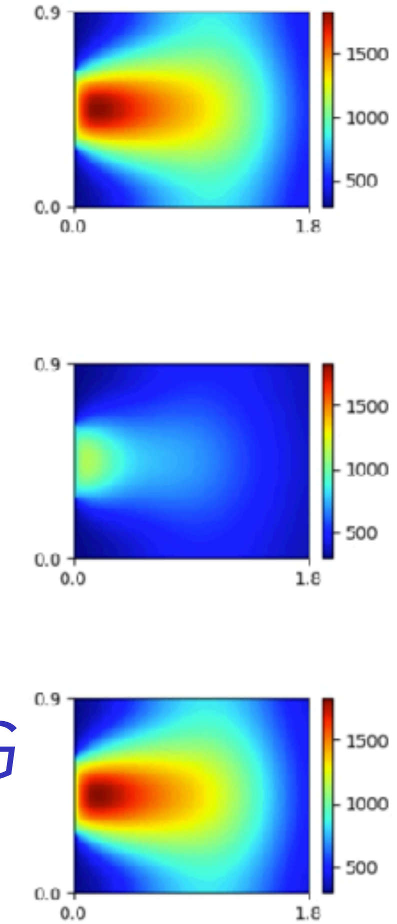


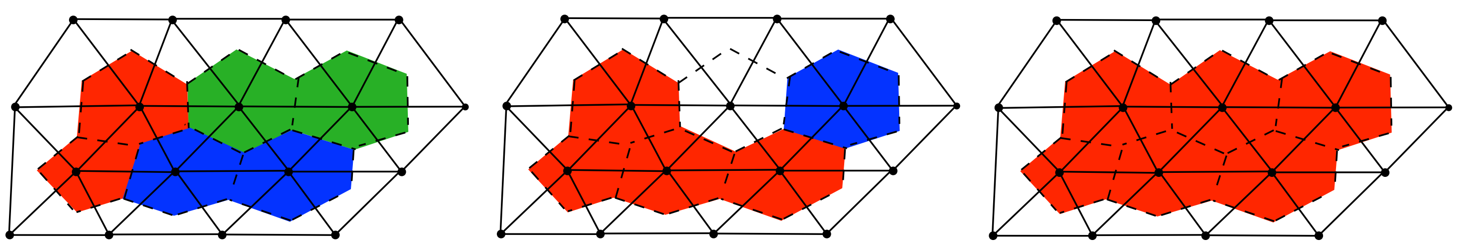


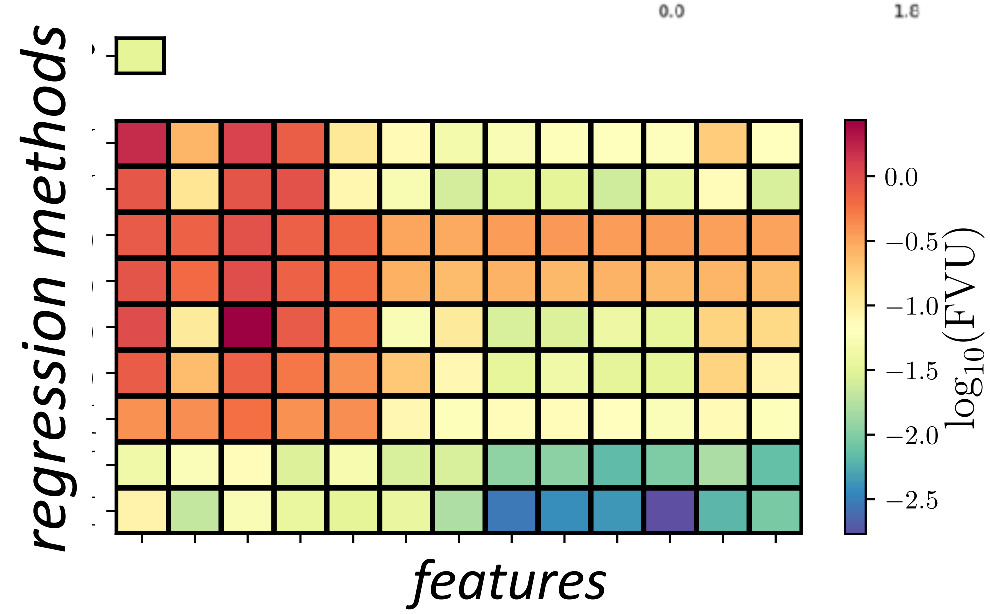

*high-fidelity model*

*POD-LSPG p=5*

*Manifold LSPG p=5*






*Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525. Lawrence Livermore National Laboratory is operated by Lawrence Livermore National Security, LLC, for the U.S. Department of Energy, National Nuclear Security Administration under Contract DE-AC52-07NA27344.*