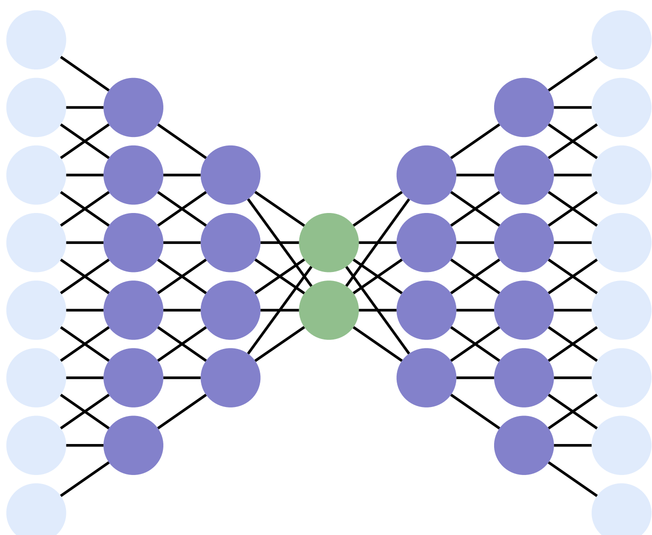
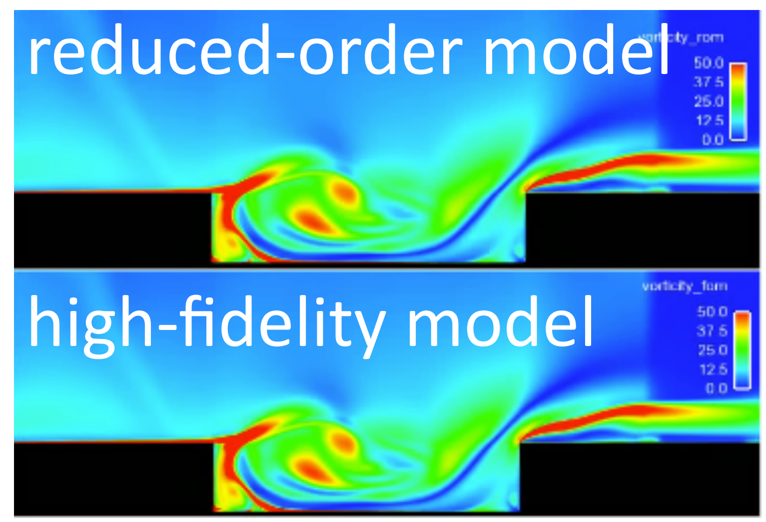
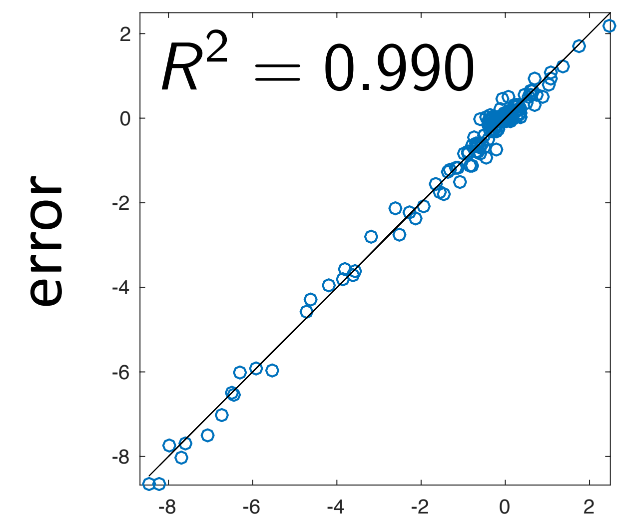
Nonlinear reduced-order modeling

Using machine learning to enable extreme-scale simulations for many-query problems







support vector machine
error prediction

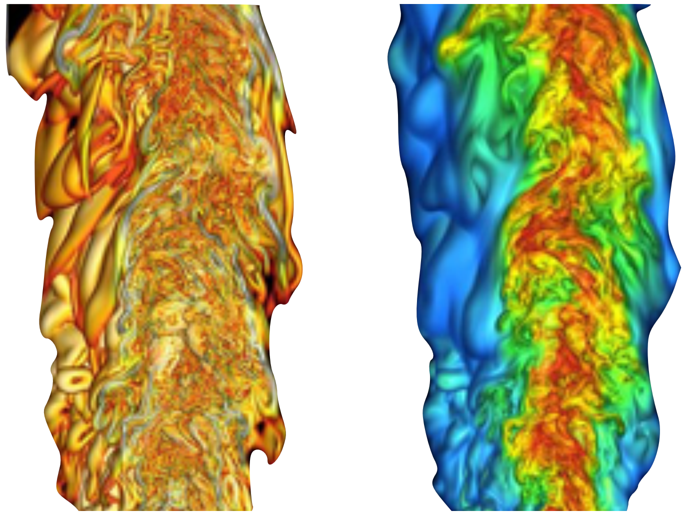
Kevin Carlberg

Sandia National Laboratories

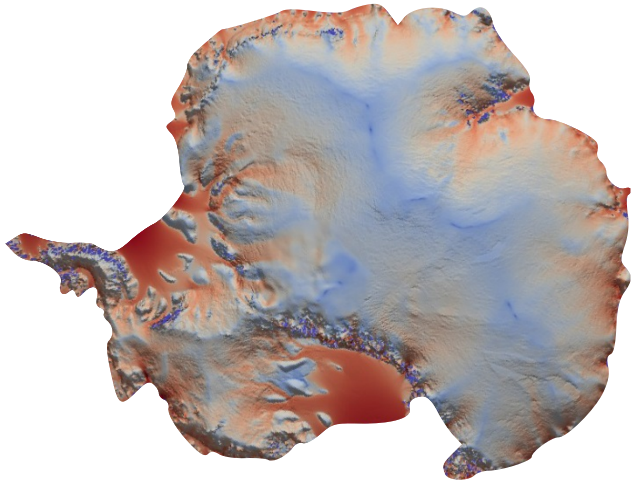
ICERM Workshop on Scientific Machine Learning
Brown University
January 29, 2019

High-fidelity simulation

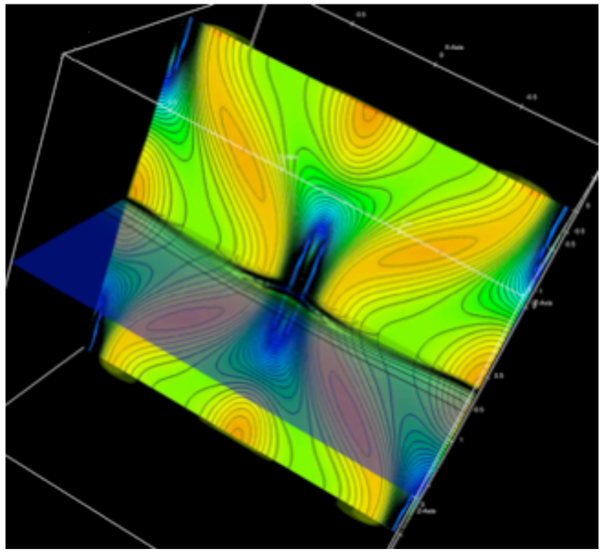
+ Indispensable across science and engineering
- *High fidelity*: extreme-scale nonlinear dynamical system models



Turbulent reacting flows
courtesy J. Chen, Sandia



Antarctic ice sheet modeling
courtesy R. Tuminaro, Sandia



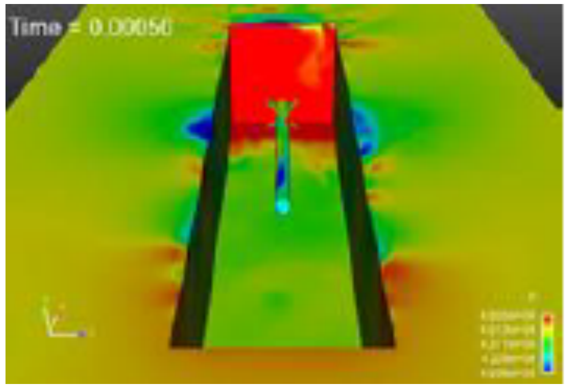
Magnetohydrodynamics
courtesy J. Shadid, Sandia

computational barrier

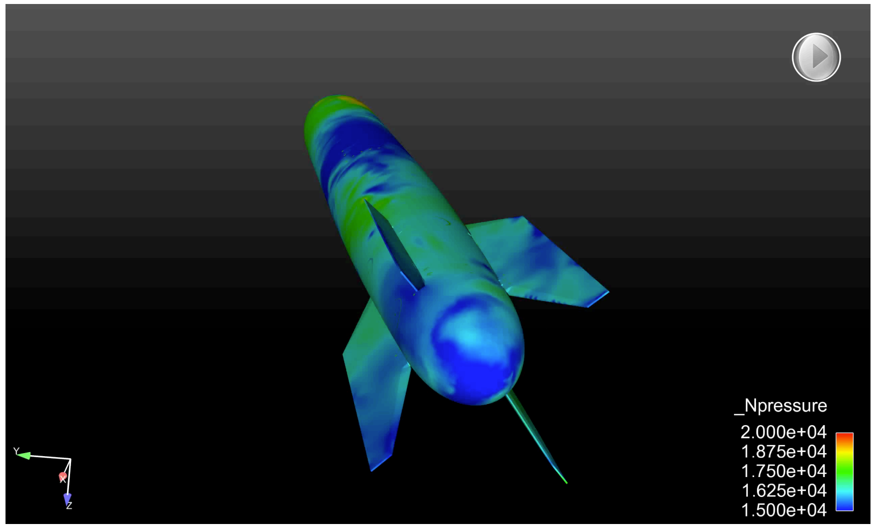
Many-query problems

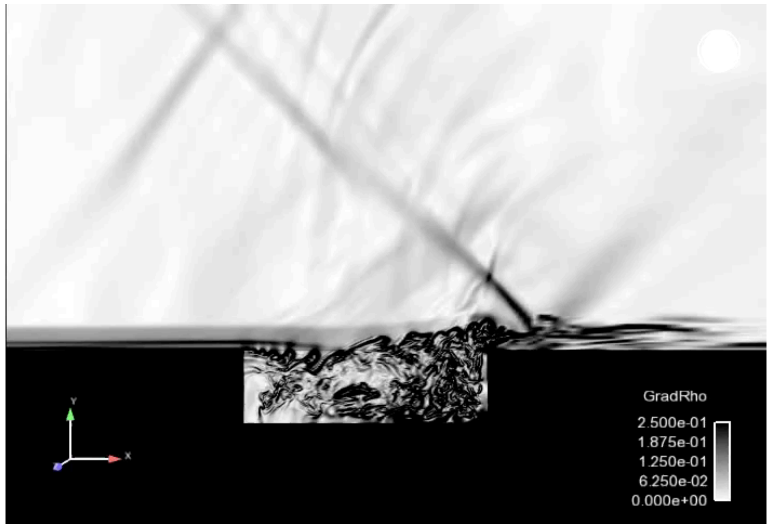
- uncertainty propagation
- Bayesian inference
- multi-objective optimization
- stochastic optimization

High-fidelity simulation: captive carry



High-fidelity simulation: captive carry





\+ *Validated and predictive*: 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

- explore flight envelope
- quantify effects of uncertainties on store load
- robust design of store and cavity

Computational barrier at NASA

The New York Times

Geniuses Wanted: NASA Challenges Coders to Speed Up Its Supercomputer

"Despite tremendous progress made in the past few decades, CFD tools are too slow for simulation of complex geometry flows… [taking] from thousands to millions of computational core-hours."

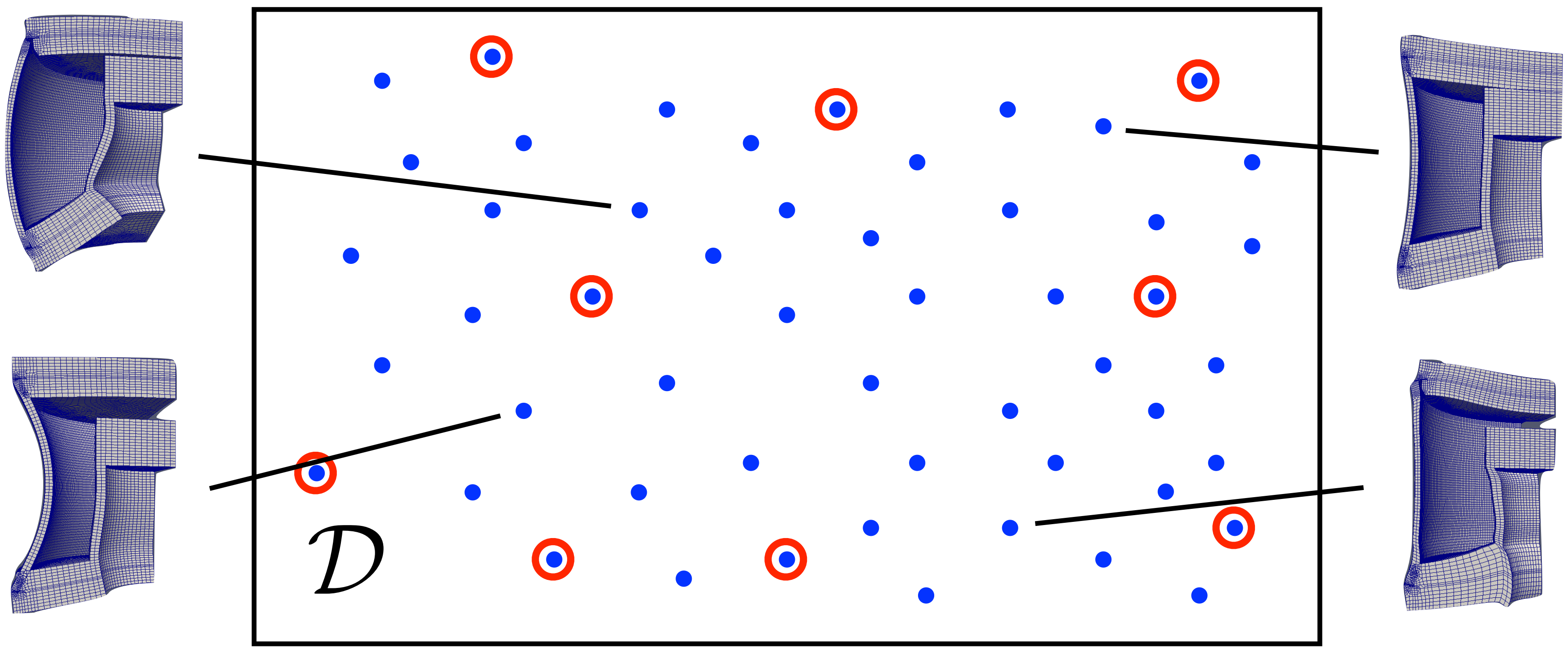
"To enable high-fidelity CFD for multi-disciplinary analysis and design, the speed of computation must be increased by orders of magnitude."

"The desired outcome is any approach that can ***accelerate calculations by a factor of 10x to 1000x****."*

Approach: exploit simulation data

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

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

Model reduction criteria

1. ***Accuracy:*** achieves less than 1% error
2. ***Low cost:*** achieves at least 100x computational savings
3. ***Structure preservation:*** preserves important physical properties
4. ***Robustness:*** guaranteed satisfaction of any error tolerance
5. ***Certification:*** accurately quantify the ROM error

Model reduction: existing approaches

Linear time-invariant systems: mature [Antoulas, 2005]

- Balanced truncation [Moore, 1981; Willcox and Peraire, 2002; Rowley, 2005]
- Transfer-function interpolation [Bai, 2002; Freund, 2003; Gallivan et al, 2004; Baur et al., 2001]

\+ *Accurate*, *reliable*, *certified*: sharp *a priori* error bounds
\+ *Inexpensive*: pre-assemble operators
\+ *Structure preservation*: guaranteed stability

Elliptic/parabolic PDEs: mature [Prud'Homme et al., 2001; Barrault et al., 2004; Rozza et al., 2008]

- Reduced-basis method

\+ *Accurate*, *reliable*, *certified*: sharp *a priori* error bounds, convergence
\+ *Inexpensive*: pre-assemble operators
\+ *Structure preservation*: preserve operator properties

Nonlinear dynamical systems: ineffective

- Proper orthogonal decomposition (POD)–Galerkin [Sirovich, 1987]

– *Inaccurate, unreliable*: often unstable
– *Not certified*: error bounds grow exponentially in time
– *Expensive*: projection insufficient for speedup
– *Structure not preserved*: dynamical-system properties ignored

Our research

Accurate, low-cost, structure-preserving, reliable, certified nonlinear model reduction

- *accuracy*: LSPG projection [C., Bou-Mosleh, Farhat, 2011; C., Barone, Antil, 2017]
- *low cost*: sample mesh [C., Farhat, Cortial, Amsallem, 2013]
- *low cost*: reduce temporal complexity
  [C., Ray, van Bloemen Waanders, 2015; C., Brencher, Haasdonk, Barth, 2017; Choi and C., 2019]
- *structure preservation* [C., Tuminaro, Boggs, 2015; Peng and C., 2017; C., Choi, Sargsyan, 2018]
- *robustness*: projection onto nonlinear manifolds [Lee, C., 2018]
- *robustness*: *h*-adaptivity [C., 2015]
- *certification*: machine learning error models
  [Drohmann and C., 2015; Trehan, C., Durlofsky, 2017; Freno and C., 2019; Pagani, Manzoni, C., 2019]

Our research

Accurate, low-cost, structure-preserving, reliable, certified nonlinear model reduction

- *accuracy*: LSPG projection [C., Bou-Mosleh, Farhat, 2011; C., Barone, Antil, 2017]
- *low cost*: sample mesh [C., Farhat, Cortial, Amsallem, 2013]
- *low cost*: reduce temporal complexity
  [C., Ray, van Bloemen Waanders, 2015; C., Brencher, Haasdonk, Barth, 2017; Choi and C., 2019]
- *structure preservation* [C., Tuminaro, Boggs, 2015; Peng and C., 2017; C., Choi, Sargsyan, 2018]
- *robustness*: projection onto nonlinear manifolds [Lee, C., 2018]
- *robustness*: *h*-adaptivity [C., 2015]
- *certification*: machine learning error models
  [Drohmann and C., 2015; Trehan, C., Durlofsky, 2017; Freno and C., 2019; Pagani, Manzoni, C., 2019]

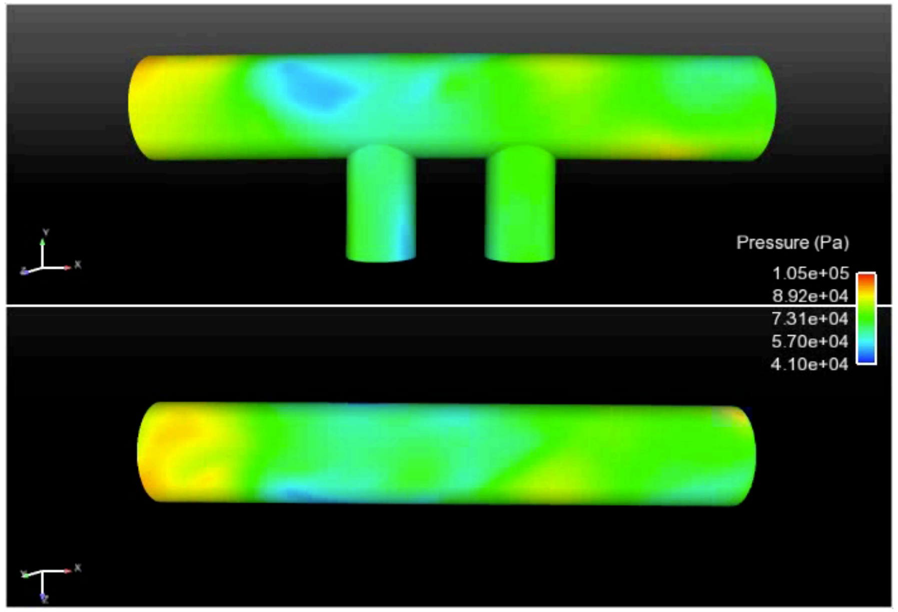
Matthew Barone

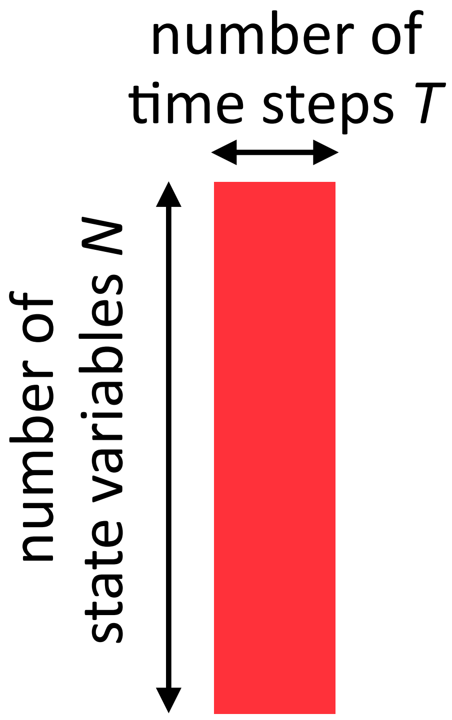
Harbir Antil (GMU)

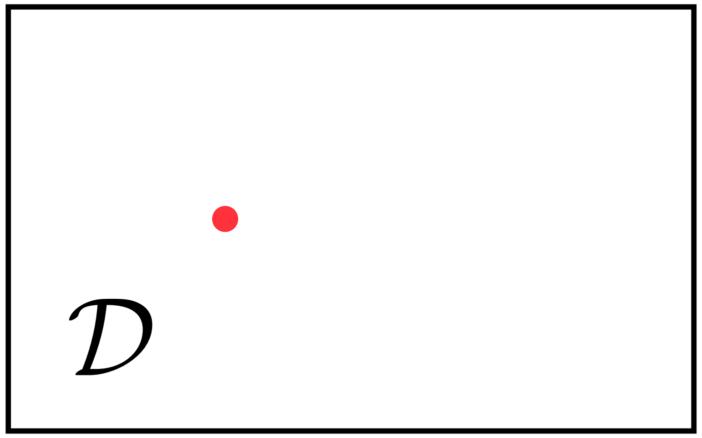
Training simulations: state tensor

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

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 $\boldsymbol{\mu} \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$



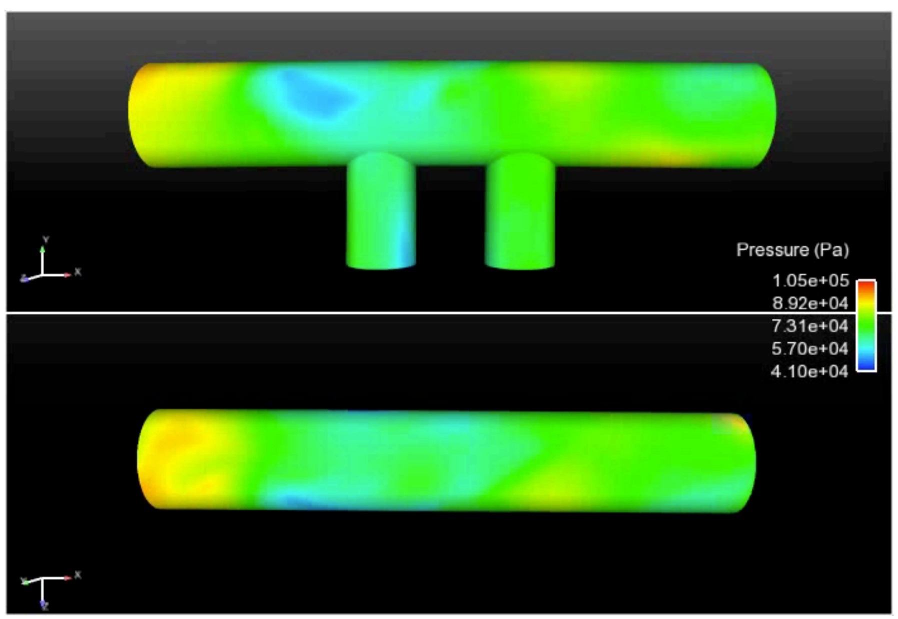




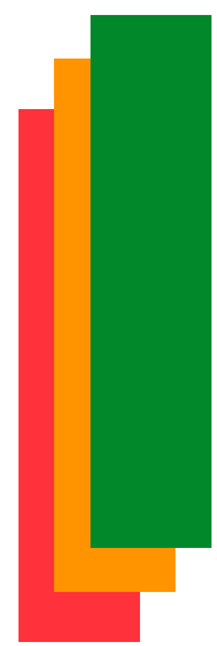
Training simulations: state tensor

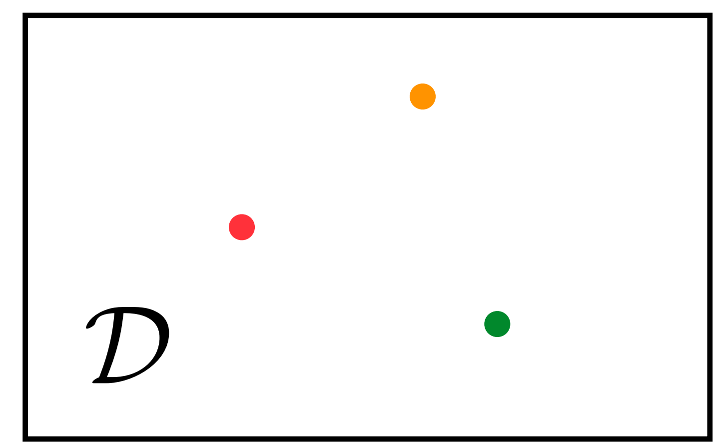
$$\text{ODE:} \quad \frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}; t, \boldsymbol{\mu})$$

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 $\boldsymbol{\mu} \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$



$\mathcal{X} =$



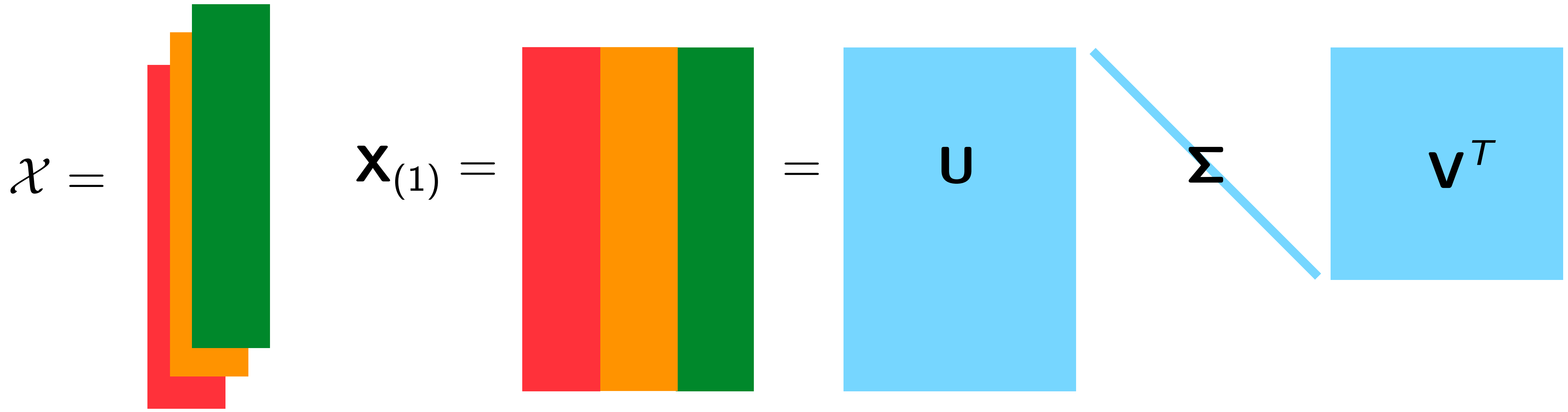


Tensor decomposition

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

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 $\boldsymbol{\mu} \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$

Compute dominant left singular vectors of mode-1 unfolding

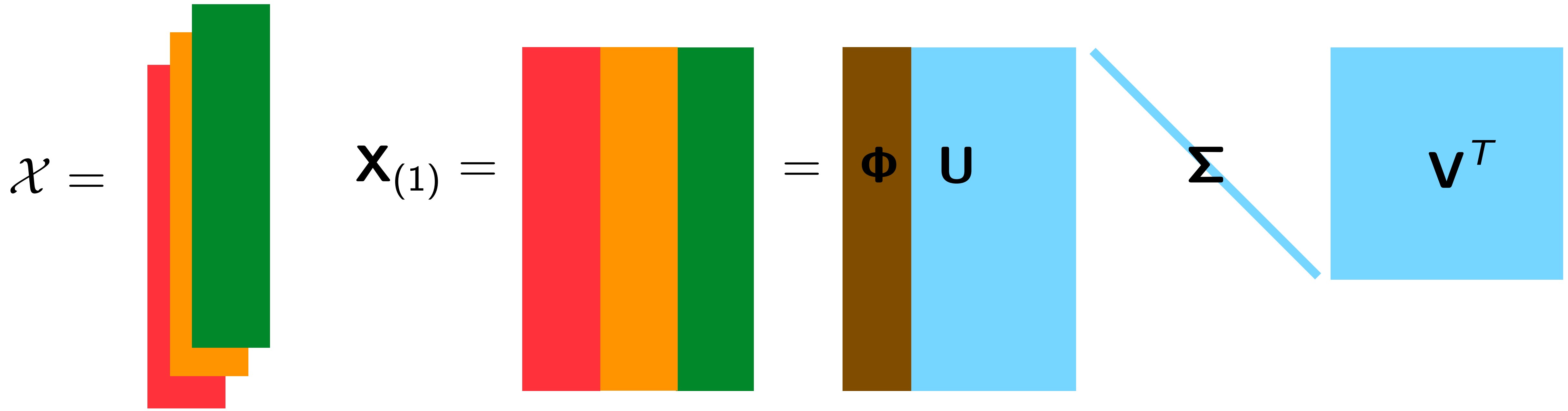


Tensor decomposition

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

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 $\boldsymbol{\mu} \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$

Compute dominant left singular vectors of mode-1 unfolding

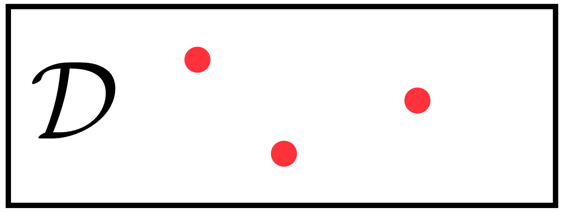


$\boldsymbol{\Phi}$ *columns are principal components of the spatial simulation data*

How to integrate these data with the computational model?

Previous state of the art: POD–Galerkin

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

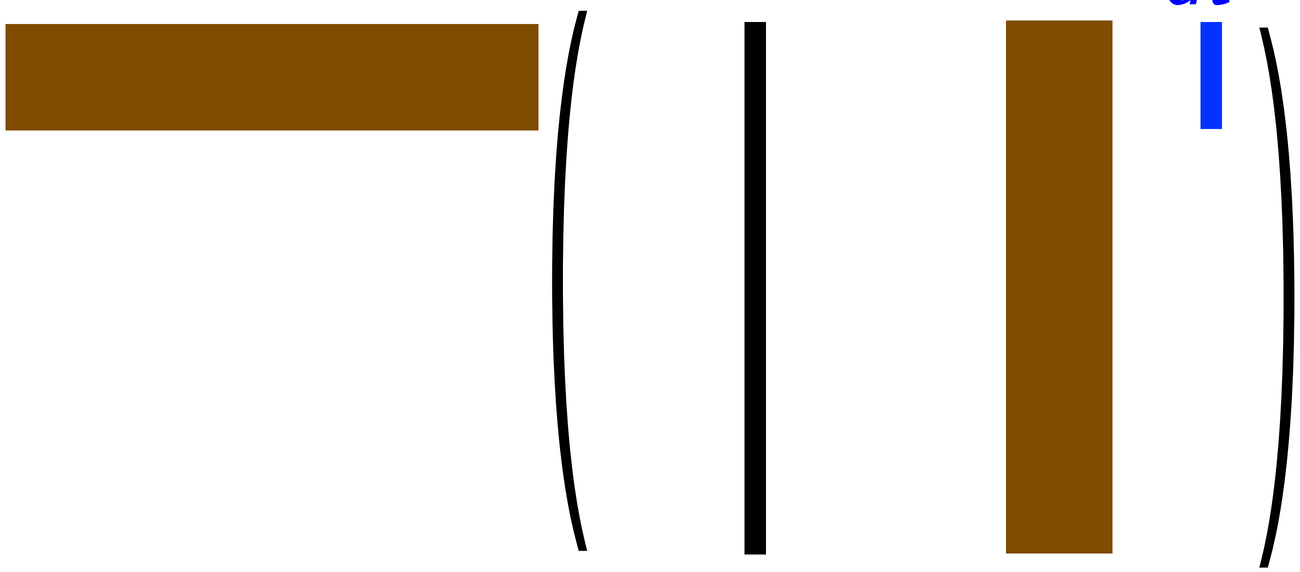


1. *Training:* Solve ODE for $\mu \in \mathcal{D}_{\text{training}}$ and collect simulation data
2. *Machine learning:* Identify structure in data
3. *Reduction:* Reduce the cost of solving ODE for $\boldsymbol{\mu} \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$

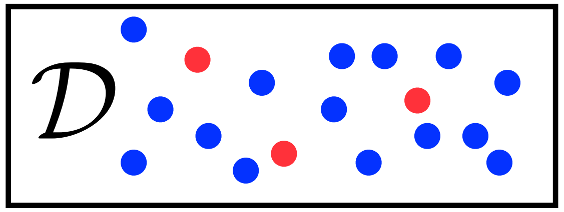
1. Reduce the number of unknowns
2. Reduce the number of equations

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

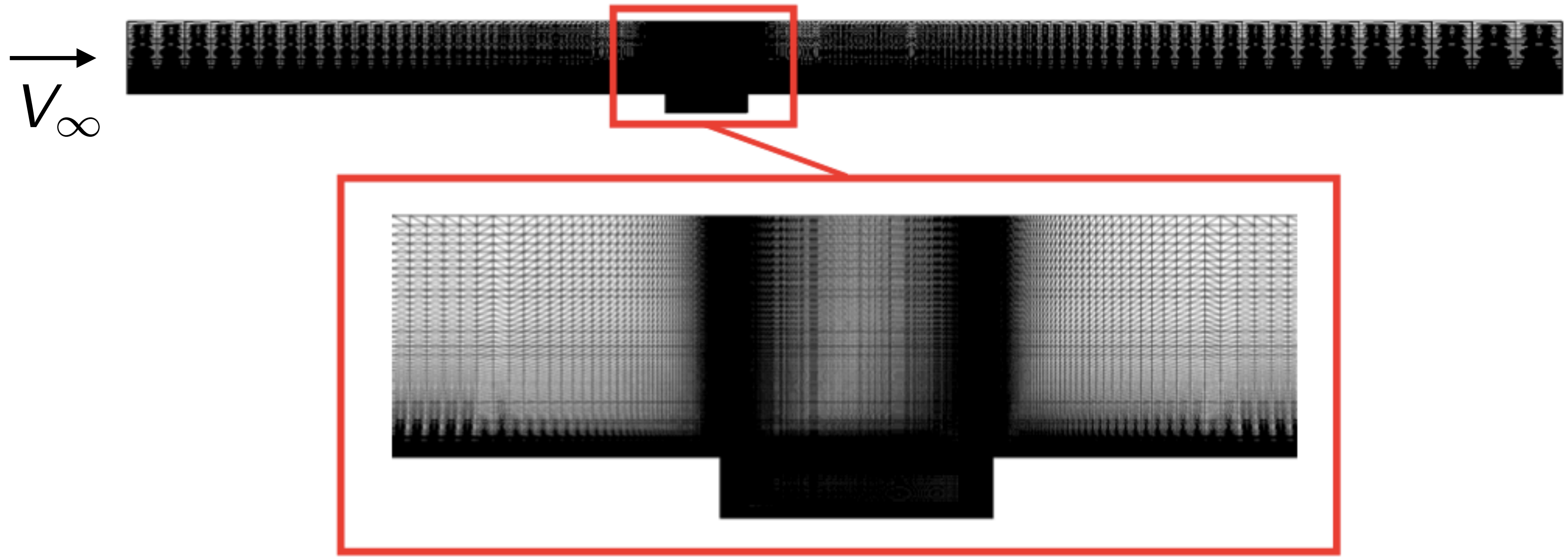
$$\boldsymbol{\Phi}^T\left(\mathbf{f}(\boldsymbol{\Phi}\hat{\mathbf{x}}; t, \boldsymbol{\mu}) - \boldsymbol{\Phi}\frac{d\hat{\mathbf{x}}}{dt}\right) = 0$$



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



Captive carry



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

Spatial discretization

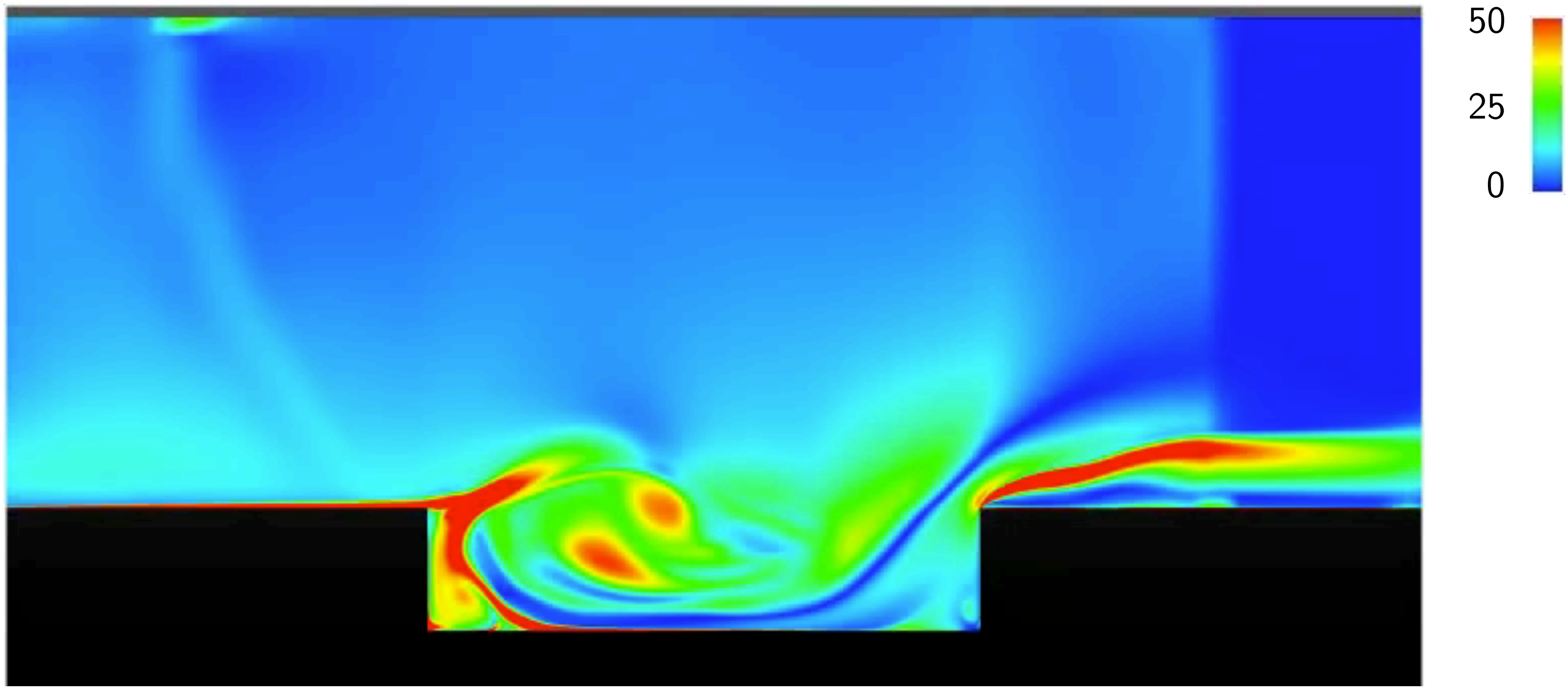
- 2nd-order finite volume
- DES turbulence model
- 1.2×10^6 degrees of freedom

Temporal discretization

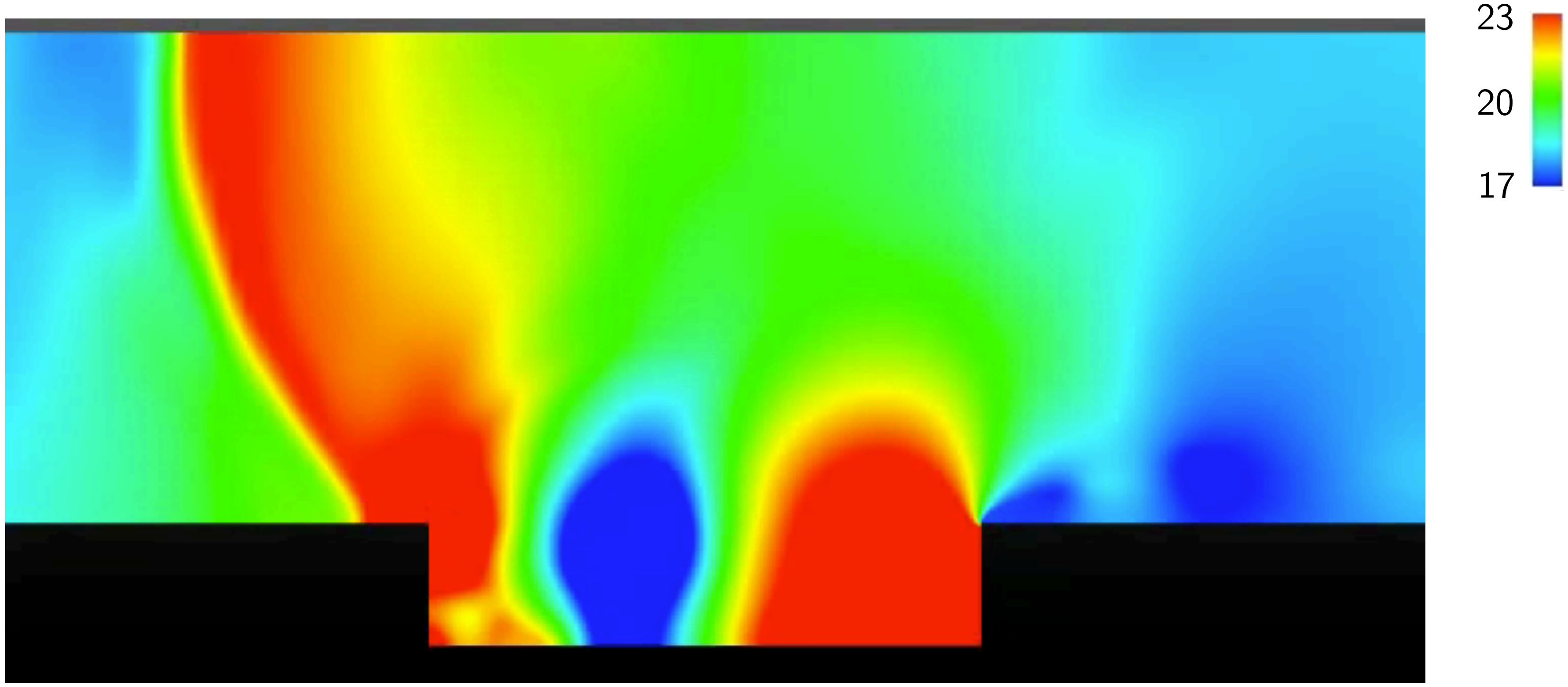
- 2nd-order BDF
- Verified time step $\Delta t = 1.5 \times 10^{-3}$
- 8.3×10^3 time instances

High-fidelity model solution

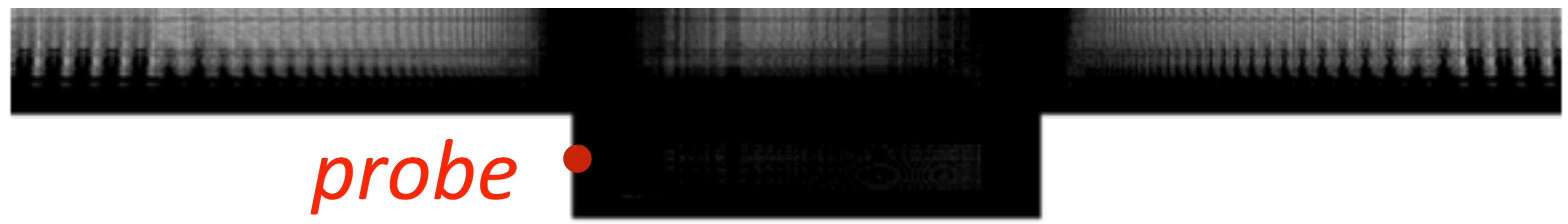
vorticity field

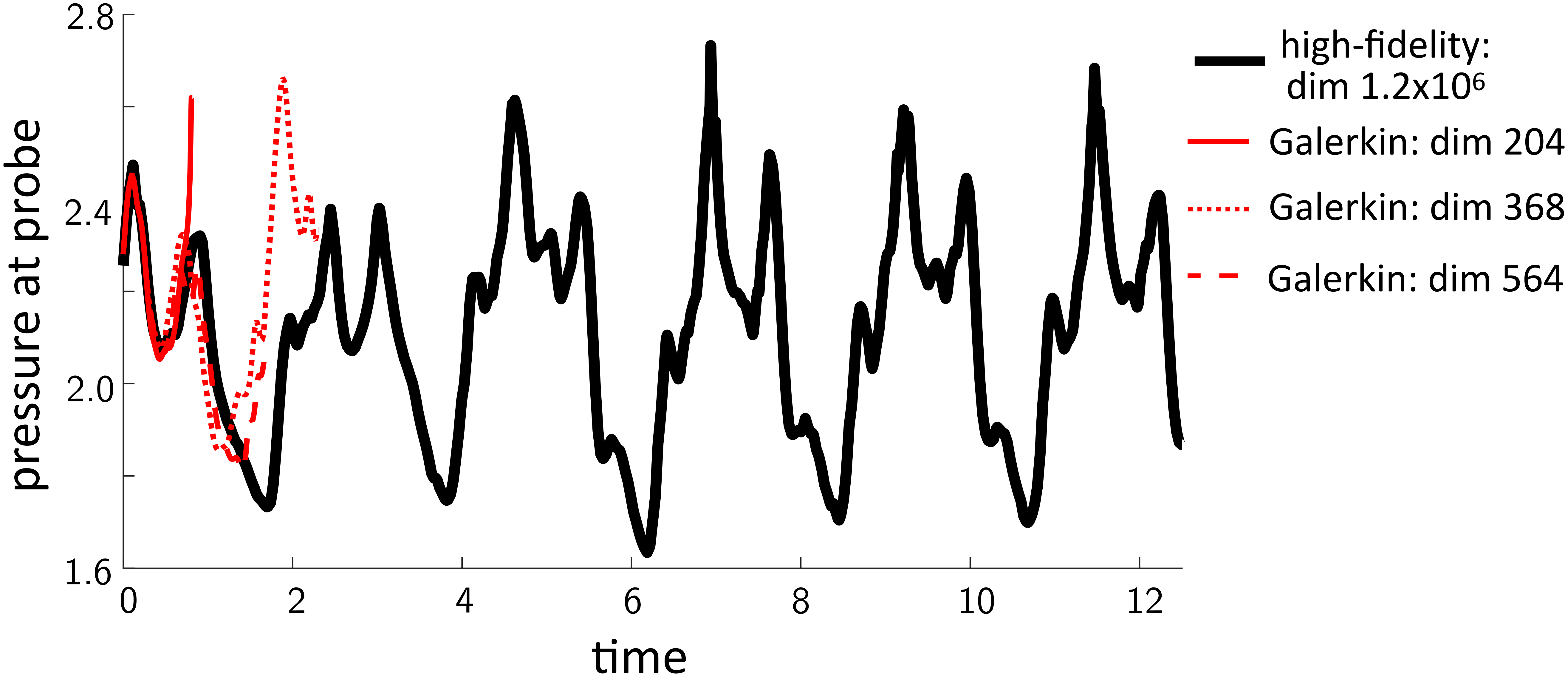


pressure field



Galerkin performance





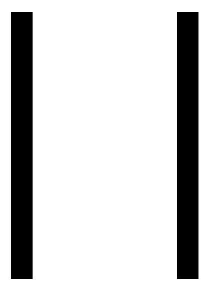
- *Galerkin projection fails regardless of basis dimension*

Can we construct a better projection?

Galerkin: time-continuous optimality

ODE

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



Galerkin ODE

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

+ *Time-continuous Galerkin solution*: optimal in the minimum-residual sense:

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

$$\mathbf{r}(\mathbf{v}, \mathbf{x}; t) := \mathbf{v} - \mathbf{f}(\mathbf{x}; t)$$

OΔE

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

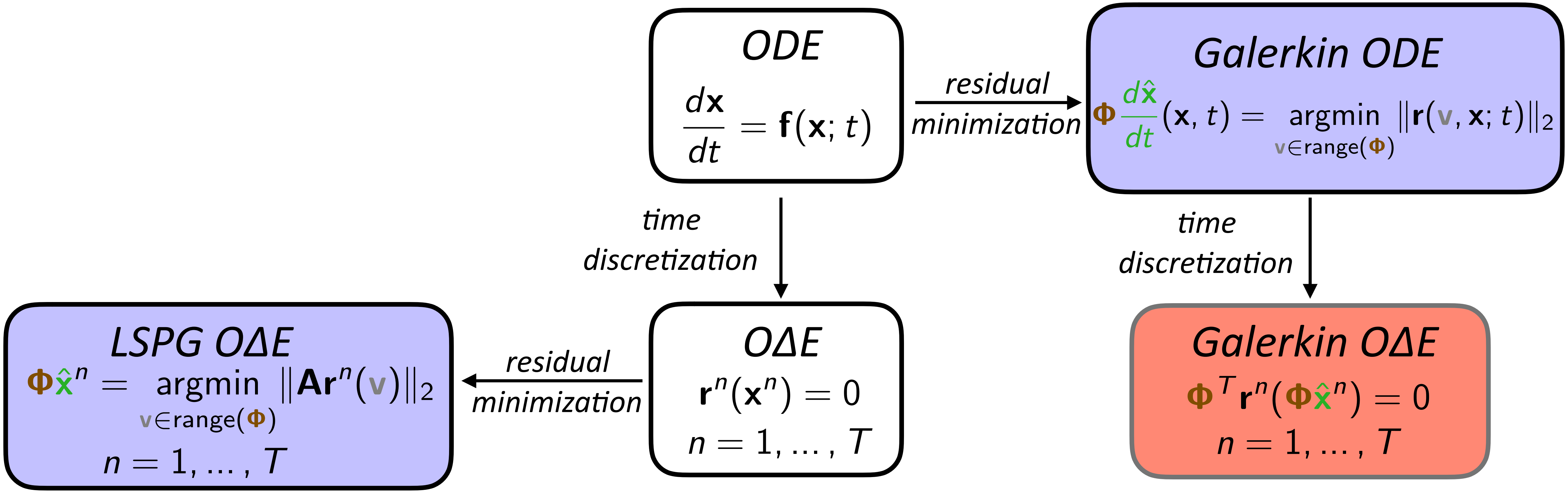
Galerkin OΔE

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

$$\mathbf{r}^n(\mathbf{x}) := \alpha_0 \mathbf{x} - \Delta t \beta_0 \mathbf{f}(\mathbf{x}; t^n) + \sum_{j=1}^{k} \alpha_j \mathbf{x}^{n-j} - \Delta t \sum_{j=1}^{k} \beta_j \mathbf{f}(\mathbf{x}^{n-j}; t^{n-j})$$

- *Time-discrete Galerkin solution*: not generally optimal in any sense

Residual minimization and time discretization



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

$$\boldsymbol{\Phi}\hat{\mathbf{x}}^n = \underset{\mathbf{v}\in\text{range}(\boldsymbol{\Phi})}{\text{argmin}} \|\mathbf{A}\mathbf{r}^n(\mathbf{v})\|_2 \quad \Leftrightarrow \quad \boldsymbol{\Psi}^n(\hat{\mathbf{x}}^n)^T\mathbf{r}^n(\boldsymbol{\Phi}\hat{\mathbf{x}}^n) = 0$$

$$\boldsymbol{\Psi}^n(\hat{\mathbf{x}}^n) := \mathbf{A}^T\mathbf{A}\left(\alpha_0\mathbf{I} - \Delta t\beta_0\frac{\partial\mathbf{f}}{\partial\mathbf{x}}(\boldsymbol{\Phi}\hat{\mathbf{x}}^n; t)\right)\boldsymbol{\Phi}$$

*Least-squares Petrov–Galerkin (**LSPG**) projection*

Discrete-time error bound

Theorem [C., Barone, Antil, 2017]

If the following conditions hold:

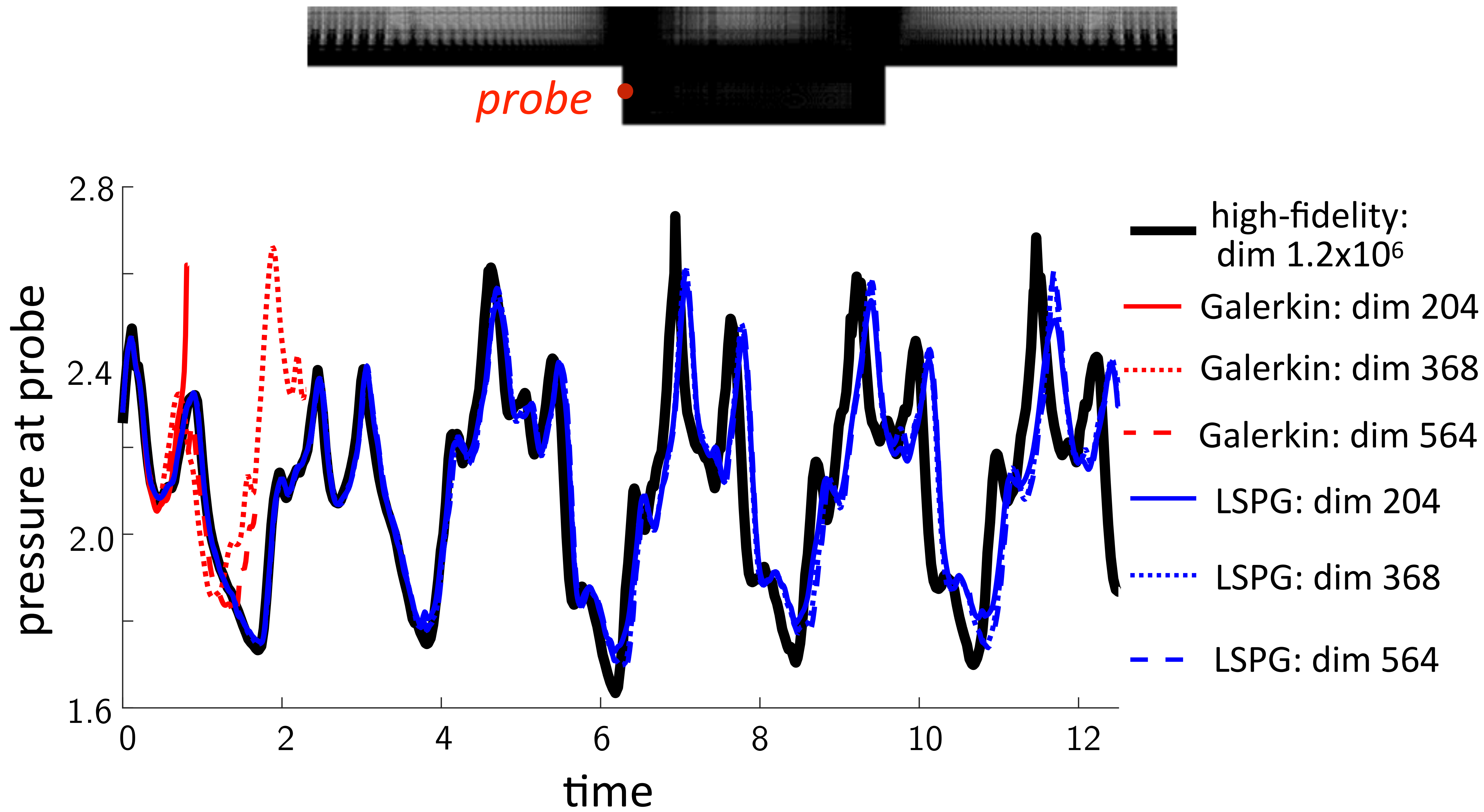
1. $\mathbf{f}(\cdot;t)$ is Lipschitz continuous with Lipschitz constant κ
2. The time step Δt is small enough such that $0 < h := |\alpha_0| - |\beta_0|\kappa\Delta t$,
3. A backward differentiation formula (BDF) time integrator is used,
4. LSPG employs $\mathbf{A} = \mathbf{I}$, then

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

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

+ LSPG sequentially minimizes the error bound

LSPG performance



+ *LSPG is far more accurate than Galerkin*

Our research

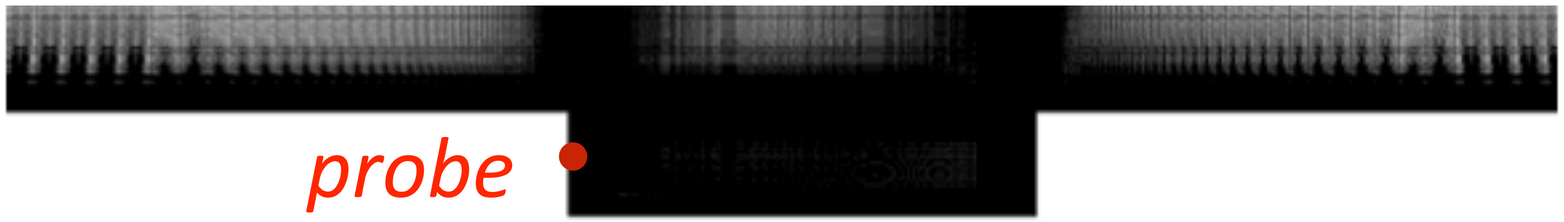
Accurate, low-cost, structure-preserving, reliable, certified nonlinear model reduction

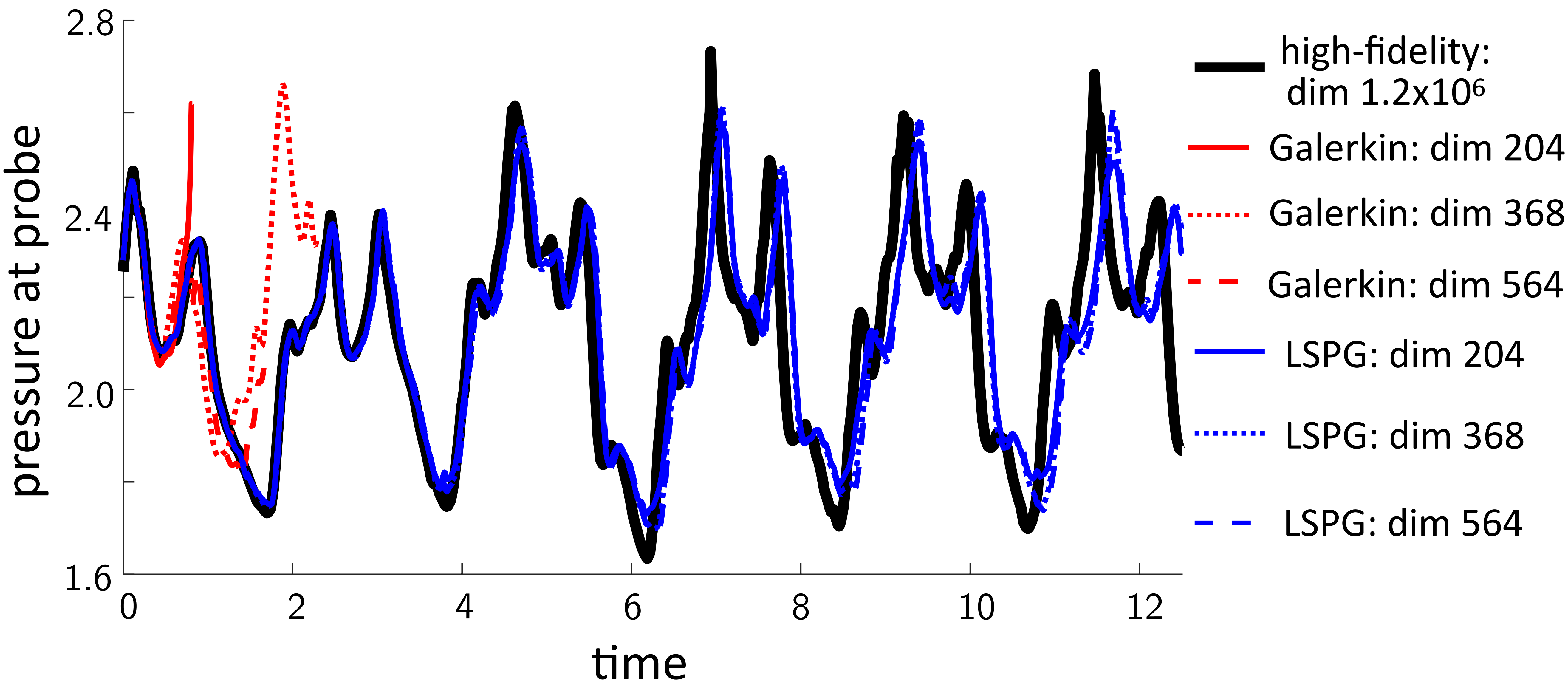
- *accuracy*: LSPG projection [C., Bou-Mosleh, Farhat, 2011; C., Barone, Antil, 2017]
- *low cost*: sample mesh [C., Farhat, Cortial, Amsallem, 2013]
- *low cost*: reduce temporal complexity [C., Ray, van Bloemen Waanders, 2015; C., Brencher, Haasdonk, Barth, 2017; Choi and C., 2019]
- *structure preservation* [C., Tuminaro, Boggs, 2015; Peng and C., 2017; C., Choi, Sargsyan, 2017]
- *robustness*: projection onto nonlinear manifolds [Lee, C., 2018]
- *robustness*: *h*-adaptivity [C., 2015]
- *certification*: machine learning error models [Drohmann and C., 2015; Trehan, C., Durlofsky, 2017; Freno and C., 2019; Pagani, Manzoni, C., 2019]

Charbel Farhat (Stanford)

Julien Cortial (Stanford)

Wall-time problem





- *High-fidelity simulation*: 1 hour, 48 cores
- *Fastest LSPG simulation*: 1.3 hours, 48 cores

Why does this occur?

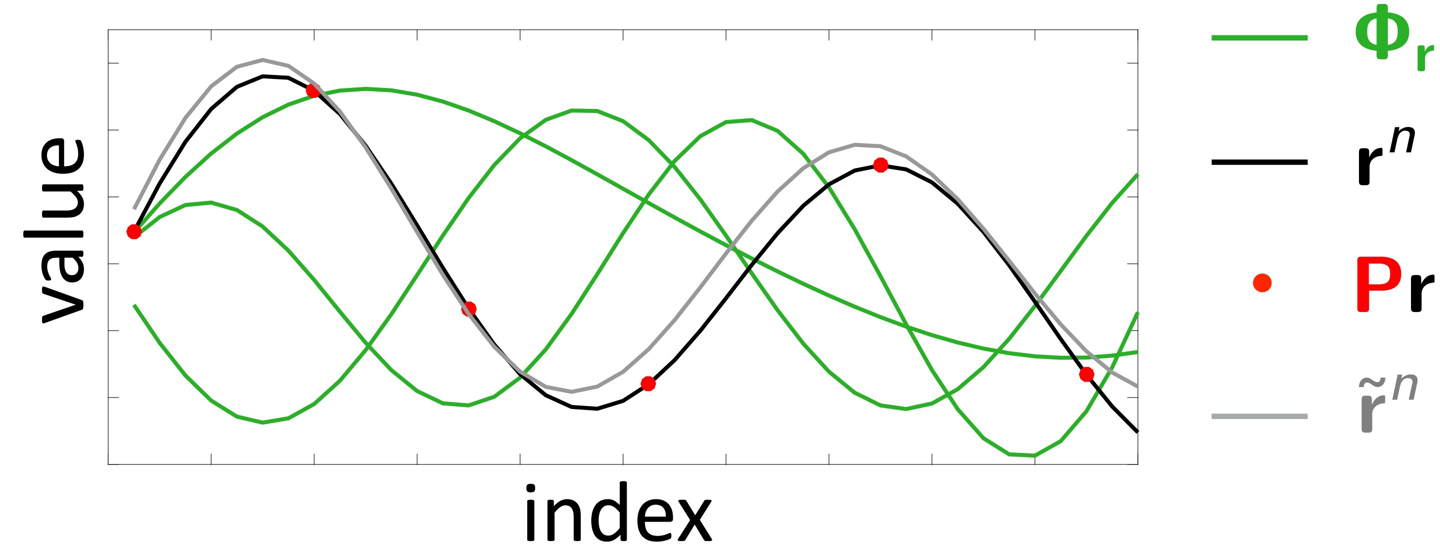
Can we fix it?

Cost reduction by gappy PCA [Everson and Sirovich, 1995]

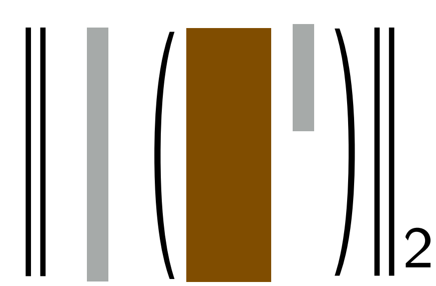
$$\underset{\hat{\mathbf{v}}}{\text{minimize}} \| \mathbf{A} \, \mathbf{r}^n(\boldsymbol{\Phi} \hat{\mathbf{v}}) \|_2$$

Can we select **A** *to make this less expensive?*

- **Training:** collect residual tensor $\mathcal{R}^{ijk}$ while solving ODE for $\boldsymbol{\mu} \in \mathcal{D}_{\text{training}}$
- **Machine learning:** compute residual PCA $\boldsymbol{\Phi}_{\mathbf{r}}$ and sampling matrix $\mathbf{P}$
- **Reduction**: compute regression approximation $\mathbf{r}^n \approx \tilde{\mathbf{r}}^n = \boldsymbol{\Phi}_{\mathbf{r}}(\mathbf{P}\boldsymbol{\Phi}_{\mathbf{r}})^+\mathbf{P}\mathbf{r}^n$



$$\underset{\hat{\mathbf{v}}}{\text{minimize}} \| \tilde{\mathbf{r}}^n(\boldsymbol{\Phi} \hat{\mathbf{v}}) \|_2$$

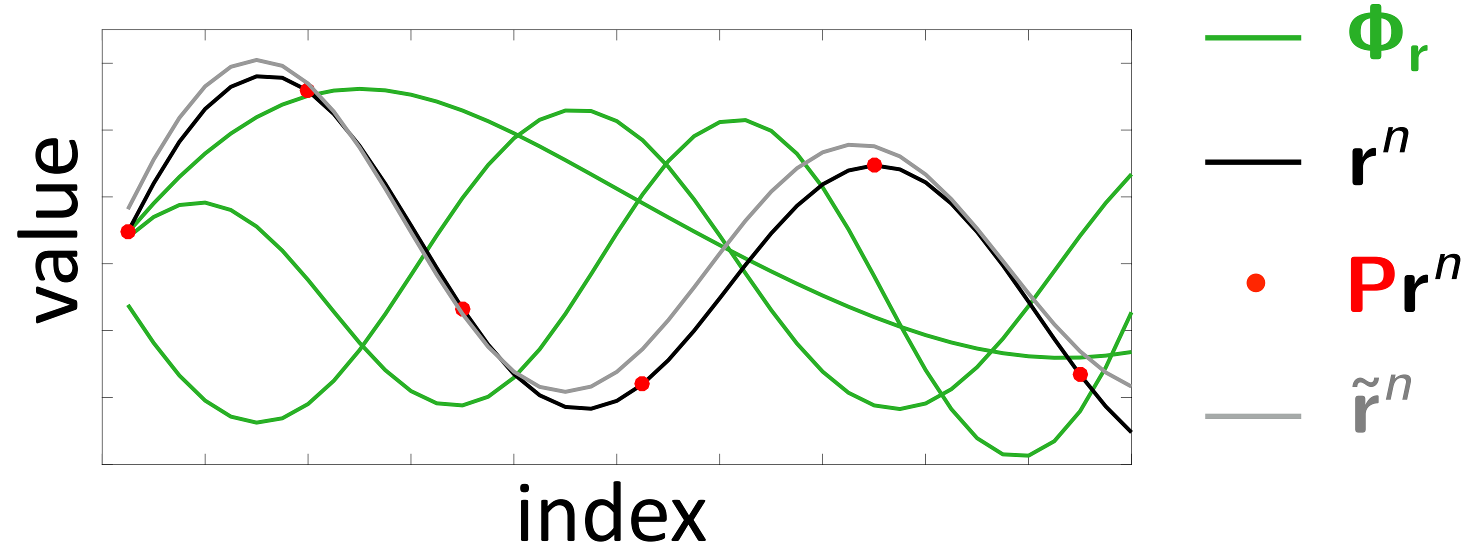


Cost reduction by gappy PCA [Everson and Sirovich, 1995]

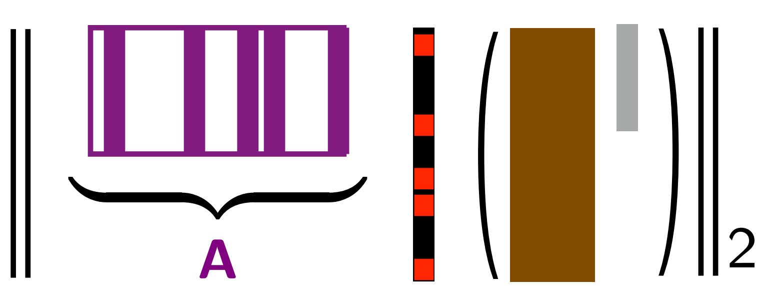
$$\underset{\hat{\mathbf{v}}}{\text{minimize}} \left\| \mathbf{A} \, \mathbf{r}^n(\boldsymbol{\Phi}\hat{\mathbf{v}}) \right\|_2$$

Can we select **A** *to make this less expensive?*

- **Training:** collect residual tensor $\mathcal{R}^{ijk}$ while solving ODE for $\boldsymbol{\mu} \in \mathcal{D}_{\text{training}}$
- **Machine learning:** compute residual PCA $\boldsymbol{\Phi}_{\mathbf{r}}$ and sampling matrix $\mathbf{P}$
- **Reduction**: compute regression approximation $\mathbf{r}^n \approx \tilde{\mathbf{r}}^n = \boldsymbol{\Phi}_{\mathbf{r}}(\mathbf{P}\boldsymbol{\Phi}_{\mathbf{r}})^+\mathbf{P}\mathbf{r}^n$



$$\underset{\hat{\mathbf{v}}}{\text{minimize}} \left\| (\mathbf{P}\boldsymbol{\Phi}_{\mathbf{r}})^+\mathbf{P} \, \mathbf{r}^n(\boldsymbol{\Phi}\hat{\mathbf{v}}) \right\|_2$$

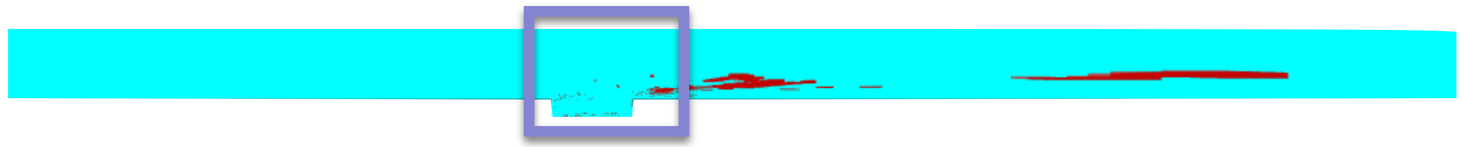


\+ *Only a few elements of* $\mathbf{r}^n$ *must be computed*

Sample mesh [C., Farhat, Cortial, Amsallem, 2013]

$$\underset{\hat{\mathbf{v}}}{\text{minimize}} \| (\mathbf{P}\boldsymbol{\Phi}_\mathbf{r})^+ \underbrace{\mathbf{P}\mathbf{r}^n}(\boldsymbol{\Phi}\hat{\mathbf{v}}) \|_2$$

sample mesh



+ *HPC on a laptop*

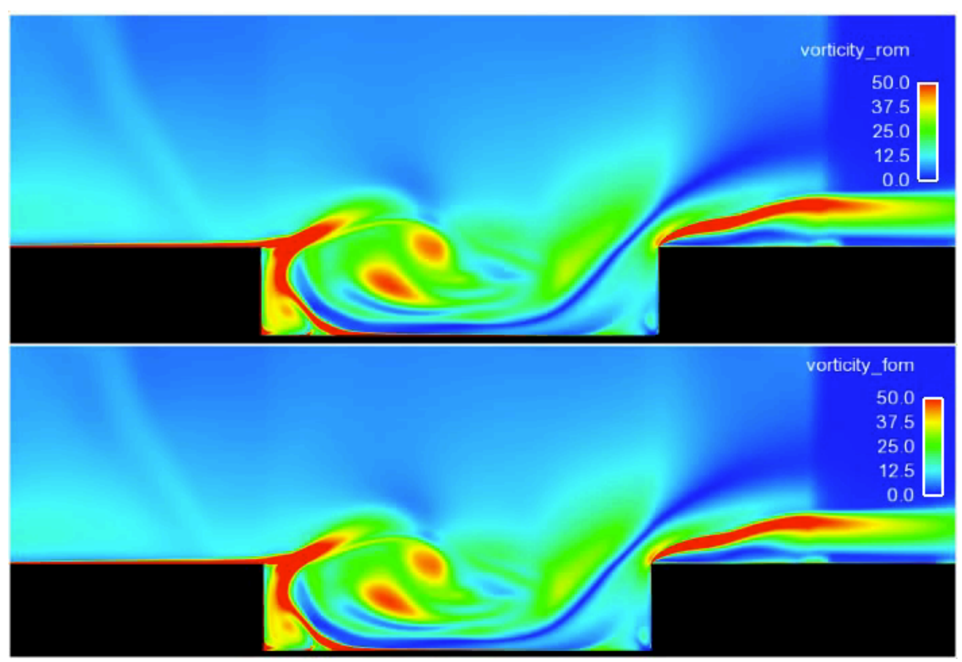
vorticity field | *pressure field*

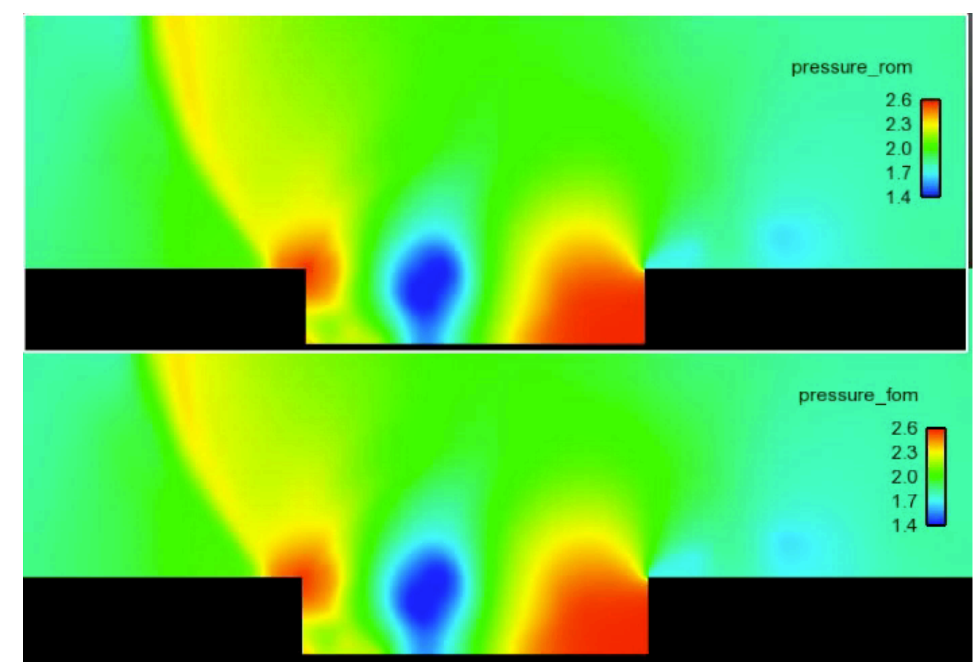
LSPG ROM with $\mathbf{A} = (\mathbf{P}\boldsymbol{\Phi}_\mathbf{r})^+\mathbf{P}$

32 min, 2 cores

high-fidelity

5 hours, 48 cores



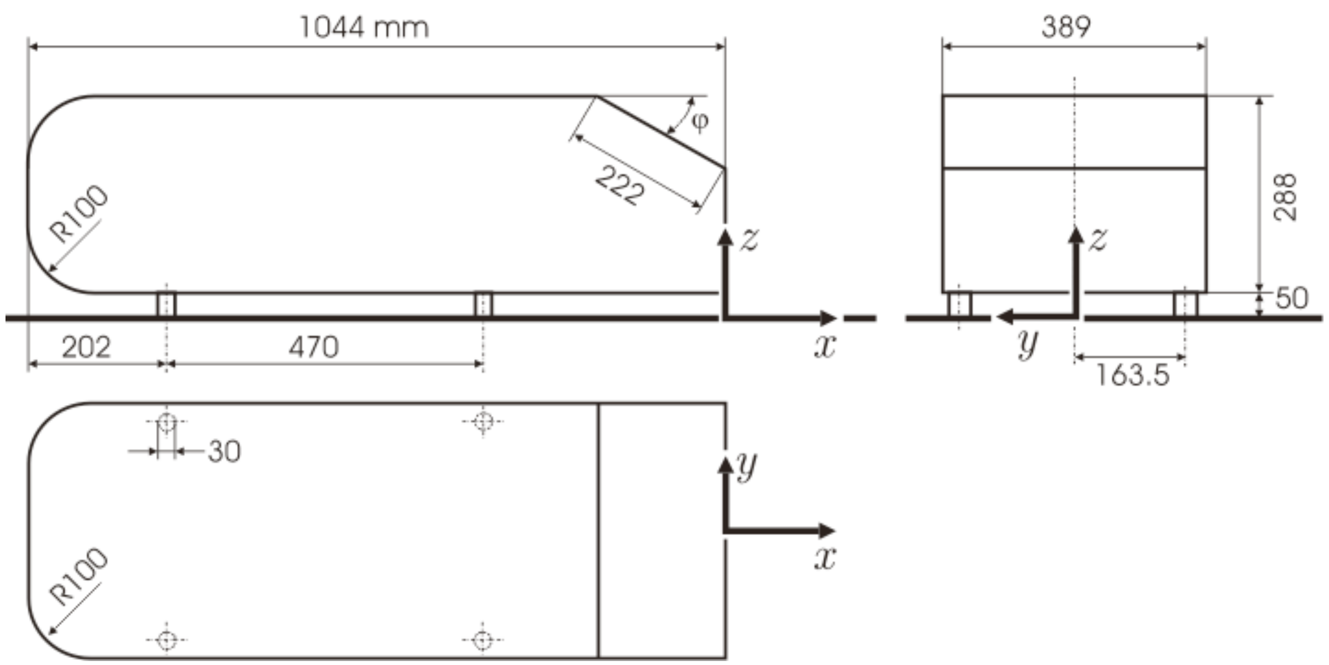


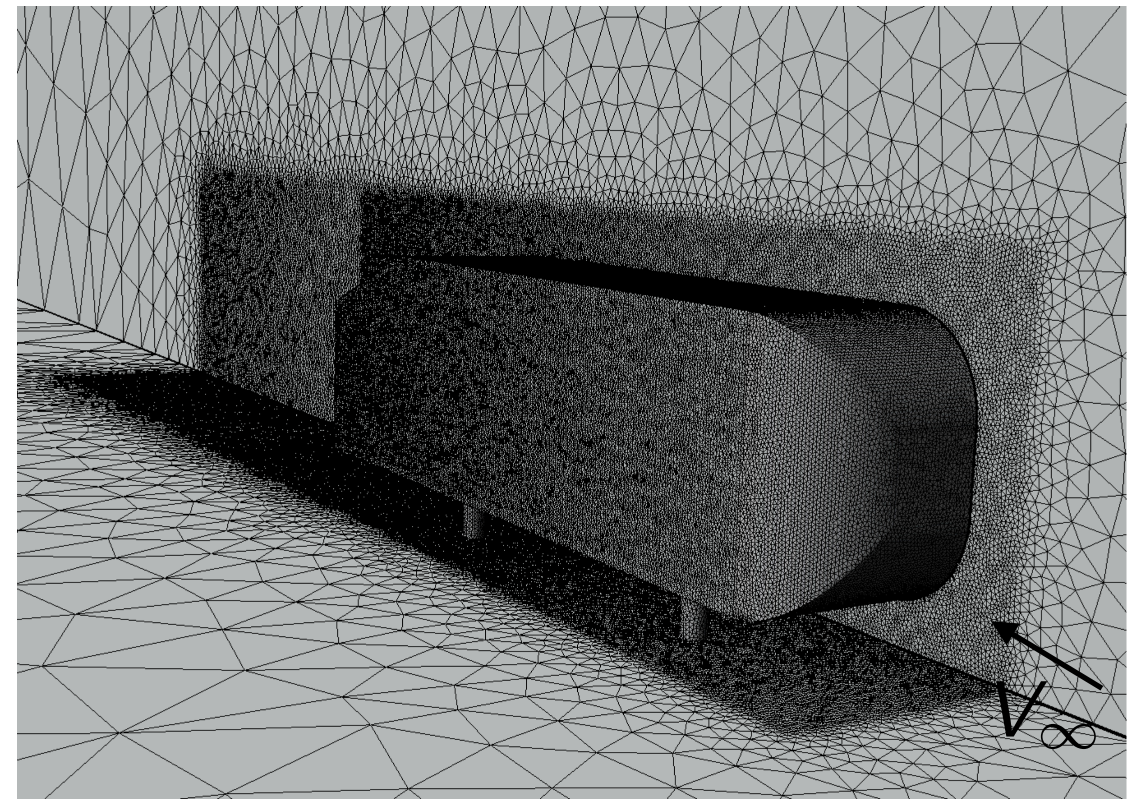
+ *229x savings in core–hours*

+ *< 1% error in time-averaged drag*

Implemented in three computational-mechanics codes at Sandia

Ahmed body [Ahmed, Ramm, Faitin, 1984]





- Unsteady Navier–Stokes
- Re = 4.3 x 10^6
- M_∞ = 0.175

Spatial discretization

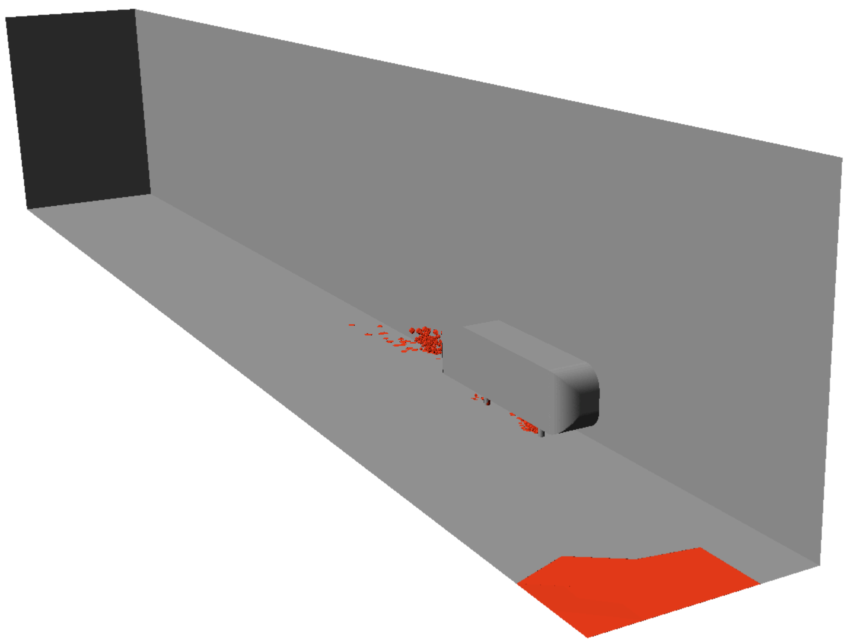
- 2nd-order finite volume
- DES turbulence model
- 1.7×10^7 degrees of freedom

Temporal discretization

- 2nd-order BDF
- Time step $\Delta t = 8 \times 10^{-5}\mathrm{s}$
- 1.3×10^3 time instances

Ahmed body results [C., Farhat, Cortial, Amsallem, 2013]

sample mesh



+ *HPC on a laptop*

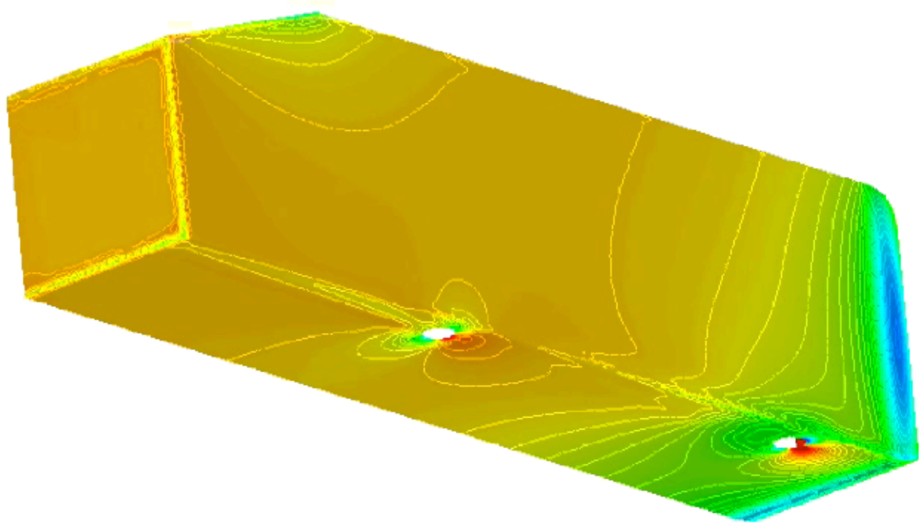
LSPG ROM with $\mathbf{A} = (\mathbf{P}\mathbf{\Phi}_r)^+\mathbf{P}$

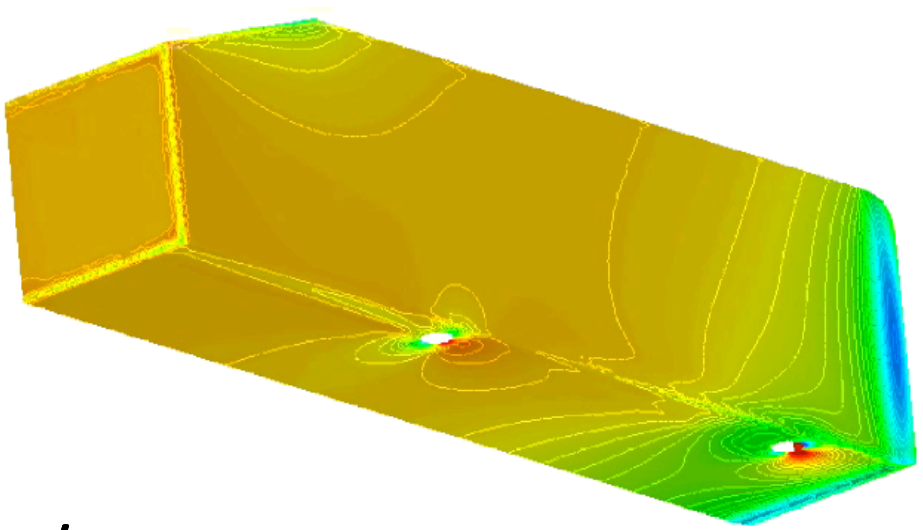
4 hours, 4 cores

high-fidelity model

13 hours, 512 cores

pressure field





+ *438x savings in core–hours*

+ *Largest nonlinear dynamical system on which ROM has ever had success*

Our research

Accurate, low-cost, structure-preserving, reliable, certified nonlinear model reduction

- *accuracy*: LSPG projection [C., Bou-Mosleh, Farhat, 2011; C., Barone, Antil, 2017]
- *low cost*: sample mesh [C., Farhat, Cortial, Amsallem, 2013]
- *low cost*: reduce temporal complexity
  [C., Ray, van Bloemen Waanders, 2015; C., Brencher, Haasdonk, Barth, 2017; Choi and C., 2019]
- *structure preservation* [C., Tuminaro, Boggs, 2015; Peng and C., 2017; C., Choi, Sargsyan, 2018]
- *robustness*: projection onto nonlinear manifolds [Lee, C., 2018]
- *robustness*: *h*-adaptivity [C., 2015]
- *certification*: machine learning error models
- [Drohmann and C., 2015; Trehan, C., Durlofsky, 2017; Freno and C., 2019; Pagani, Manzoni, C., 2019]

Our research

Accurate, low-cost, structure-preserving, reliable, certified nonlinear model reduction

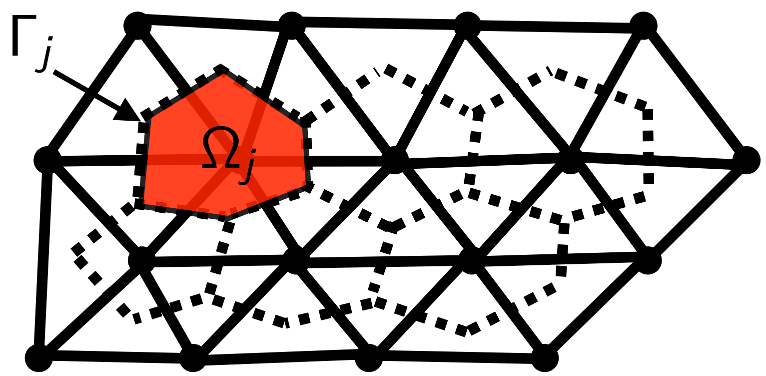
- *accuracy*: LSPG projection [C., Bou-Mosleh, Farhat, 2011; C., Barone, Antil, 2017]
- *low cost*: sample mesh [C., Farhat, Cortial, Amsallem, 2013]
- *low cost*: reduce temporal complexity [C., Ray, van Bloemen Waanders, 2015; C., Brencher, Haasdonk, Barth, 2017; Choi and C., 2019]
- *structure preservation* [C., Tuminaro, Boggs, 2015; Peng and C., 2017; C., Choi, Sargsyan, 2018]
- *robustness*: projection onto nonlinear manifolds [Lee, C., 2018]
- *robustness*: *h*-adaptivity [C., 2015]
- *certification*: machine learning error models [Drohmann and C., 2015; Trehan, C., Durlofsky, 2017; Freno and C., 2019; Pagani, Manzoni, C., 2019]

Youngsoo Choi

Syuzanna Sargsyan (U Washington)

Finite-volume method

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



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

- 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

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

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

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

Conservative model reduction [C., Choi, Sargsyan, 2018]

Galerkin

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

- min. sum of squared conservation-violation **rates**

LSPG

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

- min. sum of squared conservation violations **over time step *n***

- Neither enforces conservation!

Conservative Galerkin

$$\underset{\mathbf{v}\in \text{range}(\Phi)}{\text{minimize}} \|\mathbf{r}(\mathbf{v}, \mathbf{x}; t)\|_2$$

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

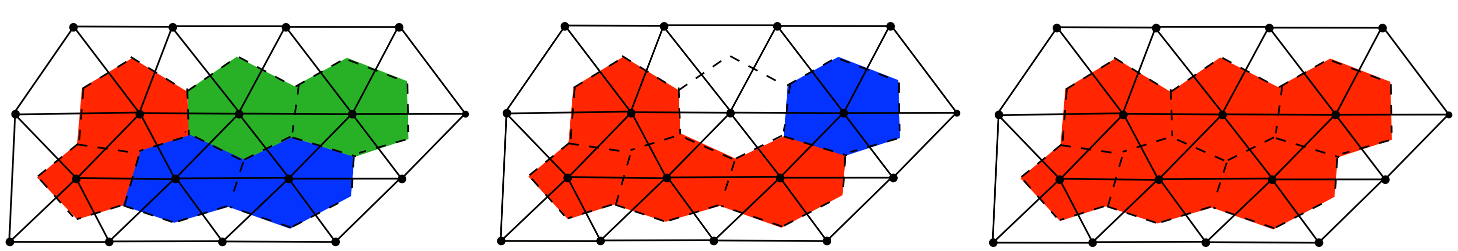
- min. sum of squared conservation-violation **rates** subject to zero conservation-violation **rates** over subdomains

Conservative LSPG

$$\underset{\mathbf{v}\in \text{range}(\Phi)}{\text{minimize}} \|\mathbf{A}\mathbf{r}^n(\mathbf{v})\|_2$$

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

- min. sum of squared conservation violations **over time step *n*** subject to zero conservation violations **over time step *n*** over subdomains



+ Conservation enforced over subdomains!

- **Experiments**: enforcing global conservation can reduce error by 10X

Our research

Accurate, low-cost, structure-preserving, reliable, certified nonlinear model reduction

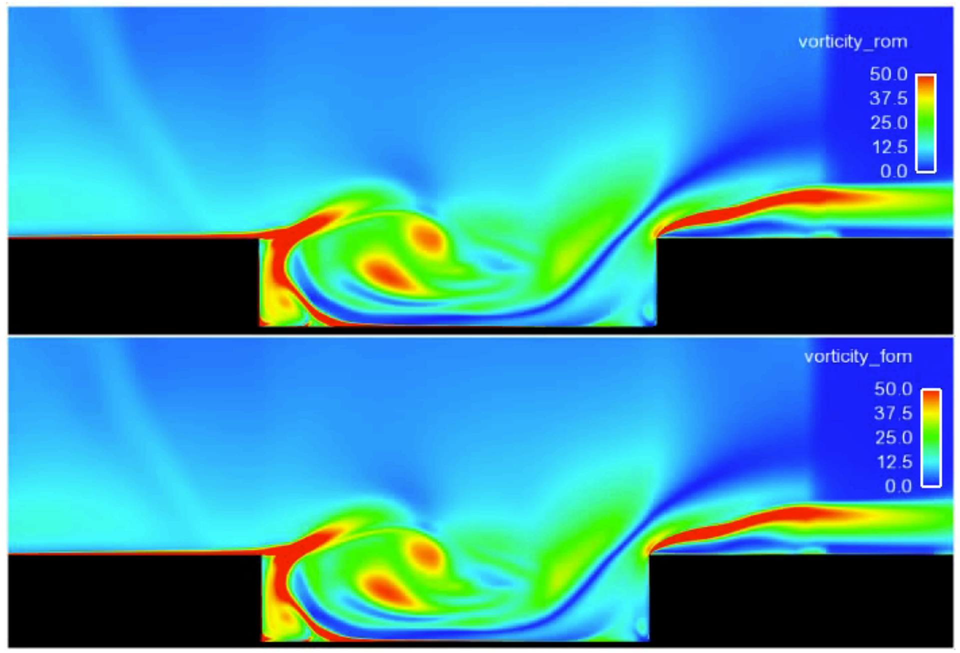
- *accuracy*: LSPG projection [C., Bou-Mosleh, Farhat, 2011; C., Barone, Antil, 2017]
- *low cost*: sample mesh [C., Farhat, Cortial, Amsallem, 2013]
- *low cost*: reduce temporal complexity [C., Ray, van Bloemen Waanders, 2015; C., Brencher, Haasdonk, Barth, 2017; Choi and C., 2019]
- *structure preservation* [C., Tuminaro, Boggs, 2015; Peng and C., 2017; C., Choi, Sargsyan, 2018]
- *robustness*: projection onto nonlinear manifolds [Lee, C., 2018]
- *robustness*: *h*-adaptivity [C., 2015]
- *certification*: machine learning error models [Drohmann and C., 2015; Trehan, C., Durlofsky, 2017; Freno and C., 2019; Pagani, Manzoni, C., 2019]

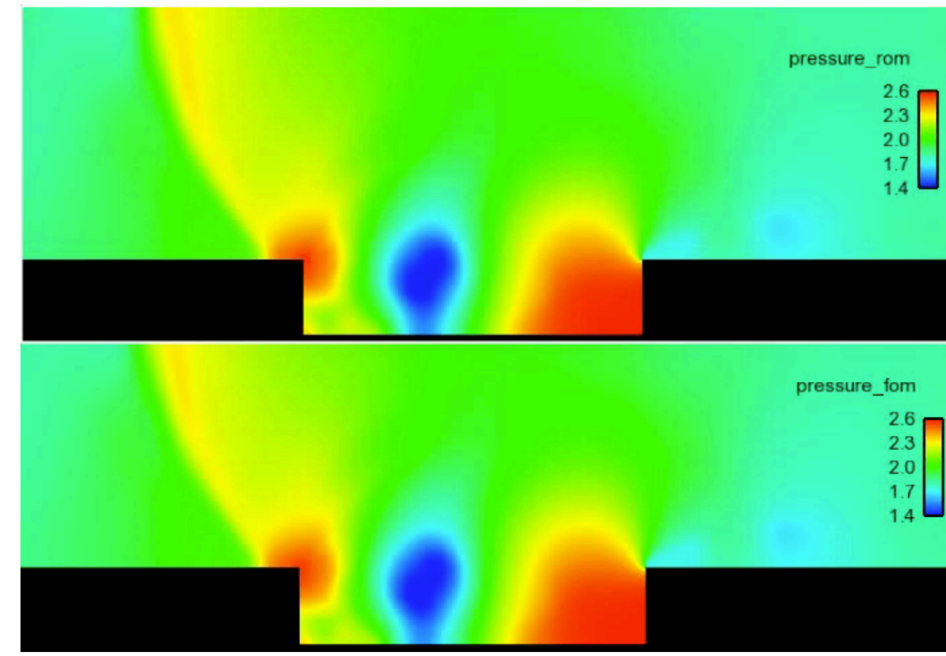
Kookjin Lee

Model reduction can work well…

vorticity field | *pressure field*

LSPG ROM with
$\mathbf{A} = (\mathbf{P}\boldsymbol{\Phi}_{\mathbf{r}})^{+}\mathbf{P}$
32 min, 2 cores



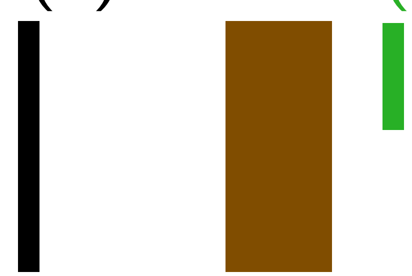


high-fidelity
5 hours, 48 cores

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

… however, this is not guaranteed

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



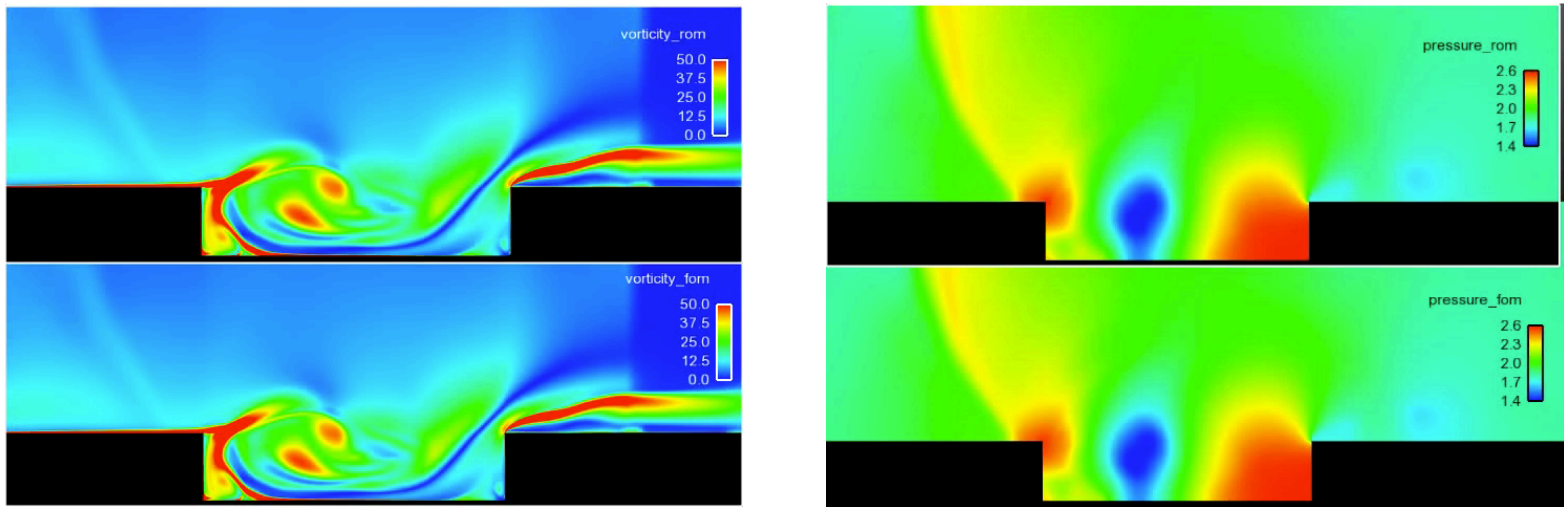
1) ***Linear-subspace assumption is strong***
2) ***Accuracy limited by information in*** $\boldsymbol{\Phi}$

Model reduction can work well…

vorticity field *pressure field*

LSPG ROM with
$\mathbf{A} = (\mathbf{P}\boldsymbol{\Phi}_r)^+\mathbf{P}$
32 min, 2 cores

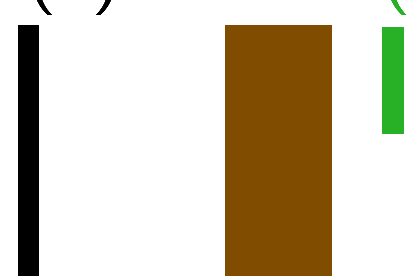
high-fidelity
5 hours, 48 cores



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

… however, this is not guaranteed

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



1) ***Linear-subspace assumption is strong***
2) ***Accuracy limited by information in $\boldsymbol{\Phi}$***

Kolmogorov-width limitation of linear subspaces

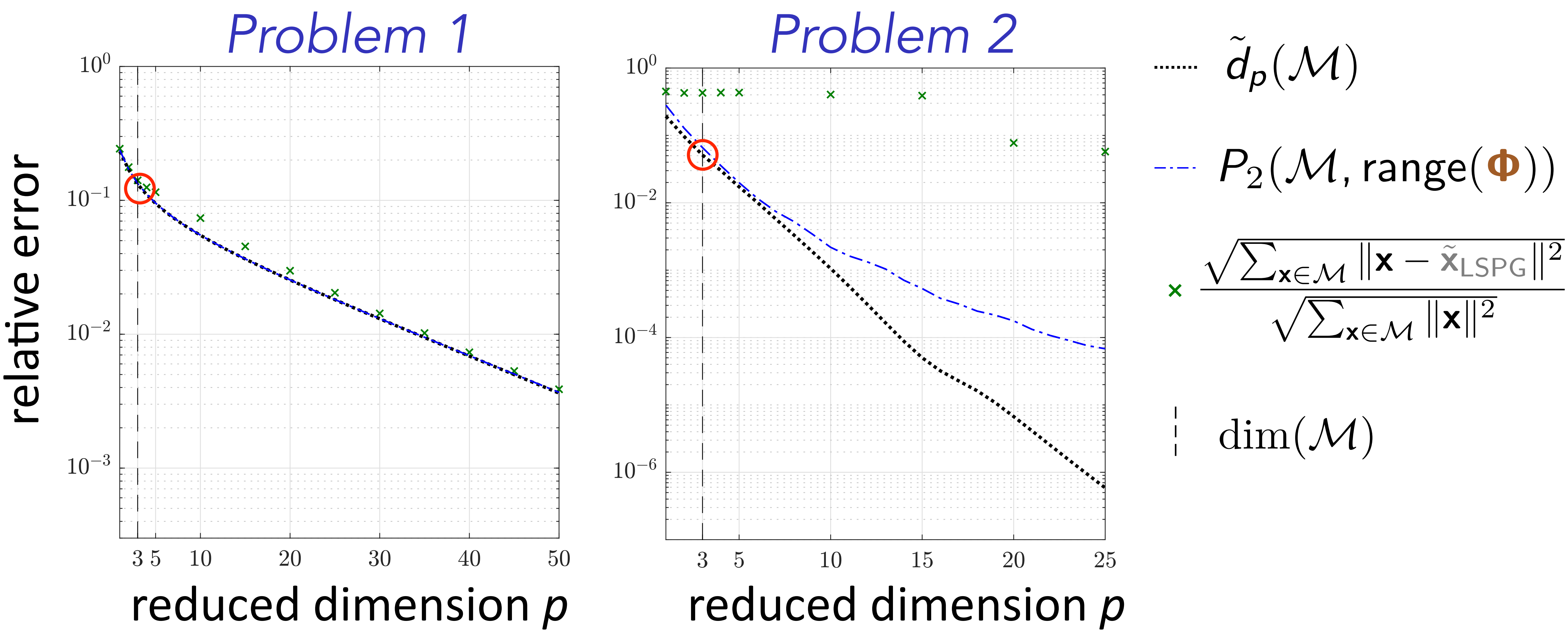
$$d_p(\mathcal{M}) := \inf_{\mathcal{S}_p} P_\infty(\mathcal{M}, \mathcal{S}_p) \qquad P_\infty(\mathcal{M}, \mathcal{S}_p) := \sup_{\mathbf{x} \in \mathcal{M}} \inf_{\mathbf{y} \in \mathcal{S}_p} \|\mathbf{x} - \mathbf{y}\|$$

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

Kolmogorov-width limitation of linear subspaces

$$\tilde{d}_p(\mathcal{M}) := \inf_{\mathcal{S}_p} P_2(\mathcal{M}, \mathcal{S}_p) \qquad P_2(\mathcal{M}, \mathcal{S}_p) := \sqrt{\sum_{\mathbf{x}\in\mathcal{M}} \inf_{\mathbf{y}\in\mathcal{S}_p} \|\mathbf{x}-\mathbf{y}\|^2} \Big/ \sqrt{\sum_{\mathbf{x}\in\mathcal{M}} \|\mathbf{x}\|^2}$$

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



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

Overcoming Kolmogorov-width limitation

Manually transform the linear subspace [Ohlberger and Rave, 2013; Iollo and Lombardi, 2014; Cagniart et al., 2019; Reiss et al., 2018; Welper, 2017; Mojgani and Balajewicz, 2017; Gerbeau and Lombardi, 2014; Nair and Balajewicz, 2019]

+ Works well on specialized problems
- Requires problem-specific knowledge
- Does not consider manifolds of general nonlinear structure

Local linear subspaces
[Dihlmann et al., 2011; Drohmann et al., 2011; Taddei et al., 2015; Amsallem et al., 2012; Peherstorfer and Willcox, 2015]

+ Tailored bases for regions of time/physical domain or state space
- Does 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 nonlinear manifolds with general nonlinear structure
+ Kinematically consistent
+ Satisfy optimality property

Practical nonlinear-manifold construction

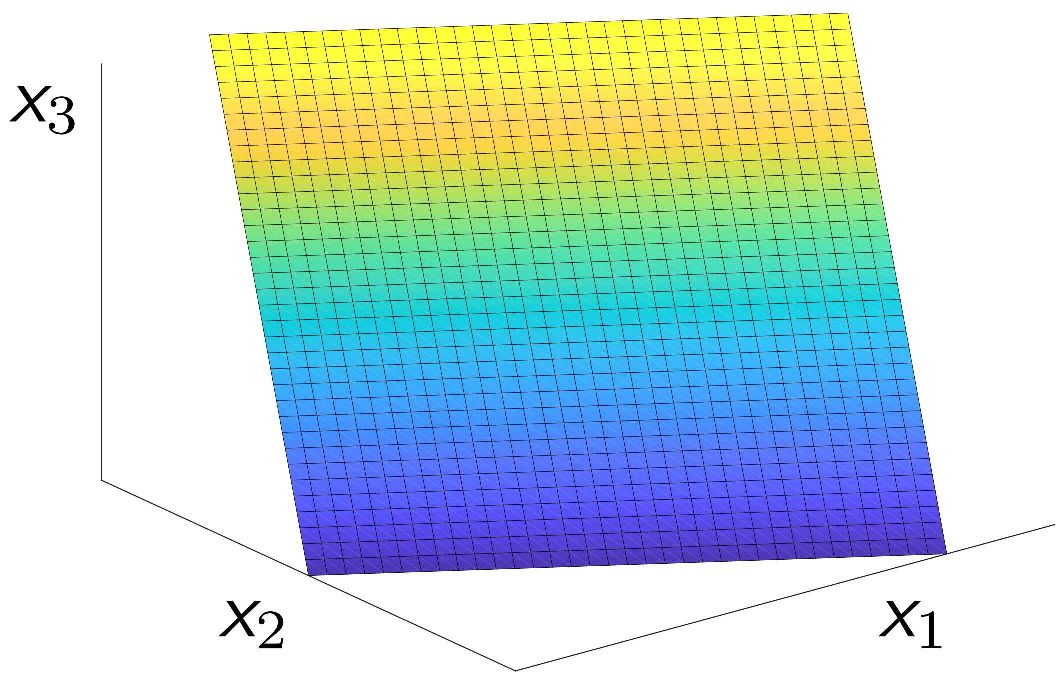
+ No problem-specific knowledge required
+ Use same snapshot data as typical linear-subspace approaches

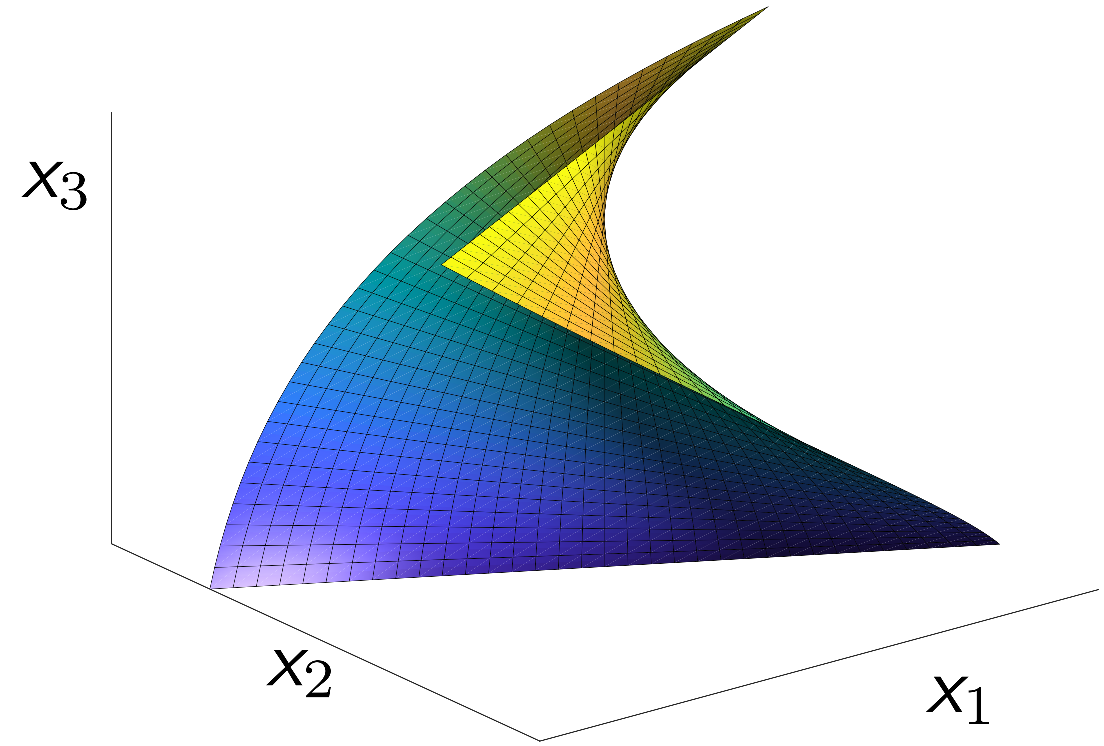
Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders [Lee and C., 2018]

Nonlinear trial manifold

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

example
N=3
p=2

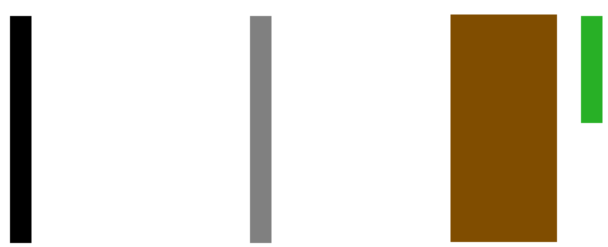




state

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

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



+ manifold has general structure

velocity

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

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

+ kinematically consistent

Manifold Galerkin and LSPG projection

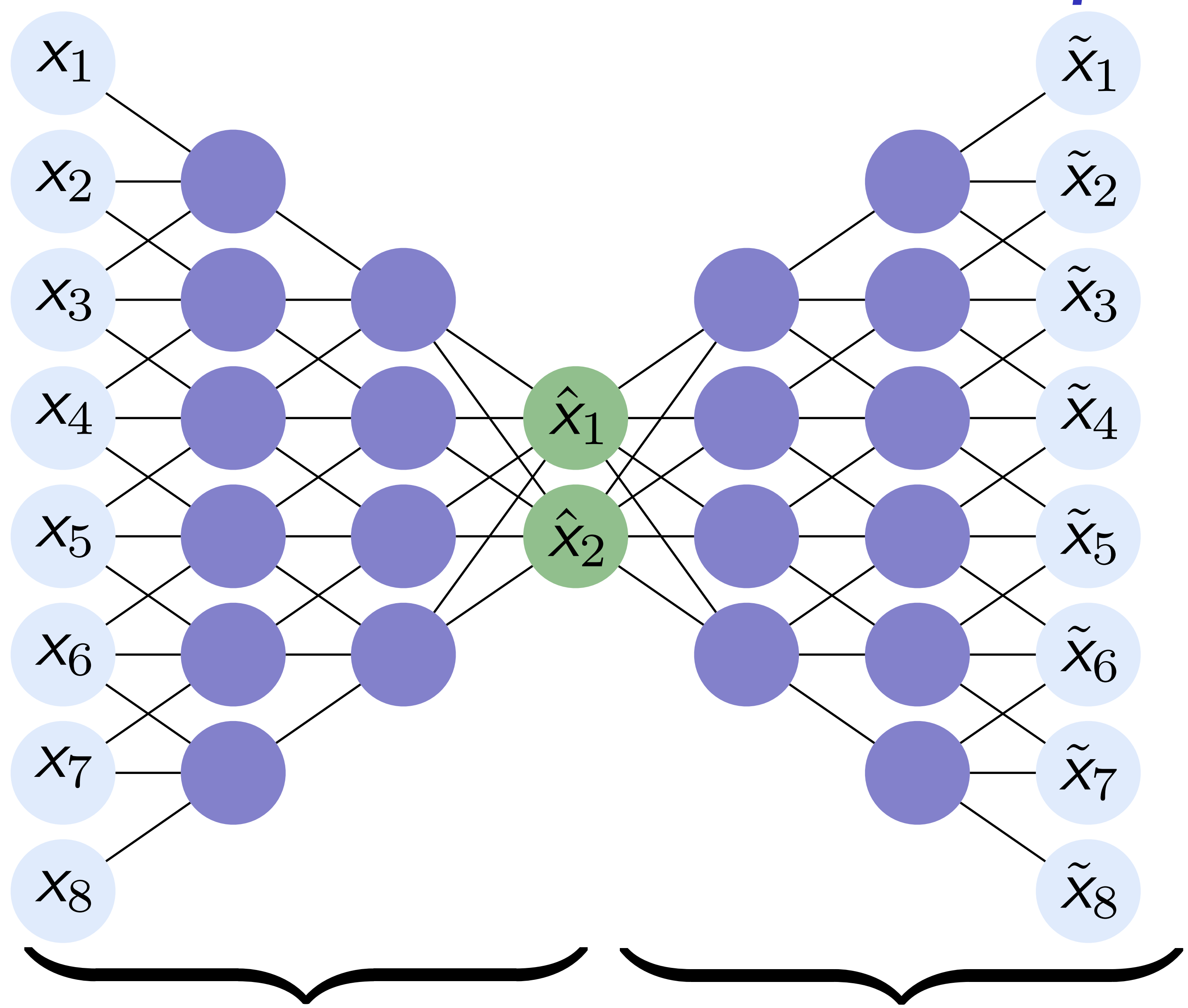
	Linear-subspace ROM	***Nonlinear-manifold ROM***
Galerkin	$\frac{d\hat{\mathbf{x}}}{dt} = \underset{\hat{\mathbf{v}} \in \mathbb{R}^n}{\operatorname{argmin}} \|\mathbf{r}(\boldsymbol{\Phi}\hat{\mathbf{v}}, \boldsymbol{\Phi}\hat{\mathbf{x}}; t)\|_2$	$\frac{d\hat{\mathbf{x}}}{dt} = \underset{\hat{\mathbf{v}}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

\+ Satisfy optimality properties

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

Deep autoencoders

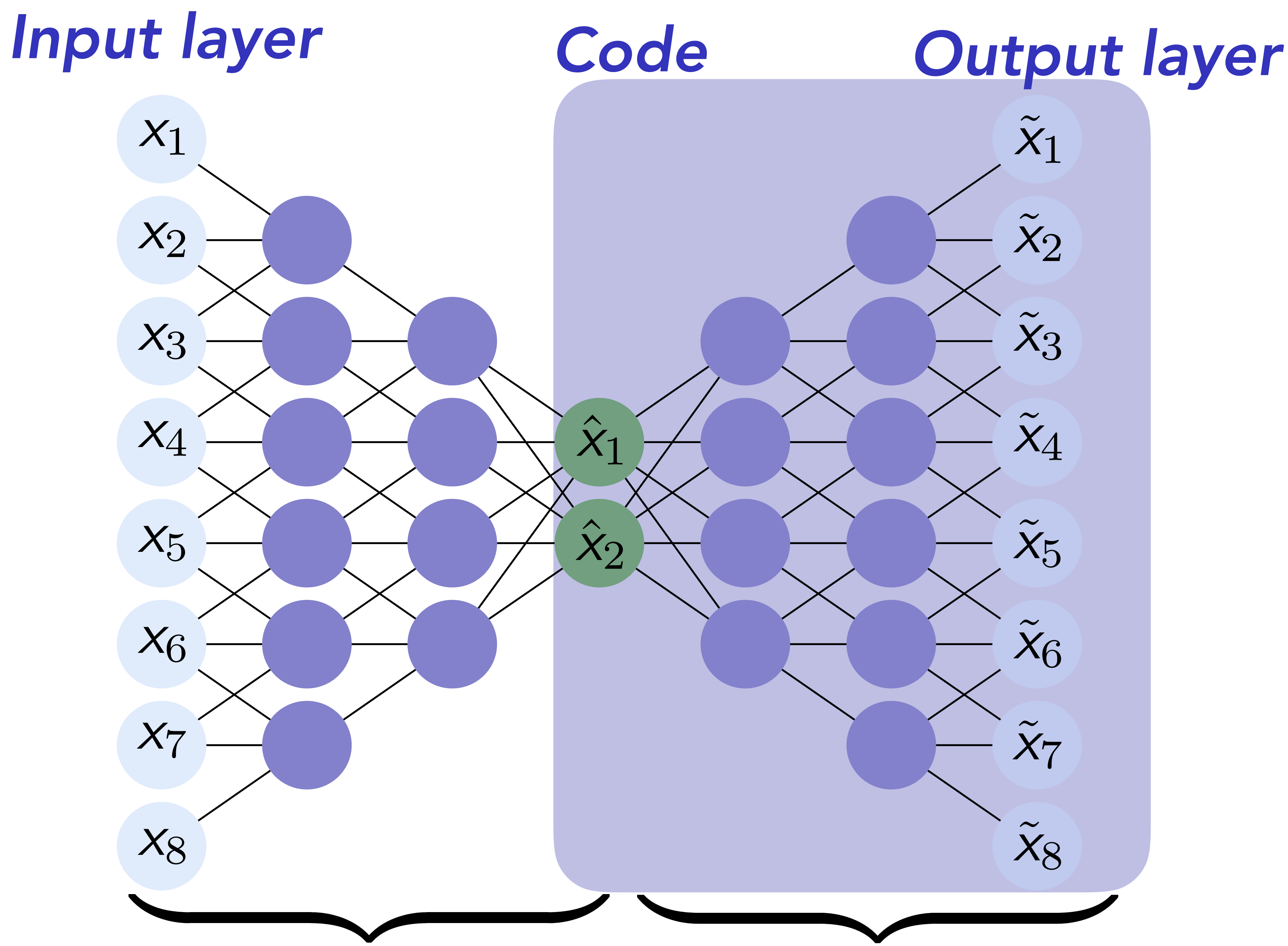
Input layer ***Code*** ***Output layer***



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

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

Deep autoencoders



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

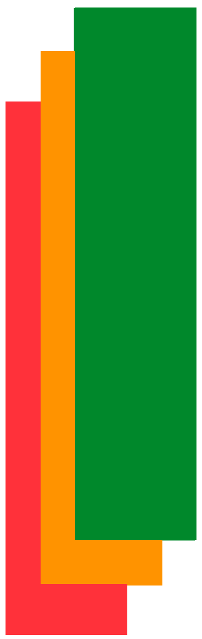
$$\tilde{\mathbf{x}} = \mathbf{h}_{\text{dec}}(\cdot;\boldsymbol{\theta}_{\text{dec}}) \circ \mathbf{h}_{\text{enc}}(\mathbf{x};\boldsymbol{\theta}_{\text{enc}})$$

+ If $\tilde{\mathbf{x}} \approx \mathbf{x}$ for parameters $\boldsymbol{\theta}^{\star}_{\text{dec}}$, $\mathbf{g} = \mathbf{h}_{\text{dec}}(\cdot;\boldsymbol{\theta}^{\star}_{\text{dec}})$ produces an accurate manifold

Algorithm

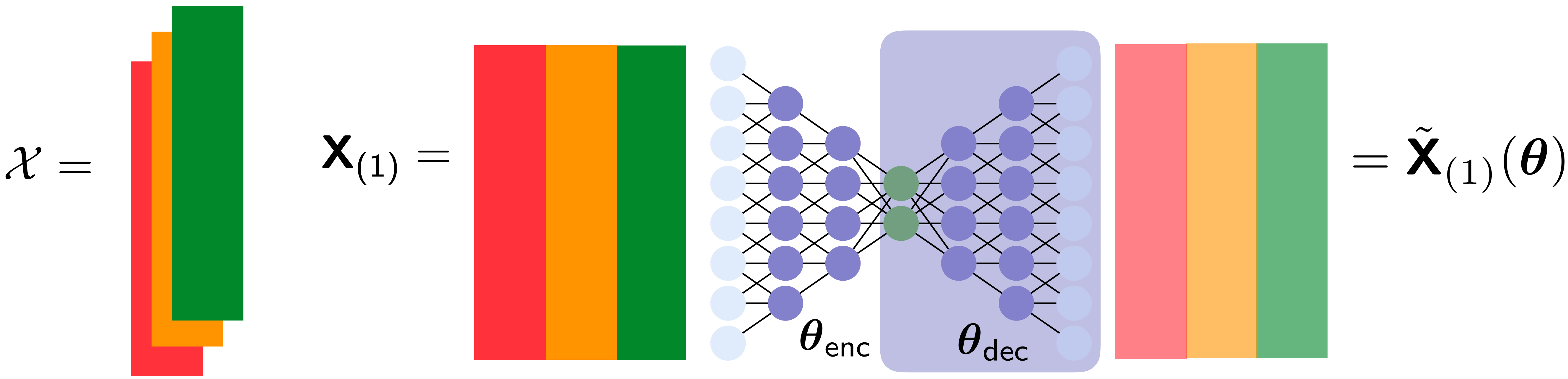
1. *Training:* Solve ODE for $\boldsymbol{\mu} \in \mathcal{D}_{\text{training}}$ and collect simulation data
2. *Machine learning:* Train deep convolutional autoencoder
3. *Reduction:* Solve manifold Galerkin or LSPG for $\boldsymbol{\mu} \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$

$\mathcal{X} =$



Algorithm

1. *Training:* Solve ODE for $\mu \in \mathcal{D}_{\text{training}}$ and collect simulation data
2. *Machine learning:* Train deep convolutional autoencoder
3. *Reduction:* Solve manifold Galerkin or LSPG for $\mu \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$



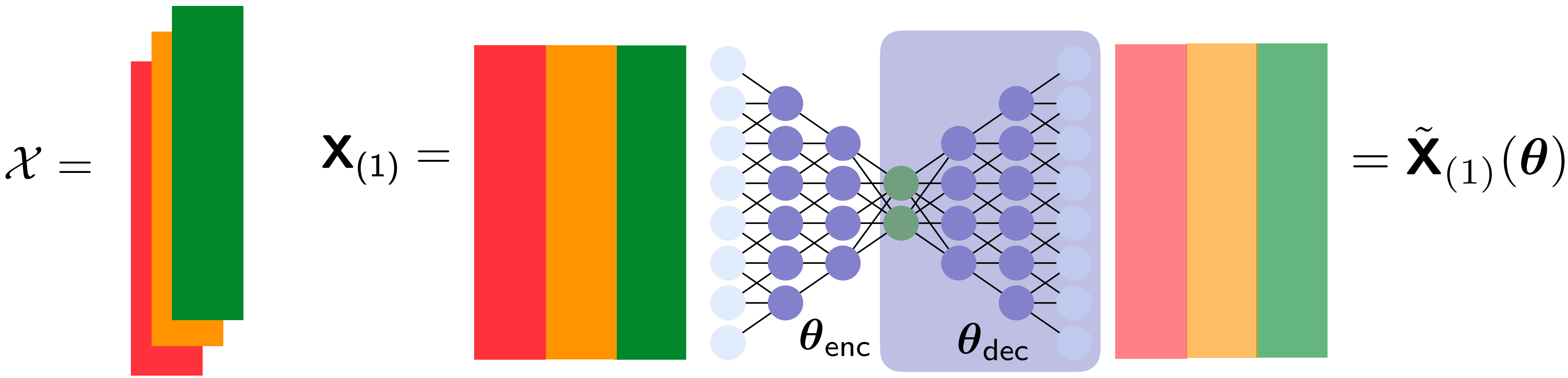
- Compute $\boldsymbol{\theta}^\star$ by approximately solving $\underset{\boldsymbol{\theta}}{\text{minimize}} \|\mathbf{X}_{(1)} - \tilde{\mathbf{X}}_{(1)}(\boldsymbol{\theta})\|_F$
- Define nonlinear trial manifold by setting $\mathbf{g} = \mathbf{h}_{\text{dec}}(\cdot; \boldsymbol{\theta}^\star_{\text{dec}})$

\+ No problem-specific knowledge required

\+ Same snapshot data

Algorithm

1. *Training:* Solve ODE for $\boldsymbol{\mu} \in \mathcal{D}_{\text{training}}$ and collect simulation data
2. *Machine learning:* Train deep convolutional autoencoder
3. *Reduction:* Solve manifold Galerkin or LSPG for $\boldsymbol{\mu} \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$



- Compute $\boldsymbol{\theta}^\star$ by approximately solving $\underset{\boldsymbol{\theta}}{\text{minimize}} \|\mathbf{X}_{(1)} - \tilde{\mathbf{X}}_{(1)}(\boldsymbol{\theta})\|_F$
- Define nonlinear trial manifold by setting $\mathbf{g} = \mathbf{h}_{\text{dec}}(\cdot; \boldsymbol{\theta}^\star_{\text{dec}})$

\+ No problem-specific knowledge required

\+ Same snapshot data

Numerical results

1D Burgers' equation

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

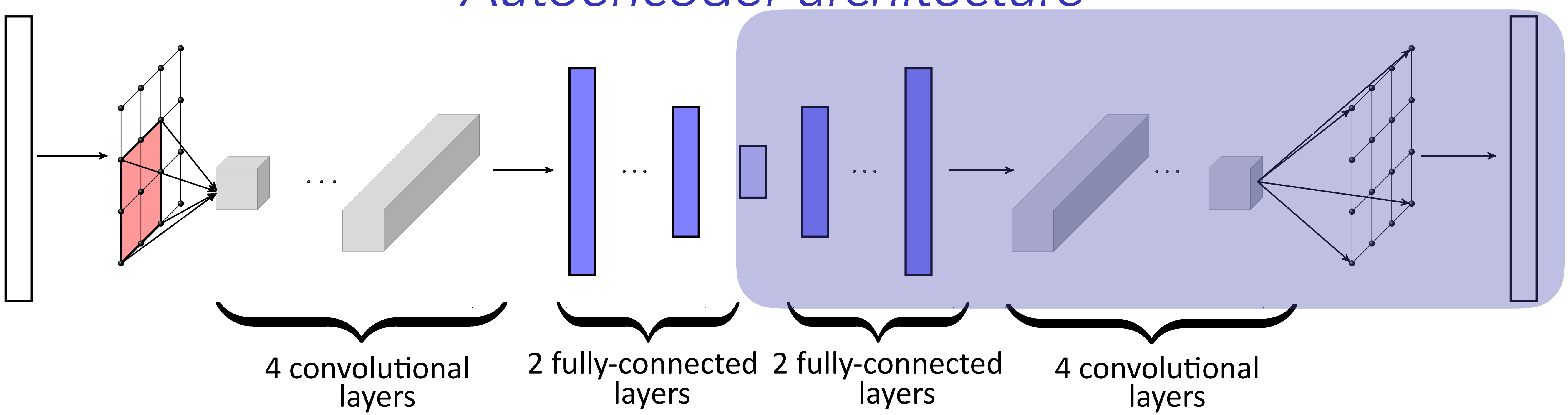
- $\boldsymbol{\mu}$: α, inlet boundary condition
- *Spatial discretization*: finite volume
- *Time integrator:* backward Euler

2D Chemically 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

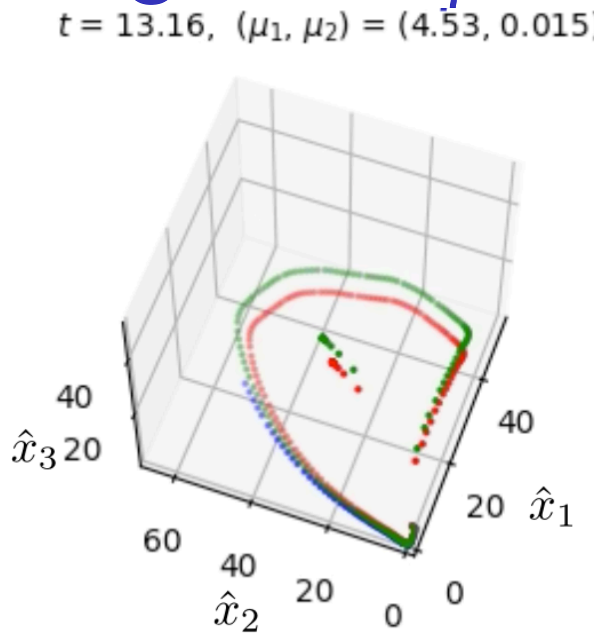


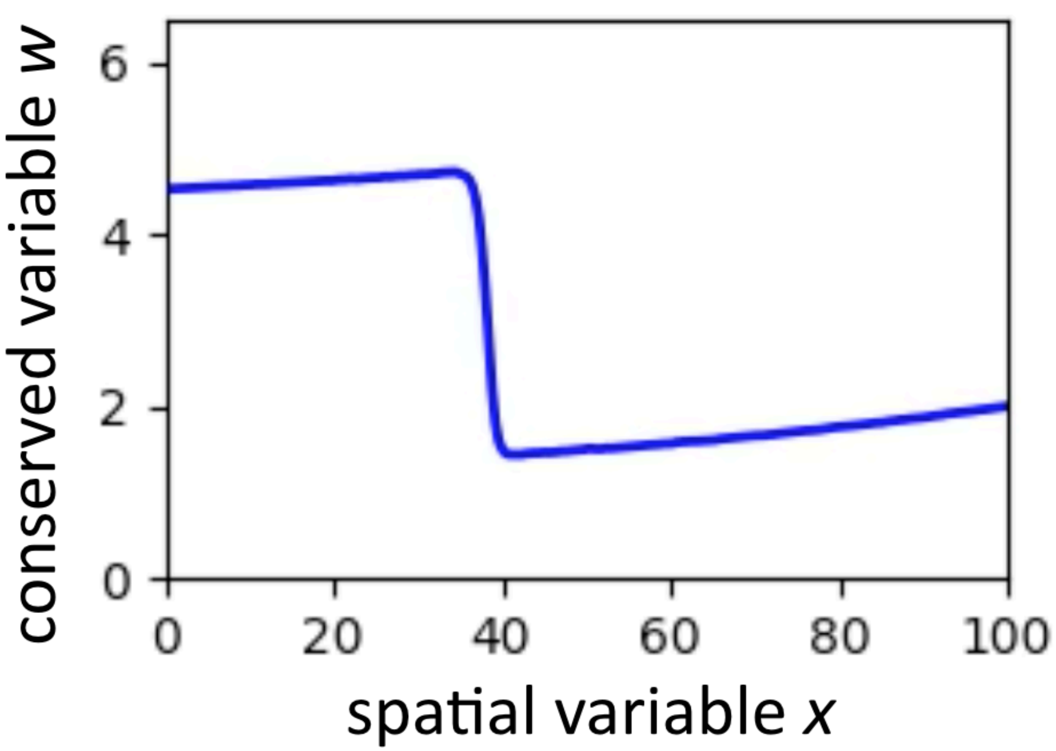
Results: nonlinear manifold interpretation

reduced state $\hat{\mathbf{x}}$

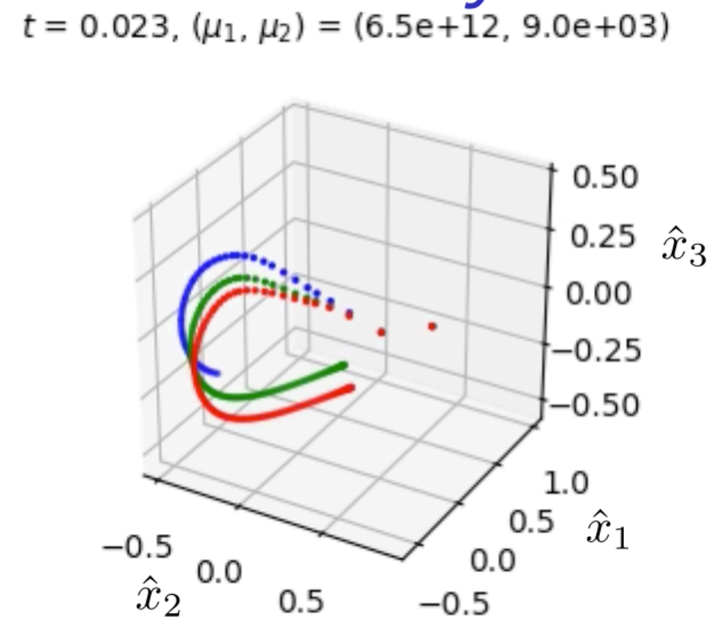
decoding $\mathbf{g}(\hat{\mathbf{x}})$

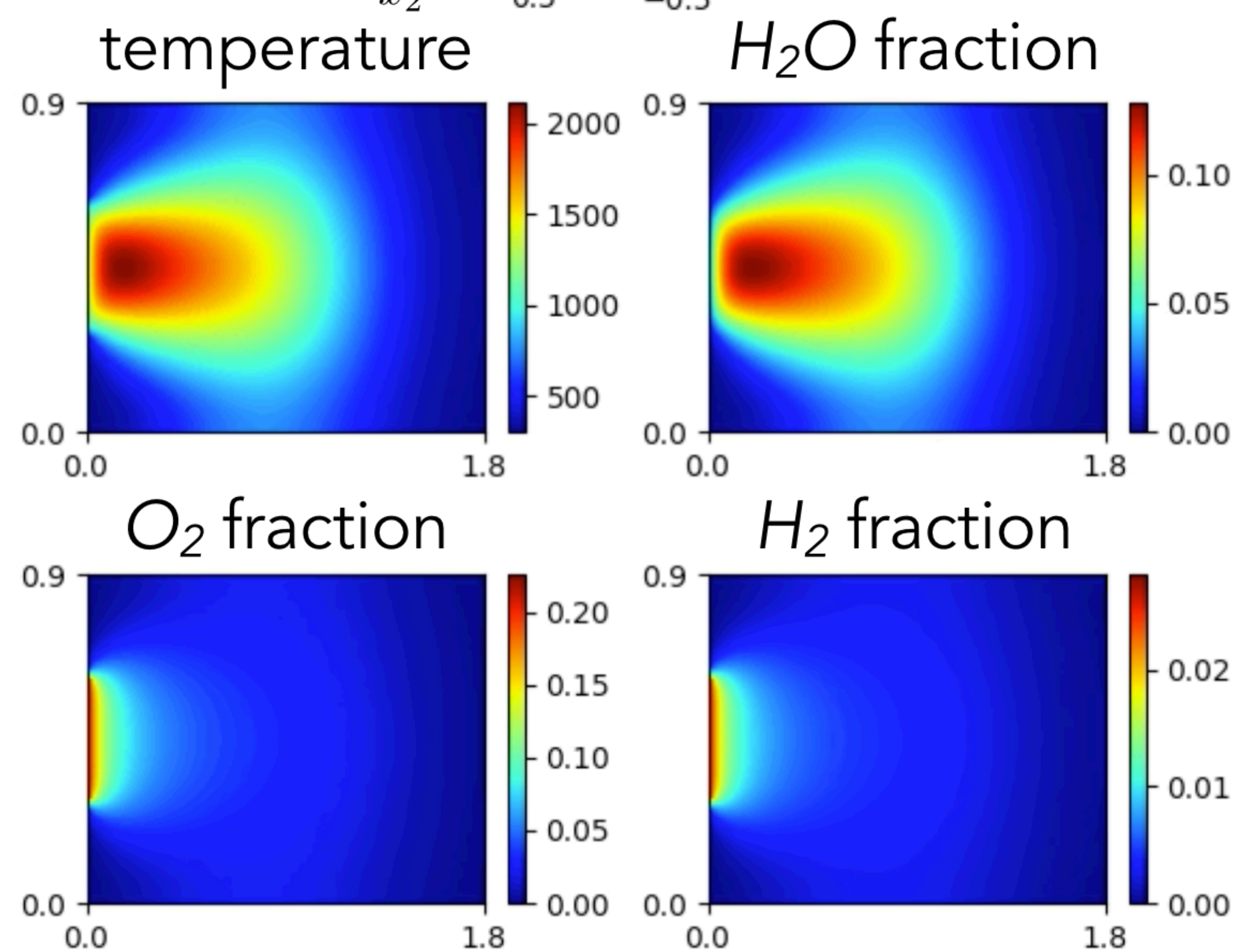
1D Burgers' equation





2D Chemically reacting flow





Manifold LSPG outperforms optimal linear subspace

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

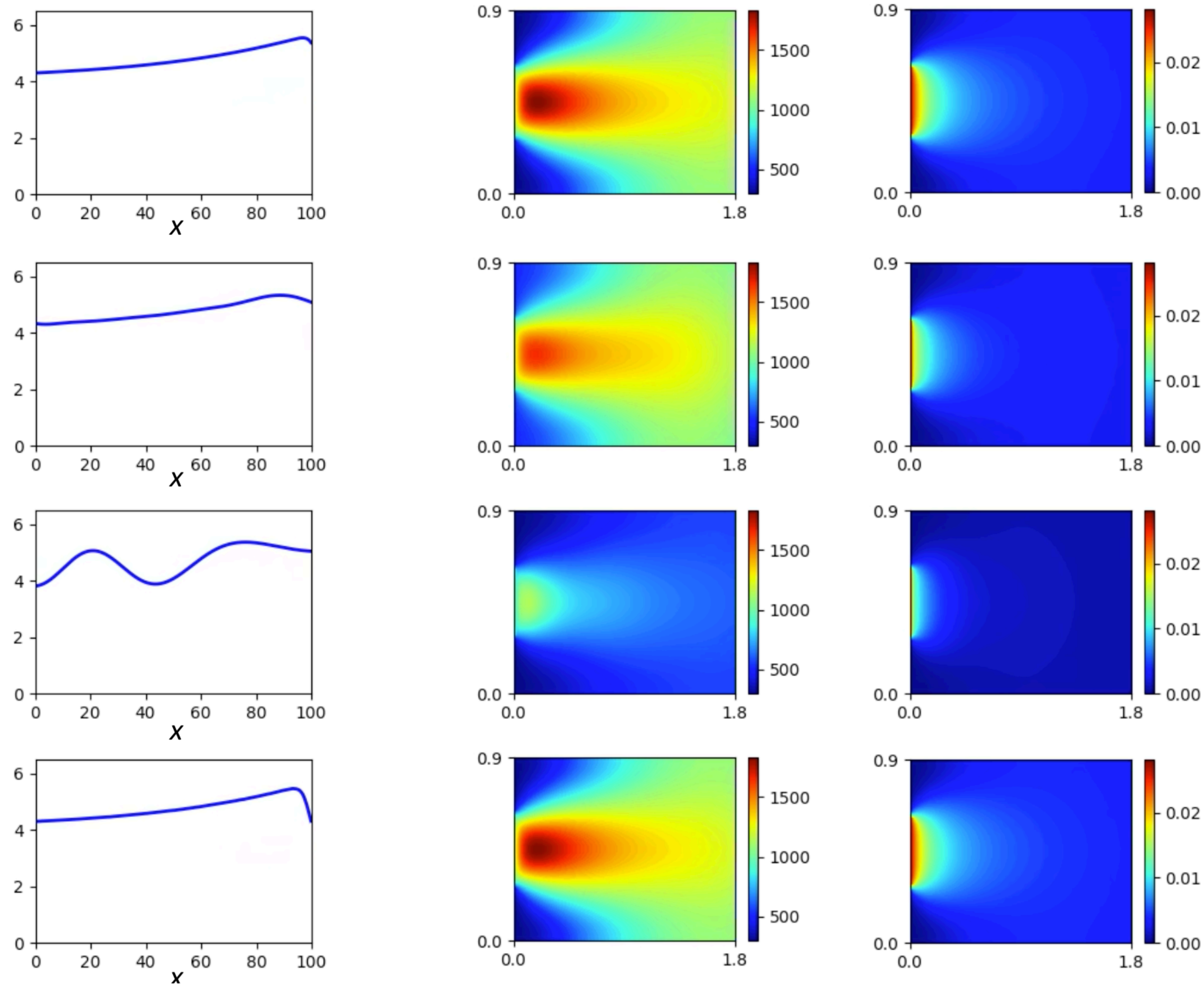
conserved variable | temperature | H_2 fraction

high-fidelity model

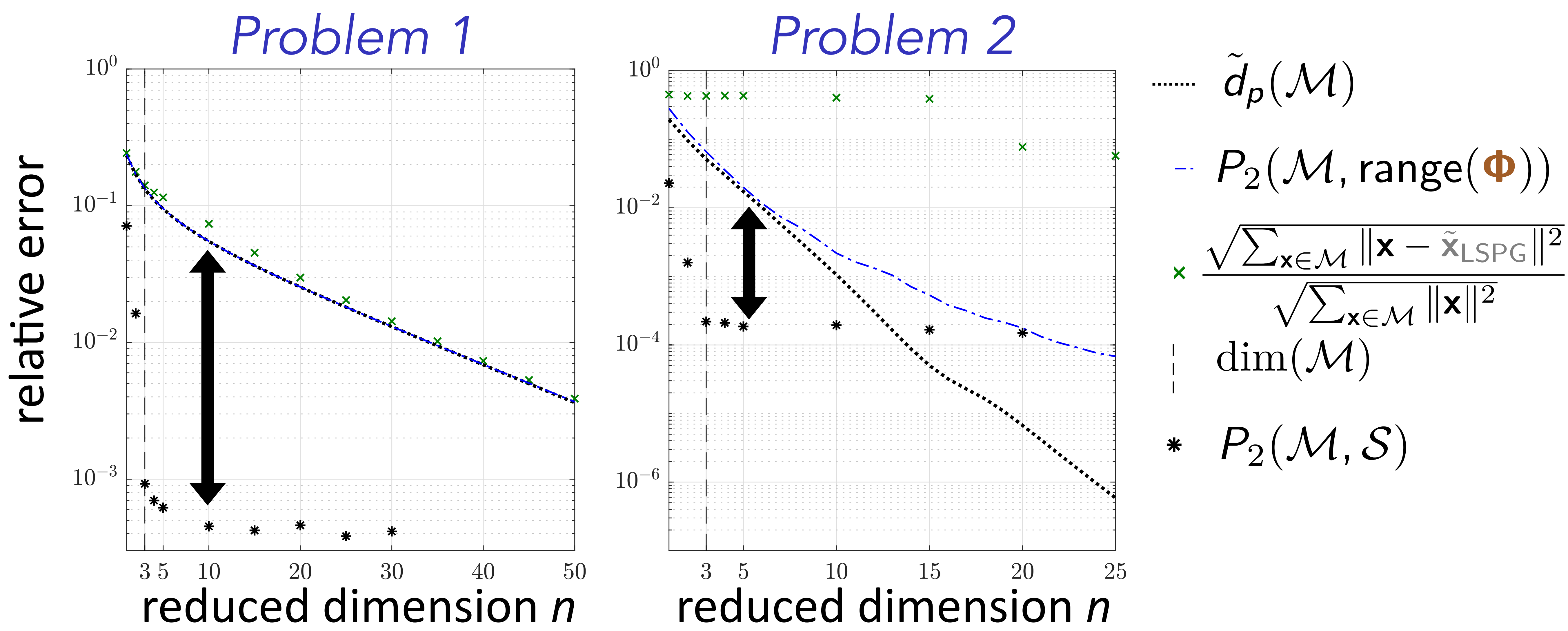
projection onto optimal linear subspace p=5

POD-LSPG p=5

Manifold LSPG p=5

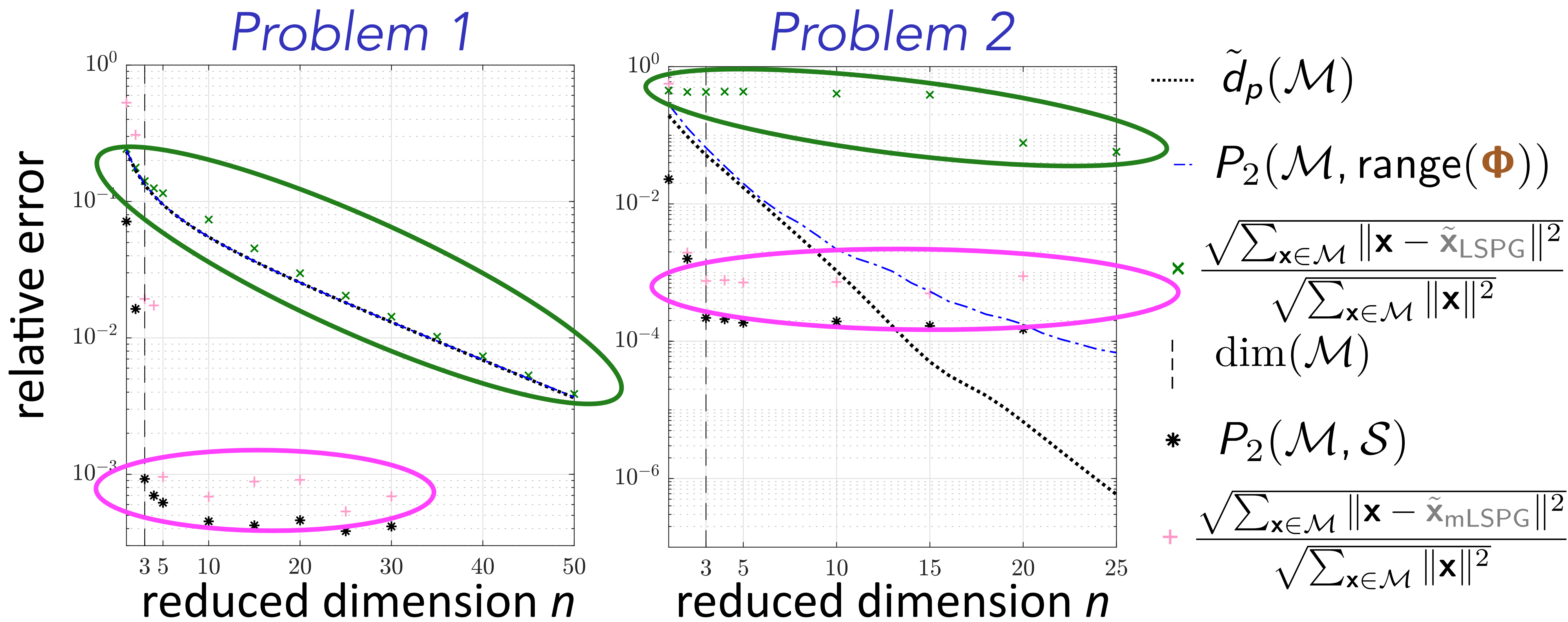


Method overcomes Kolmogorov-width limitation



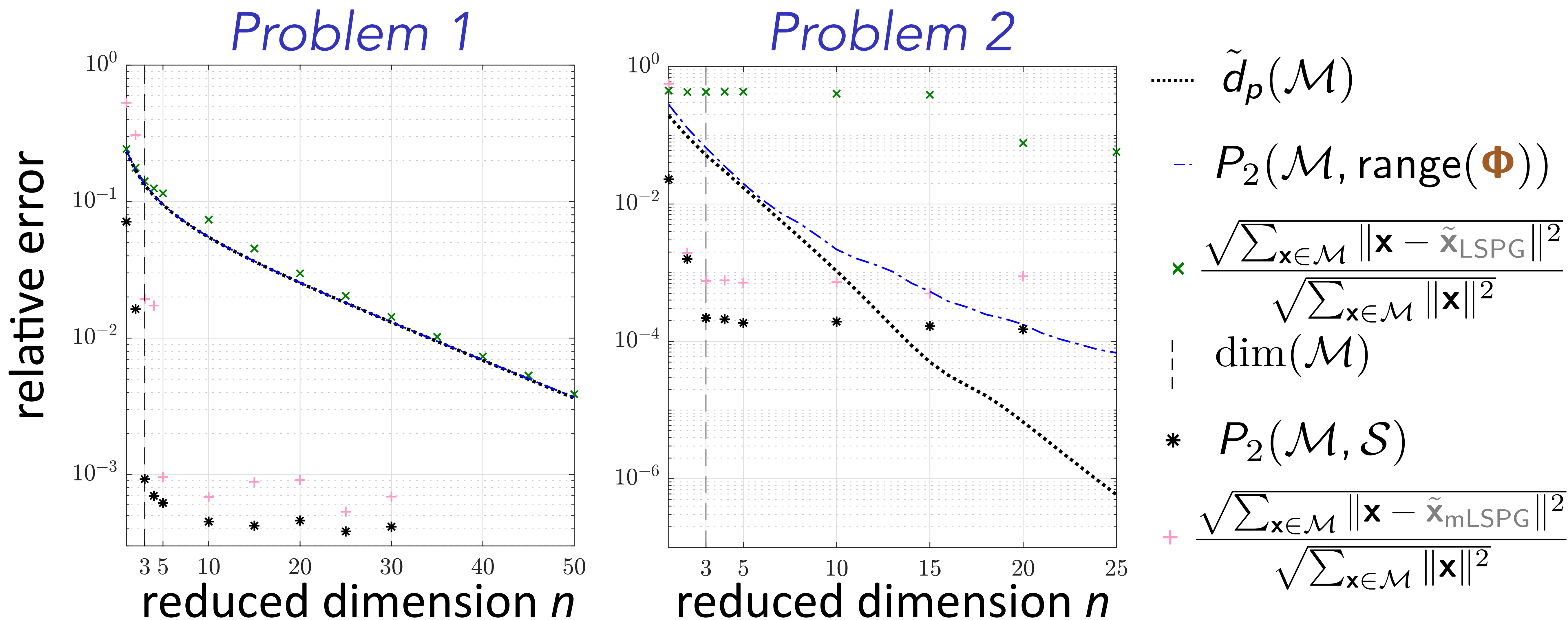
+ Autoencoder manifold significantly better than optimal linear subspace

Method overcomes Kolmogorov-width limitation



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

Method overcomes Kolmogorov-width limitation



- Autoencoder manifold significantly better than optimal linear subspace
- Manifold LSPG orders-of-magnitude more accurate than subspace LSPG
- Method overcomes Kolmogorov-width limitation

Our research

Accurate, low-cost, structure-preserving, reliable, certified nonlinear model reduction

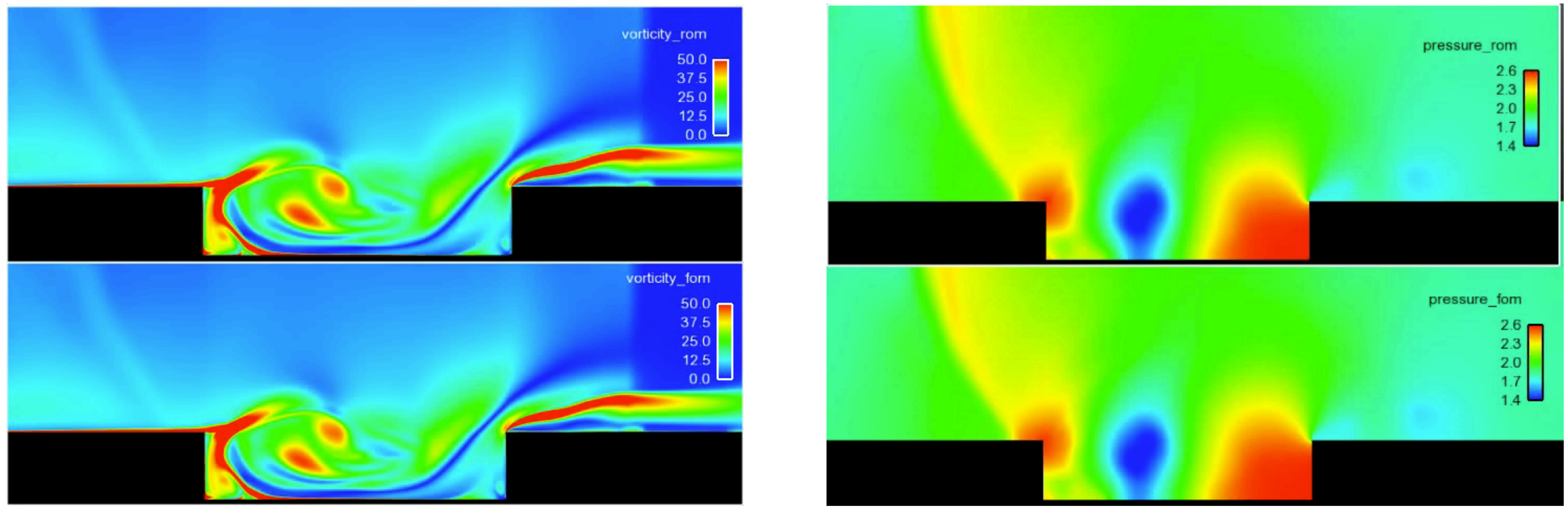
- *accuracy*: LSPG projection [C., Bou-Mosleh, Farhat, 2011; C., Barone, Antil, 2017]
- *low cost*: sample mesh [C., Farhat, Cortial, Amsallem, 2013]
- *low cost*: reduce temporal complexity [C., Ray, van Bloemen Waanders, 2015; C., Brencher, Haasdonk, Barth, 2017; Choi and C., 2019]
- *structure preservation* [C., Tuminaro, Boggs, 2015; Peng and C., 2017; C., Choi, Sargsyan, 2017]
- *robustness*: projection onto nonlinear manifolds [Lee, C., 2018]
- *robustness*: *h*-adaptivity [C., 2015]
- *certification*: machine learning error models [Drohmann and C., 2015; Trehan, C., Durlofsky, 2017; Freno and C., 2019; Pagani, Manzoni, C., 2019]

Model reduction can work well…

vorticity field *pressure field*

LSPG ROM with
$\mathbf{A} = (\mathbf{P}\mathbf{\Phi}_\mathbf{r})^+\mathbf{P}$
32 min, 2 cores

high-fidelity
5 hours, 48 cores

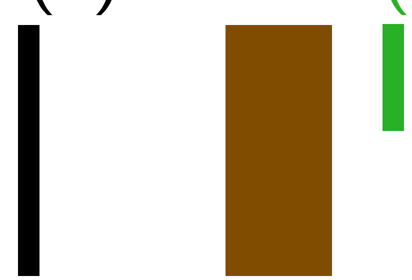


\+ *229x savings in core–hours*

\+ *< 1% error in time-averaged drag*

… however, this is not guaranteed

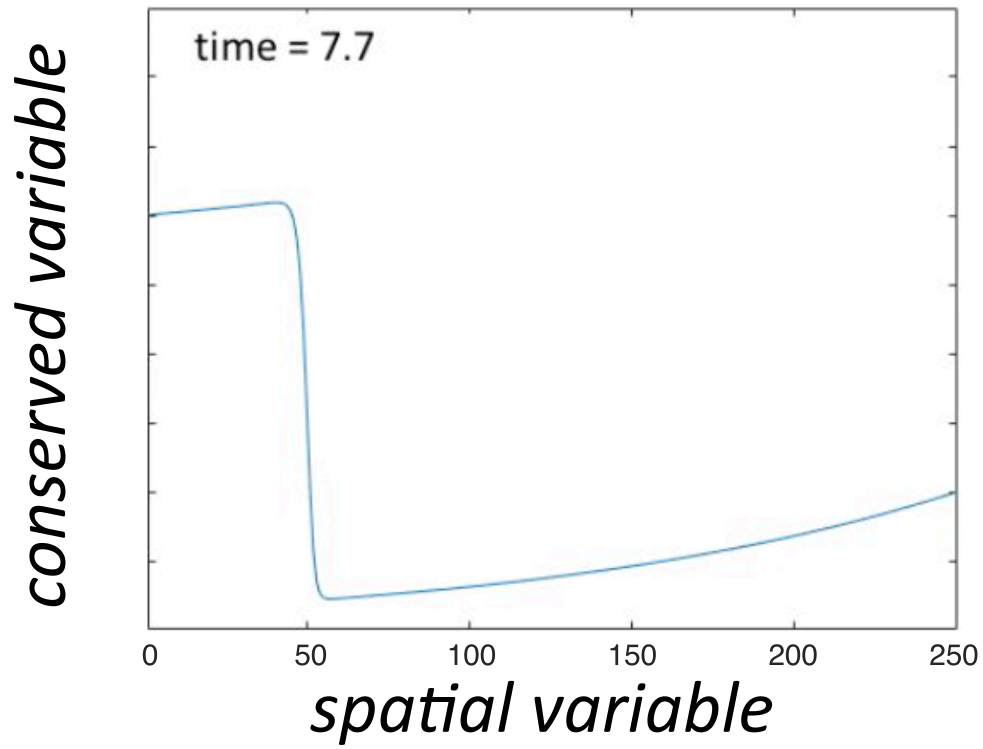
$$\mathbf{x}(t) \approx \mathbf{\Phi}\,\hat{\mathbf{x}}(t)$$



1) ***Linear-subspace assumption is strong***
2) ***Accuracy limited by information in $\mathbf{\Phi}$***

Illustration: inviscid 1D Burgers' equation

high-fidelity model



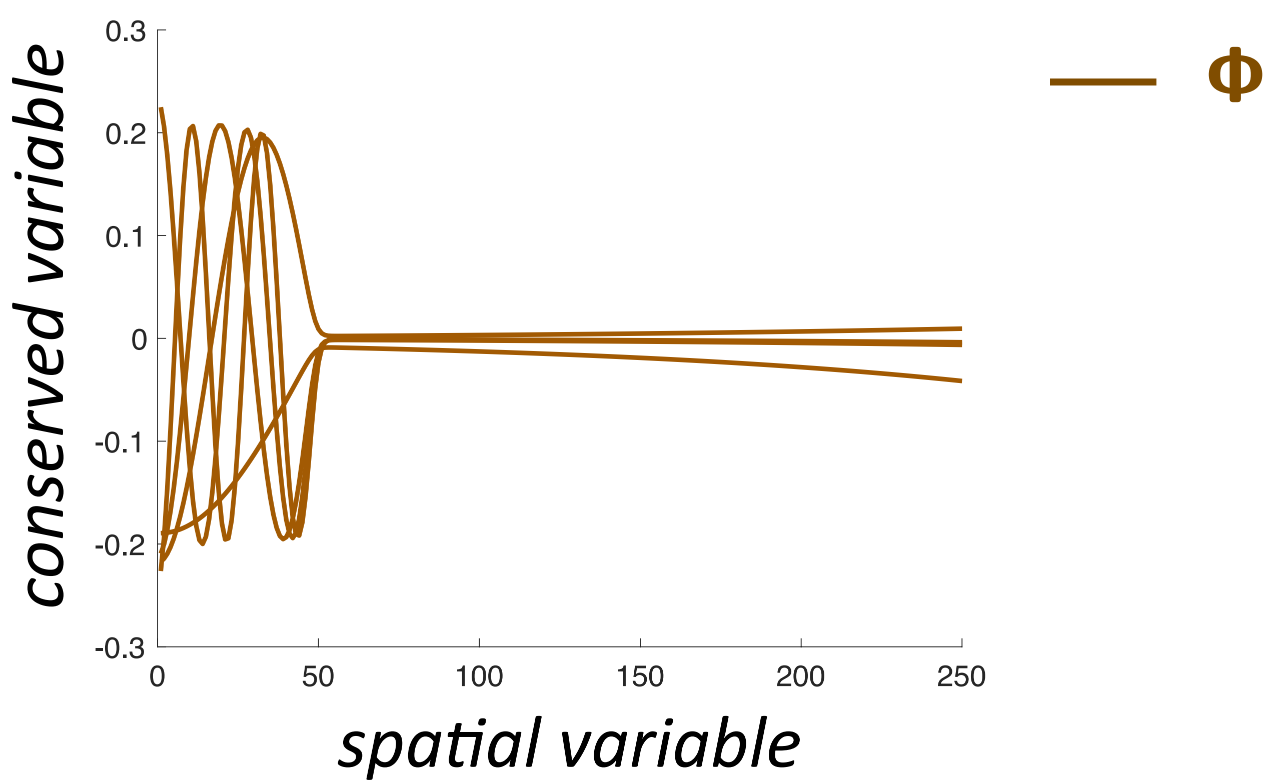
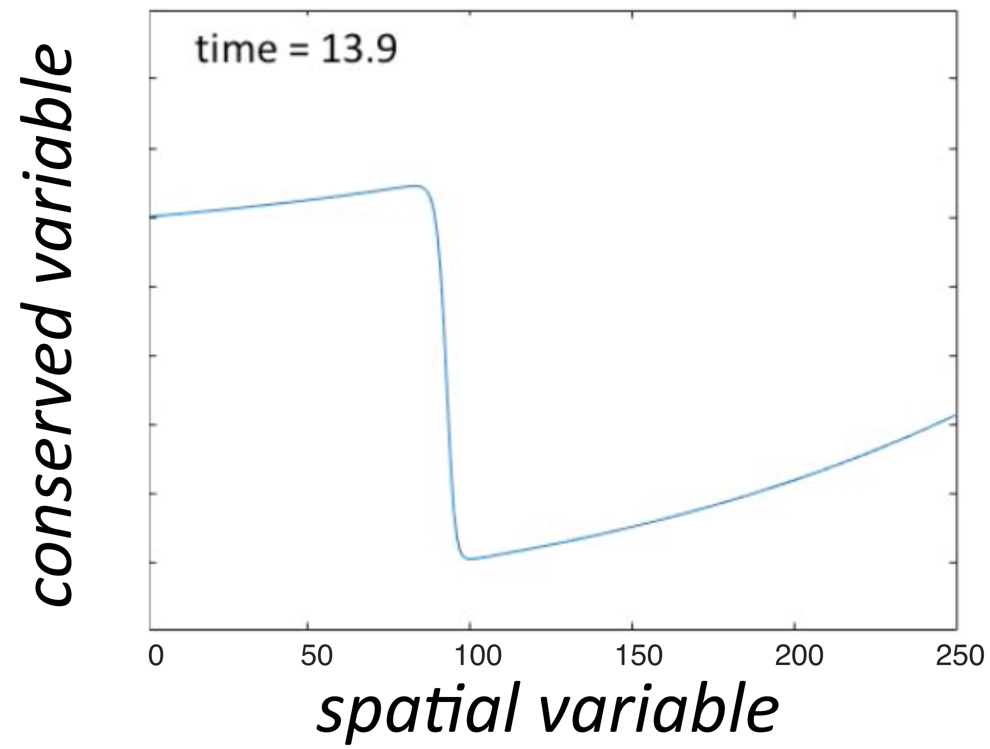
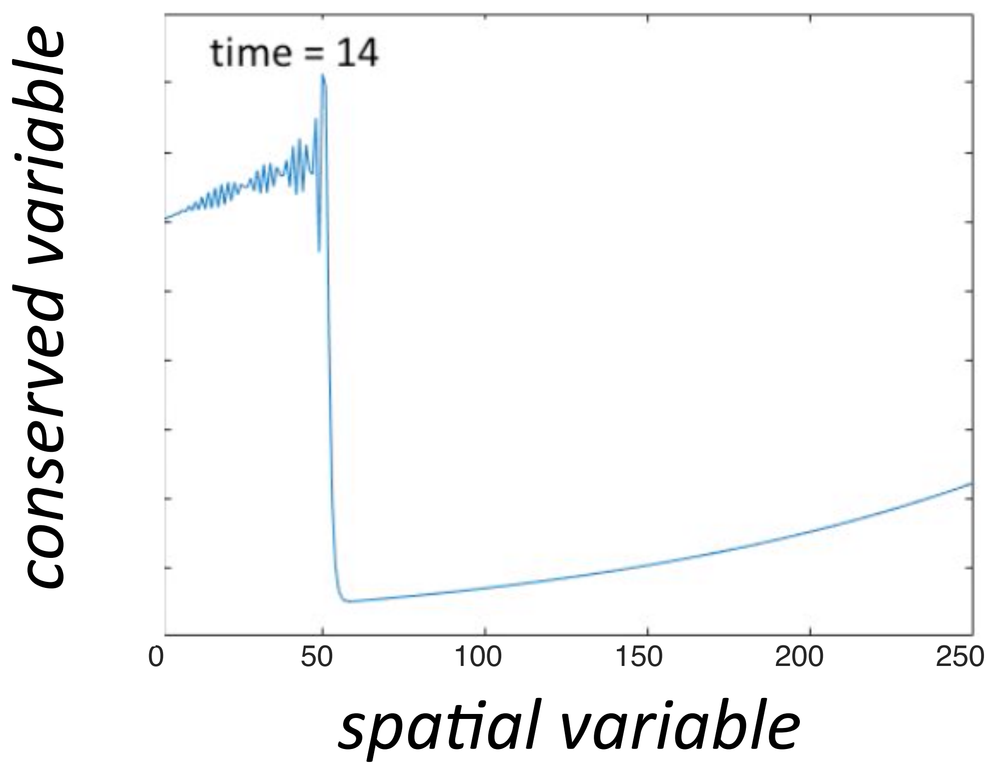


Illustration: inviscid 1D Burgers' equation

high-fidelity model



reduced-order model

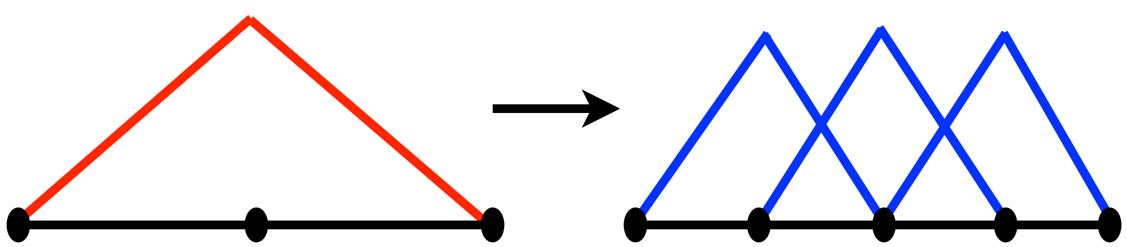


reduced-order model inaccurate when Φ insufficient

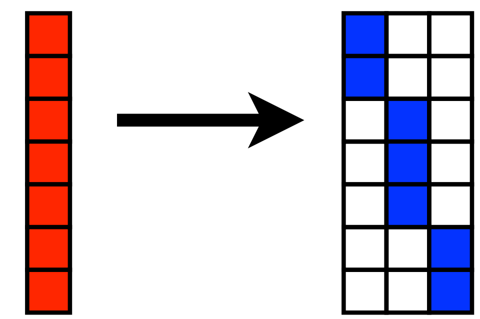
Main idea [C., 2015]

Model-reduction analogue to mesh-adaptive h-refinement

- ‘Split’ basis vectors

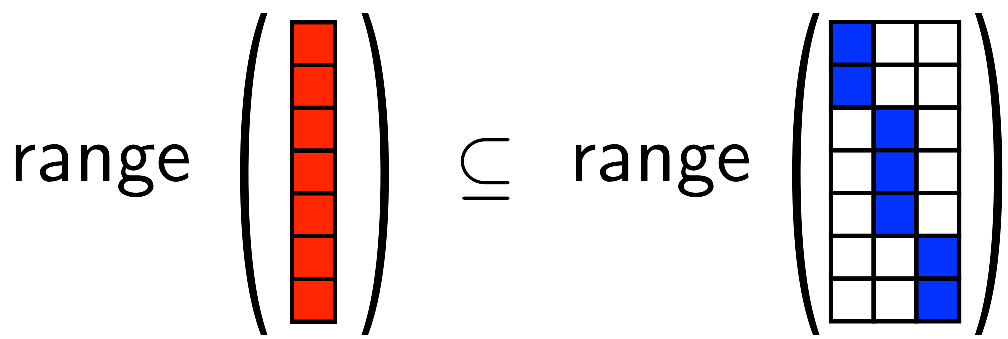


finite-element h-refinement



reduced-order-model h-refinement

- Generate hierarchical subspaces



- Converges to the high-fidelity model

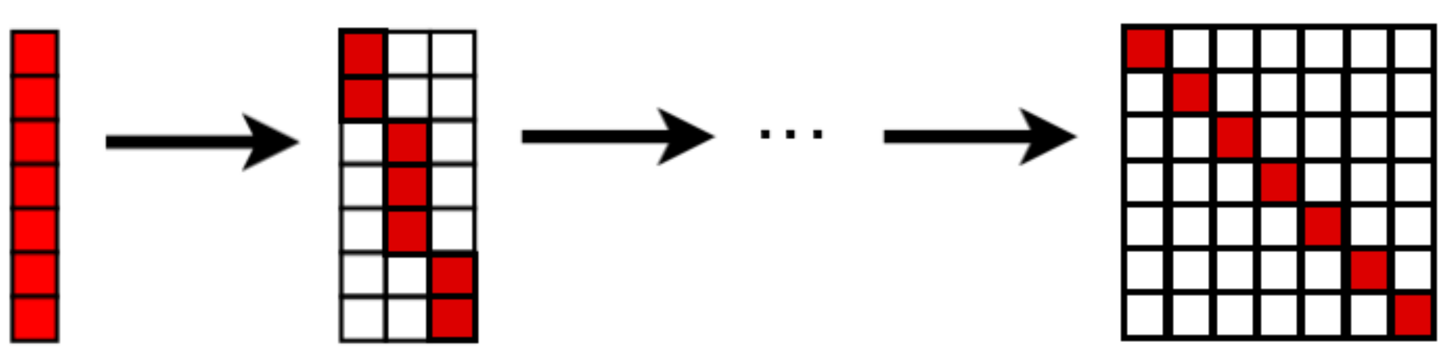
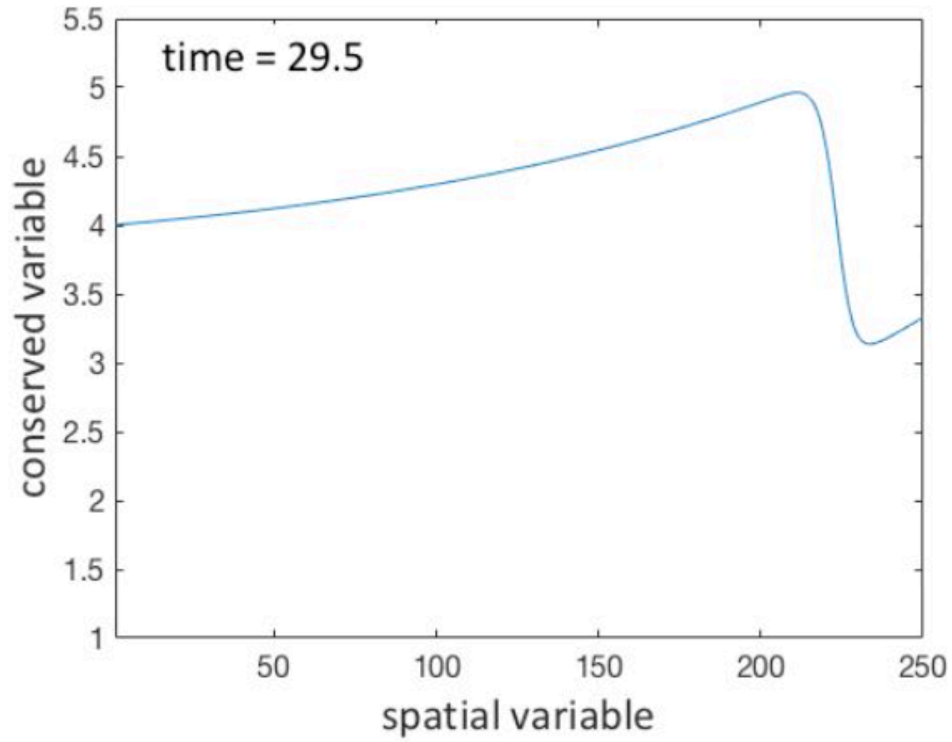
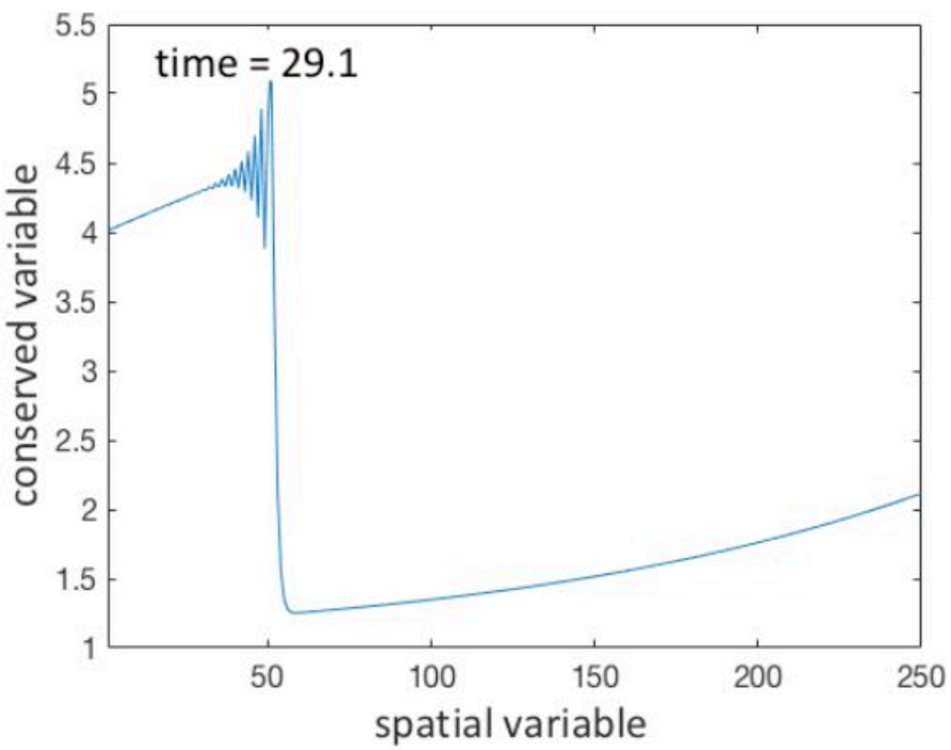


Illustration: inviscid 1D Burgers' equation

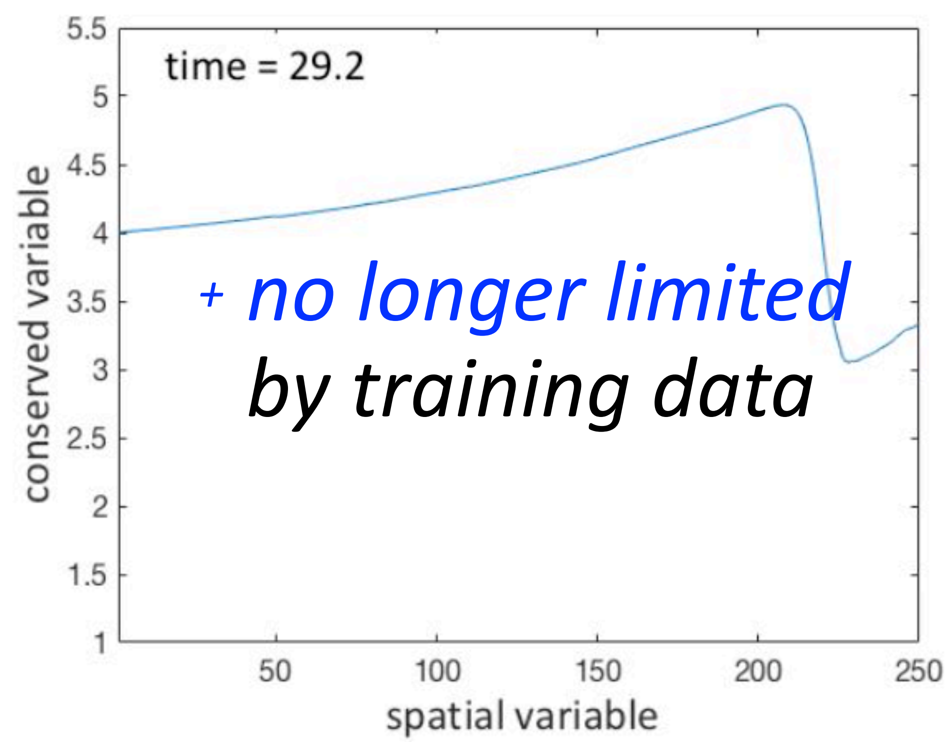
high-fidelity model



reduced-order model (dim 50)



h-adaptive ROM (mean dim 48.5)



Our research

Accurate, low-cost, structure-preserving, reliable, certified nonlinear model reduction

- *accuracy*: LSPG projection [C., Bou-Mosleh, Farhat, 2011; C., Barone, Antil, 2017]
- *low cost*: sample mesh [C., Farhat, Cortial, Amsallem, 2013]
- *low cost*: reduce temporal complexity [C., Ray, van Bloemen Waanders, 2015; C., Brencher, Haasdonk, Barth, 2017; Choi and C., 2019]
- *structure preservation* [C., Tuminaro, Boggs, 2015; Peng and C., 2017; C., Choi, Sargsyan, 2017]
- *robustness*: projection onto nonlinear manifolds [Lee, C., 2018]
- *robustness*: *h*-adaptivity [C., 2015]
- *certification*: machine learning error models [Drohmann and C., 2015; Trehan, C., Durlofsky, 2017; Freno and C., 2019; Pagani, Manzoni, C., 2019]

Brian Freno

Discrete-time error bound

Theorem [C., Barone, Antil, 2017]

If the following conditions hold:

1. $\mathbf{f}(\cdot; t)$ is Lipschitz continuous with Lipschitz constant κ
2. The time step Δt is small enough such that $0 < h := |\alpha_0| - |\beta_0|\kappa\Delta t$,
3. A backward differentiation formula (BDF) time integrator is used,
4. LSPG employs $\mathbf{A} = \mathbf{I}$, then

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

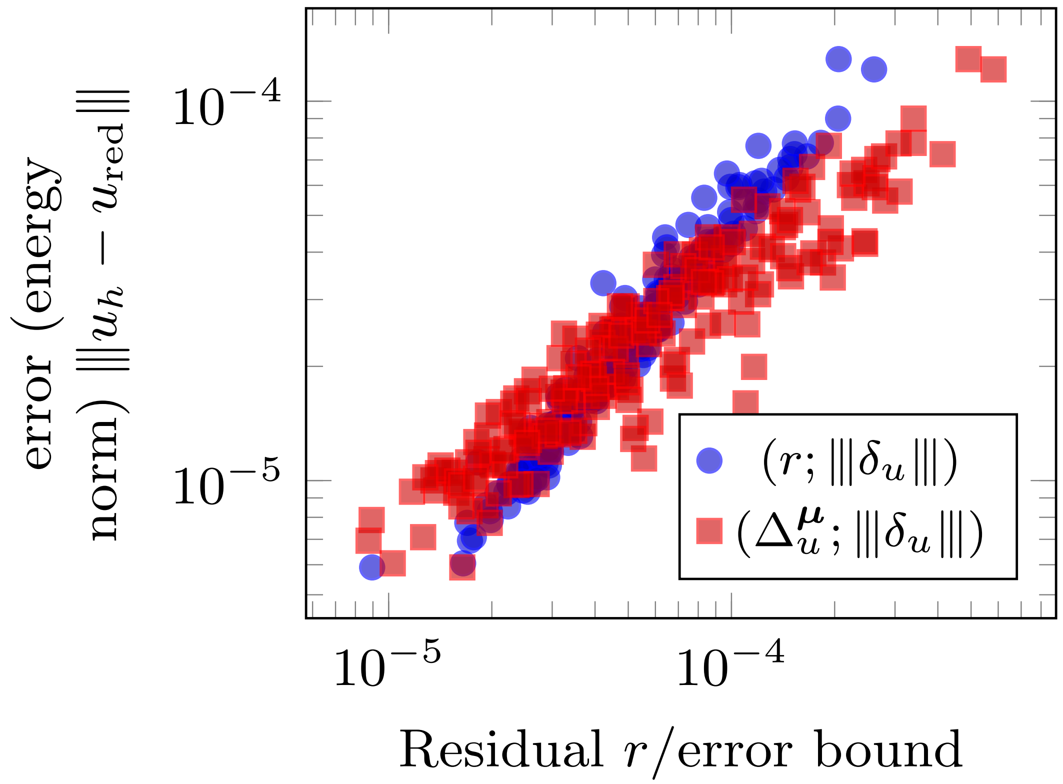
$$\|\mathbf{x}^n - \mathbf{\Phi}\hat{\mathbf{x}}_{\mathrm{LSPG}}^n\|_2 \leq \frac{\gamma_1(\gamma_2)^n \exp(\gamma_3 t^n)}{\gamma_4 + \gamma_5 \Delta t} \max_{j \in \{1,\ldots,N\}} \min_{\hat{\mathbf{v}}} \|\mathbf{r}_{\mathrm{LSPG}}^j(\mathbf{\Phi}\hat{\mathbf{v}})\|_2$$

Can we use these error bounds for error estimation?

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

Main idea

- **Observation**: residual-based quantities are informative of the error



- So, these are good features: can predict the error with low variance

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

Machine-learning error models

Machine-learning error models: formulation

$$\delta(\boldsymbol{\mu}) = \underbrace{f(\boldsymbol{\rho}(\boldsymbol{\mu}))}_{\text{deterministic}} + \underbrace{\epsilon(\boldsymbol{\rho}(\boldsymbol{\mu}))}_{\text{stochastic}}$$

- features: $\boldsymbol{\rho}(\boldsymbol{\mu}) \in \mathbb{R}^{N_{\boldsymbol{\rho}}}$
- regression function: $f(\boldsymbol{\rho}) = \mathrm{E}[\delta \mid \boldsymbol{\rho}]$
- noise: $\epsilon(\boldsymbol{\rho})$

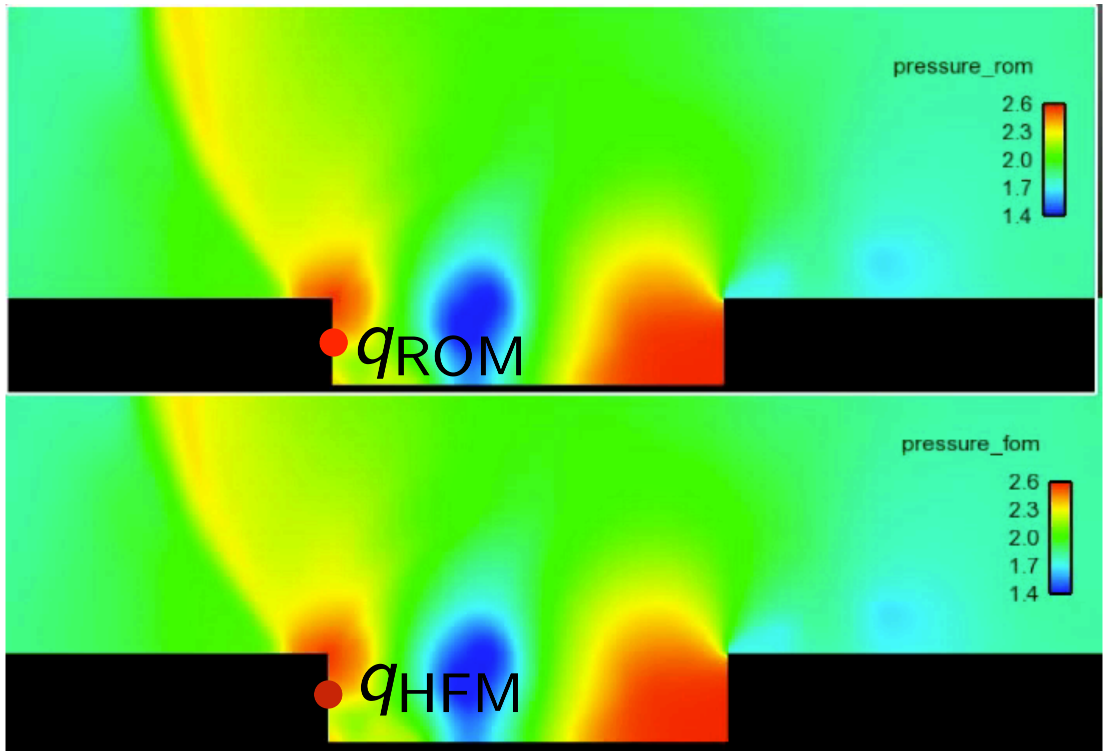
$$\tilde{\delta}(\boldsymbol{\mu}) = \underbrace{\tilde{f}(\boldsymbol{\rho}(\boldsymbol{\mu}))}_{\text{deterministic}} + \underbrace{\tilde{\epsilon}(\boldsymbol{\rho}(\boldsymbol{\mu}))}_{\text{stochastic}}$$

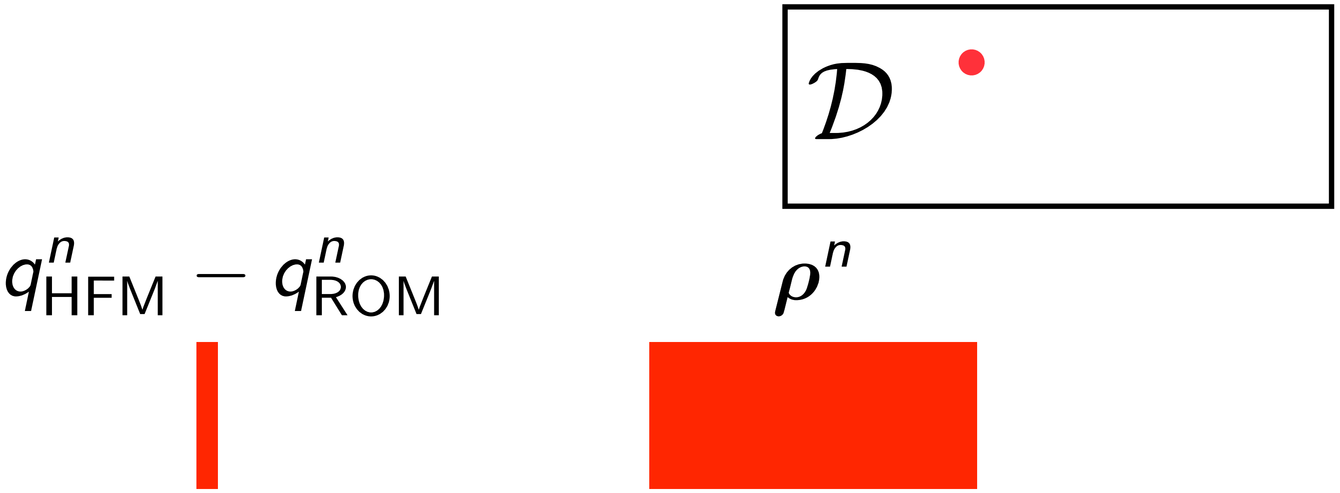
- regression-function model: $\tilde{f}(\approx f)$
- noise model: $\tilde{\epsilon}(\approx \epsilon)$

- Desired properties in error model $\tilde{\delta}$
  1. cheaply computable: features $\boldsymbol{\rho}(\boldsymbol{\mu})$ are inexpensive to compute
  2. low variance: noise model $\tilde{\epsilon}(\boldsymbol{\rho})$ has low variance
  3. generalizable: empirical distributions of δ and $\tilde{\delta}$ 'close' on test data

Training and machine learning: error modeling

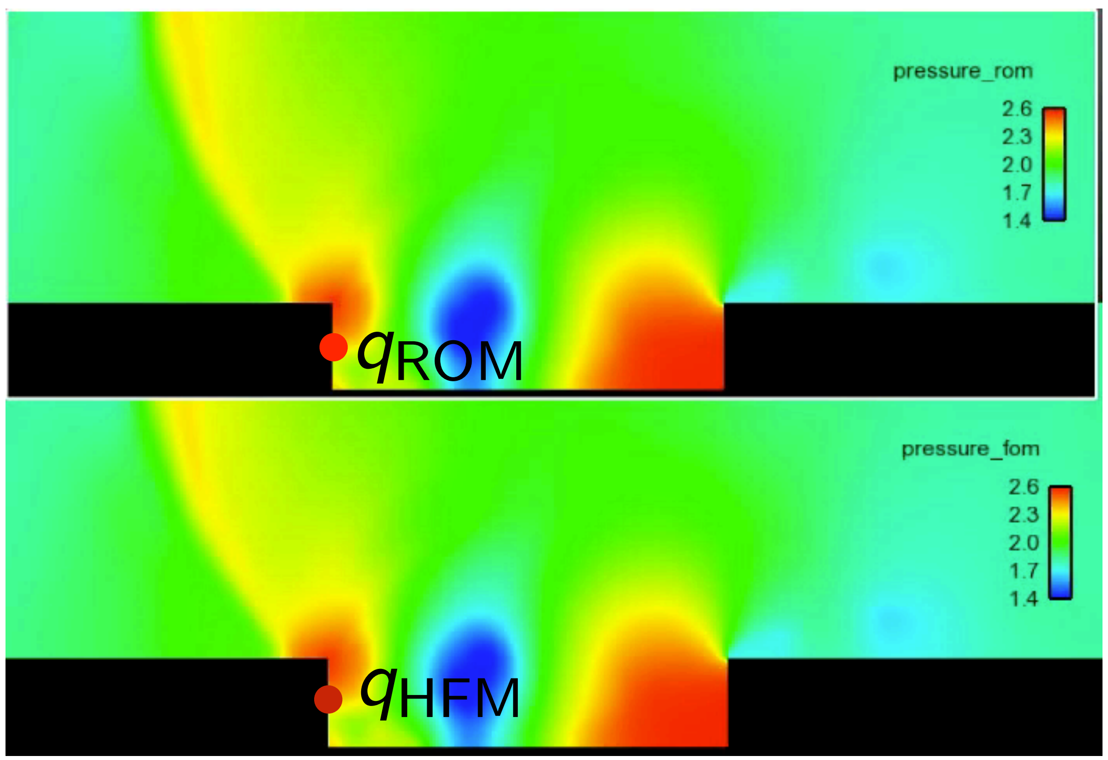
1. *Training:* Solve high-fidelity and reduced-order models for $\boldsymbol{\mu} \in \mathcal{D}_{\text{training}}$
2. *Machine learning:* Construct regression model
3. *Reduction:* predict reduced-order-model error for $\boldsymbol{\mu} \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$

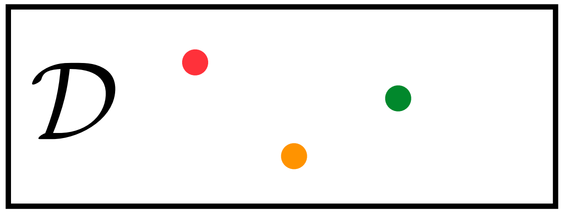




Training and machine learning: error modeling

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 $\mu \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$



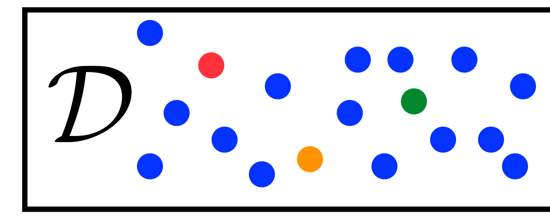


$q^n_{\text{HFM}} - q^n_{\text{ROM}}$ $\qquad$ $\boldsymbol{\rho}^n$

- randomly divide data into (1) training data and (2) testing data
- construct regression-function model $\tilde{f}$ via cross validation on **training data**
- construct noise model $\tilde{\epsilon}$ 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^n_{ROM}, $n = 1, \ldots, T$

↓

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

↓

regression model

$$\tilde{\delta}^{\text{n}}(\boldsymbol{\mu}) = \tilde{f}(\boldsymbol{\rho}^{\text{n}}(\boldsymbol{\mu})) + \tilde{\epsilon}(\boldsymbol{\rho}^{\text{n}}(\boldsymbol{\mu}))$$

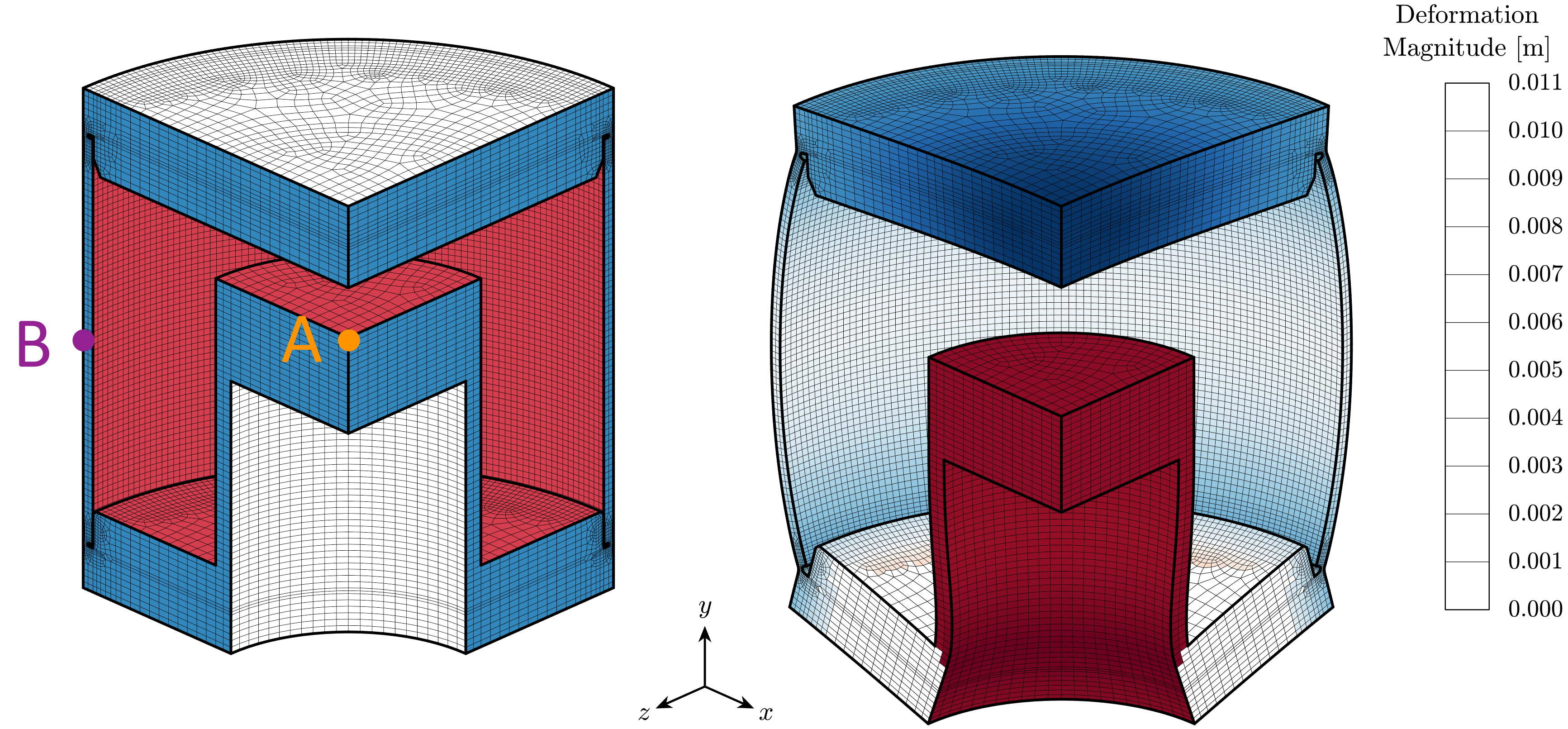
→ *machine learning error model* $\tilde{\delta}^{\text{n}}$, $n = 1, \ldots, T$

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

+ *Statistical model of high-fidelity-model output*

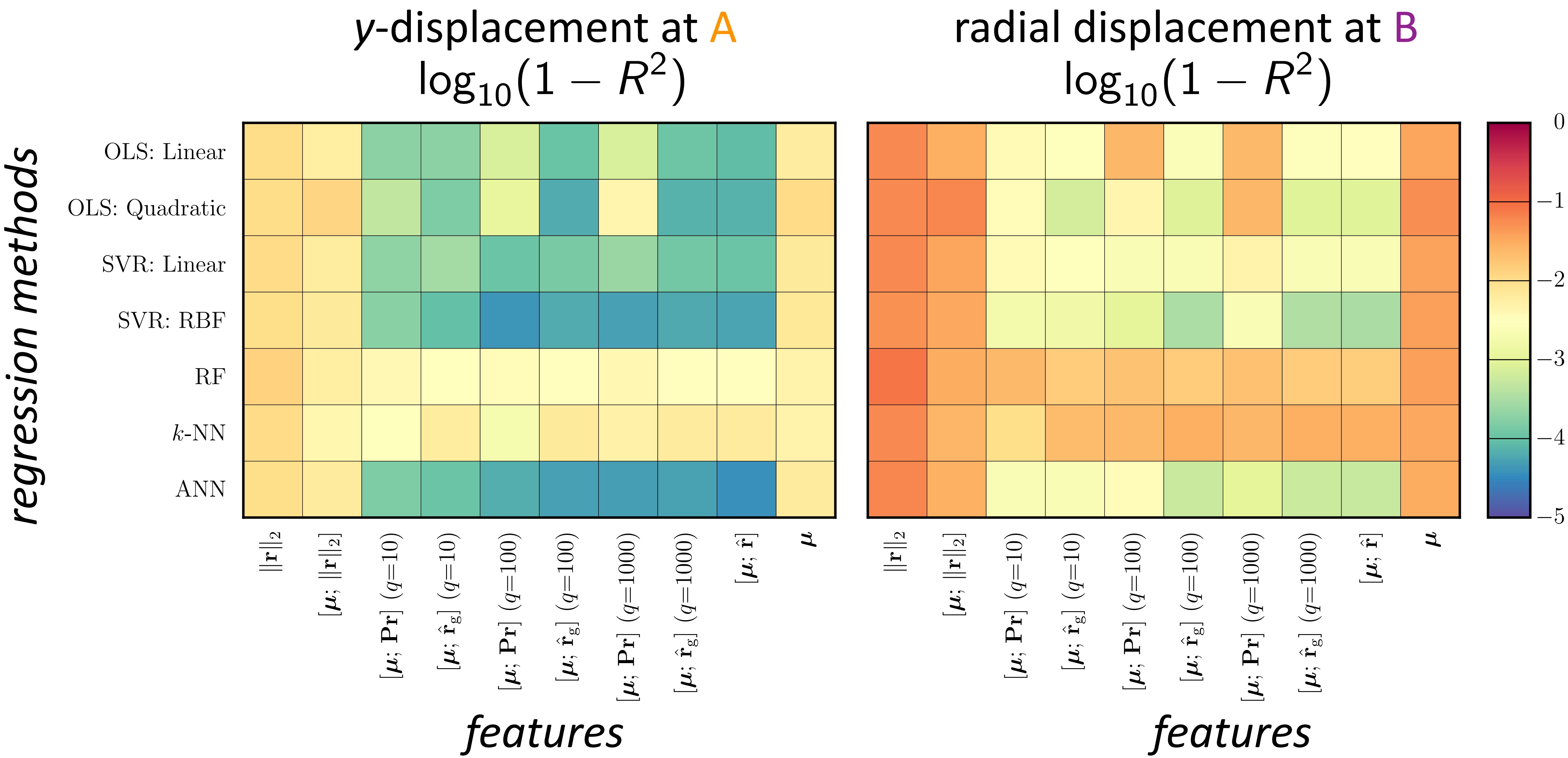
Use error analysis to engineer features $\boldsymbol{\rho}^n$

Application: Predictive capability assessment project

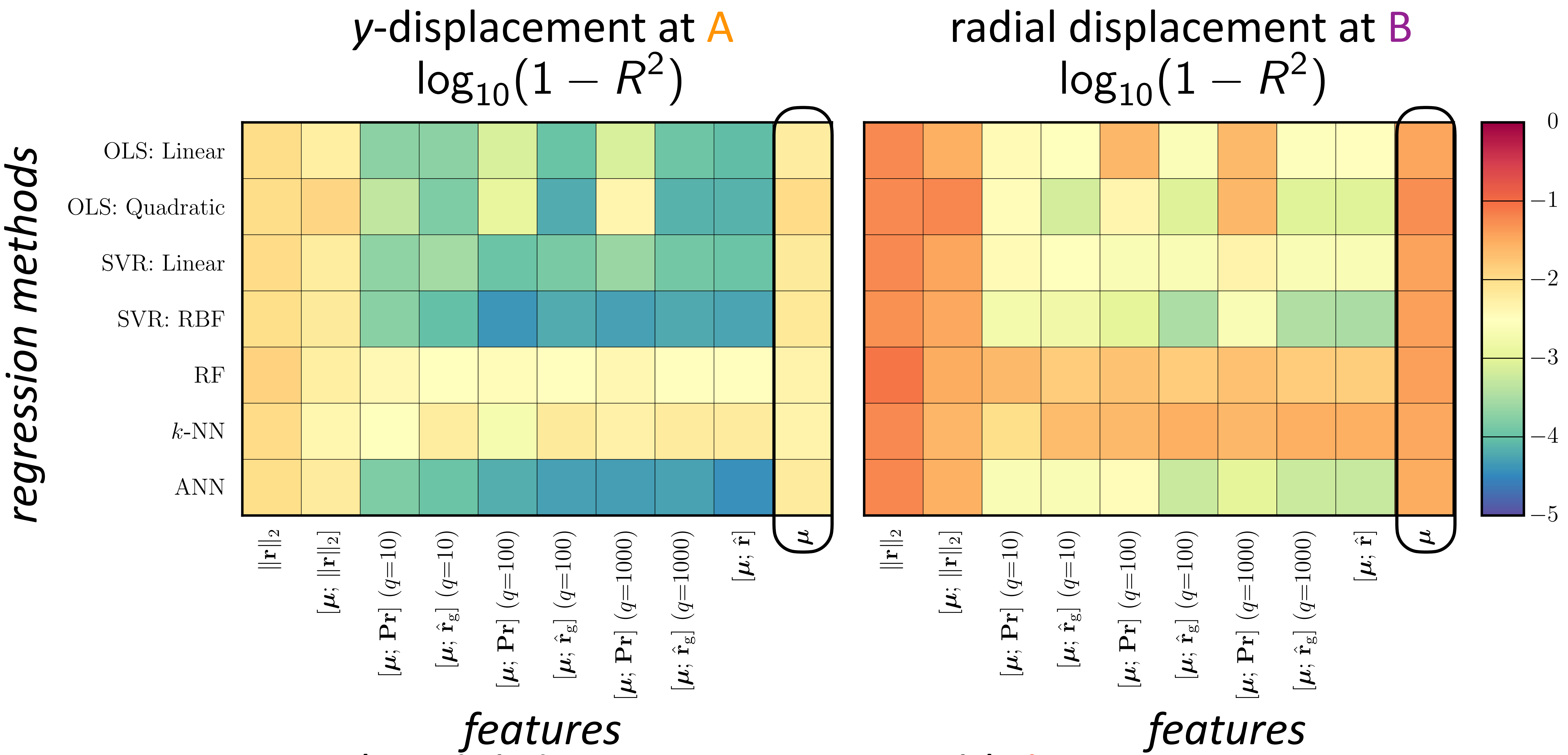


- *high-fidelity model dimension:* 2.8×10^5
- *reduced-order model dimensions:* $1, \ldots, 5$
- *inputs* $\boldsymbol{\mu}$*:* elastic modulus, Poisson ratio, applied pressure
- *quantities of interest: y*-displacement at A, radial displacement at B
- *training data*: 150 training examples, 150 testing examples

Application: Predictive capability assessment project

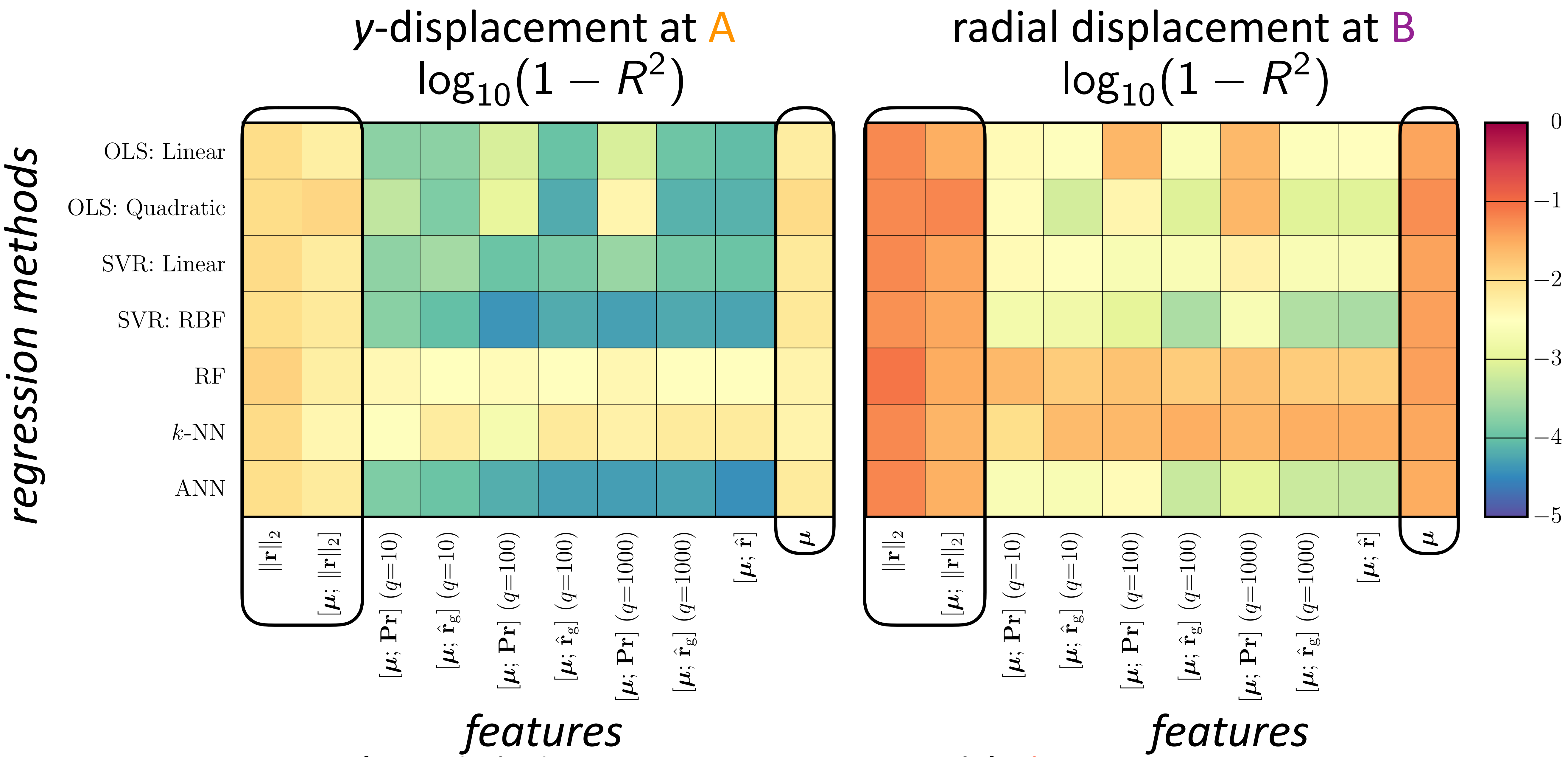


Application: Predictive capability assessment project



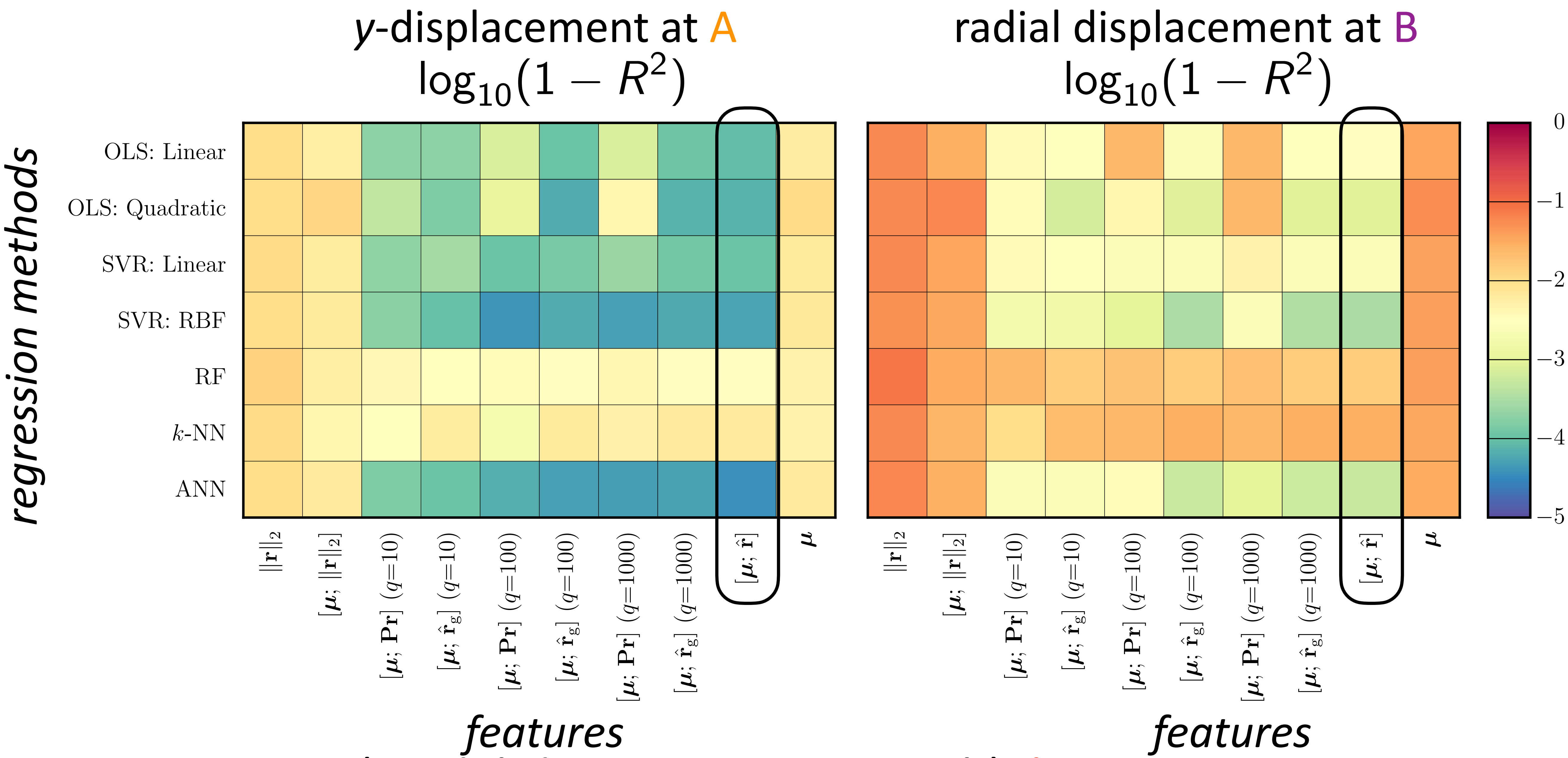
- parameters (model-discrepancy approach): large variance

Application: Predictive capability assessment project



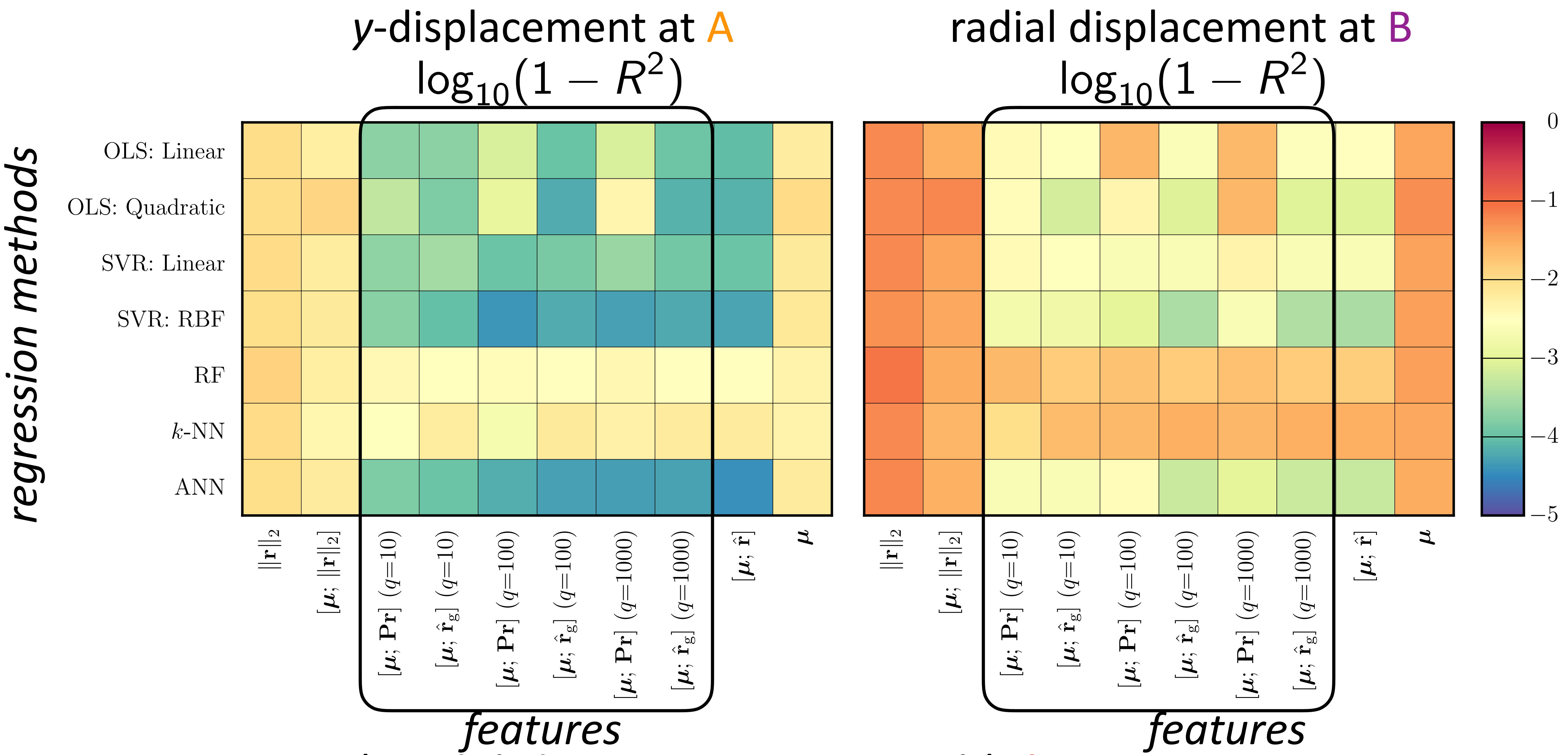
- parameters (model-discrepancy approach): large variance
- small number of low-quality features: large variance

Application: Predictive capability assessment project



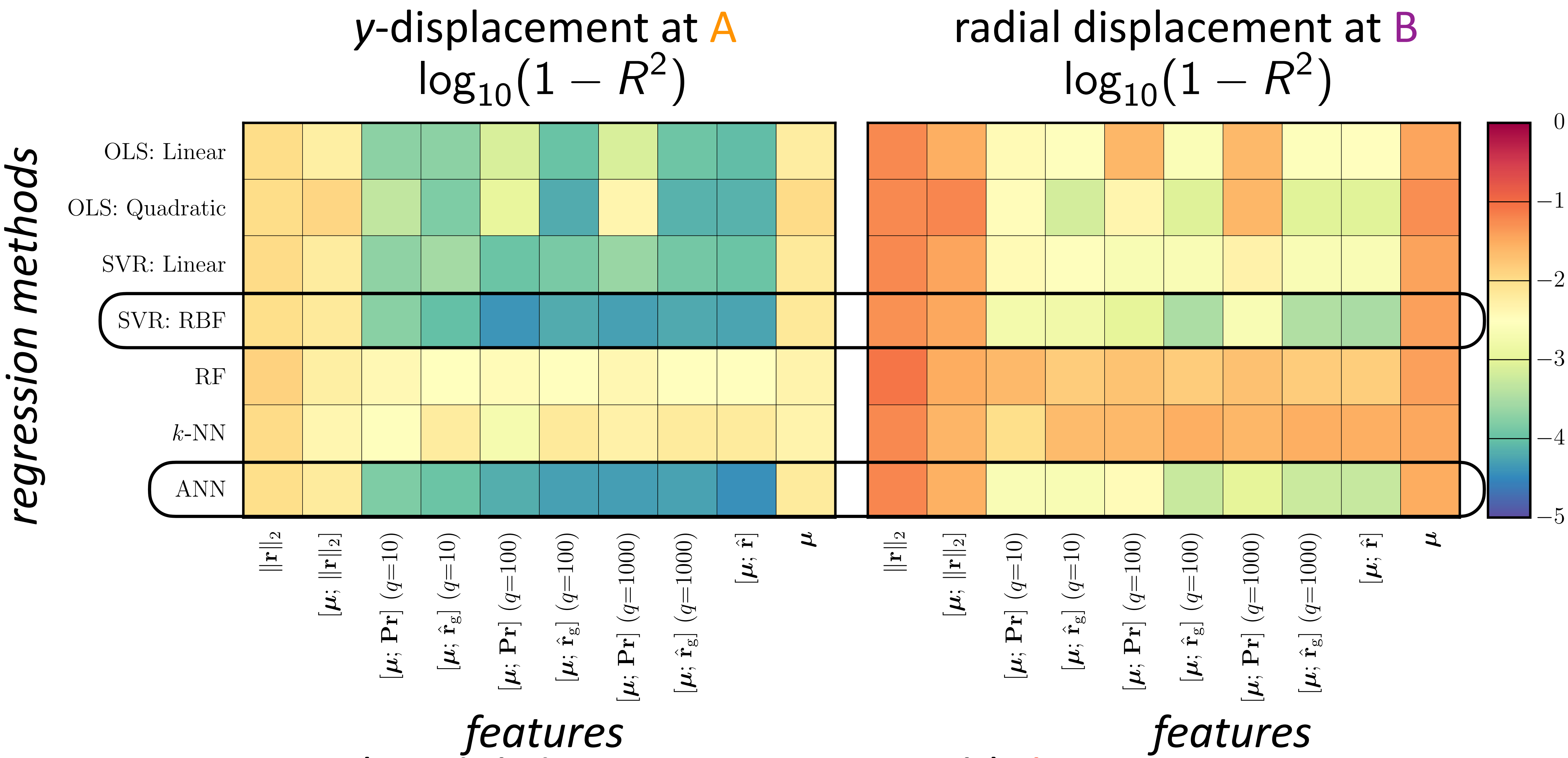
- parameters (model-discrepancy approach): large variance
- small number of low-quality features: large variance
- PCA of the residual: lowest variance overall but costly

Application: Predictive capability assessment project



- parameters (model-discrepancy approach): large variance
- small number of low-quality features: large variance
- PCA of the residual: lowest variance overall but costly
- \+ gappy PCA of the residual: nearly as low variance, but much cheaper

Application: Predictive capability assessment project



- parameters (model-discrepancy approach): large variance
- small number of low-quality features: large variance
- PCA of the residual: lowest variance overall but costly
- gappy PCA of the residual: nearly as low variance, but much cheaper
- neural networks and SVR: RBF yield lowest-variance models

Our research

- *accuracy*: LSPG projection

  K. Carlberg, M. Barone, and H. Antil. "Galerkin v. least-squares Petrov–Galerkin projection in nonlinear model reduction," Journal of Computational Physics, Vol. 330, p. 693–734 (2017).

- *low cost*: sample mesh

  K. Carlberg, C. Farhat, J. Cortial, and D. Amsallam. "The GNAT method for nonlinear model reduction: Effective implementation and application to computational fluid dynamics and turbulent flows," Journal of Computational Physics, Vol. 242, p. 623–647 (2013).

- *low cost*: reduce temporal complexity

  Y. Choi and K. Carlberg. "Space–time least-squares Petrov–Galerkin projection for nonlinear model reduction," SIAM Journal on Scientific Computing, Vol. 41, No. 1, p. A26–A58 (2019).

- *structure preservation*

  K. Carlberg, Y. Choi, and S. Sargsyan. "Conservative model reduction for finite-volume models," Journal of Computational Physics, Vol. 371, p. 280–314 (2018).

- *robustness*: projection onto nonlinear manifolds

  K. Lee and K. Carlberg. "Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders," arXiv e-Print, 1812.08373 (2018).

- *robustness*: *h*-adaptivity

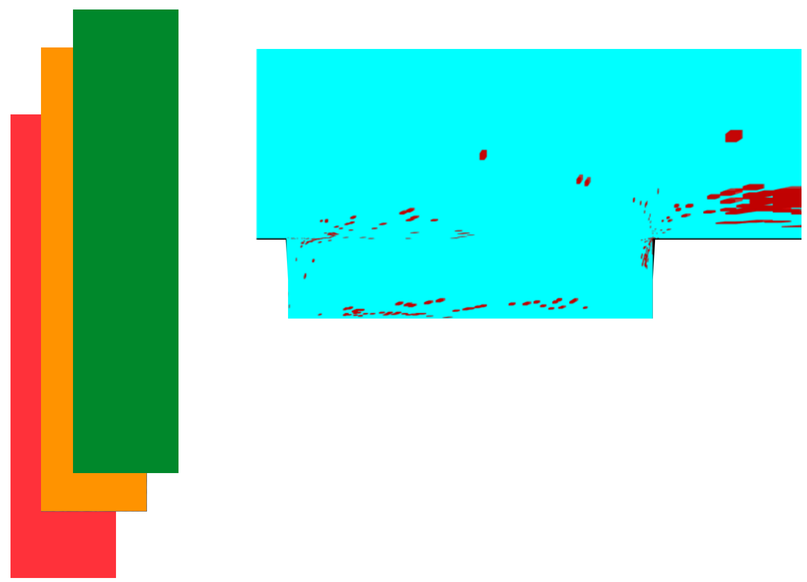
  K. Carlberg. "Adaptive h-refinement for reduced-order models," International Journal for Numerical Methods in Engineering, Vol. 102, No. 5, p.1192–1210 (2015).

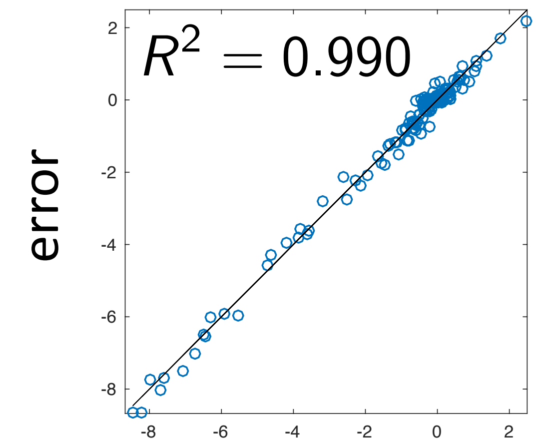
- *certification*: machine learning error models

  B. Freno and K. Carlberg. "Machine-learning error models for approximate solutions to parameterized systems of nonlinear equations," Computer Methods in Applied Mechanics and Engineering, accepted (2019).

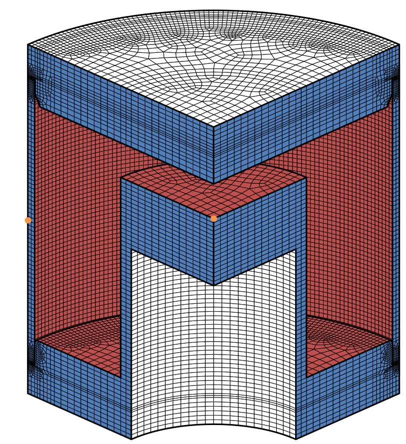
Questions?

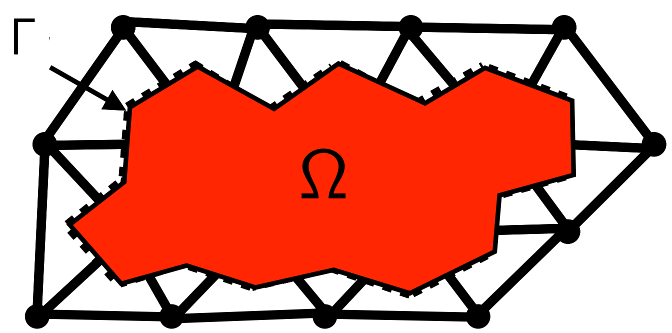
$\mathcal{X}^{ijk} =$

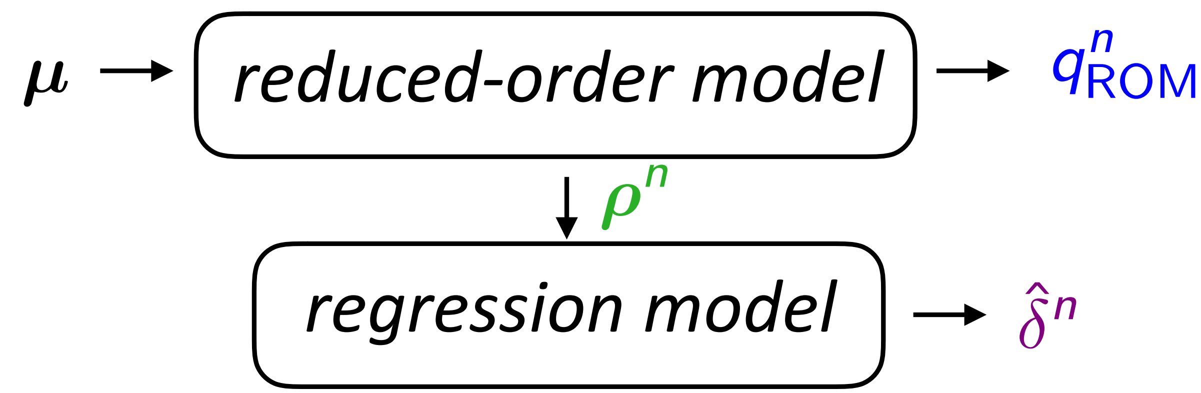


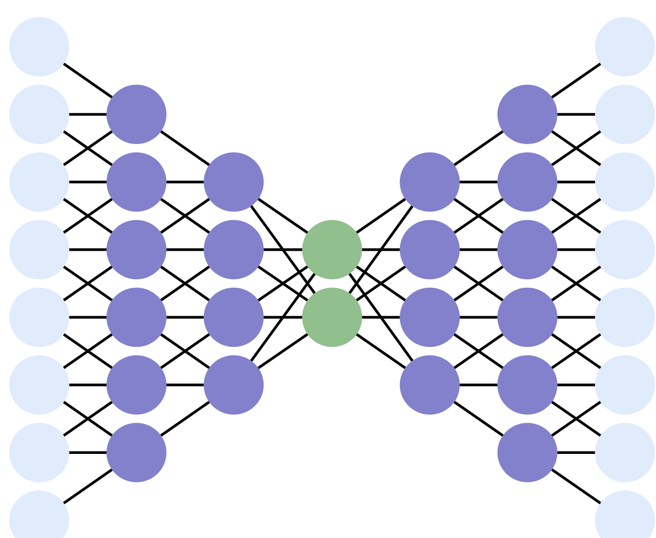


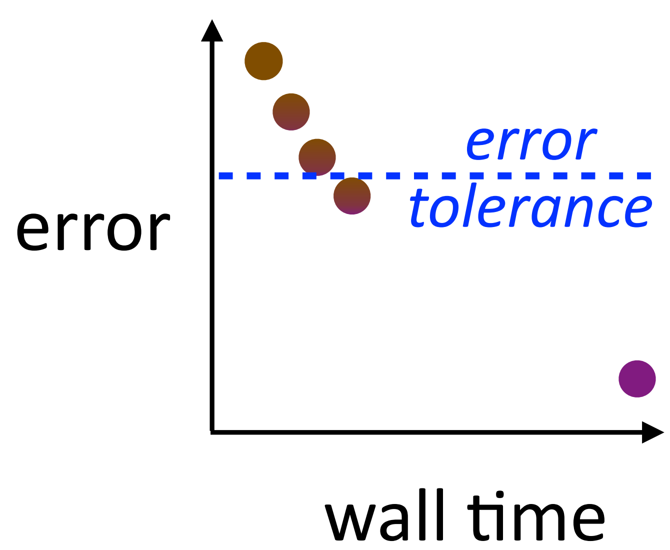
support vector machine
error prediction

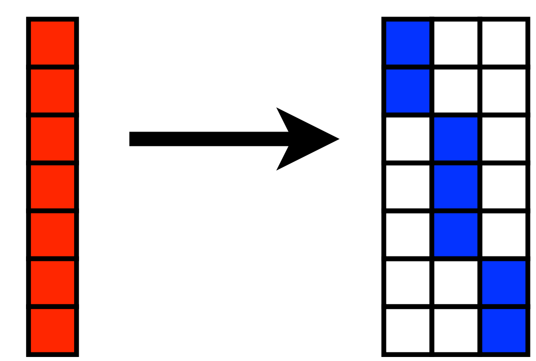












Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525