Nonlinear reduced-order modeling
Using machine learning to enable extreme-scale simulations for many-query problems

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

**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
“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: \[ \frac{dx}{dt} = f(x; t, \mu), \quad x(0, \mu) = x_0(\mu), \quad t \in [0, T_{\text{final}}], \quad \mu \in D \]

Many-query problem: solve ODE for \( \mu \in D_{\text{query}} \)

Idea: exploit simulation data collected at a few points

1. Training: Solve ODE for \( \mu \in D_{\text{training}} \) and collect simulation data
2. Machine learning: Identify structure in data
3. Reduction: Reduce cost of ODE solve for \( \mu \in D_{\text{query}} \setminus D_{\text{training}} \)
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

ODE: \[ \frac{dx}{dt} = f(x; t, \mu) \]

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 \( \mu \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}} \)
Training simulations: state tensor

\[
\text{ODE: } \quad \frac{dx}{dt} = f(x; t, \mu)
\]

1. **Training:** Solve ODE for \( \mu \in D_{\text{training}} \) and collect simulation data
2. **Machine learning:** Identify structure in data
3. **Reduction:** Reduce the cost of solving ODE for \( \mu \in D_{\text{query}} \setminus D_{\text{training}} \)
Tensor decomposition

ODE: \[
\frac{dx}{dt} = f(x; t, \mu)
\]

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

Compute dominant left singular vectors of mode-1 unfolding

\[
\chi = X(1) = U \Sigma V^T
\]
### Tensor decomposition

ODE: \[
\frac{dx}{dt} = f(x; t, \mu)
\]

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

Compute dominant left singular vectors of mode-1 unfolding

\[
X = \begin{bmatrix}
\end{bmatrix} \quad X_{(1)} = \begin{bmatrix}
\end{bmatrix} = \Phi \begin{bmatrix}
\end{bmatrix} \begin{bmatrix}
\end{bmatrix} = \begin{bmatrix}
\end{bmatrix}
\]

**\( \Phi \) 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: \[ \frac{dx}{dt} = f(x; t, \mu) \]

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

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

\[ \Phi^T f(\Phi \hat{x}; t, \mu) - \Phi \frac{d\hat{x}}{dt} = 0 \]

Galerkin ODE: \[ \frac{d\hat{x}}{dt} = \Phi^T f(\Phi \hat{x}; t, \mu) \]
Captive carry

- Unsteady Navier–Stokes
  - Re = \( 6.3 \times 10^6 \)
  - \( M_\infty = 0.6 \)

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
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**
\[
\frac{dx}{dt} = f(x; t)
\]

**Galerkin ODE**
\[
\Phi \frac{d\hat{x}}{dt} = \Phi \Phi^T f(\Phi\hat{x}; t)
\]

*Time-continuous Galerkin solution: optimal* in the minimum-residual sense:
\[
\Phi \frac{d\hat{x}}{dt} (x, t) = \arg\min_{v \in \text{range}(\Phi)} \| r(v, x; t) \|_2
\]
\[
r(v, x; t) := v - f(x; t)
\]

**ODE**
\[
r^n(x^n) = 0, \ n = 1, \ldots, T
\]

**Galerkin ODE**
\[
\Phi^T r^n(\Phi\hat{x}^n) = 0, \ n = 1, \ldots, T
\]

*Time-discrete Galerkin solution: not generally optimal* in any sense
Residual minimization and time discretization

**ODE**
\[
\begin{align*}
\frac{dx}{dt} &= f(x; t) \\
\end{align*}
\]

**Galerkin ODE**
\[
\begin{align*}
\frac{d\hat{x}}{dt}(x, t) &= \arg\min_{v \in \text{range}(\Phi)} \|r(v, x; t)\|_2 \\
\end{align*}
\]

**LSPG ODE**
\[
\Phi \hat{x}^n = \arg\min_{v \in \text{range}(\Phi)} \|A\hat{x}^n(v)\|_2 \\
^n = 1, \ldots, T
\]

**Galerkin ODE**
\[
\begin{align*}
\Phi^T r^n(\Phi \hat{x}^n) &= 0 \\
^n = 1, \ldots, T
\end{align*}
\]

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

\[
\begin{align*}
\Phi \hat{x}^n &= \arg\min_{v \in \text{range}(\Phi)} \|A\hat{x}^n(v)\|_2 \\
\iff \psi^n(\hat{x}^n)^T r^n(\Phi \hat{x}^n) &= 0 \\
\psi^n(\hat{x}^n) &:= A^T A(\alpha_0 I - \Delta t/\beta_0 \frac{\partial f}{\partial x}(\Phi \hat{x}^n; t)) \Phi
\end{align*}
\]

Least-squares Petrov–Galerkin (LSPG) projection
Theorem [C., Barone, Antil, 2017]

If the following conditions hold:

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

\[
\begin{align*}
\| x^n - \Phi \hat{x}_G^n \|_2 & \leq \frac{1}{h} \left\| r_G^n(\Phi \hat{x}_G^n) \right\|_2 + \frac{1}{h} \sum_{\ell=1}^{k} |\alpha_\ell| \| x^{n-\ell} - \Phi \hat{x}_G^{n-\ell} \|_2 \\
\| x^n - \Phi \hat{x}_{LSPG}^n \|_2 & \leq \frac{1}{h} \min_{\hat{\nu}} \left\| r_{LSPG}(\Phi \hat{\nu}) \right\|_2 + \frac{1}{h} \sum_{\ell=1}^{k} |\alpha_\ell| \| x^{n-\ell} - \Phi \hat{x}_{LSPG}^{n-\ell} \|_2
\end{align*}
\]

+ 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]
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 \cite{Everson and Sirovich, 1995}

\[
\text{minimize} \| A \mathbf{r}^n (\Phi \hat{\mathbf{v}}) \|_2
\]

*Can we select \( A \) to make this less expensive?*

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

\[\text{index} \quad \text{value}\]

\[\begin{array}{c}
\text{index} \quad \text{minimize} \| \tilde{\mathbf{r}}^n (\Phi \hat{\mathbf{v}}) \|_2 \\
\end{array}\]
Cost reduction by gappy PCA [Everson and Sirovich, 1995]

\[
\text{minimize} \| A r^n(\Phi \hat{v}) \|_2
\]

Can we select \( A \) to make this less expensive?

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

\[
\text{minimize} \| (P \Phi_r)^+ P r^n(\Phi \hat{v}) \|_2
\]

+ Only a few elements of \( r^n \) must be computed

Nonlinear reduced-order modeling

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

\[
\min_{\hat{v}} \| (P\Phi_r)^+ Pr^n(\Phi \hat{v}) \|_2
\]

+ HPC on a laptop

**Vorticity field**

**Pressure field**

LSPG ROM with \( A = (P\Phi_r)^+ P \)

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
- \( \text{Re} = 4.3 \times 10^6 \)
- \( \text{M}_\infty = 0.175 \)

Spatial discretization
- 2nd-order finite volume
- DES turbulence model
- \( 1.7 \times 10^7 \) degrees of freedom

Temporal discretization
- 2nd-order BDF
- Time step \( \Delta t = 8 \times 10^{-5} \text{s} \)
- \( 1.3 \times 10^3 \) time instances
Ahmed body results [C., Farhat, Cortial, Amsallem, 2013]

LSPG ROM with $A = (P\Phi_r)^+P$
4 hours, 4 cores

high-fidelity model
13 hours, 512 cores

+ HPC on a laptop

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

ODE: \( \frac{dx}{dt} = f(x; t) \)

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

- average value of conserved variable \( i \) over control volume \( j \)

\[ f_{I(i,j)}(x, t) = -\frac{1}{|\Omega_j|} \int_{\Gamma_j} g_i(x; \vec{x}, t) \cdot n_j(\vec{x}) \, ds(\vec{x}) + \frac{1}{|\Omega_j|} \int_{\Omega_j} s_i(x; \vec{x}, t) \, d\vec{x} \]

- flux and source of conserved variable \( i \) within control volume \( j \)

\[ r_{I(i,j)} = \frac{dx_{I(i,j)}}{dt}(t) - f_{I(i,j)}(x, t) \]

- rate of conservation violation of variable \( i \) in control volume \( j \)

\[ O\Delta E: \quad r^n(x^n) = 0, \quad n = 1, \ldots, N \]

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

- conservation violation of variable \( i \) in control volume \( j \) over time step \( n \)
Conservative model reduction [C., Choi, Sargsyan, 2018]

\[
\Phi \frac{d\hat{x}}{dt}(x, t) = \arg\min_{v \in \text{range}(\Phi)} \|r(v, x; t)\|_2
\]

- min. sum of squared conservation-violation rates

subject to \(\text{Cr}(v, x; t) = 0\)

- Neither enforces conservation!

Conservative Galerkin

\[\Phi \hat{x}^n = \arg\min_{v \in \text{range}(\Phi)} \|\text{Ar}^n(v)\|_2\]

- min. sum of squared conservation violations over time step \(n\)

subject to \(\text{Cr}^n(v) = 0\)

Conservative LSPG

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

\[\hat{x}^n = \arg\min_{v \in \text{range}(\Phi)} \|\text{Ar}^n(v)\|_2\]

- min. sum of squared conservation violations over time step \(n\) over subdomains

subject to zero conservation-violation rates over subdomains

- Conservation enforced over subdomains!

\[\text{Ar}^n(v)\]

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]
Model reduction can work well...

**vorticity field**

LSPG ROM with

\[ \mathbf{A} = (\mathbf{P} \Phi_r)^+ \mathbf{P} \]

32 min, 2 cores

**pressure field**

high-fidelity

5 hours, 48 cores

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

... however, this is **not guaranteed**

\[ \mathbf{x}(t) \approx \Phi \hat{\mathbf{x}}(t) \]

1) **Linear-subspace assumption is strong**

2) **Accuracy limited by information in** \( \Phi \)
Model reduction can work well...

vorticity field

pressure field

LSPG ROM with

\[ A = (P\Phi_r)^+ P \]

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

\[ x(t) \approx \Phi \dot{x}(t) \]

1) Linear-subspace assumption is strong

2) Accuracy limited by information in \( \Phi \)
Kolmogorov-width limitation of linear subspaces

\[ d_p(\mathcal{M}) := \inf_{S_p} P_\infty(\mathcal{M}, S_p) \quad P_\infty(\mathcal{M}, S_p) := \sup_{x \in \mathcal{M}} \inf_{y \in S_p} \| x - y \| \]

- \( \mathcal{M} := \{ x(t, \mu) \mid t \in [0, T_{\text{final}}], \mu \in \mathcal{D} \} \): solution manifold
- \( S_p \): set of all \( p \)-dimensional linear subspaces
Kolmogorov-width limitation of linear subspaces

\[ \tilde{d}_p(\mathcal{M}) := \inf_{S_p} P_2(\mathcal{M}, S_p) \]

\[ P_2(\mathcal{M}, S_p) := \sqrt{\sum_{x \in \mathcal{M}} \inf_{y \in S_p} \|x - y\|^2} / \sqrt{\sum_{x \in \mathcal{M}} \|x\|^2} \]

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

**Problem 1**

\[ \tilde{d}_p(\mathcal{M}) \]

**Problem 2**

\[ P_2(\mathcal{M}, \text{range}(\Phi)) \]

\[ \sqrt{\sum_{x \in \mathcal{M}} \|x - \tilde{x}_{\text{LSPG}}\|^2} / \sqrt{\sum_{x \in \mathcal{M}} \|x\|^2} \]

\[ \text{dim}(\mathcal{M}) \]

- Kolmogorov-width limitation: significant error for \( p = \text{dim}(\mathcal{M}) \)
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]
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 reduced-order modeling

**Linear trial subspace**

\[ \text{range}(\Phi) := \{ \Phi \hat{x} | \hat{x} \in \mathbb{R}^p \} \]

**Nonlinear trial manifold**

\[ S := \{ g(\hat{x}) | \hat{x} \in \mathbb{R}^p \} \]

*Example*

\[ N=3 \quad p=2 \]

\[ \begin{align*}
\text{state} & \quad x(t) \approx \tilde{x}(t) = \Phi \hat{x}(t) \in \text{range}(\Phi) \\
\text{velocity} & \quad \frac{dx}{dt} \approx \frac{d\tilde{x}}{dt} = \Phi \frac{d\hat{x}}{dt} \in \text{range}(\Phi)
\end{align*} \]

**manifold has general structure**

\[ \frac{dx}{dt} \approx \frac{d\tilde{x}}{dt} = \nabla g(\hat{x}) \frac{d\hat{x}}{dt} \in T_{\hat{x}}S \]

**kinematically consistent**
Manifold Galerkin and LSPG projection

**Linear-subspace ROM**

Galerkin

\[
\frac{d\hat{x}}{dt} = \arg\min_{\hat{v} \in \mathbb{R}^n} \| r(\Phi\hat{v}, \Phi\hat{x}; t) \|_2
\]

\[
\Phi \frac{d\hat{x}}{dt} = \arg\min_{\hat{v} \in \text{range}(\Phi)} \| \hat{v} - f(\Phi\hat{x}; t) \|_2
\]

\[
\frac{d\hat{x}}{dt} = \Phi^T f(\Phi\hat{x}; t)
\]

**Nonlinear-manifold ROM**

\[
\frac{d\hat{x}}{dt} = \arg\min_{\hat{v} \in \mathbb{R}^n} \| r(\nabla g(\hat{x})\hat{v}, g(\hat{x}); t) \|_2
\]

\[
\nabla g(\hat{x}) \frac{d\hat{x}}{dt} = \arg\min_{\hat{v} \in T_{\hat{x}}S} \| \hat{v} - f(g(\hat{x}); t) \|_2
\]

\[
\frac{d\hat{x}}{dt} = \nabla g(\hat{x})^+ f(g(\hat{x}); t)
\]

**LSPG**

\[
\hat{x}^n = \arg\min_{\hat{v} \in \mathbb{R}^p} \| \mathbf{A} r^n(\Phi\hat{v}) \|_2
\]

\[
\hat{x}^n = \arg\min_{\hat{v} \in \mathbb{R}^p} \| \mathbf{A} r^n(\mathbf{g}(\hat{v})) \|_2
\]

+ Satisfy optimality properties

**How to construct manifold** 

\[
S := \{ \mathbf{g}(\hat{x}) | \hat{x} \in \mathbb{R}^P \} \text{ from snapshot data?}
\]
Deep autoencoders

Input layer

Code

Output layer

Encoder $h_{enc}(\cdot; \theta_{enc})$

Decoder $h_{dec}(\cdot; \theta_{dec})$

$\hat{x} = h_{dec}(\cdot; \theta_{dec}) \circ h_{enc}(x; \theta_{enc})$
Deep autoencoders

**Input layer**

$\begin{align*}
&x_1 \\
&x_2 \\
&x_3 \\
&x_4 \\
&x_5 \\
&x_6 \\
&x_7 \\
&x_8
\end{align*}$

**Code**

$\begin{align*}
&\hat{x}_1 \\
&\hat{x}_2
\end{align*}$

**Output layer**

$\begin{align*}
&\tilde{x}_1 \\
&\tilde{x}_2 \\
&\tilde{x}_3 \\
&\tilde{x}_4 \\
&\tilde{x}_5 \\
&\tilde{x}_6 \\
&\tilde{x}_7 \\
&\tilde{x}_8
\end{align*}$

**Encoder** $h_{\text{enc}}(\cdot ; \theta_{\text{enc}})$  
**Decoder** $h_{\text{dec}}(\cdot ; \theta_{\text{dec}})$

\[
\tilde{x} = h_{\text{dec}}(\cdot ; \theta_{\text{dec}}) \circ h_{\text{enc}}(x ; \theta_{\text{enc}})
\]

+ If $\tilde{x} \approx x$ for parameters $\theta^*_{\text{dec}}$, $g = h_{\text{dec}}(\cdot ; \theta^*_{\text{dec}})$ produces an accurate manifold
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 $\mu \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}}$
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 $\theta^*$ by approximately solving $\min_{\theta} \| X_{(1)} - \tilde{X}_{(1)}(\theta) \|_F$
- Define nonlinear trial manifold by setting $g = h_{\text{dec}}(\cdot; \theta^*_{\text{dec}})$

* No problem-specific knowledge required
* Same snapshot data
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 \( \mu \in \mathcal{D}_{\text{query}} \setminus \mathcal{D}_{\text{training}} \)

\[ x = \begin{bmatrix} \beta \end{bmatrix} \]

\[ x^{(1)} = \begin{bmatrix} \beta \end{bmatrix} \]

\[ \theta_{\text{enc}} \quad \theta_{\text{dec}} \]

\[ = \tilde{x}^{(1)}(\theta) \]

- Compute \( \theta^* \) by approximately solving \( \min_{\theta} \| x^{(1)} - \tilde{x}^{(1)}(\theta) \|_F \)
- Define nonlinear trial manifold by setting \( g = h_{\text{dec}}(\cdot; \theta^*_{\text{dec}}) \)
  + **No problem-specific knowledge** required
  + **Same snapshot data**
Numerical results

1D Burgers’ equation

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

- \( \mu : \alpha \), inlet boundary condition
- \textit{Spatial discretization}: finite volume
- \textit{Time integrator}: backward Euler

2D Chemically reacting flow

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

- \( \mu : \) two terms in reaction
- \textit{Spatial discretization}: finite difference
- \textit{Time integrator}: BDF2

Autoencoder architecture

[Diagram showing autoencoder architecture with 4 convolutional layers, 2 fully-connected layers, and output layers]
Results: nonlinear manifold interpretation

1D Burgers’ equation

\[ t = 13.16, \ (\mu_1, \mu_2) = (4.53, \ 0.015) \]

2D Chemically reacting flow

\[ t = 0.023, \ (\mu_1, \mu_2) = (6.5e+12, \ 9.0e+03) \]

Reduced state: \( \hat{x} \)

Decoding: \( g(\hat{x}) \)

Conserved variable \( w \)

Spatial variable \( x \)

Temperature

\( H_2O \) fraction

\( O_2 \) fraction

\( H_2 \) fraction
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

**Problem 1**

- Autoencoder manifold **significantly better** than optimal linear subspace

**Problem 2**

\[ \tilde{d}_p(\mathcal{M}) \]

\[ P_2(\mathcal{M}, \text{range}(\Phi)) \]

\[ \sqrt{\sum_{\mathbf{x} \in \mathcal{M}} \| \mathbf{x} - \tilde{\mathbf{x}}_{\text{LSPG}} \|^2} / \sqrt{\sum_{\mathbf{x} \in \mathcal{M}} \| \mathbf{x} \|^2} \]

\[ \dim(\mathcal{M}) \]

\[ P_2(\mathcal{M}, \mathcal{S}) \]
Method overcomes Kolmogorov-width limitation

**Problem 1**

- Autoencoder manifold significantly better than optimal linear subspace

**Problem 2**

- 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...

\[
A = (P \Phi_r)^+ P
\]

32 min, 2 cores

Vorticity field

Pressure field

LSPG ROM with

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

\[
x(t) \approx \Phi \hat{x}(t)
\]

1) Linear-subspace assumption is strong

2) Accuracy limited by information in \( \Phi \)

Nonlinear reduced-order modeling
Illustration: inviscid 1D Burgers’ equation

high-fidelity model

time = 7.7

conserved variable

spatial variable

conserved variable

spatial variable
Illustration: inviscid 1D Burgers’ equation

**high-fidelity model**

- Conserved variable vs. spatial variable
- Time = 13.9

**reduced-order model**

- Conserved variable vs. spatial variable
- Time = 14

*Reduced-order model inaccurate when $\Phi$ insufficient*
Main idea \[C., \text{2015}\]

Model-reduction analogue to mesh-adaptive $h$-refinement

- ‘Split’ basis vectors

\[\text{finite-element} \quad h\text{-refinement}\]
- Generate hierarchical subspaces

\[
\text{range} \begin{pmatrix} 1 \end{pmatrix} \subseteq \text{range} \begin{pmatrix} 1 \\ 1 \\ \vdots \end{pmatrix}
\]

- Converges to the high-fidelity model

\[\text{reduced-order-model} \quad h\text{-refinement}\]
Illustration: inviscid 1D Burgers’ equation

high-fidelity model

reduced-order model (dim 50)

h-adaptive ROM (mean dim 48.5)

+ no longer limited by training data
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]
Discrete-time error bound

**Theorem** [C., Barone, Antil, 2017]

If the following conditions hold:

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

\[
\|x^n - \Phi \hat{x}^n_G\|_2 \leq \frac{\gamma_1 (\gamma_2)^n \exp(\gamma_3 t^n)}{\gamma_4 + \gamma_5 \Delta t} \max_{j \in \{1,\ldots,N\}} \|r^j_G(\Phi \hat{x}^j_G)\|_2 \\
\|x^n - \Phi \hat{x}^n_{LSPG}\|_2 \leq \frac{\gamma_1 (\gamma_2)^n \exp(\gamma_3 t^n)}{\gamma_4 + \gamma_5 \Delta t} \max_{j \in \{1,\ldots,N\}} \min_{\hat{v}} \|r^j_{LSPG}(\Phi \hat{v})\|_2
\]

*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

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

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

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

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

Desired properties in error model \( \tilde{\delta} \)

1. cheaply computable: features \( \rho(\mu) \) are inexpensive to compute
2. low variance: noise model \( \tilde{\epsilon}(\rho) \) has low variance
3. generalizable: empirical distributions of \( \delta \) and \( \tilde{\delta} \) ‘close’ on test data
Training and machine learning: error modeling

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

$$q_H^n - q_{\text{ROM}}^n$$
Training and machine learning: error modeling

1. **Training:** Solve high-fidelity and reduced-order models for $\mu \in D_{\text{training}}$

2. **Machine learning:** Construct regression model

3. **Reduction:** predict reduced-order-model error for $\mu \in D_{\text{query}} \setminus D_{\text{training}}$

- 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
1. **Training**: Solve high-fidelity and reduced-order models for $\mu \in D_{\text{training}}$

2. **Machine learning**: Construct regression model

3. **Reduction**: predict reduced-order-model error for $\mu \in D_{\text{query}} \setminus D_{\text{training}}$

**Diagram**

- **Inputs** $\mu$ → **Reduced-order model** → **Outputs** $q^n_{\text{ROM}}, n = 1, \ldots, T$

- **Features** $\rho^n, n = 1, \ldots, T$

- **Regression model** $\tilde{\delta}^n(\mu) = \tilde{f}(\rho^n(\mu)) + \tilde{\epsilon}(\rho^n(\mu))$

- **Machine learning error model** $\tilde{\delta}^n, n = 1, \ldots, T$

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

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

**Use error analysis to engineer features** $\rho^n$
Application: Predictive capability assessment project

- **high-fidelity model dimension:** $2.8 \times 10^5$
- **reduced-order model dimensions:** 1, ..., 5
- **inputs $\mu$:** 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

Regression Methods

<table>
<thead>
<tr>
<th>Features</th>
<th>OLS: Linear</th>
<th>OLS: Quadratic</th>
<th>SVR: Linear</th>
<th>SVR: RBF</th>
<th>RF</th>
<th>k-NN</th>
<th>ANN</th>
</tr>
</thead>
<tbody>
<tr>
<td>$| \mathbf{r} |_2$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu; Pr$ ($q=10$)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu; \hat{\mathbf{r}}_g$ ($q=10$)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu; Pr$ ($q=100$)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu; \hat{\mathbf{r}}_g$ ($q=100$)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu; Pr$ ($q=1000$)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu; \hat{\mathbf{r}}_g$ ($q=1000$)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Regression methods

- OLS: Linear
- OLS: Quadratic
- SVR: Linear
- SVR: RBF
- RF
- k-NN
- ANN

Frenoo & Carlberg

Machine-Learning Error Models for Approximate Solutions

$y$-displacement at A

\[ \log_{10}(1 - R^2) \]

radial displacement at B

\[ \log_{10}(1 - R^2) \]
Application: Predictive capability assessment project

\[ y\text{-displacement at } A \quad \log_{10}(1 - R^2) \]

\[ \text{radial displacement at } B \quad \log_{10}(1 - R^2) \]

<table>
<thead>
<tr>
<th>Regression methods</th>
<th>features</th>
</tr>
</thead>
<tbody>
<tr>
<td>OLS: Linear</td>
<td>parameters (model-discrepancy approach): large variance</td>
</tr>
<tr>
<td>OLS: Quadratic</td>
<td></td>
</tr>
<tr>
<td>SVR: Linear</td>
<td></td>
</tr>
<tr>
<td>SVR: RBF</td>
<td></td>
</tr>
<tr>
<td>RF</td>
<td></td>
</tr>
<tr>
<td>k-NN</td>
<td></td>
</tr>
<tr>
<td>ANN</td>
<td></td>
</tr>
</tbody>
</table>

- parameters (model-discrepancy approach): large variance
Application: Predictive capability assessment project

| regression methods | $||\mathbf{r}||_2$ | $\mu$: $Pr$ (q = 10) | $\mu$: $\hat{r}$ (q = 10) | $\mu$: $Pr$ (q = 100) | $\mu$: $\hat{r}$ (q = 100) | $\mu$: $Pr$ (q = 1000) | $\mu$: $\hat{r}$ (q = 1000) | $\mu$: $\hat{r}$ |
|--------------------|-------------------|--------------------|-------------------|--------------------|-------------------|--------------------|-------------------|-------------------|
| OLS: Linear        |                   |                    |                   |                    |                    |                    |                    |                    |
| OLS: Quadratic     |                   |                    |                   |                    |                    |                    |                    |                    |
| SVR: Linear        |                   |                    |                   |                    |                    |                    |                    |                    |
| SVR: RBF           |                   |                    |                   |                    |                    |                    |                    |                    |
| RF                 |                   |                    |                   |                    |                    |                    |                    |                    |
| k-NN               |                   |                    |                   |                    |                    |                    |                    |                    |
| ANN                |                   |                    |                   |                    |                    |                    |                    |                    |

$\log_{10}(1 - R^2)$

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

---

Nonlinear reduced-order modeling

Kevin Carlberg
**Introduction**

Parameterized Nonlinear Equations

**Experiments**

Regression Methods

- OLS: Linear
- OLS: Quadratic
- SVR: Linear
- SVR: RBF
- RF
- k-NN
- ANN

### Features

- 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

- y-displacement at A
  - $\log_{10}(1 - R^2)$
- radial displacement at B
  - $\log_{10}(1 - R^2)$

<table>
<thead>
<tr>
<th>Features</th>
<th>OLS: Linear</th>
<th>OLS: Quadratic</th>
<th>SVR: Linear</th>
<th>SVR: RBF</th>
<th>RF</th>
<th>k-NN</th>
<th>ANN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>r</td>
<td></td>
<td>^2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[μ; Pr] (q = 10)</td>
<td>[μ; Pr] (q = 10)</td>
<td>[μ; Pr] (q = 100)</td>
<td>[μ; Pr] (q = 1000)</td>
<td>[μ; Pr] (q = 1000)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[μ; ˆrg] (q = 10)</td>
<td>[μ; ˆrg] (q = 100)</td>
<td>[μ; ˆrg] (q = 1000)</td>
<td>[μ; ˆrg] (q = 1000)</td>
<td>[μ; ˆrg]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>µ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Fraction of variance unexplained (FVU) is 1
- SVR: RBF and MLP perform the best
- [μ; ˆrg] and [μ; Pr] we l l w i t h only q = 100 samples (compared to N = 278, 301)
**Application: Predictive capability assessment project**

<table>
<thead>
<tr>
<th>Regression Methods</th>
<th>y-displacement at A</th>
<th>radial displacement at B</th>
</tr>
</thead>
<tbody>
<tr>
<td>OLS: Linear</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OLS: Quadratic</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVR: Linear</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVR: RBF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>k-NN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ANN</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Features**
  - 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

<table>
<thead>
<tr>
<th>Regression Methods</th>
<th>0</th>
<th>-1</th>
<th>-2</th>
<th>-3</th>
<th>-4</th>
<th>-5</th>
</tr>
</thead>
<tbody>
<tr>
<td>OLS: Linear</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OLS: Quadratic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVR: Linear</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVR: RBF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>k-NN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ANN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Features</th>
<th>y-displacement at A</th>
<th>radial displacement at B</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\log_{10}(1 - R^2)$</td>
<td>$\log_{10}(1 - R^2)$</td>
</tr>
</tbody>
</table>

- 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

---

**Nonlinear reduced-order modeling**

Kevin Carlberg

44
Our research

‣ **accuracy**: LSPG projection

‣ **low cost**: sample mesh

‣ **low cost**: reduce temporal complexity

‣ **structure preservation**

‣ **robustness**: projection onto nonlinear manifolds

‣ **robustness**: $h$-adaptivity

‣ **certification**: machine learning error models
Questions?

\[ x^{ijk} = \]

\[ \Gamma \]

\[ \Omega \]

\[ error \]

\[ R^2 = 0.990 \]

support vector machine error prediction

\[ \mu \rightarrow \text{reduced-order model} \rightarrow q^n_{\text{ROM}} \]

\[ \rho^n \rightarrow \text{regression model} \rightarrow \hat{\delta}^n \]

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