Covariance balancing model reduction

Clancy Rowley

ICERM: Computational Learning for Model Reduction

PRINCETON UNIVERSITY

Projection methods

Start with a "full-order model"

$$\dot{x} = f(x, u)$$
$$y = g(x, u)$$

where $x \in \mathbb{R}^n$ is the state, u is an input, and y is an output.

Consider two r-dimensional subspaces of $\mathbb{R}^n$:

- Trial subspace $\mathcal{V} = \text{Range}(V)$
- Test subspace $\mathcal{W} = \text{Range}(W)$, $W^T V = I_r$

Reduced-order model (Petrov-Galerkin):

$$\dot{z} = W^T f(Vz, u)$$
$$y = g(Vz, u)$$

where $x = Vz$, $z \in \mathbb{R}^r$.

Reduced-order model

Reduced-order model is

$$\dot{z} = W^T f(Vz, u)$$
$$y = g(Vz, u)$$

Question: How should we choose V, W?

Can we do better than PCA?

Outline

Balanced truncation for linear systems

Covariance balancing reduction for nonlinear systems

Trajectory-based optimization of subspaces

Examples

Extensions using operator inference

Outline

Balanced truncation for linear systems

Covariance balancing reduction for nonlinear systems

Trajectory-based optimization of subspaces

Examples

Extensions using operator inference

Overview of balanced truncation

- Consider a linear system with state x, a control input u and output y:

$$\dot{x} = Ax + Bu$$
$$y = Cx$$

- We wish to determine a model with approximately the same input-output behavior, but smaller state dimension.
- Balanced truncation[1] is an excellent method for this
  - A priori error bound, close to the smallest possible
  - Computationally tractable, even for high-dimensional systems[2]

---

[1] BC Moore, IEEE Trans. Automat. Control, 1981

[2] Antoulas, 2005; Rowley, IJBC, 2005

ODE example

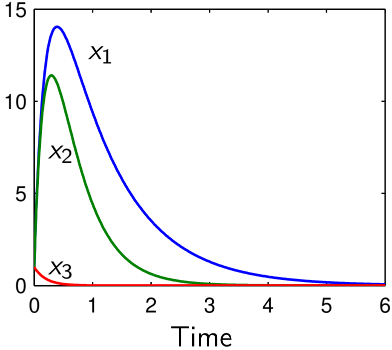
Consider the following ODE

$$\begin{aligned}
\dot{x}_1 &= -x_1 + 100x_3 + u \\
\dot{x}_2 &= -2x_2 + 100x_3 + u \\
\dot{x}_3 &= -5x_3 + u \\
y &= x_1 + x_2 + x_3
\end{aligned}$$

- u is the "input" (forcing term)
- y is the "output" (what we wish to capture with our model)
- Is there a 2-state model with nearly the same input-output behavior?

ODE response to an impulse

Consider the response to an impulsive input $u(t) = \delta(t)$
We're interested in modeling $y = x_1 + x_2 + x_3$



The state x_3 decays quickly, so we might think we can neglect it in a reduced-order model.

Naive reduced-order model

If we set $x_3 = 0$, the model becomes

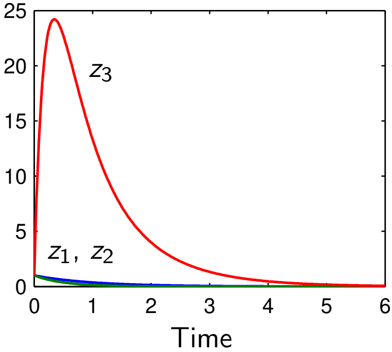
$$\begin{aligned}\dot{x}_1 &= -x_1 + u \\ \dot{x}_2 &= -2x_2 + u \\ y &= x_1 + x_2\end{aligned}$$

The impulse response decays monotonically (no transient growth)!
Clearly not a good model. What went wrong?

Look at sensitivity: adjoint system

The sensitivity of the output $y(t)$ to perturbations in the initial states are given by solving an adjoint system:

$$D_{x(0)}y(t) = (z_1(t), z_2(t), z_3(t))$$



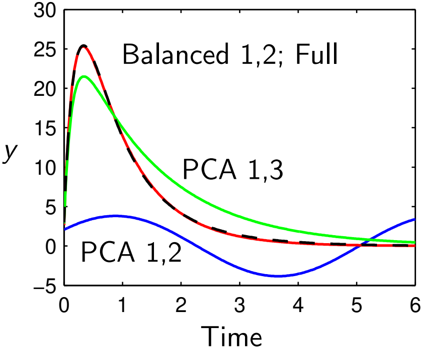
- The output is much more sensitive to perturbations in x_3 than to x_1 and x_2.
- But we neglected x_3 because it had small energy.

Balanced truncation

- Balanced truncation incorporates this sensitivity in an elegant and natural way.

Results for the ODE example

Impulse response for the projection onto two states:



- The first two PCA modes contain 99.97% of the energy, but projection onto these modes gives a poor model.
- Balanced truncation to two states matches the full model nearly perfectly.

What is balanced truncation doing?

- Most controllable states are those that are most easily excited by the input u.
  Quantified by a symmetric positive-definite matrix W_c:

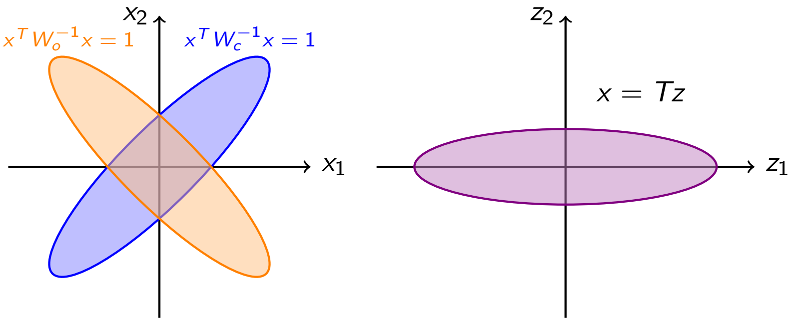
$$\text{Controllability}(x) = x^T W_c x$$

- Most observable states are those that excite the largest future outputs (with no input, $u = 0$).
  Quantified by a symmetric positive-definite matrix W_o:

$$\text{Observability}(x) = x^T W_o x$$

- Theorem: there is a change of coordinates in which W_c and W_o are equal and diagonal (under mild assumptions).
- Balanced truncation: change to these coordinates, and truncate the states that are least controllable/observable.

Geometric picture



An error bound

- Factor the controllability and observability matrices as

$$W_c = XX^T, \qquad W_o = YY^T$$

- Can prove that the error[3] between the full model and the reduced-order model with r states has a bound:

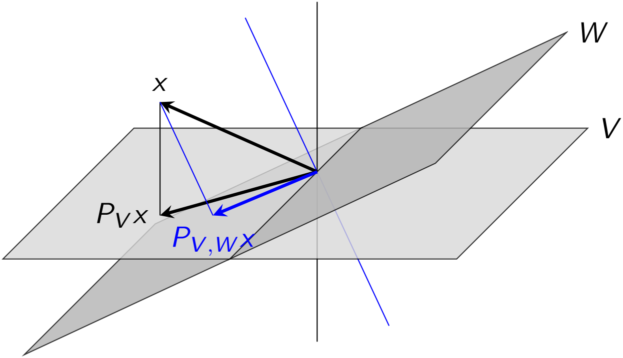
$$\text{Error} \leq 2(\sigma_{r+1} + \cdots + \sigma_n),$$

where σ_k are the singular values of $Y^T X$.

[3] in the operator norm induced by the 2-norm on signals

Balanced truncation as a projection of dynamics

- For a linear system, balanced truncation determines two subspaces, V and W



Summary of balanced truncation

- Effectively balances energy and sensitivity, and gives reduced-order models that are provably close to optimal
- Often significantly outperforms PCA, especially for "non-normal" systems with large transient energy growth
- Computationally tractable, even for high-dimensional systems
- Applies only to linear systems

Outline

Acknowledgements

Graduate students

- Sam Otto (Asst. Prof., Cornell)
- Alberto Padovan (Postdoc, UIUC)

Balancing for nonlinear systems

- We would like to find reduced-order models of a nonlinear system

$$x(t+1) = f(x(t), u(t))$$
$$y(t) = g(x(t))$$

- For linear systems, there are good methods (e.g., balanced truncation, H_2 optimal reduction)
- For nonlinear systems, the situation is much worse: available methods are
  - computationally intractable for high-dimensional systems
  - valid only in the neighborhood of an equilibrium point

Which coordinates should be retained?

- For now, we ignore the input:

$$
\begin{aligned}
x(t+1) &= f(x(t)) \\
y(t) &= g(x(t))
\end{aligned}
$$

- Consider a map from the current state x_0 to future outputs y, defined by

$$F(x_0) = (y(0), \ldots, y(L))$$

- A good set of coordinates $z = W^T x$ will allow us to approximate

$$F(x_0) \approx \tilde{F}(z_0)$$

Coordinates from projections

- Suppose V, W are $n \times r$ matrices such that $W^T V = I$. We have the decomposition

$$x = Vz + x_2, \qquad z = W^T x$$

- Given z, the optimal estimate for $F(x)$ (in the mean-square sense) is given by averaging over x_2:

$$\tilde{F}(z) = \mathbb{E}\big[F(Vz + x_2)\big]$$

- For a good estimate, want two things:
  - x_2 should have small variance
  - F should not be sensitive to variations in x_2.

Quantifying variance and sensitivity

- Variance is quantified using the state covariance

$$W_x = \mathbb{E}[xx^T].$$

- Sensitivity of F is quantified using the gradient covariance

$$W_g = \mathbb{E}\big[\nabla F(x) \nabla F(x)^T\big], \qquad \nabla F(x) = DF(x)^T$$

- Idea: change to coordinates in which W_x and W_g are equal and diagonal, and then truncate directions in which there is least variance and sensitivity.
- Observation: this is just like balanced truncation, with the covariance matrices W_x and W_g playing the role of controllability and observability.

Determining the optimal projection

- The covariance matrix W_x is easy to approximate by sampling
- The gradient covariance matrix W_g may be approximated by sampling an adjoint system
- Given these samples, the rank-r projection that balances these matrices is easily computed using singular value decomposition

We call this method Covariance Balancing Reduction using Adjoint Snapshots (CoBRAS) [4]

[4] SE Otto, A Padovan, and CW Rowley, SIAM J Scientific Computing, 45(5):A2325–A2355, 2023

An error bound

- Factor the covariance matrices as $W_x = XX^T$, $W_g = YY^T$
- Can prove that when x has Gaussian distribution,

$$\mathbb{E}\left[\|F(x) - \tilde{F}(Px)\|^2\right] \leq \sigma_{r+1}^2 + \cdots + \sigma_n^2,$$

where σ_k are the singular values of $Y^T X$.

Kernel method

- There is also a generalization of this to nonlinear projections, using a kernel method
- Lift the state and gradient vectors into a reproducing kernel Hilbert space (RKHS)
- Compute inner products implicitly via the kernel
- This allows us to extract rich nonlinear features from the system

Outline

Optimizing subspaces

- We can determine an even better choice of subspaces for projection by an iterative optimization

Trajectory-based optimization

- Consider observations $\{y_0, \dots, y_{L-1}\}$ along a trajectory from the full model.
- For given subspaces V and W, compute the corresponding observations $\{\hat{y}_0, \dots, \hat{y}_{L-1}\}$ from the reduced-order model.
- Let $L_y : \mathbb{R}^{\dim y} \to [0, \infty)$ be a smooth loss function for the predicted system outputs. We'd like to minimize

$$J(V, W) = \frac{1}{L} \sum_{l=0}^{L-1} L_y(\hat{y}_l - y_l).$$

- But we need to ensure that the subspace pairs (V, W) satisfy the non-orthogonality condition, so we introduce a regularization $\rho(V, W)$ that enforces this constraint, and define the objective

$$\boxed{J(V, W) = \frac{1}{L} \sum_{l=0}^{L-1} L_y(\hat{y}_l - y_l) + \gamma \rho(V, W).}$$

The overall method

We call this method Trajectory-based Optimization for Oblique Projection (TrOOP)

Solution of the optimization problem

- One optimizes over a manifold (two copies of the Grassmann manifold) using a geometric conjugate gradient algorithm [5] [6]
- The gradient is computed using an adjoint sensitivity method.
- This entails solving a linear ODE with the same dimension as the reduced-order model backwards in time.

[5] Absil et al., "Optimization Algorithms on Matrix Manifolds", 2008

[6] H. Sato, "A Dai–Yuan-type Riemannian conjugate gradient method with the weak Wolfe conditions", 2016

Outline

Two methods

We've described two different methods:

- Covariance Balanced Reduction using Adjoint Snapshots (CoBRAS)
- Trajectory-based Optimization for Oblique Projection (TrOOP)

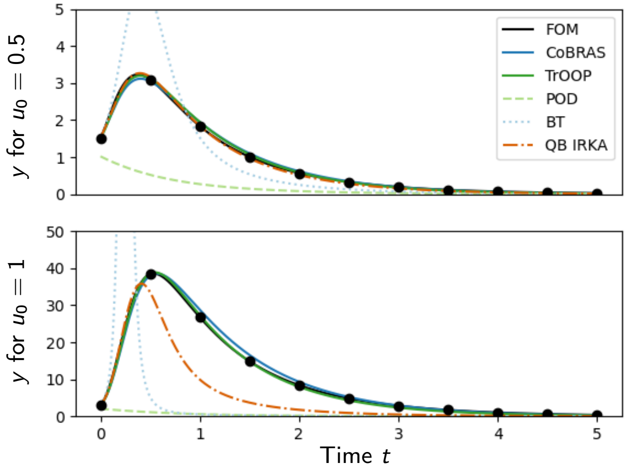
In practice, we can use CoBRAS to provide an initial guess for the optimization problem in TrOOP.

A challenging model problem

$$
\begin{aligned}
\dot{x}_1 &= -x_1 + 20x_1x_3 + u \\
\dot{x}_2 &= -2x_2 + 20x_2x_3 + u \\
\dot{x}_3 &= -5x_3 + u \\
y &= x_1 + x_2 + x_3,
\end{aligned}
$$

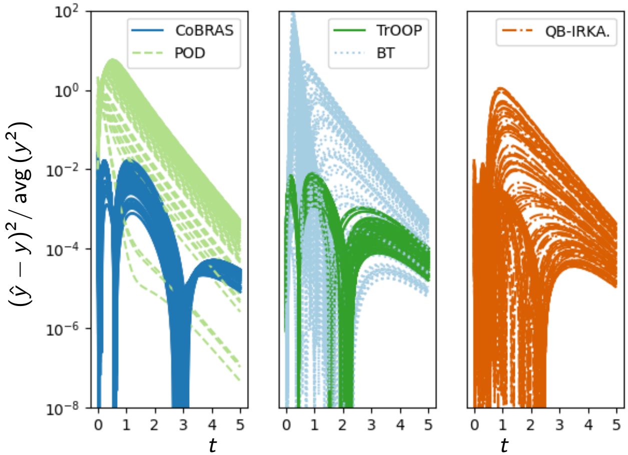
- The state x_3 is dynamically important, but remains small compared with x_1 and x_2 due to its fast decay rate.
- The linearized dynamics do not capture the system's behavior away from the origin, which exhibits transient growth.
- We seek 2-dimensional reduced-order models.

Training trajectories



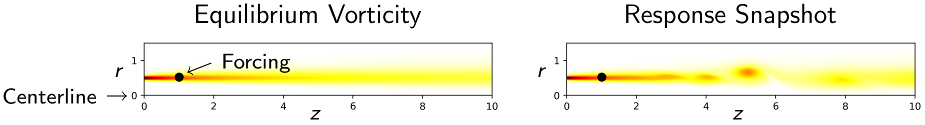
The training data consists of 22 sample points (black dots), from trajectories with impulsive inputs $u(t) = u_0\delta(t)$, with $u_0 = 0.5$ and 1.0.

Testing performance



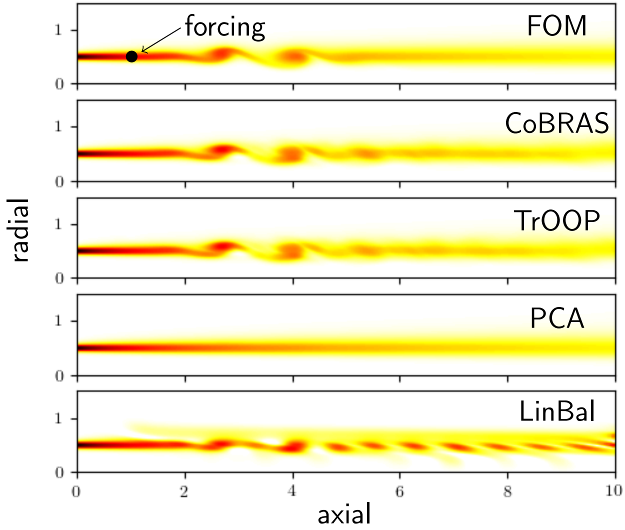
We tested our optimized Petrov-Galerkin model on 100 nonlinear impulse response trajectories with magnitudes drawn uniformly at random from the interval $[0, 1]$.

Axisymmetric jet flow example

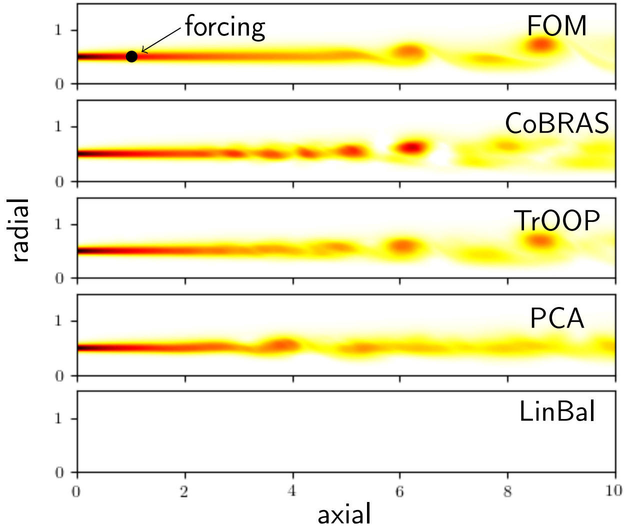


- Incompressible, axisymmetric jet flow
- Reynolds number 1000
- Full model: 100,000 states
- Actuation: body force in radial direction
- We seek 40-dimensional reduced order models capable of predicting the impulse response of the flow for a range of impulse amplitudes

Vorticity predictions for jet flow, $t = 5$

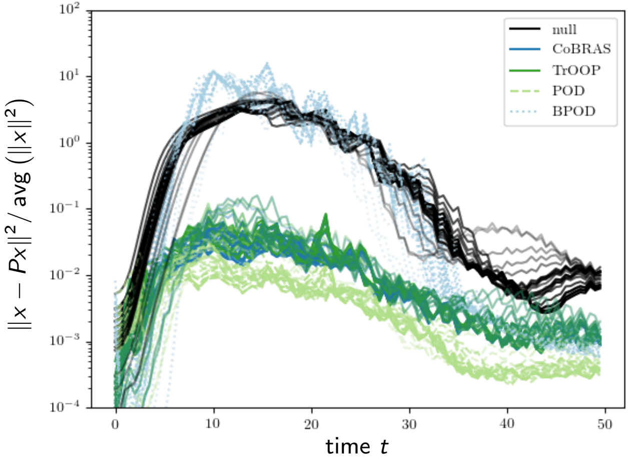


Vorticity predictions for jet flow, $t = 20$



Projection error

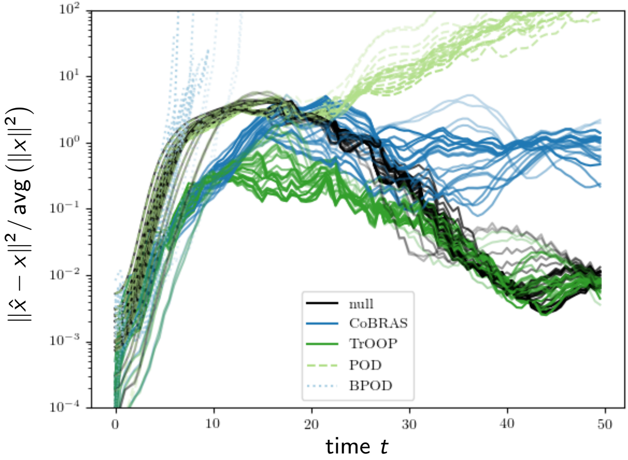
Project each testing trajectory onto 40-dimensional subspace.
The "null" projection means $Px = 0$.



PCA has the best projection error (as it must)

Forecasting error

Now, consider forecasting using the reduced-order model



CoBRAS and TrOOP both significantly outperform other methods, and PCA is no better than the "null" model.

Outline

Balanced truncation for linear systems

Covariance balancing reduction for nonlinear systems

Trajectory-based optimization of subspaces

Examples

Extensions using operator inference

Drawbacks

- Some drawbacks of these approaches:
  - They are intrusive: require knowledge of the full model
  - The require adjoint simulations, which may not be available

Recent work by Alberto Padovan addresses both of these, using operator inference.

Leveraging operator inference

An exciting recent paper (arXiv:2401.01290):

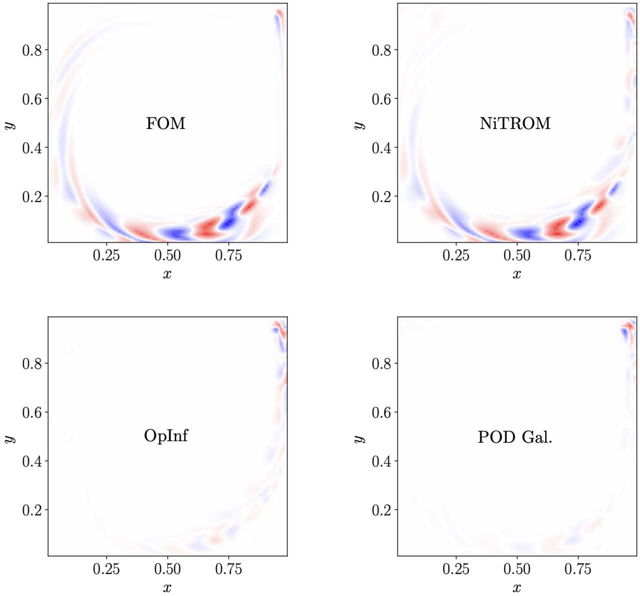
Data-driven model reduction via non-intrusive optimization of projection operators and reduced-order dynamics

Alberto Padovan*, Blaine Vollmer*, and Daniel J. Bodony*

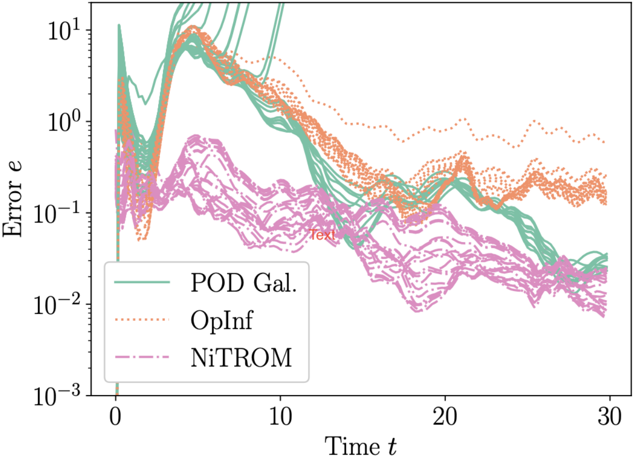
Abstract. Computing reduced-order models using non-intrusive methods is particularly attractive for systems that are simulated using black-box solvers. However, obtaining accurate data-driven models can be challenging, especially if the underlying systems exhibit large-amplitude transient growth. Although these systems may evolve near a low-dimensional subspace that can be easily identified using standard techniques such as Proper Orthogonal Decomposition (POD), computing accurate models often requires projecting the state onto this subspace via a non-orthogonal projection. While appropriate oblique projection operators can be computed using intrusive techniques that leverage the form of the underlying governing equations, purely data-driven methods currently tend to achieve dimensionality reduction via orthogonal projections, and this can lead to models with poor predictive accuracy. In this paper, we address this issue by introducing a non-intrusive framework designed to simultaneously identify oblique projection operators and reduced-order dynamics. In

- Idea:
  - Guess subspaces for projection
  - Determine an approximate model using operator inference
  - Use the adjoint of the inferred model to compute gradients needed for gradient descent
  - Update subspaces and iterate

Example: lid-driven cavity flow



Training error for lid-driven cavity flow



Takeaways

- PCA often does not give the best subspaces for reduced-order models
- Can generalize balanced truncation to nonlinear systems by balancing state covariance and gradient covariance
- Gradient covariance matrices computed efficiently from adjoint simulations
- Iterative method for further refining these subspaces, using loss function based on trajectories
- These methods are intrusive (require knowledge of the full dynamics)
- However, they play very nicely with non-intrusive methods (Operator Inference)

Acknowledgements and papers

- Alberto Padovan (Postdoc, UIUC)
- Sam Otto (Asst. Prof, Cornell)
- Funding from Air Force Office of Scientific Research

- Papers
  - Covariance balancing: SE Otto, A Padovan, and CW Rowley, SIAM J Scientific Computing, 45(5):A2325–A2355, 2023
  - Trajectory-based optimization: SE Otto, A Padovan and CW Rowley, SIAM J Scientific Computing 44(3):A1681–A1702, 2022.