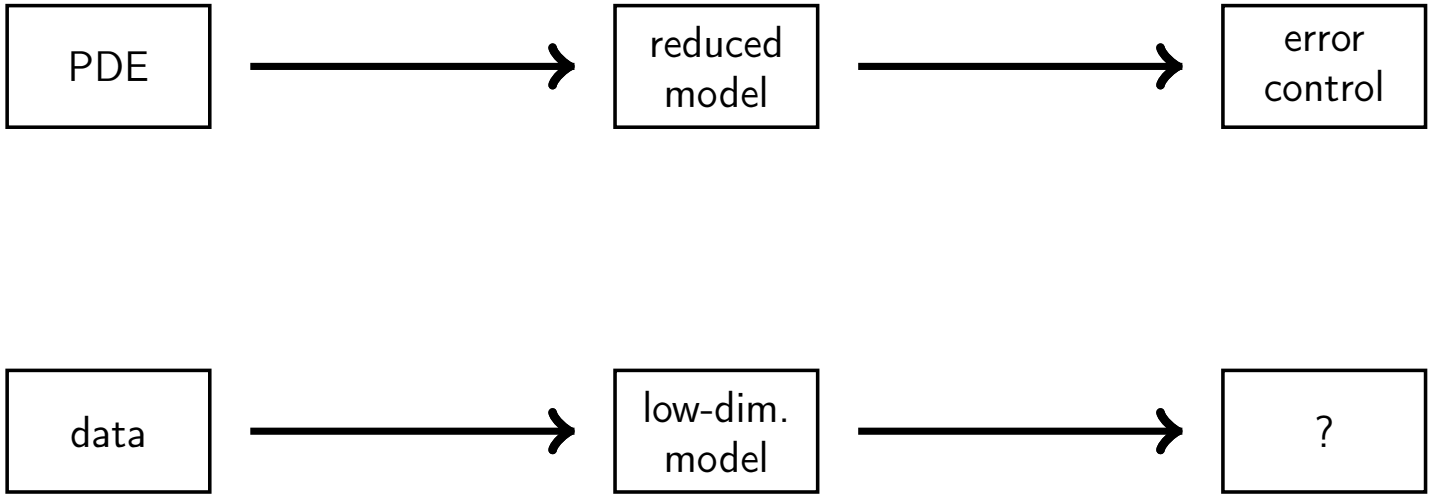
Sampling low-dimensional Markovian dynamics for learning certified reduced models from data

Wayne Isaac Tan Uy and Benjamin Peherstorfer
Courant Institute of Mathematical Sciences, New York University

February 2020

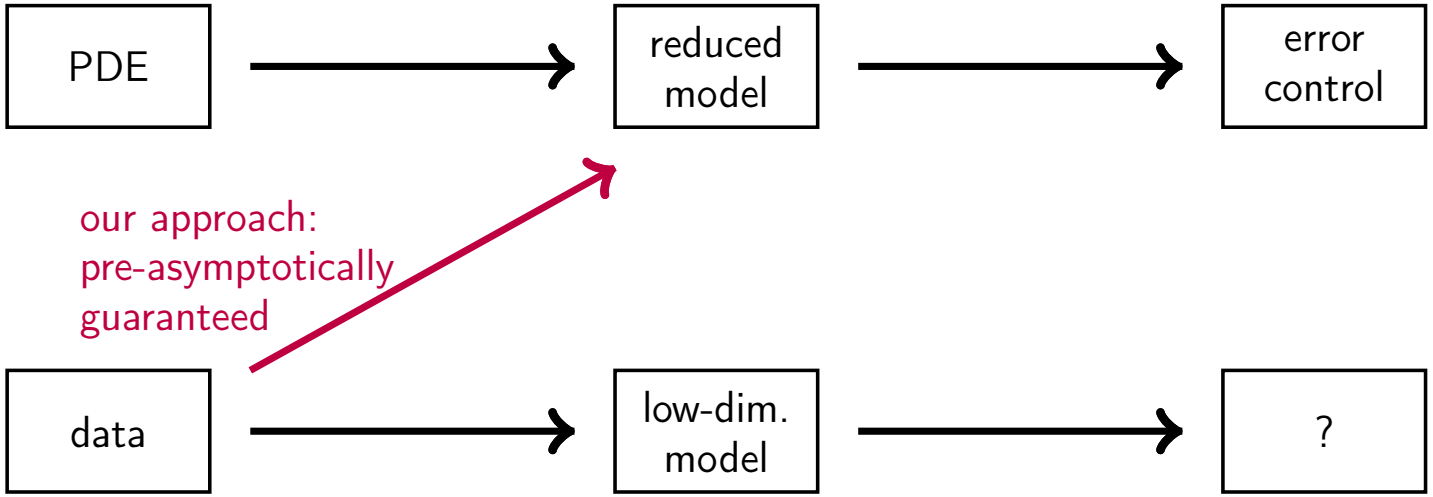
Learning dynamical-system models from data



Learn low-dimensional model from data of dynamical system

- Interpretable
- System & control theory
- Fast predictions
- Guarantees for finite data

Recovering reduced models from data



Learn low-dimensional model from data of dynamical system

- Interpretable
- System & control theory
- Fast predictions
- Guarantees for finite data

Learn reduced model from trajectories of high-dim. system

- Recover *exactly* and *pre-asymptotically* reduced models from data
- Then build on rich theory of model reduction to establish error control

Intro: Polynomial nonlinear terms

Models with polynomial nonlinear terms

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t;\boldsymbol{\mu}) =& \boldsymbol{f}(\boldsymbol{x}(t;\boldsymbol{\mu}),\boldsymbol{u}(t);\boldsymbol{\mu})\\ =& \sum_{i=1}^{\ell}\boldsymbol{A}_i(\boldsymbol{\mu})\boldsymbol{x}^i(t;\boldsymbol{\mu}) + \boldsymbol{B}(\boldsymbol{\mu})\boldsymbol{u}(t)\end{aligned}$$

- Polynomial degree $\ell \in \mathbb{N}$
- Kronecker product $\boldsymbol{x}^i(t;\boldsymbol{\mu}) = \bigotimes_{j=1}^{i}\boldsymbol{x}(t;\boldsymbol{\mu})$
- Operators $\boldsymbol{A}_i(\boldsymbol{\mu}) \in \mathbb{R}^{N\times N^i}$ for $i = 1,\dots,\ell$
- Input operator $\boldsymbol{B}(\boldsymbol{\mu}) \in \mathbb{R}^{N\times p}$

Lifting and transformations

- Lift general nonlinear systems to quadratic-bilinear ones [Gu, 2011], [Benner, Breiten, 2015], [Benner, Goyal, Gugercin, 2018], [Kramer, Willcox, 2019], [Swischuk, Kramer, Huang, Willcox, 2019], [Qian, Kramer, P., Willcox, 2019]
- Koopman lifts nonlinear systems to infinite linear systems [Rowley et al, 2009], [Schmid, 2010]

Intro: Beyond polynomial terms (nonintrusive)

arXiv.org > math > arXiv:1912.08177

Search... All fields Search

Help | Advanced Search

Mathematics > Numerical Analysis

Lift & Learn: Physics-informed machine learning for large-scale nonlinear dynamical systems

Elizabeth Qian, Boris Kramer, Benjamin Peherstorfer, Karen Willcox

(Submitted on 17 Dec 2019 (v1), last revised 23 Dec 2019 (this version, v2))

We present Lift & Learn, a physics-informed method for learning low-dimensional models for large-scale dynamical systems. The method exploits knowledge of a system's governing equations to identify a coordinate transformation in which the system dynamics have quadratic structure. This transformation is called a lifting map because it often adds auxiliary variables to the system state. The lifting map is applied to data obtained by evaluating a model for the original nonlinear system. This lifted data is projected onto its leading principal components, and low-dimensional linear and quadratic matrix operators are fit to the lifted reduced data using a least-squares operator inference procedure. Analysis of our method shows that the Lift & Learn models are able to capture the system physics in the lifted coordinates at least as accurately as traditional intrusive model reduction approaches. This preservation of system physics makes the Lift & Learn models robust to changes in inputs. Numerical experiments on the FitzHugh-Nagumo neuron activation model and the compressible Euler equations demonstrate the generalizability of our model.

Download:

- PDF
- Other formats

(license)

Current browse context:

math.NA

< prev | next >

new | recent | 1912

Change to browse by:

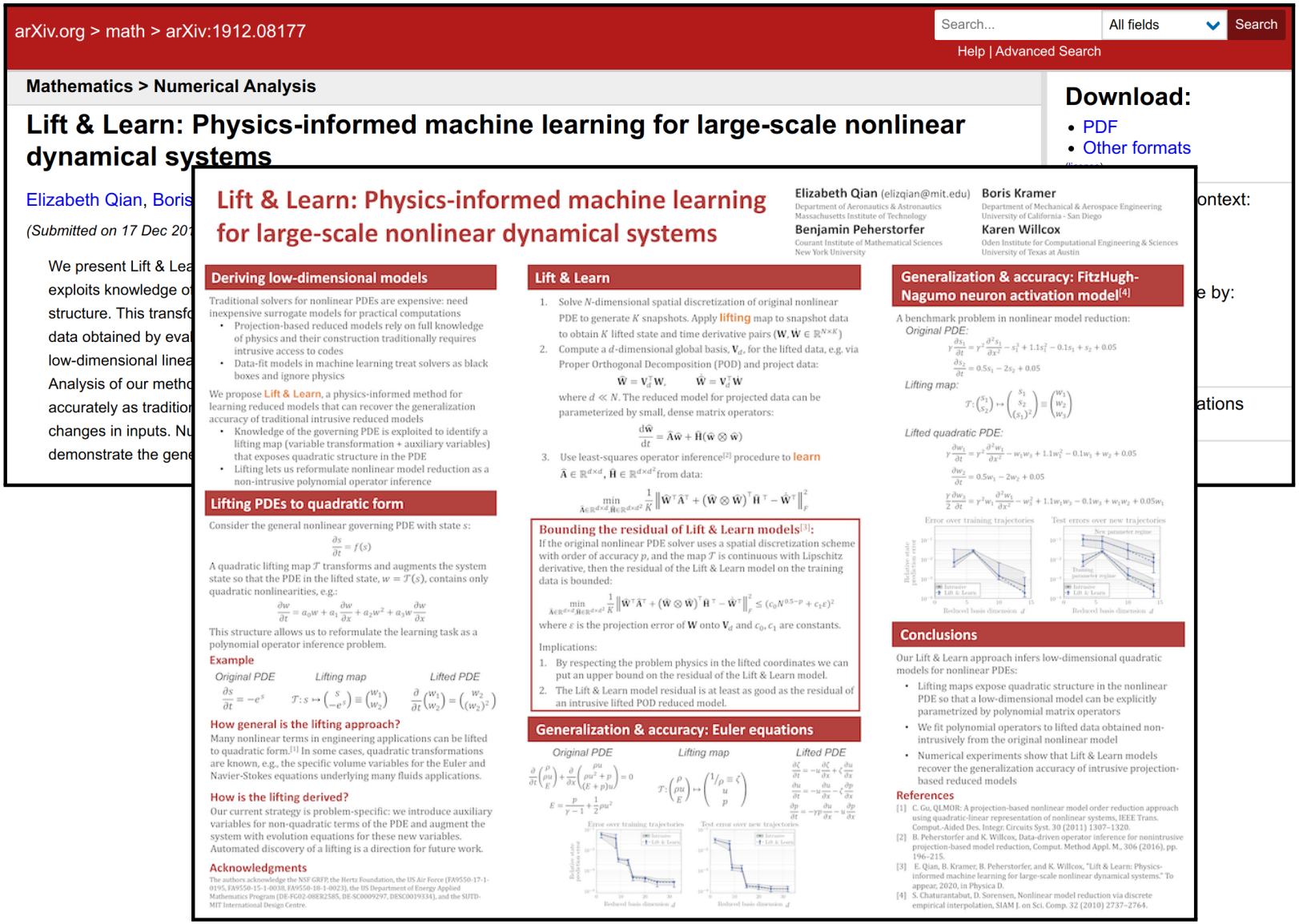
cs
cs.LG
cs.NA
math

References & Citations

- NASA ADS

Export citation
Google Scholar

Intro: Beyond polynomial terms (nonintrusive)



arXiv.org > math > arXiv:1912.08177

Search... All fields Search

Help | Advanced Search

Mathematics > Numerical Analysis

Lift & Learn: Physics-informed machine learning for large-scale nonlinear dynamical systems

Elizabeth Qian, Boris

(Submitted on 17 Dec 201

We present Lift & Lea
exploits knowledge o
structure. This transfo
data obtained by eval
low-dimensional linea
Analysis of our metho
accurately as traditio
changes in inputs. Nu
demonstrate the gene

Download:

- PDF
- Other formats

ontext:

e by:

ations

Lift & Learn: Physics-informed machine learning for large-scale nonlinear dynamical systems

Elizabeth Qian (elizqian@mit.edu)
Department of Aeronautics & Astronautics
Massachusetts Institute of Technology

Boris Kramer
Department of Mechanical & Aerospace Engineering
University of California - San Diego

Benjamin Peherstorfer
Courant Institute of Mathematical Sciences
New York University

Karen Willcox
Oden Institute for Computational Engineering & Sciences
University of Texas at Austin

Deriving low-dimensional models

Traditional solvers for nonlinear PDEs are expensive: need inexpensive surrogate models for practical computations

- Projection-based reduced models rely on full knowledge of physics and their construction traditionally requires intrusive access to codes
- Data-fit models in machine learning treat solvers as black boxes and ignore physics

We propose Lift & Learn, a physics-informed method for learning reduced models that can recover the generalization accuracy of traditional intrusive reduced models

- Knowledge of the governing PDE is exploited to identify a lifting map (variable transformation + auxiliary variables) that exposes quadratic structure in the PDE
- Lifting lets us reformulate nonlinear model reduction as a non-intrusive polynomial operator inference

Lifting PDEs to quadratic form

Consider the general nonlinear governing PDE with state s:

$$\frac{\partial s}{\partial t} = f(s)$$

A quadratic lifting map $\mathcal{T}$ transforms and augments the system state so that the PDE in the lifted state, $w = \mathcal{T}(s)$, contains only quadratic nonlinearities, e.g.:

$$\frac{\partial w}{\partial t} = a_0 w + a_1 \frac{\partial w}{\partial x} + a_2 w^2 + a_3 w \frac{\partial w}{\partial x}$$

This structure allows us to reformulate the learning task as a polynomial operator inference problem.

Example

Original PDE	Lifting map	Lifted PDE
$\frac{\partial s}{\partial t} = -e^s$	$\mathcal{T}: s \mapsto \begin{pmatrix} s \\ -e^s \end{pmatrix} \equiv \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$	$\frac{\partial}{\partial t}\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} w_2 \\ (w_2)^2 \end{pmatrix}$

How general is the lifting approach?

Many nonlinear terms in engineering applications can be lifted to quadratic form.[1] In some cases, quadratic transformations are known, e.g., the specific volume variables for the Euler and Navier-Stokes equations underlying many fluids applications.

How is the lifting derived?

Our current strategy is problem-specific: we introduce auxiliary variables for non-quadratic terms of the PDE and augment the system with evolution equations for these new variables. Automated discovery of a lifting is a direction for future work.

Acknowledgments

The authors acknowledge the NSF GRFP, the Hertz Foundation, the US Air Force (FA9550-17-1-0195, FA9550-15-1-0038, FA9550-18-1-0023), the US Department of Energy Applied Mathematics Program (DE-FG02-08ER2585, DE-SC0009297, DESC0019334), and the SUTD-MIT International Design Centre.

Lift & Learn

1. Solve N-dimensional spatial discretization of original nonlinear PDE to generate K snapshots. Apply lifting map to snapshot data to obtain K lifted state and time derivative pairs $(\mathbf{W}, \dot{\mathbf{W}} \in \mathbb{R}^{N \times K})$
2. Compute a d-dimensional global basis, $\mathbf{V}_d$, for the lifted data, e.g. via Proper Orthogonal Decomposition (POD) and project data:
$$\hat{\mathbf{W}} = \mathbf{V}_d^T \mathbf{W}, \qquad \dot{\hat{\mathbf{W}}} = \mathbf{V}_d^T \dot{\mathbf{W}}$$
where $d \ll N$. The reduced model for projected data can be parameterized by small, dense matrix operators:
$$\frac{d\hat{\mathbf{w}}}{dt} = \hat{\mathbf{A}}\hat{\mathbf{w}} + \hat{\mathbf{H}}(\hat{\mathbf{w}} \otimes \hat{\mathbf{w}})$$
3. Use least-squares operator inference[2] procedure to learn $\hat{\mathbf{A}} \in \mathbb{R}^{d \times d}$, $\hat{\mathbf{H}} \in \mathbb{R}^{d \times d^2}$ from data:
$$\min_{\hat{\mathbf{A}} \in \mathbb{R}^{d \times d}, \hat{\mathbf{H}} \in \mathbb{R}^{d \times d^2}} \frac{1}{K} \left\| \dot{\hat{\mathbf{W}}}^\top \hat{\mathbf{A}}^\top + (\hat{\mathbf{W}} \otimes \hat{\mathbf{W}})^\top \hat{\mathbf{H}}^\top - \dot{\hat{\mathbf{W}}}^\top \right\|_F^2$$

Bounding the residual of Lift & Learn models[3]:

If the original nonlinear PDE solver uses a spatial discretization scheme with order of accuracy p, and the map $\mathcal{T}$ is continuous with Lipschitz derivative, then the residual of the Lift & Learn model on the training data is bounded:

$$\min_{\hat{\mathbf{A}} \in \mathbb{R}^{d \times d}, \hat{\mathbf{H}} \in \mathbb{R}^{d \times d^2}} \frac{1}{K} \left\| \dot{\hat{\mathbf{W}}}^\top \hat{\mathbf{A}}^\top + (\hat{\mathbf{W}} \otimes \hat{\mathbf{W}})^\top \hat{\mathbf{H}}^\top - \dot{\hat{\mathbf{W}}}^\top \right\|_F^2 \leq (c_0 N^{0.5-p} + c_1 \varepsilon)^2$$

where ε is the projection error of $\mathbf{W}$ onto $\mathbf{V}_d$ and c_0, c_1 are constants.

Implications:

1. By respecting the problem physics in the lifted coordinates we can put an upper bound on the residual of the Lift & Learn model.
2. The Lift & Learn model residual is at least as good as the residual of an intrusive lifted POD reduced model.

Generalization & accuracy: Euler equations

Original PDE	Lifting map	Lifted PDE
$\frac{\partial}{\partial t}\begin{pmatrix} \rho \\ \rho u \\ E \end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} \rho u \\ \rho u^2 + p \\ (E+p)u \end{pmatrix} = 0$, $E = \frac{p}{\gamma - 1} + \frac{1}{2}\rho u^2$	$\mathcal{T}: \begin{pmatrix} \rho \\ \rho u \\ E \end{pmatrix} \mapsto \begin{pmatrix} 1/\rho \equiv \zeta \\ u \\ p \end{pmatrix}$	$\frac{\partial \zeta}{\partial t} = -u\frac{\partial \zeta}{\partial x} + \zeta\frac{\partial u}{\partial x}$, $\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} - \zeta\frac{\partial p}{\partial x}$, $\frac{\partial p}{\partial t} = -\gamma p\frac{\partial u}{\partial x} - u\frac{\partial p}{\partial x}$

Generalization & accuracy: FitzHugh-Nagumo neuron activation model[4]

A benchmark problem in nonlinear model reduction:

Original PDE:

$$\gamma\frac{\partial s_1}{\partial t} = \gamma^2\frac{\partial^2 s_1}{\partial x^2} - s_1^2 + 1.1s_1^2 - 0.1s_1 + s_2 + 0.05$$
$$\frac{\partial s_2}{\partial t} = 0.5s_1 - 2s_2 + 0.05$$

Lifting map:

$$\mathcal{T}: \begin{pmatrix} s_1 \\ s_2 \end{pmatrix} \mapsto \begin{pmatrix} s_1 \\ s_2 \\ (s_1)^2 \end{pmatrix} \equiv \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix}$$

Lifted quadratic PDE:

$$\gamma\frac{\partial w_1}{\partial t} = \gamma^2\frac{\partial^2 w_1}{\partial x^2} - w_1 w_3 + 1.1w_1^2 - 0.1w_1 + w_2 + 0.05$$
$$\frac{\partial w_2}{\partial t} = 0.5w_1 - 2w_2 + 0.05$$
$$\frac{\gamma}{2}\frac{\partial w_3}{\partial t} = \gamma^2 w_1\frac{\partial^2 w_1}{\partial x^2} - w_1^2 + 1.1w_1 w_3 - 0.1w_3 + w_1 w_2 + 0.05w_1$$

Conclusions

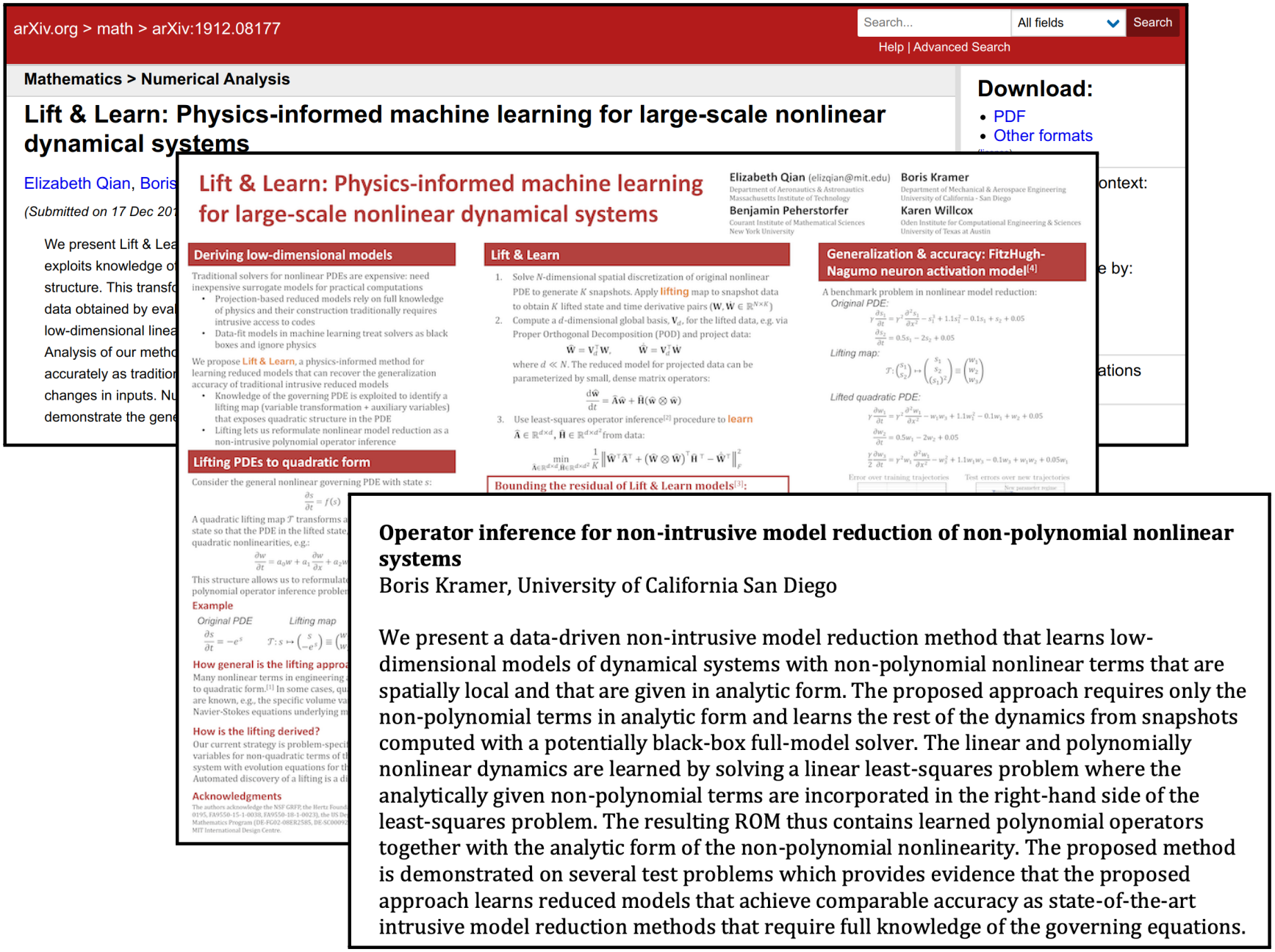
Our Lift & Learn approach infers low-dimensional quadratic models for nonlinear PDEs:

- Lifting maps expose quadratic structure in the nonlinear PDE so that a low-dimensional model can be explicitly parametrized by polynomial matrix operators
- We fit polynomial operators to lifted data obtained non-intrusively from the original nonlinear model
- Numerical experiments show that Lift & Learn models recover the generalization accuracy of intrusive projection-based reduced models

References

[1] C. Gu, QLMOR: A projection-based nonlinear model order reduction approach using quadratic-linear representation of nonlinear systems, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 30 (2011) 1307–1320.
[2] B. Peherstorfer and K. Willcox, Data-driven operator inference for nonintrusive projection-based model reduction, Comput. Method Appl. M., 306 (2016), pp. 196–215.
[3] E. Qian, B. Kramer, B. Peherstorfer, and K. Willcox, "Lift & Learn: Physics-informed machine learning for large-scale nonlinear dynamical systems." To appear, 2020, in Physica D.
[4] S. Chaturantabut, D. Sorensen, Nonlinear model reduction via discrete empirical interpolation, SIAM J. on Sci. Comp. 32 (2010) 2737–2764.

Intro: Beyond polynomial terms (nonintrusive)



Operator inference for non-intrusive model reduction of non-polynomial nonlinear systems

Boris Kramer, University of California San Diego

We present a data-driven non-intrusive model reduction method that learns low-dimensional models of dynamical systems with non-polynomial nonlinear terms that are spatially local and that are given in analytic form. The proposed approach requires only the non-polynomial terms in analytic form and learns the rest of the dynamics from snapshots computed with a potentially black-box full-model solver. The linear and polynomially nonlinear dynamics are learned by solving a linear least-squares problem where the analytically given non-polynomial terms are incorporated in the right-hand side of the least-squares problem. The resulting ROM thus contains learned polynomial operators together with the analytic form of the non-polynomial nonlinearity. The proposed method is demonstrated on several test problems which provides evidence that the proposed approach learns reduced models that achieve comparable accuracy as state-of-the-art intrusive model reduction methods that require full knowledge of the governing equations.

Intro: Parametrized systems

Consider time-invariant system with polynomial nonlinear terms

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t;\boldsymbol{\mu}) =& \boldsymbol{f}(\boldsymbol{x}(t;\boldsymbol{\mu}),\boldsymbol{u}(t);\boldsymbol{\mu})\\ =& \sum_{i=1}^{\ell}\boldsymbol{A}_i(\boldsymbol{\mu})\boldsymbol{x}^i(t;\boldsymbol{\mu}) + \boldsymbol{B}(\boldsymbol{\mu})\boldsymbol{u}(t)\end{aligned}$$

Parameters

- Infer models $\hat{\boldsymbol{f}}(\cdot,\cdot;\boldsymbol{\mu}_1),\ldots,\hat{\boldsymbol{f}}(\cdot,\cdot;\boldsymbol{\mu}_M)$ at parameters

$$\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_M \in \mathcal{D}$$

- For new $\boldsymbol{\mu}\in\mathcal{D}$, interpolate operators of [Amsallem et al., 2008], [Degroote et al., 2010]

$$\hat{\boldsymbol{f}}(\boldsymbol{\mu}_1),\ldots,\hat{\boldsymbol{f}}(\boldsymbol{\mu}_M)$$

Trajectories

$$\boldsymbol{X} = [\boldsymbol{x}_1,\ldots,\boldsymbol{x}_K] \in \mathbb{R}^{N\times K}$$
$$\boldsymbol{U} = [\boldsymbol{u}_1,\ldots,\boldsymbol{u}_K] \in \mathbb{R}^{p\times K}$$

Intro: Parametrized systems

Consider time-invariant system with polynomial nonlinear terms

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t) =& \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)) \\ =& \sum_{i=1}^{\ell} \boldsymbol{A}_i \boldsymbol{x}^i(t) + \boldsymbol{B}\boldsymbol{u}(t)\end{aligned}$$

Parameters

- Infer models $\hat{\boldsymbol{f}}(\cdot, \cdot; \boldsymbol{\mu}_1), \ldots, \hat{\boldsymbol{f}}(\cdot, \cdot; \boldsymbol{\mu}_M)$ at parameters

$$\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_M \in \mathcal{D}$$

- For new $\boldsymbol{\mu} \in \mathcal{D}$, interpolate operators of [Amsallem et al., 2008], [Degroote et al., 2010]

$$\hat{\boldsymbol{f}}(\boldsymbol{\mu}_1), \ldots, \hat{\boldsymbol{f}}(\boldsymbol{\mu}_M)$$

Trajectories

$$\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_K] \in \mathbb{R}^{N \times K}$$
$$\boldsymbol{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_K] \in \mathbb{R}^{p \times K}$$

Intro: Parametrized systems

Consider time-invariant system with polynomial nonlinear terms

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) \\ &= \sum_{i=1}^{\ell} \boldsymbol{A}_i \boldsymbol{x}_k^i + \boldsymbol{B}\boldsymbol{u}_k\,, \qquad k = 0, \ldots, K-1 \end{aligned}$$

Parameters

- Infer models $\hat{\boldsymbol{f}}(\cdot, \cdot; \boldsymbol{\mu}_1), \ldots, \hat{\boldsymbol{f}}(\cdot, \cdot; \boldsymbol{\mu}_M)$ at parameters

$$\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_M \in \mathcal{D}$$

- For new $\boldsymbol{\mu} \in \mathcal{D}$, interpolate operators of [Amsallem et al., 2008], [Degroote et al., 2010]

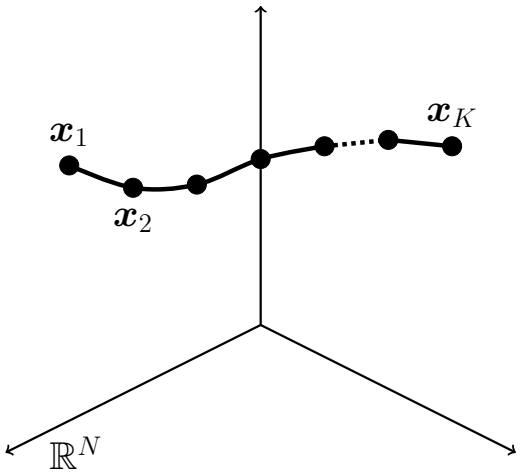
$$\hat{\boldsymbol{f}}(\boldsymbol{\mu}_1), \ldots, \hat{\boldsymbol{f}}(\boldsymbol{\mu}_M)$$

Trajectories

$$\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_K] \in \mathbb{R}^{N \times K}$$
$$\boldsymbol{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_K] \in \mathbb{R}^{p \times K}$$

Intro: Classical (intrusive) model reduction

Given full model $\boldsymbol{f}$, construct reduced $\tilde{\boldsymbol{f}}$ via projection



1. Construct n-dim. basis $\boldsymbol{V} = [\boldsymbol{v}_1, \dots, \boldsymbol{v}_n] \in \mathbb{R}^{N \times n}$

- Proper orthogonal decomposition (POD)
- Interpolatory model reduction
- Reduced basis method (RBM), ...

2. Project full-model operators $\boldsymbol{A}_1, \dots, \boldsymbol{A}_\ell, \boldsymbol{B}$ onto reduced space, e.g.,

$$\tilde{\boldsymbol{A}}_i = \underbrace{\boldsymbol{V}^T \overbrace{\boldsymbol{A}_i}^{N \times N^i} (\boldsymbol{V} \otimes \cdots \otimes \boldsymbol{V})}_{n \times n^i}, \qquad \tilde{\boldsymbol{B}} = \underbrace{\boldsymbol{V}^T \overbrace{\boldsymbol{B}}^{N \times p}}_{n \times p}$$

3. Construct reduced model

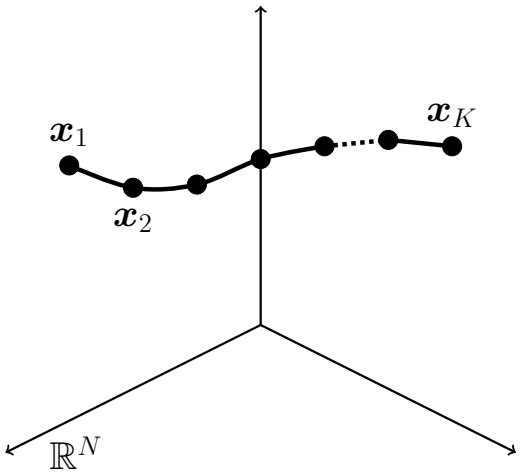
$$\tilde{\boldsymbol{x}}_{k+1} = \tilde{\boldsymbol{f}}(\tilde{\boldsymbol{x}}_k, \boldsymbol{u}_k) = \sum_{i=1}^{\ell} \tilde{\boldsymbol{A}}_i \tilde{\boldsymbol{x}}_k^i + \tilde{\boldsymbol{B}} \boldsymbol{u}_k \,, \qquad k = 0, \dots, K-1$$

with $n \ll N$ and $\|\boldsymbol{V}\tilde{\boldsymbol{x}}_k - \boldsymbol{x}_k\|$ small in appropriate norm

[Rozza, Huynh, Patera, 2007], [Benner, Gugercin, Willcox, 2015]

Intro: Classical (intrusive) model reduction

Given full model f, construct reduced $\tilde{f}$ via projection



1. **Construct n-dim. basis $V = [v_1, \dots, v_n] \in \mathbb{R}^{N \times n}$**
   - Proper orthogonal decomposition (POD)
   - Interpolatory model reduction
   - Reduced basis method (RBM), ...

2. **Project full-model operators $A_1, \dots, A_\ell, B$ onto reduced space, e.g.,**

$$\tilde{A}_i = \underbrace{V^T \overbrace{A_i}^{N \times N^i} (V \otimes \cdots \otimes V)}_{n \times n^i}, \qquad \tilde{B} = \underbrace{V^T \overbrace{B}^{N \times p}}_{n \times p}$$

3. **Construct reduced model**

$$\tilde{x}_{k+1} = \tilde{f}(\tilde{x}_k, u_k) = \sum_{i=1}^{\ell} \tilde{A}_i \tilde{x}_k^i + \tilde{B} u_k \,, \qquad k = 0, \dots, K-1$$

with $n \ll N$ and $\|V\tilde{x}_k - x_k\|$ small in appropriate norm

[Rozza, Huynh, Patera, 2007], [Benner, Gugercin, Willcox, 2015]

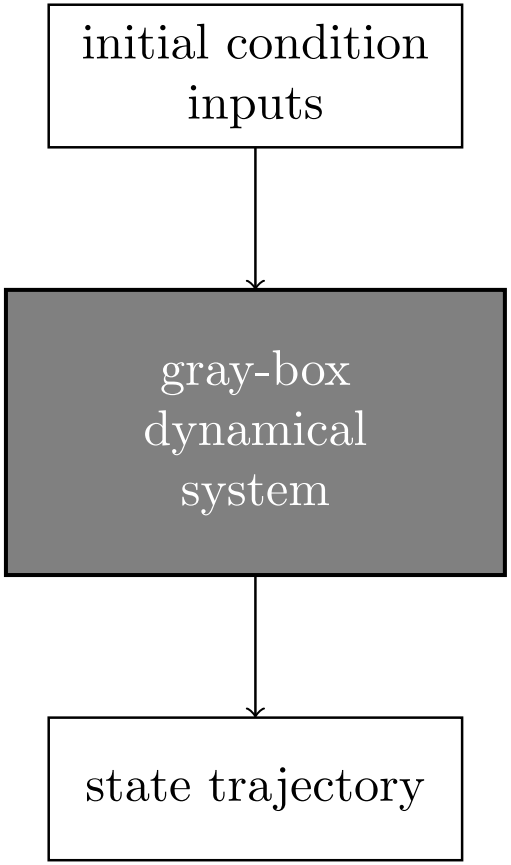
Our approach: Learn reduced models from data

Sample (gray-box) high-dimensional system with inputs

$$\boldsymbol{U} = \begin{bmatrix} \boldsymbol{u}_0 & \cdots & \boldsymbol{u}_{K-1} \end{bmatrix}$$

to obtain trajectory

$$\boldsymbol{X} = \begin{bmatrix} | & | & & | \\ \boldsymbol{x}_0 & \boldsymbol{x}_1 & \cdots & \boldsymbol{x}_K \\ | & | & & | \end{bmatrix}$$



Learn model $\hat{f}$ from data $\boldsymbol{U}$ and $\boldsymbol{X}$

$$\begin{aligned} \hat{\boldsymbol{x}}_{k+1} &= \hat{\boldsymbol{f}}(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k) \\ &= \sum_{i=1}^{\ell} \hat{\boldsymbol{A}}_i \boldsymbol{x}_k^i + \hat{\boldsymbol{B}} \boldsymbol{u}_k \,, \qquad k = 0, \ldots, K-1 \end{aligned}$$

Intro: Literature overview

System identification [Ljung, 1987], [Viberg, 1995], [Kramer, Gugercin, 2016], ...

Learning in frequency domain [Antoulas, Anderson, 1986], [Lefteriu, Antoulas, 2010], [Antoulas, 2016], [Gustavsen, Semlyen, 1999], [Drmac, Gugercin, Beattie, 2015], [Antoulas, Gosea, Ionita, 2016], [Gosea, Antoulas, 2018], [Benner, Goyal, Van Dooren, 2019], ...

Learning from time-domain data (output and state trajectories)

- Time series analysis (V)AR models, [Box et al., 2015], [Aicher et al., 2018, 2019], ...
- Learning models with dynamic mode decomposition [Schmid et al., 2008], [Rowley et al., 2009], [Proctor, Brunton, Kutz, 2016], [Benner, Himpe, Mitchell, 2018], ...
- Sparse identification [Brunton, Proctor, Kutz, 2016], [Schaeffer et al, 2017, 2018], ...
- Deep networks [Raissi, Perdikaris, Karniadakis, 2017ab], [Qin, Wu, Xiu, 2019], ...
- Bounds for LTI systems [Campi et al, 2002], [Vidyasagar et al, 2008], ...

Correction and data-driven closure modeling

- Closure modeling [Chorin, Stinis, 2006], [Oliver, Moser, 2011], [Parish, Duraisamy, 2015], [Iliescu et al, 2018, 2019], ...
- Higher order dynamic mode decomposition [Le Clainche and Vega, 2017], [Champion et al., 2018]

Outline

- Introduction and motivation
- **Operator inference for learning low-dimensional models**
- Sampling Markovian data for recovering reduced models
- Rigorous and pre-asymptotic error estimators
- Learning time delays to go beyond Markovian models
- Conclusions

OpInf: Fitting low-dim model to trajectories

1. **Construct POD (PCA) basis of dimension $n \ll N$**

$$\boldsymbol{V} = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n] \in \mathbb{R}^{N \times n}$$

2. **Project state trajectory onto the reduced space**

$$\check{\boldsymbol{X}} = \boldsymbol{V}^T \boldsymbol{X} = [\check{\boldsymbol{x}}_1, \cdots, \check{\boldsymbol{x}}_K] \in \mathbb{R}^{n \times K}$$

3. **Find operators $\hat{\boldsymbol{A}}_1, \ldots, \hat{\boldsymbol{A}}_\ell, \hat{\boldsymbol{B}}$ such that**

$$\check{\boldsymbol{x}}_{k+1} \approx \sum_{i=1}^{\ell} \hat{\boldsymbol{A}}_i \check{\boldsymbol{x}}_k^i + \hat{\boldsymbol{B}} \boldsymbol{u}_k, \qquad k = 0, \cdots, K-1$$

by minimizing the residual in Euclidean norm

$$\min_{\hat{\boldsymbol{A}}_1, \ldots, \hat{\boldsymbol{A}}_\ell, \hat{\boldsymbol{B}}} \sum_{k=0}^{K-1} \left\| \check{\boldsymbol{x}}_{k+1} - \sum_{i=1}^{\ell} \hat{\boldsymbol{A}}_i \check{\boldsymbol{x}}_k^i - \hat{\boldsymbol{B}} \boldsymbol{u}_k \right\|_2^2$$

[P., Willcox, *Data driven operator inference for nonintrusive projection-based model reduction*; Computer Methods in Applied Mechanics and Engineering, 306:196-215, 2016]

OpInf: Learning from projected trajectory

Fitting model to projected states

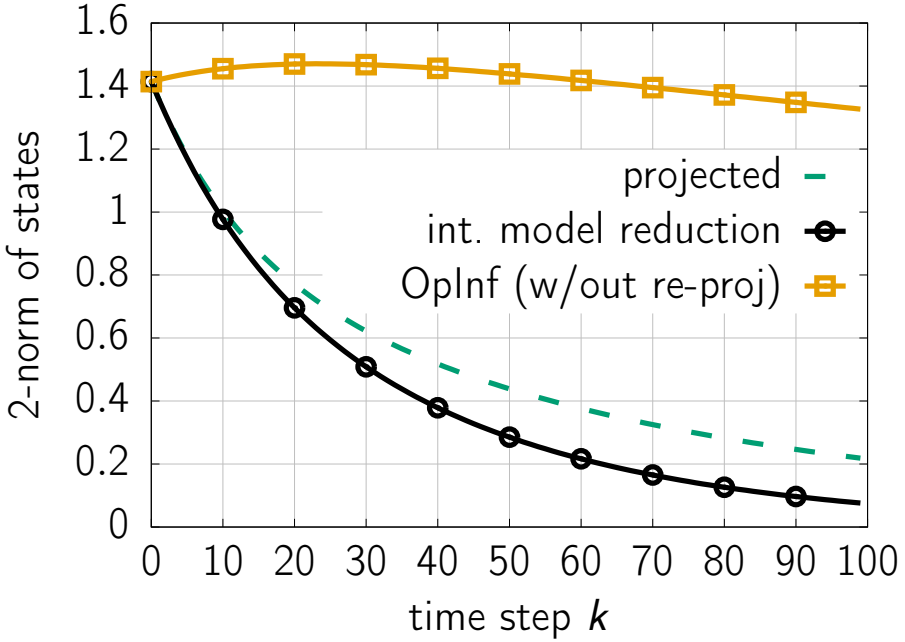
- We fit model to projected trajectory

$$\check{\boldsymbol{X}} = \boldsymbol{V}^T \boldsymbol{X}$$

- Would need $\tilde{\boldsymbol{X}} = [\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_K]$ because

$$\sum_{k=0}^{K-1} \left\| \tilde{\boldsymbol{x}}_{k+1} - \sum_{i=1}^{\ell} \tilde{\boldsymbol{A}}_i \tilde{\boldsymbol{x}}_k^i - \tilde{\boldsymbol{B}} \boldsymbol{u}_k \right\|_2^2 = 0$$

- However, trajectory $\tilde{\boldsymbol{X}}$ unavailable



Thus, $\|\hat{\boldsymbol{f}} - \tilde{\boldsymbol{f}}\|$ small critically depends on $\|\check{\boldsymbol{X}} - \tilde{\boldsymbol{X}}\|$ being small

- Increase dimension n of reduced space to decrease $\|\check{\boldsymbol{X}} - \tilde{\boldsymbol{X}}\|$

  $\Rightarrow$ increases degrees of freedom in OpInf $\Rightarrow$ ill-conditioned

- Decrease dimension n to keep number of degrees of freedom low

  $\Rightarrow$ difference $\|\check{\boldsymbol{X}} - \tilde{\boldsymbol{X}}\|$ increases

OpInf: Closure of linear system

Consider autonomous linear system

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k\,, \qquad \boldsymbol{x}_0 \in \mathbb{R}^N\,, \quad k = 0, \ldots, K-1$$

- Split $\mathbb{R}^N$ into $\mathcal{V} = \text{span}(\boldsymbol{V})$ and $\mathcal{V}_\perp = \text{span}(\boldsymbol{V}_\perp)$

$$\mathbb{R}^N = \mathcal{V} \oplus \mathcal{V}_\perp$$

- Split state

$$\boldsymbol{x}_k = \boldsymbol{V}\underbrace{\boldsymbol{V}^T\boldsymbol{x}_k}_{\boldsymbol{x}_k^{\parallel}} + \boldsymbol{V}_\perp\underbrace{\boldsymbol{V}_\perp^T\boldsymbol{x}_k}_{\boldsymbol{x}_k^{\perp}}$$

Represent system as

$$\begin{aligned}\boldsymbol{x}_{k+1}^{\parallel} &= \boldsymbol{A}_{11}\boldsymbol{x}_k^{\parallel} + \boldsymbol{A}_{12}\boldsymbol{x}_k^{\perp}\\ \boldsymbol{x}_{k+1}^{\perp} &= \boldsymbol{A}_{21}\boldsymbol{x}_k^{\parallel} + \boldsymbol{A}_{22}\boldsymbol{x}_k^{\perp}\end{aligned}$$

with operators

$$\boldsymbol{A}_{11} = \underbrace{\boldsymbol{V}^T\boldsymbol{A}\boldsymbol{V}}_{=\tilde{\boldsymbol{A}}}, \quad \boldsymbol{A}_{12} = \boldsymbol{V}^T\boldsymbol{A}\boldsymbol{V}_\perp\,, \boldsymbol{A}_{21} = \boldsymbol{V}_\perp^T\boldsymbol{A}\boldsymbol{V}\,, \quad \boldsymbol{A}_{22} = \boldsymbol{V}_\perp^T\boldsymbol{A}\boldsymbol{V}_\perp$$

[Givon, Kupferman, Stuart, 2004], [Chorin, Stinis, 2006] [Parish, Duraisamy, 2017]

OpInf: Closure term as a non-Markovian term

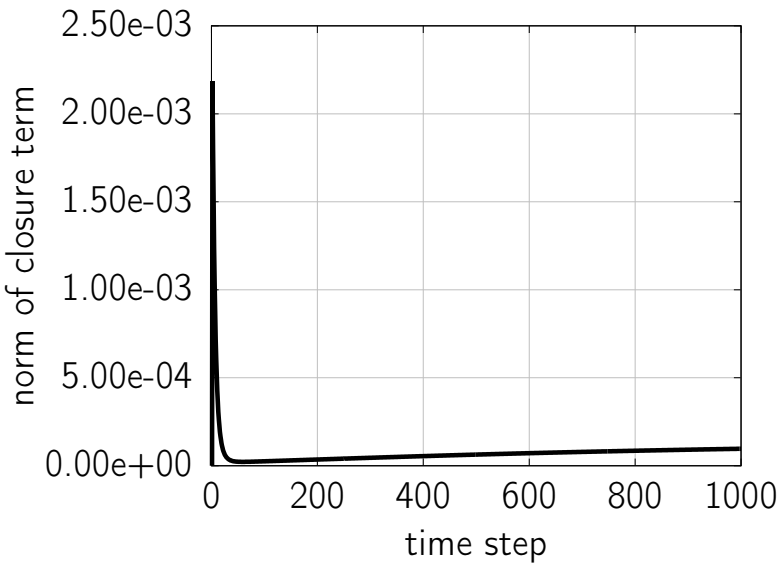
Projected trajectory $\check{X}$ mixes dynamics in $\mathcal{V}$ and $\mathcal{V}_\perp$

$$\boldsymbol{V}^T \boldsymbol{x}_{k+1} = \check{\boldsymbol{x}}_{k+1} = \boldsymbol{x}_{k+1}^{\parallel} = \boldsymbol{A}_{11}\boldsymbol{x}_k^{\parallel} + \boldsymbol{A}_{12}\boldsymbol{x}_k^{\perp}$$

Mori-Zwanzig formalism gives [Givon, Kupferman, Stuart, 2004], [Chorin, Stinis, 2006]

$$\begin{aligned}\boldsymbol{V}^T \boldsymbol{x}_{k+1} = \boldsymbol{x}_{k+1}^{\parallel} &= \boldsymbol{A}_{11}\boldsymbol{x}_k^{\parallel} + \boldsymbol{A}_{12}\boldsymbol{x}_k^{\perp} \\ &= \boldsymbol{A}_{11}\boldsymbol{x}_k^{\parallel} + \sum_{j=1}^{k-1} \boldsymbol{A}_{22}^{k-j-1}\boldsymbol{A}_{21}\boldsymbol{x}_j^{\parallel} + \boldsymbol{A}_{12}\boldsymbol{A}_{22}^{k-1}\boldsymbol{x}_0^{\perp}\end{aligned}$$

Non-Markovian (memory) term models *unobserved* dynamics



Outline

- Introduction and motivation
- Operator inference for learning low-dimensional models
- **Sampling Markovian data for recovering reduced models**
- Rigorous and pre-asymptotic error estimators
- Learning time delays to go beyond Markovian models
- Conclusions

ReProj: Handling non-Markovian dynamics

Ignore non-Markovian dynamics

- Have significant impact on model accuracy (much more than in classical model reduction?)
- Guarantees on models?

Fit models with different forms to capture non-Markovian dynamics

- Length of memory (support of kernel) typically unknown
- Time-delay embedding increase dimension of reduced states, which is what we want to reduce
- Model reduction (theory) mostly considers Markovian reduced models

Our approach: Control length of memory when sampling trajectories

- Set length of memory to 0 for sampling Markovian dynamics
- Increase length of memory in a controlled way (lag is known)
- Modify the sampling scheme, instead of learning step
- Emphasizes importance of generating the "right" data

ReProj: Avoiding closure

Mori-Zwanzig formalism explains projected trajectory as

$$\boldsymbol{V}^T \boldsymbol{x}_{k+1} = \boldsymbol{x}_{k+1}^{\parallel} = \underbrace{\boldsymbol{A}_{11}\boldsymbol{x}_k^{\parallel}}_{\text{reduced model}} + \underbrace{\sum_{j=1}^{k-1} \boldsymbol{A}_{22}^{k-j-1}\boldsymbol{A}_{21}\boldsymbol{x}_j^{\parallel}}_{\text{memory}} + \underbrace{\boldsymbol{A}_{12}\boldsymbol{A}_{22}^{k-1}\boldsymbol{x}_0^{\perp}}_{\text{noise}}$$

Sample Markovian dynamics by setting memory and noise to 0

- Set $\boldsymbol{x}_0 \in \mathcal{V}$, then noise is 0
- Take a single time step, then memory term is 0

Sample trajectory by re-projecting state of previous time step onto $\mathcal{V}$

Establishes "independence"

ReProj: Sampling with re-projection

Data sampling: Cancel non-Markovian terms via re-projection

1. Project initial condition $\boldsymbol{x}_0$ onto $\mathcal{V}$

$$\bar{\boldsymbol{x}}_0 = \boldsymbol{V}^T \boldsymbol{x}_0$$

2. Query high-dim. system for **a single time step** with $\boldsymbol{V}\bar{\boldsymbol{x}}_0$

$$\boldsymbol{x}_1 = \boldsymbol{f}(\boldsymbol{V}\bar{\boldsymbol{x}}_0, \boldsymbol{u}_0)$$

3. Re-project to obtain $\bar{\boldsymbol{x}}_1 = \boldsymbol{V}^T \boldsymbol{x}_1$
4. Query high-dim. system with re-projected initial condition $\boldsymbol{V}\bar{\boldsymbol{x}}_1$

$$\boldsymbol{x}_2 = \boldsymbol{f}(\boldsymbol{V}\bar{\boldsymbol{x}}_1, \boldsymbol{u}_1)$$

5. Repeat until end of time-stepping loop

Obtain trajectories

$$\bar{\boldsymbol{X}} = [\bar{\boldsymbol{x}}_0, \ldots, \bar{\boldsymbol{x}}_{K-1}], \qquad \bar{\boldsymbol{Y}} = [\bar{\boldsymbol{x}}_1, \ldots, \bar{\boldsymbol{x}}_K], \qquad \boldsymbol{U} = [\boldsymbol{u}_0, \ldots, \boldsymbol{u}_{K-1}]$$

[P., Sampling low-dimensional Markovian dynamics for pre-asymptotically recovering reduced models from data with operator inference. arXiv:1908.11233, 2019.]

ReProj: Operator inference with re-projection

Operator inference with re-projected trajectories

$$\min_{\hat{\boldsymbol{A}}_1,\ldots,\hat{\boldsymbol{A}}_\ell,\hat{\boldsymbol{B}}} \left\| \bar{\boldsymbol{Y}} - \sum_{i=1}^{\ell} \hat{\boldsymbol{A}}_i \bar{\boldsymbol{X}}^i - \hat{\boldsymbol{B}}\boldsymbol{U} \right\|_F^2$$

Theorem (*Simplified*) Consider time-discrete system with polynomial nonlinear terms of maximal degree ℓ and linear input. If $K \geq \sum_{i=1}^{\ell} n^i + 2$ and matrix $[\bar{\boldsymbol{X}}, \boldsymbol{U}, \bar{\boldsymbol{X}}^2, \ldots, \bar{\boldsymbol{X}}^\ell]$ has full rank, then $\|\bar{\boldsymbol{X}} - \tilde{\boldsymbol{X}}\| = 0$ and thus $\hat{\boldsymbol{f}} = \tilde{\boldsymbol{f}}$ in the sense

$$\|\hat{\boldsymbol{A}}_1 - \tilde{\boldsymbol{A}}_1\|_F = \cdots = \|\hat{\boldsymbol{A}}_\ell - \tilde{\boldsymbol{A}}_\ell\|_F = \|\tilde{\boldsymbol{B}} - \hat{\boldsymbol{B}}\|_F = 0$$

- Pre-asymptotic guarantees, in contrast to learning from projected data
- Re-projection is a nonintrusive operation
- Requires querying high-dim. system twice
- Initial conditions remain "physically meaningful"

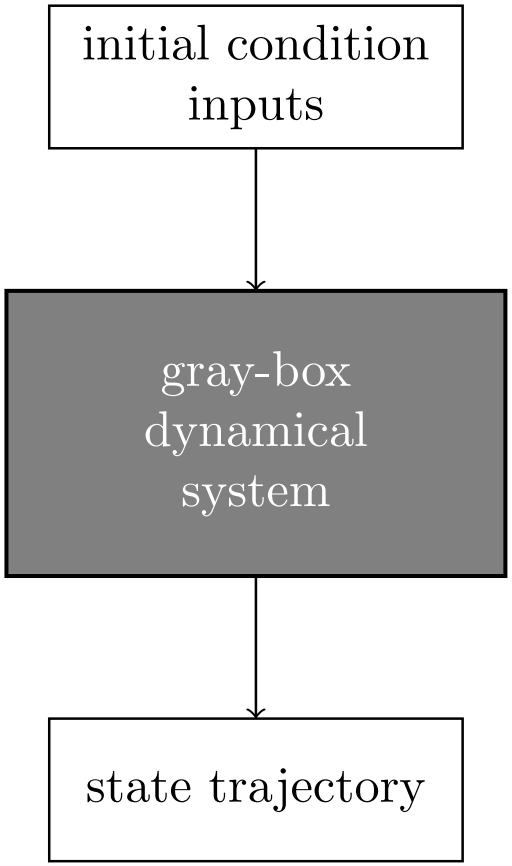
Provides a means to find model form

[P., Sampling low-dimensional Markovian dynamics for pre-asymptotically recovering reduced models from data with operator inference. arXiv:1908.11233, 2019.]

ReProj: Queryable systems

Definition: Queryable systems [Uy, P., 2020]

A dynamical system is queryable, if the trajectory $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_K]$ *with* $K \geq 1$ *can be computed for initial condition* $\boldsymbol{x}_0 \in \mathcal{V}$ *and feasible input trajectory* $\boldsymbol{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_K]$.



Details about *how* trajectories computed unnecessary

- Discretization (FEM, FD, FV, etc)
- Time-stepping scheme
- Time-step size
- In particular, neither explicit nor implicit access to operators required

Insufficient to have only data available

- Need to query system at re-projected states
- Similar requirement as for active learning

ReProj: Burgers': Burgers' example

Viscous Burgers' equation

$$\frac{\partial}{\partial t}x(\omega, t; \mu) + x(\omega, t; \mu)\frac{\partial}{\partial \omega}x(\omega, t; \mu) - \mu\frac{\partial^2}{\partial \omega^2}x(\omega, t; \mu) = 0$$

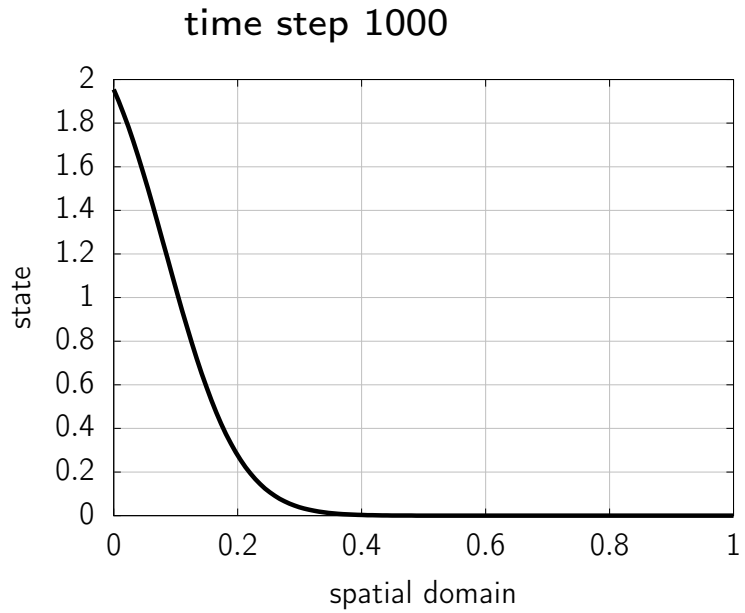
- Spatial, time, and parameter domain

$$\omega \in [0, 1]\,, \quad t \in [0, 1]\,, \quad \mu \in [0.1, 1]$$

- Dirichlet boundary conditions

$$x(0, t; \mu) = -x(1, t; \mu) = u(t)$$

- Discretize with forward Euler
- Time step size is $\delta t = 10^{-4}$



Operator inference

- Training data are 2 trajectories with random inputs
- Infer operators for 10 equidistant parameters in $[0.1, 1]$
- Interpolate inferred operators at 7 test parameters and predict

ReProj: Burgers': Burgers' example

Viscous Burgers' equation

$$\frac{\partial}{\partial t}x(\omega, t; \mu) + x(\omega, t; \mu)\frac{\partial}{\partial \omega}x(\omega, t; \mu) - \mu\frac{\partial^2}{\partial \omega^2}x(\omega, t; \mu) = 0$$

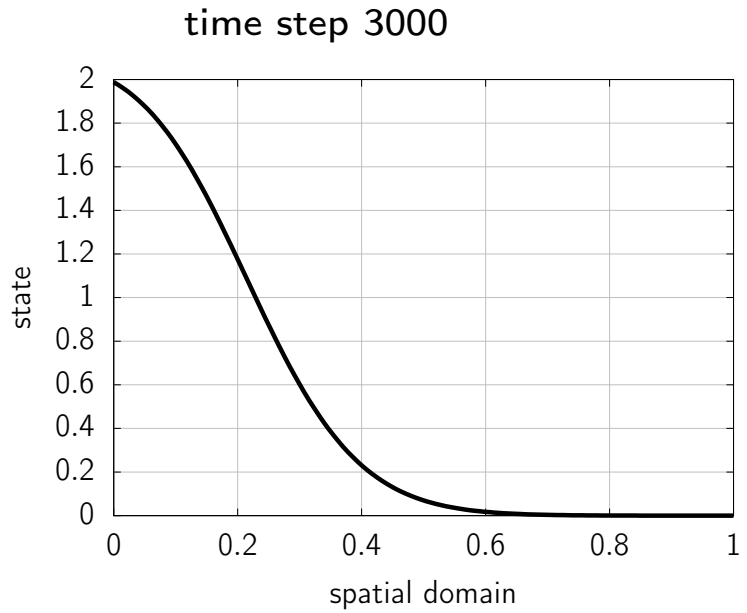
- Spatial, time, and parameter domain

$$\omega \in [0, 1]\,, \quad t \in [0, 1]\,, \quad \mu \in [0.1, 1]$$

- Dirichlet boundary conditions

$$x(0, t; \mu) = -x(1, t; \mu) = u(t)$$

- Discretize with forward Euler
- Time step size is $\delta t = 10^{-4}$



Operator inference

- Training data are 2 trajectories with random inputs
- Infer operators for 10 equidistant parameters in $[0.1, 1]$
- Interpolate inferred operators at 7 test parameters and predict

ReProj: Burgers': Burgers' example

Viscous Burgers' equation

$$\frac{\partial}{\partial t}x(\omega, t; \mu) + x(\omega, t; \mu)\frac{\partial}{\partial \omega}x(\omega, t; \mu) - \mu\frac{\partial^2}{\partial \omega^2}x(\omega, t; \mu) = 0$$

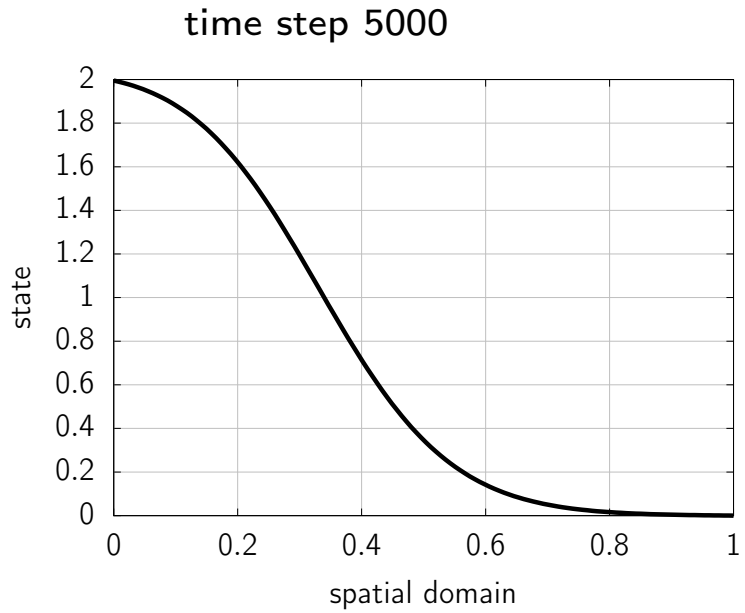
- Spatial, time, and parameter domain

$$\omega \in [0, 1]\,, \quad t \in [0, 1]\,, \quad \mu \in [0.1, 1]$$

- Dirichlet boundary conditions

$$x(0, t; \mu) = -x(1, t; \mu) = u(t)$$

- Discretize with forward Euler
- Time step size is $\delta t = 10^{-4}$



Operator inference

- Training data are 2 trajectories with random inputs
- Infer operators for 10 equidistant parameters in $[0.1, 1]$
- Interpolate inferred operators at 7 test parameters and predict

ReProj: Burgers': Burgers' example

Viscous Burgers' equation

$$\frac{\partial}{\partial t}x(\omega, t; \mu) + x(\omega, t; \mu)\frac{\partial}{\partial \omega}x(\omega, t; \mu) - \mu\frac{\partial^2}{\partial \omega^2}x(\omega, t; \mu) = 0$$

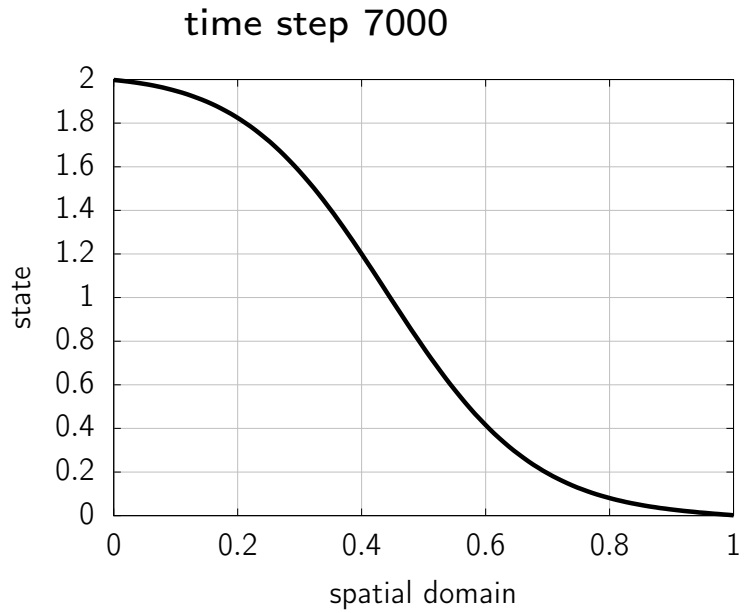
- Spatial, time, and parameter domain

$$\omega \in [0, 1]\,, \quad t \in [0, 1]\,, \quad \mu \in [0.1, 1]$$

- Dirichlet boundary conditions

$$x(0, t; \mu) = -x(1, t; \mu) = u(t)$$

- Discretize with forward Euler
- Time step size is $\delta t = 10^{-4}$



Operator inference

- Training data are 2 trajectories with random inputs
- Infer operators for 10 equidistant parameters in $[0.1, 1]$
- Interpolate inferred operators at 7 test parameters and predict

ReProj: Burgers': Burgers' example

Viscous Burgers' equation

$$\frac{\partial}{\partial t}x(\omega, t; \mu) + x(\omega, t; \mu)\frac{\partial}{\partial \omega}x(\omega, t; \mu) - \mu\frac{\partial^2}{\partial \omega^2}x(\omega, t; \mu) = 0$$

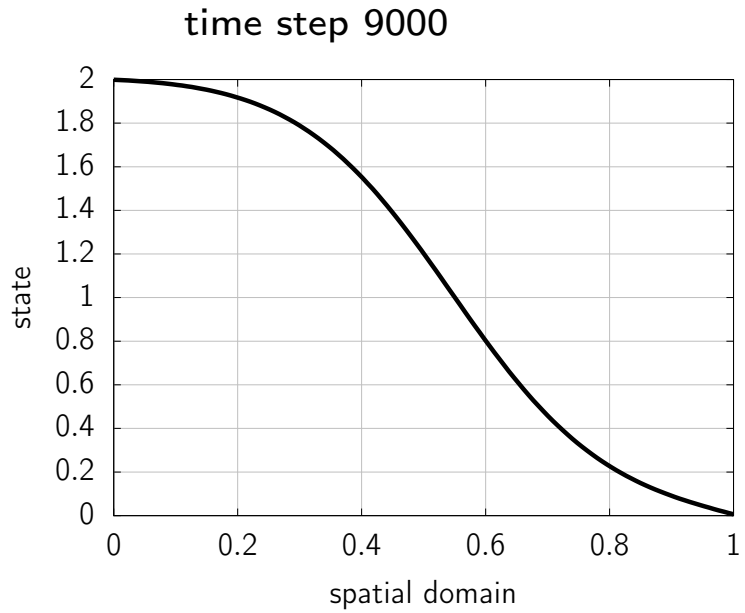
- Spatial, time, and parameter domain

$$\omega \in [0, 1]\,, \quad t \in [0, 1]\,, \quad \mu \in [0.1, 1]$$

- Dirichlet boundary conditions

$$x(0, t; \mu) = -x(1, t; \mu) = u(t)$$

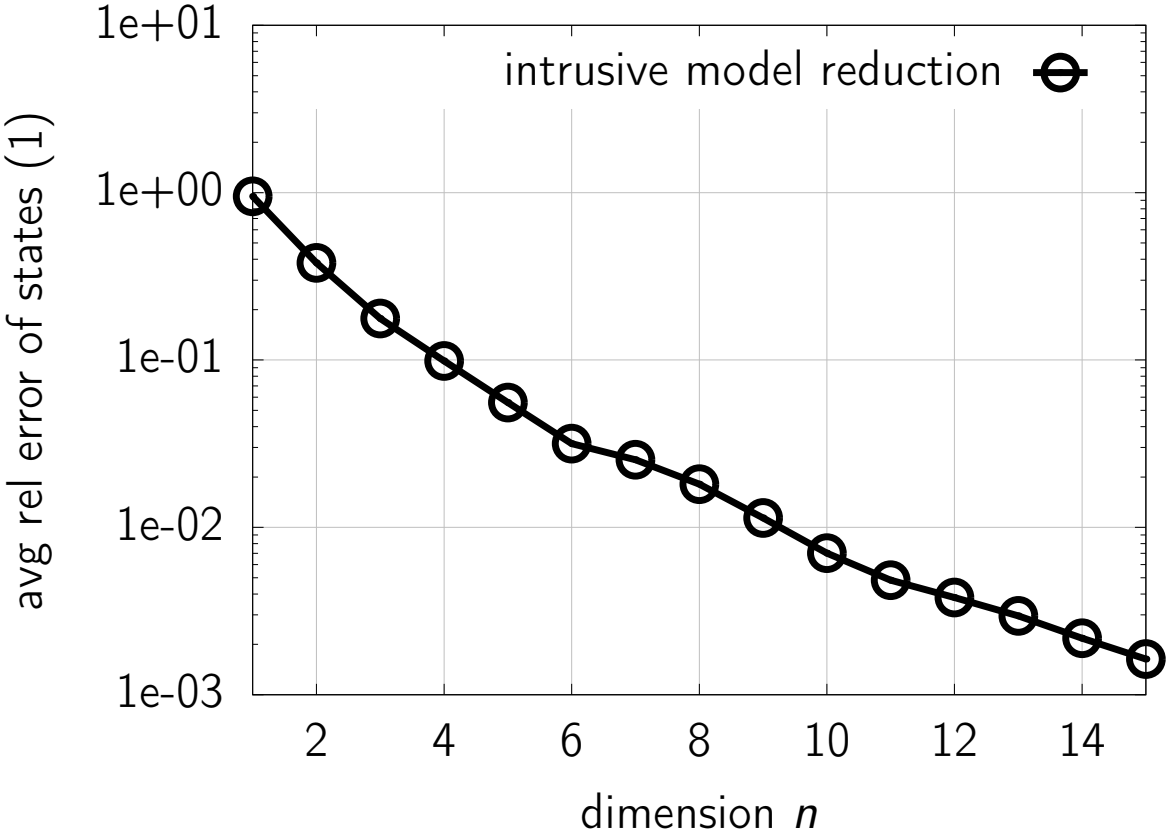
- Discretize with forward Euler
- Time step size is $\delta t = 10^{-4}$



Operator inference

- Training data are 2 trajectories with random inputs
- Infer operators for 10 equidistant parameters in $[0.1, 1]$
- Interpolate inferred operators at 7 test parameters and predict

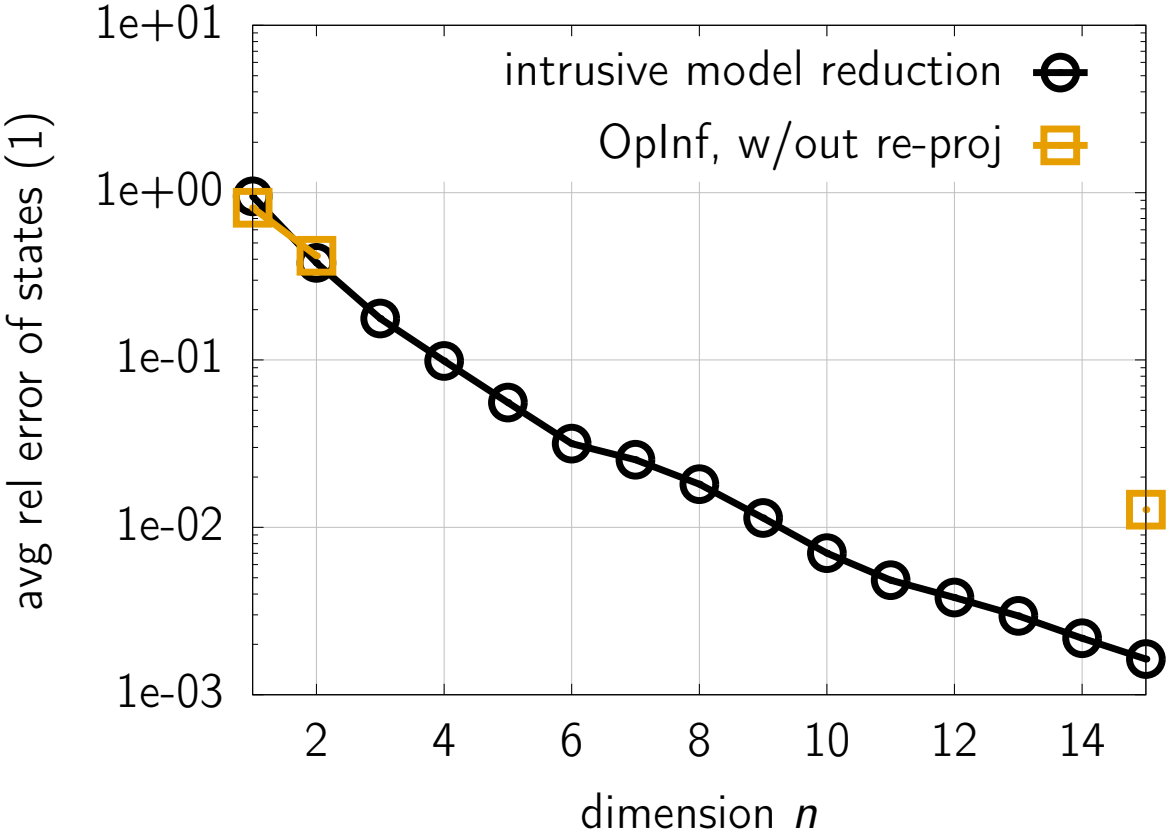
ReProj: Burgers': Operator inference



Error of reduced models at test data

- Inferring operators from projected data fails in this example
- Recover reduced model from re-projected data

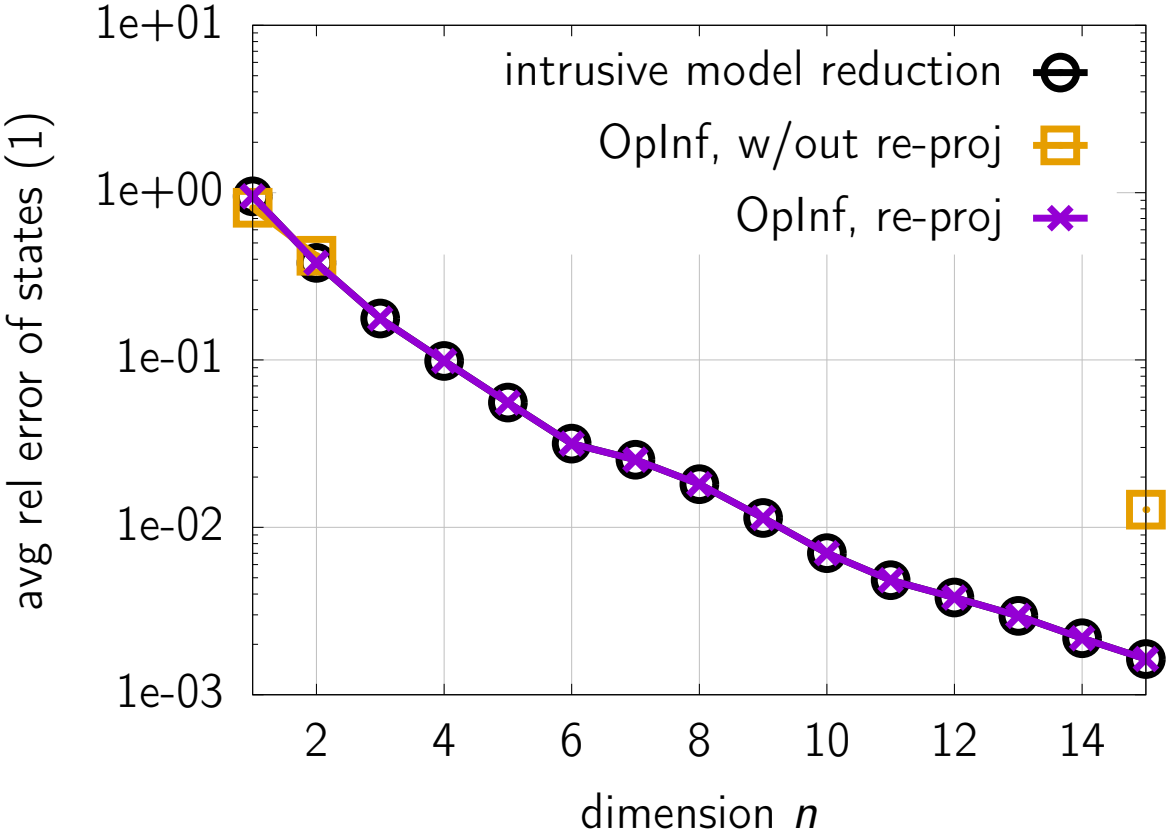
ReProj: Burgers': Operator inference



Error of reduced models at test data

- Inferring operators from projected data fails in this example
- Recover reduced model from re-projected data

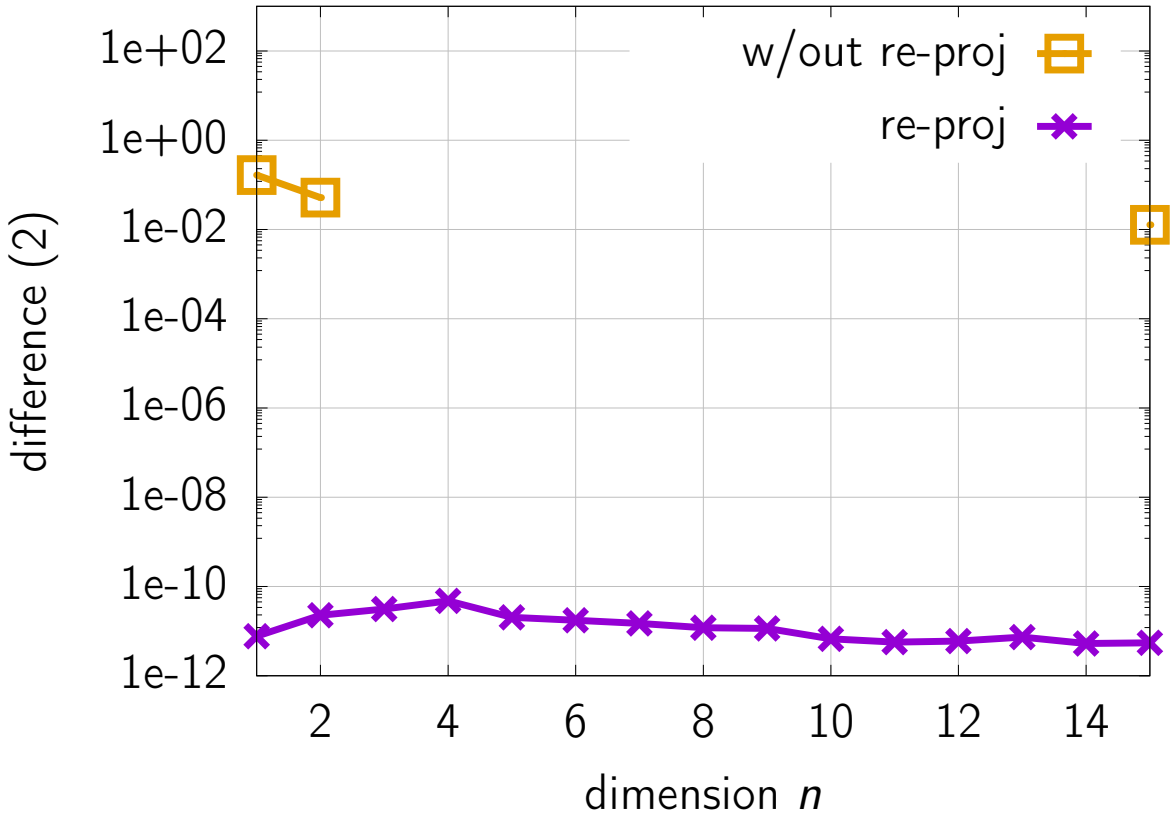
ReProj: Burgers': Operator inference



Error of reduced models at test data

- Inferring operators from projected data fails in this example
- Recover reduced model from re-projected data

ReProj: Burgers': Recovery



The difference between state trajectories

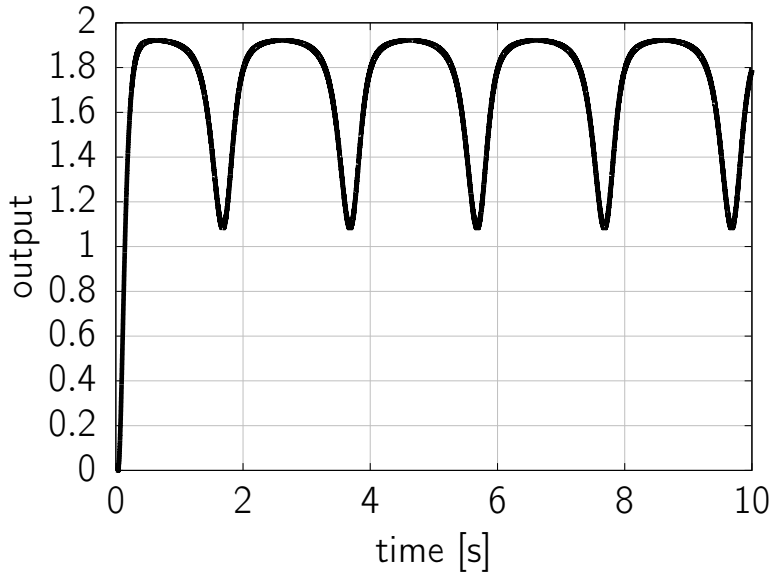
- Model from intrusive model reduction same as OpInf with re-proj.
- Model learned from state trajectories without re-projection differs

ReProj: Chafee: Chafee-Infante example

Chafee-Infante equation

$$\frac{\partial}{\partial t}x(\omega,t) + x^3(\omega,t) - \frac{\partial^2}{\partial \omega^2}x(\omega,t) - x(\omega,t) = 0$$

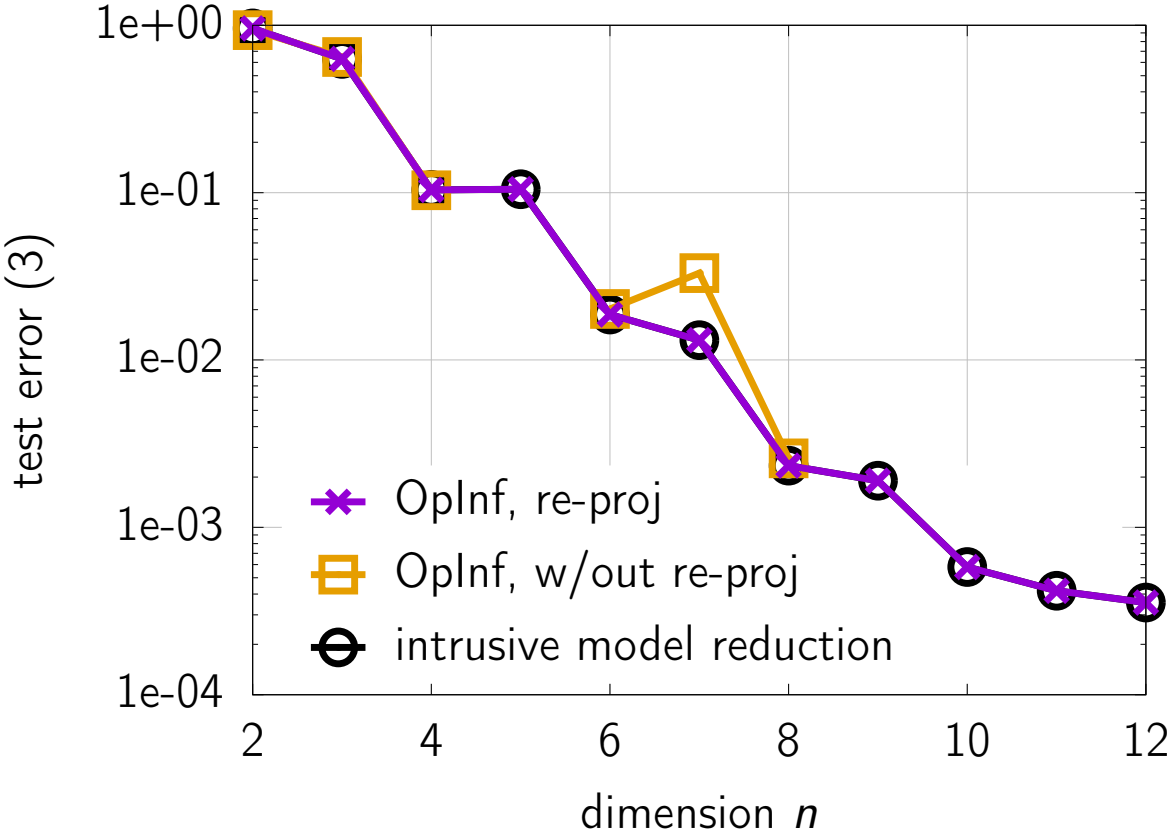
- Boundary conditions as in [Benner et al., 2018]
- Spatial domain $\omega \in [0,1]$
- Time domain $t \in [0,10]$
- Forward Euler with $\delta t = 10^{-4}$
- Cubic nonlinear term



Operator inference

- Infer operators from single trajectory corresponding to random inputs
- Test inferred model on oscillatory input

ReProj: Chafee: Recovery

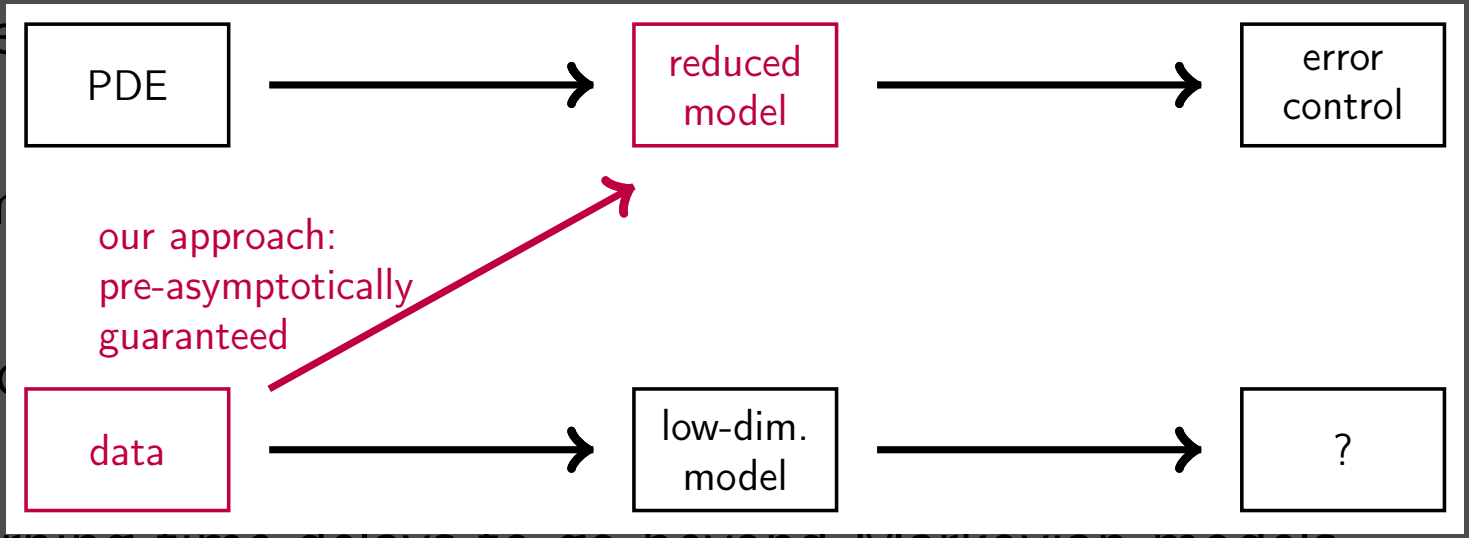


Error of reduced models on test parameters

- Projected data leads to unstable inferred model
- Inference from data with re-projection shows stabler behavior

Outline

- Introduction and motivation
- Ope
- Sam
- Rig
- Learning time delays to go beyond Markovian models
- Conclusions



Outline

- Introduction and motivation
- Operator inference for learning low-dimensional models
- Sampling Markovian data for recovering reduced models
- **Rigorous and pre-asymptotic error estimators**
- Learning time delays to go beyond Markovian models
- Conclusions

ErrEst: Error estimation for learned models

Assumptions*: Symmetric asymptotically stable linear system

- If not symmetric, then need to assume $\|\boldsymbol{A}_1\| \leq 1$ (for now...)
- Derive reduced model with operator inference and re-projection
- Requires full residual of reduced-model states in training phase

Error estimation based on [Haasdonk, Ohlberger, 2009]

- Residual at time step k

$$\boldsymbol{r}_k = \boldsymbol{A}_1 \boldsymbol{V} \hat{\boldsymbol{x}}_k + \boldsymbol{B} u_k - \boldsymbol{V} \hat{\boldsymbol{x}}_{k+1}$$

- Bound on state error if initial condition in span$\{\boldsymbol{V}\}$

$$\|\boldsymbol{x}_k - \boldsymbol{V}\hat{\boldsymbol{x}}_k\|_2 \leq C_1 \left(\sum_{i=1}^{k-1} \|\boldsymbol{r}_k\|_2 \right)$$

- Offline/online splitting of computing residual norm $\|\boldsymbol{r}_k\|_2$

$$\begin{aligned} \|\boldsymbol{r}_k\|_2^2 = & \hat{\boldsymbol{x}}_k^T \underbrace{\boldsymbol{V}^T \boldsymbol{A}_1^T \boldsymbol{A}_1 \boldsymbol{V}}_{\boldsymbol{M}_1} \hat{\boldsymbol{x}}_k + u_k \underbrace{\boldsymbol{B}^T \boldsymbol{B}}_{\boldsymbol{M}_2} u_k + \hat{\boldsymbol{x}}_{k+1} \boldsymbol{V}^T \boldsymbol{V} \hat{\boldsymbol{x}}_{k+1} \\ & + 2 u_k^T \underbrace{\boldsymbol{B}^T \boldsymbol{A}_1 \boldsymbol{V}}_{\boldsymbol{M}_3} \hat{\boldsymbol{x}}_k - 2 \hat{\boldsymbol{x}}_{k+1}^T \hat{\boldsymbol{A}}_1 \hat{\boldsymbol{x}}_{k+1} - 2 \hat{\boldsymbol{x}}_{k+1} \hat{\boldsymbol{B}} u_k \end{aligned}$$

ErrEst: Learning error operators from data

From [Haasdonk, Ohlberger, 2009] have

$$\|\boldsymbol{r}_k\|_2^2 = \hat{\boldsymbol{x}}_k^T \underbrace{\boldsymbol{V}^T \boldsymbol{A}_1^T \boldsymbol{A}_1 \boldsymbol{V}}_{\boldsymbol{M}_1} \hat{\boldsymbol{x}}_k + u_k \underbrace{\boldsymbol{B}^T \boldsymbol{B}}_{\boldsymbol{M}_2} u_k + \hat{\boldsymbol{x}}_{k+1} \boldsymbol{V}^T \boldsymbol{V} \hat{\boldsymbol{x}}_{k+1}$$
$$+ 2u_k^T \underbrace{\boldsymbol{B}^T \boldsymbol{A}_1 \boldsymbol{V}}_{\boldsymbol{M}_3} \hat{\boldsymbol{x}}_k - 2\hat{\boldsymbol{x}}_{k+1}^T \hat{\boldsymbol{A}}_1 \hat{\boldsymbol{x}}_{k+1} - 2\hat{\boldsymbol{x}}_{k+1} \hat{\boldsymbol{B}} u_k$$

Query system at training inputs to compute residual trajectories

$$\boldsymbol{R} = \begin{bmatrix} | & | & & | \\ \boldsymbol{r}_1 & \boldsymbol{r}_2 & \dots & \boldsymbol{r}_K \\ | & | & & | \end{bmatrix}$$

Learn quantities $\boldsymbol{M}_1, \boldsymbol{M}_2, \boldsymbol{M}_3$ via operator inference

- Fit error operators $\boldsymbol{M}_1, \boldsymbol{M}_2, \boldsymbol{M}_3$ to residual trajectories
- Bound constant C_1 and constants for output error

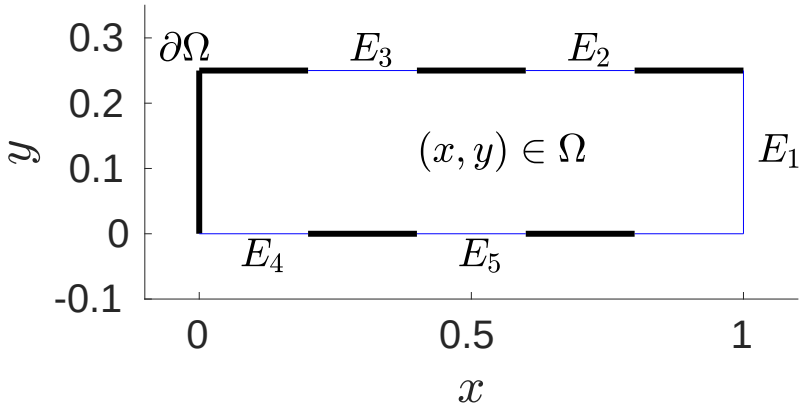
Obtain *certified* reduced models from data alone

[Uy, P., *Pre-asymptotic error bounds for low-dimensional models learned from systems governed by linear parabolic partial differential equations with control inputs*, *in preparation*, 2020]

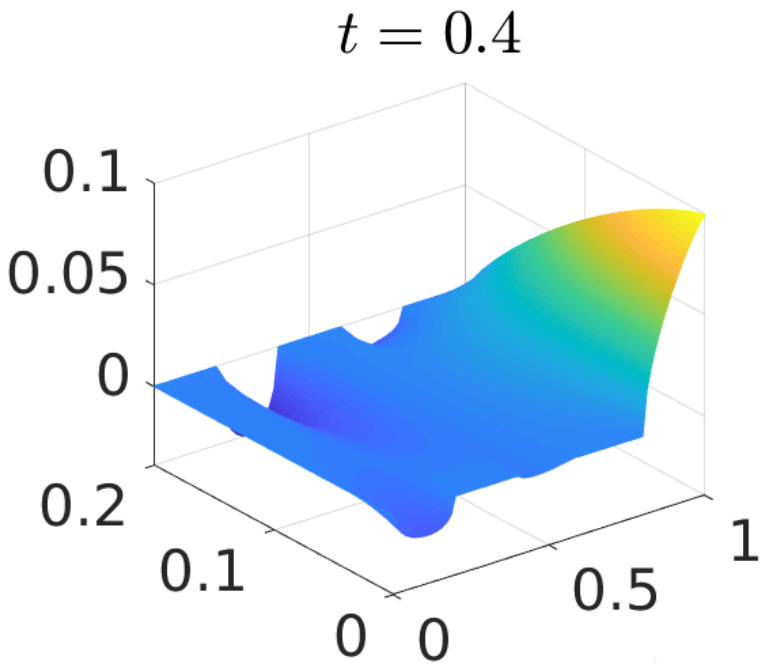
ErrEst: Convection-diffusion in a pipe

Governed by parabolic PDE

$$\frac{\partial x}{\partial t} = \Delta x - (1,1) \cdot \nabla x, \qquad \text{in } \Omega$$
$$x = 0, \qquad \Gamma \backslash \{E_i\}_{i=1}^{5}$$
$$\nabla x \cdot \mathbf{n} = g_i(t), \qquad \text{in } E_i$$



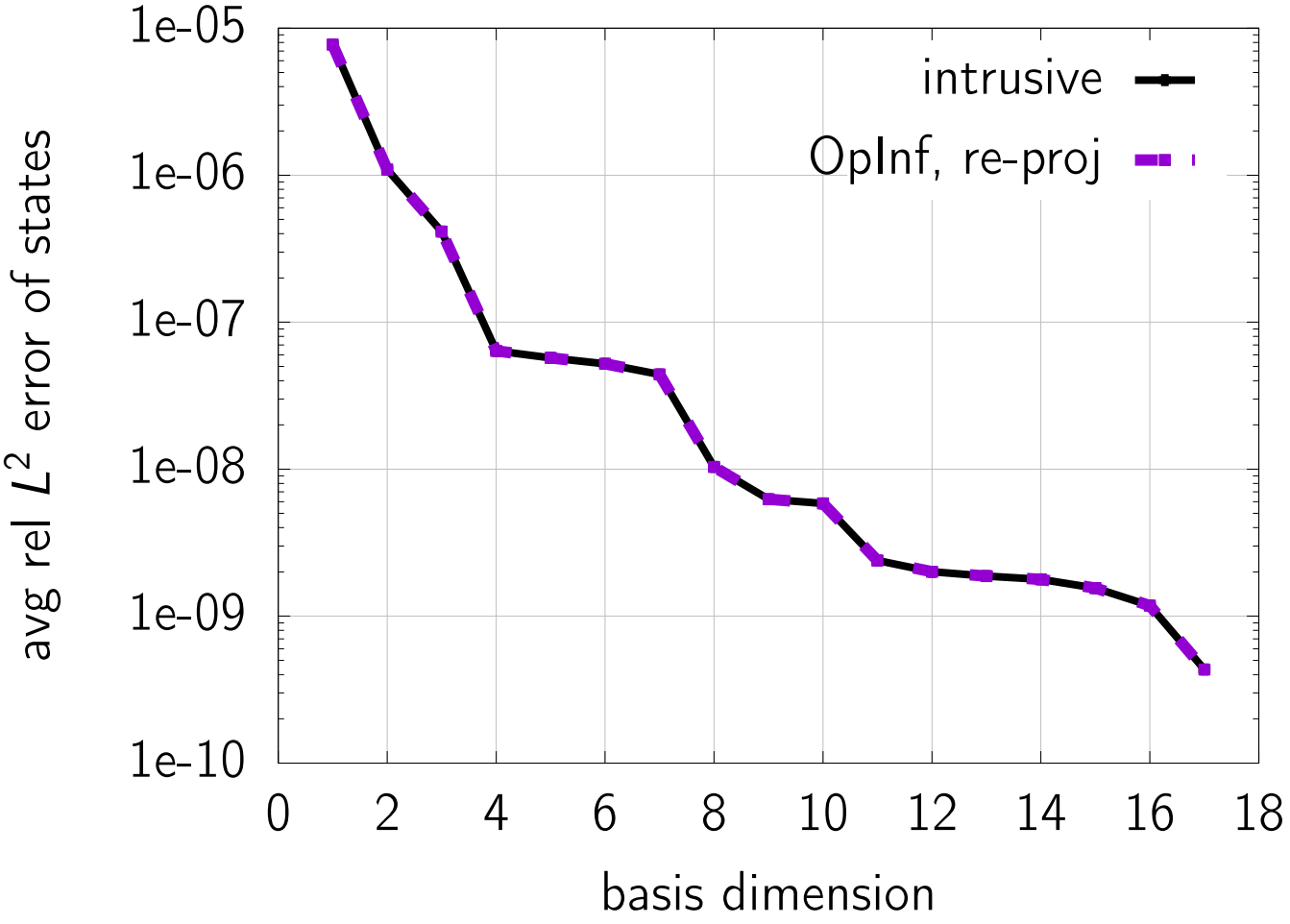
- Discretize with finite elements
- Degrees of freedom $N = 1121$
- Forward Euler method $\delta t = 10^{-5}$
- End time is $T = 0.5$



Input signals

- Training signal is sinusoidal
- Test signal is exponentially decaying sinusoidal with different frequency than training

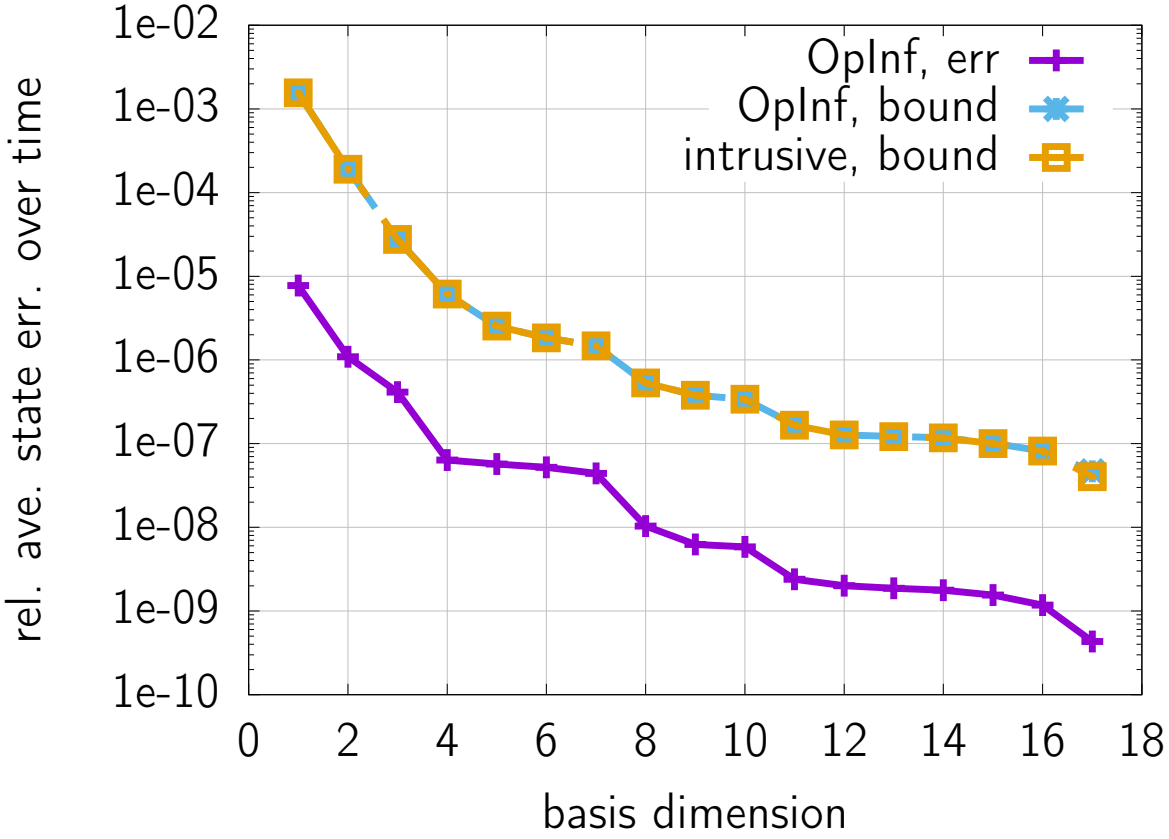
ErrEst: Recovering reduced models from data



Recover reduced models from data

- Error averaged over time
- Recover reduced model up to numerical errors

ErrEst: Error bounds

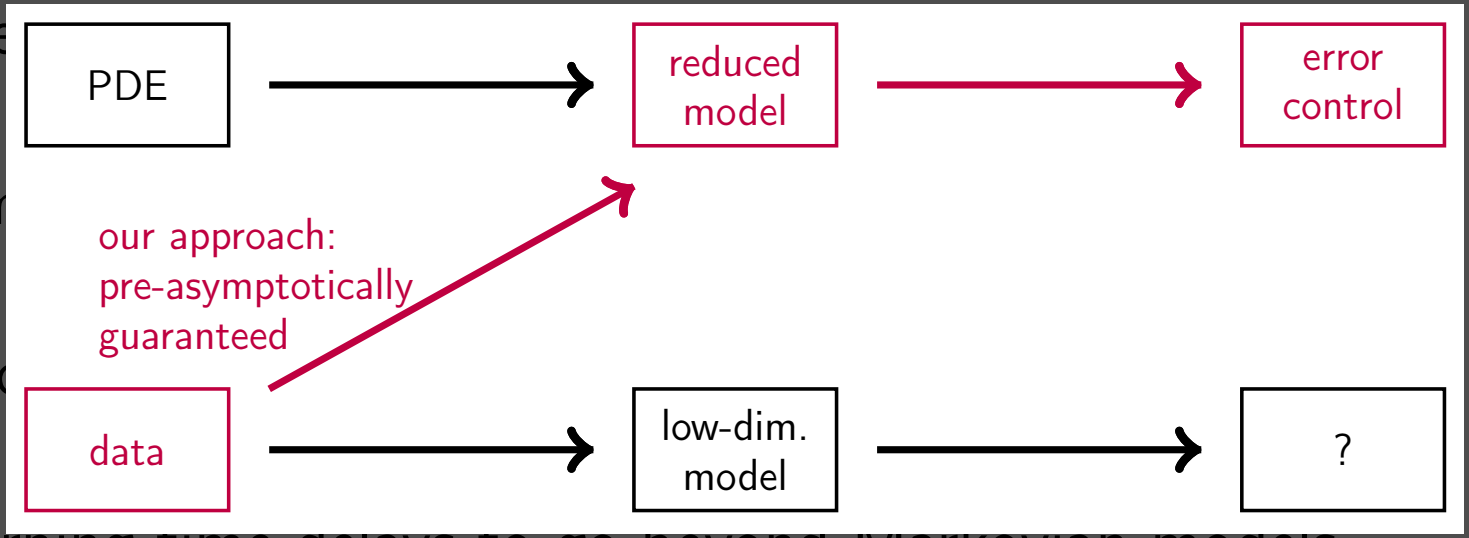


Learn certified reduced model from data alone

- Train with sinusoidal and test with exponential input
- Infer quantities from residual of full model (offline/training)
- Estimate error for test inputs

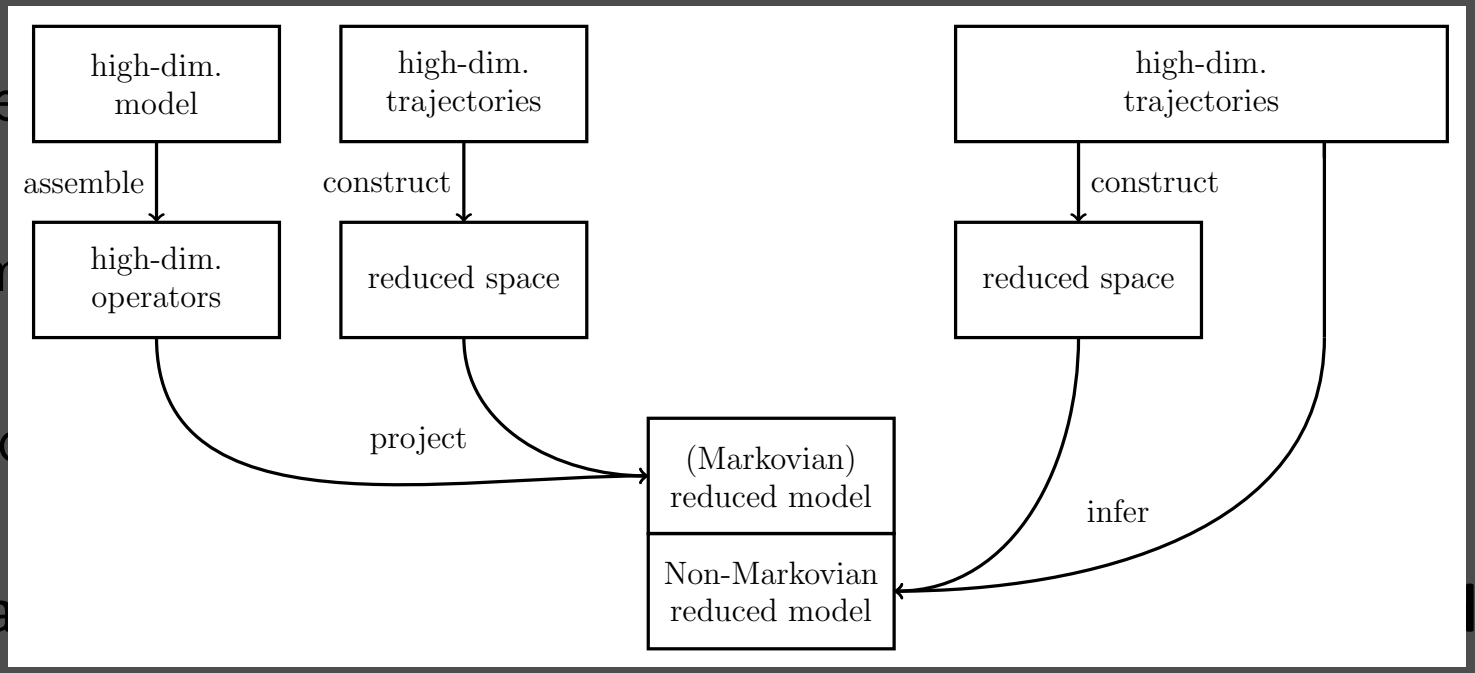
Outline

- Introduction and motivation
- Ope
- Sam
- Rig
- Learning time delays to go beyond Markovian models
- Conclusions

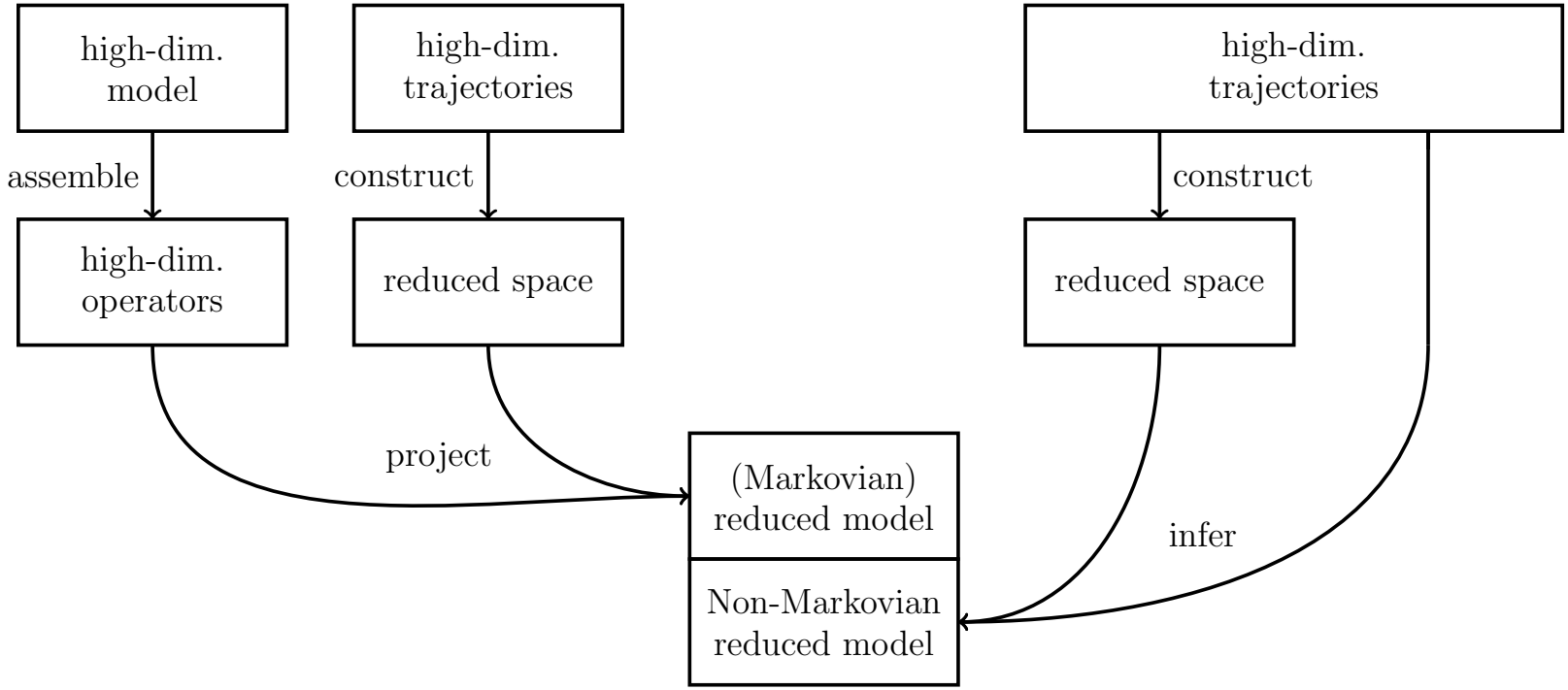


Outline

- Introduction and motivation
- Ope
- Sam
- Rig
- **Lea** **lels**
- Conclusions



NonM: Non-Markovian reduced models



Learning non-Markovian low-dim. models in model reduction

- (Full model is non-Markovian [Schulze, Unger, Beattie, Gugercin, 2018])
- Closure error is high and needs to be corrected (steep gradients, shocks)
- Only partially observed state trajectory available

NonM: Learning non-Markovian reduced models

With re-projection, exactly learn Markovian reduced model

$$\tilde{\boldsymbol{x}}_{k+1} = \sum_{i=1}^{\ell} \tilde{\boldsymbol{A}}_i \tilde{\boldsymbol{x}}_k^i + \tilde{\boldsymbol{B}} \boldsymbol{u}_k$$

However, loose dynamics modeled by non-Markovian terms

$$\check{\boldsymbol{x}}_{k+1} = \sum_{i=1}^{\ell} \tilde{\boldsymbol{A}}_i \check{\boldsymbol{x}}_k^i + \tilde{\boldsymbol{B}} \boldsymbol{u}_k + \sum_{i=1}^{k-1} \boldsymbol{\Delta}_i(\check{\boldsymbol{x}}_{k-1}, \ldots, \check{\boldsymbol{x}}_{k-i+1}, \boldsymbol{u}_k, \ldots, \boldsymbol{u}_{k-i+1}) + 0$$

Learn unresolved dynamics via approximate non-Markovian terms

$$\hat{\boldsymbol{x}}_{k+1} = \sum_{i=1}^{\ell} \hat{\boldsymbol{A}}_i \hat{\boldsymbol{x}}_k^i + \hat{\boldsymbol{B}} \boldsymbol{u}_k + \sum_{i=1}^{k-1} \hat{\boldsymbol{\Delta}}_i^{\boldsymbol{\theta}_i}(\hat{\boldsymbol{x}}_{k-1}, \ldots, \hat{\boldsymbol{x}}_{k-i+1}, \boldsymbol{u}_k, \ldots, \boldsymbol{u}_{k-i+1})$$

- Parametrization $\boldsymbol{\theta}_i \in \Theta$ for $i = 0, \ldots, K-1$
- Non-Markovian models extensively used in statistics but less so in MOR

NonM: Sampling with stage-wise re-projection

Learning model operators and non-Markovian terms at the same

$\Rightarrow$ Dynamics mixed, same issues as learning from projected states

Build on re-projection to learn non-Markovian terms stage-wise

- Sample trajectories of length $r+1$ with re-projection

$$\bar{\boldsymbol{X}}^{(0)}, \ldots, \bar{\boldsymbol{X}}^{(K-1)} \in \mathbb{R}^{n \times r+1}$$

- Infer Markovian reduced model $\hat{\boldsymbol{f}}_1$ from one-step trajectories

$$\bar{\boldsymbol{X}}_1^{(i)} = [\bar{\boldsymbol{x}}_0^{(i)}, \bar{\boldsymbol{x}}_1^{(i)}], \qquad i = 0, \ldots, K-1$$

- Simulate $\hat{\boldsymbol{f}}_1$ to obtain

$$\hat{\boldsymbol{X}}_2^{(i)} = [\hat{\boldsymbol{x}}_0^{(i)}, \hat{\boldsymbol{x}}_1^{(i)}, \hat{\boldsymbol{x}}_2^{(i)}], \qquad i = 0, \ldots, K-1$$

- Fit parameter $\boldsymbol{\theta}_1$ of non-Markovian term $\hat{\boldsymbol{\Delta}}_1^{\theta_1}$ to difference

$$\min_{\boldsymbol{\theta_1} \in \Theta} \sum_{i=0}^{K-1} \|\bar{\boldsymbol{x}}_2^{(i)} - \hat{\boldsymbol{x}}_2^{(i)} - \hat{\boldsymbol{\Delta}}_1^{(\boldsymbol{\theta_1})}(\bar{\boldsymbol{x}}_0^{(i)}, \boldsymbol{u}_i)\|_2^2$$

- Repeat this r times to learn $\hat{\boldsymbol{f}}_r$ with lag r

NonM: Learning non-Markovian terms

Parametrization of non-Markovian terms

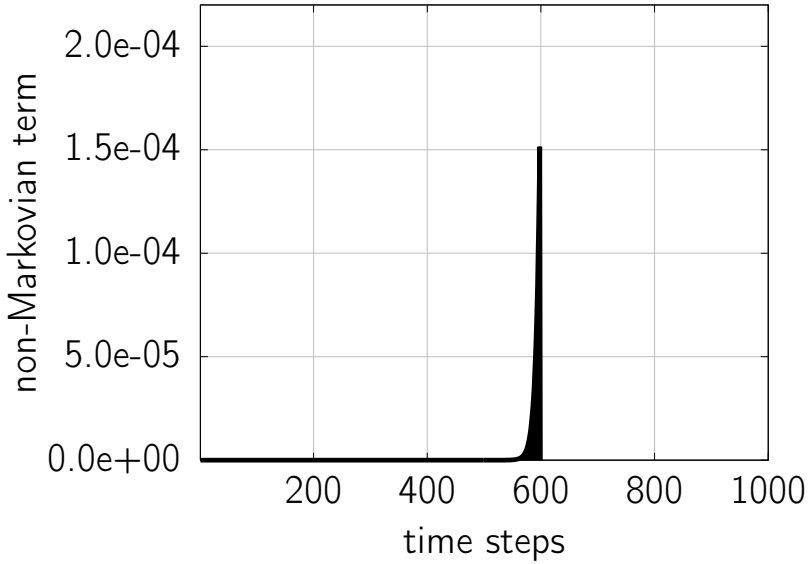
- Set $\boldsymbol{\theta}_i = [\boldsymbol{D}_i, \boldsymbol{E}_i]$ with $\boldsymbol{D}_i \in \mathbb{R}^{n \times n}$ and $\boldsymbol{E}_i \in \mathbb{R}^{n \times p}$
- Non-Markovian term is

$$\hat{\boldsymbol{\Delta}}_i^{(\boldsymbol{\theta}_i)}(\hat{\boldsymbol{x}}_{k-1}, \ldots, \hat{\boldsymbol{x}}_{k-i+1}, \boldsymbol{u}_k, \ldots, \boldsymbol{u}_{k-i+1}) = \boldsymbol{D}_i \hat{\boldsymbol{x}}_{k-i+1} + \boldsymbol{E}_i \boldsymbol{u}_{k-i+1}$$

- Other parametrizations with higher-order terms and neural networks

Choosing maximal lag

- Assumption (observation) is that non-Markovian term of system has small support
- Need to go back in time only a few steps
- Lag r can be chosen small

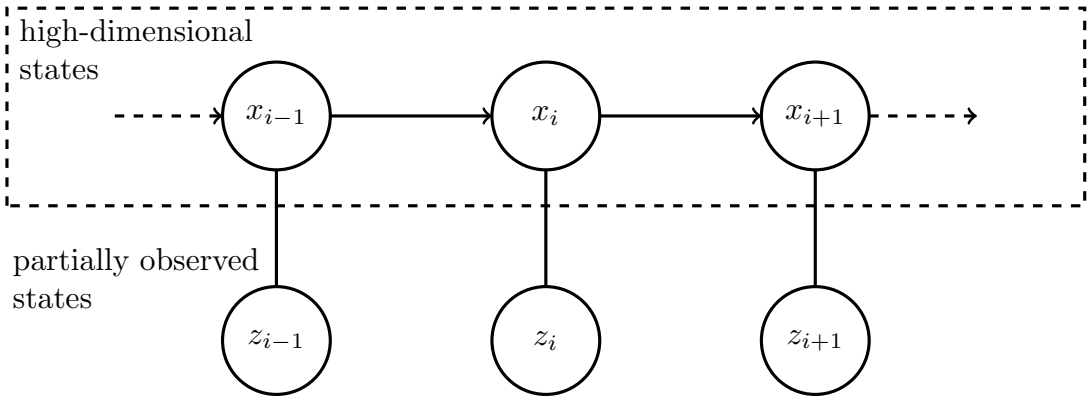


NonM: Learning from partially observed states

Partially observed state trajectories

- Unknown selection operator $\boldsymbol{S} \in \{0,1\}^{N_s \times N}$ with $N_s < N$ and

$$\boldsymbol{z}_k = \boldsymbol{S}\boldsymbol{x}_k$$



- Learn models from trajectory $\boldsymbol{Z} = [\boldsymbol{z}_0, \ldots, \boldsymbol{z}_{K-1}]$ instead of $\boldsymbol{X} = [\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{K-1}]$

- Apply POD (PCA) to $\boldsymbol{Z}$ to find basis matrix $\boldsymbol{V}$ of subspace $\mathcal{V}$ of $\mathbb{R}^{N_s}$

Non-Markovian terms to compensate unobserved state components

- Mori-Zwanzig formalism applies
- Non-Markovian terms compensate unobserved components

NonM: Burgers': Burgers' example

Viscous Burgers' equation

$$\frac{\partial}{\partial t}x(\omega, t; \mu) + x(\omega, t; \mu)\frac{\partial}{\partial \omega}x(\omega, t; \mu) - \mu\frac{\partial^2}{\partial \omega^2}x(\omega, t; \mu) = 0$$

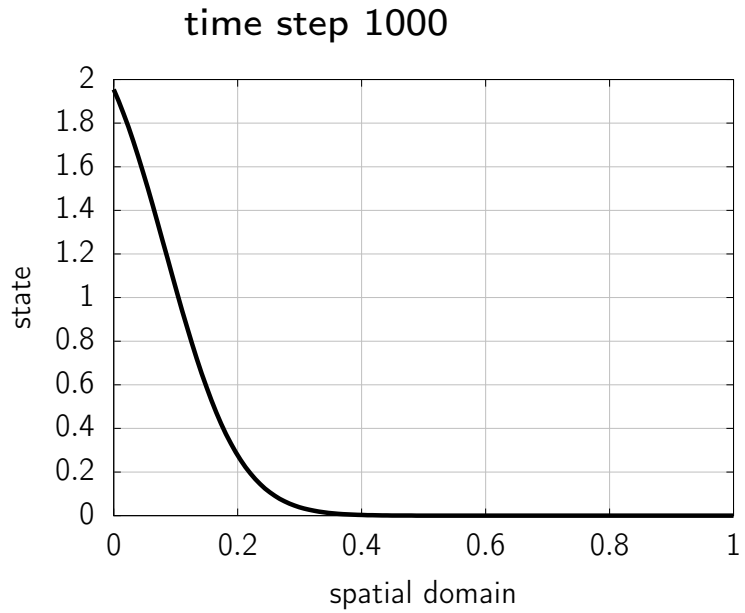
- Spatial, time, and parameter domain

$$\omega \in [0, 1]\,, \quad t \in [0, 1]\,, \quad \mu \in [0.1, 1]$$

- Dirichlet boundary conditions

$$x(0, t; \mu) = -x(1, t; \mu) = u(t)$$

- Discretize with forward Euler
- Time step size is $\delta t = 10^{-4}$



Operator inference

- Training data are 2 trajectories with random inputs
- Infer operators for 10 equidistant parameters in $[0.1, 1]$
- Interpolate inferred operators at 7 test parameters and predict

NonM: Burgers': Burgers' example

Viscous Burgers' equation

$$\frac{\partial}{\partial t}x(\omega, t; \mu) + x(\omega, t; \mu)\frac{\partial}{\partial \omega}x(\omega, t; \mu) - \mu\frac{\partial^2}{\partial \omega^2}x(\omega, t; \mu) = 0$$

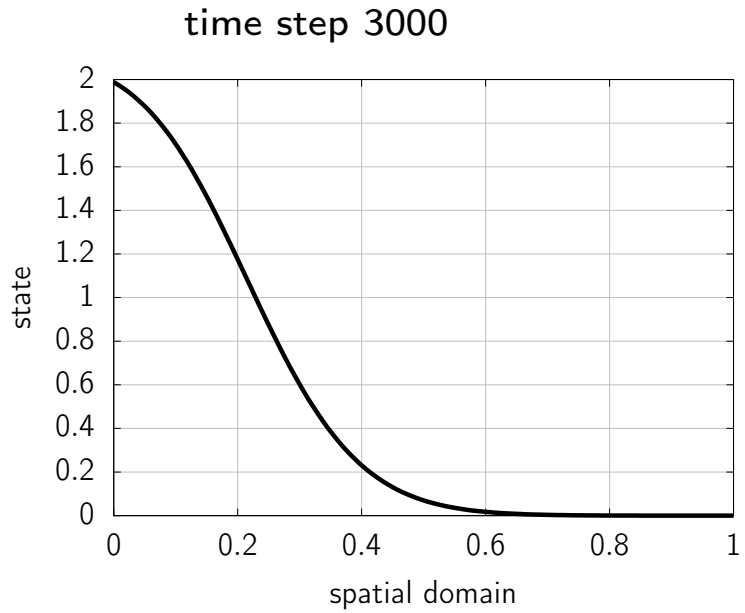
- Spatial, time, and parameter domain

$$\omega \in [0, 1]\,, \quad t \in [0, 1]\,, \quad \mu \in [0.1, 1]$$

- Dirichlet boundary conditions

$$x(0, t; \mu) = -x(1, t; \mu) = u(t)$$

- Discretize with forward Euler
- Time step size is $\delta t = 10^{-4}$



Operator inference

- Training data are 2 trajectories with random inputs
- Infer operators for 10 equidistant parameters in $[0.1, 1]$
- Interpolate inferred operators at 7 test parameters and predict

NonM: Burgers': Burgers' example

Viscous Burgers' equation

$$\frac{\partial}{\partial t}x(\omega, t; \mu) + x(\omega, t; \mu)\frac{\partial}{\partial \omega}x(\omega, t; \mu) - \mu\frac{\partial^2}{\partial \omega^2}x(\omega, t; \mu) = 0$$

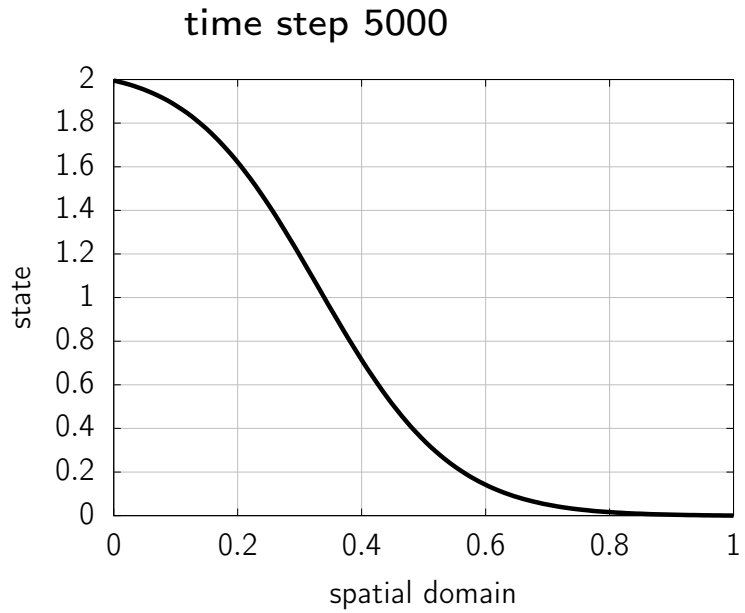
- Spatial, time, and parameter domain

$$\omega \in [0, 1]\,, \quad t \in [0, 1]\,, \quad \mu \in [0.1, 1]$$

- Dirichlet boundary conditions

$$x(0, t; \mu) = -x(1, t; \mu) = u(t)$$

- Discretize with forward Euler
- Time step size is $\delta t = 10^{-4}$



Operator inference

- Training data are 2 trajectories with random inputs
- Infer operators for 10 equidistant parameters in $[0.1, 1]$
- Interpolate inferred operators at 7 test parameters and predict

NonM: Burgers': Burgers' example

Viscous Burgers' equation

$$\frac{\partial}{\partial t}x(\omega, t; \mu) + x(\omega, t; \mu)\frac{\partial}{\partial \omega}x(\omega, t; \mu) - \mu\frac{\partial^2}{\partial \omega^2}x(\omega, t; \mu) = 0$$

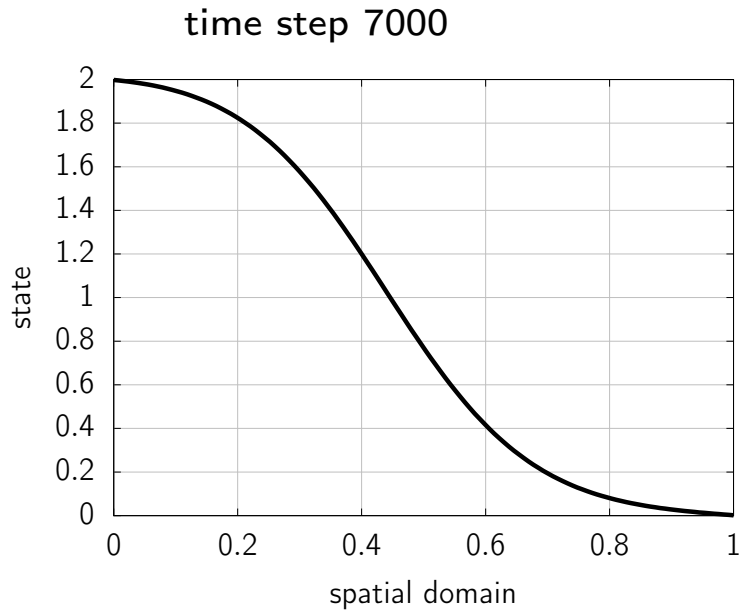
- Spatial, time, and parameter domain

$$\omega \in [0, 1]\,, \quad t \in [0, 1]\,, \quad \mu \in [0.1, 1]$$

- Dirichlet boundary conditions

$$x(0, t; \mu) = -x(1, t; \mu) = u(t)$$

- Discretize with forward Euler
- Time step size is $\delta t = 10^{-4}$



Operator inference

- Training data are 2 trajectories with random inputs
- Infer operators for 10 equidistant parameters in $[0.1, 1]$
- Interpolate inferred operators at 7 test parameters and predict

NonM: Burgers': Burgers' example

Viscous Burgers' equation

$$\frac{\partial}{\partial t}x(\omega, t; \mu) + x(\omega, t; \mu)\frac{\partial}{\partial \omega}x(\omega, t; \mu) - \mu\frac{\partial^2}{\partial \omega^2}x(\omega, t; \mu) = 0$$

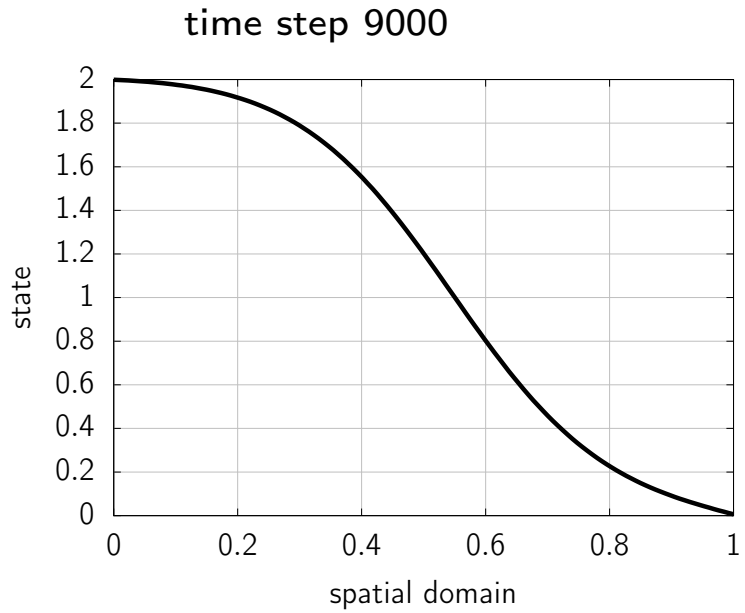
- Spatial, time, and parameter domain

$$\omega \in [0, 1]\,, \quad t \in [0, 1]\,, \quad \mu \in [0.1, 1]$$

- Dirichlet boundary conditions

$$x(0, t; \mu) = -x(1, t; \mu) = u(t)$$

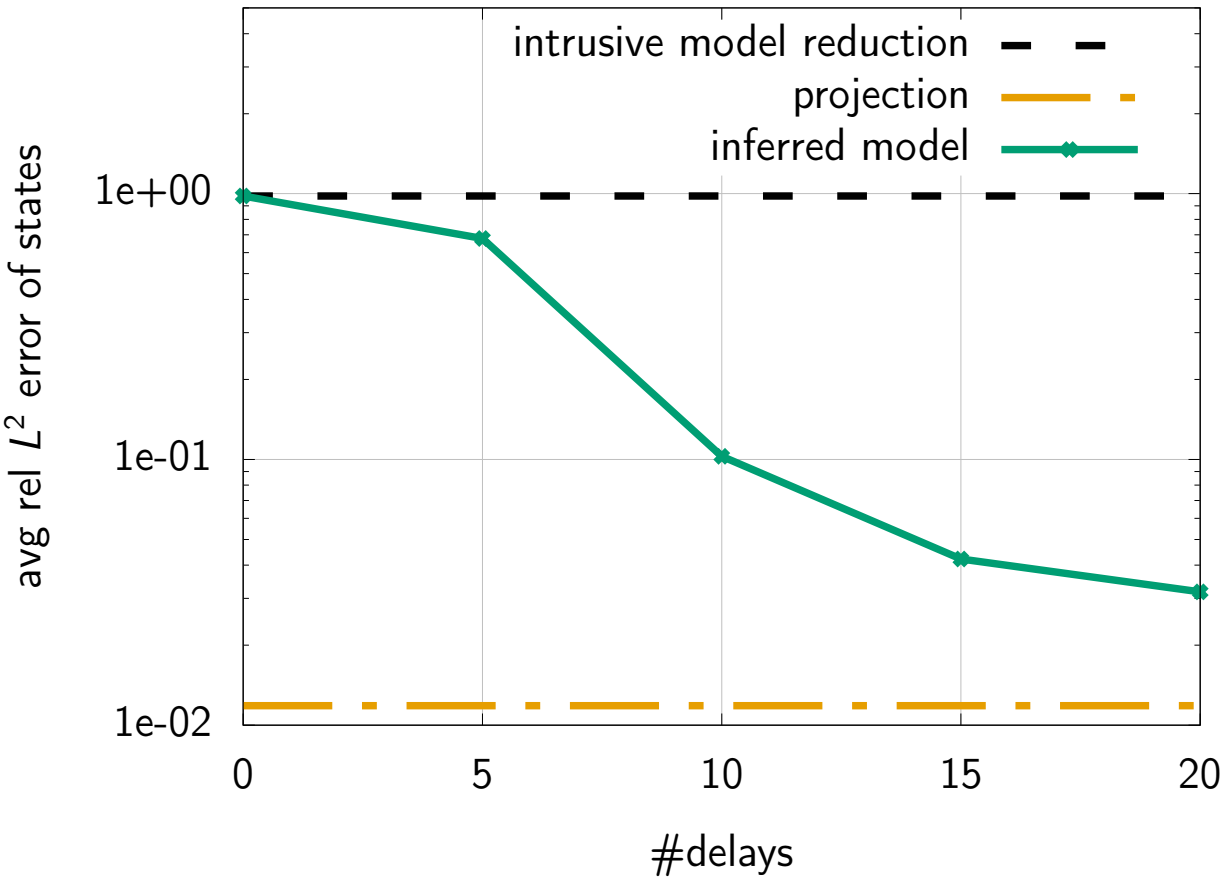
- Discretize with forward Euler
- Time step size is $\delta t = 10^{-4}$



Operator inference

- Training data are 2 trajectories with random inputs
- Infer operators for 10 equidistant parameters in $[0.1, 1]$
- Interpolate inferred operators at 7 test parameters and predict

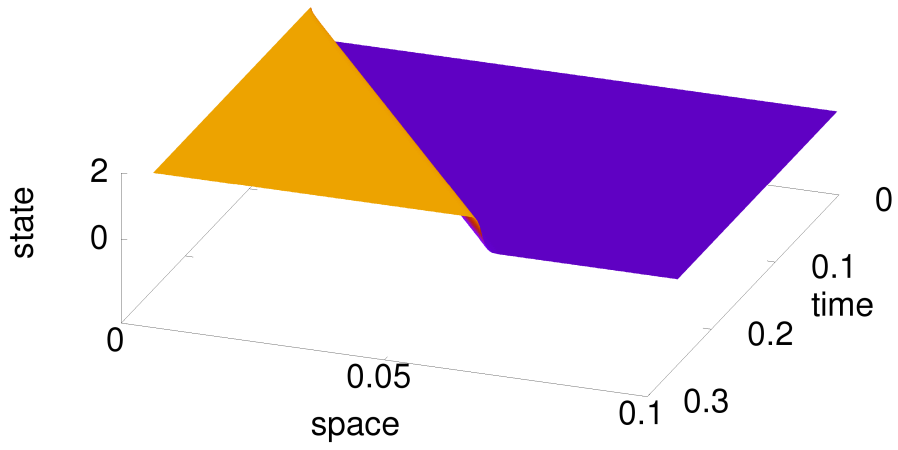
NonM: Burgers': Partial observations



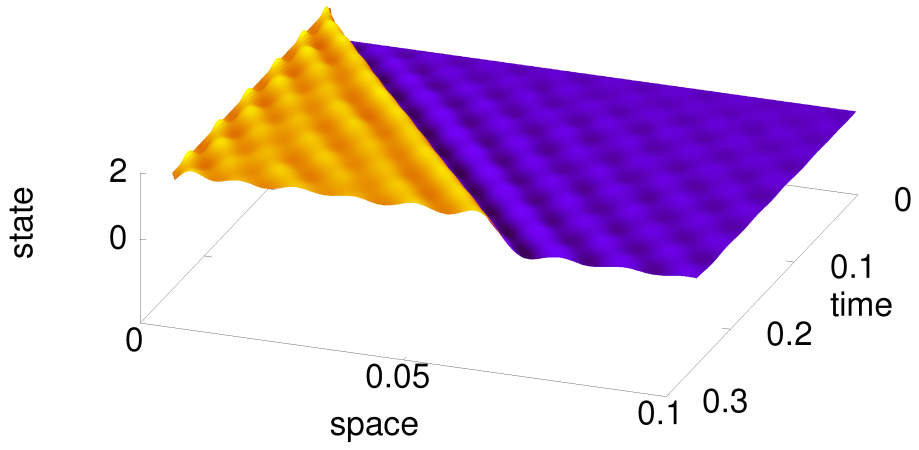
Observe only about 50% of all state components

- Linear time-delay terms with stage-wise re-projection
- Reduces error of inferred model by more than one order of magnitude

NonM: Burgers': Shock formation



(a) ground truth (full model)

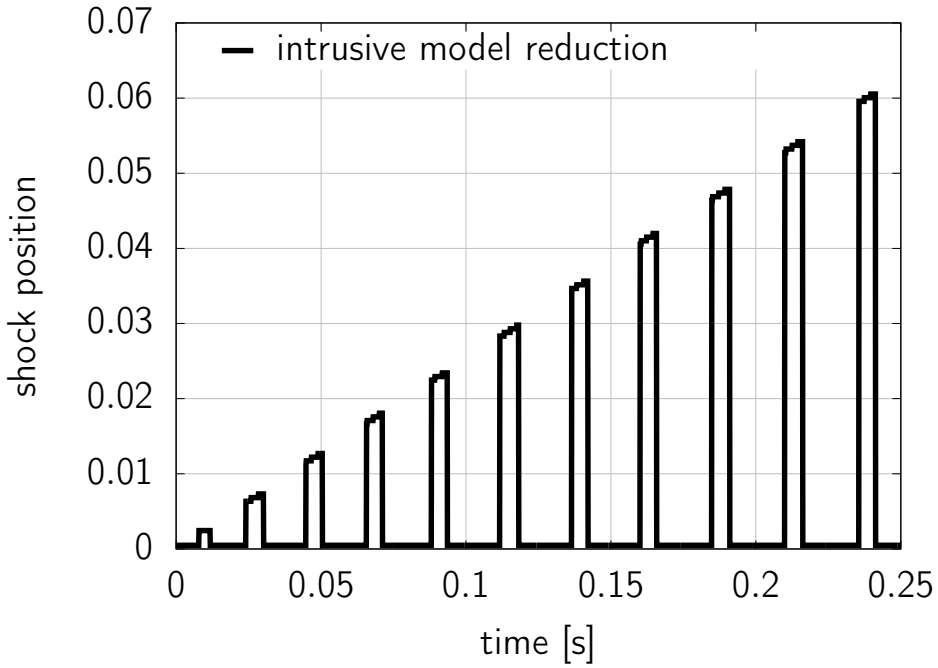


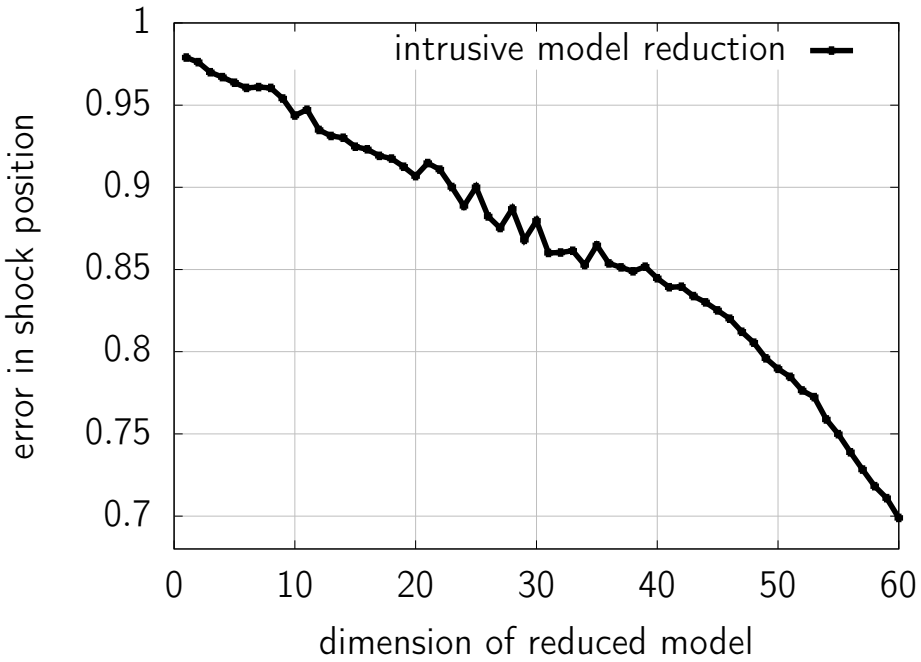
(b) intrusive model reduction

Modify coefficients of Burgers' equation to obtain solution with shock

- Solutions with shocks are challenging to reduce with model reduction
- Here, reduced model from intrusive model reduction has oscillatory error

NonM: Burgers': Capturing shock position

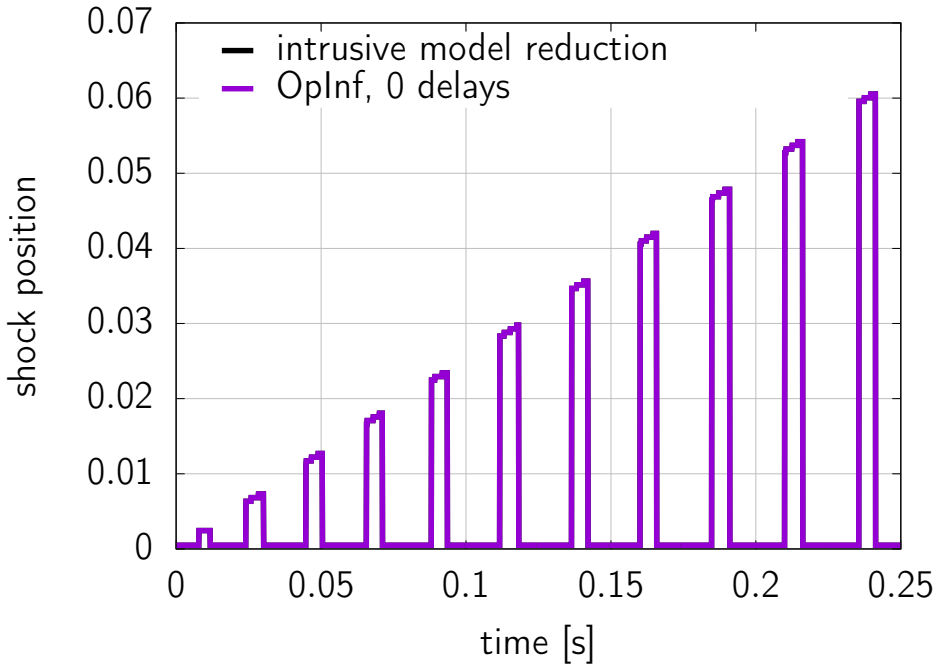


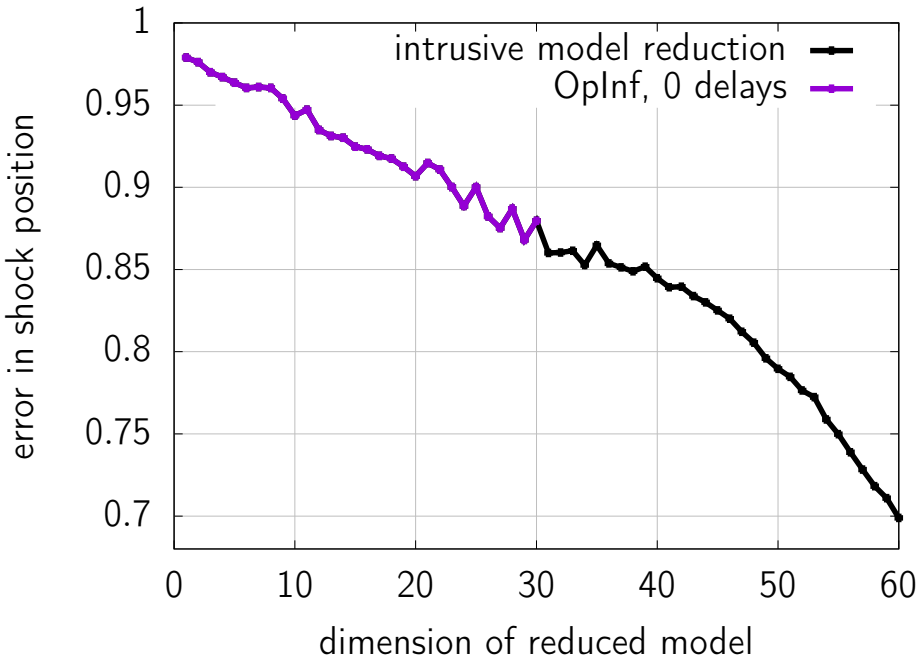


Learn time-delay terms stage-wise with (re-)re-projection

- Learn linear time-delay corrections
- In this example, time delay of order 4 sufficient to capture shock
- Higher-order time-delay terms learned in, e.g., [Pan, Duraisamy, 2018]

NonM: Burgers': Capturing shock position

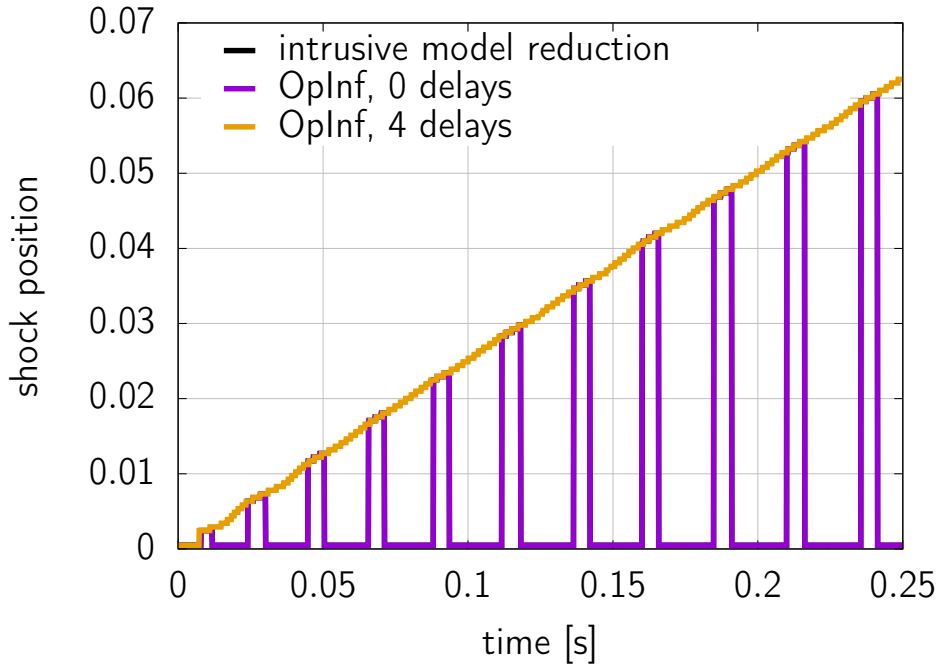


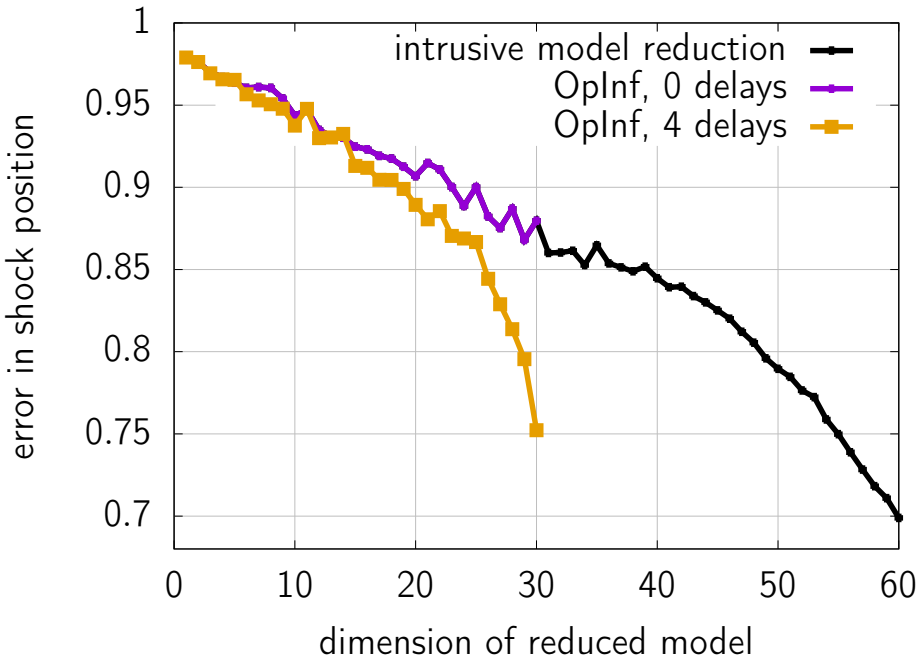


Learn time-delay terms stage-wise with (re-)re-projection

- Learn linear time-delay corrections
- In this example, time delay of order 4 sufficient to capture shock
- Higher-order time-delay terms learned in, e.g., [Pan, Duraisamy, 2018]

NonM: Burgers': Capturing shock position

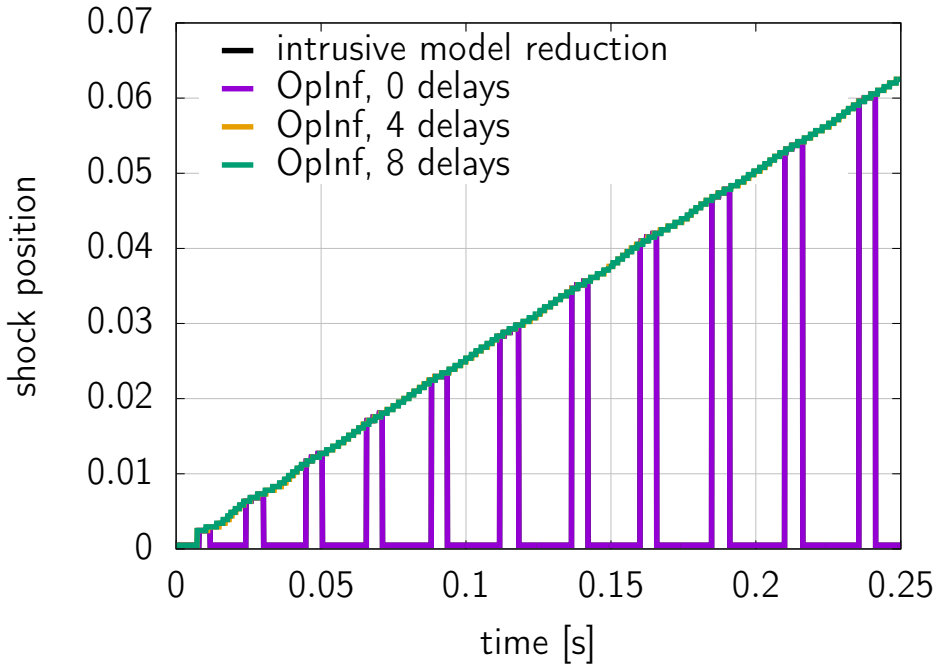


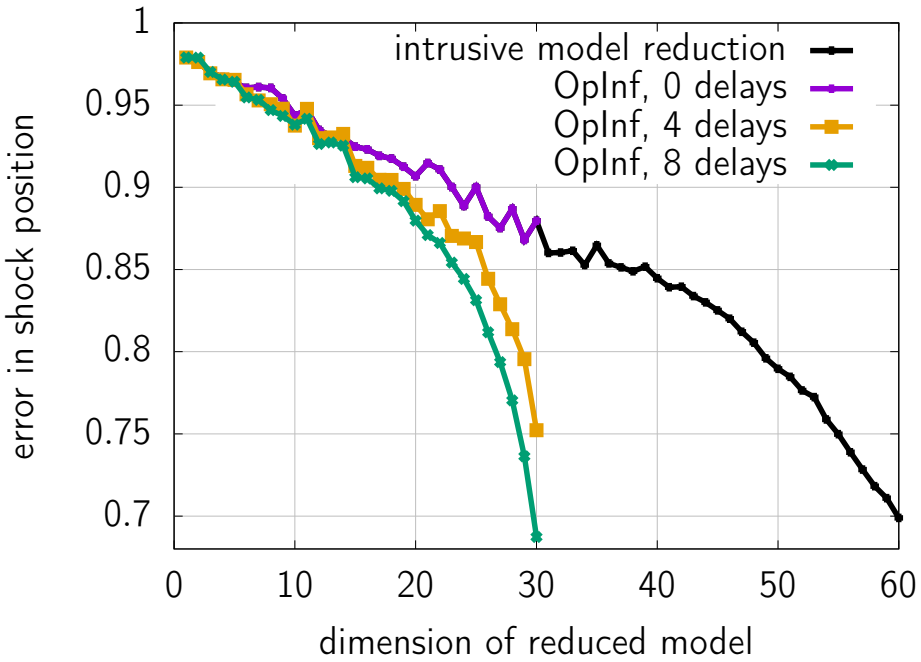


Learn time-delay terms stage-wise with (re-)re-projection

- Learn linear time-delay corrections
- In this example, time delay of order 4 sufficient to capture shock
- Higher-order time-delay terms learned in, e.g., [Pan, Duraisamy, 2018]

NonM: Burgers': Capturing shock position

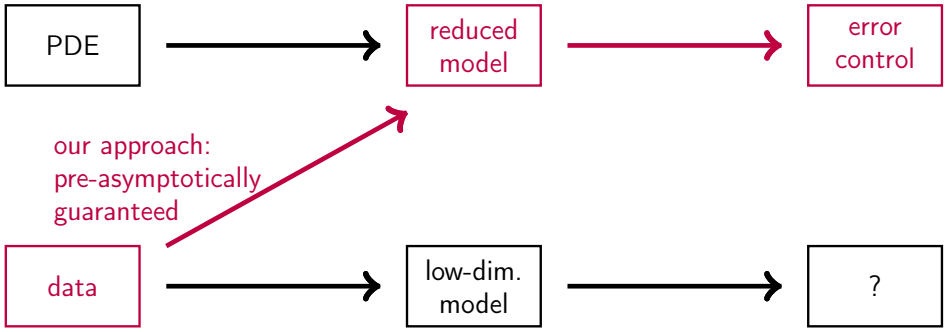


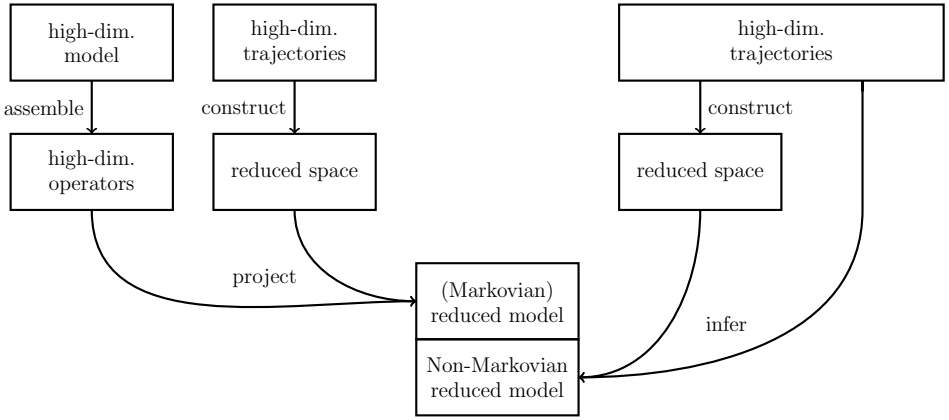


Learn time-delay terms stage-wise with (re-)re-projection

- Learn linear time-delay corrections
- In this example, time delay of order 4 sufficient to capture shock
- Higher-order time-delay terms learned in, e.g., [Pan, Duraisamy, 2018]

Conclusions





Learning dynamical-system models from data with error guarantees

- Operator inference *exactly* recovers reduced models from data
- Generating the right data is key to learning reduced models in our case
- Pre-asymptotic guarantees (finite data) under certain conditions
- Going beyond reduced models by learning non-Markovian corrections

References: https://cims.nyu.edu/∼pehersto

- Uy, P., *Pre-asymptotic error bounds for low-dimensional models learned from systems governed by linear parabolic partial differential equations with control inputs*, *in preparation*, 2020.
- P., Sampling low-dimensional Markovian dynamics for pre-asymptotically recovering reduced models from data with operator inference. arXiv:1908.11233, 2019.
- P., Willcox, Data-driven operator inference for nonintrusive projection-based model reduction. Computer Methods in Applied Mechanics and Engineering, 306:196-215, 2016.