Learning dynamical systems from data: basics

Ionuț-Gabriel Farcaș[1], Rayomand P. Gundevia[2], Ramakanth Munipalli[3], and Karen Willcox[1]

Workshop on Data-driven & Reduced Order Modeling for Multi-Scale Problems

[1]Oden Institute for Computational Engineering and Sciences, The University of Texas at Austin, Austin, TX
[2]Jacobs Engineering Group, Inc., Edwards Air Force Base, CA
[3]AFRL Combustion Devices (RQRC), Edwards Air Force Base, CA

August 30, 2023

The code for the tutorial is available at
`https://github.com/ionutfarcas/OpInf-tutorial`

Learning dynamical systems from data: general idea

Given (1) a physics/engineering system, and (2) a training data set (experimental or simulation)

Construct a surrogate/reduced model that recovers the given data and provides a predictive capability to rapidly simulate unseen conditions

A taxonomy of approaches [Ghattas, Willcox; Acta Numerica, 2021]

Black-box	Glass-box/non-intrusive	Intrusive
• purely data-driven • generally don't exploit any knowledge about the physics of the problem • require large amounts of high-quality data • can be unsuitable for large-scale applications	• generally data-driven • they look inside the box but do not modify it • exploit the knowledge of the underlying model, physical principles etc. • can be suitable for large-scale applications	• they look inside the box and modify it (e.g., modify the underlying code) • rich mathematical theory • can be suitable for large-scale applications

Learning dynamical systems from data: general idea

Given (1) a physics/engineering system, and (2) a training data set (experimental or simulation)

Construct a surrogate/reduced model that recovers the given data and provides a predictive capability to rapidly simulate unseen conditions

A taxonomy of approaches [Ghattas, Willcox; Acta Numerica, 2021]

Black-box

- purely data-driven
- generally don't exploit any knowledge about the physics of the problem
- require large amounts of high-quality data
- can be unsuitable for large-scale applications

Glass-box/non-intrusive

- generally data-driven
- they look inside the box but do not modify it
- exploit the knowledge of the underlying model, physical principles etc.
- can be suitable for large-scale applications

Intrusive

- they look inside the box and modify it (e.g., modify the underlying code)
- rich mathematical theory
- can be suitable for large-scale applications

Learning dynamical systems from data: general idea

Given (1) a physics/engineering system, and (2) a training data set (experimental or simulation)

Construct a surrogate/reduced model that recovers the given data and provides a predictive capability to rapidly simulate unseen conditions

A taxonomy of approaches [Ghattas, Willcox; Acta Numerica, 2021]

Black-box

- purely data-driven
- generally don't exploit any knowledge about the physics of the problem
- require large amounts of high-quality data
- can be unsuitable for large-scale applications

Glass-box/non-intrusive

- generally data-driven
- they look inside the box but do not modify it
- exploit the knowledge of the underlying model, physical principles etc.
- can be suitable for large-scale applications

Intrusive

- they look inside the box and modify it (e.g., modify the underlying code)
- rich mathematical theory
- can be suitable for large-scale applications

Learning dynamical systems from data: general idea

Given (1) a physics/engineering system, and (2) a training data set (experimental or simulation)

Construct a surrogate/reduced model that recovers the given data and provides a predictive capability to rapidly simulate unseen conditions

A taxonomy of approaches [Ghattas, Willcox; Acta Numerica, 2021]

Black-box

- purely data-driven
- generally don't exploit any knowledge about the physics of the problem
- require large amounts of high-quality data
- can be unsuitable for large-scale applications

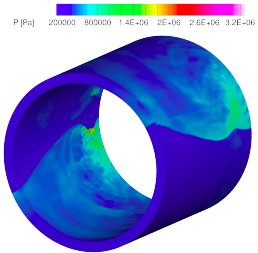
Glass-box/non-intrusive

- generally data-driven
- they look inside the box but do not modify it
- exploit the knowledge of the underlying model, physical principles etc.
- can be suitable for large-scale applications

Intrusive

- they look inside the box and modify it (e.g., modify the underlying code)
- rich mathematical theory
- can be suitable for large-scale applications

Learning dynamical systems from data: goals



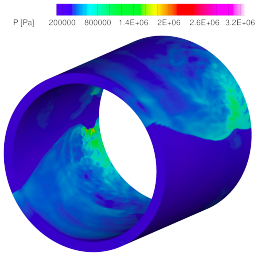
Why reduced models?

- to enable rapid/real-time state predictions
- to enable downstream parametric tasks such as design optimization or uncertainty quantification

Computational challenges

- the high-fidelity simulations are large-scale and computationally very expensive (e.g., $\mathcal{O}(10^6)$ core-hours on supercomputers)
- the resulting training data sets are often sparse
- they comprise down-sampled time instants from the high-fidelity simulation
- only few parametric instances can realistically be simulated to generate training data

Learning dynamical systems from data: goals



Why reduced models?

- to enable rapid/real-time state predictions
- to enable downstream parametric tasks such as design optimization or uncertainty quantification

Computational challenges

- the high-fidelity simulations are large-scale and computationally very expensive (e.g., $\mathcal{O}(10^6)$ core-hours on supercomputers)
- the resulting training data sets are often sparse
- they comprise down-sampled time instants from the high-fidelity simulation
- only few parametric instances can realistically be simulated to generate training data

The Operator Inference problem

Given (1) a physics/engineering system with known governing equations, and (2) a set of data in the form of state snapshots (experimental or simulation)

Infer a reduced-order model that recovers the given data and provides a predictive capability to rapidly simulate unseen conditions

$$\text{argmin}_{\mathbf{O}} \|\mathbf{D}\mathbf{O} - \mathbf{R}\|$$

- $\mathbf{O}$: low-dimensional operators that define the reduced model
- $\mathbf{D}, \mathbf{R}$ data matrix/forcing from simulation and/or experimental data

We use:

- the physics to define the structured form of the model we seek
- projection-based model reduction to cast the inference in a reduced coordinate space and to provide error estimates in some settings
- numerical linear algebra to achieve efficient scalable algorithms
- inverse theory to analyze the structure of the resulting problem and treat it numerically

The reduced model form is inspired by classical intrusive physics-based model reduction but the operators are learned directly from data

The Operator Inference problem

Given (1) a physics/engineering system with known governing equations, and (2) a set of data in the form of state snapshots (experimental or simulation)

Infer a reduced-order model that recovers the given data and provides a predictive capability to rapidly simulate unseen conditions

$$\text{argmin}_{\mathbf{O}} \|\mathbf{DO} - \mathbf{R}\|$$

- $\mathbf{O}$: low-dimensional operators that define the reduced model
- $\mathbf{D}, \mathbf{R}$ data matrix/forcing from simulation and/or experimental data

We use:

- the physics to define the structured form of the model we seek
- projection-based model reduction to cast the inference in a reduced coordinate space and to provide error estimates in some settings
- numerical linear algebra to achieve efficient scalable algorithms
- inverse theory to analyze the structure of the resulting problem and treat it numerically

The reduced model form is inspired by classical intrusive physics-based model reduction but the operators are learned directly from data

The Operator Inference problem

Given (1) a physics/engineering system with known governing equations, and (2) a set of data in the form of state snapshots (experimental or simulation)

Infer a reduced-order model that recovers the given data and provides a predictive capability to rapidly simulate unseen conditions

$$\text{argmin}_{\mathbf{O}} \|\mathbf{DO} - \mathbf{R}\|$$

- $\mathbf{O}$: low-dimensional operators that define the reduced model
- $\mathbf{D}, \mathbf{R}$ data matrix/forcing from simulation and/or experimental data

We use:

- the physics to define the structured form of the model we seek
- projection-based model reduction to cast the inference in a reduced coordinate space and to provide error estimates in some settings
- numerical linear algebra to achieve efficient scalable algorithms
- inverse theory to analyze the structure of the resulting problem and treat it numerically

The reduced model form is inspired by classical intrusive physics-based model reduction but the operators are learned directly from data

Operator inference: general idea

- starting point: a physics-based model, typically described by PDEs or ODEs
- variable transformations that expose polynomial structure in the model
- lens of projection to define the form of a structure-preserving low-dimensional model

} define the structure of the reduced model

Operator inference learning problem

- non-intrusive learning by inferring reduced model operators from data

$$\underset{\hat{\mathbf{A}},\hat{\mathbf{H}}}{\operatorname{argmin}} \left\| \hat{\mathbf{S}}^{\top}\hat{\mathbf{A}}^{\top} + \left(\hat{\mathbf{S}}\otimes\hat{\mathbf{S}}\right)^{\top}\hat{\mathbf{H}}^{\top} - \left(\frac{\mathrm{d}\hat{\mathbf{S}}}{\mathrm{d}t}\right)^{\top} \right\|_F^2 + \text{regularization}$$

- $\hat{\mathbf{A}}, \hat{\mathbf{H}}$ are the low-dimensional operators define the reduced model as a discrete system
- $\hat{\mathbf{S}}$ is obtained by projecting the snapshots generated by the high-fidelity code
- minimum residual formulation leads to linear least-squares minimization
- regularization is key to reduce overfitting, account for model misspecification etc.

Operator inference: general idea

- starting point: a physics-based model, typically described by PDEs or ODEs
- variable transformations that expose polynomial structure in the model
- lens of projection to define the form of a structure-preserving low-dimensional model

} define the structure of the reduced model

Operator inference learning problem

- non-intrusive learning by inferring reduced model operators from data

$$\underset{\hat{\mathbf{A}},\hat{\mathbf{H}}}{\operatorname{argmin}} \left\| \hat{\mathbf{S}}^{\top}\hat{\mathbf{A}}^{\top} + \left(\hat{\mathbf{S}}\otimes\hat{\mathbf{S}}\right)^{\top}\hat{\mathbf{H}}^{\top} - \left(\frac{\mathrm{d}\hat{\mathbf{S}}}{\mathrm{d}t}\right)^{\top} \right\|_F^2 + \text{regularization}$$

  - $\hat{\mathbf{A}}, \hat{\mathbf{H}}$ are the low-dimensional operators define the reduced model as a discrete system
  - $\hat{\mathbf{S}}$ is obtained by projecting the snapshots generated by the high-fidelity code
  - minimum residual formulation leads to linear least-squares minimization
  - regularization is key to reduce overfitting, account for model misspecification etc.

Setup for high-fidelity simulations I

Let $\Omega \subset \mathbb{R}^d$ denote the physical domain and let $[t_{\text{init}}, t_{\text{final}}]$, with t_{init} the initial time and t_{final} the final time denote the time domain. The nonlinear PDE

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

where

$$s = \begin{pmatrix} s_1(x,t) \\ s_2(x,t) \\ \vdots \\ s_{n_s}(x,t) \end{pmatrix} \quad \text{and} \quad f(s) = \begin{pmatrix} f_1(s) \\ f_2(s) \\ \vdots \\ f_{n_s}(s) \end{pmatrix}$$

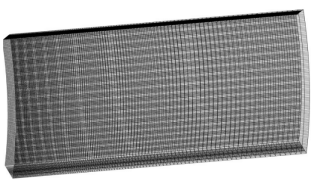
defines a dynamical system for the n_s-dimensional vector state field s, where $s_j : \Omega \times [t_{\text{init}}, t_{\text{final}}] \to \mathbb{R}$ for $j = 1, 2, \ldots, n_s$, and the nonlinear function f maps the state field to its time derivative.

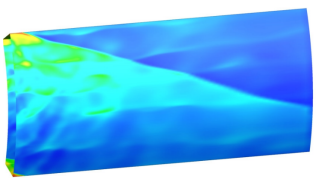
Setup for high-fidelity simulations II

We consider a semi-discrete model that depends on the spatially discretized state vector $\mathbf{s} \in \mathbb{R}^{n_x n_s}$ at some collection of n_x spatial points (e.g., in a finite difference, finite element or finite volume discretization), where n_x is typically (very) large. This results in a large-scale system of nonlinear ODEs

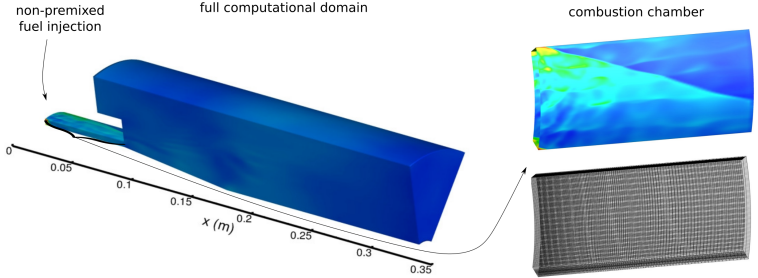
$$\frac{\mathrm{d}\mathbf{s}}{\mathrm{d}t} = \mathbf{f}(t, \mathbf{s}), \quad \mathbf{s}(t_{\text{init}}) = \mathbf{s}_{\text{init}},$$

where $\mathbf{f} : [t_{\text{init}}, t_{\text{final}}] \times \mathbb{R}^{n_x n_s} \to \mathbb{R}^{n_x n_s}$ discretizes f, and $\mathbf{s}_{\text{init}}$ is the initial condition.





Example: modeling a $45°$-degree sector of the combustion chamber of a rotating detonation rocket engine [Lietz et al., 2019]



- 45-degree sector of the full rotating detonation rocket engine (RDRE)
- large-eddy simulation (LES) of the reactive, viscous 3D Navier-Stokes equations
- modified version of the Westbrook-Dryer mechanism for the purposes of methane-oxygen detonation (originally developed for methane-air combustion)
- non-premixed fuel injection (gaseous methane and oxygen)
- the sector has nine injectors (72 in total for the full engine)
- a single wave travels stably in one direction

Steps to perform Operator Inference

1. Acquiring training data

- we collect $n_t \in \mathbb{N}$ time instants $\mathbf{s}(t)$ from the high-fidelity simulation code over training time horizon $[t_{\text{init}}, t_{\text{train}}]$, where $t_{\text{train}} < t_{\text{final}}$ into a snapshot matrix

$$\mathbf{S} = \begin{bmatrix} | & | & & | \\ \mathbf{s}_1 & \mathbf{s}_2 & \dots & \mathbf{s}_{n_t} \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{n_x n_s \times n_t},$$

  where on the kth column we have the state solution (snapshot) at time t_k, i.e., $\mathbf{s}_k = \mathbf{s}(t_k)$

Remarks

- the training data set is usually saved to disk while performing the high-fidelity simulation
- for computationally expensive large-scale applications, the available training data sets are generally sparse and down-sampled
- even when the data sets are sparse, the disk size requirements for storage can be (very) large

Steps to perform Operator Inference

1. Acquiring training data

- we collect $n_t \in \mathbb{N}$ time instants $\mathbf{s}(t)$ from the high-fidelity simulation code over training time horizon $[t_{\text{init}}, t_{\text{train}}]$, where $t_{\text{train}} < t_{\text{final}}$ into a snapshot matrix

$$\mathbf{S} = \begin{bmatrix} | & | & & | \\ \mathbf{s}_1 & \mathbf{s}_2 & \dots & \mathbf{s}_{n_t} \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{n_x n_s \times n_t},$$

  where on the kth column we have the state solution (snapshot) at time t_k, i.e., $\mathbf{s}_k = \mathbf{s}(t_k)$

Remarks

- the training data set is usually saved to disk while performing the high-fidelity simulation
- for computationally expensive large-scale applications, the available training data sets are generally sparse and down-sampled
- even when the data sets are sparse, the disk size requirements for storage can be (very) large

Steps to perform Operator Inference

2. Training data manipulation: lifting transformations

- the original OpInf formulation [Peherstorfer, Willcox; CMAME, 2016] targets systems with polynomial nonlinearities
- for more general types of nonlinearities/higher-order polynomial terms, lifting transformations can be used to expose (sometimes approximate) (lower degree) polynomial structure in the lifted governing equations [Qian et al.; Physica D, 2020]
- let $\mathcal{T} : \mathbb{R}^{n_s} \to \mathbb{R}^{n_w}$ denote such a transformation with $n_w \geq n_s$ transformed variables such that the given nonlinear PDE is polynomial in $w = \mathcal{T}(s)$[1]
- for simplicity, let $n = n_x n_w$ denote the (large) state dimension after lifting the original variables
- for each column in the given data matrix $\mathbf{S}$, we apply $\mathcal{T}$ point-wisely to obtain

$$\mathbf{W} = \begin{bmatrix} | & | & & | \\ \mathbf{w}_1 & \mathbf{w}_2 & \dots & \mathbf{w}_{n_t} \\ | & | & & | \end{bmatrix} = \begin{bmatrix} | & | & & | \\ \mathbf{T}(\mathbf{s}_1) & \mathbf{T}(\mathbf{s}_2) & \dots & \mathbf{T}(\mathbf{s}_{n_t}) \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{n \times n_t}$$

[1] please refer to [Qian et al.; Physica D, 2020] for the mathematical details

Steps to perform Operator Inference

2. Training data manipulation: lifting transformations

- the original OpInf formulation [Peherstorfer, Willcox; CMAME, 2016] targets systems with polynomial nonlinearities
- for more general types of nonlinearities/higher-order polynomial terms, lifting transformations can be used to expose (sometimes approximate) (lower degree) polynomial structure in the lifted governing equations [Qian et al.; Physica D, 2020]
- let $\mathcal{T} : \mathbb{R}^{n_s} \to \mathbb{R}^{n_w}$ denote such a transformation with $n_w \geq n_s$ transformed variables such that the given nonlinear PDE is polynomial in $w = \mathcal{T}(s)$[1]
- for simplicity, let $n = n_x n_w$ denote the (large) state dimension after lifting the original variables
- for each column in the given data matrix $\mathbf{S}$, we apply $\mathcal{T}$ point-wisely to obtain

$$\mathbf{W} = \begin{bmatrix} | & | & & | \\ \mathbf{w}_1 & \mathbf{w}_2 & \dots & \mathbf{w}_{n_t} \\ | & | & & | \end{bmatrix} = \begin{bmatrix} | & | & & | \\ \mathbf{T}(\mathbf{s}_1) & \mathbf{T}(\mathbf{s}_2) & \dots & \mathbf{T}(\mathbf{s}_{n_t}) \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{n \times n_t}$$

[1] please refer to [Qian et al.; Physica D, 2020] for the mathematical details

Example lifting transformation

One-dimensional Euler equations in conservative variables formulation:

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho u\\ E\end{pmatrix} = -\frac{\partial}{\partial x}\begin{pmatrix}\rho u\\ \rho u^2 + p\\ (E+p)u\end{pmatrix}, \quad E = \frac{p}{\gamma - 1} + \frac{1}{2}\rho u^2,$$

with state $s = [\rho\ \rho u\ E]^\top$ comprising the density ρ, specific momentum ρu, and energy E.
This representation contains several nonlinear terms that are not quadratic in the conservative state.

For constant heat capacity ratio γ, the transformation $\mathcal{T} : \begin{pmatrix}\rho\\ \rho u\\ E\end{pmatrix} \to \begin{pmatrix}u\\ p\\ \zeta = 1/\rho\end{pmatrix}$ leads to

$$\begin{aligned}\frac{\partial u}{\partial t} &= -u\frac{\partial u}{\partial x} - \zeta\frac{\partial p}{\partial x}\\ \frac{\partial p}{\partial t} &= -\gamma p\frac{\partial u}{\partial x} - u\frac{\partial p}{\partial x}\\ \frac{\partial \zeta}{\partial t} &= -u\frac{\partial \zeta}{\partial x} + \zeta\frac{\partial u}{\partial x},\end{aligned}$$

i.e., a quadratic representation in the specific volume state variables $w = [u\ p\ \zeta]^\top$.

Example lifting transformation

One-dimensional Euler equations in conservative variables formulation:

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho u\\ E\end{pmatrix} = -\frac{\partial}{\partial x}\begin{pmatrix}\rho u\\ \rho u^2 + p\\ (E+p)u\end{pmatrix}, \quad E = \frac{p}{\gamma - 1} + \frac{1}{2}\rho u^2,$$

with state $s = [\rho\ \rho u\ E]^\top$ comprising the density ρ, specific momentum ρu, and energy E.
This representation contains several nonlinear terms that are not quadratic in the conservative state.

For constant heat capacity ratio γ, the transformation $\mathcal{T} : \begin{pmatrix}\rho\\ \rho u\\ E\end{pmatrix} \to \begin{pmatrix}u\\ p\\ \zeta = 1/\rho\end{pmatrix}$ leads to

$$\begin{aligned}\frac{\partial u}{\partial t} &= -u\frac{\partial u}{\partial x} - \zeta\frac{\partial p}{\partial x}\\ \frac{\partial p}{\partial t} &= -\gamma p\frac{\partial u}{\partial x} - u\frac{\partial p}{\partial x}\\ \frac{\partial \zeta}{\partial t} &= -u\frac{\partial \zeta}{\partial x} + \zeta\frac{\partial u}{\partial x},\end{aligned}$$

i.e., a quadratic representation in the specific volume state variables $w = [u\ p\ \zeta]^\top$.

Steps to perform Operator Inference

2. Training data manipulation: centering and scaling

- in problems with multiple variables, the state data can have significantly different scales
- for example, in a reactive flow simulation, the scales can vary from the order of 10^4 to 10^6 Pa for pressure to between 0 and 1 for the species mass fractions
- we therefore center and scale the transformed data variable-by-variables to ensure that the transformed variables are on the same scale
- we first center the transformed snapshot matrix around $\mathbf{W}_{\text{ref}} \in \mathbb{R}^n$

$$\mathbf{W}_{\text{cen}} = \mathbf{W} - \mathbf{W}_{\text{ref}},$$

- we then scale the centered snapshot matrix of transformed variables

$$\mathbf{Q} = \text{scale}(\mathbf{W}_{\text{cen}}),$$

to ensure that the variables in $\mathbf{Q}$ are on the same scale

Remarks

- the choice of $\mathbf{W}_{\text{ref}}$ and the scaling function can have a significant impact on the accuracy of the resulting reduced model

Steps to perform Operator Inference

2. Training data manipulation: centering and scaling

- in problems with multiple variables, the state data can have significantly different scales
- for example, in a reactive flow simulation, the scales can vary from the order of 10^4 to 10^6 Pa for pressure to between 0 and 1 for the species mass fractions
- we therefore center and scale the transformed data variable-by-variables to ensure that the transformed variables are on the same scale
- we first center the transformed snapshot matrix around $\mathbf{W}_{\text{ref}} \in \mathbb{R}^n$

$$\mathbf{W}_{\text{cen}} = \mathbf{W} - \mathbf{W}_{\text{ref}},$$

- we then scale the centered snapshot matrix of transformed variables

$$\mathbf{Q} = \text{scale}(\mathbf{W}_{\text{cen}}),$$

to ensure that the variables in $\mathbf{Q}$ are on the same scale

Remarks

- the choice of $\mathbf{W}_{\text{ref}}$ and the scaling function can have a significant impact on the accuracy of the resulting reduced model

Steps to perform Operator Inference

2. Training data manipulation: centering and scaling

- in problems with multiple variables, the state data can have significantly different scales
- for example, in a reactive flow simulation, the scales can vary from the order of 10^4 to 10^6 Pa for pressure to between 0 and 1 for the species mass fractions
- we therefore center and scale the transformed data variable-by-variables to ensure that the transformed variables are on the same scale
- we first center the transformed snapshot matrix around $\mathbf{W}_{\text{ref}} \in \mathbb{R}^n$

$$\mathbf{W}_{\text{cen}} = \mathbf{W} - \mathbf{W}_{\text{ref}},$$

- we then scale the centered snapshot matrix of transformed variables

$$\mathbf{Q} = \text{scale}(\mathbf{W}_{\text{cen}}),$$

to ensure that the variables in $\mathbf{Q}$ are on the same scale

Remarks

- the choice of $\mathbf{W}_{\text{ref}}$ and the scaling function can have a significant impact on the accuracy of the resulting reduced model

Steps to perform Operator Inference

After training data manipulation

- let us assume, without loss of generality, that after lifting, the high-fidelity model is quadratic in the lifted (and centered + scaled) variables

$$\frac{\mathrm{d}\mathbf{q}}{\mathrm{d}t} = \mathbf{A}\mathbf{q} + \mathbf{H}(\mathbf{q} \otimes \mathbf{q}) + \mathbf{c}, \quad \mathbf{A} \in \mathbb{R}^{n \times n}, \mathbf{H} \in \mathbb{R}^{n \times n^2}, \mathbf{c} \in \mathbb{R}^n$$

- note that due to centering of the lifted data, we also have a constant term $\mathbf{c}$ in the model

Our goal is to construct a structure-preserving quadratic reduced model via OpInf that reduces the dimension from n to $r \ll n$,

$$\frac{\mathrm{d}\hat{\mathbf{q}}}{\mathrm{d}t} = \hat{\mathbf{A}}\hat{\mathbf{q}} + \hat{\mathbf{H}}(\hat{\mathbf{q}} \otimes \hat{\mathbf{q}}) + \hat{\mathbf{c}}, \quad \hat{\mathbf{A}} \in \mathbb{R}^{r \times r}, \hat{\mathbf{H}} \in \hat{\mathbb{R}}^{r \times r^2}, \hat{\mathbf{c}} \in \mathbb{R}^r,$$

by inferring the reduced operators $\hat{\mathbf{A}}, \hat{\mathbf{H}}$, and $\hat{\mathbf{c}}$ from data.

Steps to perform Operator Inference

After training data manipulation

- let us assume, without loss of generality, that after lifting, the high-fidelity model is quadratic in the lifted (and centered + scaled) variables

$$\frac{\mathrm{d}\mathbf{q}}{\mathrm{d}t} = \mathbf{A}\mathbf{q} + \mathbf{H}(\mathbf{q} \otimes \mathbf{q}) + \mathbf{c}, \quad \mathbf{A} \in \mathbb{R}^{n \times n}, \mathbf{H} \in \mathbb{R}^{n \times n^2}, \mathbf{c} \in \mathbb{R}^n$$

- note that due to centering of the lifted data, we also have a constant term $\mathbf{c}$ in the model

Our goal is to construct a structure-preserving quadratic reduced model via OpInf that reduces the dimension from n to $r \ll n$,

$$\frac{\mathrm{d}\hat{\mathbf{q}}}{\mathrm{d}t} = \hat{\mathbf{A}}\hat{\mathbf{q}} + \hat{\mathbf{H}}(\hat{\mathbf{q}} \otimes \hat{\mathbf{q}}) + \hat{\mathbf{c}}, \quad \hat{\mathbf{A}} \in \mathbb{R}^{r \times r}, \hat{\mathbf{H}} \in \hat{\mathbb{R}}^{r \times r^2}, \hat{\mathbf{c}} \in \mathbb{R}^r,$$

by inferring the reduced operators $\hat{\mathbf{A}}, \hat{\mathbf{H}}$, and $\hat{\mathbf{c}}$ from data.

Steps to perform Operator Inference

3. Compute the POD basis

- we compute the proper orthogonal decomposition (POD) basis from the thin singular-value decomposition (SVD) of $\mathbf{Q}$

$$\mathbf{Q} = \mathbf{V}\mathbf{\Sigma}\mathbf{U}^\top,$$

where
  - $\mathbf{V} \in \mathbb{R}^{n \times n_t}$ contains the left singular vectors
  - $\mathbf{\Sigma} \in \mathbb{R}^{n_t \times n_t}$ is a diagonal matrix containing the singular values of $\mathbf{Q}$ in non-decreasing order $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{n_t}$, where σ_j denotes the jth singular value
  - $\mathbf{U} \in \mathbb{R}^{n_t \times n_t}$ contains the right singular vectors
- the first $r \ll n$ columns of $\mathbf{V}$, i.e., the left singular vectors corresponding to the r largest singular values form the POD basis $\mathbf{V}_r \in \mathbb{R}^{n \times r}$
- r is typically determined via energy-based criteria, e.g., choose r such that

$$\frac{\sum_{j=1}^{r} \sigma_j^2}{\sum_{j=1}^{n_t} \sigma_j^2} \geq p,$$

where $p = 95\%$ or $p = 99\%$

Steps to perform Operator Inference

3. Compute the POD basis

- we compute the proper orthogonal decomposition (POD) basis from the thin singular-value decomposition (SVD) of $\mathbf{Q}$

$$\mathbf{Q} = \mathbf{V}\mathbf{\Sigma}\mathbf{U}^\top,$$

  where
  - $\mathbf{V} \in \mathbb{R}^{n \times n_t}$ contains the left singular vectors
  - $\mathbf{\Sigma} \in \mathbb{R}^{n_t \times n_t}$ is a diagonal matrix containing the singular values of $\mathbf{Q}$ in non-decreasing order $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{n_t}$, where σ_j denotes the jth singular value
  - $\mathbf{U} \in \mathbb{R}^{n_t \times n_t}$ contains the right singular vectors
- the first $r \ll n$ columns of $\mathbf{V}$, i.e., the left singular vectors corresponding to the r largest singular values form the POD basis $\mathbf{V}_r \in \mathbb{R}^{n \times r}$
- r is typically determined via energy-based criteria, e.g., choose r such that

$$\frac{\sum_{j=1}^{r} \sigma_j^2}{\sum_{j=1}^{n_t} \sigma_j^2} \geq p,$$

  where $p = 95\%$ or $p = 99\%$

Steps to perform Operator Inference

3. Compute the POD basis: remarks

- in problems with sparse training data sets, the size of the training data set limits the maximum reduced dimension for a polynomial reduced model
- the computational complexity of the standard thin SVD is $\mathcal{O}(nn_t^2)$ (since $n \gg n_t$)
- alternatively, randomized SVD can be used instead, with complexity $\mathcal{O}(rnn_t + r^2(n + n_t))$ for a rank-r approximation; when $r \ll n_t, n$, the leading cost is $\mathcal{O}(rnn_t)$
- in settings in which either standard or randomized SVD are computationally infeasible, other approaches, e.g., incremental SVD [Brand; European Conference on Computer Vision, 2002]

Steps to perform Operator Inference

4. Project the transformed snapshots

- we project $\mathbf{Q}$ onto the linear subspace spanned by the column vectors of $\mathbf{V}_r$:

$$\hat{\mathbf{Q}} = \mathbf{V}_r^\top \mathbf{Q} \in \mathbb{R}^{r \times n_t}$$

- note that the size of $\hat{\mathbf{Q}}$ is small since $r \ll n$
- $\hat{\mathbf{Q}}$ will be used to form the data matrix in the OpInf learning problem

Steps to perform Operator Inference

5. Infer the reduced operators

- to infer the reduced operators $\hat{\mathbf{A}}, \hat{\mathbf{H}}, \hat{\mathbf{c}}$ of a quadratic reduced model, we must solve the following regularized linear least-squares minimization problem

$$\operatorname{argmin}_{\hat{\mathbf{A}},\hat{\mathbf{H}},\hat{\mathbf{c}}} \left\| \hat{\mathbf{Q}}^\top \hat{\mathbf{A}}^\top + \left(\hat{\mathbf{Q}} \otimes \hat{\mathbf{Q}}\right)^\top \hat{\mathbf{H}}^\top + \mathbf{1}\hat{\mathbf{c}} - \left(\frac{\mathrm{d}\hat{\mathbf{Q}}}{\mathrm{d}t}\right)^\top \right\|_F^2 + \lambda_\ell \left(\left\| \hat{\mathbf{A}} \right\|_F^2 + \|\hat{\mathbf{c}}\|_F^2 \right) + \lambda_q \left\| \hat{\mathbf{H}} \right\|_F^2$$

where

- F denotes the Frobenius norm
- $\lambda_\ell, \lambda_q \in \mathbb{R}$ are scalar regularization hyperparameters

Remarks

- in problems in which $\frac{\mathrm{d}\mathbf{Q}}{\mathrm{d}t}$ is available, we can compute $\frac{\mathrm{d}\hat{\mathbf{Q}}}{\mathrm{d}t} = \mathbf{V}_r^\top \frac{\mathrm{d}\mathbf{Q}}{\mathrm{d}t}$
- otherwise, we must estimate $\frac{\mathrm{d}\hat{\mathbf{Q}}}{\mathrm{d}t}$ using $\hat{\mathbf{Q}}$ e.g., finite differences
- in problems with sparse and down-sampled training data sets, this estimation can be very inaccurate and therefore lead to inaccurate ROMs

Steps to perform Operator Inference

5. Infer the reduced operators

- to infer the reduced operators $\hat{\mathbf{A}}, \hat{\mathbf{H}}, \hat{\mathbf{c}}$ of a quadratic reduced model, we must solve the following regularized linear least-squares minimization problem

$$\operatorname{argmin}_{\hat{\mathbf{A}},\hat{\mathbf{H}},\hat{\mathbf{c}}} \left\| \hat{\mathbf{Q}}^{\top}\hat{\mathbf{A}}^{\top} + \left(\hat{\mathbf{Q}} \otimes \hat{\mathbf{Q}}\right)^{\top}\hat{\mathbf{H}}^{\top} + \mathbf{1}\hat{\mathbf{c}} - \left(\frac{\mathrm{d}\hat{\mathbf{Q}}}{\mathrm{d}t}\right)^{\top} \right\|_F^2 + \lambda_\ell \left(\left\|\hat{\mathbf{A}}\right\|_F^2 + \|\hat{\mathbf{c}}\|_F^2 \right) + \lambda_q \left\|\hat{\mathbf{H}}\right\|_F^2$$

  where
  - F denotes the Frobenius norm
  - $\lambda_\ell, \lambda_q \in \mathbb{R}$ are scalar regularization hyperparameters

Remarks

- in problems in which $\frac{\mathrm{d}\mathbf{Q}}{\mathrm{d}t}$ is available, we can compute $\frac{\mathrm{d}\hat{\mathbf{Q}}}{\mathrm{d}t} = \mathbf{V}_r^{\top}\frac{\mathrm{d}\mathbf{Q}}{\mathrm{d}t}$
- otherwise, we must estimate $\frac{\mathrm{d}\hat{\mathbf{Q}}}{\mathrm{d}t}$ using $\hat{\mathbf{Q}}$ via e.g., finite differences
- in problems with sparse and down-sampled training data sets, this estimation can be very inaccurate and therefore lead to inaccurate ROMs

Steps to perform Operator Inference

5. Infer the reduced operators

In problems where an accurate estimation of the time derivative of the projected transformed snapshots is difficult, we consider instead a fully discrete reduced model

$$\hat{\mathbf{q}}[k+1] = \hat{\mathbf{A}}\mathbf{q}[k] + \hat{\mathbf{H}}(\hat{\mathbf{q}}[k] \otimes \hat{\mathbf{q}}[k]) + \hat{\mathbf{c}},$$

for which the reduced operators are inferred as

$$\operatorname{argmin}_{\hat{\mathbf{A}},\hat{\mathbf{H}},\hat{\mathbf{c}}} \left\| \hat{\mathbf{Q}}_1^\top \hat{\mathbf{A}}^\top + \left(\hat{\mathbf{Q}}_1 \otimes \hat{\mathbf{Q}}_1\right)^\top \hat{\mathbf{H}}^\top + \mathbf{1}_{n_t-1}\hat{\mathbf{c}} - \hat{\mathbf{Q}}_2^\top \right\|_F^2 + \lambda_\ell \left(\left\| \hat{\mathbf{A}} \right\|_F^2 + \|\hat{\mathbf{c}}\|_F^2 \right) + \lambda_q \left\| \hat{\mathbf{H}} \right\|_F^2,$$

where

$$\hat{\mathbf{Q}}_1 = \begin{bmatrix} | & | & & | \\ \hat{\mathbf{q}}_1 & \hat{\mathbf{q}}_2 & \cdots & \hat{\mathbf{q}}_{n_t-1} \\ | & | & & | \end{bmatrix}, \quad \hat{\mathbf{Q}}_2 = \begin{bmatrix} | & | & & | \\ \hat{\mathbf{q}}_2 & \hat{\mathbf{q}}_3 & \cdots & \hat{\mathbf{q}}_{n_t} \\ | & | & & | \end{bmatrix}$$

Steps to perform Operator Inference

5. Infer the reduced operators: finding the optimal regularization hyperparameters

- in "simple" settings, unregularized least-squares may suffice for finding the reduced operators of a polynomial reduced model that is predictive beyond the training time horizon
- in practice, however, we have:
  - sparse training data sets $\rightarrow$ overfitting
  - errors due to model mispecification (e.g., lifting transformations leading to only an approximate polynomial model)
  - possible closure errors due to truncation of the POD modes
  - numerical noise due to approximating the time derivative of the projected snapshots
- regularization is therefore key in practice to construct reduced models that can generalize beyond the training horizon [McQuarrie, Huang, and Willcox; Journal of the Royal Society of New Zealand, 2021]
- to find $\lambda_\ell^{\text{opt}}, \lambda_q^{\text{opt}}$, we perform a grid search over candidate hyperparameters and select the pair that yields a reduced model with a reasonable behaviour over the desired time horizon
- we then infer the reduced operators $\hat{\mathbf{A}}, \hat{\mathbf{H}}, \hat{\mathbf{c}}$ using $\lambda_\ell^{\text{opt}}, \lambda_q^{\text{opt}}$ in the regularized learning problem

Steps to perform Operator Inference

5. Infer the reduced operators: finding the optimal regularization hyperparameters

- in "simple" settings, unregularized least-squares may suffice for finding the reduced operators of a polynomial reduced model that is predictive beyond the training time horizon
- in practice, however, we have:
  - sparse training data sets $\rightarrow$ overfitting
  - errors due to model mispecification (e.g., lifting transformations leading to only an approximate polynomial model)
  - possible closure errors due to truncation of the POD modes
  - numerical noise due to approximating the time derivative of the projected snapshots
- regularization is therefore key in practice to construct reduced models that can generalize beyond the training horizon [McQuarrie, Huang, and Willcox; Journal of the Royal Society of New Zealand, 2021]
- to find $\lambda_\ell^{\text{opt}}, \lambda_q^{\text{opt}}$, we perform a grid search over candidate hyperparameters and select the pair that yields a reduced model with a reasonable behaviour over the desired time horizon
- we then infer the reduced operators $\hat{\mathbf{A}}, \hat{\mathbf{H}}, \hat{\mathbf{c}}$ using $\lambda_\ell^{\text{opt}}, \lambda_q^{\text{opt}}$ in the regularized learning problem

Steps to perform Operator Inference

5. Infer the reduced operators: finding the optimal regularization hyperparameters

- in "simple" settings, unregularized least-squares may suffice for finding the reduced operators of a polynomial reduced model that is predictive beyond the training time horizon
- in practice, however, we have:
  - sparse training data sets $\rightarrow$ overfitting
  - errors due to model mispecification (e.g., lifting transformations leading to only an approximate polynomial model)
  - possible closure errors due to truncation of the POD modes
  - numerical noise due to approximating the time derivative of the projected snapshots
- regularization is therefore key in practice to construct reduced models that can generalize beyond the training horizon [McQuarrie, Huang, and Willcox; Journal of the Royal Society of New Zealand, 2021]
- to find $\lambda_\ell^{\text{opt}}, \lambda_q^{\text{opt}}$, we perform a grid search over candidate hyperparameters and select the pair that yields a reduced model with a reasonable behaviour over the desired time horizon
- we then infer the reduced operators $\hat{\mathbf{A}}, \hat{\mathbf{H}}, \hat{\mathbf{c}}$ using $\lambda_\ell^{\text{opt}}, \lambda_q^{\text{opt}}$ in the regularized learning problem

Steps to perform Operator Inference

5. Infer the reduced operators: finding the optimal regularization hyperparameters

- in "simple" settings, unregularized least-squares may suffice for finding the reduced operators of a polynomial reduced model that is predictive beyond the training time horizon
- in practice, however, we have:
  - sparse training data sets $\rightarrow$ overfitting
  - errors due to model mispecification (e.g., lifting transformations leading to only an approximate polynomial model)
  - possible closure errors due to truncation of the POD modes
  - numerical noise due to approximating the time derivative of the projected snapshots
- regularization is therefore key in practice to construct reduced models that can generalize beyond the training horizon [McQuarrie, Huang, and Willcox; Journal of the Royal Society of New Zealand, 2021]
- to find $\lambda_\ell^{\text{opt}}, \lambda_q^{\text{opt}}$, we perform a grid search over candidate hyperparameters and select the pair that yields a reduced model with a reasonable behaviour over the desired time horizon
- we then infer the reduced operators $\hat{\mathbf{A}}, \hat{\mathbf{H}}, \hat{\mathbf{c}}$ using $\lambda_\ell^{\text{opt}}, \lambda_q^{\text{opt}}$ in the regularized learning problem

Steps to perform Operator Inference

5. Infer the reduced operators: finding the optimal regularization hyperparameters

- in "simple" settings, unregularized least-squares may suffice for finding the reduced operators of a polynomial reduced model that is predictive beyond the training time horizon
- in practice, however, we have:
  - sparse training data sets $\rightarrow$ overfitting
  - errors due to model mispecification (e.g., lifting transformations leading to only an approximate polynomial model)
  - possible closure errors due to truncation of the POD modes
  - numerical noise due to approximating the time derivative of the projected snapshots
- regularization is therefore key in practice to construct reduced models that can generalize beyond the training horizon [McQuarrie, Huang, and Willcox; Journal of the Royal Society of New Zealand, 2021]
- to find $\lambda_\ell^{\text{opt}}, \lambda_q^{\text{opt}}$, we perform a grid search over candidate hyperparameters and select the pair that yields a reduced model with a reasonable behaviour over the desired time horizon
- we then infer the reduced operators $\hat{\mathbf{A}}, \hat{\mathbf{H}}, \hat{\mathbf{c}}$ using $\lambda_\ell^{\text{opt}}, \lambda_q^{\text{opt}}$ in the regularized learning problem

OpInf for nonlinear dynamical systems: summary

Learning a low-dimensional model using only snapshot data from the original high-fidelity model (non-intrusive) but using variable transformations to expose and exploit model structure

1. generate full-state training trajectories (snapshots) from high-fidelity simulation
2. transform the training snapshots
   2.1 lift the original snapshots to expose (approximate) the desired polynomial structure
   2.2 center and scale transformed snapshot data
3. compute the POD basis from centered and scaled lifted trajectories
4. project transformed trajectories onto POD basis, to obtain trajectories in low-dimensional POD coordinate space
5. infer the reduced operator of the low-dimensional polynomial model (OpInf)
   5.1 find the optimal regularization hyperparameters
   5.2 use the optimal hyperparameters to solve the OpInf linear least-squares minimization problem

convenience of black-box learning +
rigor of projection-based reduction +
structure imposed by physics

OpInf for nonlinear dynamical systems: summary

Learning a low-dimensional model using only snapshot data from the original high-fidelity model (non-intrusive) but using variable transformations to expose and exploit model structure

1. generate full-state training trajectories (snapshots) from high-fidelity simulation
2. transform the training snapshots
   2.1 lift the original snapshots to expose (approximate) the desired polynomial structure
   2.2 center and scale transformed snapshot data
3. compute the POD basis from centered and scaled lifted trajectories
4. project transformed trajectories onto POD basis, to obtain trajectories in low-dimensional POD coordinate space
5. infer the reduced operator of the low-dimensional polynomial model (OpInf)
   5.1 find the optimal regularization hyperparameters
   5.2 use the optimal hyperparameters to solve the OpInf linear least-squares minimization problem

convenience of black-box learning +
rigor of projection-based reduction +
structure imposed by physics

OpInf for nonlinear dynamical systems: summary

Learning a low-dimensional model using only snapshot data from the original high-fidelity model (non-intrusive) but using variable transformations to expose and exploit model structure

1. generate full-state training trajectories (snapshots) from high-fidelity simulation
2. transform the training snapshots
   2.1 lift the original snapshots to expose (approximate) the desired polynomial structure
   2.2 center and scale transformed snapshot data
3. compute the POD basis from centered and scaled lifted trajectories
4. project transformed trajectories onto POD basis, to obtain trajectories in low-dimensional POD coordinate space
5. infer the reduced operator of the low-dimensional polynomial model (OpInf)
   5.1 find the optimal regularization hyperparameters
   5.2 use the optimal hyperparameters to solve the OpInf linear least-squares minimization problem

convenience of black-box learning +
rigor of projection-based reduction +
structure imposed by physics

Post-processing

- after the OpInf reduced solution is computed over the full time horizon (training and prediction), we map the reduced solution back to the original coordinates
- we begin by computing

$$\mathbf{Q}_{\text{OpInf}} = \mathbf{V}_r \hat{\mathbf{Q}}_{\text{OpInf}} \in \mathbb{R}^{n \times n_p},$$

  where $n_p \in \mathbb{N}$ denotes the total number of time iterations (training + prediction)
- we then unscale $\mathbf{Q}_{\text{OpInf}}$ as

$$\mathbf{W}_{\text{cen,OpInf}} = \text{unscale}(\mathbf{Q}_{\text{OpInf}}) \in \mathbb{R}^{n \times n_p}$$

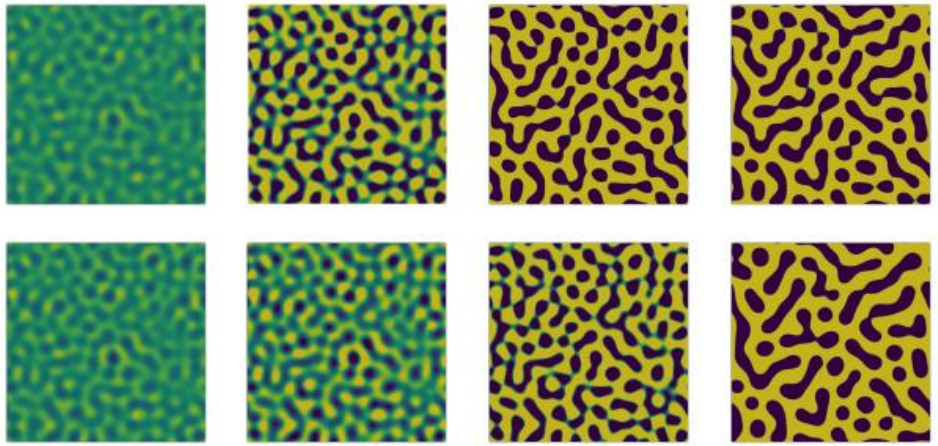
- lastly, we add the reference solution to $\mathbf{Q}_{\text{OpInf}}$ to obtain

$$\mathbf{W}_{\text{OpInf}} = \mathbf{W}_{\text{ref}} + \mathbf{W}_{\text{cen,OpInf}} \in \mathbb{R}^{n \times n_p}$$

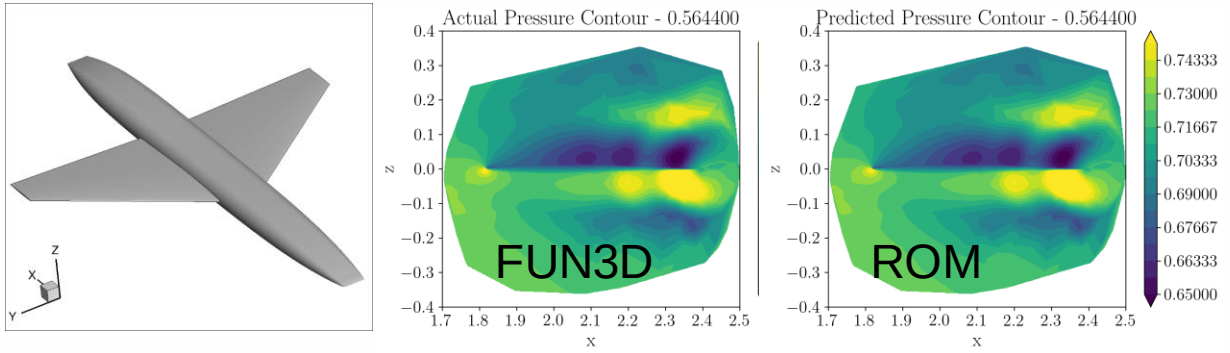
- we can then compare the reduced model solutions with the high-fidelity solutions

Operator Inference applications beyond combustion

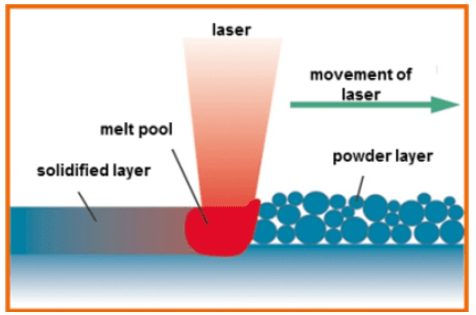
Phase-field modeling (Geelen)



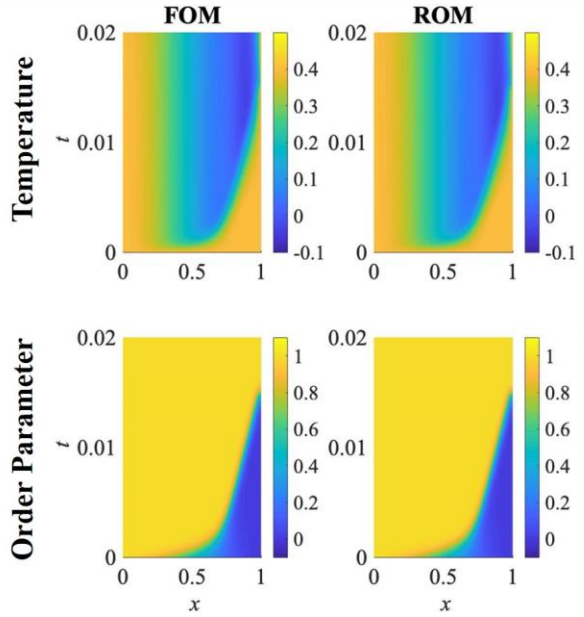
VAT Wing Aerostructural Analysis (Zastrow & Chaudhuri w/ Lockheed)



Solidification in additive manufacturing (Khodabakhshi w/ ORNL)



https://www.bintoa.com/powder-bed-fusion



Operator Inference advantages and disadvantages

Advantages

- easy to implement and use (for both state predictions over time and parametric predictions)
- the ROM construction and usage is fully decoupled from the high-fidelity simulation code
- allows rapid prototyping
- it enables ROM development across multiple sites without the need to transfer large-scale data sets from one site to another (e.g., training data generation, data manipulation, and POD basis computation at ARFL, and OpInf ROM construction and postprocessing at UT Austin)

Disadvantages

- the presented regularized OpInf forrmulation is not guaranteed to preserve properties such as energy conservation etc.
- the standard formulation is based on a linear POD basis, which inherits its challenges associated to reactive flow/transport-dominated problems
- these challenges can be mitigated by enhancing OpInf with domain-decomposition, filtering, or by using nonlinear manifolds instead of linear basis

Operator Inference advantages and disadvantages

Advantages

- easy to implement and use (for both state predictions over time and parametric predictions)
- the ROM construction and usage is fully decoupled from the high-fidelity simulation code
- allows rapid prototyping
- it enables ROM development across multiple sites without the need to transfer large-scale data sets from one site to another (e.g., training data generation, data manipulation, and POD basis computation at ARFL, and OpInf ROM construction and postprocessing at UT Austin)

Disadvantages

- the presented regularized OpInf forrmulation is not guaranteed to preserve properties such as energy conservation etc.
- the standard formulation is based on a linear POD basis, which inherits its challenges associated to reactive flow/transport-dominated problems
- these challenges can be mitigated by enhancing OpInf with domain-decomposition, filtering, or by using nonlinear manifolds instead of linear basis

Summary: basics of learning dynamical systems from data

- OpInf is a scientific machine learning approach for learning non-intrusive data-driven reduced models for systems with polynomial non-linearities
- OpInf blends the interpretability of physics-based modeling with the convenience of data-driven methods to construct physics-based reduced models from data
- lifting transformations can be used to address more generic nonlinear models
- the choice of centering and scaling the training data has an impact on the overall OpInf performance
- regularization is key to reduce overfitting, to account for model misspecification, truncation of POD modes, etc.
- for problems with down-sampled training snapshots, we consider a fully discrete formulation for the low-dimensional reduced model

References

- Peherstorfer, B., Willcox, K., 2016. *Data-driven operator inference for nonintrusive projection-based model reduction*. Computer Methods in Applied Mechanics and Engineering 306, 196–215.
- Qian, E., Kramer, B., Peherstorfer, B., Willcox, K., 2020. *Lift & Learn: Physics-informed machine learning for large-scale nonlinear dynamical systems*. Physica D: Nonlinear Phenomena 406, 132401.
- Swischuk, R., Kramer, B., Huang, C., Willcox, K., 2020. *Learning Physics-Based Reduced-Order Models for a Single-Injector Combustion Process*. AIAA Journal 58, 2658–2672.
- McQuarrie, S.A., Huang, C., Willcox, K.E., 2021. *Data-driven reduced-order models via regularised Operator Inference for a single-injector combustion process*. Journal of the Royal Society of New Zealand 51, 194–211.
- Qian, E., Farcas, I.G., Willcox, K., 2022. *Reduced Operator Inference for Nonlinear Partial Differential Equations*. SIAM Journal on Scientific Computing 44, A1934–A1959.