MAX PLANCK INSTITUTE
FOR DYNAMICS OF COMPLEX
TECHNICAL SYSTEMS
MAGDEBURG

CSC
COMPUTATIONAL METHODS IN
SYSTEMS AND CONTROL THEORY

Introduction to reduced-order modeling

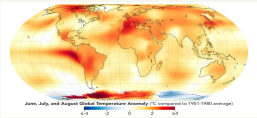
Sridhar Chellappa

TransferLab Seminar

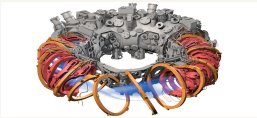
October 17, 2024

- Science seeks to *understand* and *leverage* physical phenomena for the benefit of humanity [1]

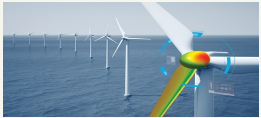
Climate science



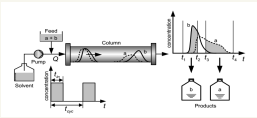
Nuclear fusion



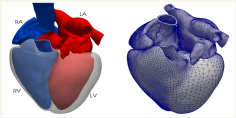
Wind farms



Chemical process



Cardiology



. . .

[1] Image:Credits: (1) NASA (2) IPP (3) Enteknograte (5) F. Regazzoni

- Science seeks to *understand* and *leverage* physical phenomena for the benefit of humanity
- Two paradigms for modeling: **PDE/physics-based** and **data-based**

(Partial) Differential Equations

$$\mathcal{N}\left(\mathbf{u}(\mathbf{x}), \partial\mathbf{u}(\mathbf{x}), \mathbf{x}, t, \mathbf{s}(t), \boldsymbol{\mu}\right) = \mathbf{0},$$
$$\mathcal{B}\left(\mathbf{u}(\mathbf{x}), \mathbf{x}, t, \boldsymbol{\mu}\right) = \mathbf{0} \quad \text{bndy cond.}$$

- $\mathbf{x} \in \mathbb{R}^d$ is the spatial variable,
- $\mathbf{u}(\mathbf{x}) \in \mathbb{R}$ is the state or field variable,
- $\mathbf{s}(t)$ are inputs, t is time and $\boldsymbol{\mu} \in \mathbb{R}^p$ are system parameters

Numerous examples:
heat equation, shallow-water equations, Maxwell's equations, Schrödinger equation, and many more

- Science seeks to *understand* and *leverage* physical phenomena for the benefit of humanity
- Two paradigms for modeling: **PDE/physics-based** and **data-based**

(Partial) Differential Equations

$$\mathcal{N}\left(\mathbf{u}(\mathbf{x}), \partial\mathbf{u}(\mathbf{x}), \mathbf{x}, t, \mathbf{s}(t), \boldsymbol{\mu}\right) = \mathbf{0},$$
$$\mathcal{B}\left(\mathbf{u}(\mathbf{x}), \mathbf{x}, t, \boldsymbol{\mu}\right) = \mathbf{0} \quad \text{bndy cond.}$$

- $\mathbf{x} \in \mathbb{R}^d$ is the spatial variable,
- $\mathbf{u}(\mathbf{x}) \in \mathbb{R}$ is the state or field variable,
- $\mathbf{s}(t)$ are inputs, t is time and $\boldsymbol{\mu} \in \mathbb{R}^p$ are system parameters

Numerous examples:
heat equation, shallow-water equations, Maxwell's equations, Schrödinger equation, and many more

Data-based

- Develop understanding based on
  - sensor measurements
  - images/video
  - historical data
- e.g., acoustics, computer vision, finance


Credit: soundonsound


Credit: Lumentum

Credit: Wikipedia

- Solving PDEs has evolved over centuries[1]

Pen-paper solve

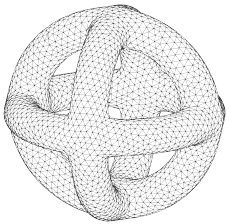
Analog computers

Human computers

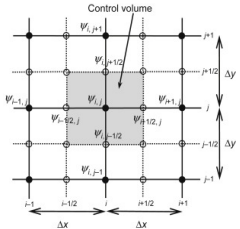
Supercomputers

[1] Images credit: (1) Chat-GPT (2) CHM (3) JPL (4) TOP500

- Solving PDEs has evolved over centuries
- Standard methods: finite difference method, finite element and finite volume methods, spectral element method



Credit: Wikipedia



Credit: Nakayama (2018)

- Solving PDEs has evolved over centuries
- Standard methods: finite difference method, finite element and finite volume methods, spectral element method
- Challenges: **nonlinear**, **coupled**, **high-dimensional**, etc.

- Solving PDEs has evolved over centuries
- Standard methods: finite difference method, finite element and finite volume methods, spectral element method
- Challenges: **nonlinear**, **coupled**, **high-dimensional**, etc.
- Applications: **design optimization**, **parameter estimation**, **uncertainty quantification**, ...
- Repeated solutions of PDEs is very expensive due to requirements such as high resolution, multi-query setting, real-time simulations

Introduction
Challenges with solving PDEs

- Solving PDEs has evolved over centuries
- Standard methods: finite difference method, finite element and finite volume methods, spectral element method
- Challenges: **nonlinear**, **coupled**, **high-dimensional**, etc.
- Applications: **design optimization**, **parameter estimation**, **uncertainty quantification**, ...
- Repeated solutions of PDEs is very expensive due to requirements such as high resolution, multi-query setting, real-time simulations

Need for reduced-order modeling for fast simulations

Full-order model (FOM)

$$\mathbf{E}(\boldsymbol{\mu})\frac{d}{dt}\mathbf{u}(t,\boldsymbol{\mu}) = \mathbf{A}(\boldsymbol{\mu})\mathbf{u}(t,\boldsymbol{\mu}) + \mathbf{f}(\mathbf{u}(t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \mathbf{B}\mathbf{s}(t),$$
$$\mathbf{u}(0,\boldsymbol{\mu}) = \mathbf{u}_0$$

- Discrete state $\mathbf{u}(t,\boldsymbol{\mu}) \in \mathbb{R}^N$
- $\mathbf{E}, \mathbf{A} \in \mathbb{R}^{N\times N}$; Nonlinearity $\mathbf{f}(\mathbf{u}) \in \mathbb{R}^N$
- Input signal $\mathbf{s}(t)$, matrix $\mathbf{B} \in \mathbb{R}^{N\times n_I}$
- Parameter $\boldsymbol{\mu} \in \mathbb{R}^p$

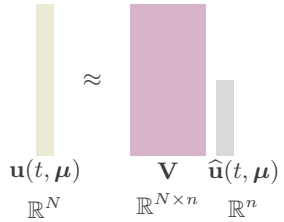
- Computational cost: $N_t \cdot n_{\texttt{newton}} \cdot \mathcal{O}(N^x)$ for linear solve for $n_{\texttt{newton}}$ Newton iterations and N_t time steps
- Repeated solution for different inputs $\mathbf{s}(t)$ and parameters $\boldsymbol{\mu}$

Full-order model (FOM)

$$\mathbf{E}(\boldsymbol{\mu})\frac{d}{dt}\mathbf{u}(t,\boldsymbol{\mu}) = \mathbf{A}(\boldsymbol{\mu})\mathbf{u}(t,\boldsymbol{\mu}) + \mathbf{f}(\mathbf{u}(t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \mathbf{B}\mathbf{s}(t),$$
$$\mathbf{u}(0,\boldsymbol{\mu}) = \mathbf{u}_0$$

- Discrete state $\mathbf{u}(t,\boldsymbol{\mu}) \in \mathbb{R}^N$
- $\mathbf{E}, \mathbf{A} \in \mathbb{R}^{N\times N}$; Nonlinearity $\mathbf{f}(\mathbf{u}) \in \mathbb{R}^N$
- Input signal $\mathbf{s}(t)$, matrix $\mathbf{B} \in \mathbb{R}^{N\times n_I}$
- Parameter $\boldsymbol{\mu} \in \mathbb{R}^p$

Approximation:

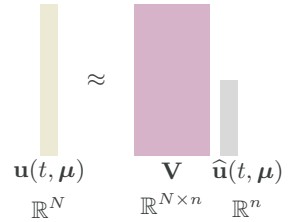


Full-order model (FOM)

$$\mathbf{E}(\boldsymbol{\mu})\frac{d}{dt}\mathbf{u}(t,\boldsymbol{\mu}) = \mathbf{A}(\boldsymbol{\mu})\mathbf{u}(t,\boldsymbol{\mu}) + \mathbf{f}(\mathbf{u}(t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \mathbf{B}\mathbf{s}(t),$$
$$\mathbf{u}(0,\boldsymbol{\mu}) = \mathbf{u}_0$$

- Discrete state $\mathbf{u}(t,\boldsymbol{\mu}) \in \mathbb{R}^N$
- $\mathbf{E}, \mathbf{A} \in \mathbb{R}^{N\times N}$; Nonlinearity $\mathbf{f}(\mathbf{u}) \in \mathbb{R}^N$
- Input signal $\mathbf{s}(t)$, matrix $\mathbf{B} \in \mathbb{R}^{N\times n_I}$
- Parameter $\boldsymbol{\mu} \in \mathbb{R}^p$

Approximation:



Petrov-Galerkin projection:

$$\mathbf{E}\frac{d}{dt}(\mathbf{V}\hat{\mathbf{u}}) \approx \mathbf{A}\mathbf{V}\hat{\mathbf{u}} + \mathbf{f}(\mathbf{V}\hat{\mathbf{u}}) + \mathbf{B}\mathbf{s}(t),$$

$$\mathbf{W}^\mathsf{T}\left(\mathbf{E}\frac{d}{dt}(\mathbf{V}\hat{\mathbf{u}}) - \mathbf{A}\mathbf{V}\hat{\mathbf{u}} - \mathbf{f}(\mathbf{V}\hat{\mathbf{u}}) - \mathbf{B}\mathbf{s}(t)\right) = \mathbf{0}$$

Reduced-order model (ROM)

$$\widehat{\mathbf{E}}(\boldsymbol{\mu})\frac{d}{dt}\widehat{\mathbf{u}}(t,\boldsymbol{\mu}) = \widehat{\mathbf{A}}(\boldsymbol{\mu})\widehat{\mathbf{u}}(t,\boldsymbol{\mu}) + \widehat{\mathbf{f}}(\mathbf{V}\widehat{\mathbf{u}}(t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \widehat{\mathbf{B}}\mathbf{s}(t),$$
$$\widehat{\mathbf{u}}(0,\boldsymbol{\mu}) = \widehat{\mathbf{u}}_0$$

- Reduced state $\widehat{\mathbf{u}}(t,\boldsymbol{\mu}) \in \mathbb{R}^n$, $n \ll N$
- $\widehat{\mathbf{E}} := \mathbf{W}^\mathsf{T}\mathbf{E}\mathbf{V}$, $\widehat{\mathbf{A}} := \mathbf{W}^\mathsf{T}\mathbf{A}\mathbf{V}$, $\widehat{\mathbf{B}} := \mathbf{W}^\mathsf{T}\mathbf{B}$
- Nonlinearity $\widehat{\mathbf{f}}(\mathbf{u}) := \mathbf{W}^\mathsf{T}\mathbf{f}(\mathbf{V}\widehat{\mathbf{u}}) \in \mathbb{R}^n$
- Projection matrices $\mathbf{V}, \mathbf{W} \in \mathbb{R}^{N\times n}$

- Affine parameter dependence is key: $\mathbf{E}(\boldsymbol{\mu}) := \sum_{i=0}^{n_E} \Theta_i(\boldsymbol{\mu})\mathbf{E}_i$
- $\implies \widehat{\mathbf{E}}(\boldsymbol{\mu}) := \sum_{i=0}^{n_E} \Theta_i(\boldsymbol{\mu})\widehat{\mathbf{E}}_i$

- Computational cost: $N_t \cdot n_{\texttt{newton}} \cdot \mathcal{O}(n^x)$ for linear solve for $n_{\texttt{newton}}$ Newton iterations and N_t time steps $\rightarrow$ expected speedup $\mathcal{O}((N/n)^x)$

Reduced-order model (ROM)

$$\widehat{\mathbf{E}}(\boldsymbol{\mu})\frac{d}{dt}\widehat{\mathbf{u}}(t,\boldsymbol{\mu}) = \widehat{\mathbf{A}}(\boldsymbol{\mu})\widehat{\mathbf{u}}(t,\boldsymbol{\mu}) + \widehat{\mathbf{f}}(\mathbf{V}\widehat{\mathbf{u}}(t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \widehat{\mathbf{B}}\mathbf{s}(t),$$
$$\widehat{\mathbf{u}}(0,\boldsymbol{\mu}) = \widehat{\mathbf{u}}_0$$

- Reduced state $\widehat{\mathbf{u}}(t,\boldsymbol{\mu}) \in \mathbb{R}^n$, $n \ll N$
- $\widehat{\mathbf{E}} := \mathbf{W}^\mathsf{T}\mathbf{E}\mathbf{V}$, $\widehat{\mathbf{A}} := \mathbf{W}^\mathsf{T}\mathbf{A}\mathbf{V}$, $\widehat{\mathbf{B}} := \mathbf{W}^\mathsf{T}\mathbf{B}$
- Nonlinearity $\widehat{\mathbf{f}}(\mathbf{u}) := \mathbf{W}^\mathsf{T}\mathbf{f}(\mathbf{V}\widehat{\mathbf{u}}) \in \mathbb{R}^n$
- Projection matrices $\mathbf{V}, \mathbf{W} \in \mathbb{R}^{N\times n}$

- Affine parameter dependence is key: $\mathbf{E}(\boldsymbol{\mu}) := \sum_{i=0}^{n_E} \Theta_i(\boldsymbol{\mu})\mathbf{E}_i$
- $\implies \widehat{\mathbf{E}}(\boldsymbol{\mu}) := \sum_{i=0}^{n_E} \Theta_i(\boldsymbol{\mu})\widehat{\mathbf{E}}_i$

$$\widehat{\mathbf{E}}_i(\boldsymbol{\mu}) = \mathbf{W}^\mathsf{T}\,\mathbf{E}_i(\boldsymbol{\mu})\,\mathbf{V}$$

- **Solving ROM is computationally cheaper than solving FOM**
- Model reduction approaches differ in how $(\mathbf{V}, \mathbf{W})$ are computed

Running example: Burgers' equation

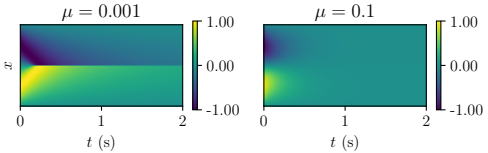
- Viscous 1-D Burgers' equation over domain $x \in [0, 1]$ and time $t \in [0, 2]$

$$\frac{\partial u(x,t)}{\partial t} = \mu \frac{\partial^2 u(x,t)}{\partial x^2} + u(x,t)\frac{\partial u(x,t)}{\partial x} + s(t), \qquad u(0,t) = u(1,t) = 0$$
$$u(x,0) = \sin(x)$$

- Variable parameter $\mu \in [0.001, 1]$
- Discretized with Finite volume method, $N = 1000$
- Full-order model:

$$\mathbf{E}\mathbf{u}(t,\mu) = \mu\mathbf{A}\mathbf{u}(t) + \mathbf{f}(\mathbf{u}(t,\mu)) + \mathbf{B}s(t),$$
$$\mathbf{u}(0) = \sin(\mathbf{x})$$

- System matrices can be extracted in software like FENICs, etc.



ROM Methods
Overview

- Physics/Projection-based ROM methods :
  - Proper orthogonal decomposition (POD) (1980 –)
  - Reduced basis method (RBM) (2000 –)
  - Balanced truncation, Moment matching, etc. ... (1980 –)
- Data-driven methods :
  - Operator inference (OpInf) (2016 –)
  - POD-NN (2019 –)
  - Deep learning-ROM (DL-ROM), POD-DL-ROM (2020 –)
  - Time series models (ARIMA, NARX, etc.) (1980 –)
  - Dynamic mode decomposition (DMD), SINDy, Koopman methods (2010 –)
  - Neural ODEs, DeepONet, Neural Operators, etc. (2018 –)
- Hybrid methods :
  - Physics-informed neural nets (PINNs) (2019 –)

ROM Methods
Overview

- Physics/Projection-based ROM methods :
  - **Proper orthogonal decomposition (POD)** (1980 –)
  - **Reduced basis method (RBM)** (2000 –)
  - Balanced truncation, Moment matching, etc. ... (1980 –)
- Data-driven methods :
  - **Operator inference (OpInf)** (2016 –)
  - **POD-NN** (2019 –)
  - **Deep learning-ROM (DL-ROM), POD-DL-ROM** (2020 –)
  - Time series models (ARIMA, NARX, etc.) (1980 –)
  - Dynamic mode decomposition (DMD), SINDy, Koopman methods (2010 –)
  - Neural ODEs, DeepONet, Neural Operators, etc. (2018 –)
- Hybrid methods :
  - Physics-informed neural nets (PINNs) (2019 –)

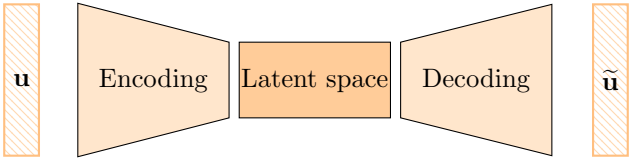


Figure: A unified structure for ROMs

Algorithm

1. **Data:** Solve FOM at $N_{\boldsymbol{\mu}}$ different parameters $\boldsymbol{\mu}$ to collect snapshot data

$$\mathcal{U} := \Big(\underbrace{\begin{bmatrix} \vert & \vert & & \vert \\ \vert & \vert & \cdots & \vert \\ \vert & \vert & & \vert \end{bmatrix}}_{\mathbf{U}(\boldsymbol{\mu}_1)} \underbrace{\begin{bmatrix} \vert & \vert & & \vert \\ \vert & \vert & \cdots & \vert \\ \vert & \vert & & \vert \end{bmatrix}}_{\mathbf{U}(\boldsymbol{\mu}_2)} \cdots \underbrace{\begin{bmatrix} \vert & \vert & & \vert \\ \vert & \vert & \cdots & \vert \\ \vert & \vert & & \vert \end{bmatrix}}_{\mathbf{U}(\boldsymbol{\mu}_{N_\mu})} \Big) \in \mathbb{R}^{N \times N_t N_\mu}$$

2. Perform proper orthogonal decomposition (POD); or singular value decomposition: $\mathrm{svd}\,(\mathcal{U}) \rightarrow \mathbf{L}\boldsymbol{\Sigma}\mathbf{Q}^{\mathsf{T}}$; projection matrix $\mathbf{V} := \mathbf{L}\,[:, :n]$
3. Obtain ROM via projection setting $\mathbf{W} = \mathbf{V}$, e.g., $\widehat{\mathbf{A}} := \mathbf{V}^T\mathbf{A}\mathbf{V}$

ROM Method: POD-ROM

Algorithm

1. **Data:** Solve FOM at $N_{\boldsymbol{\mu}}$ different parameters $\boldsymbol{\mu}$ to collect snapshot data

$$\mathcal{U} := \Bigg(\underbrace{\begin{bmatrix} \vdots & \cdots & \vdots \end{bmatrix}}_{\mathbf{U}(\boldsymbol{\mu}_1)} \underbrace{\begin{bmatrix} \vdots & \cdots & \vdots \end{bmatrix}}_{\mathbf{U}(\boldsymbol{\mu}_2)} \cdots \underbrace{\begin{bmatrix} \vdots & \cdots & \vdots \end{bmatrix}}_{\mathbf{U}(\boldsymbol{\mu}_{N_\mu})} \Bigg) \in \mathbb{R}^{N \times N_t N_\mu}$$

2. Perform proper orthogonal decomposition (POD); or singular value decomposition: $\mathtt{svd}\,(\mathcal{U}) \rightarrow \mathbf{L}\boldsymbol{\Sigma}\mathbf{Q}^{\mathsf{T}}$; projection matrix $\mathbf{V} := \mathbf{L}\,[:, :n]$
3. Obtain ROM via projection setting $\mathbf{W} = \mathbf{V}$, e.g., $\widehat{\mathbf{A}} := \mathbf{V}^T\mathbf{A}\mathbf{V}$

✓ Simple implementation

✓ Interpretable

× Needs data ($N_{\boldsymbol{\mu}}$ is large)

× Heuristic choice of n

× No accuracy guarantee

- POD-ROM for viscous Burgers' equation
- #samples $= 50$, $n = 11$
- `tolerance` $= 10^{-4}$
- Max. test set error $1.7 \cdot 10^{-2}$

Training time (s)	31
Inference time (s)	0.02
Speedup	15.5

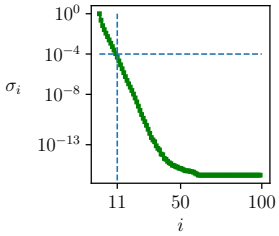


Figure: Singular value decay

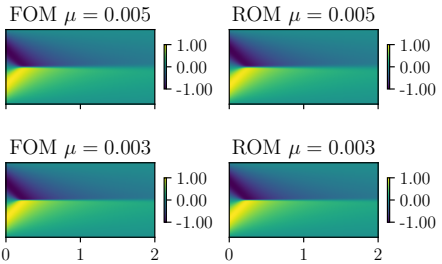


Figure: Test set performance

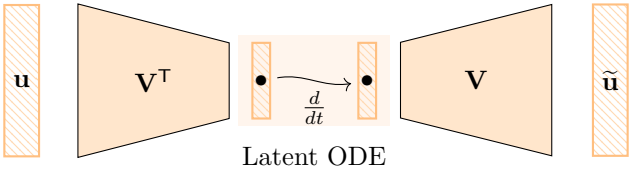


Figure: POD-ROM in the unified structure

- Latent ODE based obtained using projection of original model

ROM Method: Evaluating the nonlinear term efficiently

$$\widehat{\mathbf{E}}(\boldsymbol{\mu})\frac{d}{dt}\widehat{\mathbf{u}}(t,\boldsymbol{\mu}) = \widehat{\mathbf{A}}(\boldsymbol{\mu})\widehat{\mathbf{u}}(t,\boldsymbol{\mu}) + \widehat{\mathbf{f}}(\mathbf{V}\widehat{\mathbf{u}}(t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \widehat{\mathbf{B}}\mathbf{s}(t),$$

$$\widehat{\mathbf{u}}(0,\boldsymbol{\mu}) = \widehat{\mathbf{u}}_0$$

- Evaluating ROM involves nonlinear term $\widehat{\mathbf{f}}(\mathbf{V}\widehat{\mathbf{u}}(t,\boldsymbol{\mu}),\boldsymbol{\mu}) := \mathbf{V}^{\mathsf{T}}\mathbf{f}(\mathbf{V}\widehat{\mathbf{u}},\boldsymbol{\mu}) \in \mathbb{R}^n$
- Involves $\mathcal{O}(N)$ operations, although n-dim. quantity is computed
- **Idea: Reconstruct $\widehat{\mathbf{f}}(\cdot)$ using only few ($\ell \ll N$) evaluations of $\mathbf{f}(\cdot)$**

ROM Method: Evaluating the nonlinear term efficiently

Gappy reconstruction/ Gappy-POD [Everson '95]

- Data of the image available only partially
- Recover the image using part data

Figure: True face

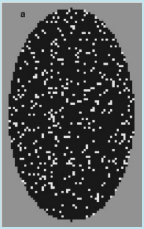


Figure: Masked face

[1] Images from: Everson and Sirovich (Journal of The Optical Society of America, 1995)

Gappy reconstruction/ Gappy-POD [Everson '95]

- Data of the image available only partially
- Recover the image using part data
- Extract **eigenfaces** from training images

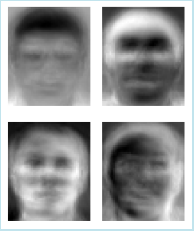


Figure: True face

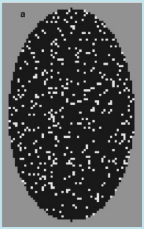


Figure: Masked face

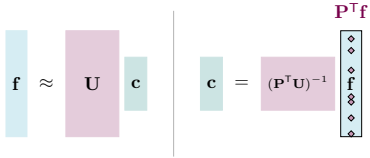
[1] Images from: Everson and Sirovich (Journal of The Optical Society of America, 1995)

ROM Method: Evaluating the nonlinear term efficiently

- Evaluating ROM involves nonlinear term $\widehat{\mathbf{f}}(\mathbf{V}\hat{\mathbf{u}}(t,\boldsymbol{\mu}),\boldsymbol{\mu}) := \mathbf{V}^{\mathsf{T}}\mathbf{f}(\mathbf{V}\hat{\mathbf{u}},\boldsymbol{\mu}) \in \mathbb{R}^n$
- **Idea: Reconstruct $\widehat{\mathbf{f}}(\cdot)$ using only few ($\ell \ll N$) evaluations of $\mathbf{f}(\cdot)$**

Hyperreduction using DEIM[2]:

1. $\widehat{\mathbf{f}} \approx \mathbf{V}^{\mathsf{T}}\mathbf{U}\left(\mathbf{P}^{\mathsf{T}}\mathbf{U}\right)^{-1}\mathbf{P}^{\mathsf{T}}\mathbf{f}$
2. Approximate $\mathbf{U}$ with POD of "snapshots" of $\mathbf{f}(\mathbf{V}\hat{\mathbf{u}})$
3. Evaluate only ℓ indices of $\mathbf{f}$



- $\mathbf{V}^{\mathsf{T}}\mathbf{U}\left(\mathbf{P}^{\mathsf{T}}\mathbf{U}\right)^{-1}$ is precomputed
- $\mathcal{O}(\ell)$ operations, $\ell \ll N$

[2] S. Chaturantabut, D. Sorensen, Nonlinear Model Reduction via Discrete Empirical Interpolation, SIAM Journal on Scientific Computing, Vol. 32, 5 (2010)

- POD-ROM for viscous Burgers' equation with hyperreduction
- #samples $= 50$, $n = 11, \ell = 12$
- Max. test set error $1.6 \cdot 10^{-2}$
- **Slightly more training time for faster inference time**

	POD-ROM	POD-ROM + hyperred.
Training time (s)	31	41
Inference time (s)	0.024	0.014

- POD-ROM for viscous Burgers' equation with hyperreduction
- #samples $= 50$, $n = 11, \ell = 12$
- Max. test set error $1.6 \cdot 10^{-2}$
- **Slightly more training time for faster inference time**
- Reducing the data need for POD and certifying the accuracy... via RBM

	POD-ROM	POD-ROM + hyperred.
Training time (s)	31	41
Inference time (s)	0.024	0.014

POD-Greedy algorithm

Set $i = 0$; Parameter sample pool $\{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_s\}$

`while` $\epsilon >$ `tolerance` :

1. Solve FOM at parameter $\boldsymbol{\mu}_i^*$ and collect snapshots

$$\mathbf{U}(\boldsymbol{\mu}_i^*) = \left[\begin{array}{ccc} \Big\| & \Big\| & \Big\| \\ \Big\| & \cdots & \Big\| \end{array} \right] \in \mathbb{R}^{N \times N_t}$$

2. Obtain $\mathbf{V}_i$ through POD of $\mathbf{U}(\boldsymbol{\mu}_i^*)$; project and compute ROM ($\widehat{\mathbf{E}}$, $\widehat{\mathbf{A}}$, etc.)
3. Choose next sample $\boldsymbol{\mu}_{i+1}^*$ as:

$$\boldsymbol{\mu}_{i+1}^* := \underset{\boldsymbol{\mu} \in \mathtt{pool}}{\arg\max}\, \Delta(\boldsymbol{\mu})$$

4. Set $i = i + 1$; $\epsilon := \Delta(\boldsymbol{\mu}_i^*)$

ROM Method: Reduced Basis Method

Basics

POD-Greedy algorithm

Set $i = 0$; Parameter sample pool $\{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_s\}$

`while` $\epsilon >$ `tolerance` :

1. Solve FOM at parameter $\boldsymbol{\mu}_i^*$ and collect snapshots

$$\mathbf{U}(\boldsymbol{\mu}_i^*) = \begin{bmatrix} | & | & \cdots & | \end{bmatrix} \in \mathbb{R}^{N \times N_t}$$

2. Obtain $\mathbf{V}_i$ through POD of $\mathbf{U}(\boldsymbol{\mu}_i^*)$; project and compute ROM ($\widehat{\mathbf{E}}$, $\widehat{\mathbf{A}}$, etc.)
3. Choose next sample $\boldsymbol{\mu}_{i+1}^*$ as:

$$\boldsymbol{\mu}_{i+1}^* := \arg\max_{\boldsymbol{\mu} \in \texttt{pool}} \Delta(\boldsymbol{\mu})$$

4. Set $i = i + 1$; $\epsilon := \Delta(\boldsymbol{\mu}_i^*)$

Error estimator
$\|\mathbf{u}(t, \boldsymbol{\mu}) - \widetilde{\mathbf{u}}(t, \boldsymbol{\mu})\| \leq \Delta(t, \boldsymbol{\mu})$

- residual-based or error bounds

✓ Needs less data, Automatic choice of n

✓ ROM has error certification

× More involved implementation

- Error estimation informs the accuracy of the current reduced-order model

$$\|\mathbf{u}(t,\boldsymbol{\mu}) - \widetilde{\mathbf{u}}(t,\boldsymbol{\mu})\| \leq \Delta(t,\boldsymbol{\mu})$$

- Residual-based error estimator (approximated with some quadrature):

$$\mathbf{r}(t,\boldsymbol{\mu}) := \mathbf{A}(\boldsymbol{\mu})\widetilde{\mathbf{u}}(t,\boldsymbol{\mu}) + \mathbf{f}(\widetilde{\mathbf{u}}(t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \mathbf{B}\mathbf{s}(t) - \mathbf{E}\dot{\mathbf{u}}(t,\boldsymbol{\mu})$$

- More sophisticated bounds available using dual/adjoint systems[3,4]
- Desired qualities:
  - Close to true error
  - Cheaply computable

[3] M. A. Grepl and A. T. Patera. A posteriori error bounds for reduced-basis approximations of parametrized parabolic partial differential equations. ESAIM: Math. Model. Numer. Anal., 39(1):157–181, 2005

[4] S. Chellappa, L. Feng, and P. Benner. Adaptive basis construction and improved error estimation for parametric nonlinear dynamical systems. Internat. J. Numer. Methods Engrg., 121(23):5320–5349, 2020

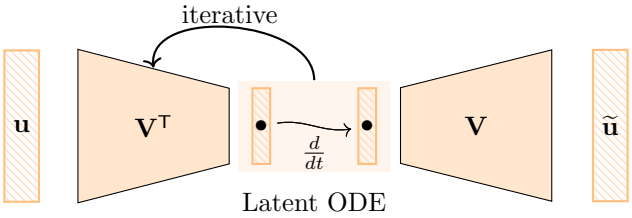


Figure: RBM in the unified structure

- Reduced Basis Method for viscous Burgers' equation
- #training samples $= 50$; $\texttt{tolerance} = 10^{-4}$
- ROM: $n = 15$, $\ell = 20$, Max. test error $7 \cdot 10^{-4}$
- **Only 6 FOM solves used!**

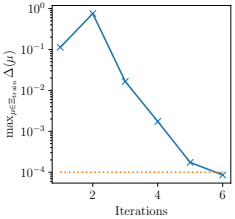


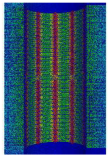
Figure: RBM error convergence

	POD-ROM	POD-ROM + hyperred.	Adaptive RBM [5]
Training time (s)	31	41	33
Inference time (s)	0.024	0.014	0.015
Test error	$1.7 \cdot 10^{-2}$	$1.6 \cdot 10^{-2}$	$7 \cdot 10^{-4}$

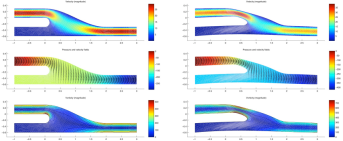
[5] S. Chellappa, L. Feng, P. Benner, A training set subsampling strategy for the reduced basis method, Journal of Scientific Computing, 2021

ROM Method: Reduced Basis Method

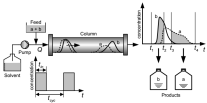
Method	Difficulty	Data/Solves	Accuracy
POD	Easy	High	Medium
POD-DEIM	Medium	High	Medium
RBM	High	Less	High



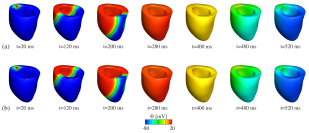
Helical Magnet [Prudhomme '15]



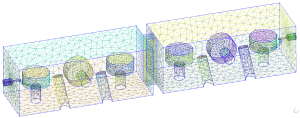
Shape optimization [Manzoni '12]



Batch chromatography [Chellappa '20]



Left ventricle [Chellappa '23]



3D dielectric filter [Chellappa '21]

AKSELOS

Some data-driven methods...

Data-driven method: Operator inference

Basics

- What if we have data $\mathcal{U}$ but no model ($\mathbf{E}$, $\mathbf{A}$, etc.)?

[6] B. Peherstorfer, K. Willcox, Data-Driven Operator Inference for Nonintrusive Projection-Based Model Reduction, Computer Methods in Applied Mechanics and Engineering 306, 2016

- What if we have data $\mathcal{U}$ but no model ($\mathbf{E}$, $\mathbf{A}$, etc.)?
- **Impose and learn model from data!**, e.g., $\dot{\mathbf{u}} = \mathbf{A}\mathbf{u} + \mathbf{H}(\mathbf{u} \otimes \mathbf{u}) + \mathbf{B}\mathbf{s}$

[6] B. Peherstorfer, K. Willcox, Data-Driven Operator Inference for Nonintrusive Projection-Based Model Reduction, Computer Methods in Applied Mechanics and Engineering 306, 2016

Data-driven method: Operator inference

Basics

- What if we have data $\mathcal{U}$ but no model ($\mathbf{E}$, $\mathbf{A}$, etc.)?
- **Impose and learn model from data!**, e.g., $\dot{\mathbf{u}} = \mathbf{A}\mathbf{u} + \mathbf{H}(\mathbf{u} \otimes \mathbf{u}) + \mathbf{B}\mathbf{s}$

Operator inference[6]

1. Apply POD on snapshot data $\mathcal{U} \in \mathbb{R}^{N \times N_t N_{\boldsymbol{\mu}}}$ to get projection matrix $\mathbf{V} \in \mathbb{R}^{N \times n}$
2. Project data on to $\mathbf{V}$

$$\widehat{\mathcal{U}} := \mathbf{V}^{\mathsf{T}}\mathcal{U} \in \mathbb{R}^{n \times N_t N_{\boldsymbol{\mu}}}$$

3. Compute derivative data $\dot{\widehat{\mathcal{U}}}$ for each column of $\widehat{\mathcal{U}}$
4. Solve

$$\underset{\widehat{\mathbf{A}},\widehat{\mathbf{H}},\widehat{\mathbf{B}}}{\arg\min} \left\| \begin{bmatrix} \widehat{\mathcal{U}}^{\mathsf{T}} & (\widehat{\mathcal{U}}^{\mathsf{T}} \otimes \widehat{\mathcal{U}}^{\mathsf{T}}) & \mathbf{S}^{\mathsf{T}} \end{bmatrix} \begin{bmatrix} \widehat{\mathbf{A}} \\ \widehat{\mathbf{H}} \\ \widehat{\mathbf{B}} \end{bmatrix} - \begin{bmatrix} \dot{\widehat{\mathcal{U}}} \end{bmatrix} \right\|_F^2$$

- Linear least-squares problem, can be computed efficiently;

[6] B. Peherstorfer, K. Willcox, Data-Driven Operator Inference for Nonintrusive Projection-Based Model Reduction, Computer Methods in Applied Mechanics and Engineering 306, 2016

Data-driven method: Operator inference

Basics

- What if we have data $\mathcal{U}$ but no model ($\mathbf{E}$, $\mathbf{A}$, etc.)?
- **Impose and learn model from data!**, e.g., $\dot{\mathbf{u}} = \mathbf{A}\mathbf{u} + \mathbf{H}(\mathbf{u} \otimes \mathbf{u}) + \mathbf{B}\mathbf{s}$

Operator inference[6]

1. Apply POD on snapshot data $\mathcal{U} \in \mathbb{R}^{N \times N_t N_{\boldsymbol{\mu}}}$ to get projection matrix $\mathbf{V} \in \mathbb{R}^{N \times n}$
2. Project data on to $\mathbf{V}$

$$\widehat{\mathcal{U}} := \mathbf{V}^{\mathsf{T}}\mathcal{U} \in \mathbb{R}^{n \times N_t N_{\boldsymbol{\mu}}}$$

3. Compute derivative data $\dot{\widehat{\mathcal{U}}}$ for each column of $\widehat{\mathcal{U}}$
4. Solve

$$\underset{\widehat{\mathbf{A}},\widehat{\mathbf{H}},\widehat{\mathbf{B}}}{\arg\min} \left\| \begin{bmatrix} \widehat{\mathcal{U}}^{\mathsf{T}} & (\widehat{\mathcal{U}}^{\mathsf{T}} \otimes \widehat{\mathcal{U}}^{\mathsf{T}}) & \mathbf{S}^{\mathsf{T}} \end{bmatrix} \begin{bmatrix} \widehat{\mathbf{A}} \\ \widehat{\mathbf{H}} \\ \widehat{\mathbf{B}} \end{bmatrix} - \begin{bmatrix} \dot{\widehat{\mathcal{U}}} \end{bmatrix} \right\|_F^2 + \gamma_1 \left\| \widehat{\mathbf{A}} \right\| + \gamma_2 \left\| \widehat{\mathbf{H}} \right\| + \gamma_3 \left\| \widehat{\mathbf{B}} \right\|$$

- Linear least-squares problem, can be computed efficiently; ... but need to regularize!

[6] B. Peherstorfer, K. Willcox, Data-Driven Operator Inference for Nonintrusive Projection-Based Model Reduction, Computer Methods in Applied Mechanics and Engineering 306, 2016

Data-driven method: Operator inference

Basics

- Any model structure can be imposed (but mostly higher-order polynomials)
- Arbitrary nonlinear models can be rewritten (by hand) using appropriate transformations as quadratic-bilinear systems $\rightarrow$ McCormick relaxation[7,8], Lift-Learn approach[9]
- Successfully applied to large-scale systems, e.g. $N \sim \mathcal{O}(10^6) - \mathcal{O}(10^7)$
- Numerous extensions...
  - Operator Inference + Hyperreduction [Benner et al. 2020]
  - Active learning with operator inference [Uy et al. 2021]

✓ No need to know model; can be inferred directly

✓ Interpretable

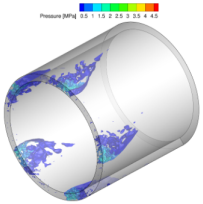
× Same drawbacks as POD-ROM

× Good regularization is crucial

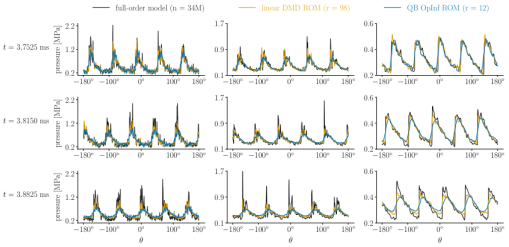
[7] C. Gu, QLMOR: A projection-based nonlinear model order reduction approach using quadratic-linear representation of nonlinear systems, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2011

[8] P. Benner, P. Goyal, S. Gugercin, $\mathcal{H}_2$-quasi-optimal model order reduction for quadratic-bilinear control systems, SIMAX 2018

[9] E. Qian, B. Kramer, B. Peherstorfer, K. Willcox, Lift & Learn: Physics-informed machine learning for large-scale nonlinear dynamical systems, Physica D: Nonlinear Phenomena, 2020



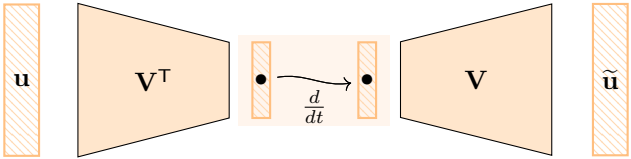
Rotating detonation engine



Approximation

- $N = 4$ million, $n = 12$

- I. Farcas, R. Gundevia, R. Munipalli, K. Willcox, Parametric non-intrusive reduced-order models via operator inference for large-scale rotating detonation engine simulations, 2024



Learn latent operators via LSQ

Figure: Operator inference in the unified structure

- Learn ROM from data, without imposing any model

POD-NN[10,11]

1. Apply POD on snapshot data $\mathcal{U} \in \mathbb{R}^{N \times N_t N_\mu}$ to get projection matrix $\mathbf{V} \in \mathbb{R}^{N \times n}$
2. Project data on to $\mathbf{V}$

$$\widehat{\mathcal{U}} := \mathbf{V}^\mathsf{T} \mathcal{U} \in \mathbb{R}^{n \times N_t N_\mu}$$

3. Learning parameter-to-POD coefficient map:

$$(t, \boldsymbol{\mu}) \to \widehat{\mathbf{u}}(t, \boldsymbol{\mu})$$

[10] S. Ubbiali, J. Hesthaven, Non-intrusive reduced order modeling of nonlinear problems using neural networks, Journal of Computational Physics, 2018

[11] Q. Wang, J. Hesthaven, D. Ray, Non-intrusive reduced order modeling of unsteady flows using artificial neural networks with application to a combustion problem, Journal of Computational Physics, 2019

- Learn ROM from data, without imposing any model

POD-NN[10,11]

1. Apply POD on snapshot data $\mathcal{U} \in \mathbb{R}^{N \times N_t N_\mu}$ to get projection matrix $\mathbf{V} \in \mathbb{R}^{N \times n}$
2. Project data on to $\mathbf{V}$

$$\widehat{\mathcal{U}} := \mathbf{V}^\mathsf{T} \mathcal{U} \in \mathbb{R}^{n \times N_t N_\mu}$$

3. Learning parameter-to-POD coefficient map:

$$(t, \boldsymbol{\mu}) \to \widehat{\mathbf{u}}(t, \boldsymbol{\mu})$$

✓ No need to know model
✓ Easy implementation
× Choosing n
× Hyperparam. optimization

[10] S. Ubbiali, J. Hesthaven, Non-intrusive reduced order modeling of nonlinear problems using neural networks, Journal of Computational Physics, 2018

[11] Q. Wang, J. Hesthaven, D. Ray, Non-intrusive reduced order modeling of unsteady flows using artificial neural networks with application to a combustion problem, Journal of Computational Physics, 2019

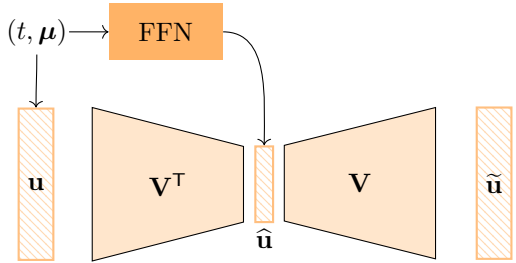


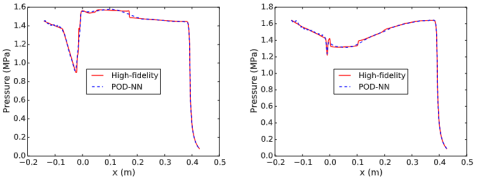
Figure: POD-NN in the unified structure

- Loss $L := \|\mathbf{V}^\mathsf{T}\mathbf{u}(t, \boldsymbol{\mu}) - \widehat{\mathbf{u}}_{\mathtt{ffn}}(t, \boldsymbol{\mu})\|^2$

Data-driven method: POD-NN

Application

- Continuously variable resonance combustor; 1-D Euler equations
- N $= 1200$, $n = 206$
- 4 parameters ($3^4 = 81$ samples), 1000 time steps



(a) $(t, \mu) = (0.1081, 303.2893, 1032.1611, 0.4309, 0.7908)$ (b) $(t, \mu) = (0.1006, 303.2893, 1035.8901, 0.4328, 0.7908)$

Q. Wang et al. / Journal of Computational Physics 384 (2019) 289–307

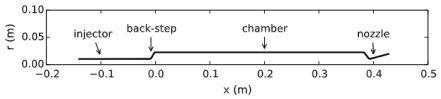


Fig. 2. Geometry of quasi-1D CVRC model.

Table 11
Test results of the POD-NN method for the ROM of the quasi-1D CVRC pressure field.

Average projection error, $\bar{\varepsilon}_V$	Average POD-NN error, $\bar{\varepsilon}_{PODNN}$
0.083 %	1.28 %

- 1 FOM solve: 10^3s; POD-NN: 10^{-3}s
- Training time: 10^5s
- Break-even: > 100 solutions

- Similar to POD-NN, but without POD step

DL-ROM[12]

1. Use convolutional autoencoder to learn data reconstruction:

$$\widetilde{\mathbf{u}}(t, \boldsymbol{\mu}) = \mathcal{D}(\mathcal{E}(\mathbf{u}(t, \boldsymbol{\mu})))$$

2. Use feed-forward network to learn $(t, \boldsymbol{\mu}) \rightarrow \widehat{\mathbf{u}}_{\mathtt{ffn}}(t, \boldsymbol{\mu})$
3. Jointly train both CAE and FFN with loss function

$$L := 0.5\|\mathbf{u}(t, \boldsymbol{\mu}) - \widetilde{\mathbf{u}}(t, \boldsymbol{\mu})\|^2 + 0.5\|\mathcal{E}(\mathbf{u}(t, \boldsymbol{\mu})) - \widehat{\mathbf{u}}_{\mathtt{ffn}}(t, \boldsymbol{\mu})\|^2$$

[12] S. Fresca, L. Dedé, A. Manzoni, A Comprehensive Deep Learning-Based Approach to Reduced Order Modeling of Nonlinear Time-Dependent Parametrized PDEs, Journal of Scientific Computing, 2021

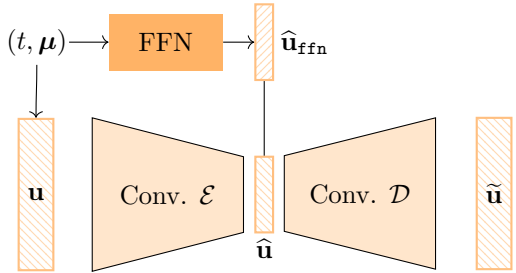
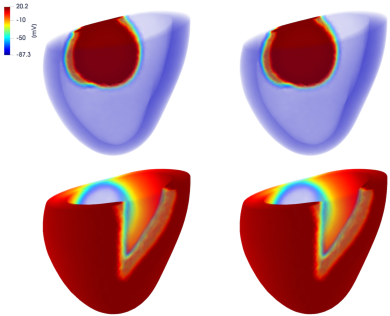


Figure: DL-ROM in the unified structure

- Loss $L := 0.5\|\mathbf{u}(t, \boldsymbol{\mu}) - \widetilde{\mathbf{u}}(t, \boldsymbol{\mu})\|^2 + 0.5\|\mathcal{E}(\mathbf{u}(t, \boldsymbol{\mu})) - \widehat{\mathbf{u}}_{\mathtt{ffn}}(t, \boldsymbol{\mu})\|^2$

- Cardiac electromechanics[13]:
- $N = 16,365$, DL-ROM $n = 10$, POD $n = 80$
- 25000 training samples



- FOM solve: 40 mins,
  DL-ROM solve: 1 minute
- DL-ROM training: 160 hours!
- Break-even: > 240 solves

[13] S. Fresca, A. Manzoni, L. Dedé, A' Quarteoni, Deep learning-based reduced order models in cardiac electrophysiology, PLOS One, 2020

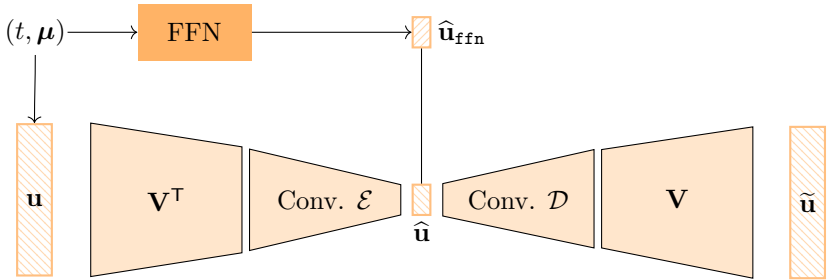


Figure: POD-DL-ROM in the unified structure

- Loss $L := 0.5\|\mathbf{V}\mathbf{V}^{\mathsf{T}}\mathbf{u}(t,\boldsymbol{\mu}) - \mathbf{V}\mathcal{D}(\mathcal{E}(\mathbf{V}^{\mathsf{T}}\mathbf{u}(t,\boldsymbol{\mu})))\|^2 + 0.5\|\mathcal{E}(\mathbf{V}^{\mathsf{T}}\mathbf{u}(t,\boldsymbol{\mu})) - \widehat{\mathbf{u}}_{\mathtt{ffn}}(t,\boldsymbol{\mu})\|^2$

- POD-DL-ROM for Navier-Stokes equations [14]
- $N = 64,892$, $n_{\text{POD}} = 256$, $n = 2$
- Parameter Reynolds number $[66, 133]$
- Training time: POD-DL-ROM 58 mins, inference 0.1 s vs.
- DL-ROM 58 hours

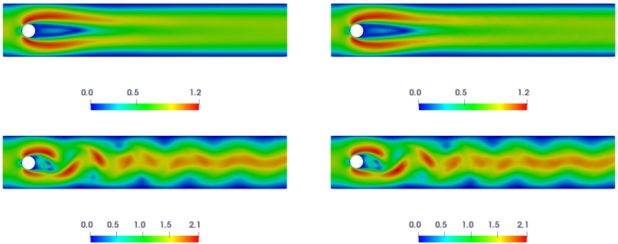


Fig. 12. *Test 4*: FOM (left) and POD-DL-ROM (right) solutions for the testing-parameter instances $\mu_{test} = 1.05$ (top) and $\mu_{test} = 1.75$ (bottom) at $t = 5.64$, with $n = 2$.

[14] S. Fesca, A. Manzoni, POD-DL-ROM: Enhancing deep learning-based reduced order models for nonlinear parametrized PDEs by proper orthogonal decomposition, CMAME, 2021

Conclusion

Summary

- Reduced-order modeling (ROM) has a long history in engineering/mathematics...
  active research topic in academia
- Projection-based/Physics-based ROM approaches are best when good base-model is available
  $\rightsquigarrow$ training cheaper, inference comparatively expensive (especially for linear ROM)
- Emerging data-driven/ML-based methods are suited when no underlying model is known/available
  $\rightsquigarrow$ costly training, inference is very fast
- Not discussed today:
  Nonlinear ROMs, Freq. domain methods, Koopman methods (DMD, SINDy), NeuralODEs, PINNs, Fourier neural operators, etc.

Conclusion

Summary

- Reduced-order modeling (ROM) has a long history in engineering/mathematics...
  active research topic in academia
- Projection-based/Physics-based ROM approaches are best when good base-model is available
  $\rightsquigarrow$ training cheaper, inference comparatively expensive (especially for linear ROM)
- Emerging data-driven/ML-based methods are suited when no underlying model is known/available
  $\rightsquigarrow$ costly training, inference is very fast
- Not discussed today:
  Nonlinear ROMs, Freq. domain methods, Koopman methods (DMD, SINDy), NeuralODEs, PINNs, Fourier neural operators, etc.

Thanks for your attention!

References

P. Benner, S. Gugercin, K. Willcox, A Survey of Projection-Based Model Reduction Methods for Parametric Dynamical Systems, *SIAM Review*, 2015

A. Quarteroni, A. Manzoni, F. Negri, Reduced Basis Methods for Partial Differential Equations, Springer, 2016

P. Benner, S. Grivet-Talocia, A. Quarteroni, G. Rozza, W. Schilders, L. Silveira, Eds., Model Order Reduction, Vol. 1, 2, & 3, De Gruyter, 2021