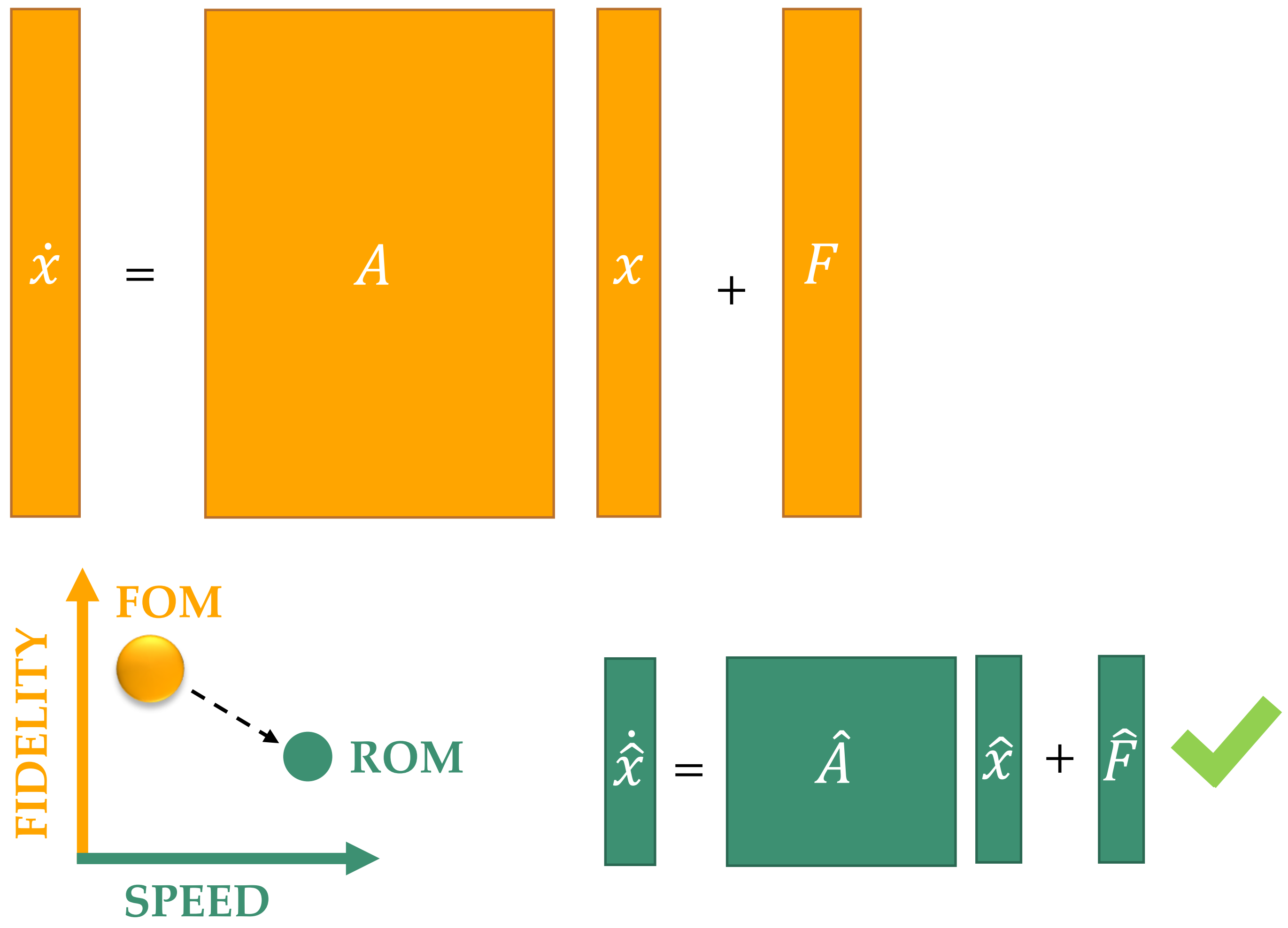


TAMIDS WORKSHOP

Reduced Order Modeling with pylibROM

Suparno Bhattacharyya, Pravija Danda,
Jian Tao, Eduardo Gildin,
Jean C. Ragusa,
libROM Team

Lawrence Livermore National Laboratory
TEXAS A&M UNIVERSITY

Prerequisites

- Introductory Finite Element Analysis
- Linear Algebra
- Numerical Analysis
- Familiarity with Dynamical Systems.
- Familiarity with Numerical Solutions of Ordinary/Partial Differential Equations.
- *Installed pylibROM docker container.*
- *Installed paraview.*

Table of Contents

Access to Jupyter Notebooks

https://github.com/TAMIDS-WORKSHOPS/pylibROM-workshop-material/

Session – 1: Introduction to Reduced Order Modeling and pylibROM

Suparno Bhattacharyya

Reduced Order/Surrogate Models Improve Computational Efficiency

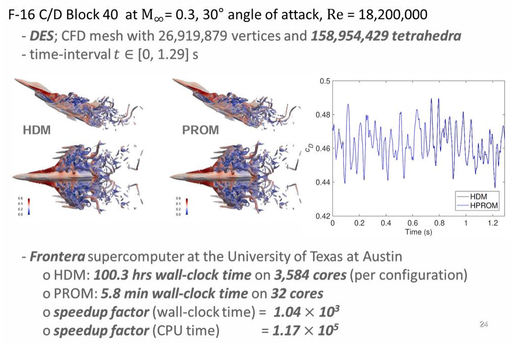
- Large-scale systems: turbulence modeling in fluid mechanics, neutron diffusion/neutron transport in nuclear engineering, reservoir simulation in petroleum engineering, simulations in computational mechanics/structural dynamics etc. are inherently high-dimensional and/or nonlinear.

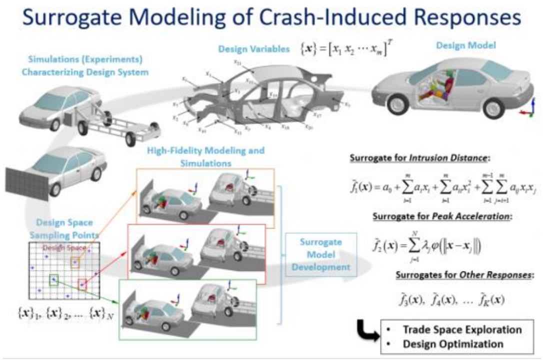
- Simulation demands consumption of vast computational resources (can take days/ weeks to complete!).

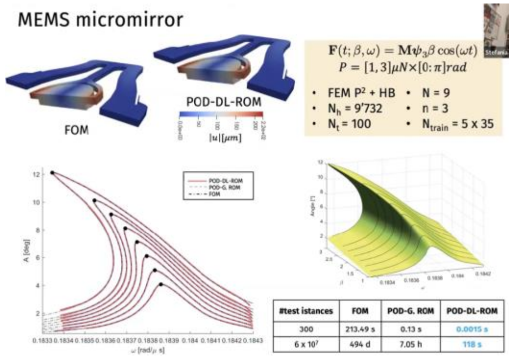
- ROMs, reduces complexity of such models, improving computational efficiency *without loosing much accuracy*.

- Embedded Systems, Robotic Surgery, Digital Twins, Hypersonics.

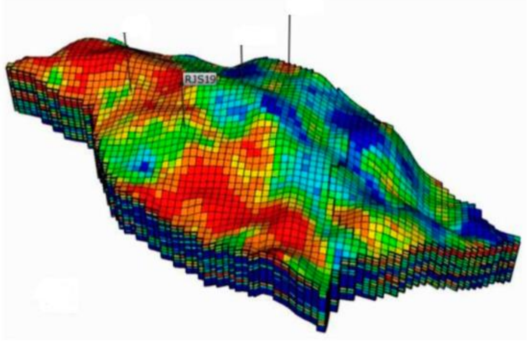
F-16, Re = 18,200,000

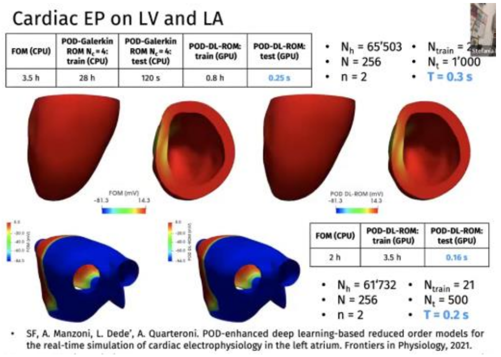


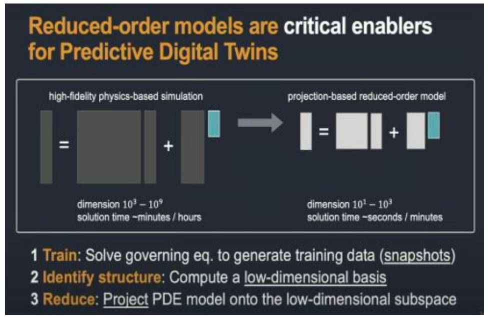




Reservoir Simulation







[1-6]

Reduced Order/Surrogate Models Improve Computational Efficiency

- Large-scale systems: turbulence modeling in fluid mechanics, neutron diffusion/neutron transport in nuclear engineering, reservoir simulation in petroleum engineering, simulations in computational mechanics/structural dynamics etc. are inherently high-dimensional and/or nonlinear.

- Simulation demands consumption of vast computational resources (can take days/ weeks to complete!).

- ROMs, reduces complexity of such models, improving computational efficiency *without loosing much accuracy*.

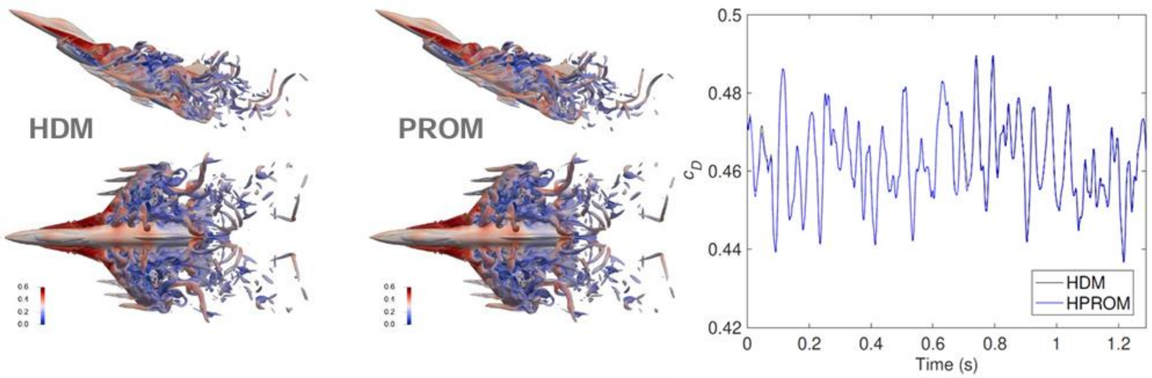
- Embedded Systems, Robotic Surgery, Digital Twins, Hypersonics.

[1]

F-16, Re = 18,200,000

F-16 C/D Block 40 at M_∞= 0.3, 30° angle of attack, Re = 18,200,000

- ***DES***; CFD mesh with 26,919,879 vertices and ***158,954,429 tetrahedra***
- time-interval $t \in [0, 1.29]$ s



- ***Frontera*** supercomputer at the University of Texas at Austin
  - o HDM: ***100.3 hrs wall-clock time*** on ***3,584 cores*** (per configuration)
  - o PROM: ***5.8 min wall-clock time*** on ***32 cores***
  - o ***speedup factor*** (wall-clock time) = ***1.04*** × ***10³***
  - o ***speedup factor*** (CPU time) = ***1.17*** × ***10⁵***

24

Relevance

- Compute-intensive science and applications:
  - Parametric studies, stochastic analysis, uncertainty analysis
  - Multidisciplinary modeling, multiscale modeling
  - Multiphysics design optimization, optimal control

- Time-critical applications (technology & industrial representatives):

  Boeing, Intel, Toyota, VW, ANSYS, etc.

- Funding agencies:

  DOD, DOE, NASA, NSF

[1]

- FAST SIMULATION
- IDEAL FOR MANY-QUERY COMPUTATION
  - DESIGN OPTIMIZATION
  - UNCERTAINTY QUANTIFICATION
  - OPTIMAL CONTROL

Reduced Order Modeling Strategies

INTRUSIVE

- Requires governing equations (high fidelity model). May or may not need data.
- Example: projection-based reduced order model (pROMs)
- Inherits physics of the system, at least partially
- Sessions: 3

REDUCED ORDER MODELING

NON-INTRUSIVE (SURROGATE)

- Requires only data
- Example:
    - Koopman operator/DMD
    - Sci-ML: Neural (O/P/S)DEs, Neural Operators
- Physics integrated separately (if needed)
- Session: 4 (DMD)

Reduced Order Modeling Strategies

INTRUSIVE

- Requires governing equations (high fidelity model). May or may not need data.
- Example: projection-based reduced order model (pROMs)
- Inherits physics of the system, at least partially
- Sessions: 3

REDUCED ORDER MODELING

NON-INTRUSIVE (SURROGATE)

- Requires only data
- Example:
  - Koopman operator/DMD
  - Sci-ML: Neural (O/P/S)DEs, Neural Operators
- Physics integrated separately (if needed)
- Session: 4 (DMD)

Two Major Categories of Problems Tackled Through Intrusive ROMs

INTRUSIVE

MODELS FORMULATED BASED ON SYSTEM-THEORETIC APPROACH

PROBLEM CATEGORIES

MODELS DESCRIBED WITH PARTIAL DIFFERENTIAL EQUATIONS

[14, 16]

Two Major Categories of Problems Tackled Through Intrusive ROMs

MODELS FORMULATED BASED ON SYSTEM-THEORETIC APPROACH

- HDM: Large-scale ODEs/DAEs; dynamics characterized by a large state vector, driven by input forcing.
- Output & Input states < true state-space dimension (*N*).
- <u>ROM obtained by projecting HDM onto a subspace that discards state variables poorly coupled to the inputs or barely contributing to the outputs.</u>
- Not strictly data-driven.
- Examples: Balanced Truncation, Moment-matching methods, etc.

[14,16,18]

Large-Scale Model

System-theoretic model / PDE after discretization

$$\mathbf{w} \in R^N$$

$$\Sigma : \begin{cases} \frac{d\mathbf{w}(\mathbf{x},t)}{dt} = \mathbf{F}(\mathbf{w}(\mathbf{x},t), \mathbf{u}(t)) \\ \mathbf{y}(t) = H(\mathbf{w}(\mathbf{x},t), \mathbf{u}(t)) \end{cases}$$

Inputs $\left\{ \begin{array}{l} u_1(t) \\ u_2(t) \\ \vdots \\ u_m(t) \end{array} \right.$ $\left. \begin{array}{l} y_1(t) \\ y_2(t) \\ \vdots \\ y_p(t) \end{array} \right\}$ Outputs

N: LARGE

$$\Sigma : \begin{cases} \dot{\mathbf{w}}(t) = \mathsf{A}\mathbf{w}(t) + \mathsf{B}\mathbf{u}(t) \\ \mathbf{y}(t) = \mathsf{C}\mathbf{w}(t) + \mathsf{D}\mathbf{u}(t) \end{cases}$$

Example: LTI systems

Two Major Categories of Problems Tackled Through Intrusive ROMs

$$\frac{\partial w(x,t)}{\partial t} = \underbrace{\frac{\partial}{\partial x} f(w(x,t))}_{\text{advection}} + \underbrace{\frac{\partial}{\partial x}\left(\kappa(x,t;\lambda)\frac{\partial w(x,t)}{\partial x}\right)}_{\text{diffusion}} + \underbrace{g(w(x,t);\gamma)}_{\text{reaction}}$$

FEM/FDM/FVM

$\mathbf{w} \in \mathbb{R}^N$

N: LARGE

$$\mathsf{M}\dot{\mathbf{w}} + \mathsf{K}(\lambda)\mathbf{w} = \mathbf{G}(\mathbf{w},\gamma)$$
$$\mathbf{w}(0;\mu) = \mathbf{w}^0(\mu)$$

MODELS DESCRIBED WITH PARTIAL DIFFERENTIAL EQUATIONS

- HDM: Parametric PDE defined over a continuous domain, discretized in space and time using FEM/FVM/FDM.
- Finer discretization to achieve higher accuracy, leads to large-scale ODEs.
- Solution over the entire domain is sought after.
- MOR process requires data.
- Reduced subspace is derived based on a few high-fidelity simulation data such that it captures maximum variability of the data.
- ROM obtained by projecting HDM onto a subspace.
- Examples: POD, PGD, Reduced Basis method, Hyper-reduction.

Two Major Categories of Problems Tackled Through Intrusive ROMs

$$\frac{\partial w(x,t)}{\partial t} = \underbrace{\frac{\partial}{\partial x} f(w(x,t))}_{\text{advection}} + \underbrace{\frac{\partial}{\partial x}\left(\kappa(x,t;\lambda)\frac{\partial w(x,t)}{\partial x}\right)}_{\text{diffusion}} + \underbrace{g(w(x,t);\gamma)}_{\text{reaction}}$$

FEM/FDM/FVM

$\mathbf{w} \in \mathbb{R}^N$

N: LARGE

$$\mathsf{M}\dot{\mathbf{w}} + \mathsf{K}(\lambda)\mathbf{w} = \mathbf{G}(\mathbf{w},\gamma)$$
$$\mathbf{w}(0;\mu) = \mathbf{w}^0(\mu)$$

MODELS DESCRIBED WITH PARTIAL DIFFERENTIAL EQUATIONS

- HDM: PDE defined over a continuous domain, discretized in space and time using FEM/FVM/FDM.
- Finer discretization to achieve higher accuracy, leads to large-scale ODEs.
- Solution over the entire domain is sought after.
- MOR process requires data.
- Reduced subspace is derived based on a few high-fidelity simulation data such that it captures maximum variability of the data.
- ROM obtained by projecting HDM onto a subspace.
- Examples: POD, PGD, Reduced Basis method, Hyper-reduction.

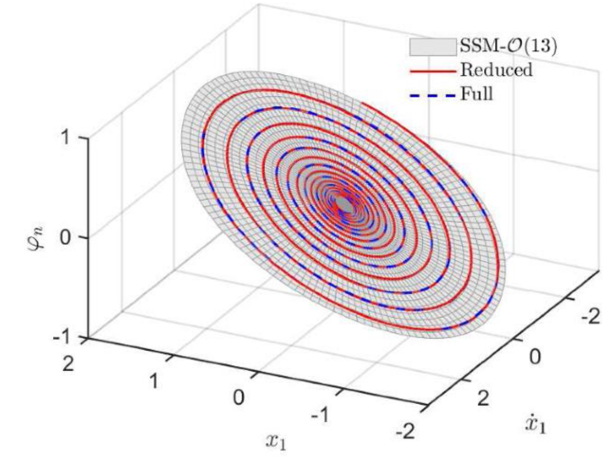
Projection-based Reduced Order Model

“Build the lowest-dimensional model that can capture the dominant behavior of the system of interest by projecting a given High-Dimensional computational Model (HDM) on a subspace constructed after learning something (data/physics) about the system of interest”

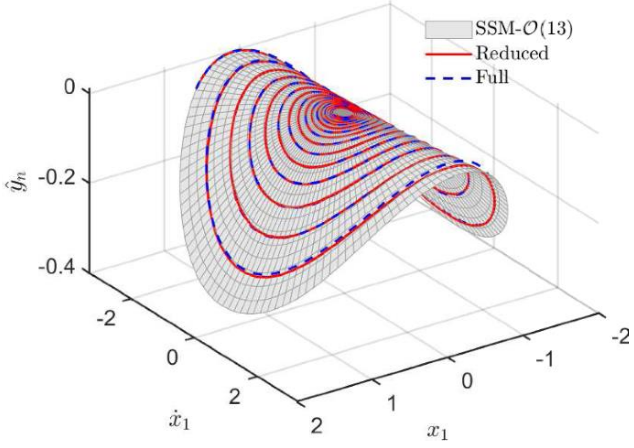
[1]

Fundamental Idea Behind MOR: Leverage Low-dimensionality of Solution Manifold

- The intrinsic dimensionality of the solutions of Large-Scale systems is often low, as these solutions reside in a lower-dimensional space within the high-dimensional state space.

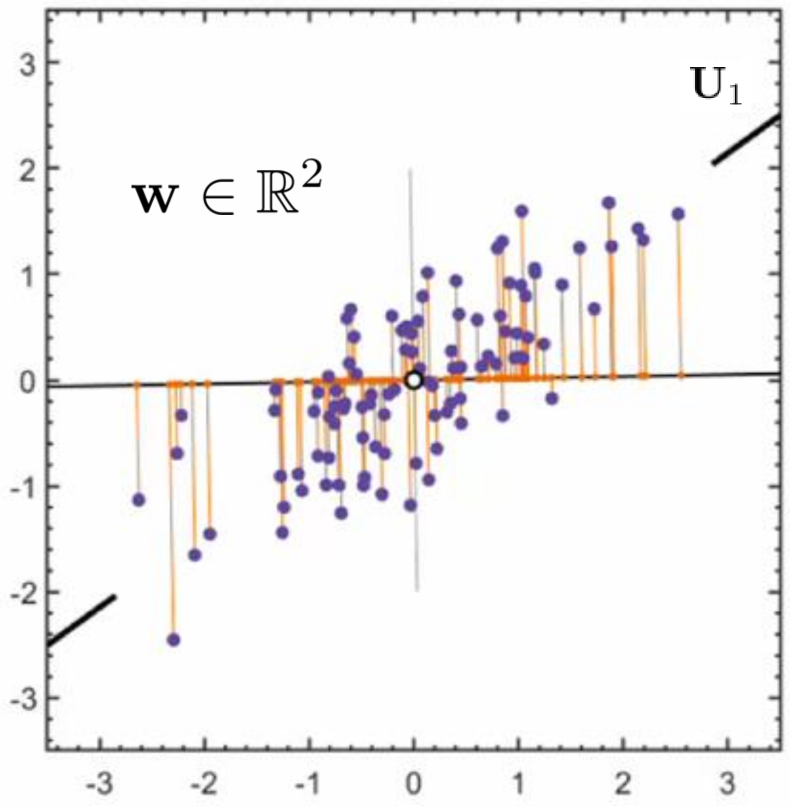


- Leveraging this phenomenon, model order reduction techniques derive reduced order models (ROMs) with fewer dimensions, while closely mimicking the behavior of complex, large-scale systems.



[15]

Identifying Intrinsic Dimensionality of Data Using POD/PCA



[18,20,24] **POD applied to 2D data**

Objective: Find a direction (or a unit vector **U**) that minimizes data reconstruction error

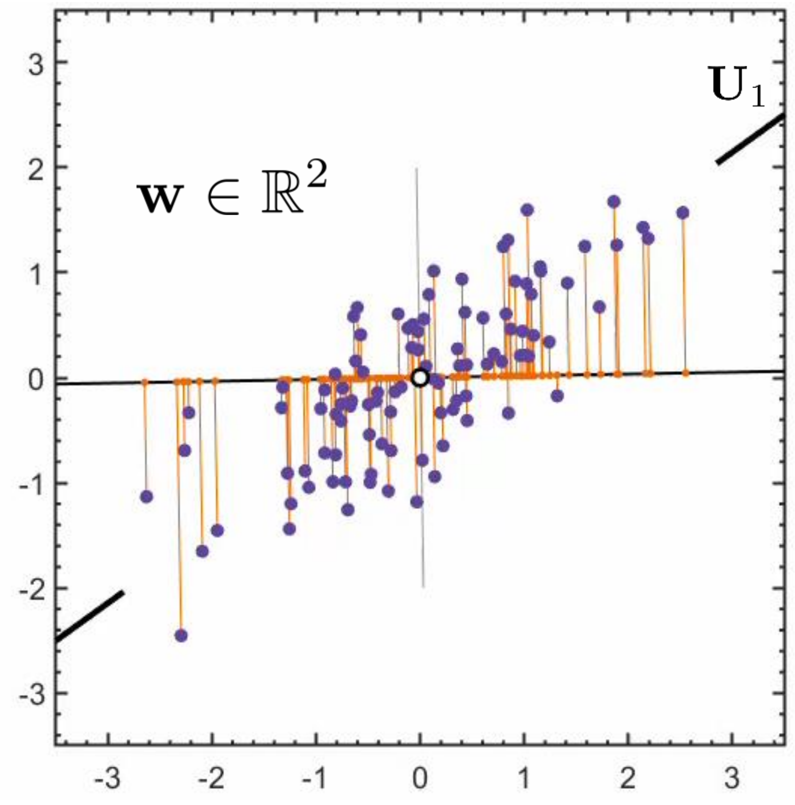
$$\mathbf{w} \in \mathbb{R}^N$$

$$E = \sum_{i=1}^{N_s} \|\mathbf{w}_i - \hat{\mathbf{w}}_i\|^2 \qquad \hat{\mathbf{w}}_i = ({\mathbf{w}_i}^T \mathbf{U})\mathbf{U}$$

$$E = \sum_{i=1}^{N_s} \|\mathbf{w}_i\|^2 - \boxed{({\mathbf{w}_i}^T \mathbf{U})^2}$$

Maximize this!

Identifying Intrinsic Dimensionality of Data Using POD/PCA



[18,20,24] **POD applied to 2D data**

Modified objective: Find a direction (or a vector **U**) that maximizes the variance of the projected data.

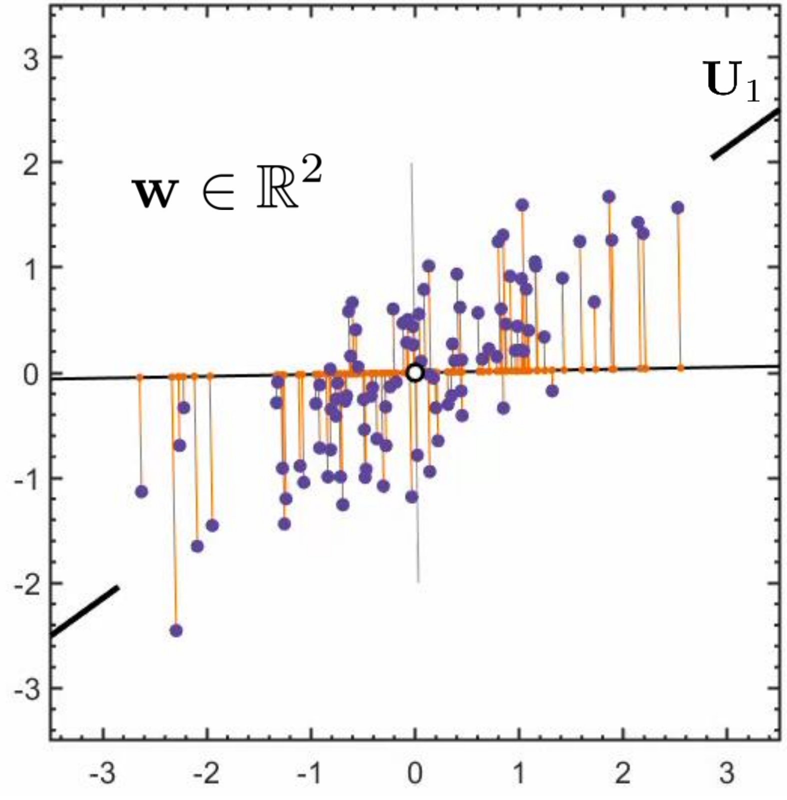
$$V = \frac{1}{N_s - 1}\sum_{i=1}^{N_s}(\mathbf{w}_i{}^T\mathbf{U})^2 = \frac{1}{N_s-1}\mathbf{U}^T\left(\sum_{i=1}^{N_s}\mathbf{w}_i\mathbf{w}_i^T\right)\mathbf{U} = \mathbf{U}^T\boxed{\left(\frac{1}{N_s-1}\mathrm{W}\mathrm{W}^T\right)}\mathbf{U}$$

Covariance matrix

$$\mathrm{W} = \begin{bmatrix} | & | & & | \\ \mathbf{w}_1 & \mathbf{w}_2 & \cdots & \mathbf{w}_{N_s} \\ | & | & & | \end{bmatrix}_{N \times N_s}$$

$$\max_{\|\mathbf{U}\|=1} \mathbf{U}^T \boxed{\mathrm{C}} \mathbf{U} \xrightarrow{\text{EVP}} \mathbf{C}\mathbf{U} = \lambda\mathbf{U}$$

POD provides optimal low-rank representation of the data



[18,20,24] **POD applied to 2D data**

$$\mathbf{CU} = \lambda \mathbf{U}$$

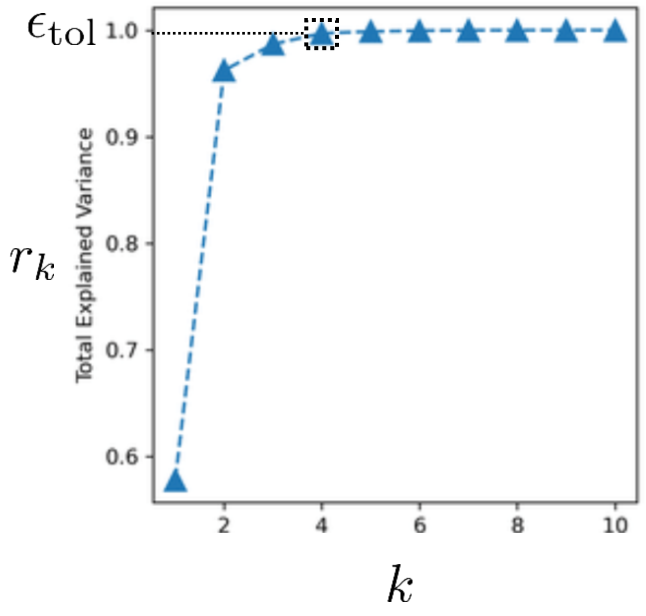
$\mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_N$ PROPER ORTHOGONAL MODES (POMs)

$\lambda_1 > \lambda_1 \ldots > \lambda_N$

$$r_k = \frac{\sum_{i=1}^{k} \boxed{\lambda_i}}{\sum_{i=1}^{N} \lambda_i}$$

Variance of the data along U_i, often referred to as "energies", a misnomer from a physics stand-point

$$n = \min_k r_k > \epsilon_{tol}$$

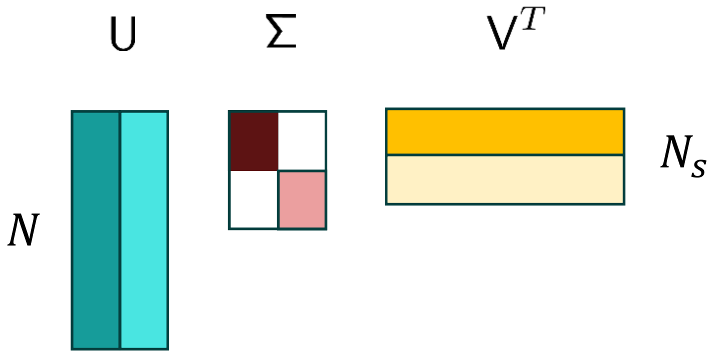


$$\mathsf{W} \approx \hat{\mathsf{W}} = \boxed{\mathsf{U}_n \mathsf{U}_n^T}\mathsf{W} \qquad \mathsf{U}_n = \begin{bmatrix} | & | & & | \\ \mathbf{U}_1 & \mathbf{U}_2 & \cdots & \mathbf{U}_n \\ | & | & & | \end{bmatrix}_{N \times n}$$

NOT $\mathrm{I}_{n\times n}$

Apply SVD on Data Directly to Obtain $\mathbf{U}$

$$\mathsf{W} = \begin{bmatrix} | & | & & | \\ \mathbf{w}_1 & \mathbf{w}_2 & \cdots & \mathbf{w}_{N_s} \\ | & | & & | \end{bmatrix}_{N \times N_s} =$$



$$\mathsf{C} = \frac{1}{N_s - 1}\mathsf{W}\mathsf{W}^T = \frac{1}{N_s - 1}\mathsf{U}\Sigma\mathsf{V}^T\mathsf{V}\Sigma\mathsf{U}^T = \mathsf{U}\left(\frac{1}{N_s - 1}\Sigma^2\right)\mathsf{U}^T = \mathsf{U}\Lambda\mathsf{U}^T$$

$$\lambda_{\mathrm{POD}_i} = \lambda^2_{\mathrm{SVD}_i}/(N_s - 1)$$

[18,24]

Deriving Projection-based MOR

- Approximate the states by a linear combination of basis vectors obtained by using POD on observed data.

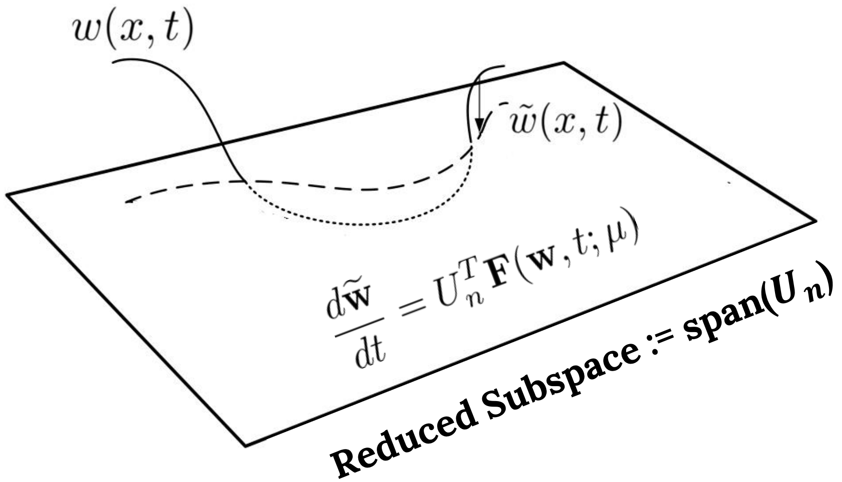
$$\boxed{\mathbf{w} \approx \sum_{i=1}^{n} \mathbf{U}_i \widetilde{w}_i = \mathsf{U}_n \widetilde{\mathbf{w}}} \qquad \widetilde{\mathbf{w}} \in \mathbb{R}^n$$

HIGH-FIDELITY MODEL (HFM)

$$\dot{\mathbf{w}}(\mathbf{x}, t) = \mathbf{F}(\mathbf{w}(\mathbf{x}, t), \mathbf{u}(t))$$

$$\boxed{\mathbf{w}^{\mathrm{sol}} = \mathsf{U}_n \widetilde{\mathbf{w}}_n^{\mathrm{sol}}}$$

GALERKIN PROJECTION



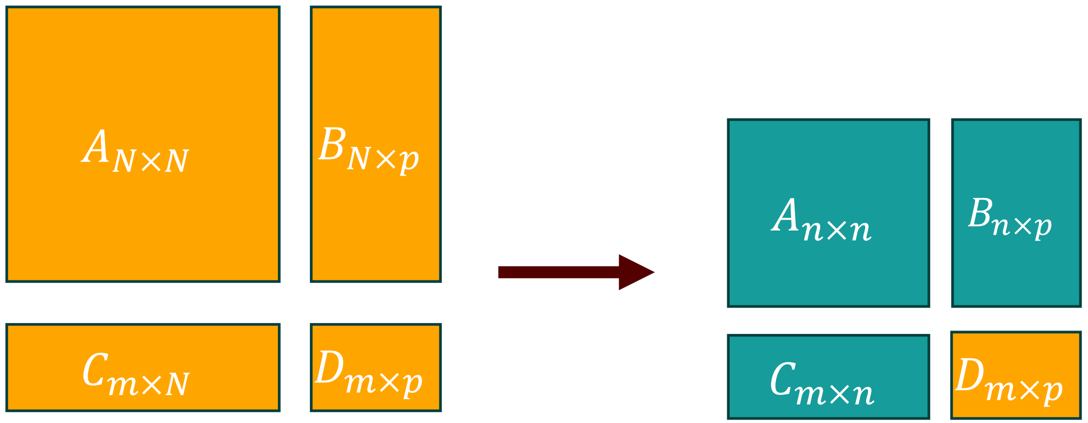
Deriving Projection-based MOR

- Approximate the states by a linear combination of basis vectors obtained by using POD on observed data.

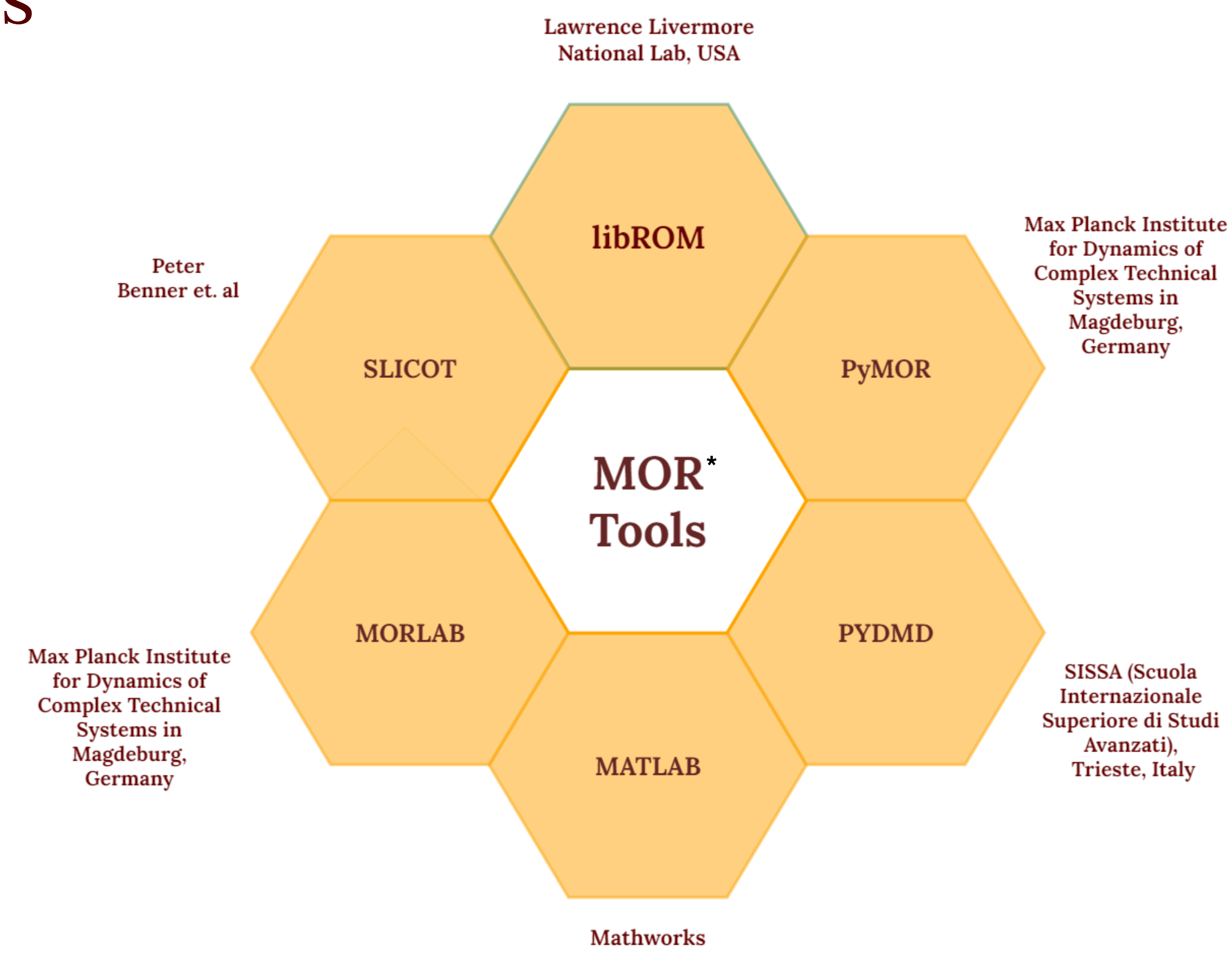
$$\boxed{\mathbf{w} \approx \sum_{i=1}^{n} \mathbf{U}_i \widetilde{w}_i = \mathsf{U}_n \widetilde{\mathbf{w}}}$$

Example: Reduced LTI

$$\Sigma_r : \begin{cases} \dot{\widetilde{\mathbf{w}}}_n(t) = \underbrace{\mathsf{U}^T\mathsf{A}\mathsf{U}}_{\widetilde{\mathsf{A}}_{n\times n}} \widetilde{\mathbf{w}}_n(t) + \underbrace{\mathsf{U}^T\mathsf{B}}_{\widetilde{\mathsf{B}}_{n\times p}} \mathbf{u}(t) \\ y_m(t) = \underbrace{\mathsf{CU}}_{\widetilde{\mathsf{C}}_{m\times n}} \widetilde{\mathbf{w}}_n(t) + \mathsf{D}\mathbf{u}(t) \end{cases}$$

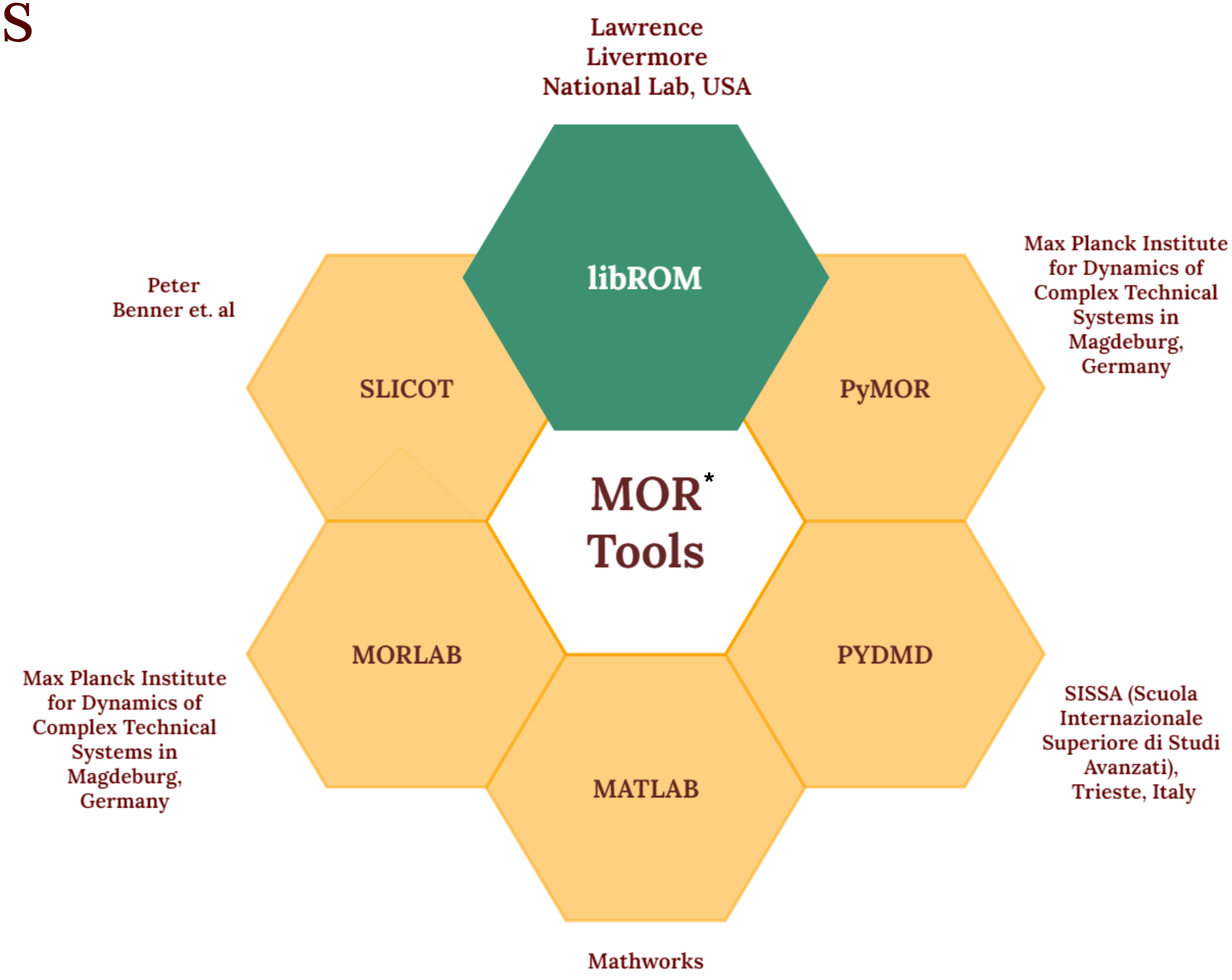


MOR Tools



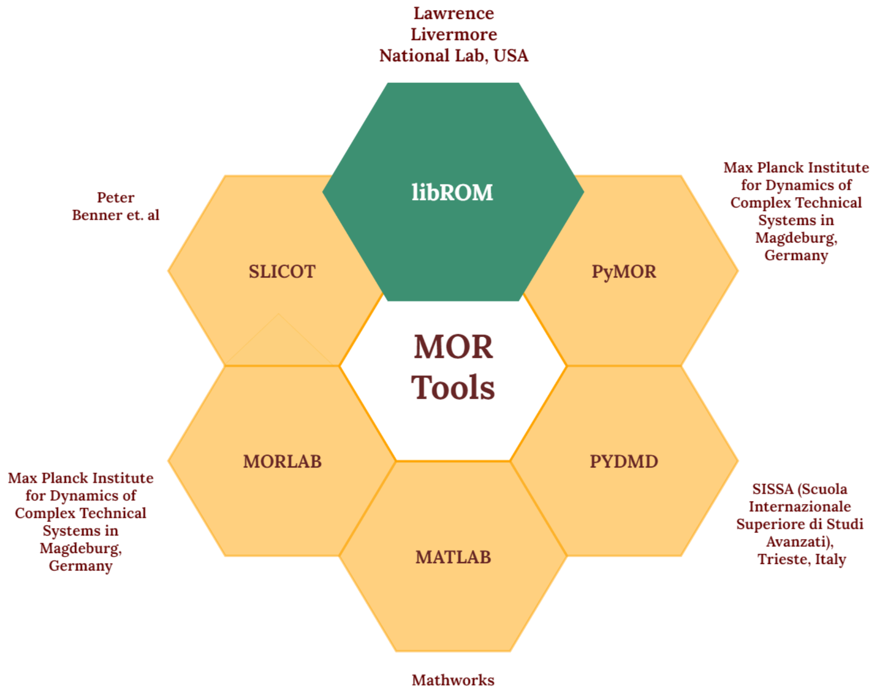
* list not exhaustive

MOR Tools



* list not exhaustive

MOR Tools



Contributors

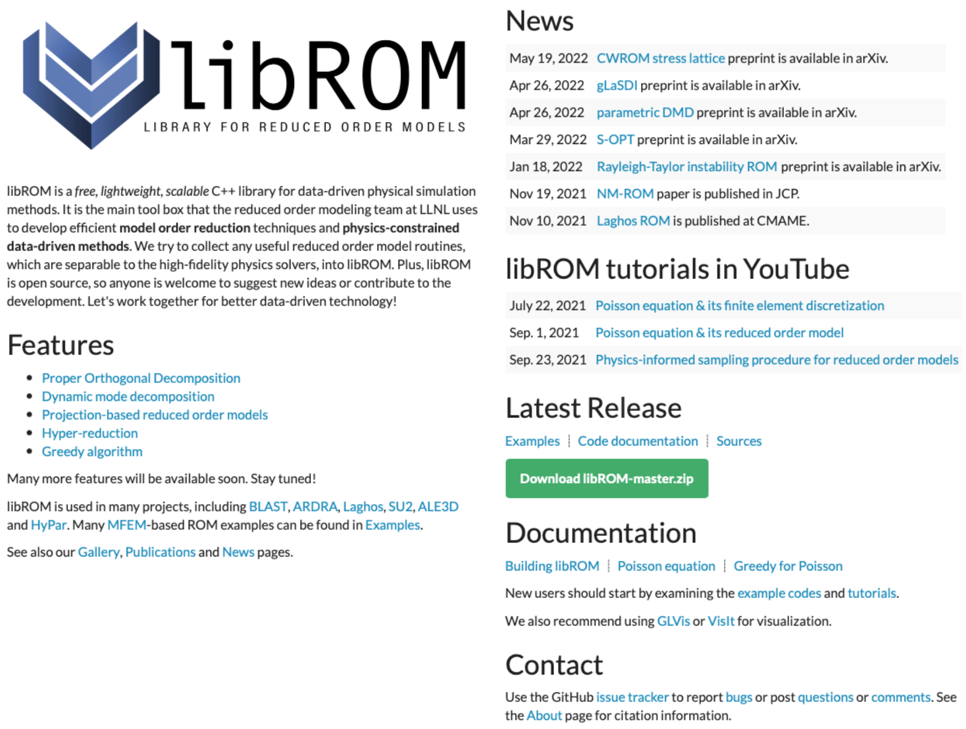
- Bob Anderson
- William Michael Anderson
- William Arrighi
- Suparno Bhattacharyya
- Kyle Chand
- Siu Wun Cheung
- Eric Chin
- Youngsoo Choi
- "Kevin" Seung Whan Chung
- Dylan Copeland
- Pravija Danda
- William Fries
- Debojyoti Ghosh
- Xiaolong He
- Kevin Huynh
- Coleman James Kendrick
- Tanya Kostova-Vassilevska
- Axel Larsson
- Jessica Lauzon
- Sean McBane
- Geoffrey Oxberry
- Pratanu Roy
- Yeonjong Shin
- Jian Tao
- Paul Jeffrey Tranquilli

Introduction to pylibROM

Suparno Bhattacharyya

pylibROM: open-source python library from LLNL for data-driven physical simulations

- GitHub repo for libROM: https://github.com/LLNL/libROM
- Webpage for libROM: https://www.librom.net
- GitHub repo for pylibROM: https://github.com/LLNL/pylibROM/



libROM
LIBRARY FOR REDUCED ORDER MODELS

libROM is a *free*, *lightweight*, *scalable* C++ library for data-driven physical simulation methods. It is the main tool box that the reduced order modeling team at LLNL uses to develop efficient **model order reduction** techniques and **physics-constrained data-driven methods**. We try to collect any useful reduced order model routines, which are separable to the high-fidelity physics solvers, into libROM. Plus, libROM is open source, so anyone is welcome to suggest new ideas or contribute to the development. Let's work together for better data-driven technology!

Features

- Proper Orthogonal Decomposition
- Dynamic mode decomposition
- Projection-based reduced order models
- Hyper-reduction
- Greedy algorithm

Many more features will be available soon. Stay tuned!

libROM is used in many projects, including BLAST, ARDRA, Laghos, SU2, ALE3D and HyPar. Many MFEM-based ROM examples can be found in Examples.

See also our Gallery, Publications and News pages.

News

May 19, 2022	CWROM stress lattice preprint is available in arXiv.
Apr 26, 2022	gLaSDI preprint is available in arXiv.
Apr 26, 2022	parametric DMD preprint is available in arXiv.
Mar 29, 2022	S-OPT preprint is available in arXiv.
Jan 18, 2022	Rayleigh-Taylor instability ROM preprint is available in arXiv.
Nov 19, 2021	NM-ROM paper is published in JCP.
Nov 10, 2021	Laghos ROM is published at CMAME.

libROM tutorials in YouTube

July 22, 2021	Poisson equation & its finite element discretization
Sep. 1, 2021	Poisson equation & its reduced order model
Sep. 23, 2021	Physics-informed sampling procedure for reduced order models

Latest Release

Examples | Code documentation | Sources

Download libROM-master.zip

Documentation

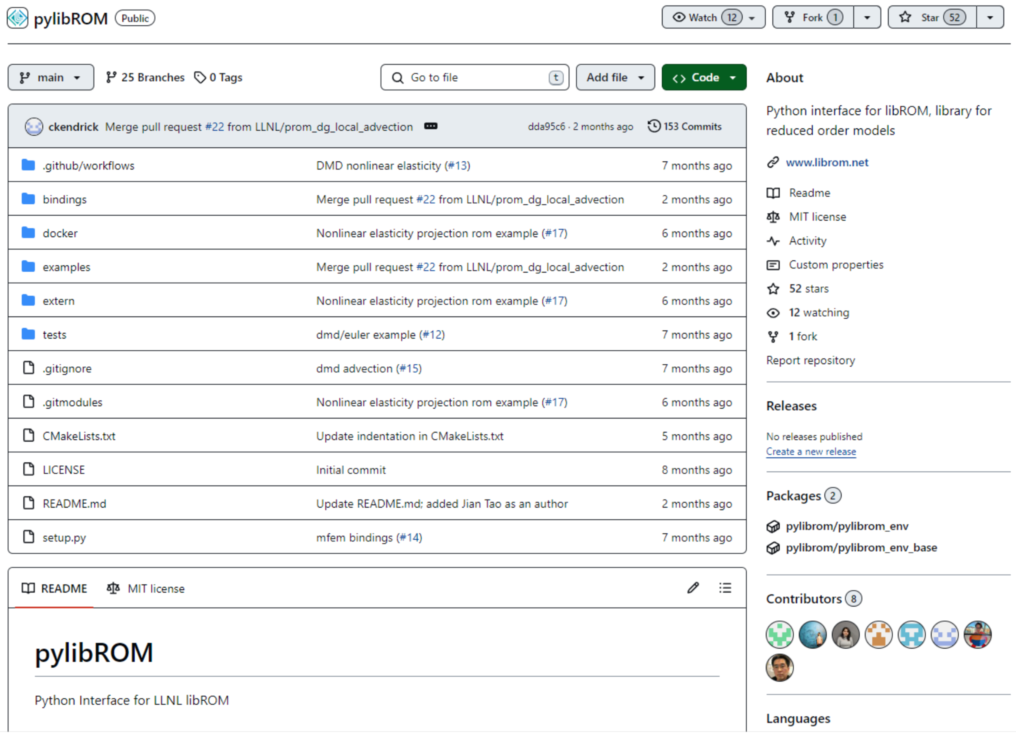
Building libROM | Poisson equation | Greedy for Poisson

New users should start by examining the example codes and tutorials.

We also recommend using GLVis or VisIt for visualization.

Contact

Use the GitHub issue tracker to report bugs or post questions or comments. See the About page for citation information.



pylibROM Public

Watch 12 | Fork 1 | Star 52

main | 25 Branches | 0 Tags | Go to file | Add file | Code

ckendrick Merge pull request #22 from LLNL/prom_dg_local_advection | dda95c6 · 2 months ago | 153 Commits

.github/workflows	DMD nonlinear elasticity (#13)	7 months ago
bindings	Merge pull request #22 from LLNL/prom_dg_local_advection	2 months ago
docker	Nonlinear elasticity projection rom example (#17)	6 months ago
examples	Merge pull request #22 from LLNL/prom_dg_local_advection	2 months ago
extern	Nonlinear elasticity projection rom example (#17)	6 months ago
tests	dmd/euler example (#12)	7 months ago
.gitignore	dmd advection (#15)	7 months ago
.gitmodules	Nonlinear elasticity projection rom example (#17)	6 months ago
CMakeLists.txt	Update indentation in CMakeLists.txt	5 months ago
LICENSE	Initial commit	8 months ago
README.md	Update README.md: added Jian Tao as an author	2 months ago
setup.py	mfem bindings (#14)	7 months ago

README | MIT license

pylibROM

Python Interface for LLNL libROM

About

Python interface for libROM, library for reduced order models

www.librom.net

Readme
MIT license
Activity
Custom properties
52 stars
12 watching
1 fork
Report repository

Releases

No releases published
Create a new release

Packages 2

pylibrom/pylibrom_env
pylibrom/pylibrom_env_base

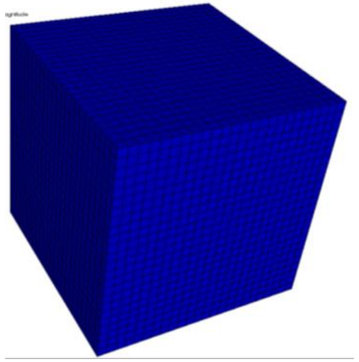
Contributors 8

Languages

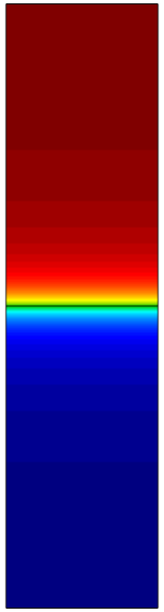
pylibROM: Intrusive MOR

Sedov blast
Explosion
(Lagrange ROM)

Relative error: $\mathbf{10^{-5}}$
Speedup: 26.5



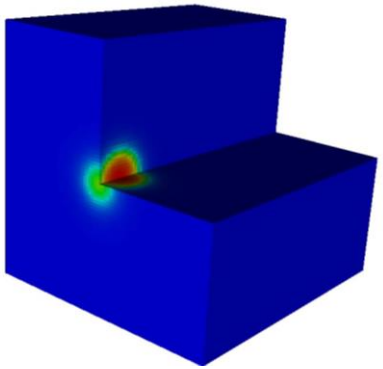
Rayleigh-Taylor
Instability
(Lagrange ROM)



Relative error: $\mathbf{10^{-5}}$
Speedup: 48.6

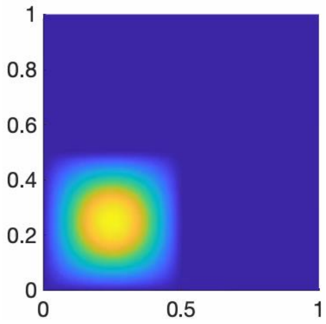
Particle
Transport
(Space-time
ROM)

Relative error: $\mathbf{10^{-2}}$
Speedup: 2700

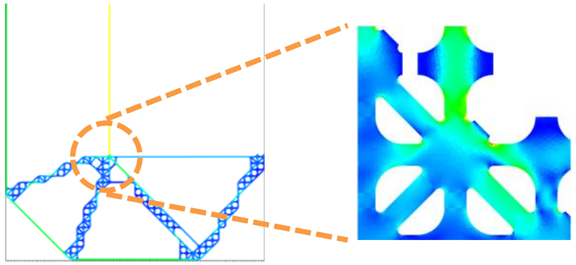


2D Burgers –
advection
(Nonlinear
manifold ROM)

Relative error: $\mathbf{10^{-2}}$
Speedup: 11.6



Relative error: $\mathbf{5 \times 10^{-2}}$
Speedup: 150

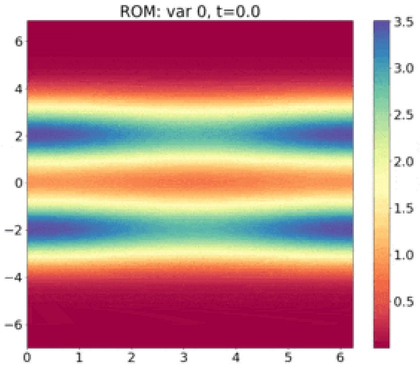


Design Optimization
(Component-wise ROM)

[7-11]

pylibROM: Non-intrusive MOR (DMD)

[12, 13]

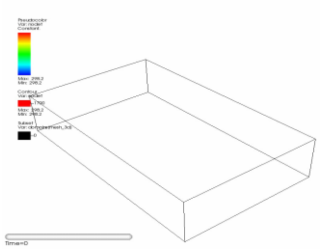


1D1V Vlasov equation

Relative error: $\mathbf{10^{-5}}$

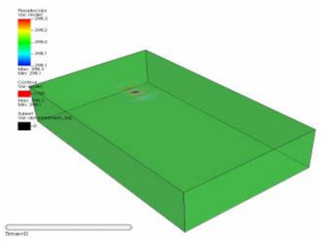
Speedup: 26.5

Full-order model



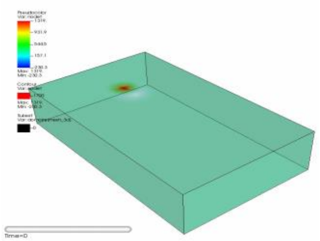
- 100 μs thermal simulation in ALE3D
- ~25k DOF
- ~1 hr analysis time

ROM dimension = 80

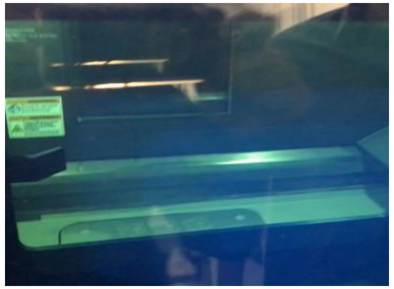


- 0.1% max relative error
- 0.07 s prediction time
- ~50,000x speedup

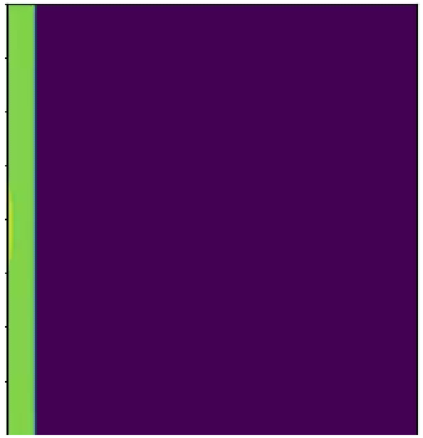
ROM dimension = 20



- 8.1% max relative error
- 0.007 s prediction time
- ~500,000x speedup



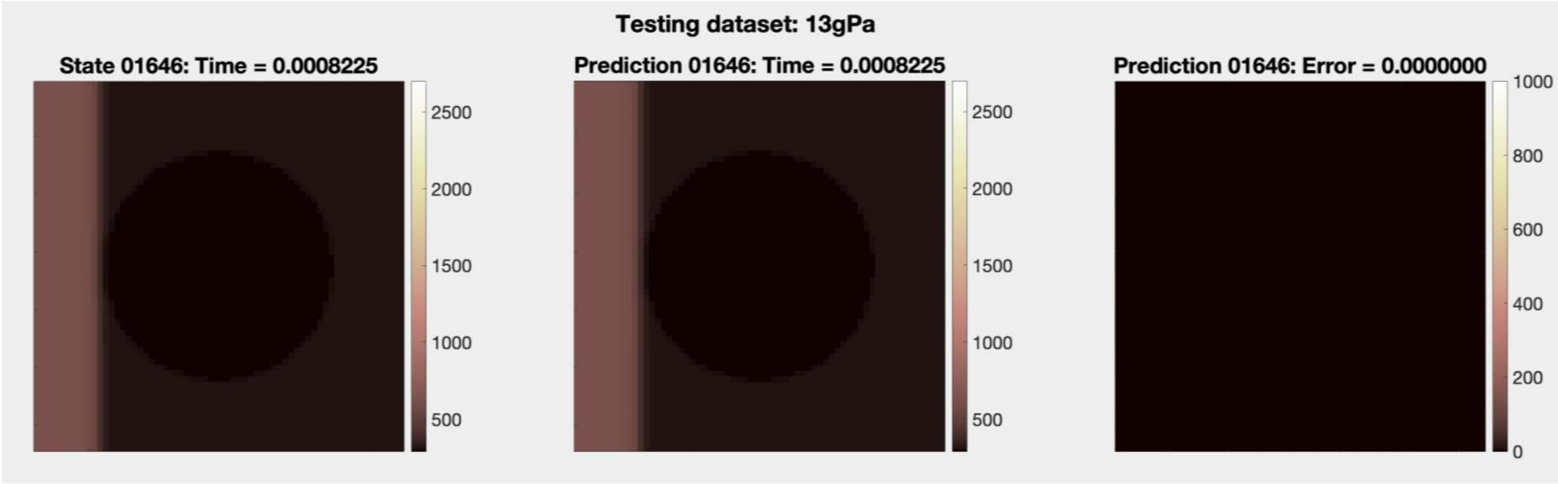
Laser ray tracing by ALE3D



Shock-induced flyer plate

Relative error: $\mathbf{10^{-5}}$

Speedup: 26.5



High fidelity simulation:
~1 week on 1024 cores

DMD prediction:
~1 minute on 1 core

Pore collapse by ALE3D

Session– 2:
Setting Up pylibROM

Pravija Danda

APPROACH – 1:
Native Installation

Required Software and Dependencies for pylibROM

Operating System Base: Ubuntu 22.04 (pylibROM is only compatible with Ubuntu)

C/C++ Development Tools:

- make : Build automation tool.
- gcc and gfortran : Compiler for C and Fortran programming languages.
- libssl-dev : Development files for OpenSSL cryptographic library.
- cmake : Cross-platform build system.

Python Dependencies: Various Python libraries required by pylibROM :-

- numpy, scipy, argparse, tables, pyYAML, h5py, pybind11, pytest, mpi4py
- These dependencies can be installed using pip.

SWIG : Simplified Wrapper and Interface Generator for connecting C/C++ libraries with Python.

pyMFEM : Finite element simulation library for solving partial differential equations.

Install Dependencies

- sudo apt install git make gcc gfortran libssl-dev cmake libopenblas-dev libmpich-dev libblas-dev liblapack-dev libscalapack-mpi-dev libhdf5-mpi-dev hdf5-tools pkg-config wget python3.8

- pip install swig==4.1.1 mpi4py numpy scipy argparse tables pyyaml h5py pybind11 pytest

- pip install mfem --install-option="--with-parallel" --install-option="--with-gslib" --verbose

Installation Guide for pylibROM

- Clone Repository and Sub-modules :

```
$ git clone --recurse-submodules git@github.com:LLNL/pylibROM.git
```

- Compile and Build pylibROM(from top-level pylibROM repo) :

```
$ pip install ./
```

- Speed Up Build : If libROM has been pre-compiled, you can speed up the build process using the following command : -

```
$ pip install ./ --global-option="--librom_dir=/path/to/pre-installed-libROM"
```

APPROACH – 2:
Using Docker Container

Introduction to Docker :

- What is Docker?
- Docker is a containerization platform that allows developers to package applications and their dependencies into lightweight containers.
- Containers are isolated environments that contain everything needed to run an application, including the code, runtime, system tools, libraries, and settings.
- Installing Docker :
- Docker can be installed on various operating systems, including Linux, Windows, and macOS.
- Linux (Ubuntu) : Docker can be installed on Ubuntu using package managers such as *apt*. Follow the installation methods outlined in the official Docker documentation for Ubuntu.
- Windows and macOS : Docker provides Docker Desktop for both Windows and macOS. Docker Desktop includes the Docker Engine, Docker CLI, and Docker Compose. Users can download and install Docker Desktop from the official Docker website.

pylibROM using Docker :

- Clone Repository and Sub-modules :

```
$ git clone --recurse-submodules git@github.com:LLNL/pylibROM.git
```

- To install pylibROM using Docker, pull the pre-built Docker image from the following location:

```
$ docker pull ghcr.io/llnl/pylibrom/pylibrom_env:latest
```

- After pulling the Docker image, execute the following command to run the Docker image. The below command mounts the local directory containing pylibROM code into the Docker container.

```
$ docker run -it --volume /path/to/folder/pylibROM:/home/test/pylibROM
ghcr.io/llnl/pylibrom/pylibrom_env:latest
```

- Once the Docker container is running, compile and build pylibROM using the following command(make sure that you are in the folder which contains setup.py):

```
$ pip install ./ --global-option="--librom_dir=/env/dependencies/libROM"
```

Using pylibROM in Jupyter Notebook

- If you prefer using pylibROM in a Jupyter Notebook environment, pull the below docker image specifically designed for this purpose.

```
$ docker pull suparnob100/pylibrom_jupyter:latest
```

- Once the image is built, you can run a container and start a Jupyter Notebook server. Replace /path/to/host/folder with the absolute path to the local directory you want to mount inside the container for Jupyter notebooks:

```
$ docker run -p 8888:8888 -v /path/to/host/folder:/notebooks -w /notebooks pylibrom_jupyter:latest
```

- Note: Ensure Docker is installed on your system before using the Docker images. These Docker containers come pre-configured with all the dependencies required for running pylibROM.

Break: 10 mins

Session– 3: pylibROM in action

Part-1: Fundamentals of Coding with pylibROM

Pravija Danda

Session– 3: pylibROM in action

Part-2: MOR Example

Suparno Bhattacharyya

Steady-state Linear Heat Conduction

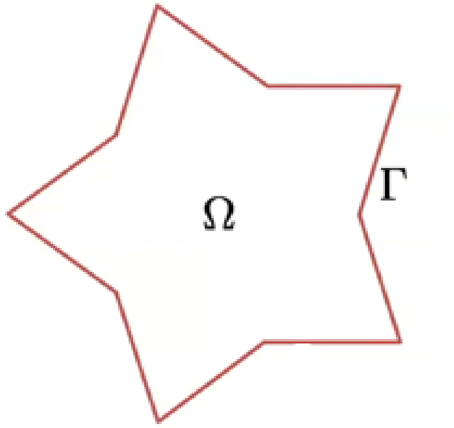
Steady-state linear heat conduction model provides a simplified yet representative parametric partial differential equation (PDE) for case study.

$$\boxed{\nabla \cdot (k \nabla T(\mathbf{x})) + q(\mathbf{x}, \mu) = 0}$$

Parameter

Poisson's Eq. defined over some domain **Ω** with boundary **Γ**

- Fixed temperature at boundary (Dirichlet) : $T|_{\Gamma_d} = T_b$
- Fixed heat flux at boundary (Neuman) : $-k \frac{\partial T}{\partial n}\Big|_{\Gamma_n} = q_n$
- Robin (Mixed) Condition : $-k \frac{\partial T}{\partial n}\Big|_{\Gamma_r} + h(T - T_{\text{ext}})|_{\Gamma_r} = 0$



Weak-form for FEM Analysis

$$\nabla \cdot (k \nabla T) + q(\mathbf{x}, \mu) = 0$$

$$\int_\Omega [\nabla \cdot (k \nabla T) + q(\mathbf{x}, \mu)]\, v(\mathbf{x})\, d\Omega = 0$$

$$\int_\Omega k \nabla T \cdot \nabla v \, d\Omega - \int_\Gamma k(\nabla T \cdot \mathbf{n}) v \, d\Gamma = \int_\Omega q(\mathbf{x}, \mu) v \, d\Omega$$

$$\nabla \cdot (\psi \mathbf{A}) = \psi \, \nabla \cdot \mathbf{A} + (\nabla \psi) \cdot \mathbf{A}$$

&

$$\int_\Omega \nabla \cdot \mathbf{u} \, d\Omega = \int_\Gamma \mathbf{u} \cdot \mathbf{n} \, d\Gamma$$

[23]

Weak-form for FEM Analysis

$$\nabla \cdot (k \nabla T) + q(\mathbf{x}, \mu) = 0$$

$$\int_\Omega \left[\nabla \cdot (k \nabla T) + q(\mathbf{x}, \mu)\right] v(\mathbf{x}) \, d\Omega = 0$$

$$\boxed{\int_\Omega k \nabla T \cdot \nabla v \, d\Omega} - \int_\Gamma k(\nabla T \cdot \mathbf{n}) v \, d\Gamma = \int_\Omega q(\mathbf{x}, \mu) v \, d\Omega$$

Bi-linear form

Remember jargon for MFEM example later

Boundary term

Dirichlet

linear form

Remember jargon for MFEM example later

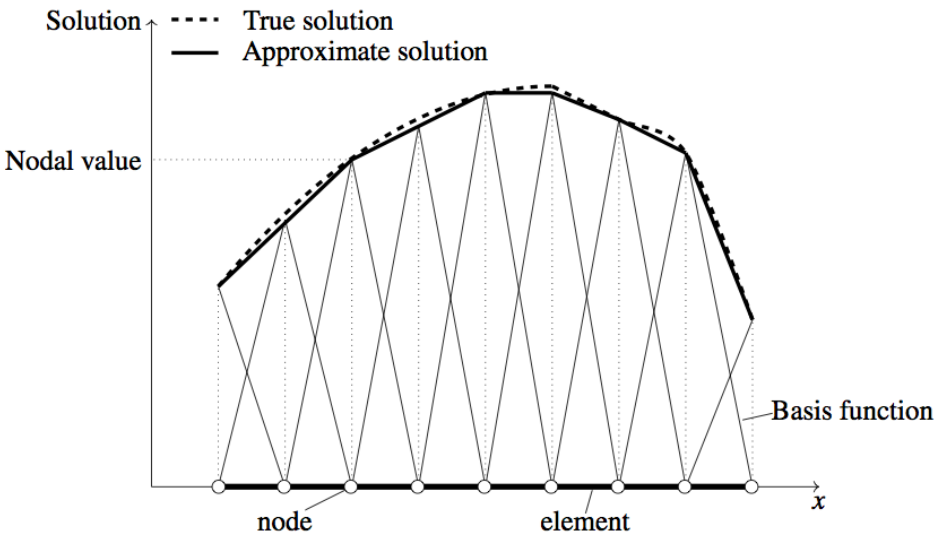
Solution as the Linear Combination of FE Bases

$$T(\mathbf{x}) = \sum_{i=1}^{N} T_i \boxed{\Psi_i(\mathbf{x})}$$

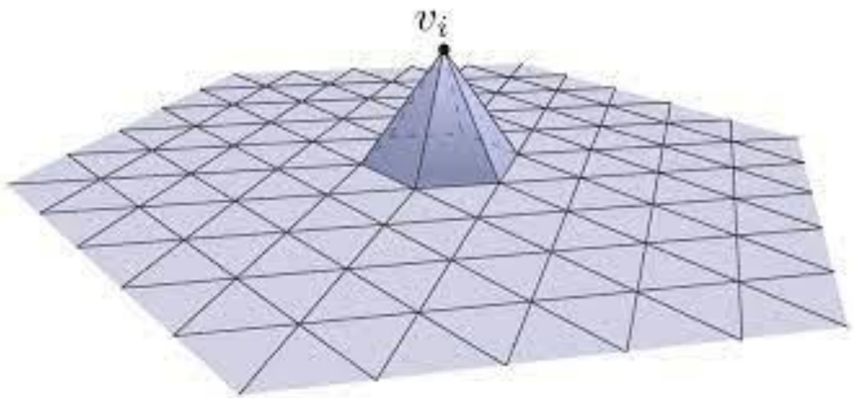
Shape/basis function

$$\mathbf{T} = [T_1, T_2, T_3, \ldots, T_N]^{\mathrm{T}}$$

1D



2D



Assembled FE Model

$$T(\mathbf{x}) = \sum_{i=1}^{N} T_i \boxed{\Psi_i(\mathbf{x})} \longrightarrow \int_\Omega k \nabla T \cdot \nabla v \, d\Omega = \int_\Omega q(\mathbf{x}, \mu) v \, d\Omega$$

Shape function

$$\boxed{\mathsf{K}_{N\times N} \mathbf{T}_{N\times 1} = \mathbf{Q}_{N\times 1}}$$

Solve with pyMFEM

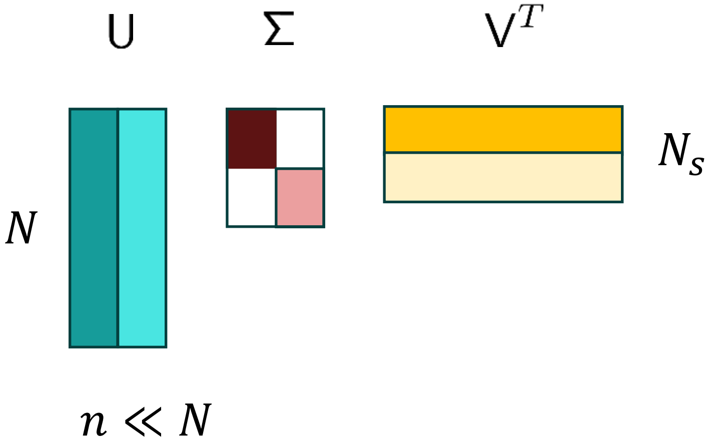
$$K_{ij} = \sum_{e=1}^{n_e} \int_{\Omega_e} k \nabla \Psi_i^e \cdot \nabla \Psi_j^e \, d\Omega_e$$

$$Q_i = \sum_{e=1}^{n_e} \int_{\Omega_e} \Psi_i^e q(\mathbf{x}, \mu) \, d\Omega_e$$

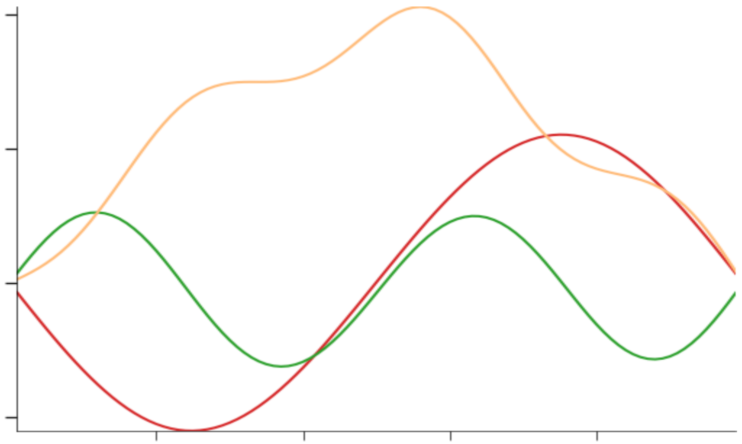
Study with large set of parameters becomes computationally EXPENSIVE for LARGE ***N***

Unlike FE Basis, POMs are Defined Over the Entire Domain

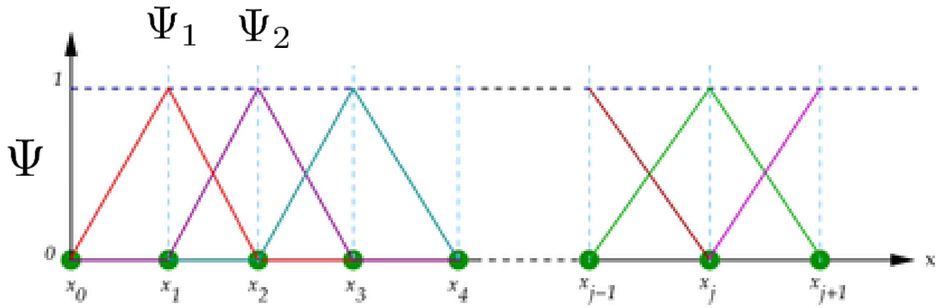
$$\begin{bmatrix} | & | & & | \\ \mathbf{T}^{\mu_1} & \mathbf{T}^{\mu_2} & \cdots & \mathbf{T}^{\mu_{N_s}} \\ | & | & | & | \end{bmatrix}_{N \times N_s} = U \Sigma V^T$$



$$\widetilde{\mathsf{U}} = \mathsf{U}[:, :n] \in \mathbb{R}^{N \times n}$$



Shape functions



Use POMs to Build the ROM

$$\widetilde{\mathsf{U}} = \mathsf{U}[:, : n] \in \mathbb{R}^{N \times n}$$

$$\mathsf{K}_{N \times N} \mathbf{T}_{N \times 1} = \mathbf{Q}_{N \times 1}$$

$$\mathbf{T} = \widetilde{\mathsf{U}} \mathbf{T}_n$$

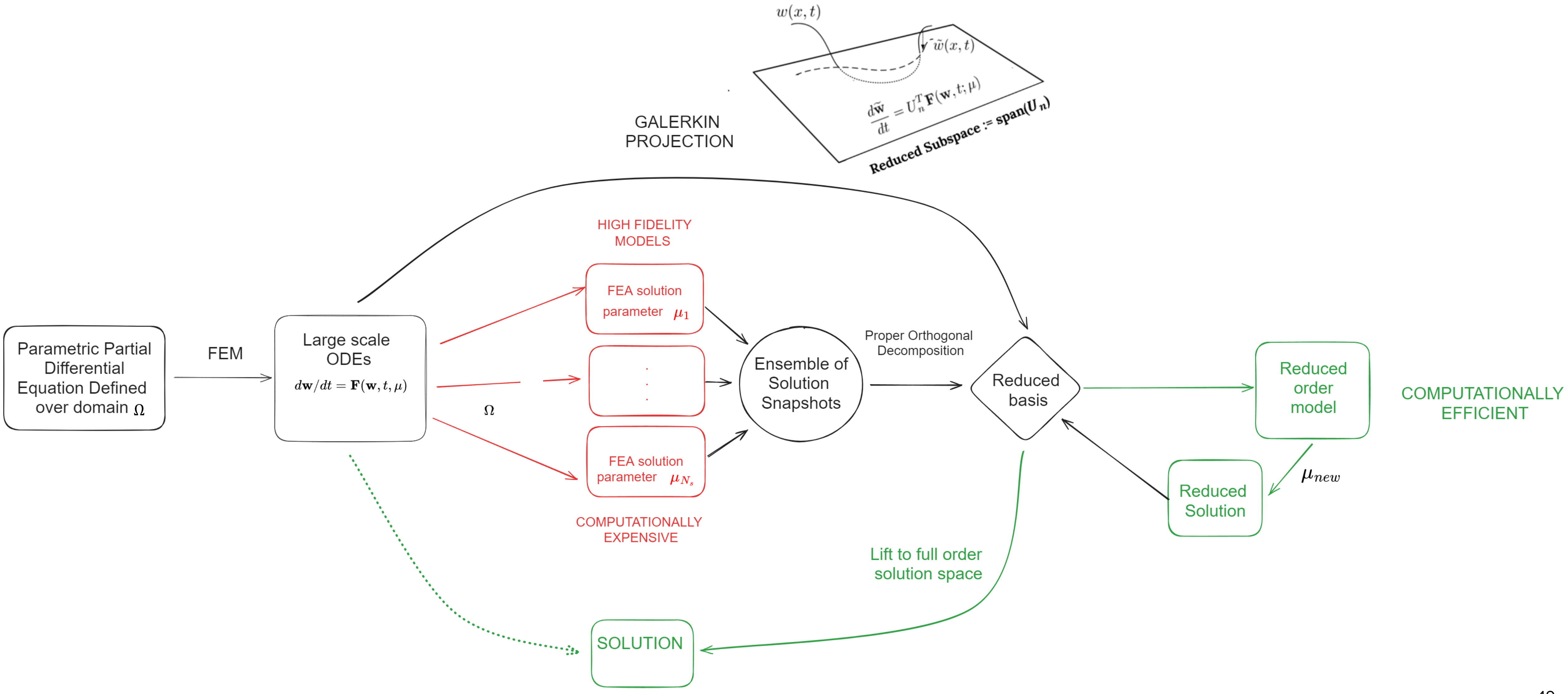
$$\widetilde{\mathsf{U}}^T \mathsf{K} \widetilde{\mathsf{U}} \, \mathbf{T}_n = \widetilde{\mathsf{U}}^T \mathbf{Q}$$

$$\text{ROM:} \quad \mathsf{K}_n \mathbf{T}_n = \mathbf{Q}_n$$

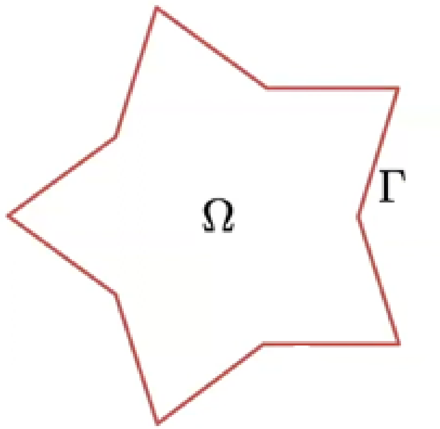
REDUCED SYSTEM
IMPROVED SPEED

$$\boxed{\mathbf{T}^{\mathrm{sol}} = \widetilde{\mathsf{U}} \mathbf{T}_n^{\mathrm{sol}}}$$

Deriving Projection-based MOR: Summary



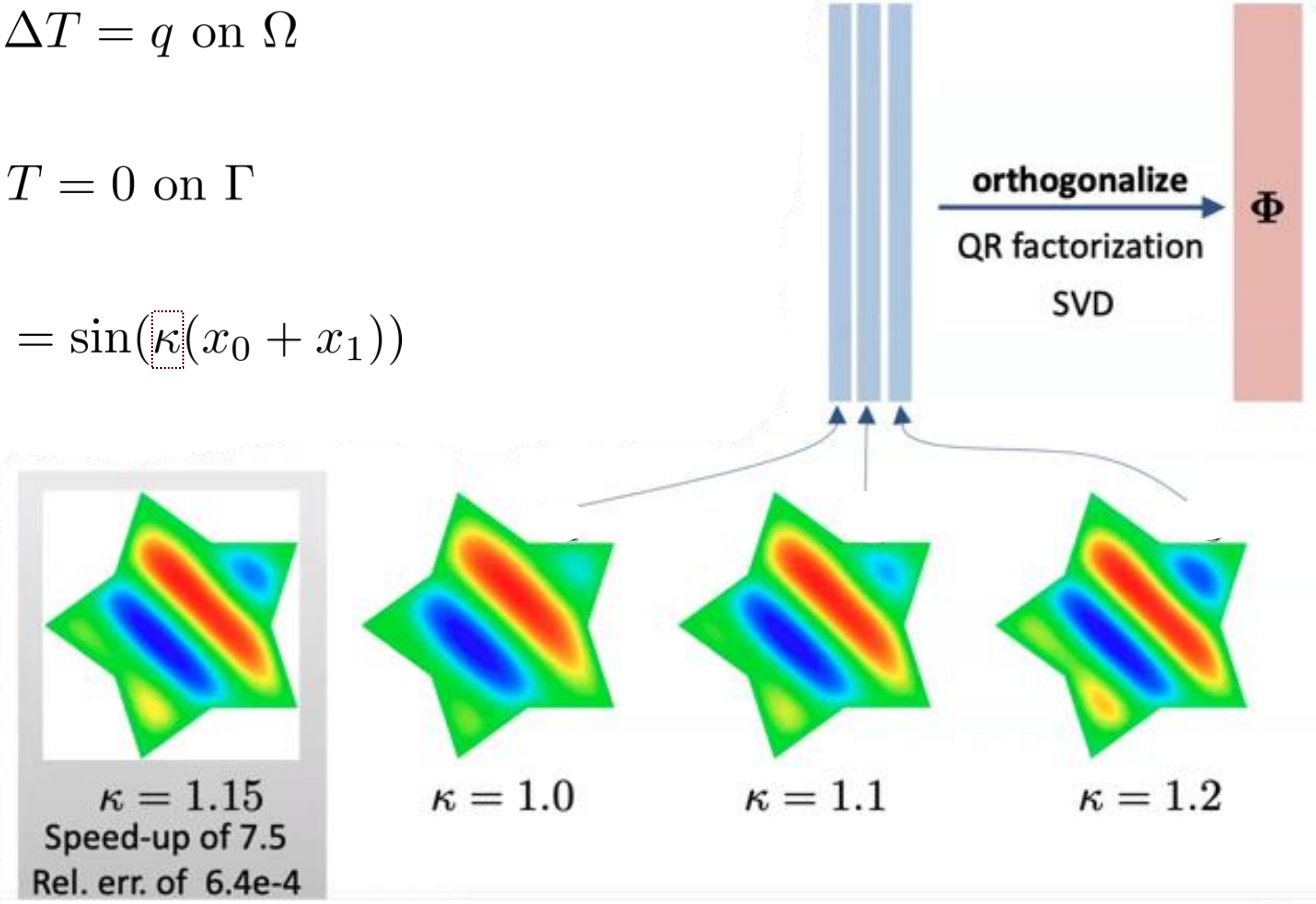
MOR of Parametric Poisson's Equation Using pylibROM



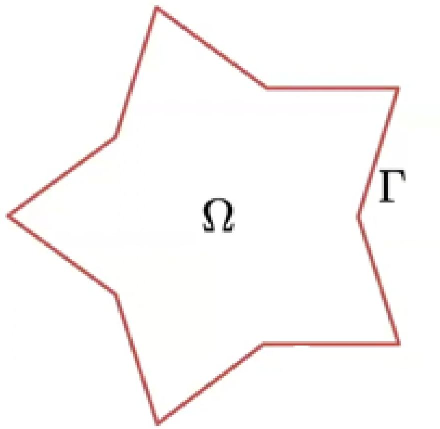
$$-\Delta T = q \text{ on } \Omega$$

$$T = 0 \text{ on } \Gamma$$

$$q(\mathbf{x}, \kappa) = \sin(\kappa(x_0 + x_1))$$



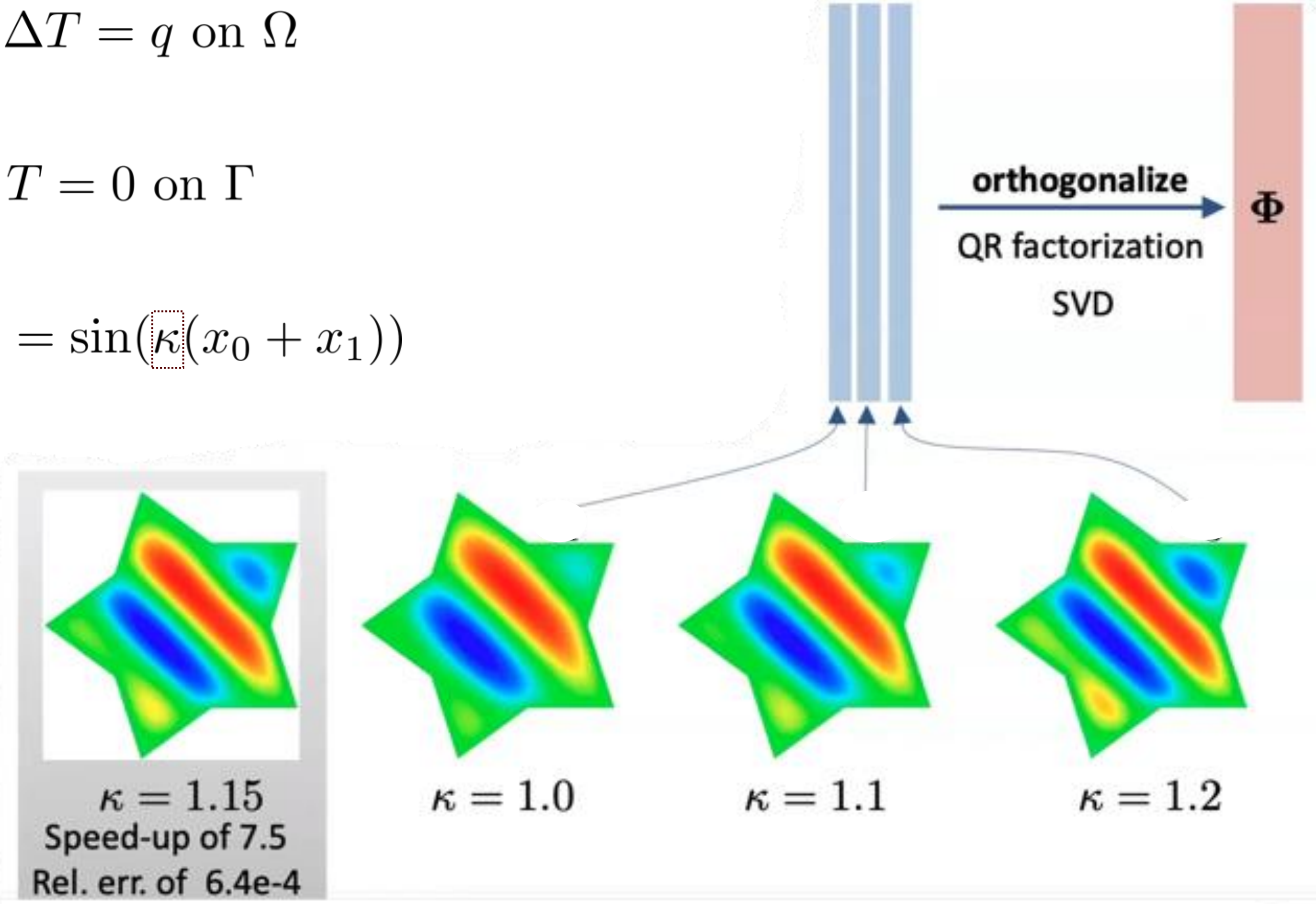
MOR of Parametric Poisson’s Equation Using pylibROM



$$-\Delta T = q \text{ on } \Omega$$

$$T = 0 \text{ on } \Gamma$$

$$q(\mathbf{x}, \kappa) = \sin(\boxed{\kappa}(x_0 + x_1))$$



Let’s head over to Jupyter notebook and see how this problem can be solved with pyMFEM and pylibROM.

Break: 10 mins

Session– 4

Non-intrusive modeling of Dynamical Systems:

Dynamic Mode Decomposition

Suparno Bhattacharyya

Reduced Order Modeling Strategies

INTRUSIVE

- Requires data and/or governing equations (high fidelity model)
- Example: projection-based reduced order model (pROMs)
- Inherits physics of the system, at least partially
- Sessions: 1-3

REDUCED ORDER MODELING

NON-INTRUSIVE (SURROGATE)

- Requires only data
- Example:
  - Koopman operator/DMD
  - Sci-ML: Neural (O/P/S)DEs, Neural Operators
- Physics integrated separately (if needed)

The Model Discovery Problem

- Given the data from a dynamical system can we estimate a model?

- Linear models are often preferred (ease of analyzing, predicting, simulating numerically, estimating, and controlling)

$$\frac{d}{dt}\mathbf{T} = \mathsf{A}_c\mathbf{T}$$

- In a discrete-setting (since we have time-discrete data) we seek for the best-fit operator A satisfying:

$$\mathbf{T}(t_k + \Delta t) = \mathsf{A}\mathbf{T}(t_k)$$

[18]

The Model Discovery Problem

- More generally, we seek a mapping A such that:

$$\mathsf{T}' \approx \mathsf{A}\mathsf{T}$$

where,

A

$$\mathsf{T} = \begin{bmatrix} | & | & | & | & | & | \\ \mathbf{T}(t_0) & \mathbf{T}(t_1) & \cdots & \mathbf{T}(t_k) & \cdots & \mathbf{T}(t_{N_s-2}) \\ | & | & | & | & | & | \end{bmatrix}_{N\times(N_s-1)} \qquad \mathsf{T}' = \begin{bmatrix} | & | & | & | & | & | \\ \mathbf{T}(t_1) & \mathbf{T}(t_2) & \cdots & \mathbf{T}(t_{k+1}) & \cdots & \mathbf{T}(t_{N_s-1}) \\ | & | & | & | & | & | \end{bmatrix}_{N\times(N_s-1)}$$

$$\mathbf{T}(t_k + \Delta t) = \mathsf{A}\mathbf{T}(t_k) = \mathsf{A}^{k+1}\mathbf{T}(t_0) = \Phi\Lambda^{k+1}\Phi^T\mathbf{T}(t_0) \qquad \boxed{\Phi, \Lambda \in \mathbb{R}^{N\times N}}$$

eigen-decomposition

A Simple Pseudo-inverse Yields an Expensive Model

- Assuming uniform sampling in time:

$$\mathsf{T}' \approx \mathsf{A}\mathsf{T}$$

$$\mathsf{A} = \underset{\mathsf{A}}{\operatorname{argmin}} \|\mathsf{T}' - \mathsf{A}\mathsf{T}\|_F = \mathsf{T}'\mathsf{T}^{\dagger}$$

$\dagger :=$ pseudo-inverse

- The *catch* is that A obtained in this way is prohibitively large $\mathbb{R}^{N \times N}$ since N is large.

- Calculating or simulating with such a large A is expensive.

DMD Algorithm: The Fundamental Idea

- The Dynamic Mode Decomposition (DMD) algorithm derives the leading eigen-decomposition of the optimal linear operator A, that links the matrices of time-sequenced snapshots.

- DMD, instead of deriving A, focuses on the projection of A , $\widetilde{\mathsf{A}}$, onto a low-dimensional subspace derived from the snapshot-matrix T, assuming that the dynamics is embedded within that subspace.

- The subspace is obtained by applying SVD on T and selecting n dominant modes.

$$\mathsf{T} = \mathsf{U}\Sigma\mathsf{V} \xrightarrow[n \text{ modes}]{\text{Truncate up to}} \mathsf{T} \approx \widetilde{\mathsf{U}}_{N\times n}\widetilde{\Sigma}_{n\times n}\widetilde{\mathsf{V}}_{n\times N}$$

$$\boxed{\widetilde{\mathsf{A}}_{n\times n} = \widetilde{\mathsf{U}}^T\mathsf{A}\widetilde{\mathsf{U}} = \widetilde{\mathsf{U}}^T\mathsf{T}'\mathsf{T}^\dagger\widetilde{\mathsf{U}} = \widetilde{\mathsf{U}}^T\mathsf{T}'\widetilde{\mathsf{V}}\widetilde{\Sigma}^{-1}}$$

[18]

DMD Algorithm

- The Dynamic Mode Decomposition (DMD) algorithm derives the leading eigen-decomposition of the optimal linear operator A, that links the matrices of time-sequenced snapshots.

- DMD, instead of deriving A, focuses on the projection of A , $\widetilde{\mathsf{A}}$, onto a low-dimensional subspace derived from the snapshot-matrix T, assuming that the dynamics is embedded within that subspace.

- The subspace is obtained by applying SVD on T and selecting n dominant modes.

$$\widetilde{\mathsf{A}}_{n\times n} = \widetilde{\mathsf{U}}^T \mathsf{T}' \widetilde{\mathsf{V}} \widetilde{\Sigma}^{-1}$$

estimate

$$\mathbf{T}(t_k + \Delta t) = \boxed{\widetilde{\Phi}\widetilde{\Lambda}}^{k+1} \widetilde{\Phi}^T \mathbf{T}(t_0)$$

$$\widetilde{\Phi} \in \mathbb{R}^{N\times n}$$
$$\widetilde{\Lambda} \in \mathbb{R}^{n\times n}$$

[18]

DMD Algorithm

- It can be theoretically shown that the leading columns of the eigenvectors Φ corresponding to the leading eigenvalues in Λ (the eigenpair of A), can be calculated using the eigen pair of $\widetilde{\mathrm{A}}$.

- This calculation is computationally much cheaper

$$\widetilde{\mathsf{A}}\mathsf{W} = \widetilde{\Lambda}\mathsf{W}$$

$\widetilde{\Lambda}$ is a subset of Λ.

$\widetilde{\Lambda}, \mathsf{W} \in \mathbb{R}^{n \times n}$

$$\widetilde{\Phi} = \mathsf{T}'\widetilde{\mathsf{V}}\widetilde{\Sigma}^{-1}\mathsf{W}$$

$\widetilde{\Phi}$ is a subset of Φ

$\widetilde{\Phi} \in \mathbb{R}^{N \times n}$

$$\mathbf{T}(t_k + \Delta t) = \widetilde{\Phi}\widetilde{\Lambda}^{k+1}\widetilde{\Phi}^T\mathbf{T}(t_0)$$

Summary

- Useful for large-scale time-dependent (parametric) problems.
- Used when HDM is not available.
- Derives purely data-driven *linear* surrogates.
- Applicable for both linear and nonlinear problems.

DMD with pylibROM

Suparno Bhattacharyya

FE Model for Transient Heat Conduction

$$\rho c_p \frac{\partial T}{\partial t} = \nabla \cdot (k \nabla T)$$

$$\rho c_p \int_\Omega \dot{T} v(\mathbf{x})\, d\Omega = \int_\Omega \nabla \cdot (k\nabla T)\, v(\mathbf{x})\, d\Omega$$

Dirichlet

$$\rho c_p \int_\Omega \dot{T} v\, d\Omega + \int_\Omega k \nabla T \cdot \nabla v\, d\Omega - \int_\Gamma k(\nabla T \cdot \mathbf{n}) v\, d\Gamma = 0$$

Reflective

Assembled FE Model

$$T(\mathbf{x}) = \sum_{i=1}^{N} T_i \boxed{\Psi_i(\mathbf{x})} \longrightarrow \rho c_p \int_\Omega \dot{T} v \, d\Omega + k \nabla T \cdot \nabla v \, d\Omega = 0$$

Shape function

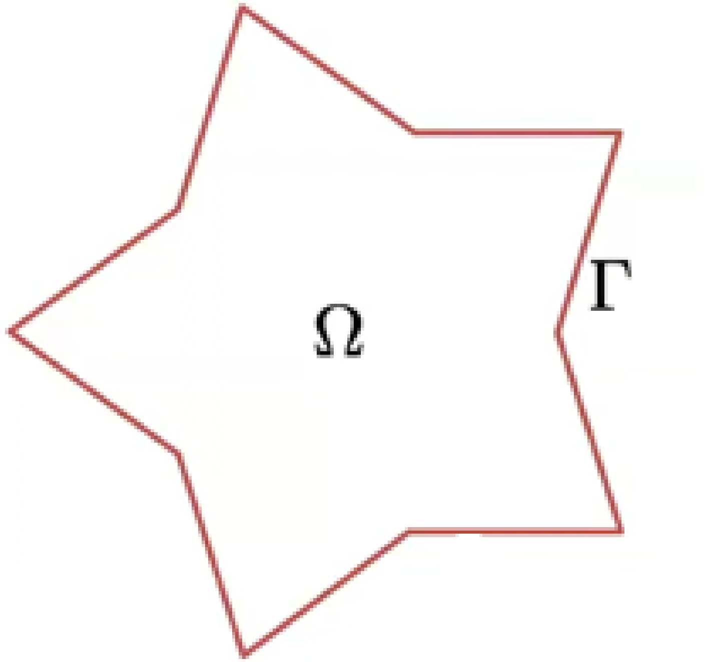
$$\boxed{\mathsf{M}_{N\times N}\dot{\mathbf{T}}_{N\times 1} + \mathsf{K}_{N\times N}\mathbf{T}_{N\times 1} = 0}$$

Solve stiff ODE with implicit scheme

$$M_{ij} = \sum_{e=1}^{n_e} \int_{\Omega_e} \rho c_p \Psi_i^e \Psi_j^e \, d\Omega_e \qquad \mathbf{T}(t=0) = \mathbf{T}_{\text{init}}$$

$$K_{ij} = \sum_{e=1}^{n_e} \int_{\Omega_e} k(\mathbf{x}, \mu) \nabla \Psi_i^e \cdot \nabla \Psi_j^e \, d\Omega_e$$

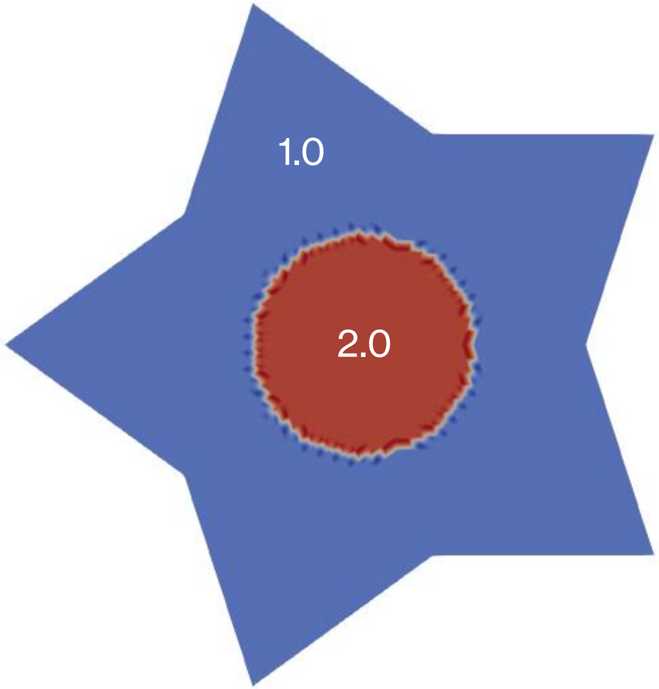
DMD for Transient Linear Heat Conduction Using pylibROM



$$\frac{\partial T}{\partial t} = \kappa \Delta T \text{ on } \Omega$$

$$\nabla T \cdot \mathbf{n} = 0 \text{ on } \Gamma$$

$$\kappa = 1.0$$



$$\mathbf{T}(t = 0) = \mathbf{T}_{\text{init}}$$

Online resources

- Steve Brunton – YouTube
- Nathan Kutz – YouTube
- Data-driven Physical Simulations (DDPS) Seminar Series – YouTube
- ETH Zürich DLSC: Course Introduction (youtube.com)
- https://www.youtube.com/@PhysicsInformedMachineLearning
- ThatMathThing – YouTube
- StatQuest: Principal Component Analysis (PCA), Step-by-Step (youtube.com)
- MFEM - Finite Element Discretization Library

References

[1] Farhat, C. (n.d.). AA216/CME345: Model Reduction - Introduction. Stanford University. Retrieved from https://web.stanford.edu/group/frg/course_work/CME345/CA-CME345-Ch1.pdf

[2]"Rais-Rohani: Reduced Order & Surrogate Modeling Research - Mechanical Engineering - University of Maine," Mechanical Engineering, May 24, 2017. https://umaine.edu/mecheng/rais-rohani-research-rom/ (accessed Apr. 15, 2024).

[3] S. Fresca, Giorgio Gobat, P. Fedeli, Attilio Frangi, and A. Manzoni, "Deep learning-based reduced order models for the real-time simulation of the nonlinear dynamics of microstructures," International journal for numerical methods in engineering, vol. 123, no. 20, pp. 4749–4777, Jun. 2022, doi: https://doi.org/10.1002/nme.7054.

[4] A. Samnioti and Vassilis Gaganis, "Applications of Machine Learning in Subsurface Reservoir Simulation—A Review—Part II," Energies, vol. 16, no. 18, pp. 6727–6727, Sep. 2023, doi: https://doi.org/10.3390/en16186727.

[5] S. Fresca, A. Manzoni, Luca Dedè, and Alfio Quarteroni, "POD-Enhanced Deep Learning-Based Reduced Order Models for the Real-Time Simulation of Cardiac Electrophysiology in the Left Atrium," Frontiers in physiology, vol. 12, Sep. 2021, doi: https://doi.org/10.3389/fphys.2021.679076.

[6] Willcox, K. E. (2021, November 15). Predictive Digital Twins: From physics-based modeling to scientific machine learning [Video]. YouTube. https://www.youtube.com/watch?v=ZuSx0pYAZ_I

References

[7] Dylan Matthew Copeland, Siu Wun Cheung, K. Huynh, and Y. Choi, “Reduced order models for Lagrangian hydrodynamics,” Computer methods in applied mechanics and engineering, vol. 388, pp. 114259–114259, Jan. 2022, doi: https://doi.org/10.1016/j.cma.2021.114259.

[8] Siu Wun Cheung, Y. Choi, Dylan Matthew Copeland, and K. Huynh, “Local Lagrangian reduced-order modeling for the Rayleigh-Taylor instability by solution manifold decomposition,” Journal of computational physics (Print), vol. 472, pp. 111655–111655, Jan. 2023, doi: https://doi.org/10.1016/j.jcp.2022.111655.

[9] “US11514210B2 - Component-wise reduced-order model design optimization such as for lattice design optimization - Google Patents,” Google.com, Dec. 10, 2019. https://patents.google.com/patent/US11514210B2/en (accessed Apr. 15, 2024).

[10] Y. Kim, Karen May Wang, and Y. Choi, “Efficient Space–Time Reduced Order Model for Linear Dynamical Systems in Python Using Less than 120 Lines of Code,” Mathematics, vol. 9, no. 14, pp. 1690–1690, Jul. 2021, doi: https://doi.org/10.3390/math9141690.

[11] Y. Kim, Y. Choi, D. Widemann, and T. Zohdi, “Efficient nonlinear manifold reduced order model,” arXiv.org, 2020. https://arxiv.org/abs/2011.07727 (accessed Apr. 15, 2024).

[12] E. B. Chin, "Dynamic Mode Decomposition for Fast, Predictive Additive Manufacturing Simulations," presented at the US National Congress on Computational Mechanics, Albuquerque, NM, 2023, LLNL-PRES-852033.

References

[13] S. W. Cheung, Y. Choi, S. H. Keo, and T. Kadeethum, “Data-scarce surrogate modeling of shock-induced pore collapse process,” arXiv.org, 2023. https://arxiv.org/abs/2306.00184 (accessed Apr. 15, 2024).

[14] P. Benner and et al, System- and Data-Driven Methods and Algorithms. Walter de Gruyter GmbH & Co KG, 2021. Accessed: Apr. 15, 2024. [Online]. Available: https://www.degruyter.com/document/isbn/9783110606133/html

[15] Li, M., Jain, S. & Haller, G. Model reduction for constrained mechanical systems via spectral submanifolds. Nonlinear Dyn 111, 8881–8911 (2023). https://doi.org/10.1007/s11071-023-08300-5

[16] P. Benner, S. Grivet-Talocia, A. Quarteroni, G. Rozza, W. Schilders, and L. M. Silveira, Eds., “System- and Data-Driven Methods and Algorithms.” De Gruyter, Oct. 08, 2021. doi: 10.1515/9783110498967.

[17] “Snapshot-Based Methods and Algorithms.” De Gruyter, Jan. 01, 2021. doi: 10.1515/9783110671490.

[18] Brunton, S. L., & Kutz, J. N. (2019). Data-Driven Science and Engineering. Cambridge University Press. https://doi.org/10.1017/9781108380690

[19] S. Bhattacharyya and J. P. Cusumano, “An Energy Closure Criterion for Model Reduction of a Kicked Euler–Bernoulli Beam,” Journal of vibration and acoustics, vol. 143, no. 4, Nov. 2020, doi: https://doi.org/10.1115/1.4048663.

References

[20] Y. Yang, M. Ghasemi, E. Gildin, Yalchin Efendiev, and V. Calo, “Fast Multiscale Reservoir Simulations With POD-DEIM Model Reduction,” SPE journal, vol. 21, no. 06, pp. 2141–2154, Jun. 2016, doi: https://doi.org/10.2118/173271-pa.

[21] Huhn, Q. A., Tano, M. E., & Ragusa, J. C. (2023). Physics-Informed Neural Network with Fourier Features for Radiation Transport in Heterogeneous Media. In Nuclear Science and Engineering (Vol. 197, Issue 9, pp. 2484–2497). Informa UK Limited. https://doi.org/10.1080/00295639.2023.2184194

[22] “PETE 689 / NUEN 689 (Fall 2023) – Digital Twin Lab,” Tamu.edu, 2023. https://dtl.tamids.tamu.edu/pete-689-nuen-689/ (accessed Apr. 15, 2024).

[23] Reddy, J. N. 2019. Introduction to the Finite Element Method. 4th ed. New York: McGraw-Hill Education.

[24] Holmes P, Lumley JL, Berkooz G, Rowley CW. 2012. Turbulence, Coherent Structures, Dynamical Systems and Symmetry. 2nd ed. Cambridge University Press.

[25] Antoulas, Athanasios C. Approximation of large-scale dynamical systems. Society for Industrial and Applied Mathematics, 2005.