

Hyper-reduced nonlinear manifold reduced order model

Physics-constrained learning III

ML4I

August 12, 2021



Youngsoo Choi



Awesome team



Dylan Copeland



Kevin Huynh



Tony Cheung



Youngkyu Kim



David Widemann

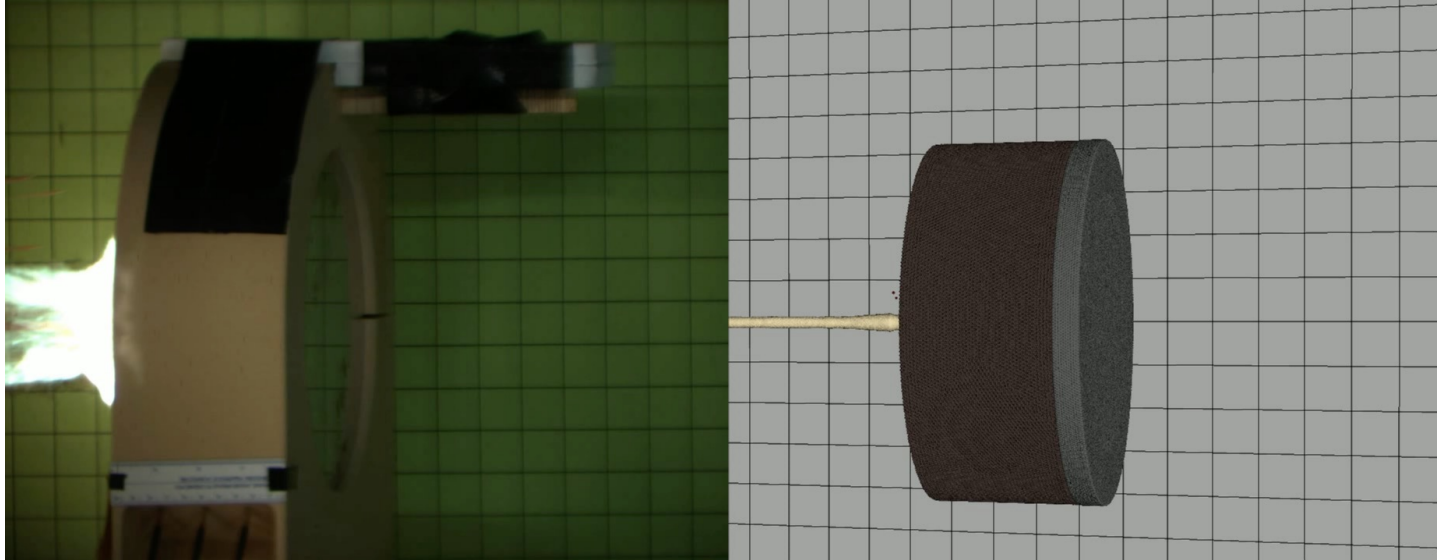


Tarek Zohdi

Smooth Particle Hydrodynamic (SPH) modeling for jet [courtesy: Karen Wang]

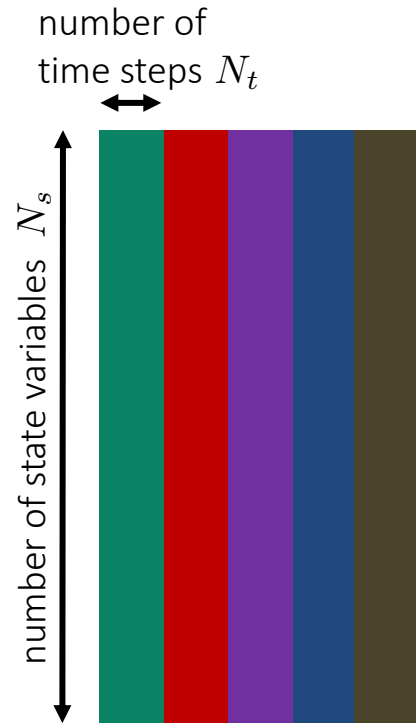
One forward simulation:

- 10.1 million particles
- 1,440 processors
- 178 hours (7.4 days)



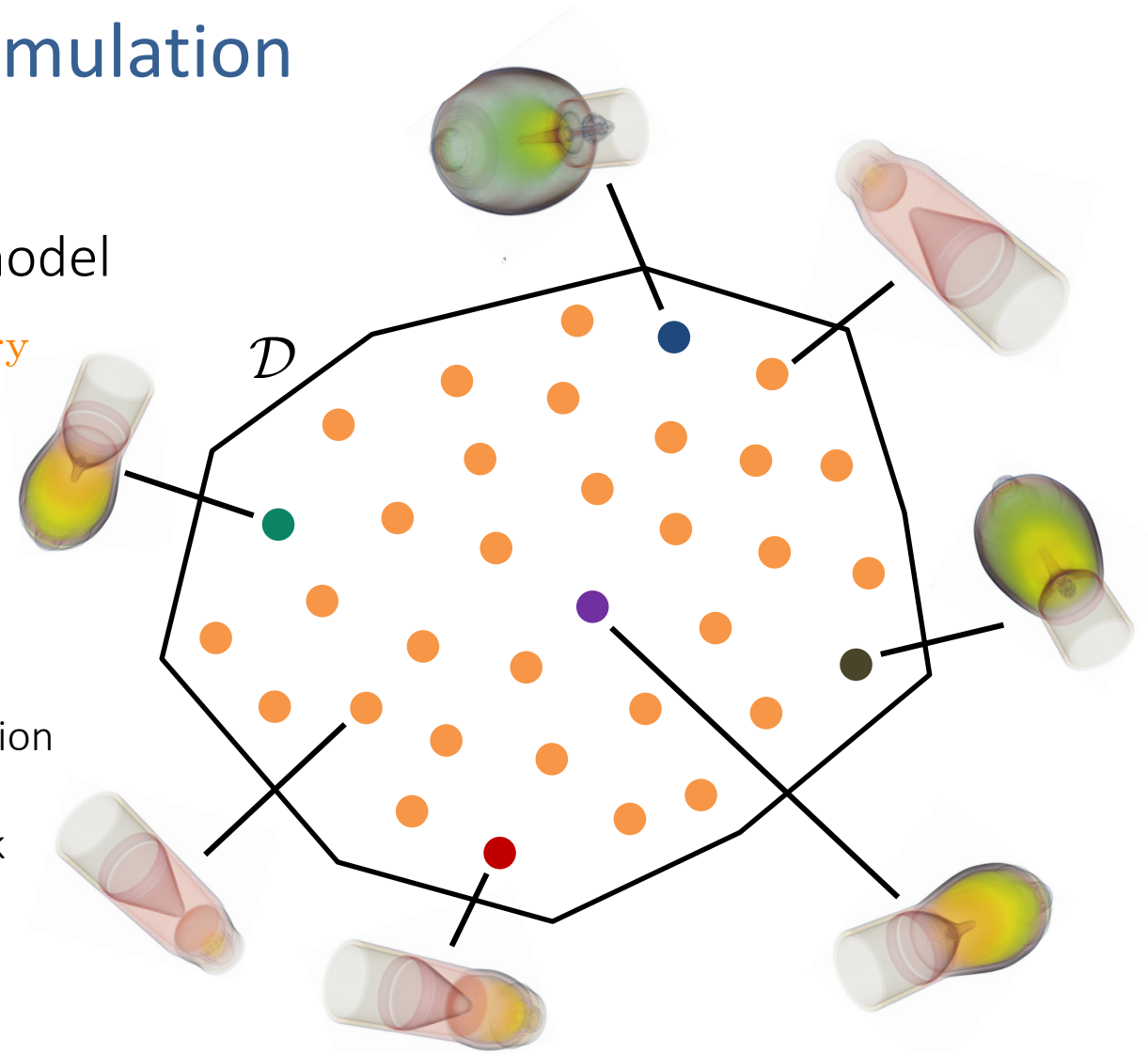
Approach: Data-driven physical simulation

1. *Collect data*: solve FOM for $\mu \in \mathcal{D}_{\text{sample}}$
2. *Machine learning*: build a reduced order model
3. *Accelerate* physical simulation, $\mu \in \mathcal{D}_{\text{query}}$

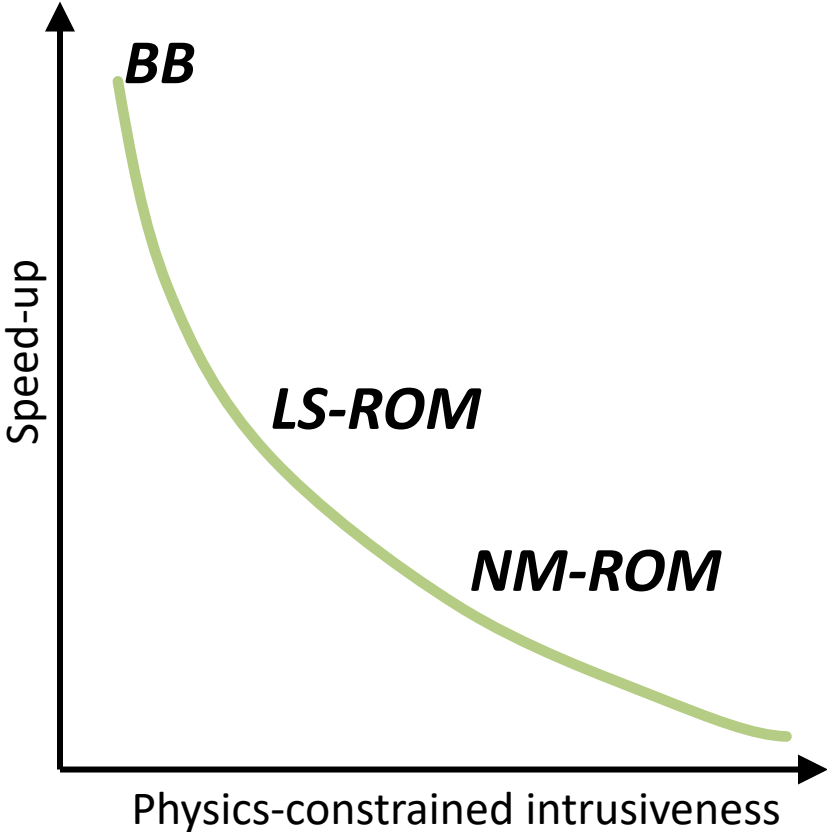
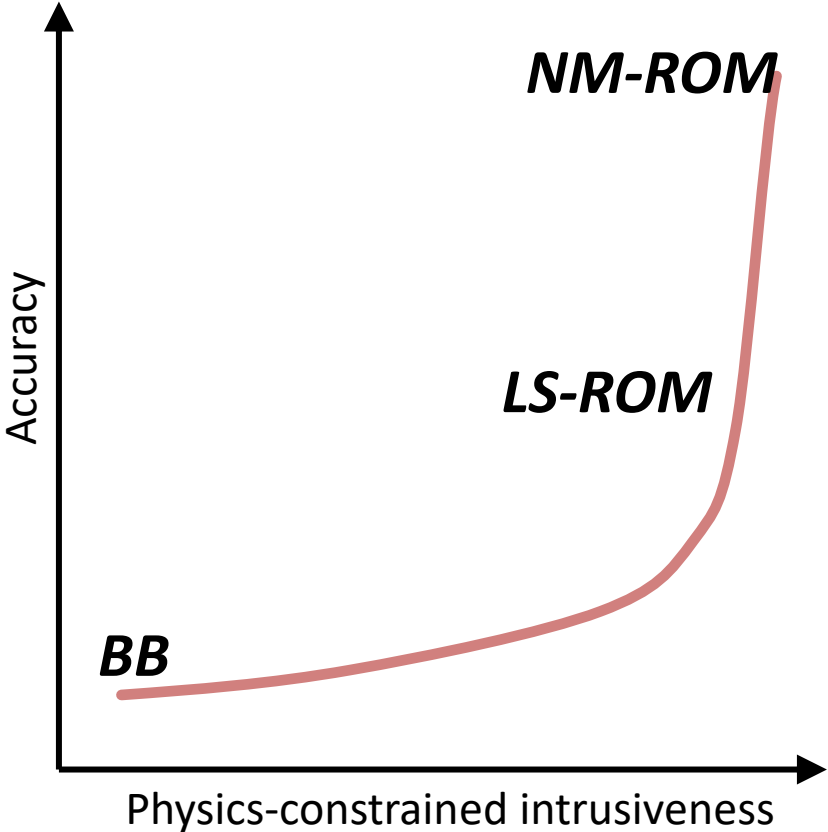


- Principal component analysis
- Tucker decomposition
- Canonical Polyadic decomposition
- Autoencoder
- Generative adversarial network
- Gaussian processes

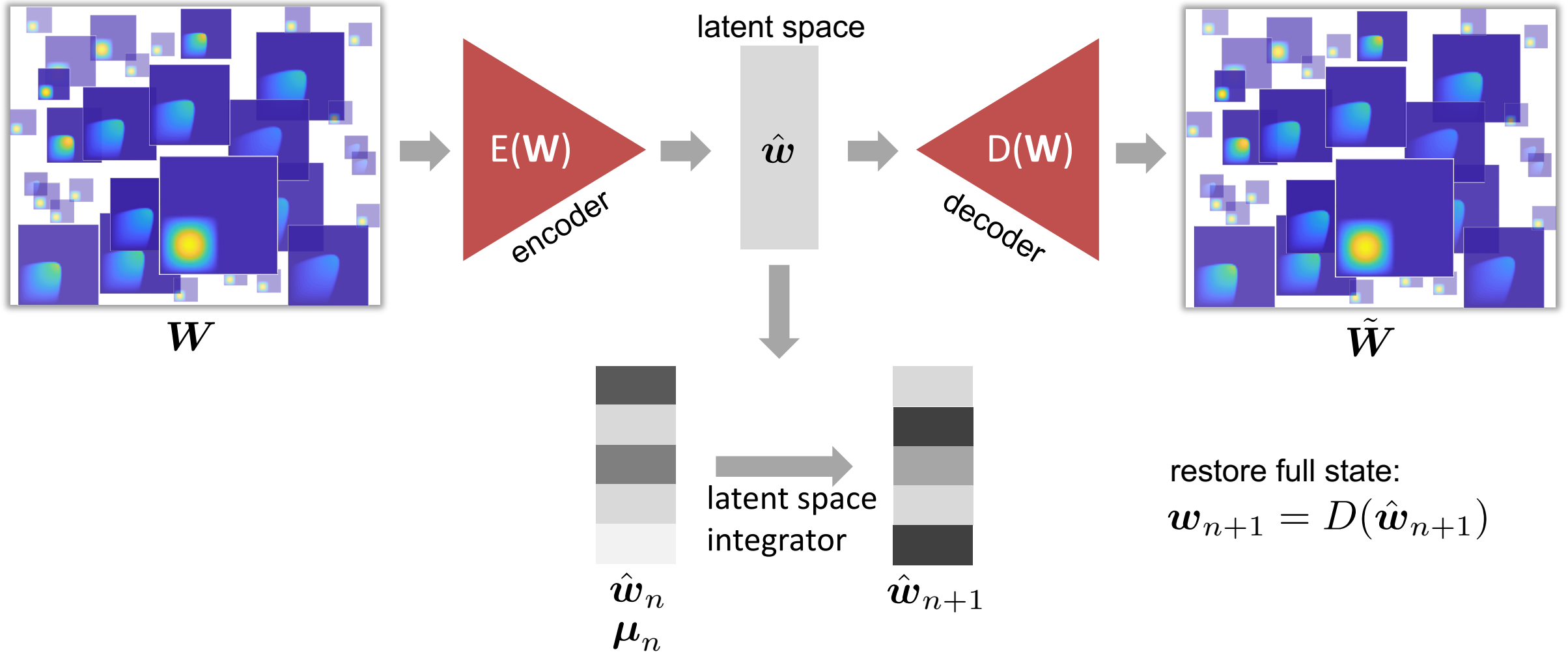
Intrusive vs. non-intrusive



Category of reduced order models via level of intrusiveness



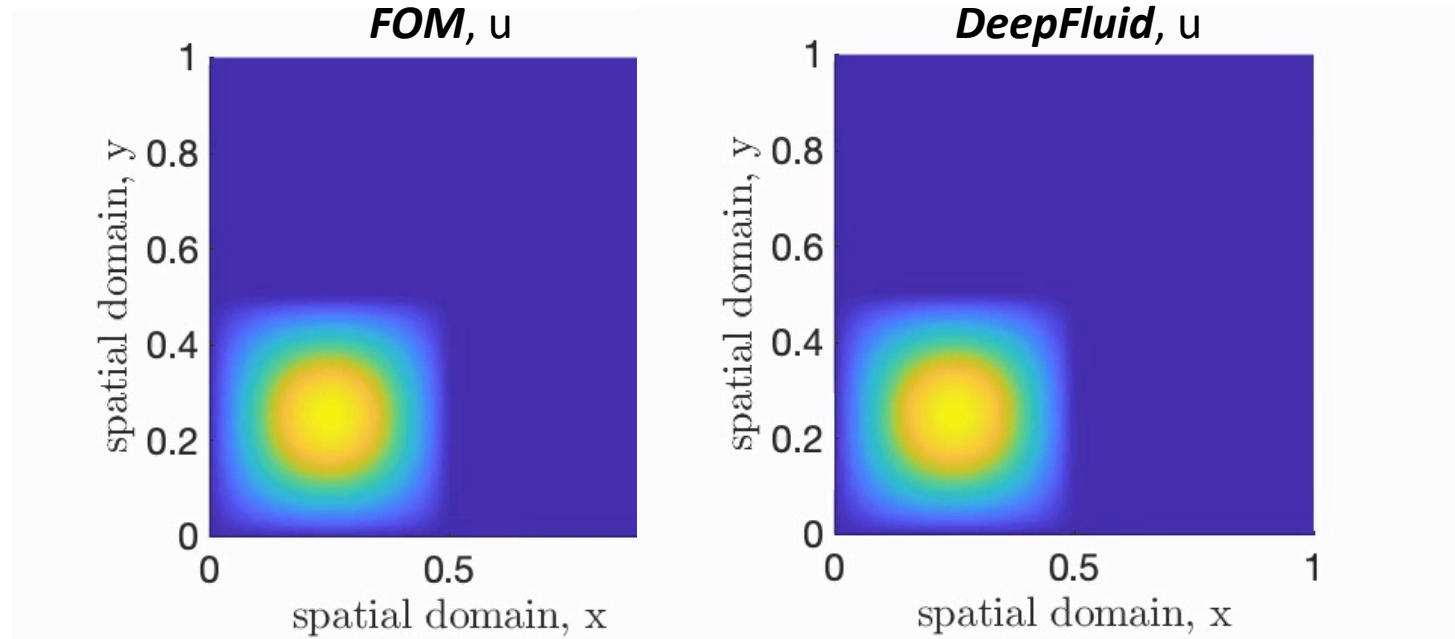
Black box approach, Deep Fluid: Nonlinear manifold learning*



*Kim, Byungsoo, et al. "Deep fluids: A generative network for parameterized fluid simulations." *Computer Graphics Forum*. Vol. 38. No. 2. 2019.

Results: Burgers' equation* [with Youngkyu Kim, David Widemann, Tarek Zohdi]

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} = \nu \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) \quad \frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} = \nu \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right) \quad Re = 1/\nu = 10,000$$



method	DeepFluid
max. rel. error (%)	38.6
speed-up	119

*Kim, Choi, Widemann, and Zohdi, "Efficient nonlinear manifold reduced order model." *Workshop on machine learning for engineering modeling, simulation and design @ NeurIPS 2020*

Projection-based linear subspace reduced order model

- Governing equation: $\frac{d\mathbf{w}}{dt} = \mathbf{f}(\mathbf{w}, t; \boldsymbol{\mu}), \quad \mathbf{w}, \mathbf{f} \in \mathbb{R}^{N_s}$
- Solution approximation:

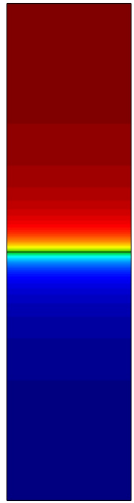
$$\mathbf{w} \approx \tilde{\mathbf{w}} = \mathbf{w}_{\text{ref}} + \boldsymbol{\Phi} \hat{\mathbf{w}}, \quad \boldsymbol{\Phi} \in \mathbb{R}^{N_s \times n_s}, \quad n_s \ll N_s$$

- Reduced system after Galerkin projection: $\frac{d\hat{\mathbf{w}}}{dt} = \boldsymbol{\Phi}^T \mathbf{f}(\mathbf{w}_{\text{ref}} + \boldsymbol{\Phi} \hat{\mathbf{w}}, t; \boldsymbol{\mu})$
- Backward Euler time integrator: $\hat{\mathbf{w}}_n = \hat{\mathbf{w}}_{n-1} + \Delta t \underbrace{\boldsymbol{\Phi}^T \mathbf{f}}_{\text{Scales with FOM size: } \mathbb{R}^{N_s}}(\mathbf{w}_{\text{ref}} + \boldsymbol{\Phi} \hat{\mathbf{w}}, t; \boldsymbol{\mu})$



Scales with FOM size: \mathbb{R}^{N_s}

Successful applications of linear subspace ROM



Rayleigh–Taylor

Kinematic dofs: **8,514**

Energy dofs: 4,096

Wall-clock time: 127 sec

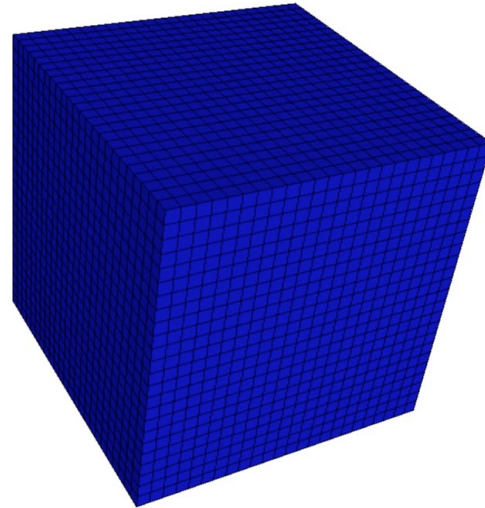
Relative error & speedup

Position: **$5.3e-5$**

Velocity: **$7.8e-3$**

Energy : **$243e-5$**

Speedup: **14.6**



Sedov blast

Kinematic dofs: **14,739**

Energy dofs: 4,096

Wall-clock time: 191 sec

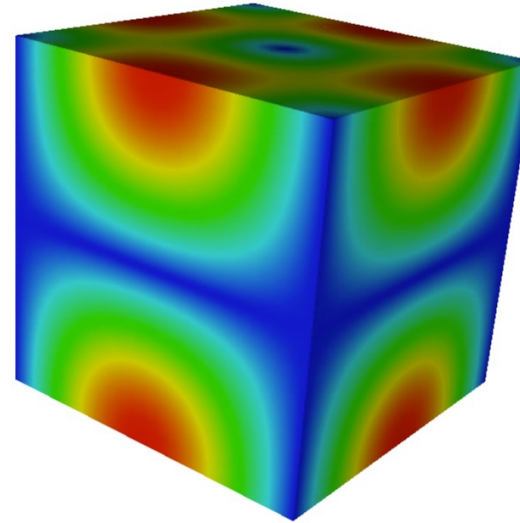
Relative error & speedup

Position: **$2.2e-5$**

Velocity: **$2.2e-4$**

Energy : **$2.3e-4$**

Speedup: **22.8**



Taylor–Green

Kinematic dofs: **14,739**

Energy dofs: 4,096

Wall-clock time: 170 sec

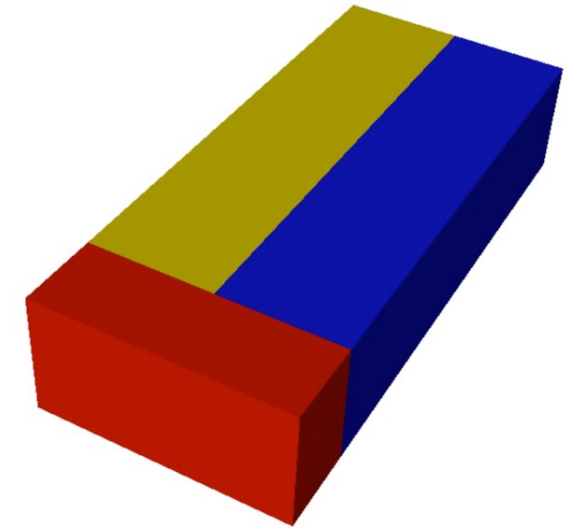
Relative error & speedup

Position: **$1.8e-8$**

Velocity: **$1.1e-6$**

Energy : **$1.0e-7$**

Speedup: **31.2**



Triple-point problem

Kinematic dofs: **38,475**

Energy dofs: 10,752

Wall-clock time: 122 sec

Relative error & speedup

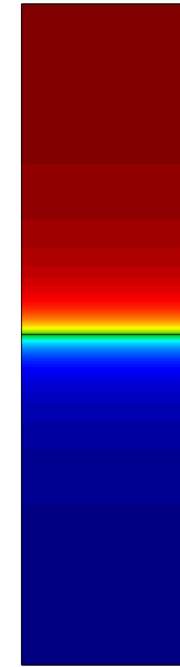
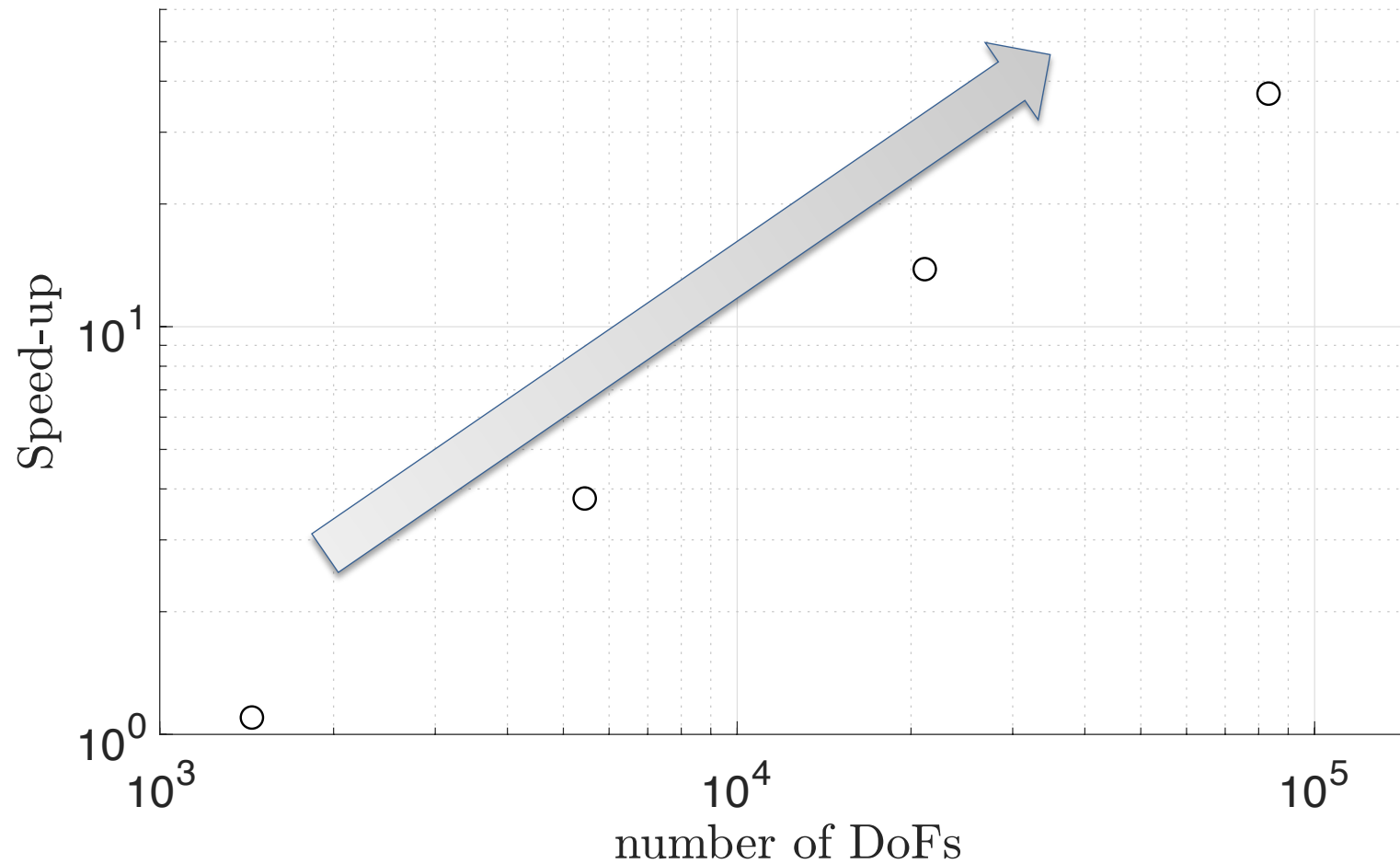
Position: **$3.1e-5$**

Velocity: **$8.1e-4$**

Energy : **$2.8e-4$**

Speedup: **87.8**

Speedup increases as problem size increases

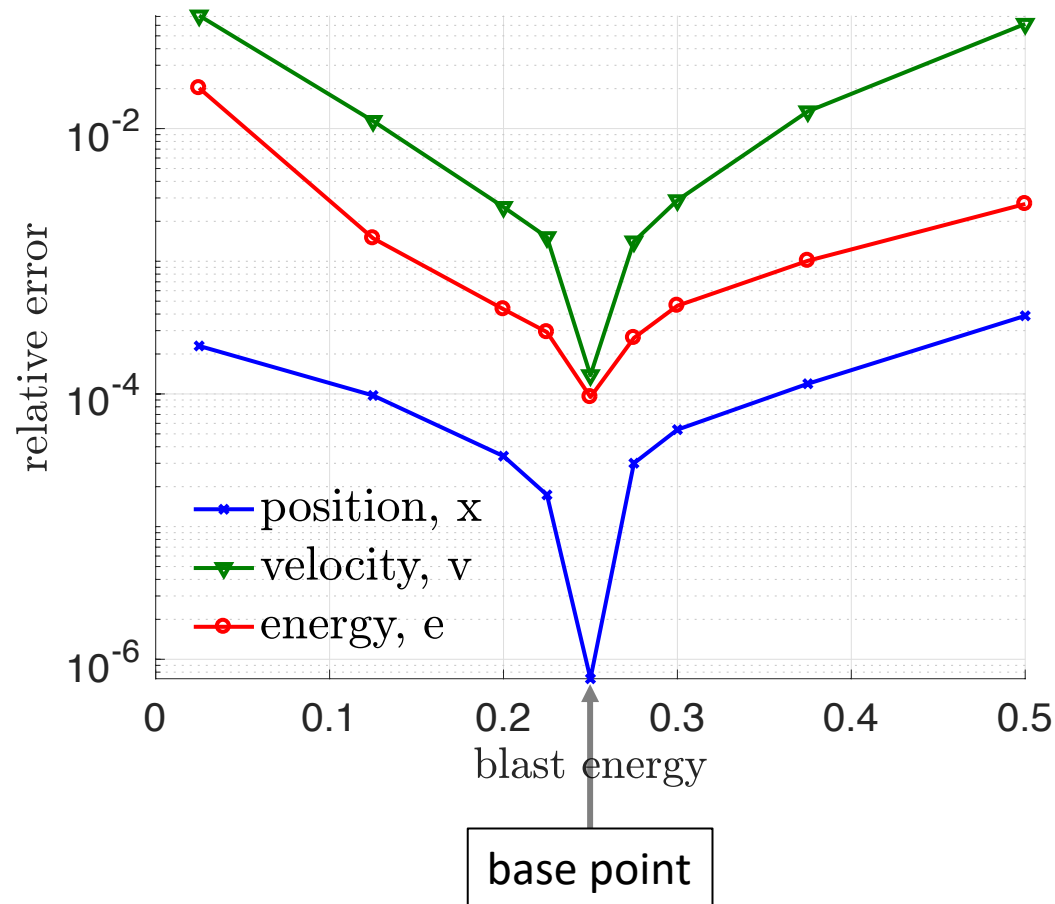


Kinematic dofs: 594
Energy dofs: 256



Kinematic dofs: 33,410
Energy dofs: 16,384

Robustness of a local ROM in extrapolation: Sedov blast



(Blast energy variation)

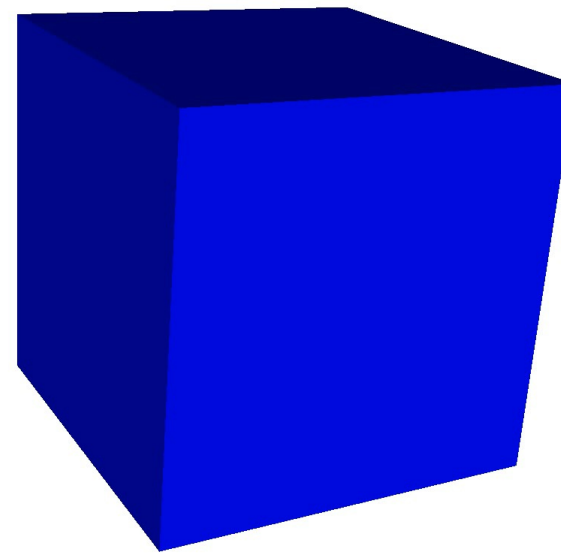
- + Allows a fast gradient computation!
- + Easy to increase the size of the parameter space

Greedy algorithm

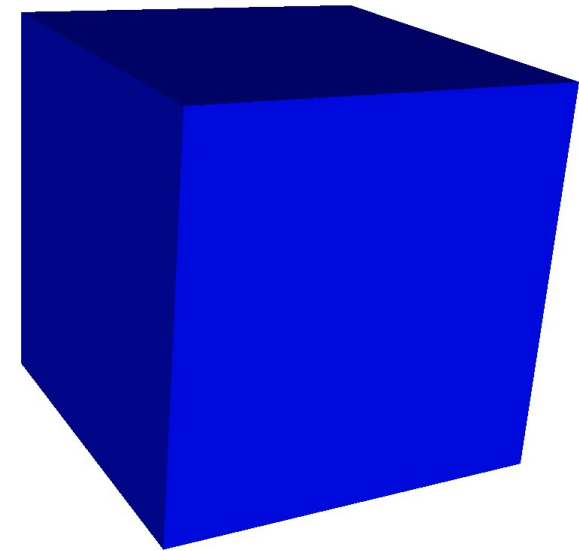
Goal: find an optimal set of local ROMs whose overall accuracy is less than 0.03.

Set parameter space: Blast energy [0.075, 1.25]

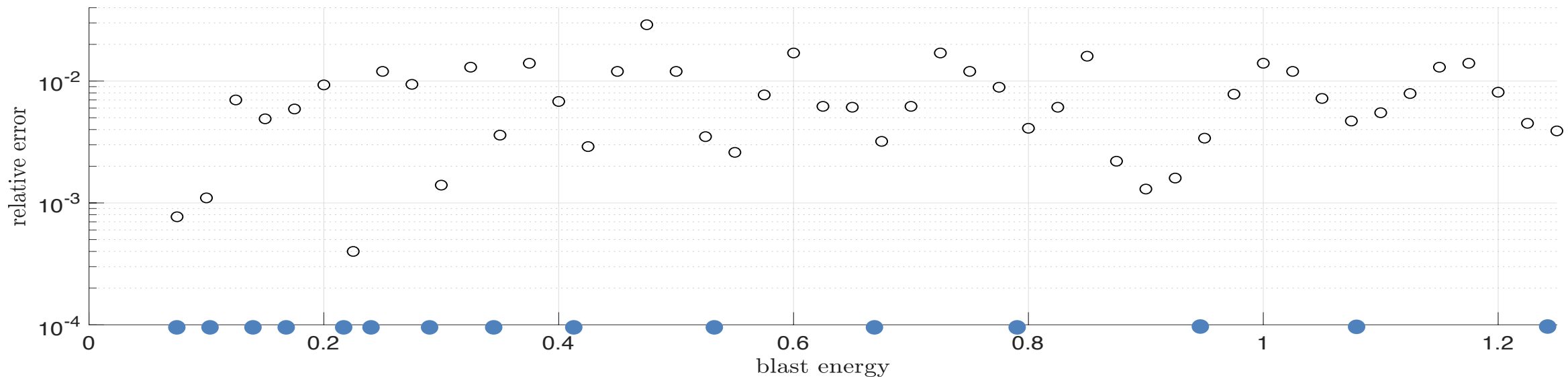
Error indicator: cheap to compute but a good indicator for error measure



Blast energy: 0.075




Blast energy: 1.25



Linear subspace ROM for hydrodynamics

- **Paper:** Copeland, Cheung, Huynh, Choi, “Reduced order models for Lagrangian hydrodynamics” *arXiv preprint*, arXiv:2014.11404, 2021.
- **Software:**
 - libROM (library for reduced order models): <https://github.com/LLNL/libROM>
 - Laghos (physics solver for hydrodynamics): <https://github.com/CEED/Laghos/tree/rom>
- **Webpage** for libROM under construction
- libROM YouTube tutorial under production
 - <https://youtu.be/YaZPtIbGay4>



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. libROM is open source, so anyone is welcome to contribute to the development!

Features

- Efficient data collection
- Proper Orthogonal Decomposition, POD
- Greedy algorithm
- Hyper-reduction
- ...and many more.

libROM is used in many projects, including MFEM, BLAST, ARDRA, and SU2. See also our [Gallery](#), [Publications](#) and [News](#) pages.

News

- July 19, 2021 [spacetimeROM-Python](#) paper is published in Mathematics
- Apr 23, 2021 [LaghosROM](#) arXiv preprint is available
- Jan 1, 2021 [spacetimeROM](#) paper is published in Journal of Computational Physics

libROM tutorials in YouTube

July 22, 2021 [Poisson equation & its finite element discretization](#)

Latest Release

[Examples](#) | [Code documentation](#) | [Sources](#)

[Download libROM.tgz](#)

[Older releases](#) | [Python wrapper](#)

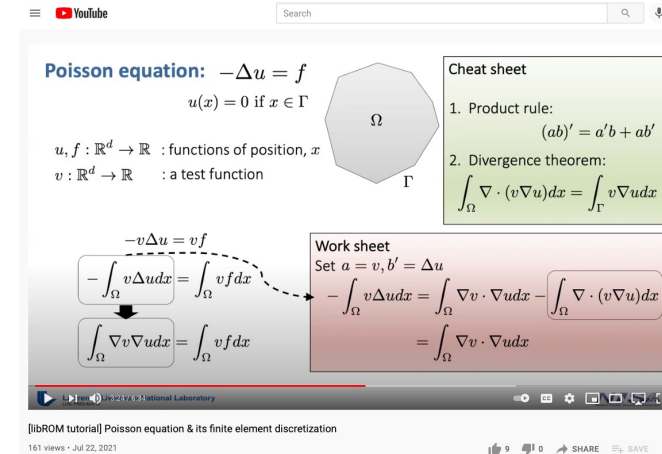
Documentation

[Building libROM](#) | [Poisson equation](#) | [Code Overview](#)

New users should start by examining the [example codes](#) and [tutorials](#). We also recommend using [GLVis](#) for visualization.

Contact

Use the GitHub [issue tracker](#) to report [bugs](#) or post [questions](#) or [comments](#). See the [About](#) page for citation information.



Poisson equation: $-\Delta u = f$
 $u(x) = 0$ if $x \in \Gamma$

$u, f : \mathbb{R}^d \rightarrow \mathbb{R}$: functions of position, x
 $v : \mathbb{R}^d \rightarrow \mathbb{R}$: a test function

Cheat sheet

1. Product rule:
 $(ab)' = a'b + ab'$
2. Divergence theorem:
 $\int_{\Omega} \nabla \cdot (v \nabla u) dx = \int_{\Gamma} v \nabla u dx$

Work sheet

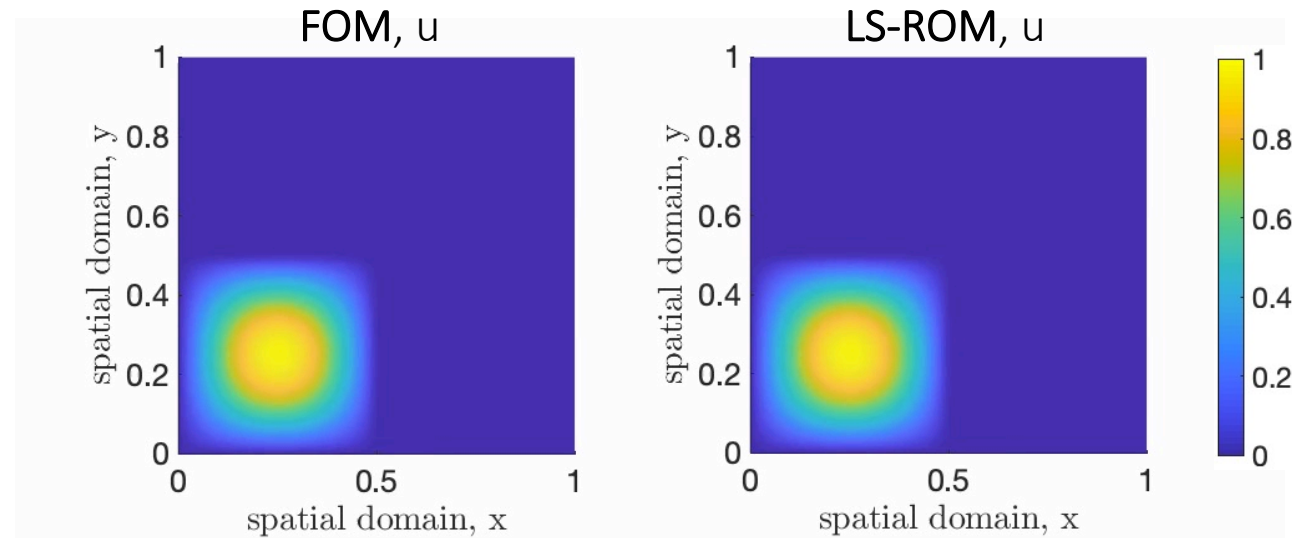
Set $a = v, b' = \Delta u$

$$-\int_{\Omega} v \Delta u dx = \int_{\Omega} \nabla v \cdot \nabla u dx - \int_{\Gamma} v \cdot \nabla u dx$$
$$= \int_{\Omega} \nabla v \cdot \nabla u dx$$

161 views · Jul 22, 2021

Numerical result: 2D viscous Burgers equation (advection-dominated)

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} = \nu \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) \quad \frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} = \nu \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right) \quad Re = 1/\nu = 10,000$$



method	LS-ROM
max. rel. error (%)	34.4
speed-up	26.8

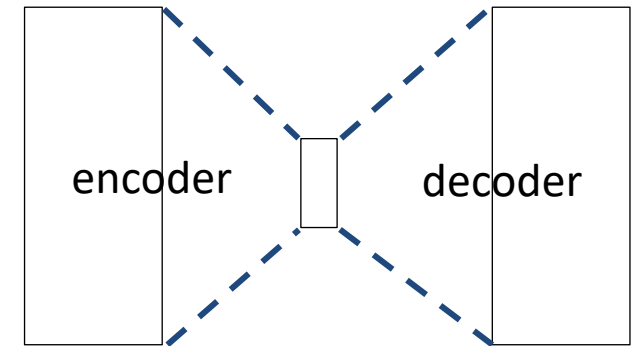
Nonlinear manifold reduced order models

Goal: exploit data to build nonlinear manifold solution representation that achieves ***much better accuracy and robustness*** than linear subspace-based reduced order model

- Governing equation: $\frac{d\mathbf{w}}{dt} = \mathbf{f}(\mathbf{w}, t; \boldsymbol{\mu}), \quad \mathbf{w}, \mathbf{f} \in \mathbb{R}^{N_s}$
- Solution approximation:

$$\mathbf{w} \approx \tilde{\mathbf{w}} = \mathbf{w}_{\text{ref}} + \mathbf{g}(\hat{\mathbf{w}}), \quad \hat{\mathbf{w}} \in \mathbb{R}^{n_s}, \quad n_s \ll N_s$$

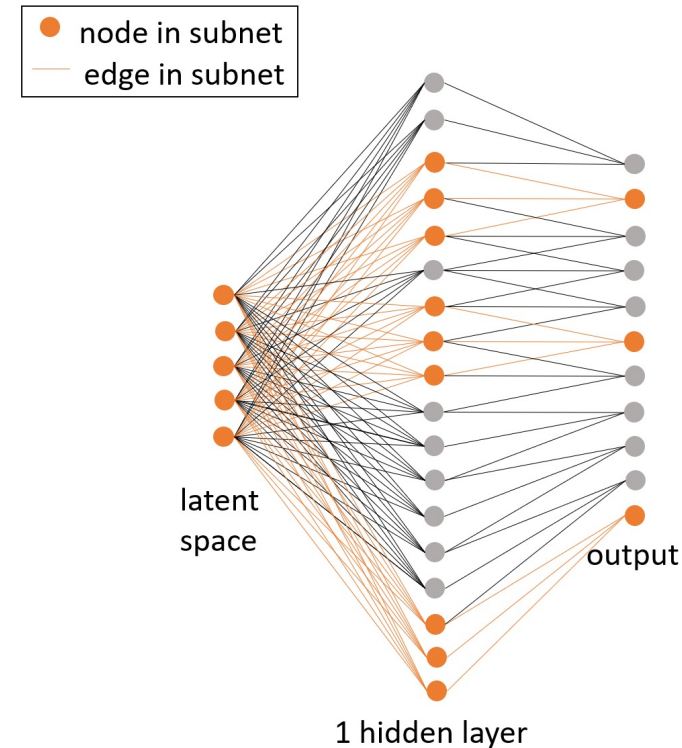
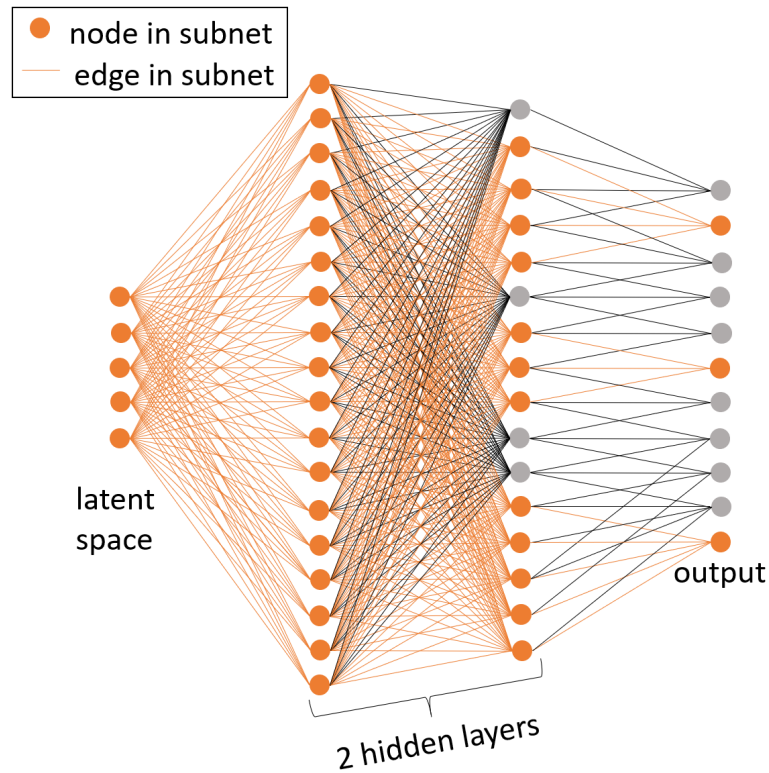
$$\begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix} + \mathbf{g}\left(\begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix}\right)$$



where $\mathbf{g} : \mathbb{R}^{n_s} \rightarrow \mathbb{R}^{N_s}$ defines a nonlinear manifold from reduced to full state

- The over-determined system: $\mathbf{J}_g(\hat{\mathbf{w}}) \frac{d\hat{\mathbf{w}}}{dt} = \mathbf{f}(\mathbf{w}_{\text{ref}} + \mathbf{g}(\hat{\mathbf{w}}), t; \boldsymbol{\mu})$ Hyper-reduction

Shallow vs. deep NN in the perspective of hyper-reduction



- A shallow neural network can provide sparser structure than a deep one in a subnet
- A sparse network is a key for successful NM-ROM hyper-reduction!

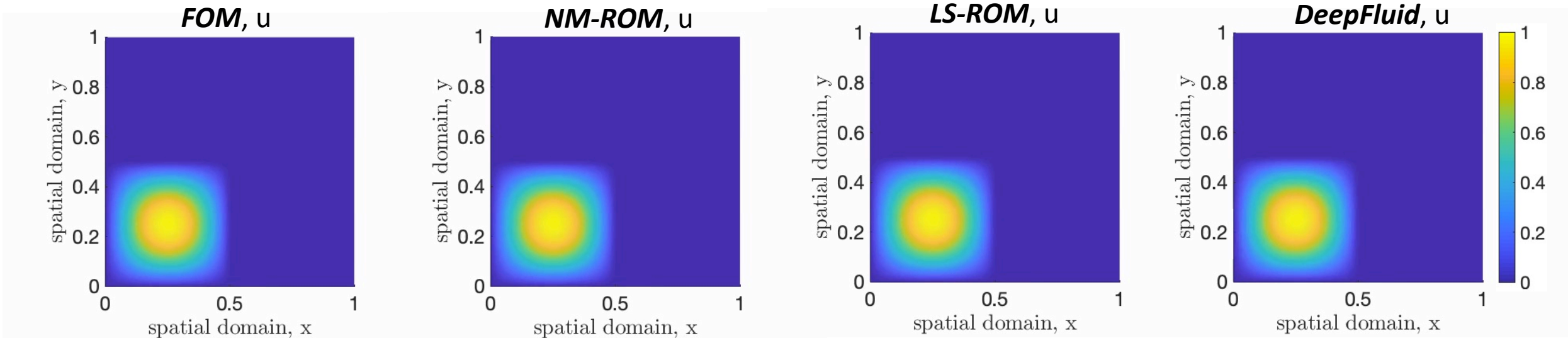
*Kim, Choi, Widemann, and Zohdi, "A fast and accurate physics-informed neural network reduced order model with shallow masked autoencoder." *arXiv preprint, arXiv:2009.11990, 2020.*

Result: 2D viscous Burgers equation (advection-dominated)*

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} = \nu \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right)$$

$$\frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} = \nu \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right)$$

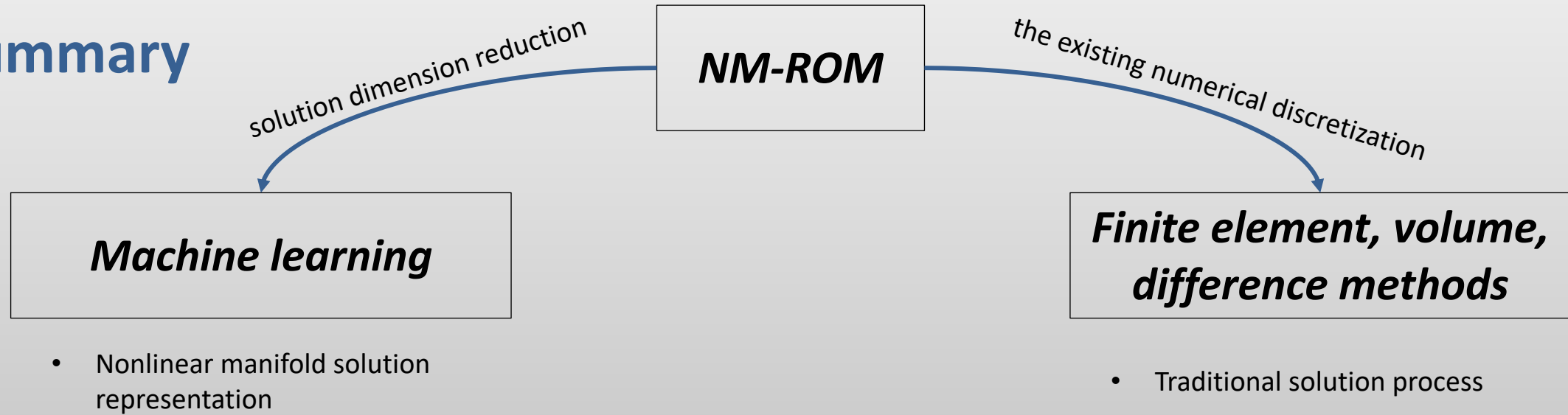
$$Re = 1/\nu = 10,000$$



method	NM-ROM	LS-ROM	BB
max. rel. error (%)	0.93	34.4	38.6
speed-up	11.6	26.8	119

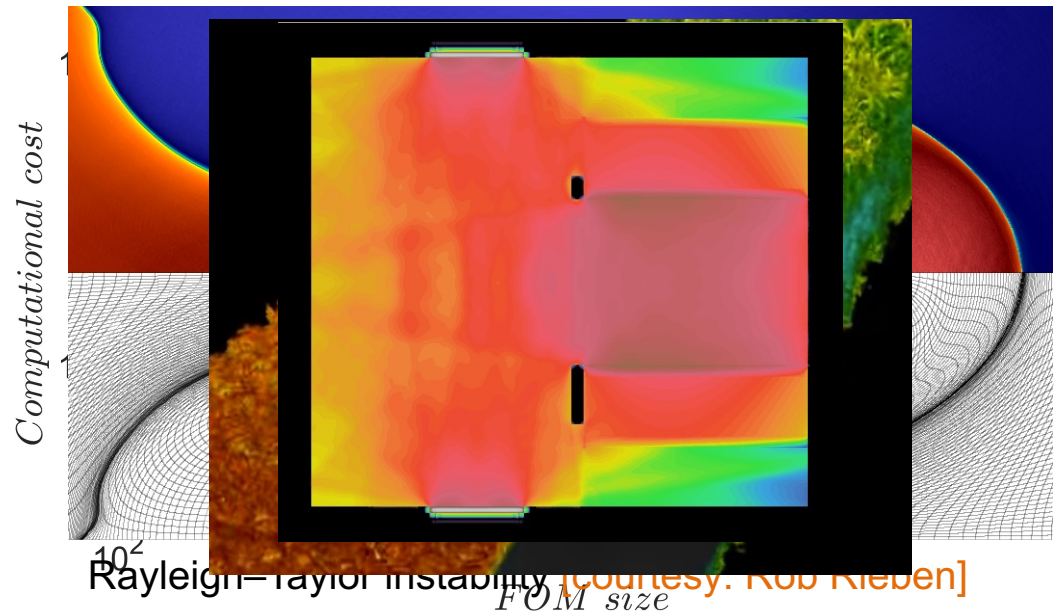
*Kim, Choi, Widemann, and Zohdi, "A fast and accurate physics-informed neural network reduced order model with shallow masked autoencoder." *arXiv preprint*, arXiv:2009.11990, 2020.

Summary

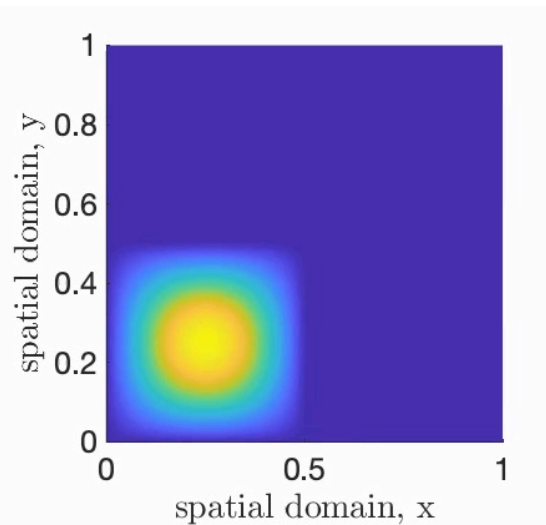


Future work

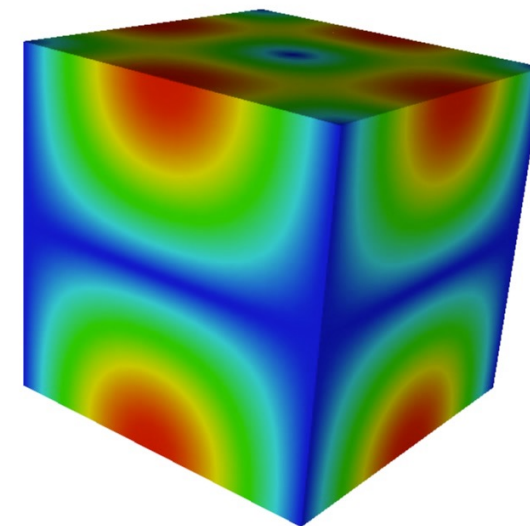
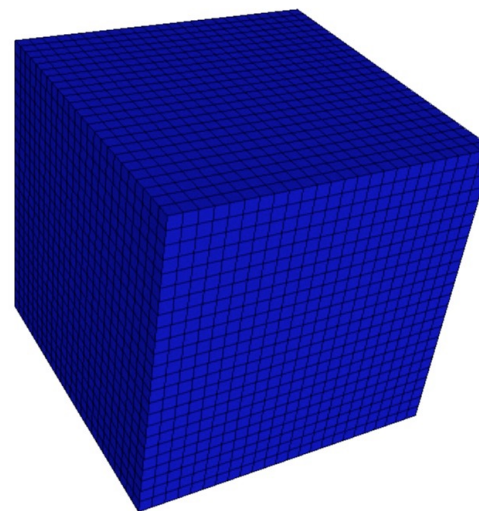
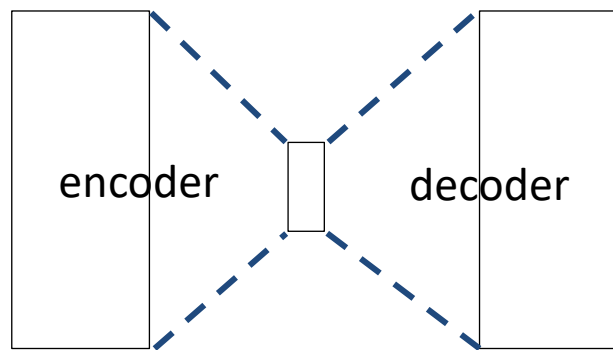
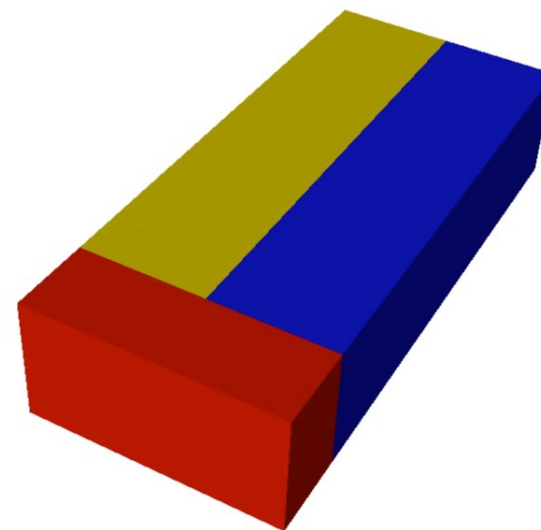
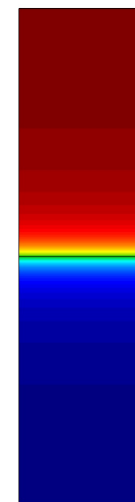
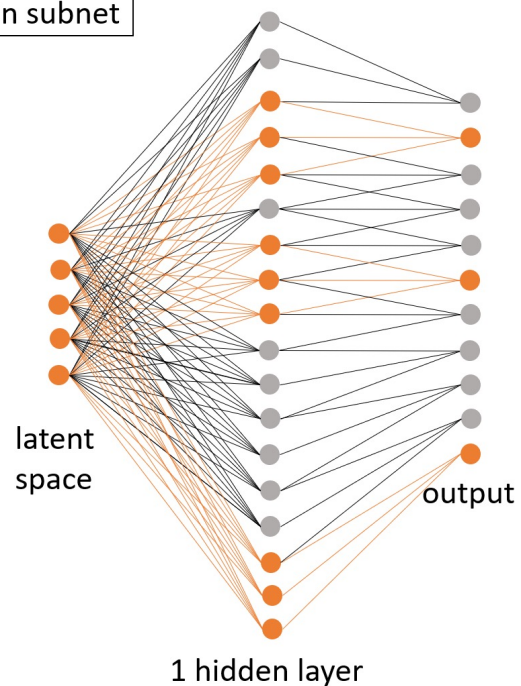
- **NM-ROM** for large-scale problems
- **NM-ROM** for mission critical problems, such as instability to turbulence, thermal radiative transfer, and shape charge simulations.



Questions? Email choi15@llnl.gov



● node in subnet
— edge in subnet





Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.