Reduced order modeling for finite element simulations through the partnership of MFEM and libROM

Siu Wun Cheung

CASC, LLNL

2nd MFEM Community Workshop

Oct 25, 2022

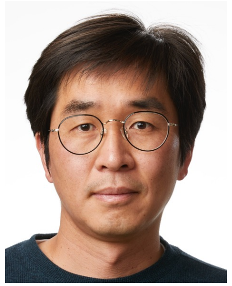
LLNL-PRES-841632

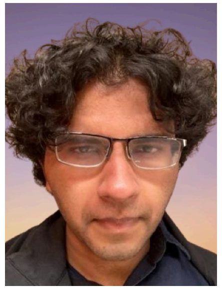
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

Lawrence Livermore National Laboratory

Awesome reduced order model team and collaborators


D. Copeland


Y. Choi


P. Vempati


K. Huynh

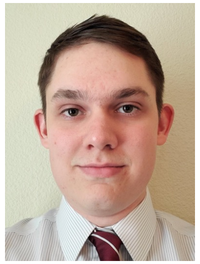

Y. Kim

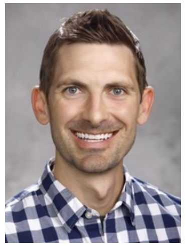

D. Widemann


T. Zohdi


J. Lauzon


J. Belof


Q. Huhn


K. Carlberg


P. Brown


B. Anderson


S. McBane


K. Willcox


R. Rieben


D. Ghosh

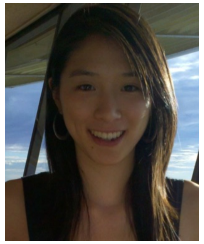

E. Chin

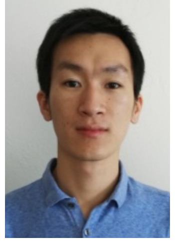

K. Springer


D. White


T. Kirchdoerfer


M. Juhasz

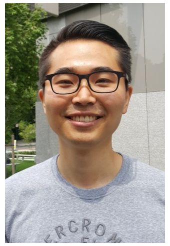

K. Wang


X. He


W. Fries


T. Kadeethum


N. Bouklas


Y. Shin


S. Khairallah


B. Afeyan


J. Ragusa


C.F. Jekel

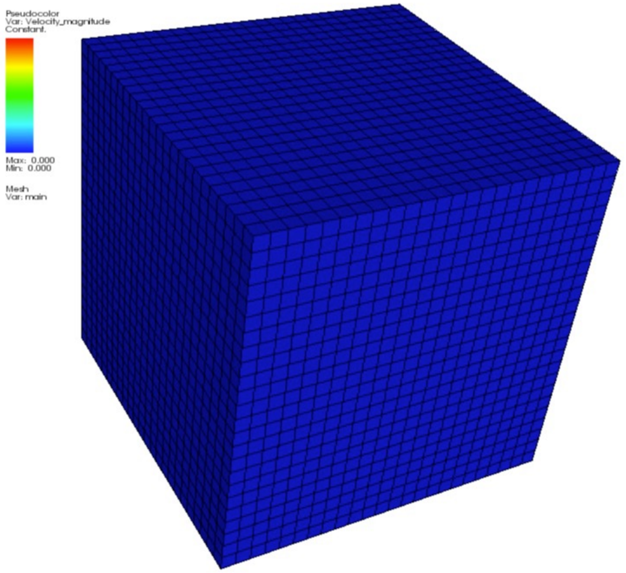

X. Zhong

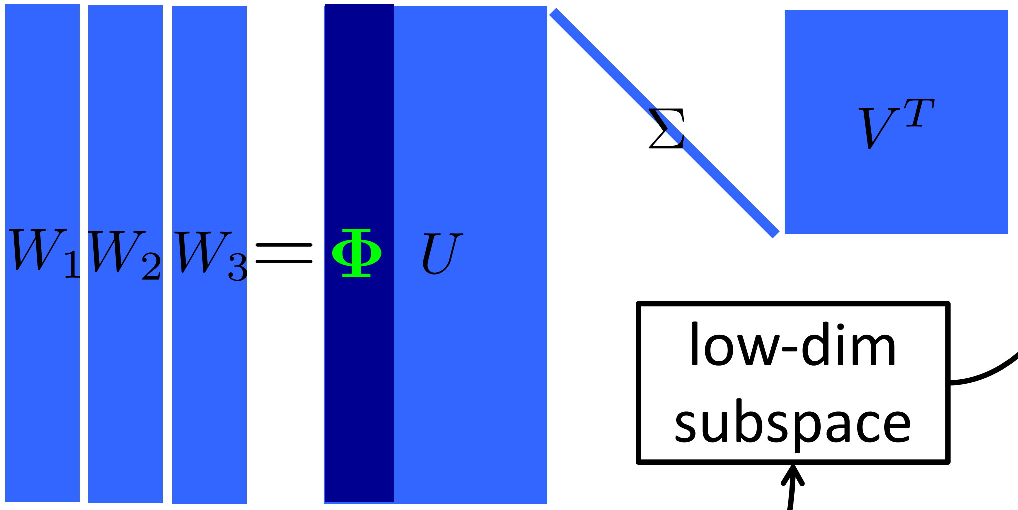

G. Oxberry

What is reduced order model (ROM)?

Goal: accelerate physics simulation without losing much accuracy by exploiting data [**data-driven**] (and governing equations [**physics-aware**]).

high fidelity models / experiments

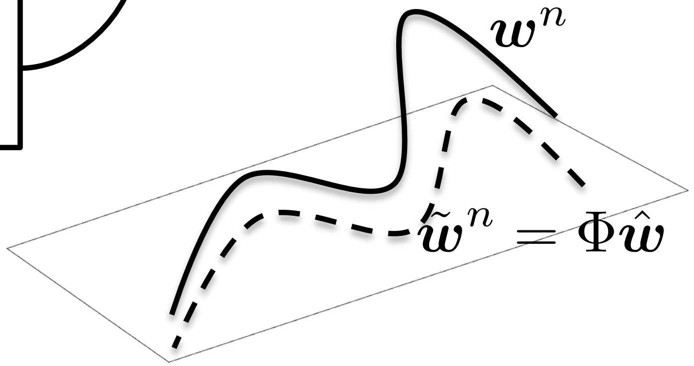




collect & compress data

low-dim subspace

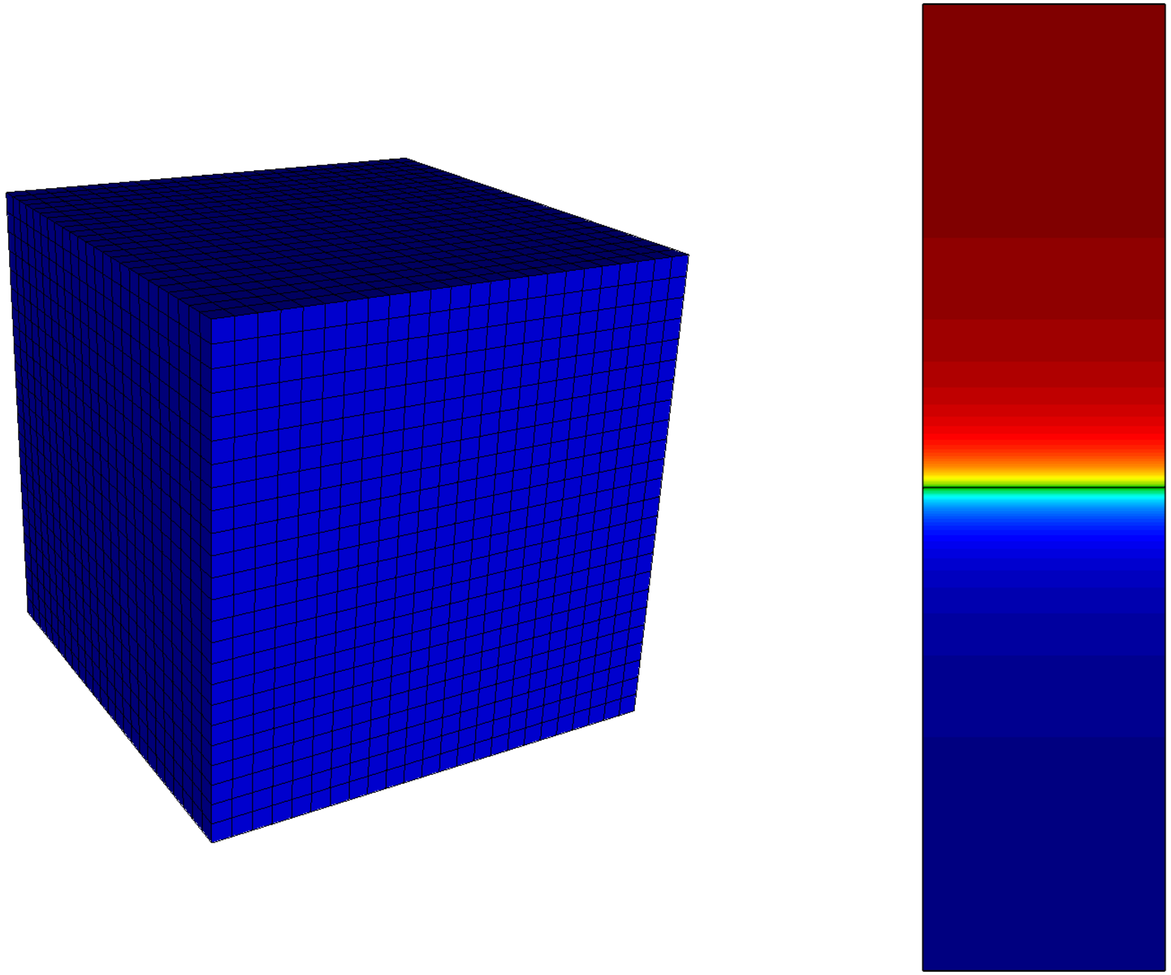
reduced-order system



approximate solution

MFEM examples & miniapps

- Example 1: Poisson equation
- Example 2: Linear elasticity
- Example 9: DG advection
- Example 10: Nonlinear elasticity
- Example 16: Nonlinear heat conduction
- Example 18: DG Euler equation
- Laghos: Lagrangian hydrodynamics



MFEM examples (https://mfem.org/examples)
MFEM examples on libROM (https://www.librom.net/examples.html)

libROM: open-source C++ library for data-driven physical simulations

- **GitHub page**: https://github.com/LLNL/libROM
- **Webpage** for libROM: www.librom.net

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

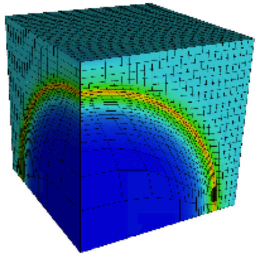
Use the GitHub issue tracker to report bugs or post questions or comments. See the About page for citation information.

Laghos ROM Miniapp

Laghos (LAGrangian High-Order Solver) is a miniapp that solves the time-dependent Euler equations of compressible gas dynamics in a moving Lagrangian frame using unstructured high-order finite element spatial discretization and explicit high-order time-stepping. **LaghosROM** introduces reduced order models of Laghos simulations.

A list of example problems that you can solve with LaghosROM includes Sedov blast, Gresho vortex, Taylor-Green vortex, triple-point, and Rayleigh-Taylor instability problems. Below are command line options for each problems and some numerical results. For each problem, four different phases need to be taken, i.e., the offline, hyper-reduction preprocessing, online, and restore phase. The online phase runs necessary full order model (FOM) to generate simulation data. libROM dynamically collects the data as the FOM simulation marches in time domain. In the hyper-reduction preprocessing phase, the libROM builds a library of reduced basis as well as hyper-reduction operators. The online phase runs the ROM and the restore phase projects the ROM solutions to the full order model dimension.

Sedov blast problem



Sedov blast problem is a three-dimensional standard shock hydrodynamic benchmark test. An initial delta source of internal energy deposited at the origin of a three-dimensional cube is considered. The computational domain is the unit cube $\tilde{\Omega} = [0,1]^3$ with wall boundary conditions on all surfaces, i.e., $v \cdot n = 0$. The initial velocity is given by $v = 0$. The initial density is given by $\rho = 1$. The initial energy is given by a delta function at the origin. The adiabatic index in the ideal gas equations of state is set $\gamma = 1.4$. The initial mesh is a uniform Catesian hexahedral mesh, which deforms over time. It can be seen that the radial symmetry is maintained in the shock wave propagation in both FOM and ROM simulations. One can reproduce the numerical result, following the command line options described below:

- **offline**: ./laghos -o twp_sedov -m ../data/cube01_hex.mesh -pt 211 -tf 0.8 -s 7 -pa -offline -visit -romsvds -ef 0.9999 -writesol -romos -rostype load -romsns -nwinsamp 21 -sample-stages
- **hyper-reduction preprocessing**: ./laghos -o twp_sedov -m ../data/cube01_hex.mesh -pt 211 -tf 0.8 -s 7 -pa -online -romsvds -romos -rostype load -romhrprep -romsns -romgs -nwin 66 -sfacv 2 -sface 2
- **online**: ./laghos -o twp_sedov -m ../data/cube01_hex.mesh -pt 211 -tf 0.8 -s 7 -pa -online -romsvds -romos -rostype load -romhr -romsns -romgs -nwin 66 -sfacv 2 -sface 2
- **restore**: ./laghos -o twp_sedov -m ../data/cube01_hex.mesh -pt 211 -tf 0.8 -s 7 -pa -restore -soldiff -romsvds -romos -rostype load -romsns -romgs -nwin 66

FOM solution time	ROM solution time	Speed-up	Velocity relative error
191 sec	8.3 sec	22.8	2.2e-4

libROM features

- Data Compression
  - Static SVD
  - Incremental SVD
  - Randomized SVD
- Hyper-reduction
  - DEIM
  - GNAT
  - S-OPT
  - EQP
- Non-intrusive methods
  - DMD
  - LaSDI
- Local reduced order model
  - Parametric interpolation
  - Greedy algorithm

SVD: singular value decomposition

```
(CAROM is the namespace of libROM)
Taking a MFEM Vector as snapshot
MFEM::Vector w;
CAROM::BasisGenerator generator;
generator.takeSample(w.GetData(), t, dt);
```

- Snapshot matrix:

$$\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3, \cdots, \mathbf{w}_m] \in \mathbb{R}^{N_s \times m}, \text{ where } \operatorname{rank}(\mathbf{W}) = n \leq m \leq N_s.$$

- Singular value decomposition:

$$\mathbf{W} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top = \sum_{i=1}^{n} \sigma_i \mathbf{u}_i \mathbf{v}_i^\top, \text{ where } \mathbf{U}^\top\mathbf{U} = \mathbf{V}^\top\mathbf{V} = \mathbf{I}_n$$

- Best low-rank approximation:

$$\mathbf{W}_k = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^\top \implies \begin{aligned} \inf_{\operatorname{rank}(\mathbf{Z})=k} \|\mathbf{W}-\mathbf{Z}\|_2 &= \|\mathbf{W}-\mathbf{W}_k\|_2 = \sigma_{k+1} \\ \inf_{\operatorname{rank}(\mathbf{Z})=k} \|\mathbf{W}-\mathbf{Z}\|_F &= \|\mathbf{W}-\mathbf{W}_k\|_F = \sum_{i=k+1}^{n} \sigma_i^2 \end{aligned}$$

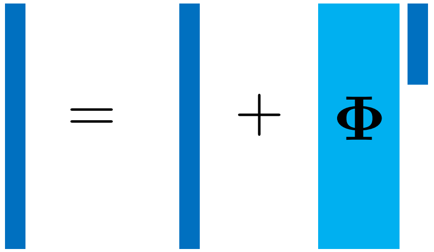
- High energy dominant modes:

$$\boldsymbol{\Phi} = \mathbf{U}[\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \cdots, \mathbf{e}_{n_s}] \in \mathbb{R}^{N_s \times n_s}, \text{ where } \sum_{i=n_s+1}^{n} \sigma_i^2 \leq (1-\delta) \sum_{i=1}^{n} \sigma_i^2$$

POD: proper orthogonal decomposition

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

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




```
Converting basis to MFEM::DenseMartix
CAROM::Matrix* Phi;
MFEM::DenseMatrix *reducedBasisT = new
DenseMatrix(Phi->getData(), ns, Ns);
```

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

Scales with FOM size: $\mathbb{R}^{N_s}$ → Hyper-reduction!

libROM features

- Data Compression
  - Static SVD
  - Incremental SVD
  - Randomized SVD
- Hyper-reduction
  - DEIM
  - GNAT
  - S-OPT
  - EQP
- Non-intrusive methods
  - DMD
  - LaSDI
- Local reduced order model
  - Parametric interpolation
  - Greedy algorithm

Hyper-reduction: nonlinear model reduction

- Approximate nonlinear term:

$$\mathbf{f} \approx \mathbf{\Phi}_f \hat{\mathbf{f}}, \quad \mathbf{\Phi}_f \in \mathbb{R}^{N_s \times n_f}, \quad n_s \leq n_f \ll N_s$$

- Interpolation at sampled rows with indices $\mathcal{Z} \subset \{1, 2, 3, \ldots, N_s\}$:

$$\hat{\mathbf{f}} = \arg\min_{\hat{\mathbf{g}} \in \mathbb{R}^{n_z}} \|\underbrace{\mathbf{Z}^\top}(\mathbf{f} - \mathbf{\Phi}_f \hat{\mathbf{g}})\|_2 \quad \rightarrow \quad \hat{\mathbf{f}} = (\mathbf{Z}^\top \mathbf{\Phi}_f \hat{f})^\dagger \mathbf{Z}^\top \mathbf{f}$$

$$\mathbf{Z} = [e_i]_{i \in \mathcal{Z}} \in \mathbb{R}^{N_s \times n_z}, n_f \leq |\mathcal{Z}| = n_z \ll N_s$$

- Replace nonlinear term:

$$\hat{\mathbf{w}}_n = \hat{\mathbf{w}}_{n-1} + \Delta t \underbrace{\mathbf{\Phi}^\top \mathbf{\Phi}_f (\mathbf{Z}^\top \mathbf{\Phi}_f)^\dagger}\underbrace{\mathbf{Z}^\top \mathbf{f}}(\mathbf{w}_{\text{ref}} + \mathbf{\Phi}\hat{\mathbf{w}}, t_{n+1}; \mu)$$

Offline stage: precompute

Online stage: evaluate only sampled entries

$$\mathbf{\Phi}^\top \mathbf{\Phi}_f (\mathbf{Z}^\top \mathbf{\Phi}_f)^\dagger \in \mathbb{R}^{n_f \times n_z} \qquad \mathbf{Z}^\top \mathbf{f} \in \mathbb{R}^{n_z}$$

libROM class of sample mesh for MFEM
`CAROM::SampleMeshManager`

- Sample mesh: all the connected indices to the sampled rows

Hyper-reduction: optimal sampling

- Oblique projection operator: $\mathbf{P}(\mathcal{Z}) = \mathbf{\Phi}_f(\mathbf{Z}^\top \mathbf{\Phi}_f)^\dagger \mathbf{Z}^\top$
- Oblique projection error: $\varepsilon(\mathbf{f};\mathcal{Z}) = \|(\mathbf{I} - \mathbf{P}(\mathcal{Z}))\mathbf{f}\|_2$

libROM routines of precompute hyperreduction

```
CAROM::DEIM(Phi_f, ns, nf, …
CAROM::QDEIM(Phi_f, ns, nf, …
CAROM::GNAT(Phi_f, ns, nf, …
CAROM::S_OPT(Phi_f, ns, nf, …
```

- Optimality of sampling indices:
  - True optimal set is not feasible: $\mathcal{Z}^*(\mathbf{f}) = \arg\min_{\mathcal{Z}} \varepsilon(\mathbf{f};\mathcal{Z})$
  - Greedy sampling for suboptimality: $\|\varepsilon(\mathbf{f};\mathcal{Z})\|_2 \leq \|(\mathbf{Z}^\top \mathbf{Q})^\dagger\|_2 \|(\mathbf{I} - \mathbf{\Phi}_f \mathbf{\Phi}_f^\dagger)\mathbf{f}\|_2$
    - Discrete Empirical Interpolation Method (DEIM[1,2])
    - Gauss Newton Approximate Tensor (GNAT[3])
    - S-OPT[4]

1. Chaturantabut and Sorensen. “Nonlinear model reduction via discrete empirical interpolation”. SISC 32 (2010), pp. 2737–2764.
2. Drmac and Gugercin. “A new selection operator for the discrete empirical interpolation method—improved a priori error bound and extensions”. SISC 38 (2016), A631–A648.
3. Carlberg, Bou-Mosleh, and Farhat. “Efficient non-linear model reduction via a least-squares Petrov-Galerkin projection and compressive tensor approximations”, IJNME 86 (2011), 155–181.
4. Lauzon, Cheung, Shin, Copeland, Huynh, and Choi. “S-OPT: A Points Selection Algorithm for Hyper-Reduction in Reduced Order Models”, arXiv preprint arXiv:2203.16494.

Hyper-reduction: reduced quadrature rule

- MFEM action on the basis:

$$(\boldsymbol{\Phi}^\top \mathbf{f})_i = Q(f\phi_i) = \sum_{q=1}^{N_q} \omega_q f(x_q)\phi_i(x_q)$$

- Reduced quadrature rule:

$$Q(f\phi_i) \approx \tilde{Q}(f\phi_i) = \sum_{q=1}^{N_q} \tilde{\omega}_q f(x_q)\phi_i(x_q), \text{ where } \tilde{\omega}_q = 0 \text{ except for } n_q \text{ many } q$$

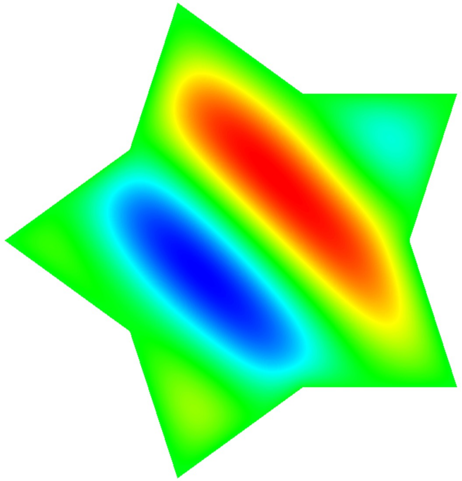
- Empirical quadrature procedure (EQP):

$$|Q(f_j\phi_i) - \tilde{Q}(f_j\phi_i)| \leq \delta \text{ for all } i = 1, 2, 3, \ldots, n_s, \text{ and } j = 1, 2, 3, \ldots, m$$

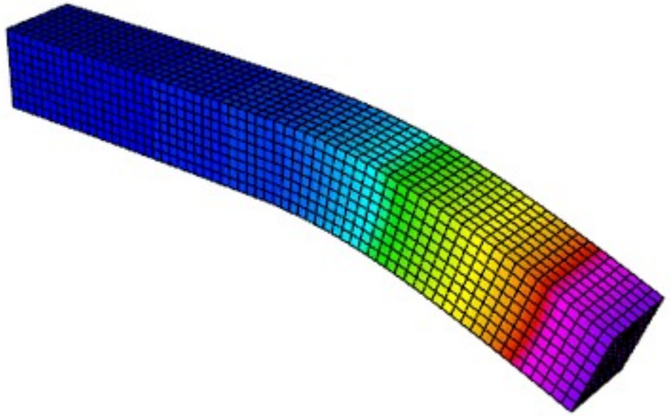
- Optimization solved by non-negative least-squares (NNLS)
- Applied to nonlinear heat conduction
- Key idea: more quadrature points than constraints
    - NURBS patch quadrature (MFEM PR#3088 by Dylan Copeland)

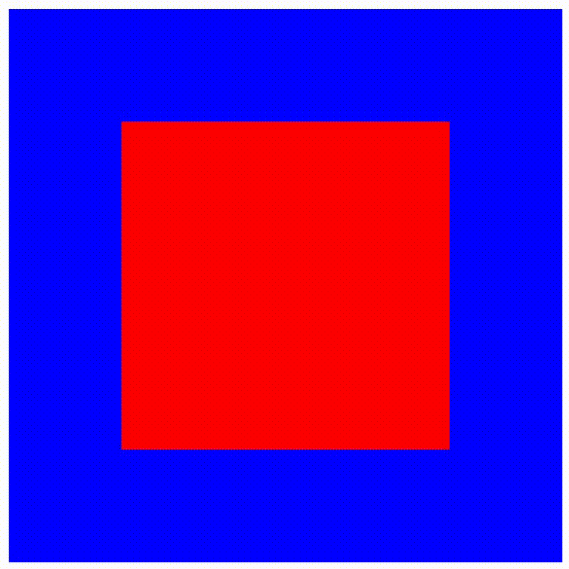
Du and Yano. “Efficient hyperreduction of high-order discontinuous Galerkin methods: Element-wise and point-wise reduced quadrature formulations”. JCP 466 (2022), 111399.

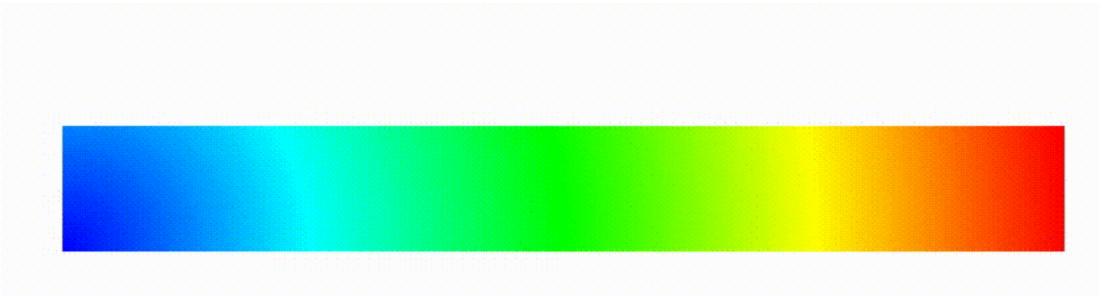
libROM examples of accelerating MFEM simulations by pROM

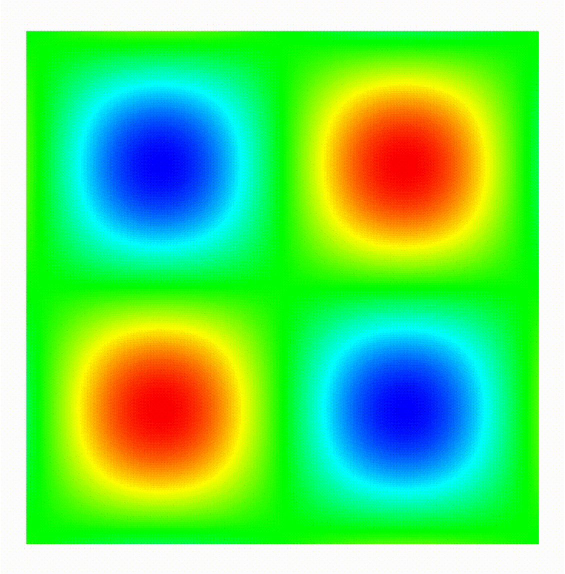


Example	Rel. error	Speed-up
Poisson problem	6.4e-4	7.5
Linear elasticity	8.1e-4	1.4e4
DG advection	1.2e-2	62.5
Nonlinear elasticity	6.6e-3	8.5
Nonlinear heat conduction	1.6e-3	24.5









MFEM examples (https://mfem.org/examples)
MFEM examples on libROM (https://www.librom.net/examples.html)

Nonlinear pROM for Lagrangian hydrodynamics

- POD reduced basis

$$\boldsymbol{\Phi}_v = \begin{bmatrix} \phi_v^1 & \cdots & \phi_v^{n_v} \end{bmatrix} \in \mathbb{R}^{N_\mathcal{V} \times n_v}$$
$$\boldsymbol{\Phi}_e = \begin{bmatrix} \phi_e^1 & \cdots & \phi_e^{n_e} \end{bmatrix} \in \mathbb{R}^{N_\mathcal{E} \times n_e}$$
$$\boldsymbol{\Phi}_x = \begin{bmatrix} \phi_x^1 & \cdots & \phi_x^{n_x} \end{bmatrix} \in \mathbb{R}^{N_\mathcal{V} \times n_x}$$

- Solution representation

$$\tilde{\boldsymbol{v}}(t;\boldsymbol{\mu}) = \boldsymbol{v}_{\mathrm{os}}(\boldsymbol{\mu}) + \boldsymbol{\Phi}_v \widehat{\boldsymbol{v}}(t;\boldsymbol{\mu})$$
$$\tilde{\boldsymbol{e}}(t;\boldsymbol{\mu}) = \boldsymbol{e}_{\mathrm{os}}(\boldsymbol{\mu}) + \boldsymbol{\Phi}_e \widehat{\boldsymbol{e}}(t;\boldsymbol{\mu})$$
$$\tilde{\boldsymbol{x}}(t;\boldsymbol{\mu}) = \boldsymbol{x}_{\mathrm{os}}(\boldsymbol{\mu}) + \boldsymbol{\Phi}_x \widehat{\boldsymbol{x}}(t;\boldsymbol{\mu})$$

- Reduced mass matrices

$$\widehat{\boldsymbol{M}}_\mathcal{V} = \boldsymbol{\Phi}_v^T \boldsymbol{M}_\mathcal{V} \boldsymbol{\Phi}_v$$
$$\widehat{\boldsymbol{M}}_\mathcal{E} = \boldsymbol{\Phi}_e^T \boldsymbol{M}_\mathcal{E} \boldsymbol{\Phi}_e$$

$$\frac{d\widehat{\boldsymbol{v}}}{dt} = -\widehat{\mathsf{F}^1}(\boldsymbol{v}_{\mathrm{os}} + \boldsymbol{\Phi}_v \widehat{\boldsymbol{v}}, \boldsymbol{e}_{\mathrm{os}} + \boldsymbol{\Phi}_e \widehat{\boldsymbol{e}}, \boldsymbol{x}_{\mathrm{os}} + \boldsymbol{\Phi}_x \widehat{\boldsymbol{x}}, t; \boldsymbol{\mu})$$
$$\frac{d\widehat{\boldsymbol{e}}}{dt} = \widehat{\mathsf{F}^{tv}}(\boldsymbol{v}_{\mathrm{os}} + \boldsymbol{\Phi}_v \widehat{\boldsymbol{v}}, \boldsymbol{e}_{\mathrm{os}} + \boldsymbol{\Phi}_e \widehat{\boldsymbol{e}}, \boldsymbol{x}_{\mathrm{os}} + \boldsymbol{\Phi}_x \widehat{\boldsymbol{x}}, t; \boldsymbol{\mu})$$
$$\frac{d\widehat{\boldsymbol{x}}}{dt} = \boldsymbol{\Phi}_x^T \boldsymbol{v}_{\mathrm{os}} + \boldsymbol{\Phi}_x^T \boldsymbol{\Phi}_v \widehat{\boldsymbol{v}}$$

- DEIM oblique projection

$$\mathcal{P}_{\mathsf{F}^1} = \boldsymbol{\Phi}_{\mathsf{F}^1} \left(\mathbf{Z}_{\mathsf{F}^1}^T \boldsymbol{\Phi}_{\mathsf{F}^1}\right)^\dagger \mathbf{Z}_{\mathsf{F}^1}^T \in \mathbb{R}^{N_\mathcal{V} \times N_\mathcal{V}}$$
$$\mathcal{P}_{\mathsf{F}^{tv}} = \boldsymbol{\Phi}_{\mathsf{F}^{tv}} \left(\mathbf{Z}_{\mathsf{F}^{tv}}^T \boldsymbol{\Phi}_{\mathsf{F}^{tv}}\right)^\dagger \mathbf{Z}_{\mathsf{F}^{tv}}^T \in \mathbb{R}^{N_\mathcal{E} \times N_\mathcal{E}}$$

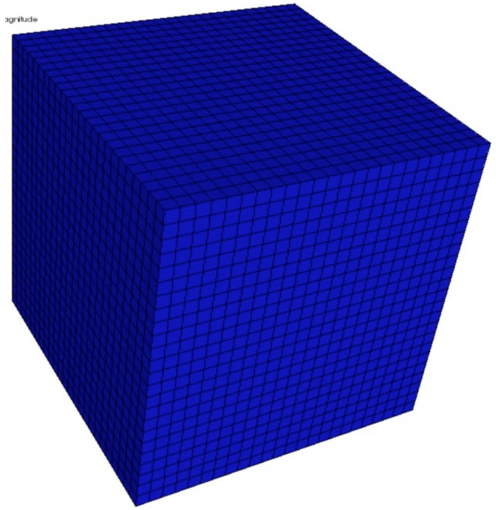
- SNS source term basis

$$\boldsymbol{\Phi}_{\mathsf{F}^1} = \boldsymbol{M}_\mathcal{V} \boldsymbol{\Phi}_v$$
$$\boldsymbol{\Phi}_{\mathsf{F}^{tv}} = \boldsymbol{M}_\mathcal{E} \boldsymbol{\Phi}_e$$

1. Dobrev, Kolev, and Rieben. “High-order curvilinear finite element methods for Lagrangian hydrodynamics”, SISC 34 (2012), B606–B641. **Laghos miniapp** (https://github.com/CEED/Laghos)
2. Copeland, Cheung, Huynh, Choi, “Reduced order models for Lagrangian hydrodynamics,” CMAME 388 (2022), 114259. ROM for **pROM for Laghos miniapp** (https://github.com/CEED/Laghos/tree/rom)

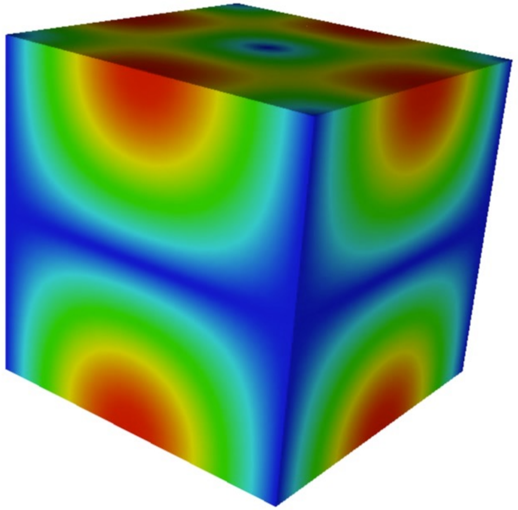
Nonlinear pROM for Lagrangian hydrodynamics

Sedov blast explosion



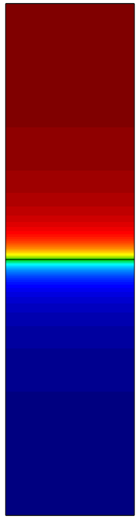
Relative error
Position: 2.2e-5
Velocity: 2.2e-4
Energy : 2.3e-4
Speedup: 22.8

Taylor-Green vortex



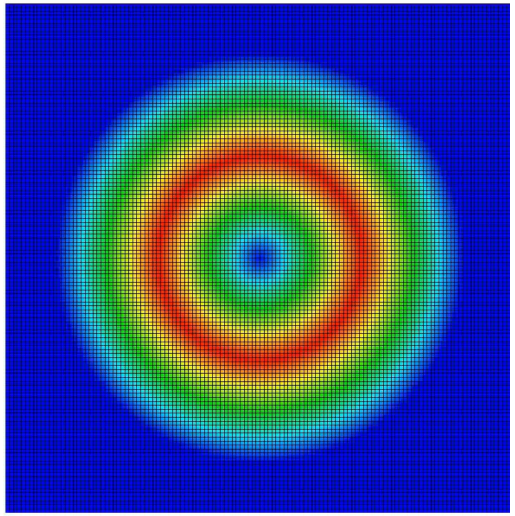
Relative error
Position: 1.8e-8
Velocity: 1.1e-6
Energy : 1.0e-7
Speedup: 31.2

Rayleigh-Taylor instability



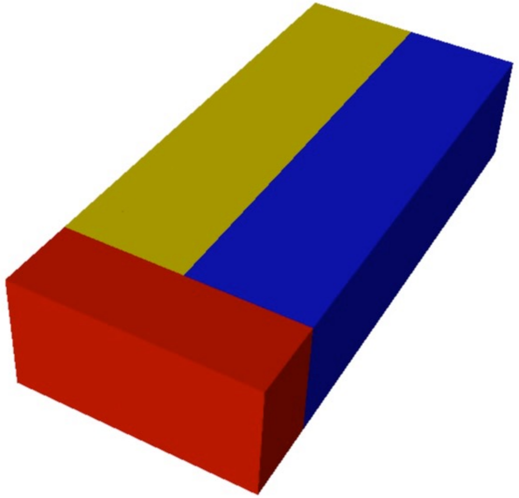
Relative error
Position: 5.3e-5
Velocity: 7.8e-3
Energy : 2.4e-5
Speedup: 14.62

Gresho vortex



Relative error
Position: 4.1e-6
Velocity: 2.1e-4
Energy : 2.4e-5
Speedup: 25.9

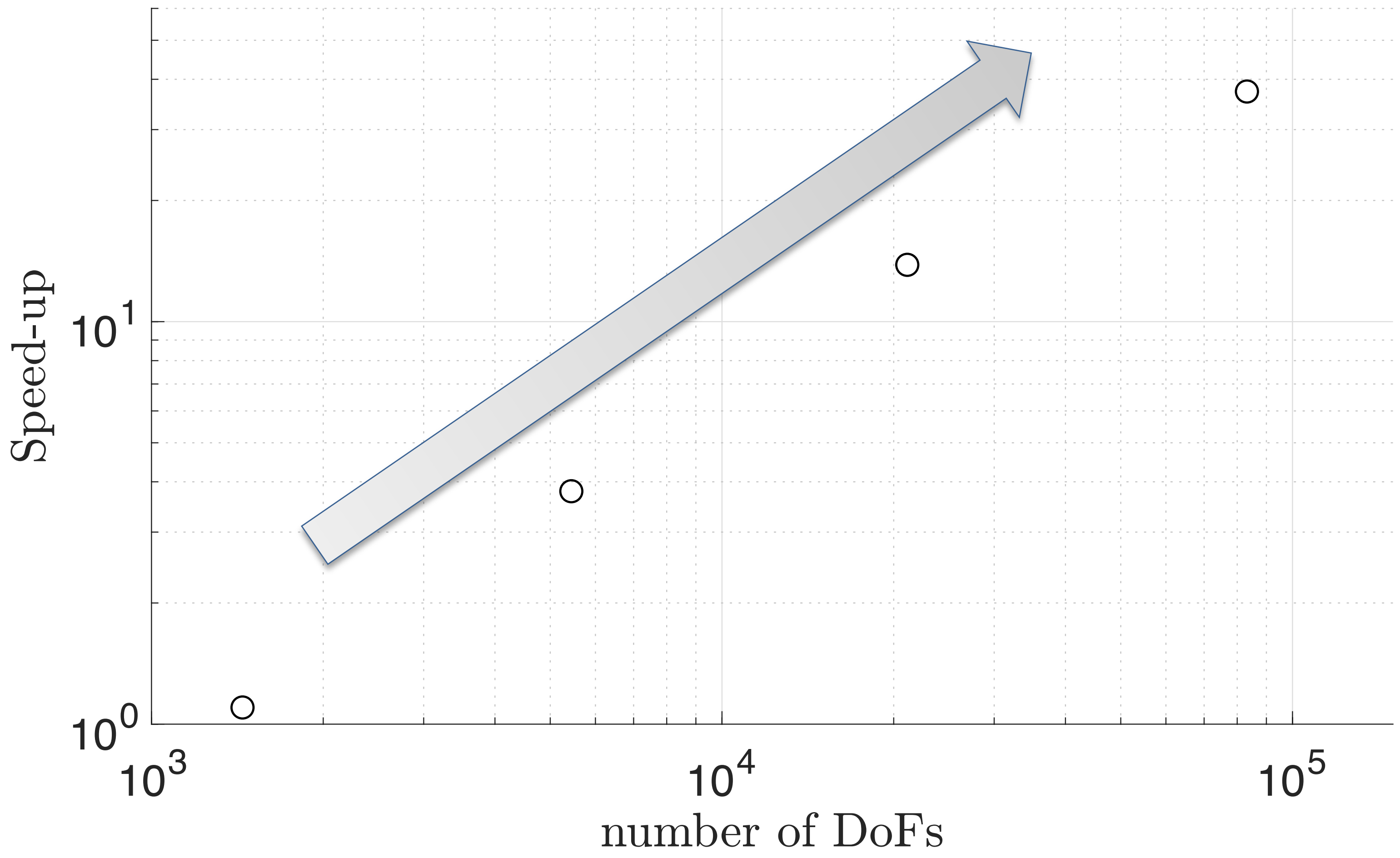
Triple-point expansion

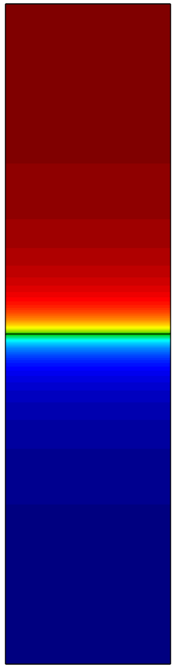


Relative error
Position: 3.1e-5
Velocity : 8.1e-4
Energy : 2.8e-4
Speedup: 87.8

* Copeland, Cheung, Huynh, Choi, “Reduced order models for Lagrangian hydrodynamics." CMAME, 388, 110841, 2022.
* Cheung, Choi, Copeland, Huynh, “Local Lagrangian reduced-order modeling for the Rayleigh–Taylor instability by solution manifold decomposition.” JCP, 472, 111655, 2023.

Speed-up increases with problem size





Kinematic dofs: 594
Energy dofs: 256

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

libROM features

- Data Compression
  - Static SVD
  - Incremental SVD
  - Randomized SVD
- Hyper-reduction
  - DEIM
  - GNAT
  - S-OPT
  - EQP
- Non-intrusive methods
  - DMD
  - LaSDI
- Local reduced order model
  - Parametric interpolation
  - Greedy algorithm

DMD: Dynamic mode decomposition

- Linear approximation of discrete dynamic system:

$$\text{Find } \mathbf{A} \in \mathbb{R}^{N_s \times N_s} \text{ such that } \mathbf{w}_k \approx \mathbf{A}\mathbf{w}_{k-1}, \text{ where } \mathbf{w}_k = \mathbf{w}(t_0 + k\Delta t) \in \mathbb{R}^{N_s}$$

- Snapshot matrices:

$$\mathbf{W}^- = [\mathbf{w}_0, \mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_{m-1}] \in \mathbb{R}^{N_s \times m}$$
$$\mathbf{W}^+ = [\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3, \cdots, \mathbf{w}_m] \in \mathbb{R}^{N_s \times m}$$

Some libROM routines of DMD for MFEM

```
MFEM::Vector w;
CAROM::DMD(Ns, dt);
dmd.takeSample(w.GetData(), t);
dmd.train(ef);
CAROM::Vector* result_u = dmd.predict(t);
```

- Data compression by truncated SVD:

$$\mathbf{W}^- = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top \qquad \mathbf{\Phi} = \mathbf{U}[\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \cdots, \mathbf{e}_{n_s}] \in \mathbb{R}^{N_s \times n_s}$$
$$\mathbf{\Sigma}_r = \operatorname{diag}(\sigma_1, \sigma_2, \sigma_3, \cdots, \sigma_{n_s}) \qquad \mathbf{\Psi} = \mathbf{V}[\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \cdots, \mathbf{e}_{n_s}] \in \mathbb{R}^{m \times n_s}$$

- Low-rank linear approximation:

$$\mathbf{A}_r = \mathbf{\Phi}^\top \mathbf{W}^+ \mathbf{\Psi} \mathbf{\Sigma}_r^{-1} \approx \mathbf{\Phi}^\top \mathbf{A} \mathbf{\Phi}$$

- Dynamic modes:

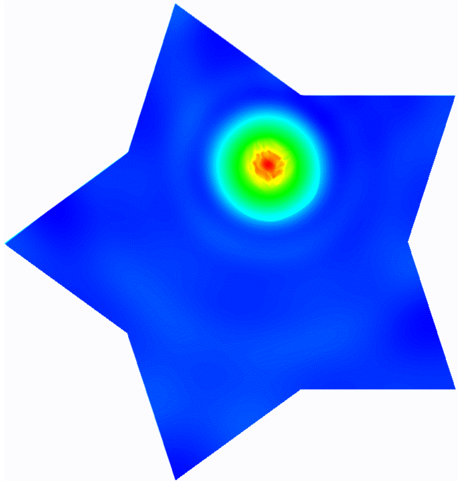
$$\mathbf{A}_r \mathbf{X} = \mathbf{X}\mathbf{\Lambda}, \quad \mathbf{Y} = \mathbf{\Phi}\mathbf{X}$$

- DMD prediction:

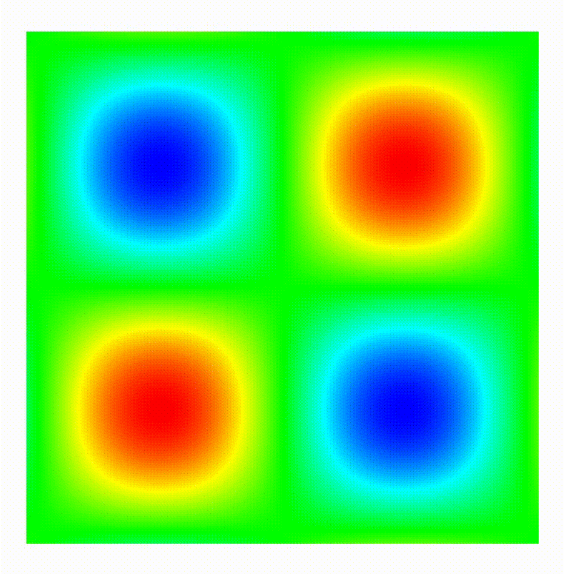
$$\tilde{\mathbf{w}}(t) = \mathbf{\Phi}\mathbf{X}\mathbf{\Lambda}^{\frac{t-t_0}{\Delta t}}\mathbf{X}^{-1}\mathbf{\Phi}^\top \mathbf{w}_0$$

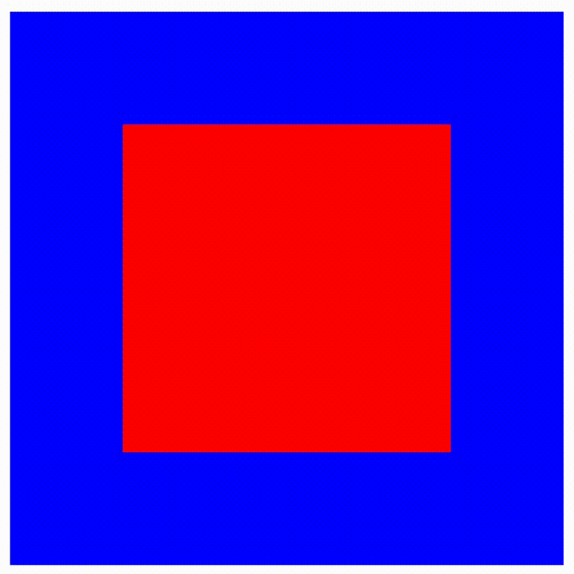
Does not require invasive changes in the high-fidelity solver! Applied to accelerate MFEM, Blast, ALE3D, HyPar, etc.

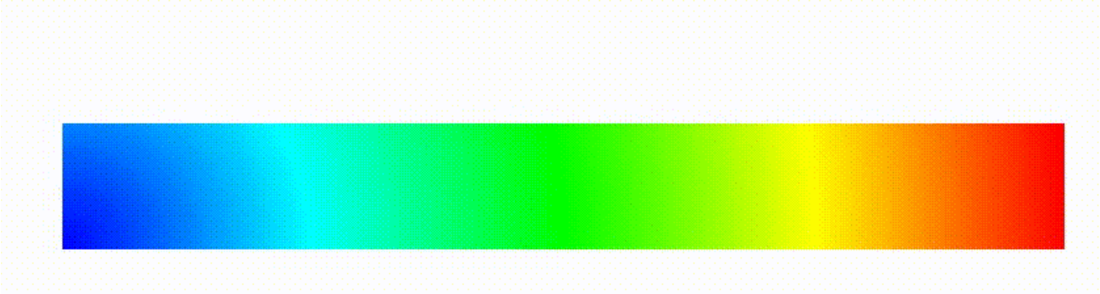
libROM examples of accelerating MFEM simulations by DMD



Example	Rel. error	Speed-up
DG advection	1.9e-4	2.7e2
Nonlinear elasticity	1.4e-3	9.5
DG Euler equation	1.2e-4	4.0e3
Nonlinear heat conduction	7.0e-3	11.1







MFEM examples (https://mfem.org/examples)
MFEM examples on libROM (https://www.librom.net/examples.html)

libROM features

- Data Compression
  - Static SVD
  - Incremental SVD
  - Randomized SVD
- Hyper-reduction
  - DEIM
  - GNAT
  - S-OPT
  - EQP
- Non-intrusive methods
  - DMD
  - LaSDI
- Local reduced order model
  - Parametric interpolation
  - Greedy algorithm

Local ROM v.s. Global ROM

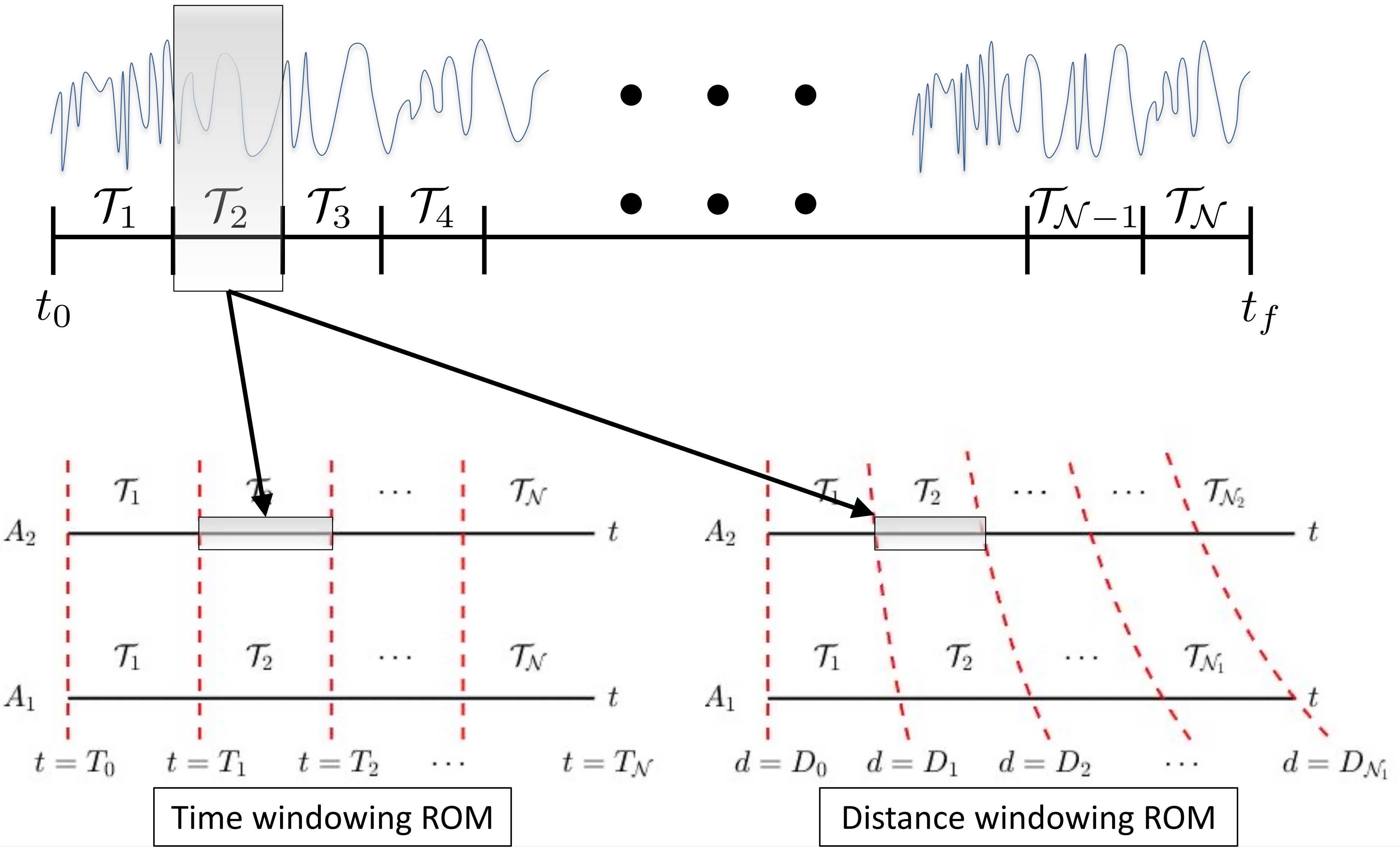
- Components of ROM
  - Compression operator, e.g. reduced basis $\mathbf{\Phi}$
  - Time-marching/prediction components, e.g. sampling matrix $\mathbf{Z}$ in hyper-reduction or eigenpairs $(\mathbf{\Lambda}, \mathbf{X})$ in DMD.

- Global ROM: identical ROM components for all $(t, \mu) \in [t_0, t_f] \times \mathcal{D}$

- Local ROM: ROM components depend on $(t, \mu) \in [t_0, t_f] \times \mathcal{D}$
  - Enhanced data compressibility!
  - Faster and more accurate prediction!
  - Higher generalization ability!
  - Localization approaches
    - Time/distance-windowing ROM for advective phenomena
    - Parametric interpolation

Time/distance windowing ROM for Lagrangian hydrodynamics

- *Offline phase*
  - Collect data
  - Classify data
  - Compress data
- *Online phase*
  - Assign local ROM
  - Form low-order system
  - Solve inexpensive ROM solution

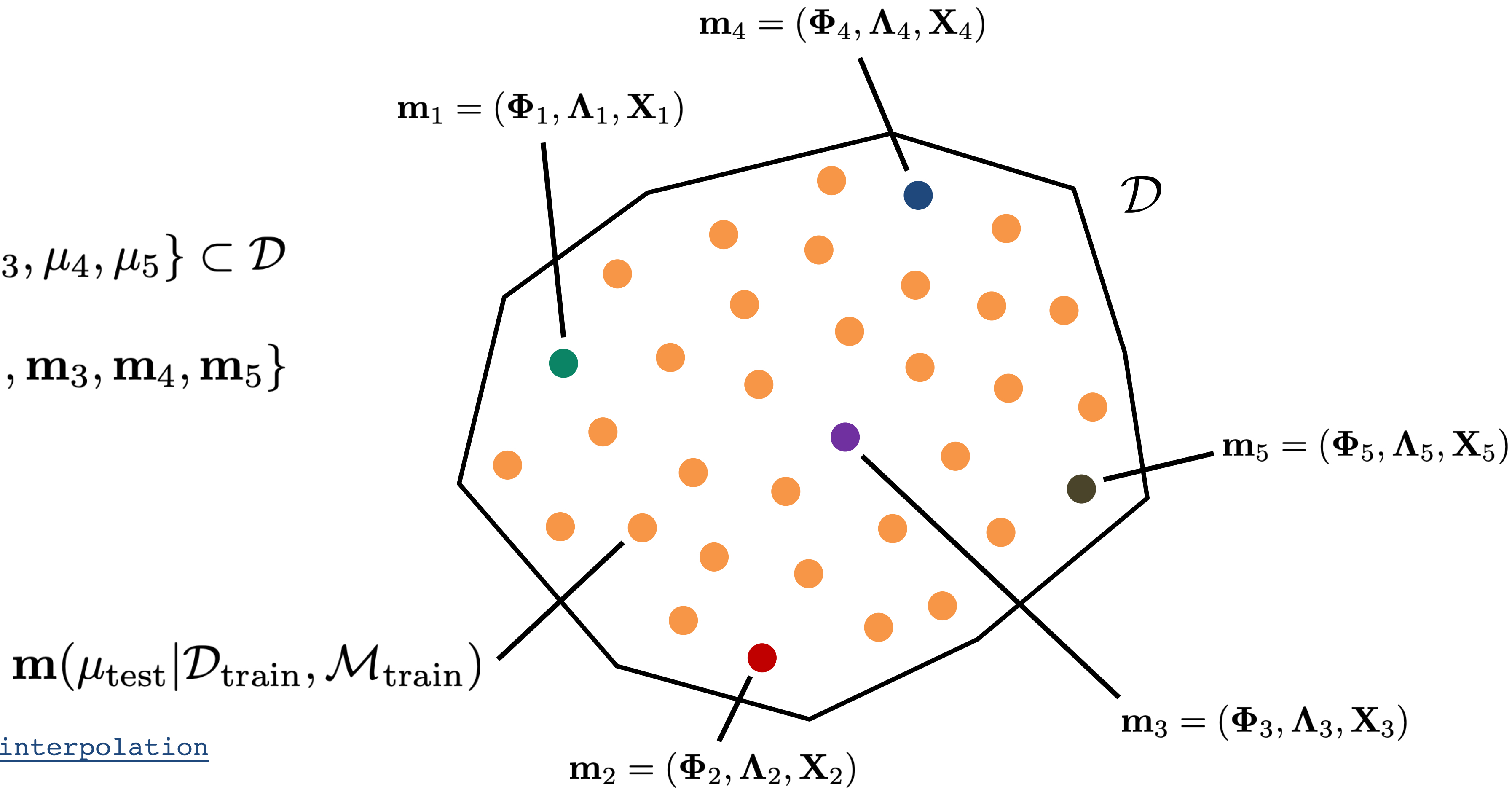


Parametric interpolation

$$\tilde{\mathbf{w}}(t; \mathbf{w}_0, \mathbf{m}) = \mathbf{\Phi}\mathbf{X}\mathbf{\Lambda}^{\frac{t-t_0}{\Delta t}}\mathbf{X}^{-1}\mathbf{\Phi}^\top \mathbf{w}_0 \qquad \mathbf{m} = (\mathbf{\Phi}, \mathbf{\Lambda}, \mathbf{X})$$

$$\mathcal{D}_{\text{train}} = \{\mu_1, \mu_2, \mu_3, \mu_4, \mu_5\} \subset \mathcal{D}$$

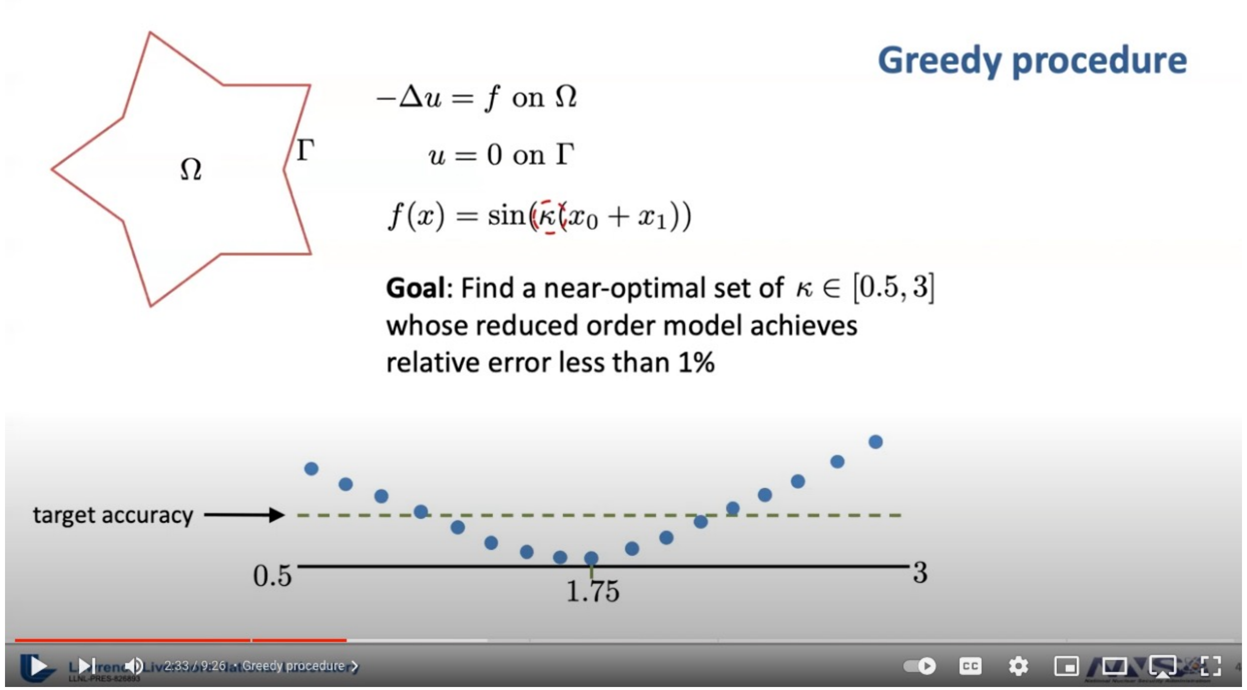
$$\mathcal{M}_{\text{train}} = \{\mathbf{m}_1, \mathbf{m}_2, \mathbf{m}_3, \mathbf{m}_4, \mathbf{m}_5\}$$

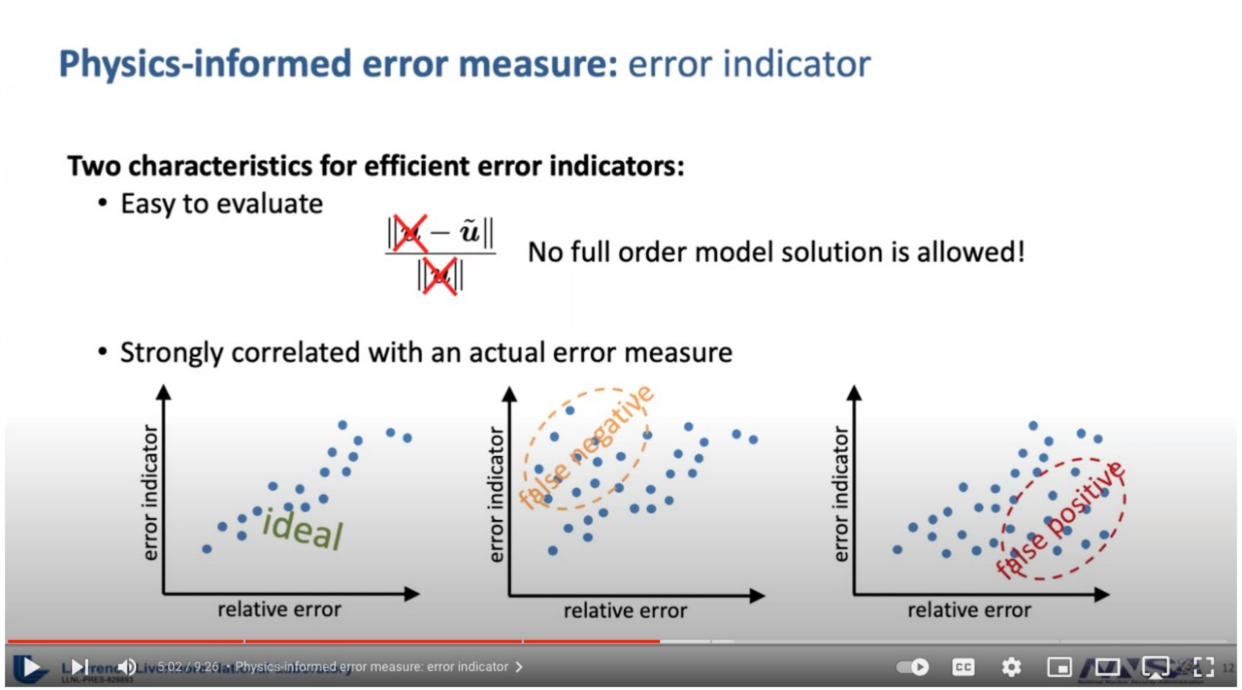


[libROM class of parametric interpolation]
`CAROM::MatrixInterpolator`

Physics-informed greedy sampling

- Watch this YouTube video (less than 10 minutes) : https://youtu.be/A5JlIXRHxrI

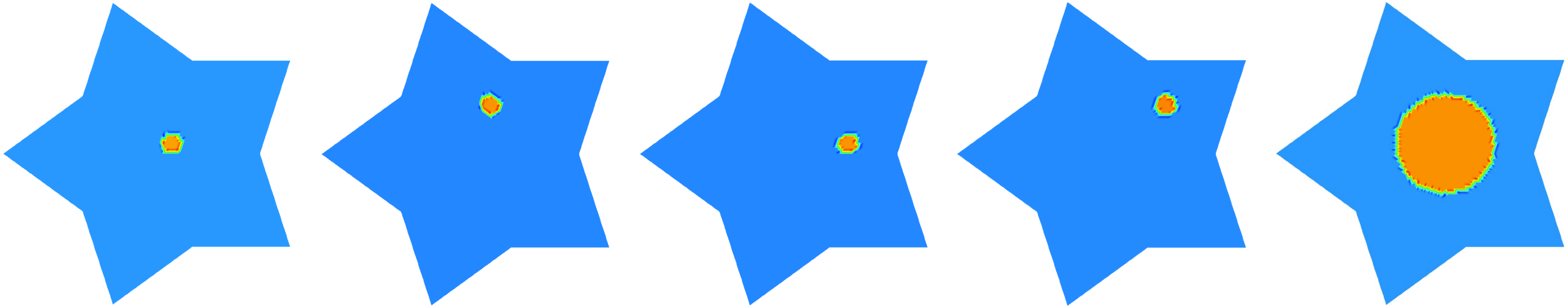




libROM class of greedy sampling
CAROM::GreedySampler

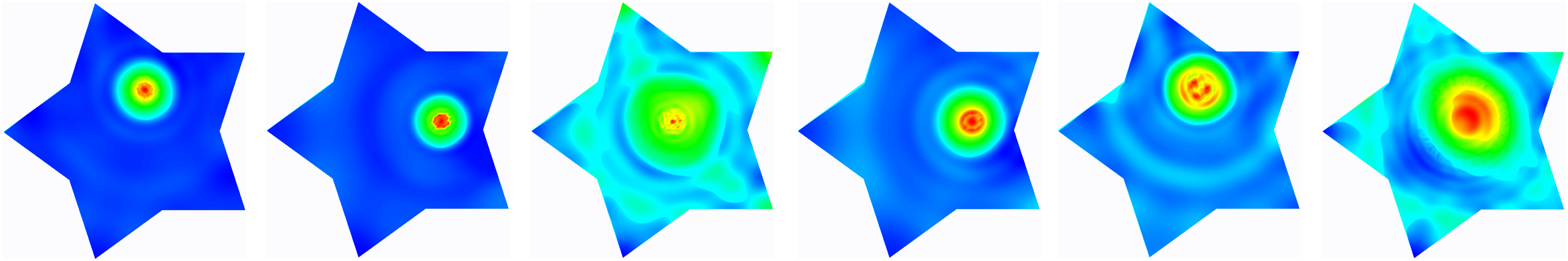
pDMD: Parametric Dynamic Mode Decomposition

Training initial conditions



$(r, cx, cy) = (0.1, 0.1, 0.1)$ $(0.1, 0.1, 0.5)$ $(0.1, 0.5, 0.1)$ $(0.1, 0.5, 0.5)$ $(0.5, 0.1, 0.1)$

Testing points



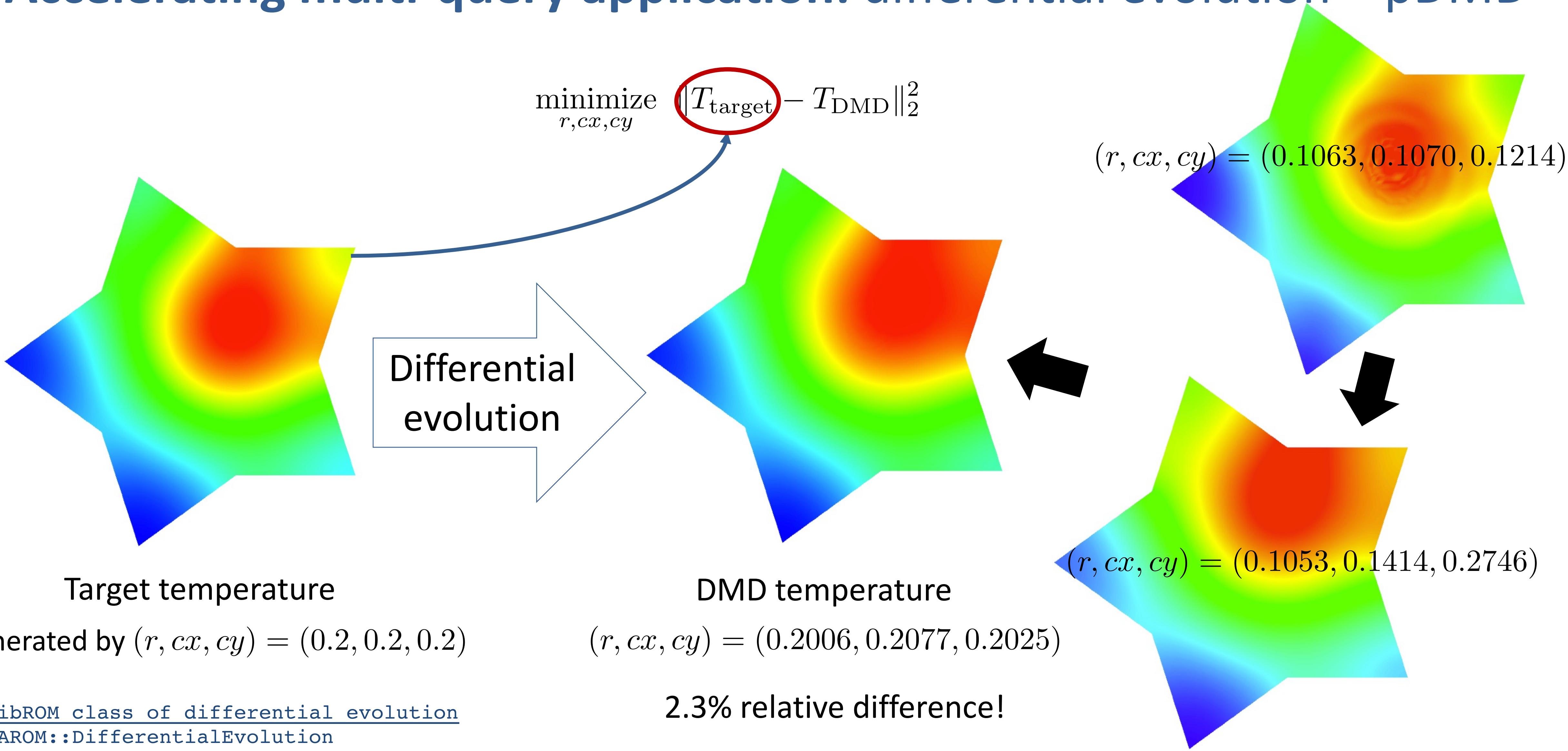
	$(0.25, 0.2, 0.4)$	$(0.2, 0.4, 0.2)$	$(0.3, 0.3, 0.3)$	$(0.3, 0.4, 0.2)$	$(0.2, 0.3, 0.4)$	$(0.4, 0.2, 0.3)$
DMD rel. error	**6.9e-3**	**4.1e-3**	**1.3e-2**	**8.4e-3**	**7.9e-3**	**9.6e-3**

`libROM routine for parametric DMD`
`CAROM::getParametricDMD`

Huhn, Tano, Ragusa, Choi, "Parametric Dynamic Mode Decomposition for Reduced Order Modeling." arXiv:2204.12006, 2022.

Accelerating multi-query application: differential evolution + pDMD

$$\underset{r,cx,cy}{\text{minimize}} \; \|T_{\text{target}} - T_{\text{DMD}}\|_2^2$$



Target temperature

generated by $(r, cx, cy) = (0.2, 0.2, 0.2)$

DMD temperature

$(r, cx, cy) = (0.2006, 0.2077, 0.2025)$

2.3% relative difference!

`libROM class of differential evolution`
`CAROM::DifferentialEvolution`

Questions?

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

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

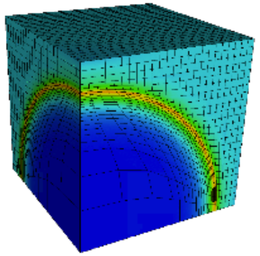
Use the GitHub issue tracker to report bugs or post questions or comments. See the About page for citation information.

Laghos ROM Miniapp

Laghos (LAGrangian High-Order Solver) is a miniapp that solves the time-dependent Euler equations of compressible gas dynamics in a moving Lagrangian frame using unstructured high-order finite element spatial discretization and explicit high-order time-stepping. LaghosROM introduces reduced order models of Laghos simulations.

A list of example problems that you can solve with LaghosROM includes Sedov blast, Gresho vortex, Taylor-Green vortex, triple-point, and Rayleigh-Taylor instability problems. Below are command line options for each problems and some numerical results. For each problem, four different phases need to be taken, i.e., the offline, hyper-reduction preprocessing, online, and restore phase. The online phase runs necessary full order model (FOM) to generate simulation data. libROM dynamically collects the data as the FOM simulation marches in time domain. In the hyper-reduction preprocessing phase, the libROM builds a library of reduced basis as well as hyper-reduction operators. The online phase runs the ROM and the restore phase projects the ROM solutions to the full order model dimension.

Sedov blast problem



Sedov blast problem is a three-dimensional standard shock hydrodynamic benchmark test. An initial delta source of internal energy deposited at the origin of a three-dimensional cube is considered. The computational domain is the unit cube $\tilde{\Omega} = [0,1]^3$ with wall boundary conditions on all surfaces, i.e., $v \cdot n = 0$. The initial velocity is given by $v = 0$. The initial density is given by $\rho = 1$. The initial energy is given by a delta function at the origin. The adiabatic index in the ideal gas equations of state is set $\gamma = 1.4$. The initial mesh is a uniform Catesian hexahedral mesh, which deforms over time. It can be seen that the radial symmetry is maintained in the shock wave propagation in both FOM and ROM simulations. One can reproduce the numerical result, following the command line options described below:

- **offline**: ./laghos -o twp_sedov -m ../data/cube01_hex.mesh -pt 211 -tf 0.8 -s 7 -pa -offline -visit -romsvds -ef 0.9999 -writesol -romos -rostype load -romsns -nwinsamp 21 -sample-stages
- **hyper-reduction preprocessing**: ./laghos -o twp_sedov -m ../data/cube01_hex.mesh -pt 211 -tf 0.8 -s 7 -pa -online -romsvds -romos -rostype load -romhrprep -romsns -romgs -nwin 66 -sfacv 2 -sface 2
- **online**: ./laghos -o twp_sedov -m ../data/cube01_hex.mesh -pt 211 -tf 0.8 -s 7 -pa -online -romsvds -romos -rostype load -romhr -romsns -romgs -nwin 66 -sfacv 2 -sface 2
- **restore**: ./laghos -o twp_sedov -m ../data/cube01_hex.mesh -pt 211 -tf 0.8 -s 7 -pa -restore -soldiff -romsvds -romos -rostype load -romsns -romgs -nwin 66

FOM solution time	ROM solution time	Speed-up	Velocity relative error
191 sec	8.3 sec	22.8	2.2e-4

Thank you for your attention!

Lawrence Livermore National Laboratory