Sparse, approximate quadrature for acceleration of isogeometric analysis and reduced order models

2024 MFEM Community Workshop

October 23, 2024

Dylan Copeland
Center for Applied Scientific Computing

LLNL-PRES-870663
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

Outline

- Nonnegative least-squares solver (NNLS) for algebraic constraints
- Reduction of NURBS patch quadrature rules
- Empirical quadrature procedure (EQP) for reduced order models
- Conclusions

Algebraic construction of sparse rules

- Numerical quadrature rules are usually applied element-wise, with the form

$$\int_{\Omega} c(x) f(x)\, \mathrm{dx} = \sum_{K} \int_{K} c(x) f(x)\, \mathrm{dx} \approx \sum_{K} \sum_{i} w_i c(x_i) f(x_i)$$

- Quadrature rules are usually chosen to be exact for a prescribed polynomial order.
- In a subspace, sparser rules can maintain accuracy.
- Constraints for accurate integration of finitely many functions yields an algebraic system defining possibly sparser rules.
- An algebraic method can be used for any basis type and quadrature type.

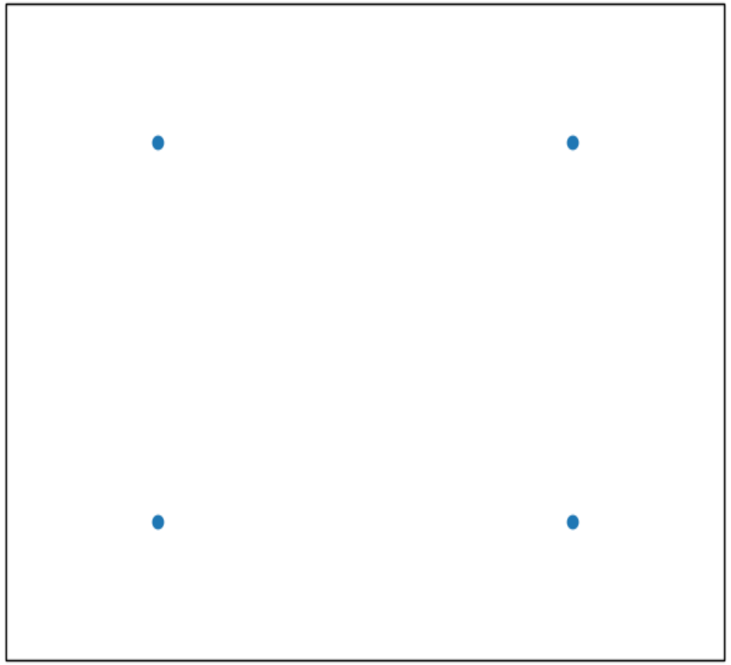
Basic example of a quadrature rule

- For a Lagrange element of order 1 on a rectangular element, `DiffusionIntegrator` uses a tensor-product rule exact for order 2 polynomials.

- For higher order, more quadrature points are used.

```
void QuadratureFunctions1D::GaussLegendre(const int np, IntegrationRule* ir)
{
   ir->SetSize(np);
   ir->SetPointIndices();
   ir->SetOrder(2*np - 1);

   switch (np)
   {
      case 1:
         ir->IntPoint(0).Set1w(0.5, 1.0);
         return;
      case 2:
         ir->IntPoint(0).Set1w(0.21132486540518711775, 0.5);
         ir->IntPoint(1).Set1w(0.78867513459481288225, 0.5);
```



Basic example of reduced quadrature for a subspace

- If we constrain the order 2 rule only to integrate constants, a single point suffices with changed weight, e.g.: $p = (a, a)$, $w = 1$, $a = 0.21132486540518711775$.

- The under-determined constraint system would be

$$Cw = b, \qquad C = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}, \qquad w = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{bmatrix}^T, \qquad b = [1]$$

$$\tilde{w} = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}$$

- For a single basis function product x^2, a single point still works:

$$Cw = b, \qquad C = \begin{bmatrix} a^2 & a^2 & (1-a)^2 & (1-a)^2 \end{bmatrix}, \qquad w = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{bmatrix}^T, \qquad b = \begin{bmatrix} \frac{1}{3} \end{bmatrix}$$

$$\tilde{w} = \begin{bmatrix} \frac{1}{3a^2} & 0 & 0 & 0 \end{bmatrix}$$

$$\tilde{w} = \begin{bmatrix} 0 & 0 & \frac{1}{3(1-a)^2} & 0 \end{bmatrix}$$

- For a single linear combination of basis functions, a single point still works.

Nonnegative Least Squares (NNLS) Solver

- We solve an under-determined constraint system for weights.
- For more details, see Lawson and Hanson 1974 or Adrian Humphry's thesis, 2023.
- Weights can be negative in quadrature rules, e.g. MFEM triangle rule for degree 3.
- For positive definite forms, nonnegativity preserves the positive definiteness.
- Linear programming has also been used by Masayuki Yano, but NNLS has been more successful.
- MFEM has a serial implementation. libROM has a parallel version, using SCALAPACK.

Algorithm 1: Non-negative Least Squares

Input: $\mathbf{A} \in \mathbb{R}^{m \times K^h}, \mathbf{b} \in \mathbb{R}^m, \delta \in \mathbb{R}^{K^h}$

Output: $\rho \in \mathbb{R}^{K^h}$

$\mathcal{Z} \leftarrow \{1, 2, \ldots, K^h\}$, $\mathcal{P} \leftarrow \emptyset$, $\rho \leftarrow \mathbf{0}$;

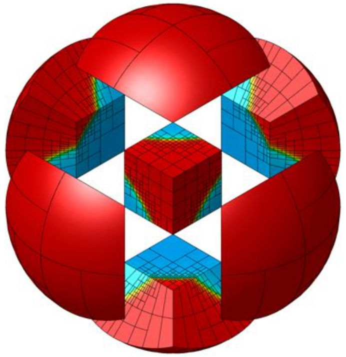
while $|\mathbf{A}\rho - \mathbf{b}| \not< \delta$ **do**

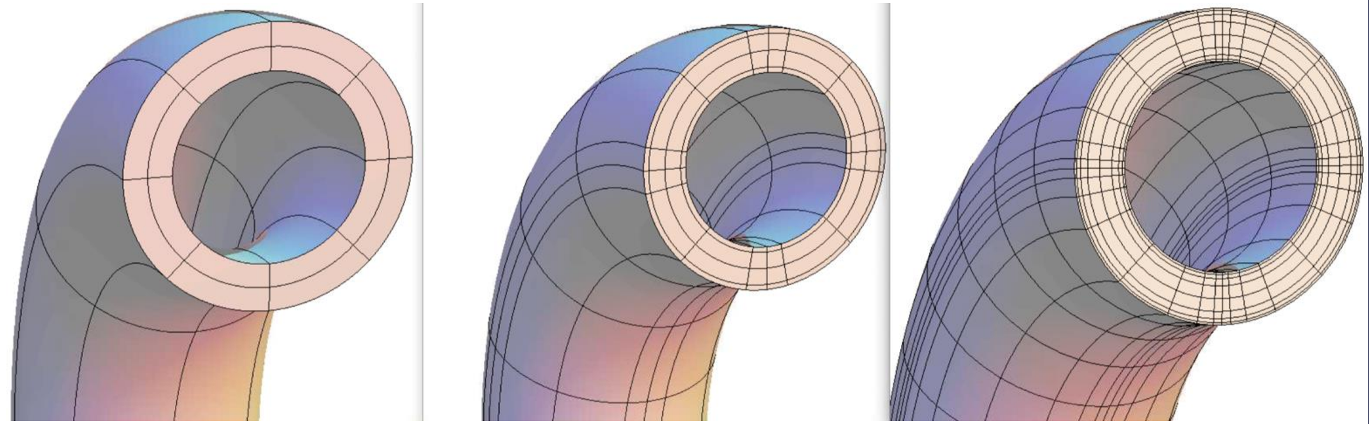
- $\mathbf{r} \leftarrow \mathbf{b} - \mathbf{A}\rho$;
- $\lambda \leftarrow \mathbf{A}^{\mathbf{T}}\mathbf{r}$;
- $i \leftarrow \arg\max_{i \in \mathcal{Z}}(\lambda_i)$;
- $\mathcal{Z} \leftarrow \mathcal{Z} \backslash \{i\}$;
- $\mathcal{P} \leftarrow \mathcal{P} \cup \{i\}$;
- $\tilde{\rho} \leftarrow \arg\min_{\mathbf{x} \in \mathbb{R}^{|\mathcal{P}|}} \|\mathbf{A}_{\mathcal{P}}\mathbf{x} - \mathbf{b}\|_2$;
- **if** *any* $\tilde{\rho} < 0$ **then**
  - $a \leftarrow \min_{i \in \mathcal{P}: \tilde{\rho}_i < 0}\{\rho_i / (\rho_i - \tilde{\rho}_i)\}$;
  - $\tilde{\rho} \leftarrow \tilde{\rho} + a(\rho - \tilde{\rho})$;
  - **for** $i \in \mathcal{P}$ **do**
    - **if** $\tilde{\rho}_i = 0$ **then**
      - $\mathcal{Z} \leftarrow \mathcal{Z} \cup \{i\}$;
      - $\mathcal{P} \leftarrow \mathcal{P} \backslash \{i\}$;
    - **end**
  - **end**
- **end**
- $\rho \leftarrow \tilde{\rho}$;

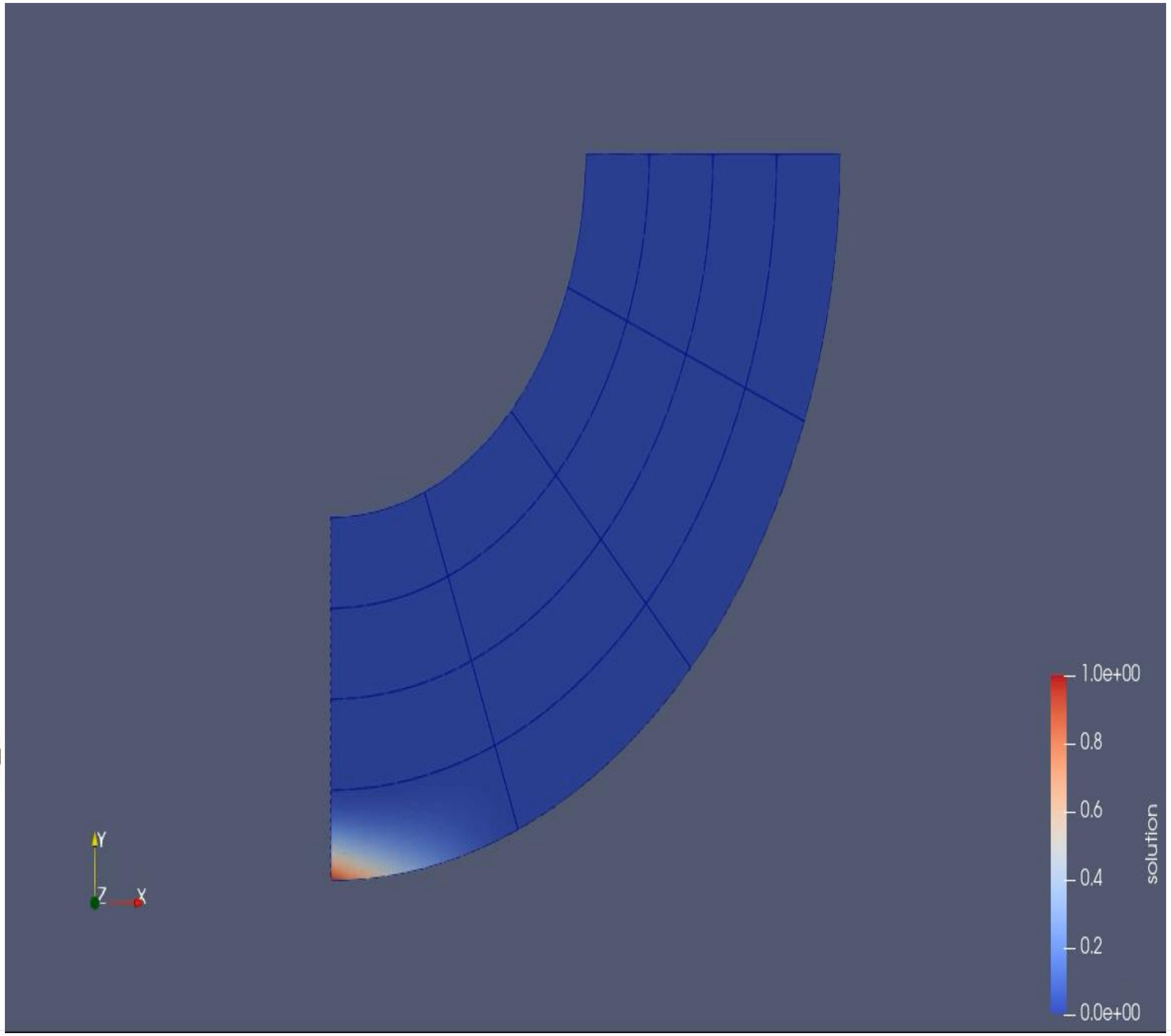
end

NURBS patches in isogeometric analysis (IGA)

- Curved geometry
- High order, tensor product bases
- Any number of elements
- Spacing formulas

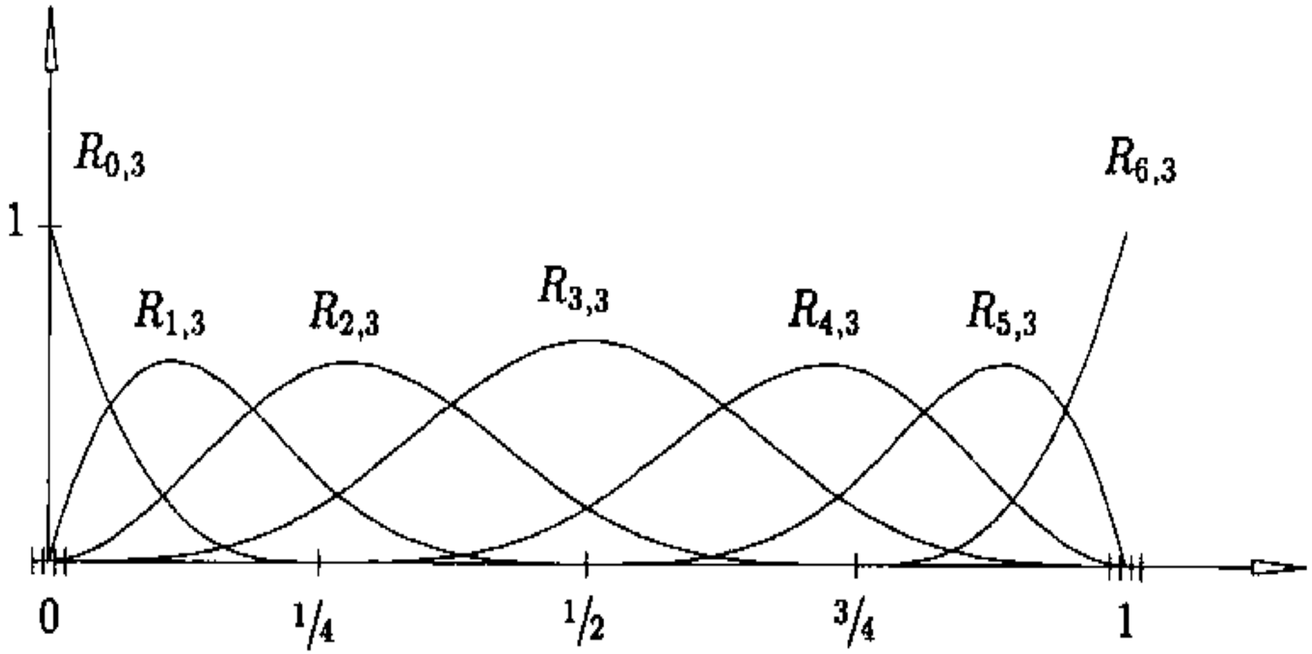




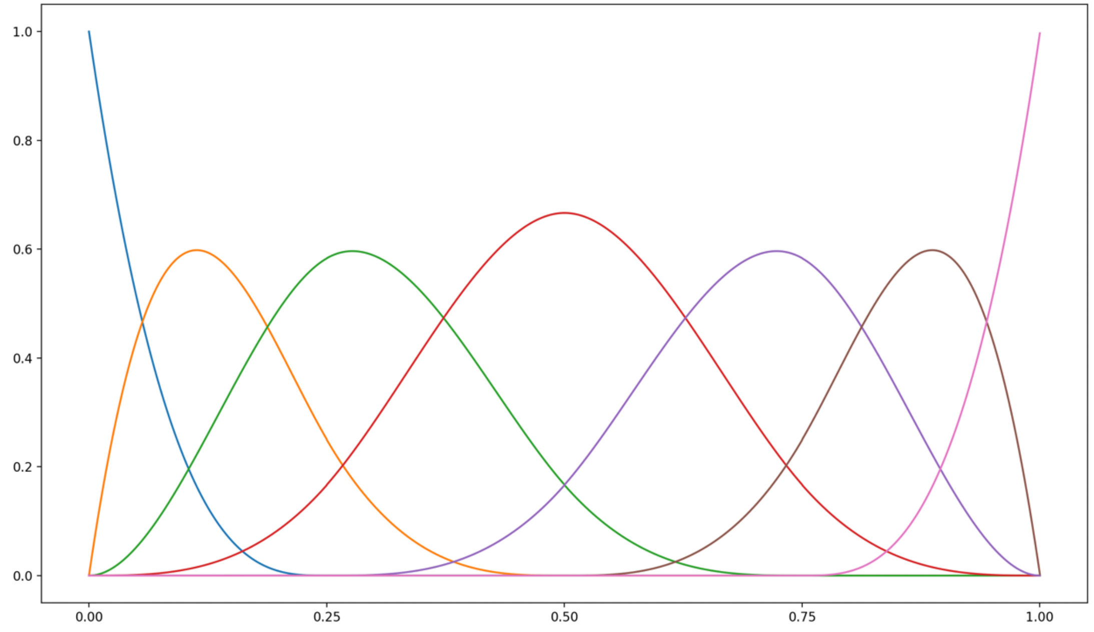


NURBS basis functions are tensor products of 1D functions

- Example of degree 3 NURBS with 4 uniform elements in 1D.
- Basis functions span multiple elements, with limited interaction.

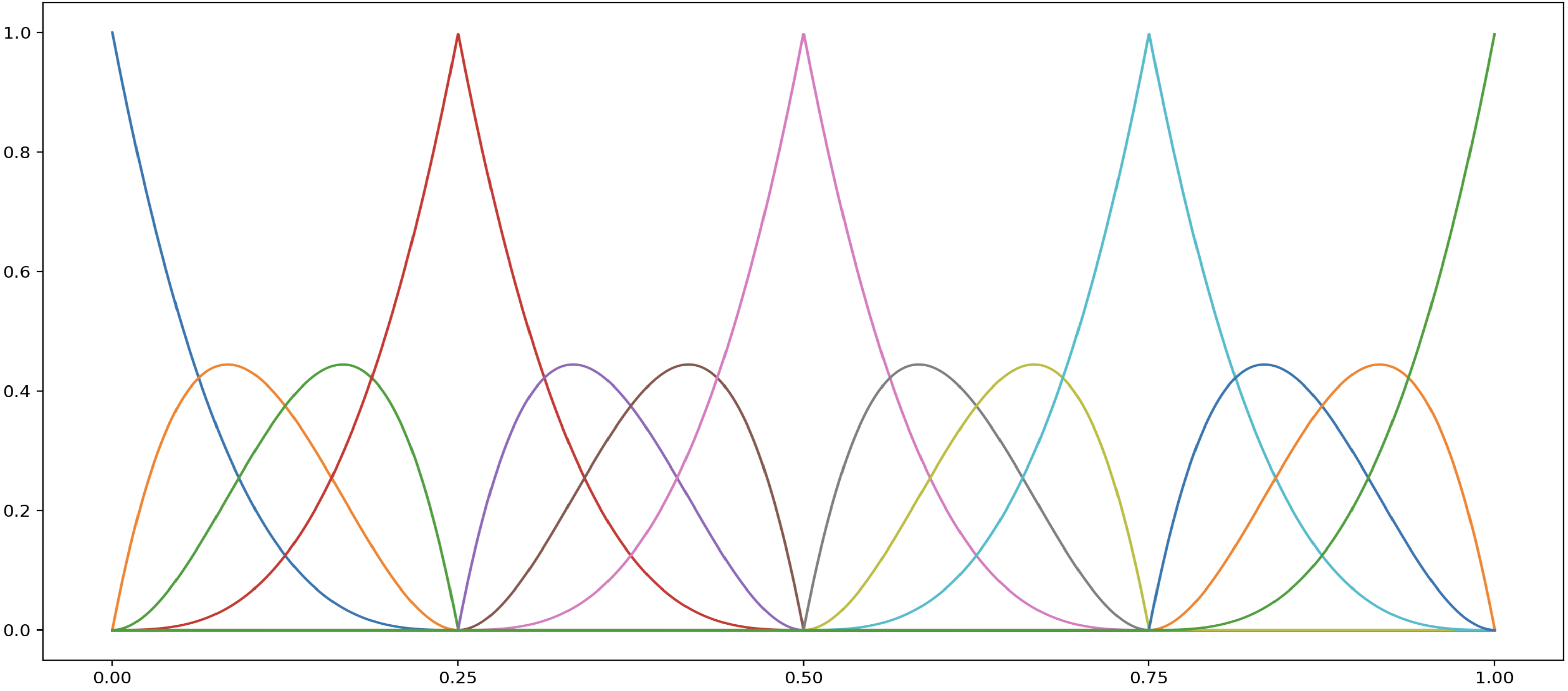


The NURBS Book, Fig. 4.3(a)

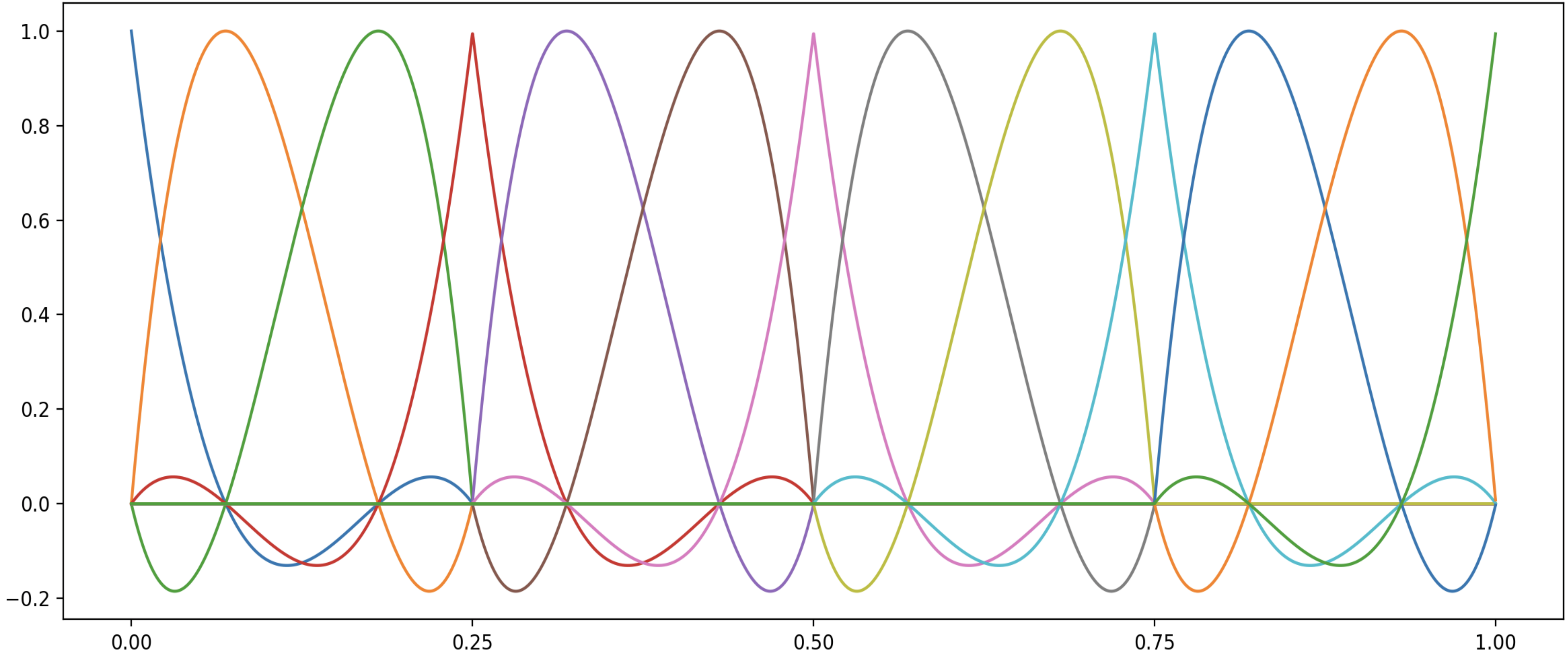


MFEM reproduction

1D order 3 NURBS basis on 4 elements



1D order 3 Lagrange basis on 4 elements



NURBS patches in isogeometric analysis (IGA)

- Joint work with Derek Thomas and Coreform LLC, 2022 SBIR Phase I DE-SC0022458.
- Instead of element-wise quadrature, we can use fewer points with patch-wise quadrature.
- MFEM PR 3088 introduced patch-wise integration in 2023.
- Example: mfem/miniapps/nurbs/nurbs_patch_ex1.cpp
- MFEM now supports:
  1. Patch-wise matrix assembly or partial assembly using exact integration, exploiting sum factorization and limited interactions of basis functions.
  2. Patch-wise matrix assembly with sum factorization and reduced rules in each dimension.

Diffusion integrator with sum factorization

Poisson problem *in variational form*

$$Find\ u \in Q_p \subset \mathcal{H}^1_0\ \ s.t.\ \ \forall v \in Q_p,$$

$$\int_\Omega \nabla u \cdot \nabla v = \int_\Omega f v$$

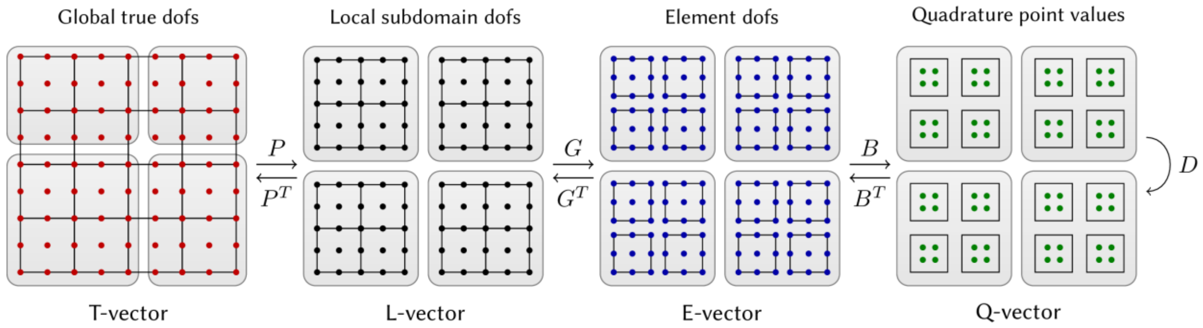
Stiffness matrix *(unit coefficient)*

$$\underbrace{\int_\Omega \nabla\varphi_i \nabla\varphi_j}_{A_{ij}} = \sum_E \int_E \nabla\varphi_i \nabla\varphi_j$$

$$= \sum_E \sum_k \alpha_k\, J_E^{-1}(q_k)\hat\nabla\hat\varphi_i(q_k)\, J_E^{-1}(q_k)\hat\nabla\hat\varphi_j(q_k)\,|J_E(q_k)|$$

$$= \underbrace{\sum_E}_{G,G^T} \sum_k \underbrace{\hat\nabla\hat\varphi_i(q_k)}_{(B^T)_{ik}} \underbrace{\left(\alpha_k J_E^{-T}(q_k) J_E^{-1}(q_k)|J_E(q_k)|\right)}_{D_{kk}} \underbrace{\hat\nabla\hat\varphi_j(q_k)}_{B_{kj}}$$

$$\boxed{A = P^T G^T B^T D B G P}$$



Applying the operator to a vector can be done in *O(p^{d+1})* operations instead of *O(p^{2d})*:

$$B_{kj} = B^{1d}_{k_1 j_1} B^{1d}_{k_2 j_2} \qquad k = (k_1, k_2),\ \ j = (j_1, j_2)$$

$$q_k = \sum_j B_{kj} u_j = \sum_{j_2} B^{1d}_{k_2 j_2} \left\{ \sum_{j_1} B^{1d}_{k_1 j_1} u_{j_1 j_2} \right\}$$

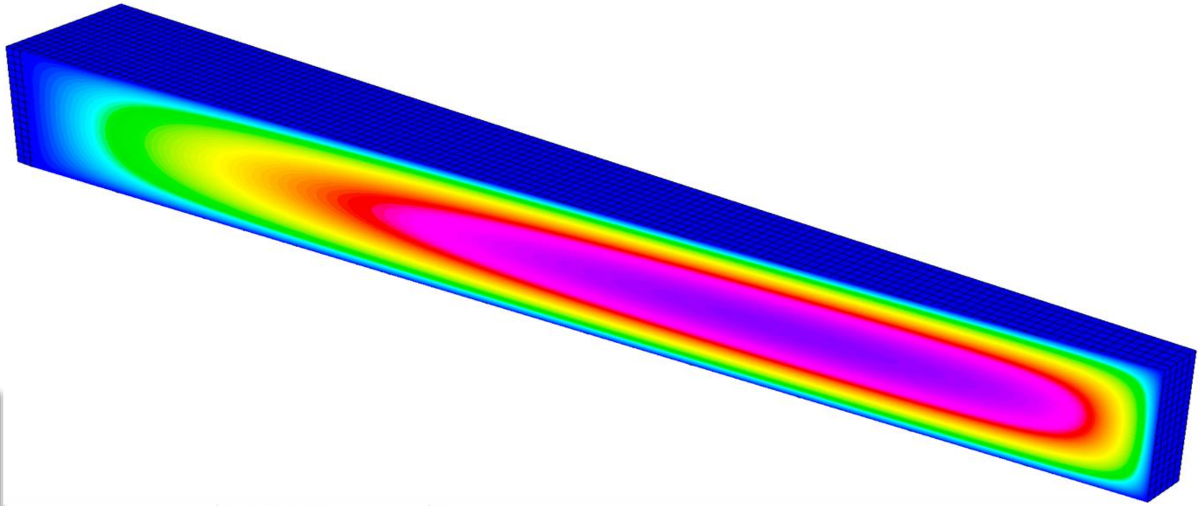
NURBS patches in isogeometric analysis (IGA)

- NURBS spaces may have fewer basis functions per patch than Lagrange elements.
- Instead of element-wise quadrature, we can use fewer points with patch-wise quadrature.
- Hiemstra 2019[1] presents methods for efficient assembly with NURBS bases, using sum factorization, but we additionally use NNLS for reduction of rules.
- In each dimension of the tensor product, products of 1D basis functions or their gradients are integrated.
- A reduced rule can be computed on the fly in each dimension of each patch, for each row (basis function) in matrix assembly.

1. Hiemstra, Sangalli, Tani, Calabro, Hughes, *Fast formation and assembly of finite element matrices with application to isogeometric linear elasticity*. Comput. Methods Appl. Mech. Engrg. 355, 2019, 234 - 260.

Miniapp nurbs_patch_ex1

- Serial CPU experiments.
- 61200 DOFs, 128x16x16 mesh elements (order 4), 2 patches.
- ”Ground truth” element-wise rule for order 8 = 2 * 4.
- Performance gains increase with FE order.



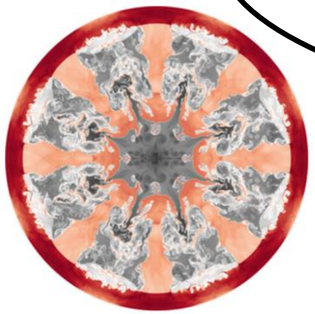
Assembly type	Setup time (s)	Relative l2 error
Element-wise matrix	274	0
Patch-wise matrix, full rules (optimized)	50	2.01023e-13
Patch-wise matrix, reduced rules	32	4.0273e-12
Patch-wise PA, full rules (optimized)	14	3.22942e-09

0.118463	0
0.239314	0
0.284444	0.280177
0.239314	0.496372
0.118463	0
0.118463	0
0.239314	0.376844
0.284444	0
0.239314	0.712754
0.118463	0
0.118463	0
0.239314	0
0.284444	0.690239
0.239314	0
0.118463	0
0.118463	0.38905
0.239314	0.0568541
0.284444	0.467273
0.239314	0
0.118463	0

Reduced order modeling (ROM)

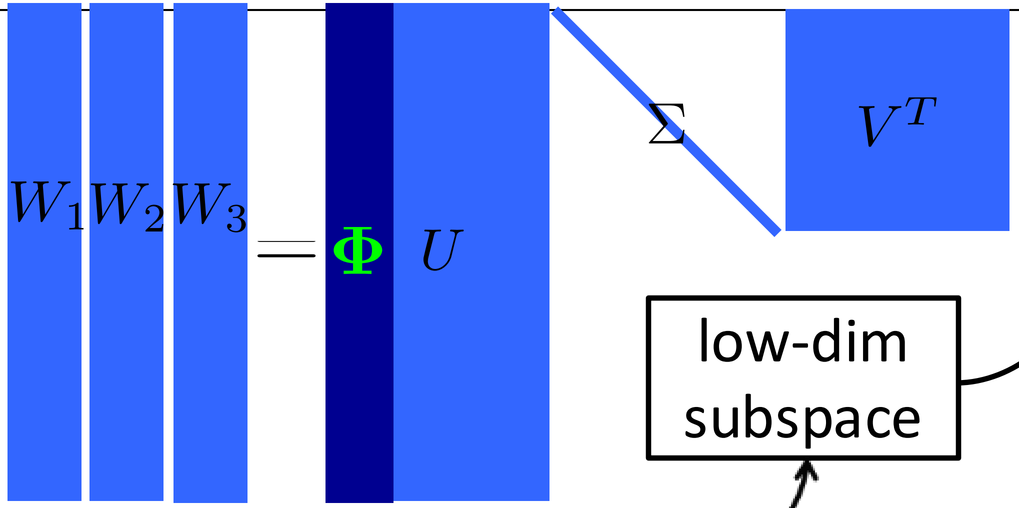
Goal: accelerate physics simulations without losing much accuracy by exploiting data and governing equations [**Data-driven and projection-based**].

high fidelity models / experiments



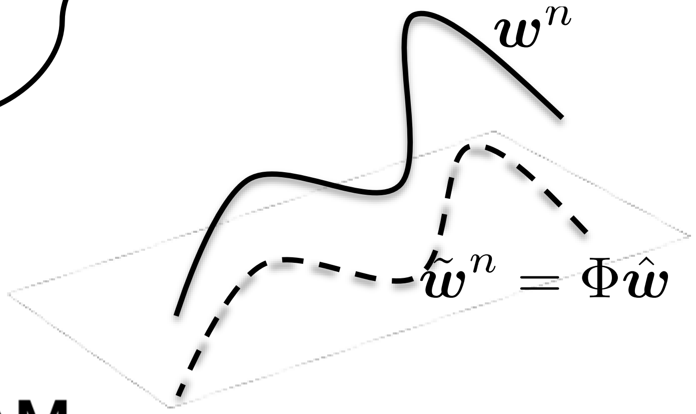
ICF-like Implosion (BLAST)

collect & compress data



low-dim subspace

project & propagate ***governing equation*** in time



approximate system solution

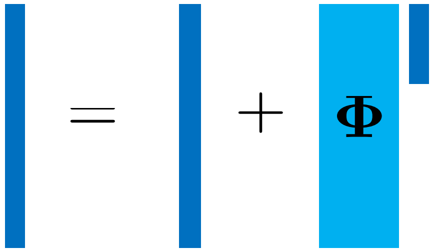
libROM
LIBRARY FOR REDUCED ORDER MODELS

Open source libROM: https://github.com/LLNL/libROM
https://github.com/LLNL/pylibROM

Projection-based (POD) reduced order modeling

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

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



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

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

Empirical quadrature procedure (EQP)

- Previous work on EQP includes:
  - Doug James et al: Optimizing cubature for efficient integration of subspace deformations, 2008.
  - ECSW by Charbel Farhat et al.
  - DG formulations by Masayuki Yano et al.

- EQP produces a sparse quadrature rule on a small number of elements.

- Integration of nonlinear terms is of the form

$$\int_\Omega N(u,\mu)\cdot L(v)\,dx \approx \tilde{r}_h(u,v,\mu) := \sum_K \sum_{i=1}^{m} w_{K,i} N(u,\mu)(x_{K,i})\cdot L(v)(x_{K,i})$$

for a linear operator $L(v)$ of test functions and nonlinear operator $N(u,\mu)$ of trial functions.

Empirical quadrature procedure (EQP)

- EQP aims to find a sparse global quadrature rule with minimal *M*, satisfying

$$\tilde{r}_h(u_i, v_j, \mu_i) \approx \sum_{m=1}^{M} w_m N(u_i, \mu_i)(x_m) \cdot L(v_j)(x_m)$$

  for all snapshots u_i (with parameter μ_i) and reduced basis vectors v_j.

- Defining the matrix with entries

$$A_{pm} = N(u_i, \mu_i)(x_m) \cdot L(v_j)(x_m), \qquad p = in_s + j$$

  the sparse rule is represented by the solution r, with M nonzeros, of

$$Ar = Af \qquad f = [w_{K,i}]$$

- This underdetermined system is solved approximately by nonnegative least squares (NNLS, Lawson and Hanson 1974).

Lagrangian hydrodynamics in Laghos miniapp

- Semi-discrete Lagrangian nonlinear conservation laws

Momentum conservation: $$\boldsymbol{M}_{\mathcal{V}} \frac{d\boldsymbol{v}}{dt} = -\boldsymbol{F} \cdot \boldsymbol{1}$$

Energy conservation: $$\boldsymbol{M}_{\mathcal{E}} \frac{d\boldsymbol{e}}{dt} = -\boldsymbol{F}^T \cdot \boldsymbol{v}$$

Equation of motion: $$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{v}$$

- Explicit time integration with adaptive time-stepping: RK4, RK2-Average.
- Mesh nodes move with the position variable $\boldsymbol{x}$.
- Laghos (high-order FEM Lagrangian hydrodynamics MFEM miniapp) is used for full-order simulation.

ROM formulation for Laghos

- Substitute bases for each variable, then use Galerkin projection:

Momentum conservation: $$\hat{\boldsymbol{M}}_{\mathcal{V}} \frac{d\hat{\boldsymbol{v}}}{dt} = -\boldsymbol{\Phi}_v^T \boldsymbol{F} \cdot \mathbf{1}$$

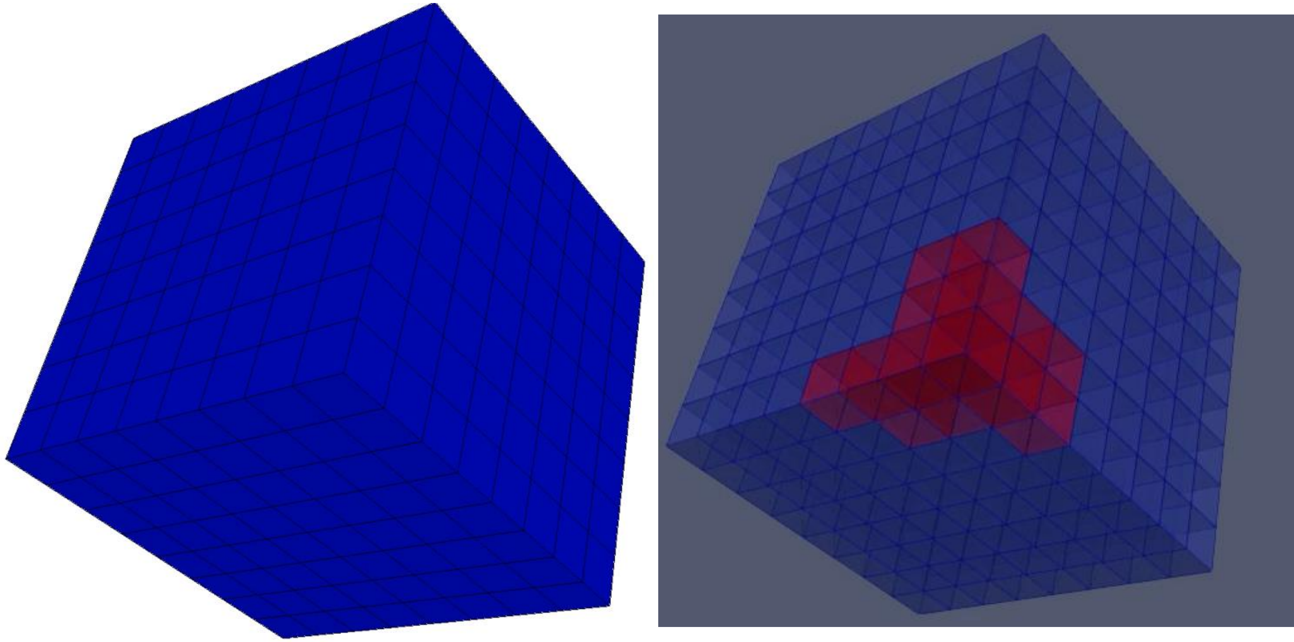
Energy conservation: $$\hat{\boldsymbol{M}}_{\mathcal{E}} \frac{d\hat{\boldsymbol{e}}}{dt} = -\boldsymbol{\Phi}_e^T \boldsymbol{F}^T \cdot \boldsymbol{v}$$

Equation of motion: $$\frac{d\hat{\boldsymbol{x}}}{dt} = \boldsymbol{\Phi}_x^T \boldsymbol{\Phi}_v \hat{\boldsymbol{v}}$$

  - Reduced mass matrices: $\hat{\boldsymbol{M}}_{\mathcal{V}} \equiv \boldsymbol{\Phi}_v^T \boldsymbol{M}_{\mathcal{V}} \boldsymbol{\Phi}_v, \quad \hat{\boldsymbol{M}}_{\mathcal{E}} \equiv \boldsymbol{\Phi}_e^T \boldsymbol{M}_{\mathcal{E}} \boldsymbol{\Phi}_e$
- Time-windowing is used to limit the reduced bases and quadrature rules.
- Separate rules are computed for momentum and energy equation RHS.

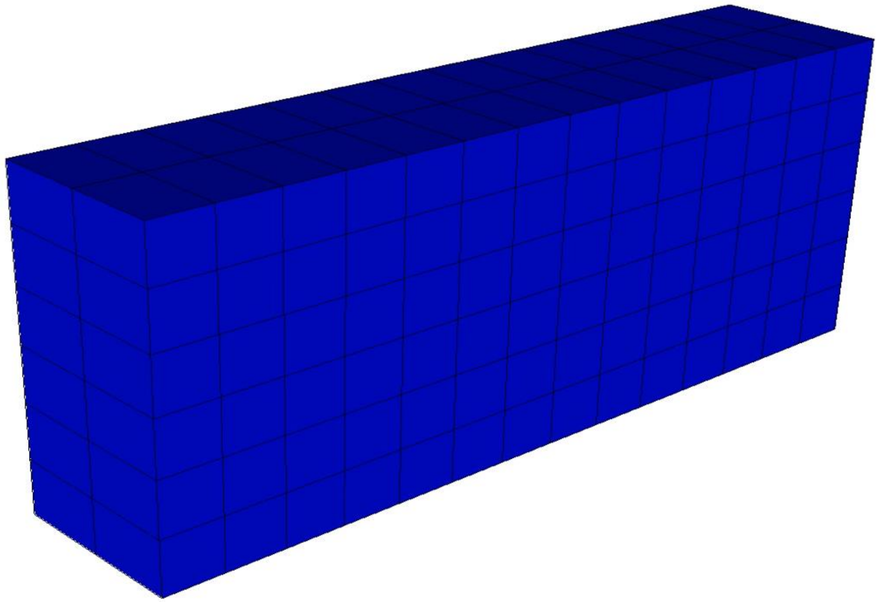
Sedov blast example

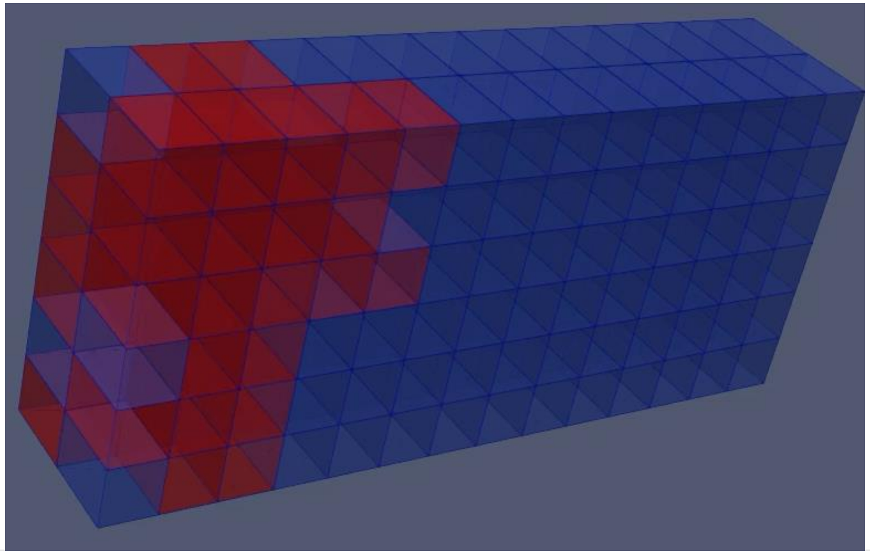
- Reproductive testing (training data).
- Mesh is 8x8x8, with 64 = 4^3 quadrature points per element, FE orders (2,1).
- Average (over 49 time windows) reduced number of points: 94 / 32768
- ROM-EQP relative L2 error: 3.77326e-05
- ROM relative L2 error: 2.71326e-05
- FOM run time: 139 s
- ROM run time: 3.91 s
- Speedup factor: 36



Triple point example, low order

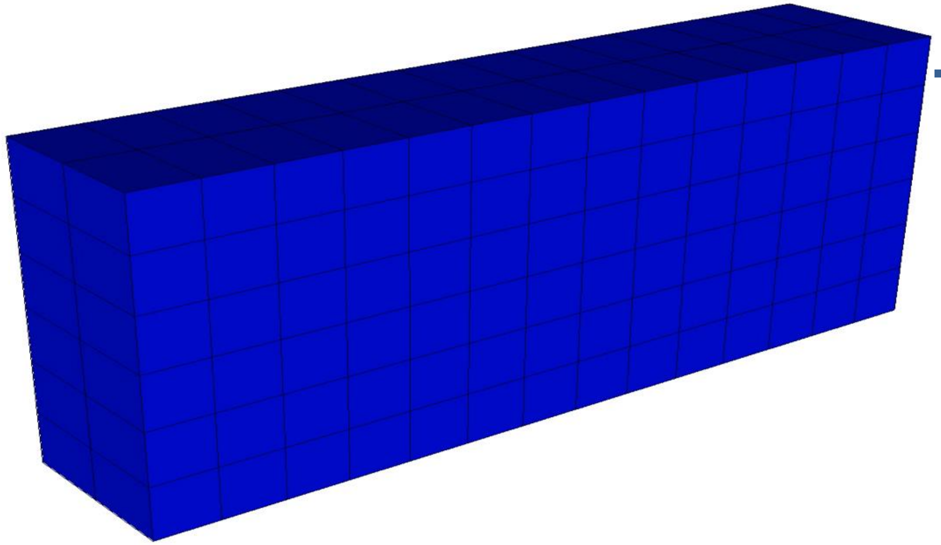
- Reproductive testing (training data).
- Mesh is 14x6x2 = 168, with 64 points per element, FE orders (2, 1).
- Average (over 66 time windows) reduced number of points: 45 / 10752
- ROM-EQP relative L2 error: 5.28973e-06
- ROM relative L2 error: 2.14006e-06
- FOM run time: 57.6 s
- ROM run time: 6.46 s
- Speedup factor: 8.9

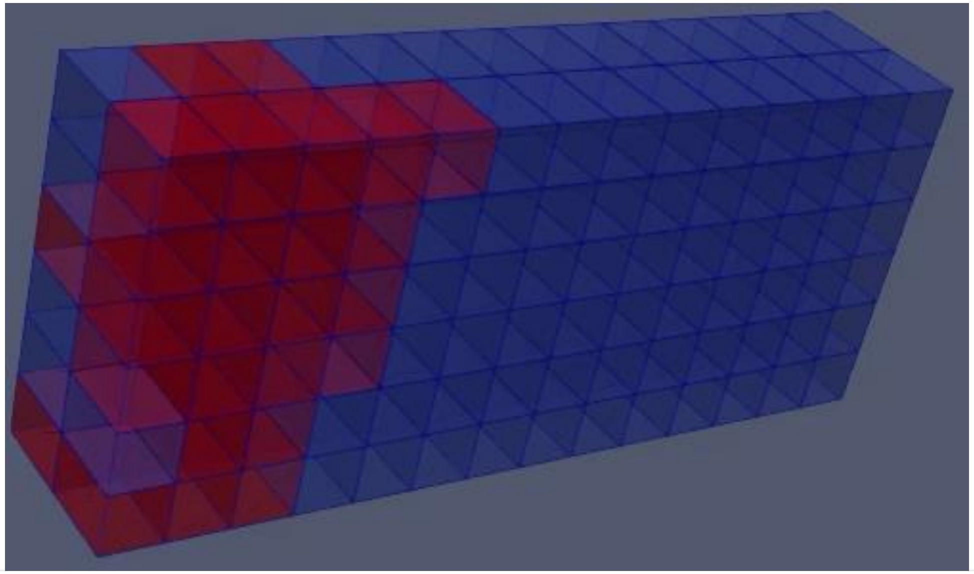




Triple point example, higher order

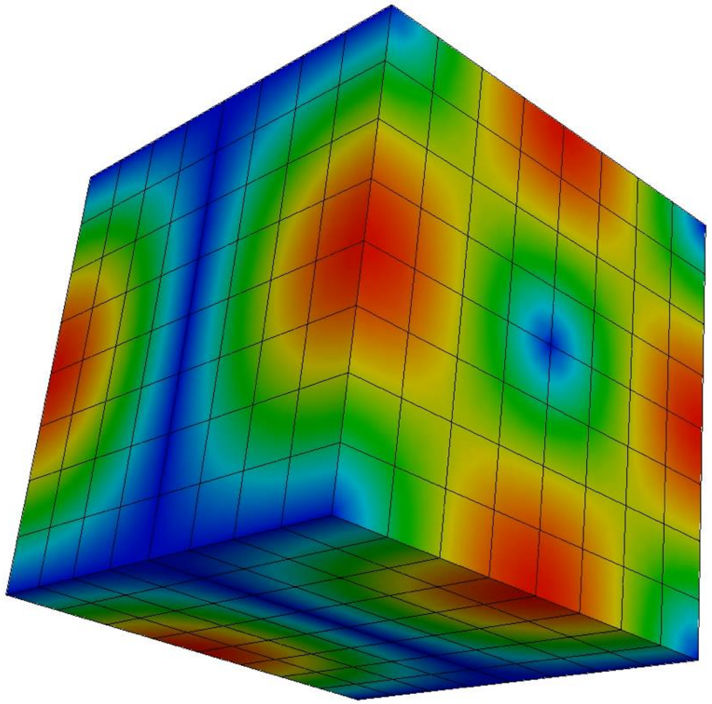
- Reproductive testing (training data).
- Mesh is 14x6x2 = 168, with 216 = 6^3 points per element, FE orders (3, 2).
- Average (over 136 time windows) reduced number of points: 44 / 36288
- ROM-EQP relative L2 error: 6.94712e-05
- FOM run time: 365 s
- ROM run time: 16.9 s
- Speedup factor: 22

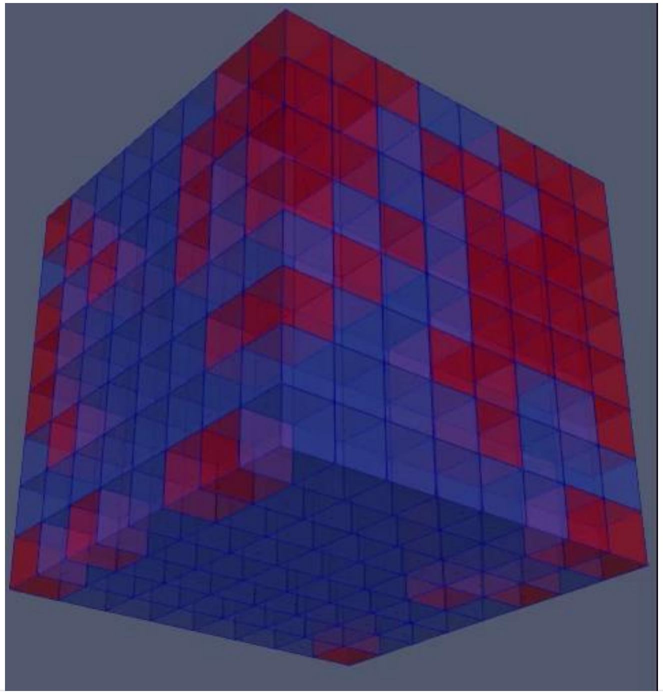




Taylor-Green example

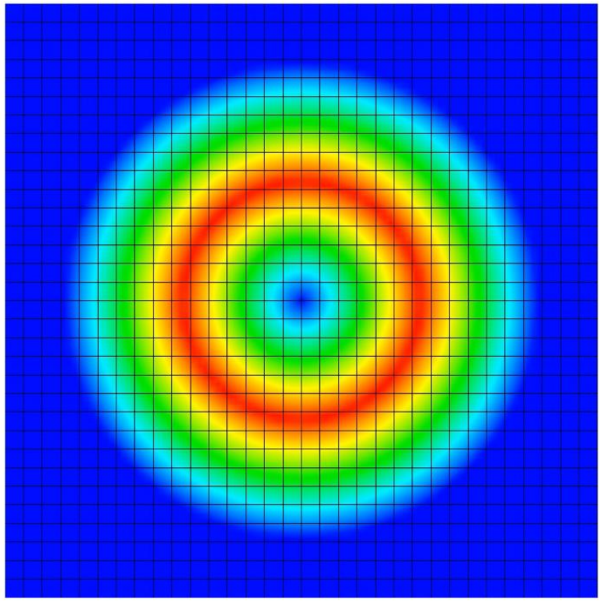
- Mesh is 8x8x8, with 64 quadrature points per element, FE orders (2,1).
- Average (over 34 time windows) reduced number of points: 65 / 32768
- ROM-EQP rel. L2 error: 2.13665e-06
- ROM rel. L2 error: 1.03051e-06
- FOM run time: 16.3 s
- ROM run time: 2.58 s
- Speedup factor: 6.3
- Reproductive testing (training data).

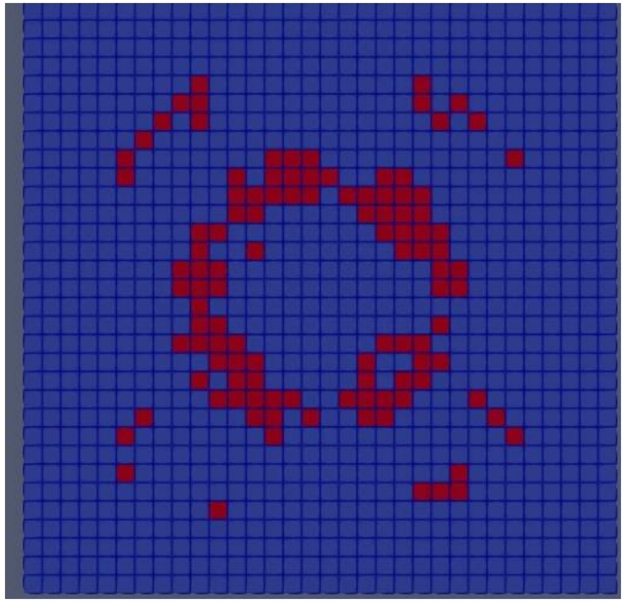




Gresho vortices example

- Reproductive testing (training data).
- Mesh is 32x32, with 36 points per element, FE orders (3, 2).
- Average (over 59 time windows) reduced number of points: 63 / 36864
- ROM-EQP rel. L2 error: 4.16793e-05
- ROM rel. L2 error: 1.5083e-05
- FOM run time: 16.8 s
- ROM run time: 6.62 s
- Speedup factor: 2.5





Energy-conserving EQP for Laghos

- Joint work with Siu Wun (Tony) Cheung and Chris Vales (Dartmouth).
- The assumptions are imposed as constraints in the ROM simulation.

Theorem 1. *Assume that*

1. $\boldsymbol{v}_{os} = \mathbf{0}_{\mathcal{V}}$;
2. $\mathbf{1}_{\mathcal{E}} = \boldsymbol{\Phi}_e \widehat{\mathbf{1}}_{\mathcal{E}}$ *for some* $\widehat{\mathbf{1}}_{\mathcal{E}} \in \mathbb{R}^{n_e}$;
3. $\widehat{\boldsymbol{v}}^T \cdot \boldsymbol{\Phi}_v^T \boldsymbol{M}_{\mathcal{V}}^{-1} \boldsymbol{F}(\boldsymbol{\Phi}\widehat{\boldsymbol{w}}) \cdot \mathbf{1}_{\mathcal{E}} = \widehat{\mathbf{1}}_{\mathcal{E}}^T \cdot \boldsymbol{\Phi}_e^T \boldsymbol{M}_{\mathcal{E}}^{-1} \boldsymbol{F}(\boldsymbol{\Phi}\widehat{\boldsymbol{w}})^T \cdot \boldsymbol{\Phi}_v \widehat{\boldsymbol{v}}$
   for all $\widehat{\boldsymbol{v}} \in \mathbb{R}^{n_v}$ *and* $\widehat{\boldsymbol{w}} \in \mathbb{R}^{n}$.

Then the discrete hyperreduced model with RK2-average time integration conserves the discrete total energy:

$$IE(\tilde{\boldsymbol{w}}_{k+1}) + KE(\tilde{\boldsymbol{w}}_{k+1}) = IE(\tilde{\boldsymbol{w}}_k) + KE(\tilde{\boldsymbol{w}}_k)$$

for all $k \geq 0$.

Energy-conserving EQP for Laghos

- Reproductive testing (training data), ~same reduced basis for basic EQP and CEQP.
- Energy is conserved to machine precision, unlike the basic EQP.

Problem	Sedov	Gresho	Triple point	Taylor-Green
t_f	0.3	0.4	0.8	0.4
N_w	91	213	39	418
ΔE_{EQP}	4.23e-04	4.93e-06	1.15e-06	1.36e-10
ΔE_{CEQP}	6.44e-15	1.92e-14	4.88e-14	3.60e-14

Conclusions

- Computing reduced 1D rules for tensor product spaces is much faster than 3D rules.
- The algebraic formulation is generic, for any type of quadrature or finite element.
- Physical constraints can be imposed, even with reduced rules.
- Great performance gains can be attained without losing accuracy.

Open-source libraries

- https://github.com/mfem/mfem
- https://github.com/CEED/Laghos/tree/rom
- https://github.com/LLNL/libROM
- https://github.com/LLNL/pylibROM

librom.net

libROM Features Examples Documentation Gallery Download DDPS HLCS

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 (libROM, pylibROM)
- Dynamic mode decomposition (libROM, pylibROM)
- Projection-based reduced order models (libROM, pylibROM)
- Hyper-reduction (libROM, pylibROM)
- Greedy algorithm (libROM, pylibROM)
- Latent space dynamics identification (LaSDI, gLaSDI, GPLaSDI)
- Domain decomposition nonlinear manifold reduced order model (DD-NM-ROM)

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.

Recent News

Sep 1, 2024	tLaSDI paper is published in CMAME
Aug 1, 2024	S-OPT paper is published in SISC
July 13, 2024	Data-scarce paper is published in Shock Waves
July 8, 2024	Progressive paper is published in Scientific Reports
May 17, 2024	DD-NM-ROM open source code is available in gitHub
May 15, 2024	DDFEM paper is published in CMAME
Apr 29, 2024	wLaSDI paper is published in CMAME
Apr 10, 2024	Gappy AE paper is published in CMAME
Mar 25, 2024	DD-NM-ROM paper is published in CMAME
Mar 16, 2024	LaSDI book chapter arXiv paper is available

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
Sep. 11, 2022	Local reduced order models and interpolation-based parameterization
Sep. 23, 2022	Projection-based reduced order model for nonlinear system
Aug. 23, 2023	Complete derivation of dynamic mode decomposition

Thank you for your attention!

copeland11@llnl.gov