Adaptive *h*-refinement in nonlinear model reduction: capturing moving discontinuities

SAND2016-6866C

Kevin Carlberg

Sandia National Laboratories

Livermore, California

7th European Congress of Mathematics
TU-Berlin
July 21, 2016

Parameterized PDEs at Sandia

(Loading movie ...)

(Loading movie ...)

- CFD model
  - 100 million cells
  - 200,000 time steps
- High simulation costs
  - 6 weeks, 5000 cores
  - 6 runs maxes out Cielo

Parameterized PDEs at Sandia

(Loading movie ...)

(Loading movie ...)

- CFD model
  - 100 million cells
  - 200,000 time steps
- High simulation costs
  - 6 weeks, 5000 cores
  - 6 runs maxes out Cielo

Barrier

- Design engineers require faster simulations
- Uncertainty quantification

Parameterized PDEs at Sandia

(Loading movie ...) (Loading movie ...)

- CFD model
  - 100 million cells
  - 200,000 time steps
- High simulation costs
  - 6 weeks, 5000 cores
  - 6 runs maxes out Cielo

Barrier

- Design engineers require faster simulations
- Uncertainty quantification

Objective: break barrier

Reduced-order modeling: accuracy limitation

- Full-order model (FOM)

$$\boldsymbol{r}^k(\boldsymbol{x}^k; \boldsymbol{\mu}) = 0, \quad k = 1, \ldots, t \tag{1}$$

  - residual: $\boldsymbol{r}^k \in \mathbb{R}^n$
  - state: $\boldsymbol{x}^k \in \mathbb{R}^n$

Reduced-order modeling: accuracy limitation

- Full-order model (FOM)

$$\boldsymbol{r}^k(\boldsymbol{x}^k;\boldsymbol{\mu}) = 0, \quad k = 1,\ldots,t \tag{1}$$

  - residual: $\boldsymbol{r}^k \in \mathbb{R}^n$
  - state: $\boldsymbol{x}^k \in \mathbb{R}^n$

- Reduced-order model (ROM)
  1. *Offline*: construct low dimensional basis $\boldsymbol{V} \in \mathbb{R}^{n\times p}$ with $p \ll n$
  2. *Online*: approximate $\boldsymbol{x}^k \approx \boldsymbol{V}\hat{\boldsymbol{x}}^k$

$$\text{solve } \boldsymbol{V}^T\boldsymbol{r}^k(\boldsymbol{V}\hat{\boldsymbol{x}}^k;\boldsymbol{\mu}) = 0, \quad k = 1,\ldots,t \tag{2}$$

Additional approximations needed if $\boldsymbol{r}$ nonlinear or nonaffine

Reduced-order modeling: accuracy limitation

- Full-order model (FOM)

$$\boldsymbol{r}^k(\boldsymbol{x}^k;\boldsymbol{\mu}) = 0, \quad k = 1, \ldots, t \tag{1}$$

  - residual: $\boldsymbol{r}^k \in \mathbb{R}^n$
  - state: $\boldsymbol{x}^k \in \mathbb{R}^n$

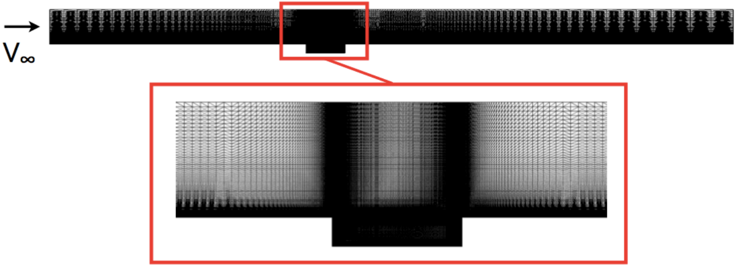
- Reduced-order model (ROM)
  1. *Offline*: construct low dimensional basis $\boldsymbol{V} \in \mathbb{R}^{n\times p}$ with $p \ll n$
  2. *Online*: approximate $\boldsymbol{x}^k \approx \boldsymbol{V}\hat{\boldsymbol{x}}^k$

solve $\boldsymbol{V}^T\boldsymbol{r}^k(\boldsymbol{V}\hat{\boldsymbol{x}}^k;\boldsymbol{\mu}) = 0, \quad k = 1, \ldots, t$ (2)

Additional approximations needed if $\boldsymbol{r}$ nonlinear or nonaffine

\+ ROMs are almost always *fast* ($p \ll n$).

Example: Cavity-flow problem (collaborators: M Barone, S Arunajatesan)



- FOM: Compressible Navier–Stokes
  - Nonlinear dynamical system
  - Re $= 6.3 \times 10^6$
  - $M_\infty = 0.6$
  - DES turbulence model
  - 3 point BDF integrator
  - 1.2 million degrees of freedom
- ROM: GNAT [?]
  - 179 degrees of freedom

When a ROM works

(Loading movie ...)

- FOM: 48 cpu x 5 hours
- ROM: 2 cpu x 31 min = 229X cpu savings

When a ROM works

(Loading movie ...)

- FOM: 48 cpu x 5 hours
- ROM: 2 cpu x 31 min = 229X cpu savings
- However, ROMs are not guaranteed to be *accurate*.

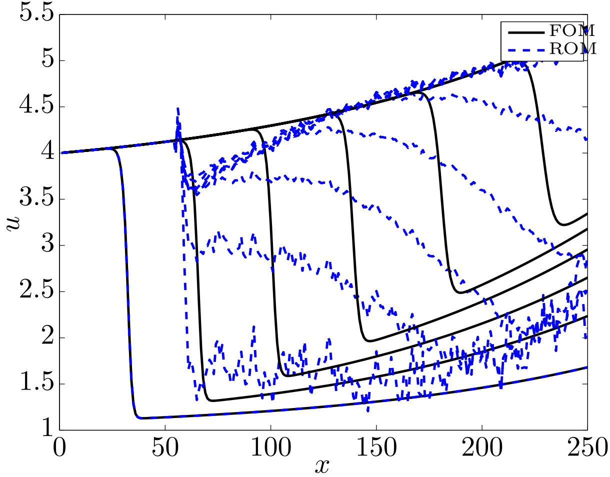
ROM accuracy is limited by the information in $\boldsymbol{V}$.

Example: inviscid Burgers equation [?]

$$\begin{aligned}\frac{\partial u(x,\tau)}{\partial \tau} + \frac{1}{2}\frac{\partial\left(u^2(x,\tau)\right)}{\partial x} &= 0.02e^{0.02x}\\ u(0,\tau) &= 3,\ \forall \tau > 0\\ u(x,0) &= 1,\ \forall x \in [0,\ 100],\end{aligned}$$

- Discretization: Godunov's scheme
- Simulate $\tau \in [0, 50]$
- FOM: 250 degrees of freedom
- ROM: 150 degrees of freedom
- $\boldsymbol{V}$ constructed via POD using snapshots in $\tau_{\mathrm{train}} \in [0, 2.5]$

ROM accuracy limited by relevance of training data



- ROM inaccurate when outside predictive domain of $\boldsymbol{V}$

Existing *a posteriori* ROM adaptation methods

- Revert to the FOM, solve it, and add solution to the basis [?, ?, ?]

\+ Improves the ROM *a posteriori*

\- Incurs large-scale operations

Existing *a posteriori* ROM adaptation methods

- Revert to the FOM, solve it, and add solution to the basis [?, ?, ?]

\+ Improves the ROM *a posteriori*

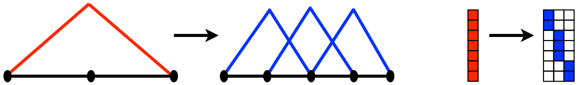
\- Incurs large-scale operations

Goal: *Cheap, a posteriori improvement of the ROM*

Main idea

ROM analog to mesh-adaptive h-refinement

- 'Split' basis vectors

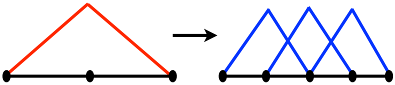


finite element h-refinement *ROM h-refinement*

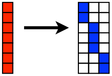
Main idea

ROM analog to mesh-adaptive h-refinement

- 'Split' basis vectors

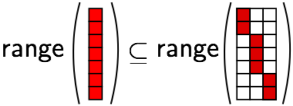


finite element h-refinement



ROM h-refinement

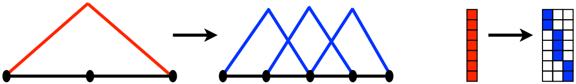
- Generate hierarchical subspaces



Main idea

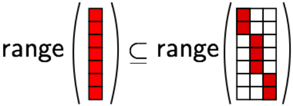
ROM analog to mesh-adaptive h-refinement

- 'Split' basis vectors

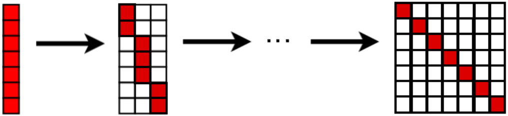


finite element h-refinement *ROM h-refinement*

- Generate hierarchical subspaces

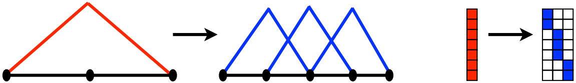


- ROM converges to the FOM



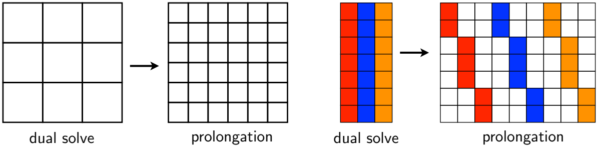
h-refinement ingredients

1. *Refinement*: split into basis vectors with different support



finite element h-refinement *ROM h-refinement*

2. *Error indicators*: a) dual solve (coarse), b) prolongation (fine)

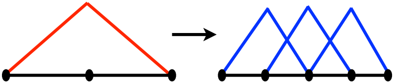


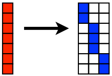
finite element h-refinement *ROM h-refinement*

3. *Adaptive algorithm*: refine vectors with large error indicators

Ingredient 1: Refinement

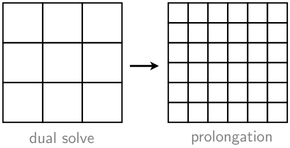
1. *Refinement*: split into basis vectors with different support

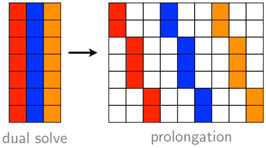




finite element h-refinement *ROM h-refinement*

2. *Error indicators*: a) dual solve (coarse), b) prolongation (fine)

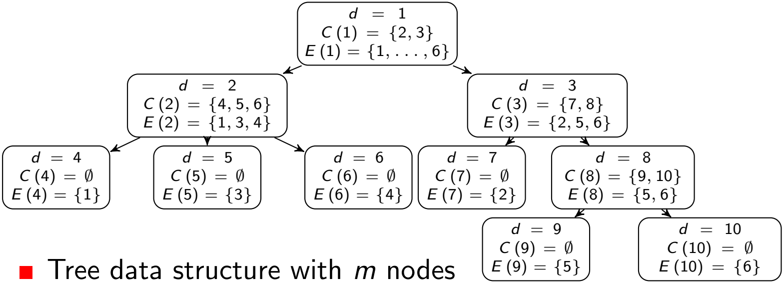




finite element h-refinement *ROM h-refinement*

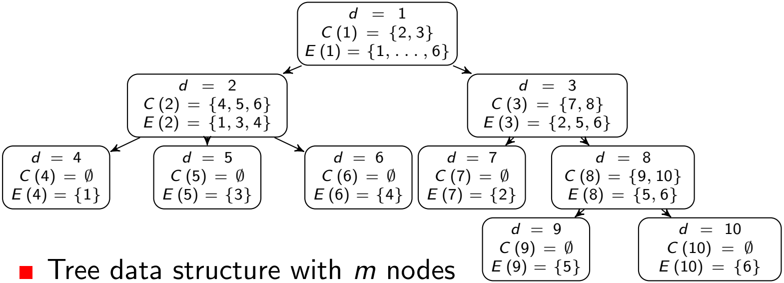
3. *Adaptive algorithm*: refine vectors with large error indicators

Tree structure



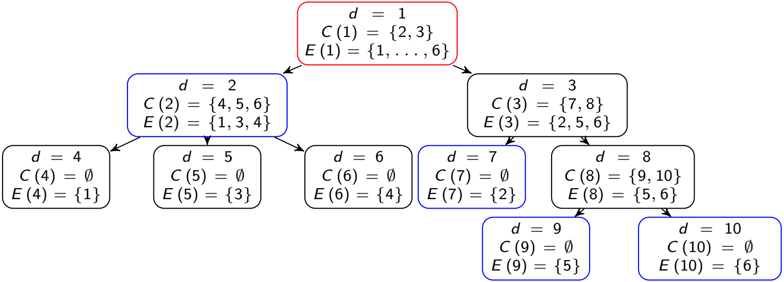
- Tree data structure with m nodes
  - *child* function $C : \mathbb{N}(m) \rightarrow \mathcal{P}(\mathbb{N}(m))$
  - *element* function $E : \mathbb{N}(m) \rightarrow \mathcal{P}(\mathbb{N}(n))$
  - each vector $\mathbf{v_i}$ assigned a node d_i, support $E(d_i)$, splits $C(d_i)$

Tree structure



- Tree data structure with m nodes
  - *child* function $C : \mathbb{N}(m) \to \mathcal{P}(\mathbb{N}(m))$
  - *element* function $E : \mathbb{N}(m) \to \mathcal{P}(\mathbb{N}(n))$
  - each vector $\mathbf{v_i}$ assigned a node d_i, support $E(d_i)$, splits $C(d_i)$
- Requirements
  1. Root node includes all elements $E(1)$
  2. Each element has a single leaf node
  3. Disjoint support of children $E(j) \cap E(k) = \emptyset, \ \forall j \neq k \in C(i)$
  4. $\cup_{j \in C(i)} E(j) = E(i)$
  - + **??–??** ensure the ROM converges to the FOM
  - + **??** ensures hierarchical refined subspaces

Tree example: $n = 6$



$$\boldsymbol{V}^{(0)} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{bmatrix} \quad \rightarrow \quad \boldsymbol{V} = \begin{bmatrix} v_1 & 0 & 0 & 0 \\ 0 & v_2 & 0 & 0 \\ v_3 & 0 & 0 & 0 \\ v_4 & 0 & 0 & 0 \\ 0 & 0 & v_5 & 0 \\ 0 & 0 & 0 & v_6 \end{bmatrix}.$$

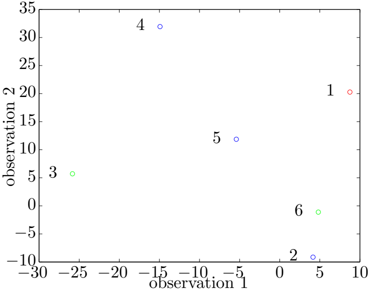
Tree construction

State variables that are strongly **correlated** *or* **anticorrelated** *should reside in the same tree node.*

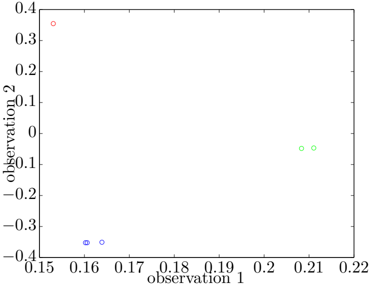
Tree construction

State variables that are strongly **correlated** *or* **anticorrelated** *should reside in the same tree node.*

1. Normalize state-variable observation history
2. If first observation is negative, flip over origin
3. Recursively apply **k-means clustering**



(a) before modification

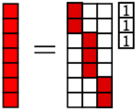


(b) after modification

Figure: State-variable observation history (variable index labeled).

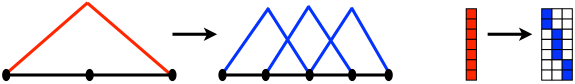
Refinement machinery

$$\boldsymbol{V}^H = \boldsymbol{V}^h \boldsymbol{I}_H^h$$



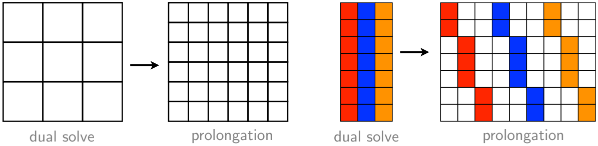
- coarse basis $\boldsymbol{V}^H \in \mathbb{R}^{n\times p}$ $\boldsymbol{V}^{(\mathbf{0})})$
- fine basis $\boldsymbol{V}^h \in \mathbb{R}^{n\times q}$ with $q = \sum_{i=1}^{p} \mathrm{card}\left(C\left(d_i\right)\right)$
- prolongation operator $\boldsymbol{I}_H^h \in \{0,1\}^{q\times p}$
- prolongated generalized coordinates $\hat{\boldsymbol{x}}_H^h = \boldsymbol{I}_H^h \hat{\boldsymbol{x}}^H$
- restriction operator $\boldsymbol{I}_h^H = \left(\boldsymbol{I}_H^h\right)^+$

- **Main idea**: ROM analog to mesh-adaptive *h*-refinement
  1. *Refinement*: split into basis vectors with different support



finite element h-refinement *ROM h-refinement*

  2. *Error indicators*: a) dual solve (coarse), b) prolongation (fine)



finite element h-refinement *ROM h-refinement*

  3. *Adaptive algorithm*: refine vectors with large error indicators

Dual-weighted residual error indicators

- **Goal-oriented**: reduce the error in output $z(\boldsymbol{x})$
- Analogous to duality-based error control for
  - differential equations [?, ?]
  - finite elements [?, ?, ?, ?, ?, ?],
  - finite volumes [?, ?, ?, ?]
  - discontinuous Galerkin methods [?, ?]

Dual-weighted residual error indicators

- Approximate fine output:

$$z(\boldsymbol{V}^h\hat{\boldsymbol{x}}^h) \approx z(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H) + \frac{\partial z}{\partial \boldsymbol{x}}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)\boldsymbol{V}^h(\hat{\boldsymbol{x}}^h - \boldsymbol{I}_H^h\hat{\boldsymbol{x}}^H) \quad (3)$$

Dual-weighted residual error indicators

- Approximate fine output:

$$z(\boldsymbol{V}^h \hat{\boldsymbol{x}}^h) \approx z(\boldsymbol{V}^H \hat{\boldsymbol{x}}^H) + \frac{\partial z}{\partial \boldsymbol{x}}(\boldsymbol{V}^H \hat{\boldsymbol{x}}^H)\boldsymbol{V}^h(\hat{\boldsymbol{x}}^h - \boldsymbol{I}_H^h \hat{\boldsymbol{x}}^H) \quad (3)$$

- Approximate the fine residual:

$$0 = (\boldsymbol{V}^h)^T \boldsymbol{r}(\boldsymbol{V}^h \hat{\boldsymbol{x}}^h) \approx (\boldsymbol{V}^h)^T \boldsymbol{r}(\boldsymbol{V}^H \hat{\boldsymbol{x}}^H) + (\boldsymbol{V}^h)^T \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{x}}(\boldsymbol{V}^H \hat{\boldsymbol{x}}^H)\boldsymbol{V}^h(\hat{\boldsymbol{x}}^h -$$

- Solve for the error:

$$(\hat{\boldsymbol{x}}^h - \boldsymbol{I}_H^h \hat{\boldsymbol{x}}^H) \approx -[(\boldsymbol{V}^h)^T \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{x}}(\boldsymbol{V}^H \hat{\boldsymbol{x}}^H)\boldsymbol{V}^h]^{-1}(\boldsymbol{V}^h)^T \boldsymbol{r}(\boldsymbol{V}^H \hat{\boldsymbol{x}}^H) \quad (4)$$

Dual-weighted residual error indicators

- Approximate fine output:

$$z(\boldsymbol{V}^h\hat{\boldsymbol{x}}^h) \approx z(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H) + \frac{\partial z}{\partial \boldsymbol{x}}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)\boldsymbol{V}^h(\hat{\boldsymbol{x}}^h - \boldsymbol{I}_H^h\hat{\boldsymbol{x}}^H) \quad (3)$$

- Approximate the fine residual:

$$0 = (\boldsymbol{V}^h)^T\boldsymbol{r}(\boldsymbol{V}^h\hat{\boldsymbol{x}}^h) \approx (\boldsymbol{V}^h)^T\boldsymbol{r}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H) + (\boldsymbol{V}^h)^T\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{x}}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)\boldsymbol{V}^h(\hat{\boldsymbol{x}}^h -$$

- Solve for the error:

$$(\hat{\boldsymbol{x}}^h - \boldsymbol{I}_H^h\hat{\boldsymbol{x}}^H) \approx -[(\boldsymbol{V}^h)^T\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{x}}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)\boldsymbol{V}^h]^{-1}(\boldsymbol{V}^h)^T\boldsymbol{r}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H) \quad (4)$$

- Substitute (**??**) in (**??**):

$$\boxed{z(\boldsymbol{V}^h\hat{\boldsymbol{x}}^h) - z(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H) \approx -(\hat{\boldsymbol{y}}^h)^T(\boldsymbol{V}^h)^T\boldsymbol{r}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)},$$

with the fine adjoint solution $\hat{\boldsymbol{y}}^h \in \mathbb{R}^q$ satisfying

$$\boxed{(\boldsymbol{V}^{h\,T}\frac{\partial \boldsymbol{r}^k}{\partial \boldsymbol{x}}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)^T\boldsymbol{V}^h\hat{\boldsymbol{y}}^h = \boldsymbol{V}^{h\,T}\frac{\partial z}{\partial \boldsymbol{x}}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)^T}.$$

Dual-weighted residual error indicators

$$\boxed{z(\boldsymbol{V}^h\hat{\boldsymbol{x}}^h) - z(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H) \approx -(\hat{\boldsymbol{y}}^h)^T(\boldsymbol{V}^h)^T\boldsymbol{r}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)} \tag{5}$$

$$\boxed{(\boldsymbol{V}^h)^T\frac{\partial \boldsymbol{r}^k}{\partial \boldsymbol{x}}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)^T\boldsymbol{V}^h\hat{\boldsymbol{y}}^h = (\boldsymbol{V}^h)^T\frac{\partial z}{\partial \boldsymbol{x}}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)^T} \tag{6}$$

- We want to avoid fine solve (**??**), so approximate $\hat{\boldsymbol{y}}^h$ as

$$\hat{\boldsymbol{y}}^h_H = \boldsymbol{I}^h_H\hat{\boldsymbol{y}}^H,$$

where $\hat{\boldsymbol{y}}^H$ is the coarse adjoint solution to

$$(\boldsymbol{V}^H)^T\frac{\partial \boldsymbol{r}^k}{\partial \boldsymbol{x}}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)^T\boldsymbol{V}^H\hat{\boldsymbol{y}}^H = (\boldsymbol{V}^H)^T\frac{\partial z}{\partial \boldsymbol{x}}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)^T.$$

Dual-weighted residual error indicators

$$\boxed{z(\boldsymbol{V}^h\hat{\boldsymbol{x}}^h) - z(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H) \approx -(\hat{\boldsymbol{y}}^h)^T(\boldsymbol{V}^h)^T\boldsymbol{r}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)} \quad (5)$$

$$\boxed{(\boldsymbol{V}^h)^T\frac{\partial \boldsymbol{r}^k}{\partial \boldsymbol{x}}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)^T\boldsymbol{V}^h\hat{\boldsymbol{y}}^h = (\boldsymbol{V}^h)^T\frac{\partial z}{\partial \boldsymbol{x}}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)^T} \quad (6)$$

- We want to avoid fine solve (**??**), so approximate $\hat{\boldsymbol{y}}^h$ as

$$\hat{\boldsymbol{y}}^h_H = \boldsymbol{I}^h_H\hat{\boldsymbol{y}}^H,$$

  where $\hat{\boldsymbol{y}}^H$ is the coarse adjoint solution to

$$(\boldsymbol{V}^H)^T\frac{\partial \boldsymbol{r}^k}{\partial \boldsymbol{x}}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)^T\boldsymbol{V}^H\hat{\boldsymbol{y}}^H = (\boldsymbol{V}^H)^T\frac{\partial z}{\partial \boldsymbol{x}}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)^T.$$

- Substituting $\hat{\boldsymbol{y}}^h_H$ for $\hat{\boldsymbol{y}}^h$ in (**??**) yields cheaply computable

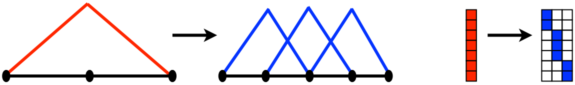
$$z(\boldsymbol{V}^h\hat{\boldsymbol{x}}^h) - z(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H) \approx -(\hat{\boldsymbol{y}}^h_H)^T(\boldsymbol{V}^h)^T\boldsymbol{r}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H).$$

- The RHS can be bounded by cheaply computable error indicators

$$|(\hat{\boldsymbol{y}}^h_H)^T(\boldsymbol{V}^h)^T\boldsymbol{r}(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H)| \leq \sum_{i=1}^{q}\boldsymbol{\Delta}^h_i, \qquad \boxed{\boldsymbol{\Delta}^h_i = |\left[\hat{y}^h_H\right]_i\left(\boldsymbol{v}^h_i\right)^T\boldsymbol{r}\left(\boldsymbol{V}^H\hat{\boldsymbol{x}}^H\right)|.}$$

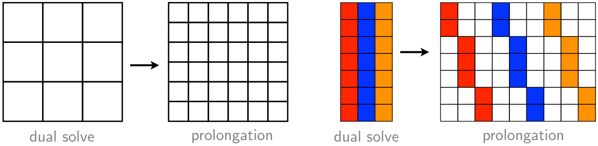
Ingredient 3: Adaptive algorithm

- **Main idea**: ROM analog to mesh-adaptive *h*-refinement
  1. *Refinement*: split into basis vectors with different support



finite element h-refinement *ROM h-refinement*

  2. *Error indicators*: a) dual solve (coarse), b) prolongation (fine)



finite element h-refinement *ROM h-refinement*

  3. *Adaptive algorithm*: refine vectors with large error indicators

Adaptive algorithm

Algorithm 1 Outer loop

Input: timestep k, basis $\boldsymbol{V}$
Output: updated basis $\boldsymbol{V}$, generalized state $\hat{\boldsymbol{x}}^k$

Compute ROM solution $\hat{\boldsymbol{x}}^k$ satisfying $\boldsymbol{V}^T \boldsymbol{r}^k(\boldsymbol{V}\hat{\boldsymbol{x}}^k; \boldsymbol{\mu}) = 0$.
if estimate of output error is 'too large' **then**
    Refine basis via Algorithm **??**: $\boldsymbol{V} \leftarrow \text{Refine}\left(\boldsymbol{V}, \hat{\boldsymbol{x}}^k\right)$.
    Return to Step **??**.
end if
if $\mod(k, n_{\text{reset}}) = 0$ **then**
    Reset basis $\boldsymbol{V} \leftarrow \boldsymbol{V}^{(\mathbf{0})}$.
end if

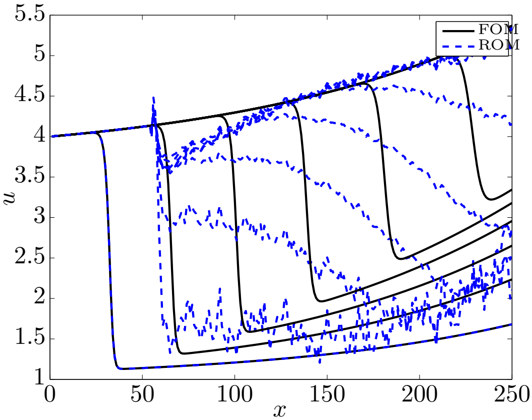
Adaptive algorithm

Algorithm 2 Refine

Input: initial basis $\boldsymbol{V}$, reduced solution $\hat{\boldsymbol{x}}$
Output: refined basis $\boldsymbol{V}$

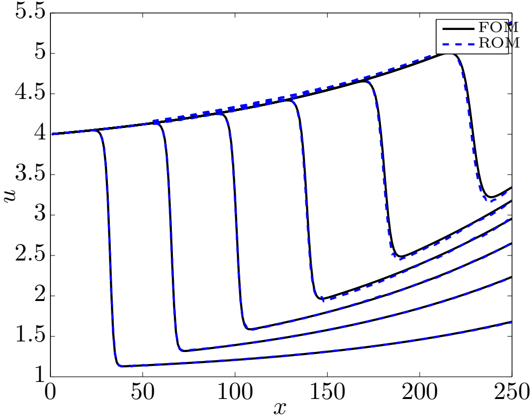
Compute prolongation operator $\boldsymbol{I}_H^h$ and fine basis $\boldsymbol{V}^h$
Solve: Compute coarse adjoint solution $\hat{\boldsymbol{y}}^H$ and prolongation $\hat{\boldsymbol{y}}_H^h$
Estimate: Compute fine error indicators $\boldsymbol{\Delta}_i^h$, $i = 1, \ldots, q$
Mark: Identify basis vectors to refine I
for $i \in I$ **do**
    Refine: Split $\boldsymbol{v}_i$ into $\mathrm{card}\,(C\,(d_i))$ vectors
end for
Compute QR factorization with column pivoting $\boldsymbol{V} = \boldsymbol{Q}\boldsymbol{R}$, $\boldsymbol{R}\bar{\boldsymbol{\Pi}} = \bar{\boldsymbol{Q}}\bar{\boldsymbol{R}}$
Ensure full-rank matrix $\boldsymbol{V} \leftarrow \boldsymbol{V}\,[\bar{\boldsymbol{\pi}}_1 \ \cdots \ \bar{\boldsymbol{\pi}}_r]$

Previous example



- Generated by $\boldsymbol{V} \in \mathbb{R}^{250\times 150}$ using $\tau_{\text{train}} \in [0, 2.5]$
- Now try *h*-adaptivity with $\boldsymbol{V}^{(0)} \in \mathbb{R}^{250\times 10}$, $\tau_{\text{train}} \in [0, 2.5]$.

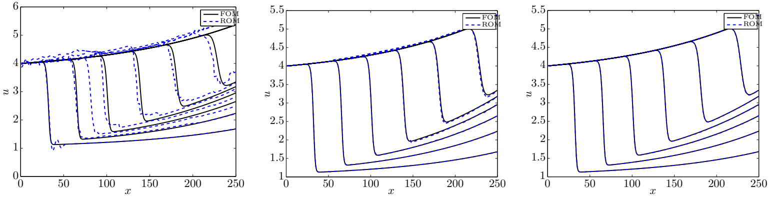
Previous example with h-adaptivity (FOM tolerance $\epsilon = 0.05$)



- $\dim \boldsymbol{V}^{(0)} = 10$
- $\text{mean}(\dim \boldsymbol{V}) = 44.3$

h-adaptation allows the ROM to capture phenomena not present in the training data!

h-adaptivity enables uncertainty/error control



(a) tolerance $\epsilon = 0.35$ (b) tolerance $\epsilon = 0.05$ (c) tolerance $\epsilon = 0.01$

	$\epsilon = 0.35$	$\epsilon = 0.05$	$\epsilon = 0.01$
average basis dimension per Newton iteration	33.6	44.2507	53.9
relative error (%) online time (seconds)	12.2 4.61	0.51 4.63	0.078 7.64

Conclusions

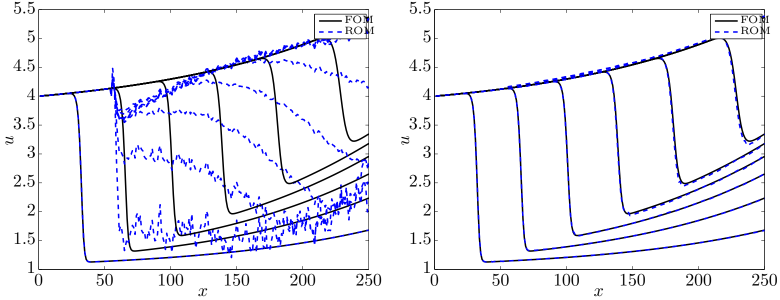
- Adaptive *h*-refinement via splitting
  - + Incrementally improves ROM
  - + Does not require large-scale operations
  - + Enables uncertainty/error control
  - + Extends utility of ROMs to hyperbolic PDEs
- Future work
  - Production implementation
  - Adaptive *coarsening*
  - Adaptive *p*-refinement: add basis vectors from a library

Acknowledgments

- Matthew Zahr: providing the model-reduction testbed that was modified to generate the numerical results
- Seshadhri Comandur, Khachik Sargsyan: helpful discussions
- This research was supported in part by an appointment to the Sandia National Laboratories Truman Fellowship in National Security Science and Engineering, sponsored by Sandia Corporation (a wholly owned subsidiary of Lockheed Martin Corporation) as Operator of Sandia National Laboratories under its U.S. Department of Energy Contract No. DE-AC04-94AL85000.

Questions?

- K. Carlberg. "Adaptive *h*-refinement for reduced-order models," International Journal for Numerical Methods in Engineering, in press (2014). doi:10.1002/nme.4800



(d) no adaptivity (e) *h*-adaptivity

Bibliography I

Amsallem, D., Cortial, J., Carlberg, K., and Farhat, C. (2009).
A method for interpolating on manifolds structural dynamics reduced-order models.
International Journal for Numerical Methods in Engineering, 80(9):1241–1258.

Amsallem, D. and Farhat, C. (2008).
An interpolation method for adapting reduced-order models and application to aeroelasticity.
AIAA Journal, 46(7):1803–1813.

Amsallem, D., Zahr, M. J., and Farhat, C. (2012).
Nonlinear model order reduction based on local reduced-order bases.
International Journal for Numerical Methods in Engineering, 92(10):891–916.

Bibliography II

Arian, E., Fahl, M., and Sachs, E. W. (2000).
Trust-region proper orthogonal decomposition for flow control.
Technical Report 25, ICASE.

Babuška, I. and Miller, A. (1984).
The post-processing approach in the finite element method—part 1: Calculation of displacements, stresses and other higher derivatives of the displacements.
International Journal for numerical methods in engineering, 20(6):1085–1109.

Bangerth, W. and Rannacher, R. (1999).
Finite element approximation of the acoustic wave equation: Error control and mesh adaptation.
East West Journal of Numerical Mathematics, 7(4):263–282.

Bibliography III

Bangerth, W. and Rannacher, R. (2003).
Adaptive finite element methods for differential equations.
Springer.

Becker, R. and Rannacher, R. (1996).
Weighted a posteriori error control in finite element methods, volume preprint no. 96-1.
Universitat Heidelberg.

Becker, R. and Rannacher, R. (2001).
An optimal control approach to a posteriori error estimation in finite element methods.
Acta Numerica 2001, 10:1–102.

Bibliography IV

Carlberg, K. and Farhat, C. (2009).
An adaptive POD-Krylov reduced-order model for structural optimization.
8th World Congress on Structural and Multidisciplinary Optimization, Lisbon, Portugal.

Carlberg, K., Farhat, C., Cortial, J., and Amsallem, D. (2013).
The GNAT method for nonlinear model reduction: effective implementation and application to computational fluid dynamics and turbulent flows.
Journal of Computational Physics, 242:623–647.

Dihlmann, M., Drohmann, M., and Haasdonk, B.
Model reduction of parametrized evolution problems using the reduced basis method with adaptive time partitioning.

Bibliography V

Drohmann, M., Haasdonk, B., and Ohlberger, M. (2011).
Adaptive reduced basis methods for nonlinear convection–diffusion equations.
In *Finite Volumes for Complex Applications VI Problems & Perspectives*, pages 369–377. Springer.

Eftang, J. L., Knezevic, D. J., and Patera, A. T. (2011).
An hp certified reduced basis method for parametrized parabolic partial differential equations.
Mathematical and Computer Modelling of Dynamical Systems, 17(4):395–422.

Eftang, J. L., Patera, A. T., and Rønquist, E. M. (2010).
An 'hp' certified reduced basis method for parametrized elliptic partial differential equations.
SIAM Journal on Scientific Computing, 32(6):3170–3200.

Eldred, M. S., Giunta, A. A., Collis, S. S., Alexandrov, N. A., and Lewis, R. M. (2004).
Second-order corrections for surrogate-based optimization with model hierarchies.
In *Proceedings of the 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, NY*, number AIAA Paper 2004-4457.

Eldred, M. S., Weickum, G., and Maute, K. (2009).
A multi-point reduced-order modeling approach of transient structural dynamics with application to robust design optimization.
Structural and Multidisciplinary Optimization, 38(6):599–611.

Bibliography VII

Estep, D. (1995).
A posteriori error bounds and global error control for approximation of ordinary differential equations.
SIAM Journal on Numerical Analysis, 32(1):1–48.

Fidkowski, K. J. (2007).
A simplex cut-cell adaptive method for high-order discretizations of the compressible Navier-Stokes equations.
PhD thesis, Massachusetts Institute of Technology.

Haasdonk, B., Dihlmann, M., and Ohlberger, M. (2011).
A training set and multiple bases generation approach for parameterized model reduction based on adaptive grids in parameter space.
Mathematical and Computer Modelling of Dynamical Systems, 17(4):423–442.

Bibliography VIII

Huynh, D. B. P., Knezevic, D. J., Chen, Y., Hesthaven, J. S., and Patera, A. T. (2010).
A natural-norm successive constraint method for inf-sup lower bounds.
Comput. Methods Appl. Mech. Engrg., 199:1963–1975.

Lu, J. C.-C. (2005).
An a posteriori error control framework for adaptive precision optimization using discontinuous Galerkin finite element method.
PhD thesis, Massachusetts Institute of Technology.

Bibliography IX

Nemec, M. and Aftosmis, M. (2007).
Adjoint error estimation and adaptive refinement for embedded-boundary cartesian meshes.
In *18th AIAA CFD Conference, Miami. Paper AIAA-2007-4187.*

Ng, L. and Eldred, M. S. (2012).
Multifidelity uncertainty quantification using non-intrusive polynomial chaos and stochastic collocation.
In *AIAA 2012-1852.*

Park, M. A. (2004).
Adjoint-based, three-dimensional error prediction and grid adaptation.
AIAA journal, 42(9):1854–1862.

Patera, A. T. and Rozza, G. (2006).
Reduced basis approximation and a posteriori error estimation for parametrized partial differential equations.
MIT.

Peherstorfer, B., Butnaru, D., Willcox, K., and Bungartz, H.-J. (2014).
Localized discrete empirical interpolation method.
SIAM Journal on Scientific Computing, 36(1):A168—A192.

Pierce, N. A. and Giles, M. B. (2000).
Adjoint recovery of superconvergent functionals from pde approximations.
SIAM review, 42(2):247–264.

Bibliography XI

Rannacher, R. (1999).
The dual-weighted-residual method for error control and mesh adaptation in finite element methods.
MAFELEAP, 99:97–115.

Rasmussen, C. and Williams, C. (2006).
Gaussian Processes for Machine Learning.
MIT Press.

Rewienski, M. J. (2003).
A Trajectory Piecewise-Linear Approach to Model Order Reduction of Nonlinear Dynamical Systems.
PhD thesis, Massachusetts Institute of Technology.

Ryckelynck, D. (2005).
A priori hyperreduction method: an adaptive approach.
Journal of Computational Physics, 202(1):346–366.

Venditti, D. and Darmofal, D. (2000).
Adjoint error estimation and grid adaptation for functional outputs: Application to quasi-one-dimensional flow.
Journal of Computational Physics, 164(1):204–227.

Venditti, D. A. and Darmofal, D. L. (2002).
Grid adaptation for functional outputs: application to two-dimensional inviscid flows.
Journal of Computational Physics, 176(1):40–69.

Washabaugh, K., Amsallem, D., Zahr, M., and Farhat, C. (2012).
Nonlinear model reduction for cfd problems using local reduced-order bases.

Bibliography XIII

In *42nd AIAA Fluid Dynamics Conference and Exhibit, Fluid Dynamics and Co-located Conferences, AIAA Paper*, volume 2686.

Wirtz, D., Sorensen, D. C., and Haasdonk, B. (2012).
A-posteriori error estimation for DEIM reduced nonlinear dynamical systems.
Preprint Series, Stuttgart Research Centre for Simulation Technology.

Yano, M., Patera, A. T., and Urban, K. (2012).
A space-time certified reduced-basis method for Burgers' equation.
Math. Mod. Meth. Appl. S., submitted.

Inviscid Burgers equation: results

	typical ROM			h-adaptive ROM				
dim $\boldsymbol{V}^{(0)}$ basis-reset frequency	10	45	150	5 50	10 50	20 50	10 100	10 25
mean(dim $\boldsymbol{V}$) Avg splits	10	45	150	41.4 0.20	44.3 0.19	58 0.14	73 0.13	37 0.28
error (%) online time (s)	45.8 1.4	43.9 2.14	8.5 5.77	0.3 5.53	0.5 4.63	0.2 7.27	0.2 6.90	0.3 7.46

+ Low errors achievable only with *h*-adaptivity
- *Smaller initial basis*: smaller basis dimension, but more splits
- *More frequent basis resetting*: smaller basis dimension, but more splits