CFD motivated applications of model order reduction

Martin Isoz[1,2]

[1] UCT Prague, Department of mathematics
[2] Czech Academy of Sciences, Institute of thermomechanics

Seminar on Numerical Analysis,
Ostrava, January 30 – February 3, 2017

Outline

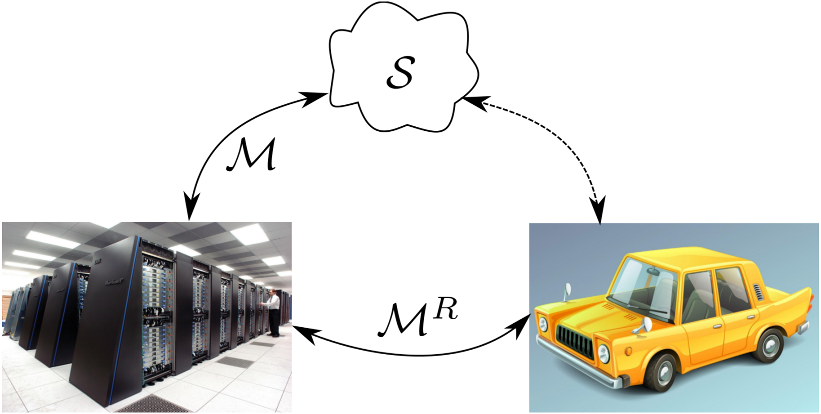
From the research motivation to results

1. Introduction
   - Research motivation
   - Problem setting
2. POD & DEIM based MOR
   - ROM
   - POD
   - Nonlinearities
3. Link with OpenFOAM
   - oF basics
   - NS and p-U coupling
4. Applications
   - Full scale application example
5. Conclusions

Introduction

Research motivation

Reducing the computational cost of modeling of complex systems



Problem setting

Modeling of large systems of differential equations

Original system

$$\dot{y} = Ay + f(t, y), \quad y(t) \in \mathbb{R}^m, \quad y(0) = y_0, \, t \in [0, T],$$

$$\text{system matrix} \quad \ldots \quad A \in \mathbb{R}^{m \times m},$$

$$\text{nonlinearities} \quad \ldots \quad f(t, y) \in \mathbb{R}^m$$

Reduced-order system

$$\dot{\eta}^\ell = A^\ell \eta^\ell + f^\ell(t, \eta^\ell), \quad \eta^\ell(t) \in \mathbb{R}^\ell, \quad \eta^\ell(0) = \eta_0^\ell, \, t \in [0, T],$$

$$\text{system matrix} \quad \ldots \quad A^\ell \in \mathbb{R}^{\ell \times \ell},$$

$$\text{nonlinearities} \quad \ldots \quad f^\ell(t, \eta^\ell) \in \mathbb{R}^\ell$$

$$\text{gain} \quad \ldots \quad \ell \ll m$$

Basic principles of Model order reduction

- Prerequisities:

$$\dot{y} = Ay + f(t,y), \quad y(t) \in \mathbb{R}^m, \quad y(0) = y_0,\, t \in [0,T]$$

$$y(t) \in V = \text{span}\{\psi_j\}_{j=1}^d \quad \forall t \in [0,T]$$

$\Psi = \{\psi_j\}_{j=1}^d$ …orthonormal, POD basis

$$y(t) = \sum_{j=1}^{d} \langle y(t), \psi_j \rangle_W \, \psi_j,\, \forall t \in [0,T], \quad W \ldots \text{appropriate weights}$$

- Ansatz for Galerkin projection, $\ell < d$

$$y^\ell(t) := \sum_{j=1}^{\ell} \langle y^\ell(t), \psi_j \rangle_W \, \psi_j,\, \forall t \in [0,T], \quad \eta_j^\ell(t) := \langle y^\ell(t), \psi_j \rangle_W$$

- Put the above together, !! $\psi_j \in \mathbb{R}^m$, $j = 1, \ldots, \ell$, $m > \ell$!!

$$\begin{aligned} \textstyle\sum_{j=1}^{\ell} \dot{\eta}_j^\ell \psi_j &= \textstyle\sum_{j=1}^{\ell} \eta_j^\ell A\psi_j + f(t, y^\ell(t)), \quad t \in (0,T) \\ y_0 &= \textstyle\sum_{j=1}^{\ell} \eta_j^\ell(0) \psi_j \end{aligned}$$

Reduced order model

Construct a corresponding model of reduced order (ROM)

- Assume, that the above holds after projection on $V^\ell = \mathrm{span}\{\psi_j\}_{j=1}^{\ell}$, remember that $\langle \psi_j, \psi_i \rangle_W = \delta_{ij}$ and write,

$$\dot{\eta}_i^\ell = \sum_{j=1}^{\ell} \eta_j^\ell \langle A\psi_j, \psi_i \rangle_W + \langle f(t, y^\ell), \psi_i \rangle_W, \quad 1 \le i \le l \text{ and } t \in (0, T]$$

- Define the matrix $A^\ell = (a_{ij}^\ell) \in \mathbb{R}^{l \times l}$ with $a_{ij}^\ell = \langle A\psi_j, \psi_i \rangle_W$
- Define the vector valued mapping $\eta^\ell = (\eta_1^\ell, \ldots, \eta_l^\ell)^{\mathrm{T}} : [0, T] \to \mathbb{R}^\ell$
- Define the non-linearity $f^\ell = (f_1^\ell, \ldots, f_l^\ell)^{\mathrm{T}} : [0, T] \to \mathbb{R}^\ell$, where

$$f_i^\ell(t, \eta) = \left\langle f\left(t, \sum_{j=1}^{\ell} \eta_j \psi_j\right), \psi_i \right\rangle_W$$

- Introduce the IC, $\eta^\ell(0) = \eta_0^\ell = (\langle y_0, \psi_1 \rangle_W, \ldots, \langle y_0, \psi_1 \rangle_W)^{\mathrm{T}}$
- Write the ROM, $\dot{\eta}^\ell = A^\ell \eta^\ell + f^\ell(t, \eta^\ell)$, for $t \in (0, T]$, $\eta^\ell(0) = \eta_0^\ell$

Algorithm 1 POD basis of rank ℓ with weighted inner product

Require: Snapshots $\{y_j\}_{j=1}^n$, POD rank $\ell \leq d$, symmetric positive-definite matrix of weights $W \in \mathbb{R}^{m\times m}$

1: Set $Y = [y_1, \dots, y_n] \in \mathbb{R}^{m\times n}$;
2: Determine $\bar{Y} = W^{1/2} Y \in \mathbb{R}^{m\times n}$;
3: Compute SVD, $[\bar{\Psi}, \Sigma, \bar{V}] = \mathrm{svd}(\bar{Y})$;
4: Set $\sigma = \mathrm{diag}(\Sigma)$;
5: Compute $\varepsilon(\ell) = \sum_{i=1}^{\ell} \sigma_i / \sum_{i=1}^{d} \sigma_i$;
6: Truncate $\bar{\Psi} \leftarrow [\bar{\psi}_1, \dots, \bar{\psi}_l] \in \mathbb{R}^{m\times \ell}$;
7: Compute $\Psi = W^{-1/2} \bar{\Psi} \in \mathbb{R}^{m\times \ell}$;
8: **return** POD basis, Ψ, and ratio $\varepsilon(\ell)$

Notes:

- All the operations on W have to be cheap, including its inversion.
- Do not perform the full SVD, $\Sigma \in \mathbb{R}^{d\times d}$, $d = \mathrm{rank}(\bar{Y})$.

Treatment of non-linearities I

Identify the problem and provide a remedy

- Identify the problem,

$$f_i^\ell(t,\eta) = \left\langle f\left(t, \sum_{j=1}^{\ell} \eta_j \psi_j\right), \psi_i \right\rangle_W \quad \dots \sum_{j=1}^{\ell} \eta_j \psi_j \in \mathbb{R}^m \leftarrow \text{ FO}$$

- Approximate the non-linearities, $f(t,y)$ via POD basis, Φ,

$$b(t) := f(t, \Psi\eta^\ell) \approx \sum_{k=1}^{p} \phi_k c_k(t) = \Phi c(t) \ \dots \text{ Galerkin ansatz}$$

- Approximate $f^\ell(t,\eta^\ell)$ through Ψ, W, Φ,

$$f^\ell(t,\eta^\ell) = \Psi^{\mathrm{T}} W f(t, \Psi\eta^\ell) = \Psi^{\mathrm{T}} W b(t) \approx \Psi^{\mathrm{T}} W \Phi c(t)$$

- Current situation,

$$c(t) \in \mathbb{R}^p \quad \text{but} \quad c \dots \text{new unknown}$$

Discrete Empirical Interpolation Method

POD & Greedy algorithm based method for handling non-linearities

Algorithm 2 DEIM

Require: p and matrix $F = [f(t_1, y_1), \dots, f(t_1, y_1)] \in \mathbb{R}^{m \times n}$

1: Compute POD basis $\Phi = [\phi_1, \dots, \phi_p]$ for F

2: $\mathrm{idx} \leftarrow \arg\max_{j=1,\dots,m} |(\phi_1)_{\{j\}}|$;

3: $U = [\phi_1]$ and $\vec{i} = \mathrm{idx}$;

4: **for** $i = 2$ **to** p **do**

5: $\quad u \leftarrow \phi_i$;

6: $\quad$ Solve $U_{\vec{i}} c = u_{\vec{i}}$;

7: $\quad r \leftarrow u - Uc$;

8: $\quad \mathrm{idx} \leftarrow \arg\max_{j=1,\dots,m} |(r)_{\{j\}}|$;

9: $\quad U \leftarrow [U, u]$ and $\vec{i} \leftarrow [\vec{i}, \mathrm{idx}]$;

10: **end for**

11: **return** $\Phi \in \mathbb{R}^{m \times p}$ and index vector, $\vec{i} \in \mathbb{R}^p$

Notes:

- Most of the computational cost is hidden on line 6.

- Plug in the matrix
  $P := [e_{\vec{i}1}, \ldots, e_{\vec{i}p}] \in \mathbb{R}^{m\times p}$, $e_{\vec{i}k} = (0, \ldots, 0, 1, 0, \ldots, 0)^{\mathrm{T}} \in \mathbb{R}^m$,

$$P^{\mathrm{T}}\Phi c(t) \approx P^{\mathrm{T}} b(t), \leftarrow c(t) \in \mathbb{R}^p,\ \Phi \in \mathbb{R}^{m\times p},\ b(t) \in \mathbb{R}^m$$

$$\det(P^{\mathrm{T}}\Phi) \neq 0 \implies c(t) \approx (P^{\mathrm{T}}\Phi)^{-1} P^{\mathrm{T}} b(t) = (P^{\mathrm{T}}\Phi)^{-1} P^{\mathrm{T}} f(t, \Psi\eta^\ell)$$

- If $f(t, \Psi\eta^\ell)$ is pointwise evaluable,

$$(P^{\mathrm{T}}\Phi)^{-1} P^{\mathrm{T}} f(t, \Psi\eta^\ell) = (P^{\mathrm{T}}\Phi)^{-1} f(t, P^{\mathrm{T}}\Psi\eta^\ell), \quad P^{\mathrm{T}}\Psi\eta^\ell \in \mathbb{R}^p$$

- Write the final ROM

$$\dot{\eta}^\ell = A^\ell\eta^\ell + f^\ell(t, \eta^\ell),\ \text{for}\ t \in (0, T], \quad \eta^\ell(0) = \eta^\ell_0,$$

where

$$f^\ell(t, \eta^\ell) = \Psi^{\mathrm{T}} W \Phi (P^{\mathrm{T}}\Phi)^{-1} f(t, P^{\mathrm{T}}\Psi\eta^\ell)$$

Link with OpenFOAM

Rewrite OpenFOAM discretization as above studied problem

- With $\Delta\Omega^h := \mathrm{diag}(\delta\Omega_i^h) \in \mathbb{R}^{m\times m}$ a FVM semi-discretized problem can be written as,

$$\Delta\Omega^h \dot{y} + \mathcal{L}^h(t,y) = 0 \implies \dot{y} = -(\Delta\Omega^h)^{-1}\mathcal{L}^h(t,y),$$

$\mathcal{L}^h = -\tilde{A}(t)y - \tilde{b}(t,y)$ ... FVM spatial discretization operator

- It is possible to formally write (almost) the same system as before,

$$\dot{y} = A(t)y + b(t,y), \quad A(t) = (\Delta\Omega^h)^{-1}\tilde{A}(t),\, b(t,y) = (\Delta\Omega^h)^{-1}\tilde{b}(t,y)$$

- The time dependence of A is a result of the linearization process. E.g. $\nabla\cdot(u^k \otimes u^k) \approx \nabla\cdot(u^{k-1}\otimes u^k)$
- The POD-DEIM approach to ROM creation will have to be slightly modified

Modifications to POD-DEIM ROM creation

Extended snapshots and interpolation

Address the risen difficulties

- Needed snapshots, $\{(y_i, A_i, b_i)\}_{i=1}^{n}$, $A_i \in \mathbb{R}^{m \times m}$, $i = 1, \dots, m$ but A_i are sparse matrices, with $\sim 5m$ non-zero elements $\implies$ $\sim 5m$ floats and $\sim 8m$ integers will be stored.
- A way for ROM evaluation between the stored snapshots is needed $\implies$ I need to interpolate between A_{i-1} and A_i and b_{i-1} and b_i, $i = 2, n$
- Simplest case: linear interpolation,

$$\varpi(t) = \frac{t - t_{i-1}}{t_i - t_{i-1}},\ \hat{A}(t) = \varpi(t) A_{i-1} + (1 - \varpi(t)) A_i$$

$$\hat{A}^{\ell}(t) = \Psi^{\mathrm{T}} W \hat{A}(t) \Psi = \Psi^{\mathrm{T}} W \left(\varpi(t) A_{i-1} + (1 - \varpi(t)) A_i\right) \Psi =$$

$$= \varpi(t) \Psi^{\mathrm{T}} W A_{i-1} \Psi + (1 - \varpi(t)) \Psi^{\mathrm{T}} W A_i \Psi = \varpi(t) A^{\ell}_{i-1} + (1 - \varpi(t)) A^{\ell}_i$$

- Same trick can be done for $b(t, y)$ and after the ROM creation, I do not need to store the full data.

interFoam – Volume-of-Fluid model for multiphase flow

$$\alpha_t + \nabla \cdot (u\alpha) + \nabla \cdot (u_r \alpha (1-\alpha)) = 0$$

$$\alpha_t + \mathcal{L}^h_\alpha(t,\alpha) = 0 \rightarrow \alpha_t = A_\alpha(t)\alpha + b_\alpha(t,\alpha) \rightarrow \dot{\eta}^\ell_\alpha = \hat{A}^\ell_\alpha(t)\eta^\ell_\alpha + \hat{b}^\ell_\alpha(t,\eta^\ell_\alpha)$$

Wanted: $\dot{y}_\alpha = A_\alpha(t) y_\alpha + b_\alpha(t,y)$

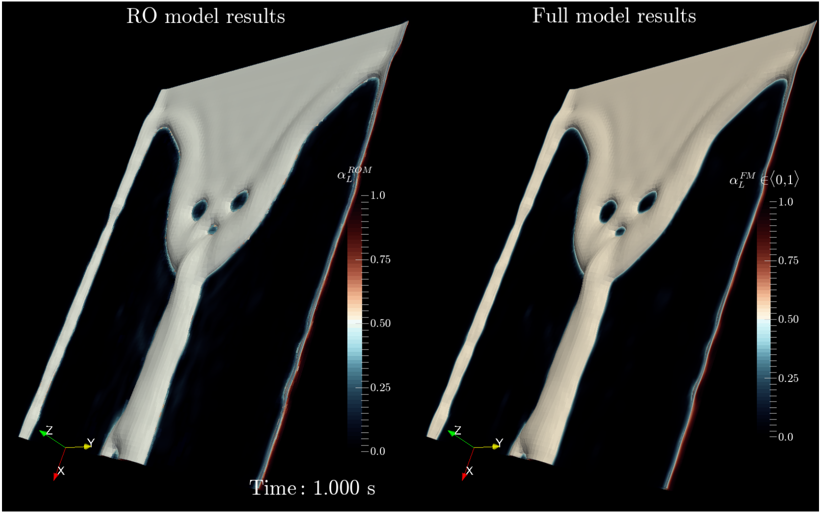
Example of implementation in OpenFOAM

```
    fvm::div( phi, alpha1, alphaScheme )
  + fvc::div(
       -fvc::flux(
          -phir, scalar(1)-alpha1, alpharScheme
        ),
        alpha1, alpharScheme
    ) == 0
```

Link: `fvm` $\rightarrow A_\alpha(t)$, `fvc` $\rightarrow b_\alpha(t,y)$

Example 1 – Passive scalar advection

Numerical results



Saddle-point problem

$$\begin{aligned} u_t + \nabla\cdot(u\otimes u) - \nabla\cdot(\nu\nabla u) &= -\nabla\tilde{p} + \tilde{f} \\ \nabla\cdot u &= 0 \end{aligned} \rightsquigarrow \begin{pmatrix} M & N^T \\ N & 0 \end{pmatrix}\begin{pmatrix} u^h \\ p^h \end{pmatrix} = \begin{pmatrix} f^h \\ 0 \end{pmatrix}$$

Schurr complement based pressure equation

$$N{\color{red}M^{-1}}N^{\mathrm{T}}p^h = N{\color{red}M^{-1}}f^h, \quad u^h = {\color{red}M^{-1}}(N^{\mathrm{T}}p^h - f^h)$$

Jacobi iterations with Schur-complement based p-U coupling

$$u^* \leftarrow Mu^* = f - N^{\mathrm{T}}p^{j-1}, \quad M = D + L + U$$

$$p^j \leftarrow ND^{-1}N^{\mathrm{T}}p^j = ND^{-1}\left(f - (L+U)u^*\right)$$

$$u^j \leftarrow D^{-1}\left(f - (L+U)u^* - N^{\mathrm{T}}p^j\right)$$

At convergence

$$ND^{-1}N^{\mathrm{T}}p = ND^{-1}\left(f - (L+U)u^*\right) \approx NM^{-1}N^{\mathrm{T}}p = NM^{-1}f$$

$$u \approx D^{-1}\left(f - (L+U)u^*\right) - D^{-1}N^{\mathrm{T}}p$$

$\Longrightarrow$ **"Natural" would seem to construct ROM for p** $\Longleftarrow$

Construction of ROM for p

Implementation of pressure equation in OpenFOAM and construction of ROM based on it

Notation

$D^{-1} \to \texttt{rAU}$ and $D^{-1}(f - (L+U)u^*) \to \texttt{HbyA}$ (in oF, $*\texttt{Eqn.A}() \to D$)

Implementation of pressure eqauation in OpenFOAM

```
fvm::laplacian(rAU, p) == fvc::div(HbyA)
```

Wanted: $\dot{y}_p = A_p(t)y_p + b_p(t, y_p)$

Implicit definition of time derivative for pressure

$$\begin{aligned} \nabla \cdot (u \otimes u) - \nabla \cdot (\nu \nabla u) &= -\nabla \tilde{p} + \tilde{f} \\ \nabla \cdot u &= 0 \end{aligned} \quad \rightsquigarrow \quad \begin{gathered} \texttt{UEqnMORE} \\ D_h^{-1} \to \texttt{rAUMORE} \\ D_h^{-1}(f_h - (L_h + U_h)u_h*) \to \\ \to \texttt{HMOREbyAMORE} \end{gathered}$$

```
fvm::laplacian(rAUMORE, p) == fvc::div(HMOREbyAMORE)
```

Link: $\texttt{fvm} \to A_p(t)$, $\texttt{fvc} \to b_p(t, y_p)$

Reconstruction of the velocity field

Create ROM or expand snapshot

Expansion of snapshots for pressure

Standard approach snapshots: $\mathcal{S} = \{y_{k,i}, A_{k,i}, b_{k,i}\}_{i=1}^{n}, k = p, U$

Expanded snapshots for pressure:

$$\mathcal{S}^e = \{y_{p,i}, A_{p,i}, b_{p,i}, \texttt{rAUMORE}_i, \texttt{HMOREbyAMORE}_i\}_{i=1}^{n}$$

Storage

$\mathcal{S} \ldots n\left[(1+3)m + (5+5)m + (1+3)m\right] \approx 15nm$ values

$\mathcal{S}^e \ldots n(m + 5m + m + 1m + 3m) \approx 11nm$ values

Computational cost

$\mathcal{S} \ldots \sim 4n$ calculations of $\Psi^{\mathrm{T}} W A(t) \Psi$, evaluation of ~ 4 ROMs

$\mathcal{S}^e \ldots$

$\sim n$ calculations of $\Psi^{\mathrm{T}} W A(t) \Psi$,

$\sim n$ calculations of $\Psi^{\mathrm{T}} W \texttt{rAUMORE}_i \Psi$,

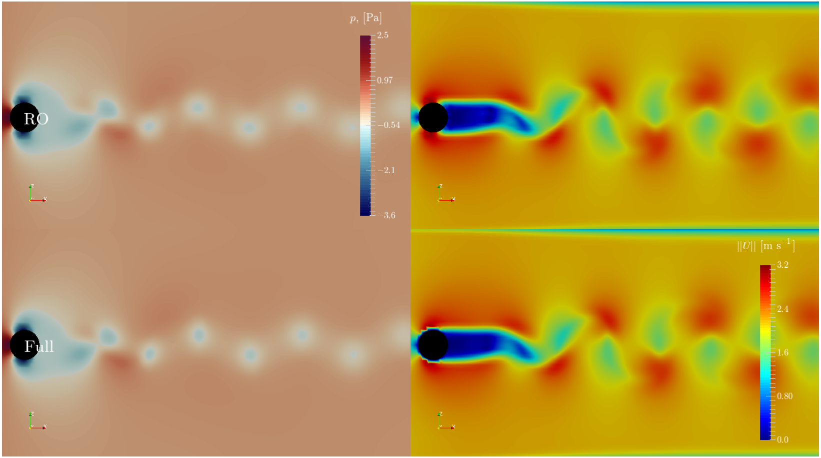
$\sim n$ calculations of $\Psi^{\mathrm{T}} W \texttt{HMOREbyAMORE}_i \Psi$,

evaluation of 1 ROM + interpolation between $\texttt{rAUMORE}_i^{ROM}$ and between $\texttt{HMOREbyAMORE}_i^{ROM}$

$$U_i \approx \texttt{HMOREbyAMORE}^{ROM} + \texttt{rAUMORE}^{ROM} \nabla p^{ROM}$$

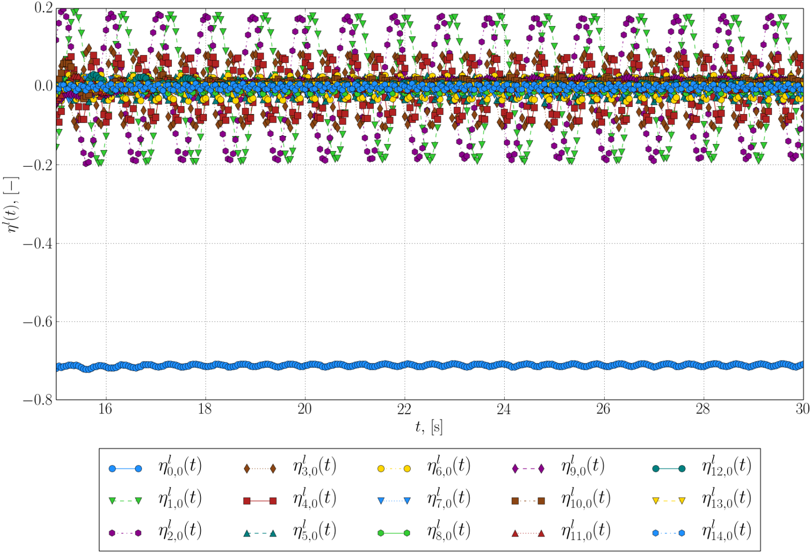
Example 2 – Von Karman vortex row

Validation of the approach



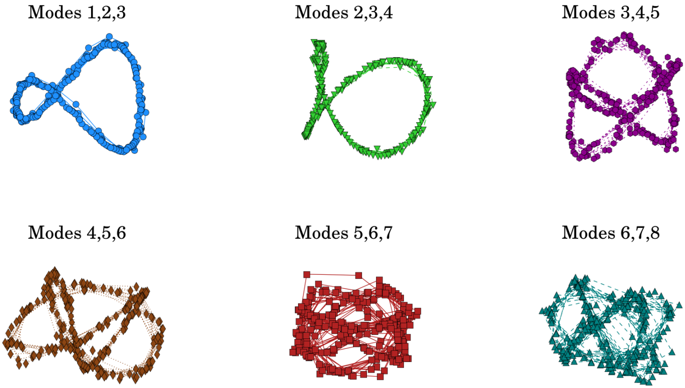
Example 2 – Von Karman vortex row

Validation of the approach



Example 2 – Von Karman vortex row

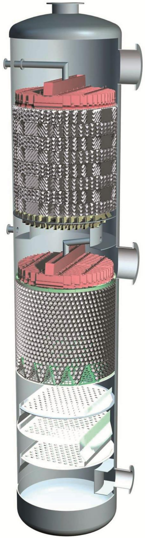
Validation of the approach



Applications

Real-life applications

ROM is a tremendous tool for parametric studies or repeated model evaluations



[Sulzer ChemTech]

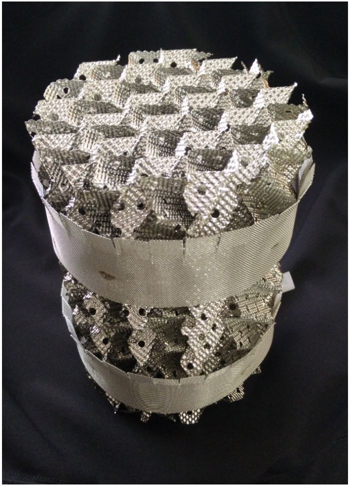
Importance

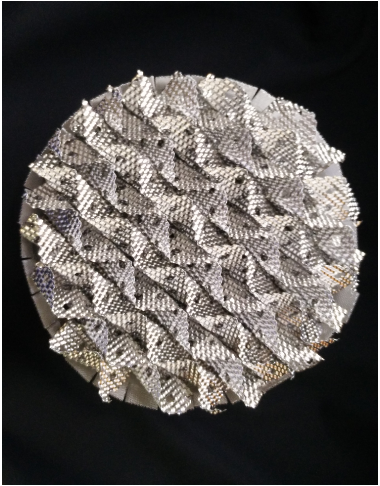
- Chemical industry creates mixtures but sells ”pure species” (e.g. oil)
- 2014, 3% of energy consumption of the USA was due to the separation columns

Challenges

- Multiphase flow $\rightarrow$ non-steady process
- Complex geometry
- Simultaneous heat and mass transfer

Challenge: Geometry of structured packing

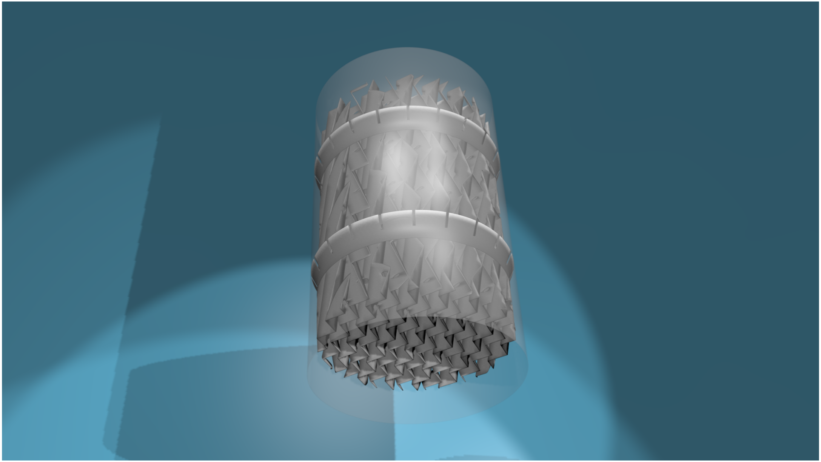




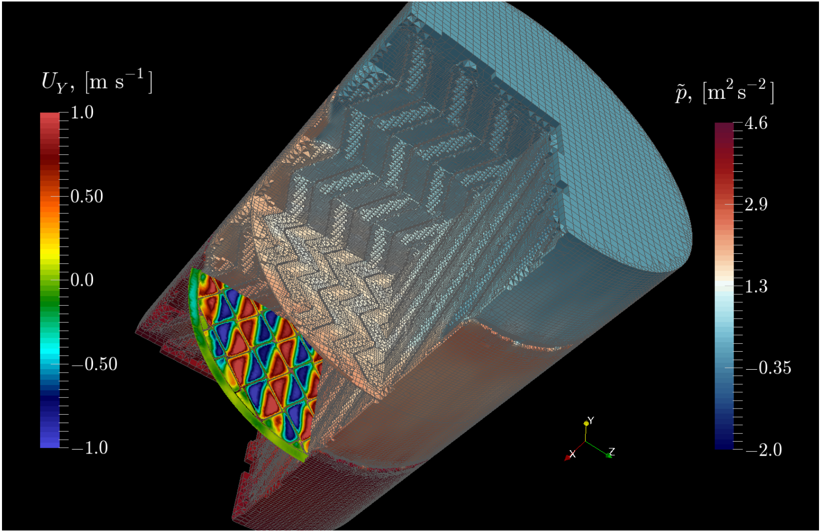
Semi-industrial scale CFD

Steady-state RANS simulation of flow in complex geometry

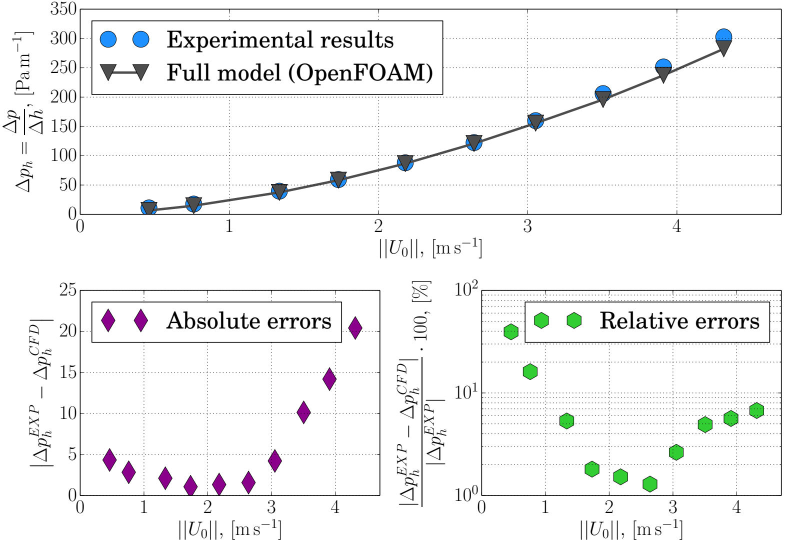
Geometry: Mellapak 250.X structured packing



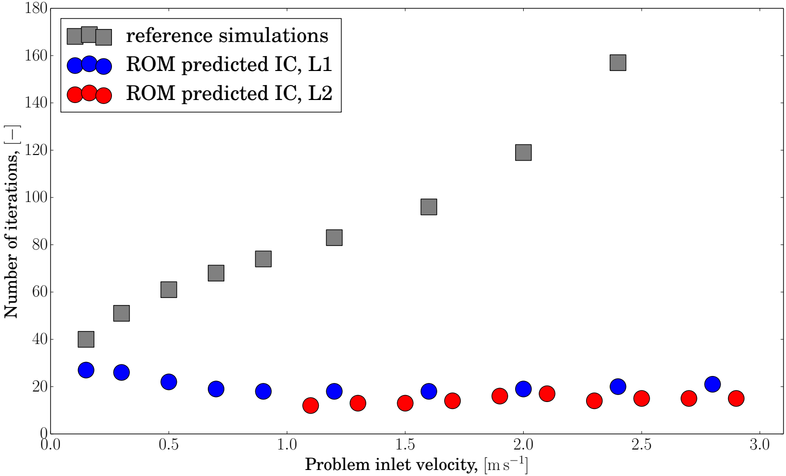
Gas flow simulation: Incompressible steady state RANS simulation



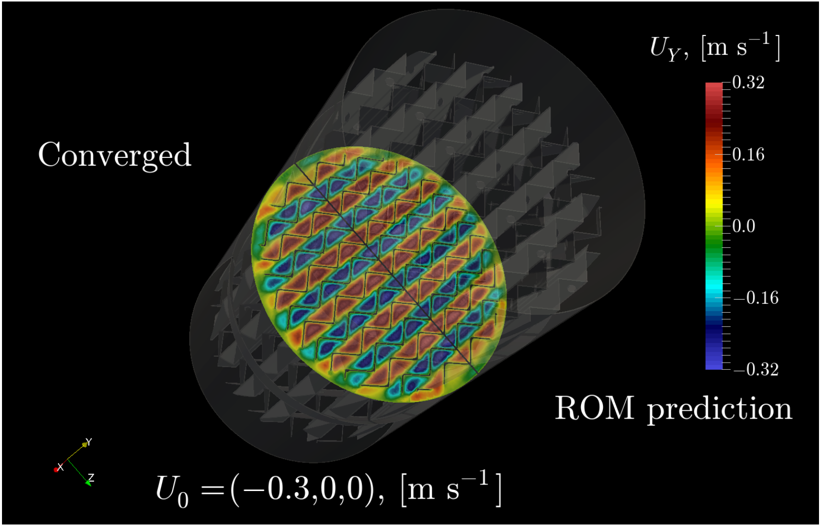
Comparison with experimental data: [Haidl, J. UCT Prague]



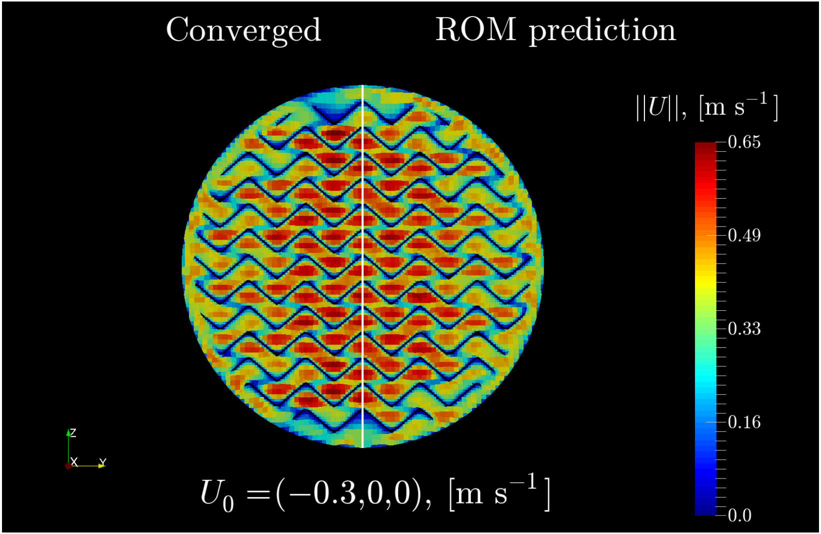
Full case: Flow through the Mellapak 250.X packing



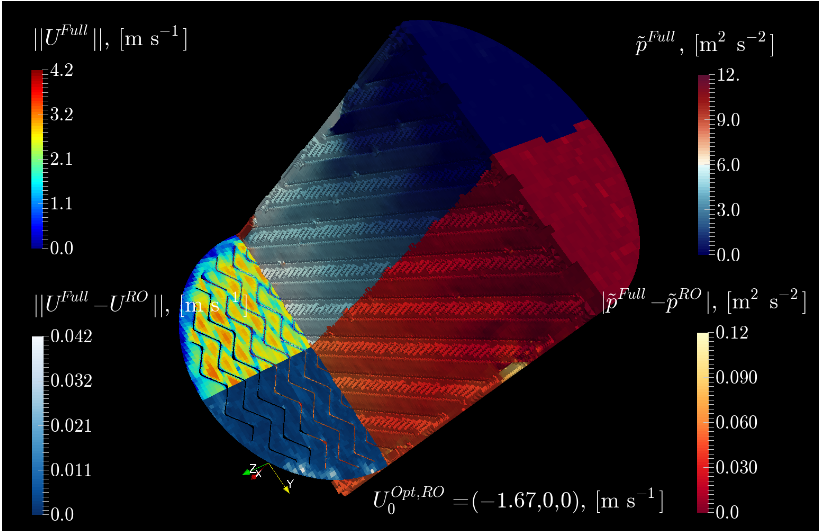
Full case: Predicted vs. converged solution in L1



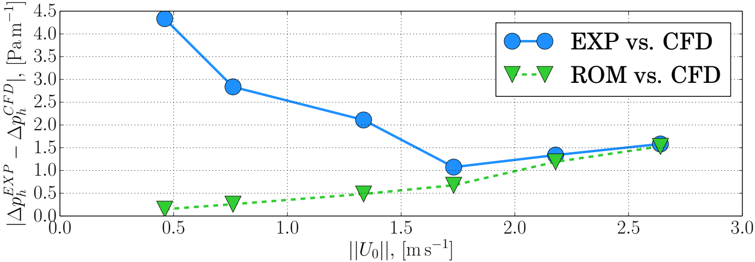
Full case: Predicted vs. converged solution in L1

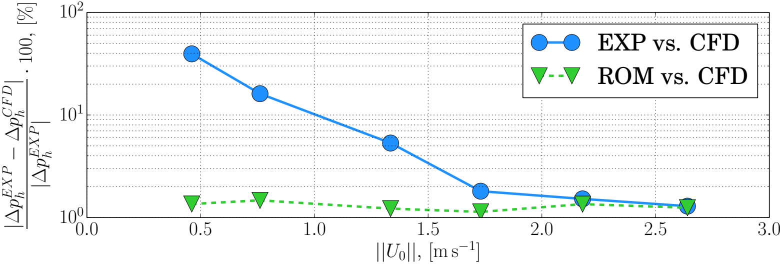


Full case: Predicted vs. converged solution in L1



Comparison with experimental data: [Haidl, J. UCT Prague]





Conclusions

Conclusions

POD-DEIM ROMs backed by OpenFOAM CFD machinery

Currently available

- Extended snapshot preparation for `simpleFoam` and `pimpleFoam`
- Unfinished: Extended snapshot preparation for `interFoam`
- Python module for ROM creation based on prepared outputs from OpenFOAM

Advantages

- Snaphots are created during postprocessing - simulations can be ran in parallel
- All the OpenFOAM capabilities are accessible (including e.g. MRF or turbulence modeling)

Disadvantages

- Extended shapshots have to be stored - a lot of data
- Creation of A_i^ℓ, $i = 1, \ldots, n$ is time consuming

Acknowledgments

The work of M. Isoz was supported by the Centre of Excellence for nonlinear dynamic behaviour of advanced materials in engineering CZ.02.1.01/0.0/0.0/15_003/0000493 (Excellent Research Teams) in the framework of Operational Programme Research, Development and Education. Moreover, the author thankfully acknowledges financial support from IGA of UCT Prague, grant numbers A2_FTOP_2016_024 and A1_FCHI_2016_004 and from the Czech Science Foundation, grant number GACR 13-01251S. Finally, the author would like to express his deepest thanks to the Mass Transfer Laboratory of UCT Prague for providing theirs, yet unpublished, experimental data for the model validation.

References

[1] Volkwein, S.: Proper Orthogonal Decomposition: Theory and Reduced-Order Modelling. *LN*, University of Konstanz, 2013.

[2] Volkwein, S.: Proper Orthogonal Decomposition: Applications in Optimization and Control

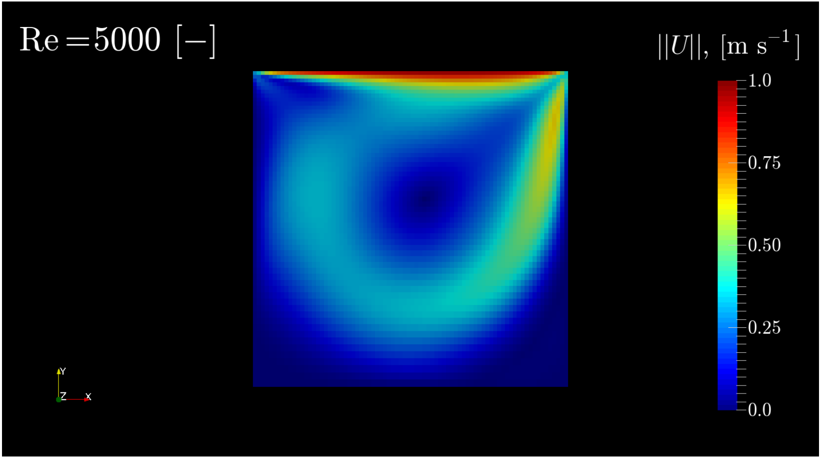
[3] Chaturantabut, S. Sorensen, D. C.: Nonlinear Model Reduction Via Discrete Empirical Interpolation, *SIAM J. Sci. Comput.*, vol. 32, (2010) pp. 2737–2764.

[4] Chaturantabut, S. Sorensen, D. C.: Application of POD and DEIM on Dimension Reduction of Nonlinear Miscible Viscous Fingering in Porous Media, *Math. Comput. Model. Dyn. Syst., (Technical Report: CAAM)*, Rice University, TR09-25

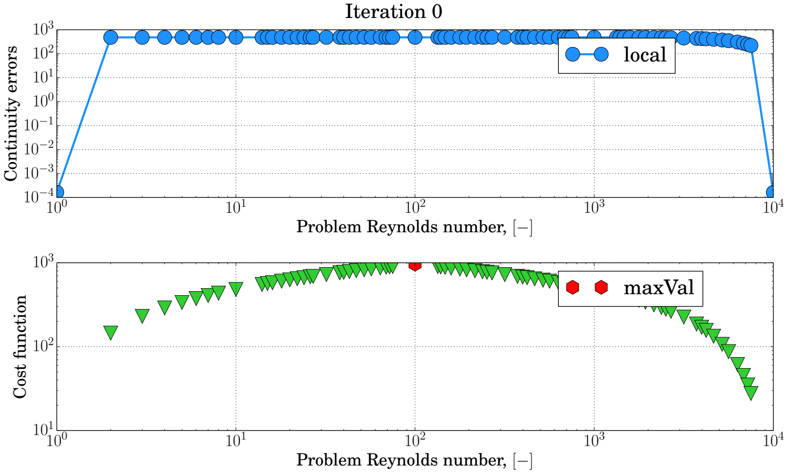
[5] Alla, A. Kutz, J. N.: Nonlinear Model Order Reduction Via Dynamic Mode Decomposition, *preprint*

Thank you for your attention

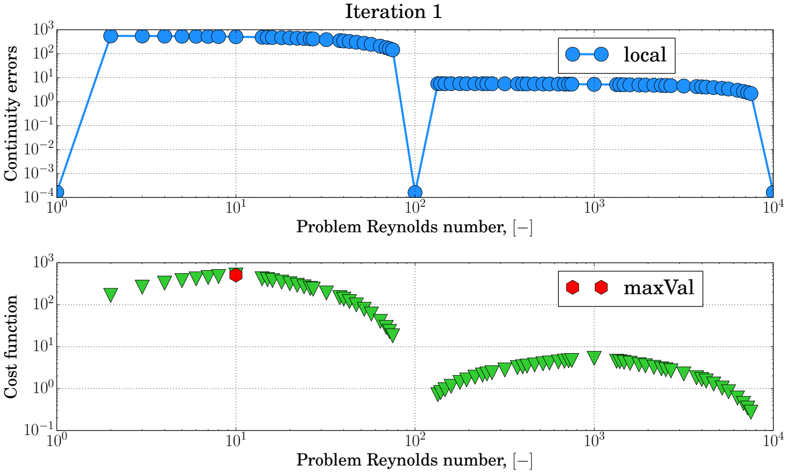
ROM size reduction: $\mathcal{S}^e$ selection based on greedy algorithm



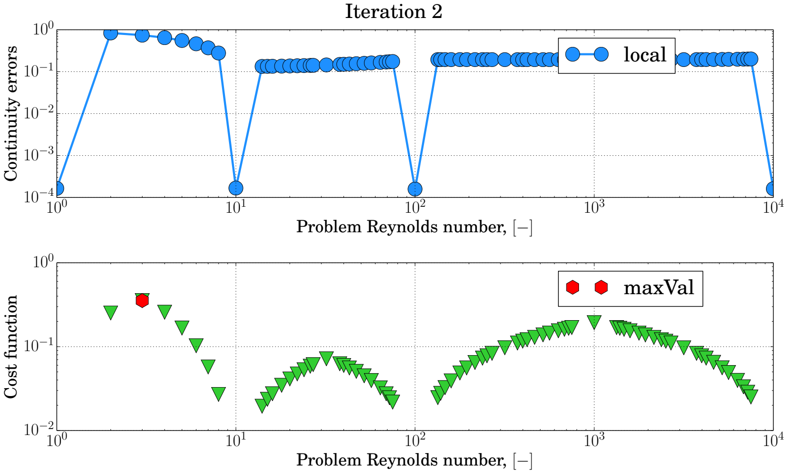
ROM size reduction: $\mathcal{S}^e$ selection based on greedy algorithm



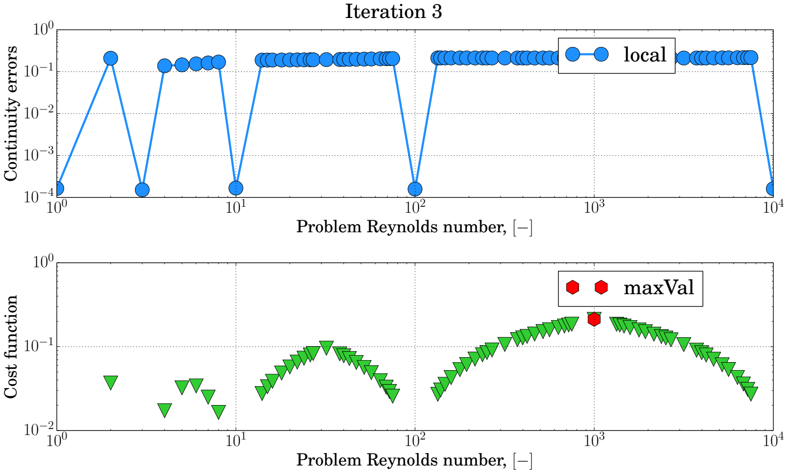
ROM size reduction: $\mathcal{S}^e$ selection based on greedy algorithm



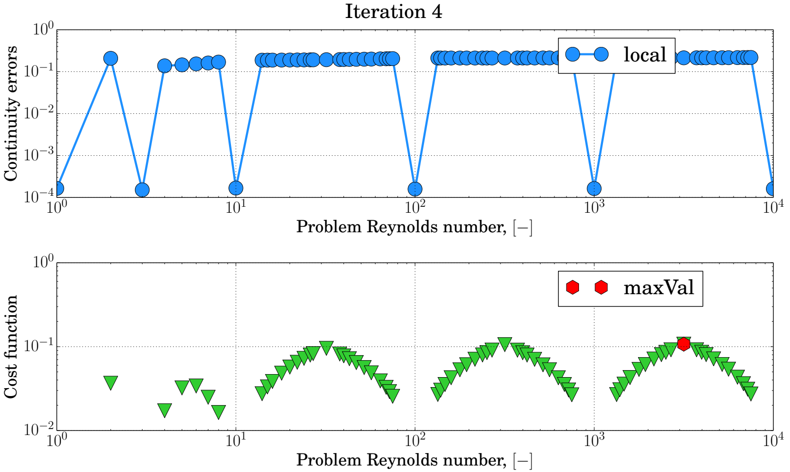
ROM size reduction: $\mathcal{S}^e$ selection based on greedy algorithm



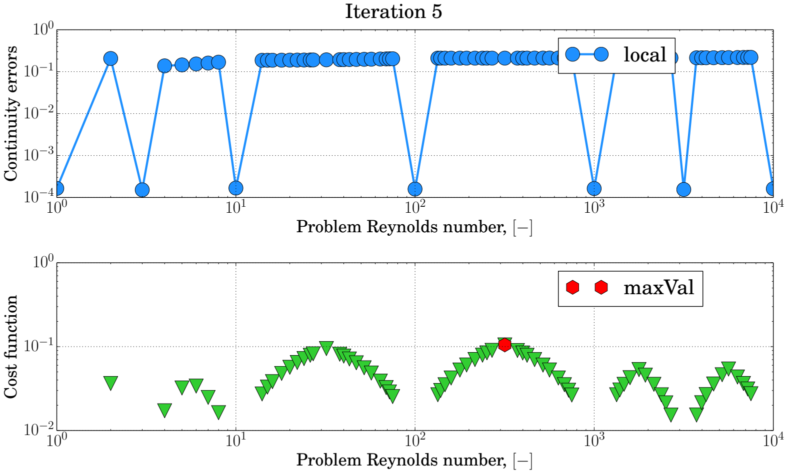
ROM size reduction: $\mathcal{S}^e$ selection based on greedy algorithm



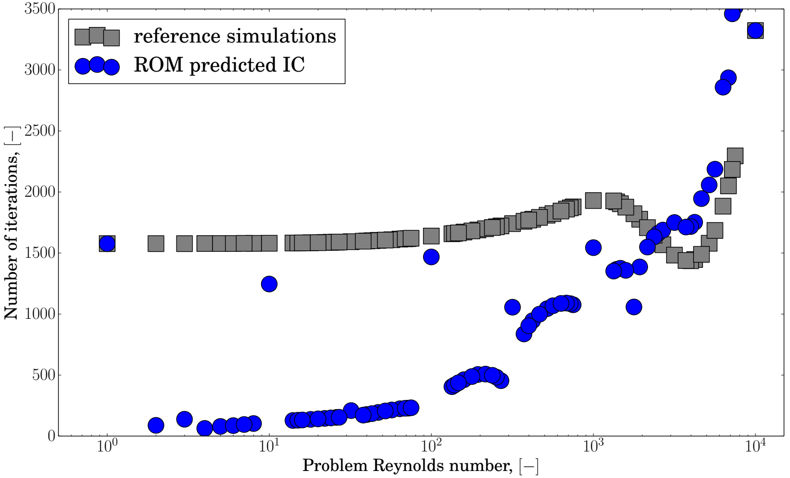
ROM size reduction: $\mathcal{S}^e$ selection based on greedy algorithm



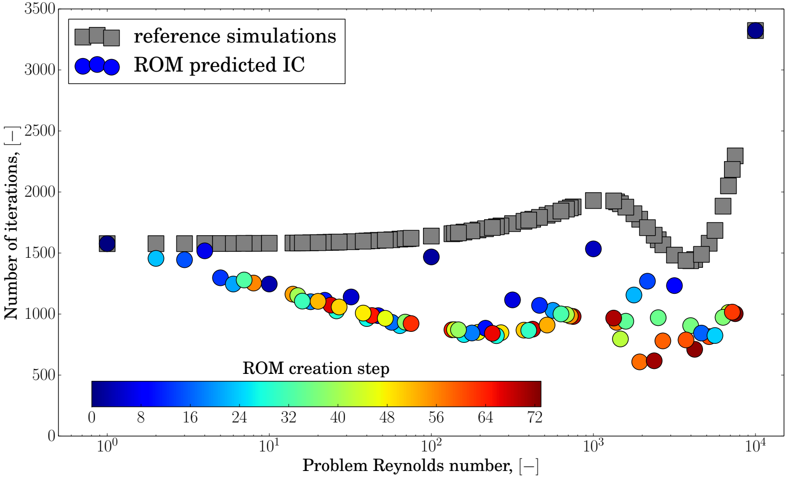
ROM size reduction: $\mathcal{S}^e$ selection based on greedy algorithm



ROM size reduction: $\mathcal{S}^e$ selection based on greedy algorithm



ROM size reduction: $\mathcal{S}^e$ selection based on greedy algorithm



Natural weights for FVM problems

Introduction of the L^2-norm weighted inner product

- Let us have rather nice functions defined on a nice domain,

$$\varphi, \tilde{\varphi} \in L^2(\Omega), \quad \Omega \subset \mathbb{R}^3 \ldots \text{bounded, connected, } \ldots$$

- A brief reminder,

$$\langle \varphi, \tilde{\varphi} \rangle_{L^2(\Omega)} = \int_\Omega \varphi \tilde{\varphi} \,\mathrm{d}x, \quad ||\varphi||_{L^2(\Omega)} = \sqrt{\langle \varphi, \varphi \rangle_{L^2(\Omega)}}$$

- Denote Ω^h a FVM discretization of Ω and $\delta\Omega_i^h$ the volume of the i-th cell,

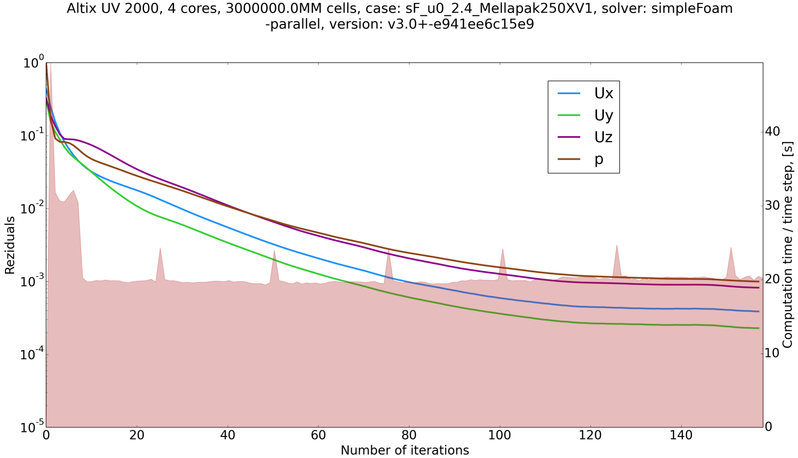
$$\Omega \approx \Omega^h = \bigcup_{i=1}^{\texttt{nCells}} \Omega_i^h, \quad V(\Omega) \approx V(\Omega^h) = \sum_{i=1}^{\texttt{nCells}} \delta\Omega_i^h$$

- Introduce a discrete inner product, $\langle \varphi, \tilde{\varphi} \rangle_{L_h^2}$,

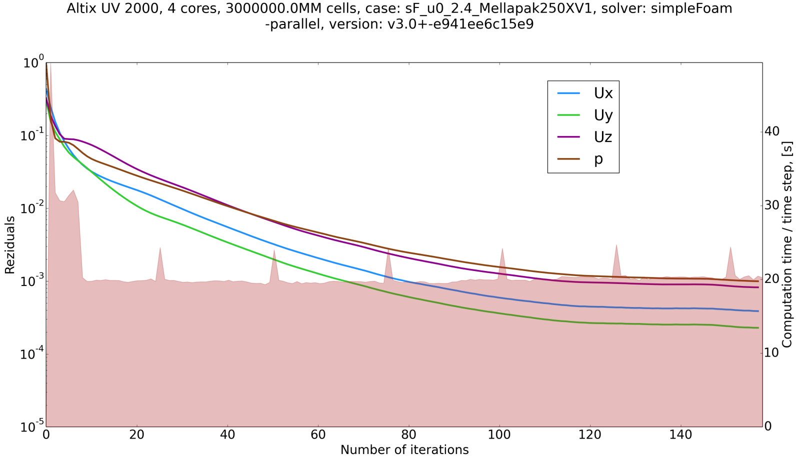
$$\langle \varphi, \tilde{\varphi} \rangle_{L^2(\Omega)} = \int_\Omega \varphi \tilde{\varphi} \,\mathrm{d}x \approx \sum_{i=1}^{\texttt{nCells}} \int_{\Omega_i^h} \varphi \tilde{\varphi} \,\mathrm{d}x = \sum_{i=1}^{\texttt{nCells}} \varphi_i^h \tilde{\varphi}_i^h \delta\Omega_i^h = \langle \varphi, \tilde{\varphi} \rangle_{L_h^2}$$

- Denote $W = \mathrm{diag}(\delta\Omega_1^h, \ldots, \delta\Omega_{\texttt{nCells}}^h)$. Hence, $\langle \varphi, \tilde{\varphi} \rangle_{L_h^2} = (\varphi^h)^{\mathrm{T}} W \varphi^h$.

Full case: Residuals evolution, from potentialFoam initialized fields



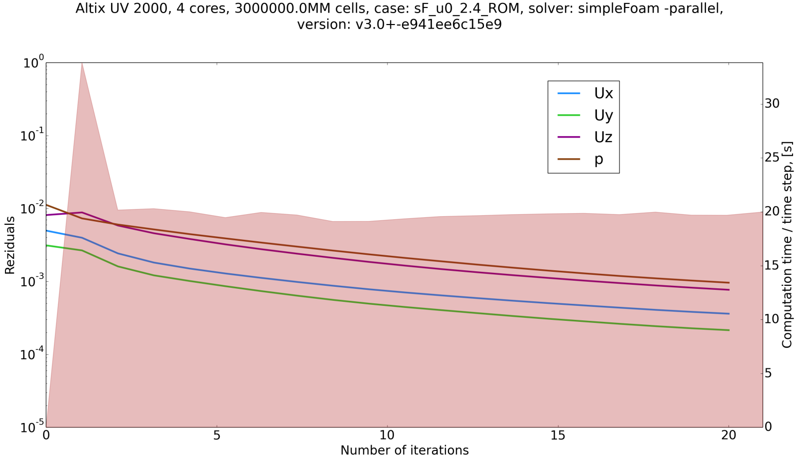
Full case: Residuals evolution, from potentialFoam initialized fields



Residuals evolution

Comparison of the residuals evolution for Mellapak cases in L0,L1 and L2

Full case: Residuals evolution, from ROM predicted fields, L1



Residuals evolution

Comparison of the residuals evolution for Mellapak cases in L0,L1 and L2

Full case: Residuals evolution, from ROM predicted fields, L2