POD-DEIM based ROMs for CFD applications

Martin Isoz[1],
Michael Hinze[2]

[1] UCT Prague, Department of mathematics
[2] University Hamburg, Department of mathematics

MORE seminar,
Prague November 7 2016

Outline

From the research motivation to results

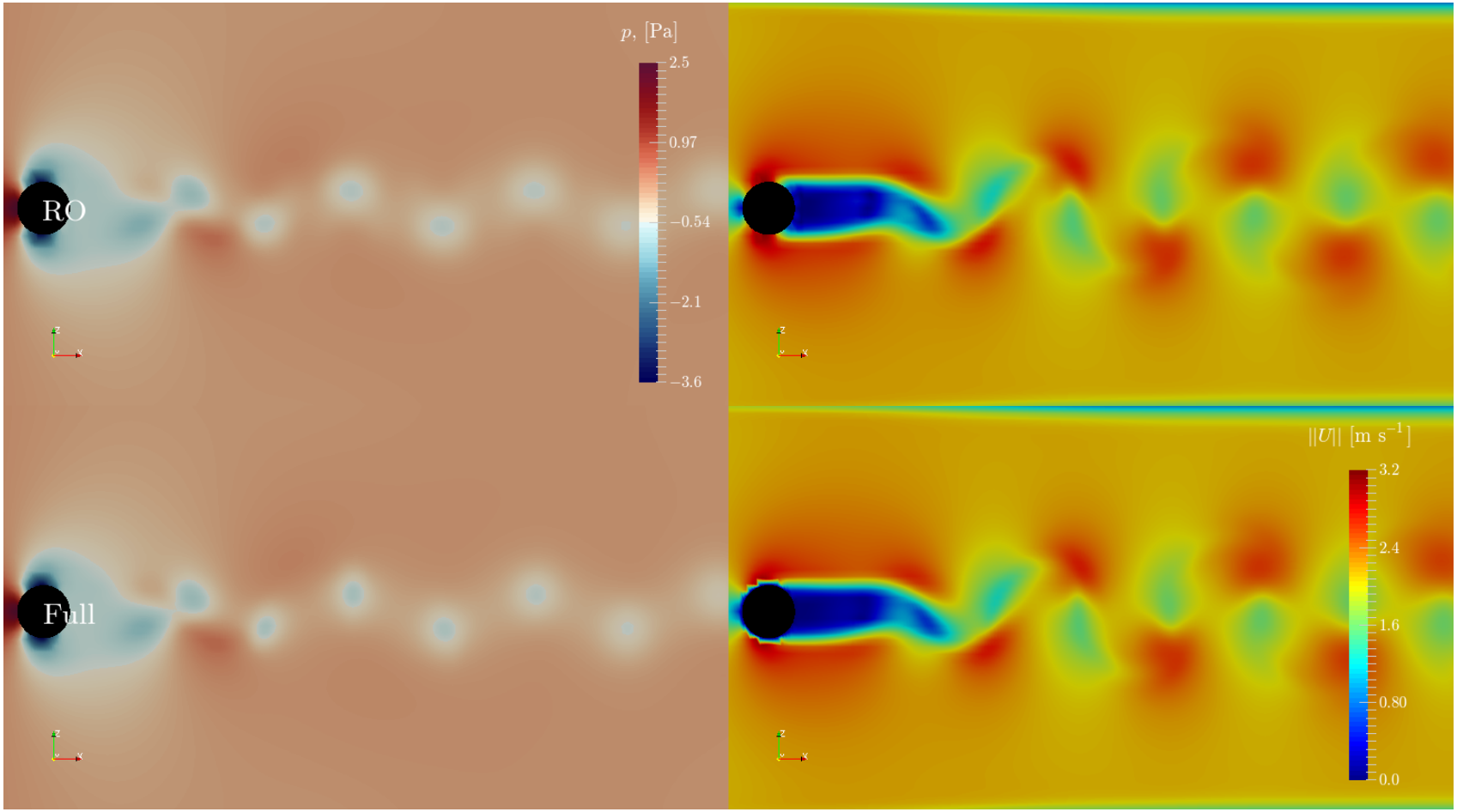
UNIVERSITY OF CHEMISTRY AND TECHNOLOGY PRAGUE

Introduction

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

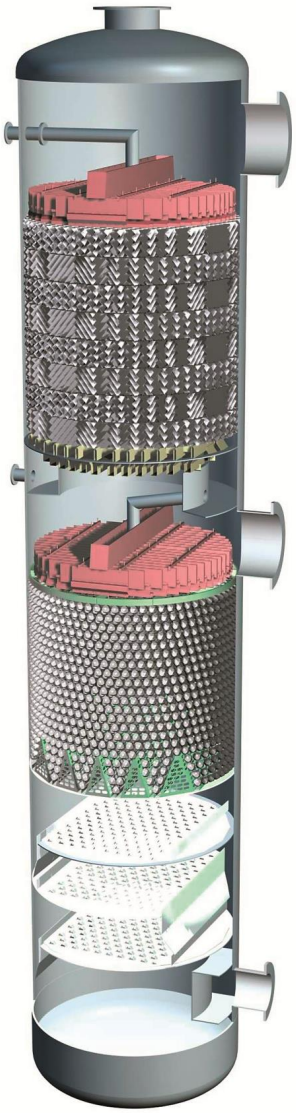
Research motivation

Reducing the computational cost of modeling of complex systems



Real-life applications

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



[Sulzer ChemTech]

Importance

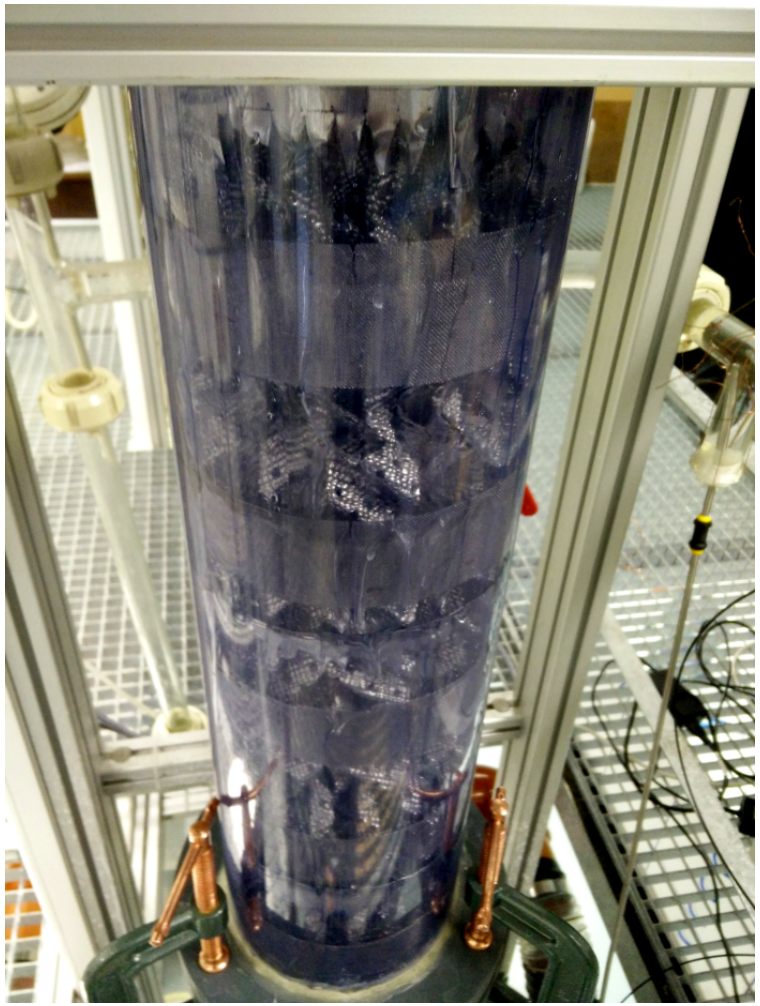
- Chemical industry creates mixtures but sells ”pure species” (e.g. oil)
- 2014, 3% of energy consumption of the USA was due to the separation columns

Challenges

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

Packed column

Complex multiphase flow



Separation columns modeling

Multiphase hydrodynamics, simultaneous mass and heat transfer, no chemical reaction

Momentum and continuity equations, N phases

$$\rho_i \frac{\partial}{\partial t}(u_i) + \nabla \cdot (\rho_i u_i \otimes u_i) = \nabla \cdot \tau + F_i, \quad i = 1, \ldots, N$$

$$\rho_i = \rho(c_i, T_i), \quad \nabla \cdot (u_i) = S_i^\rho = \sum_{j=1}^{M} \hat{R}_{i,j}^c$$

Mass transfer (no reaction), M species

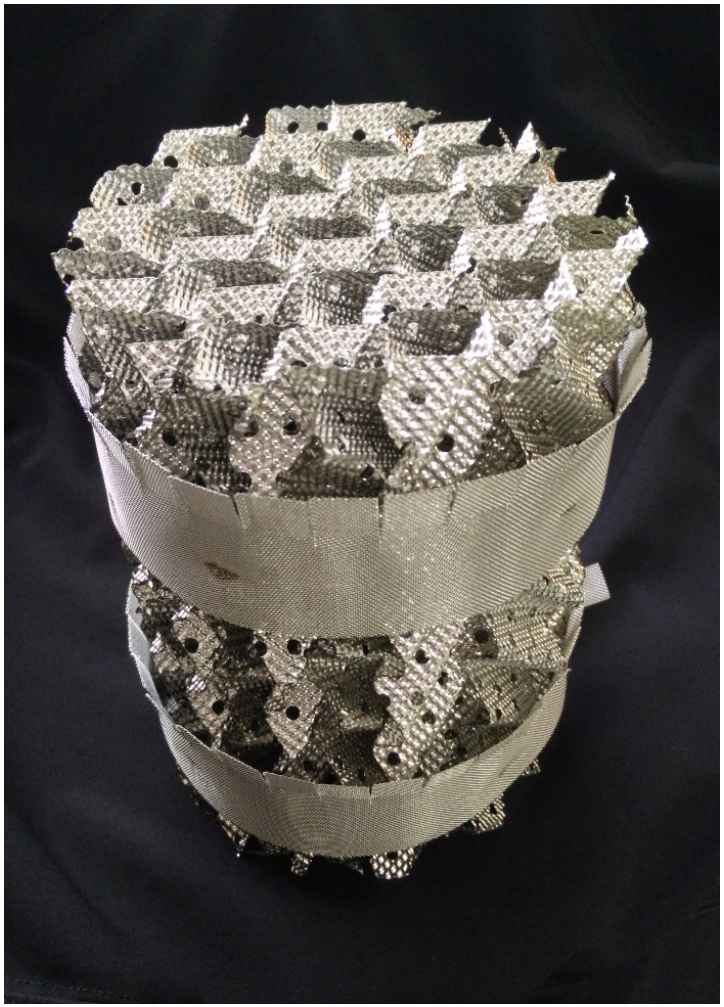
$$\frac{\partial}{\partial t} c_{i,j} + \nabla \cdot (u_i \, c_{i,j}) = \nabla \cdot \left(\Gamma_{i,j}^c \, \nabla c_{i,j}\right) + S_{i,j}^c, \qquad j = 1, \ldots, M$$

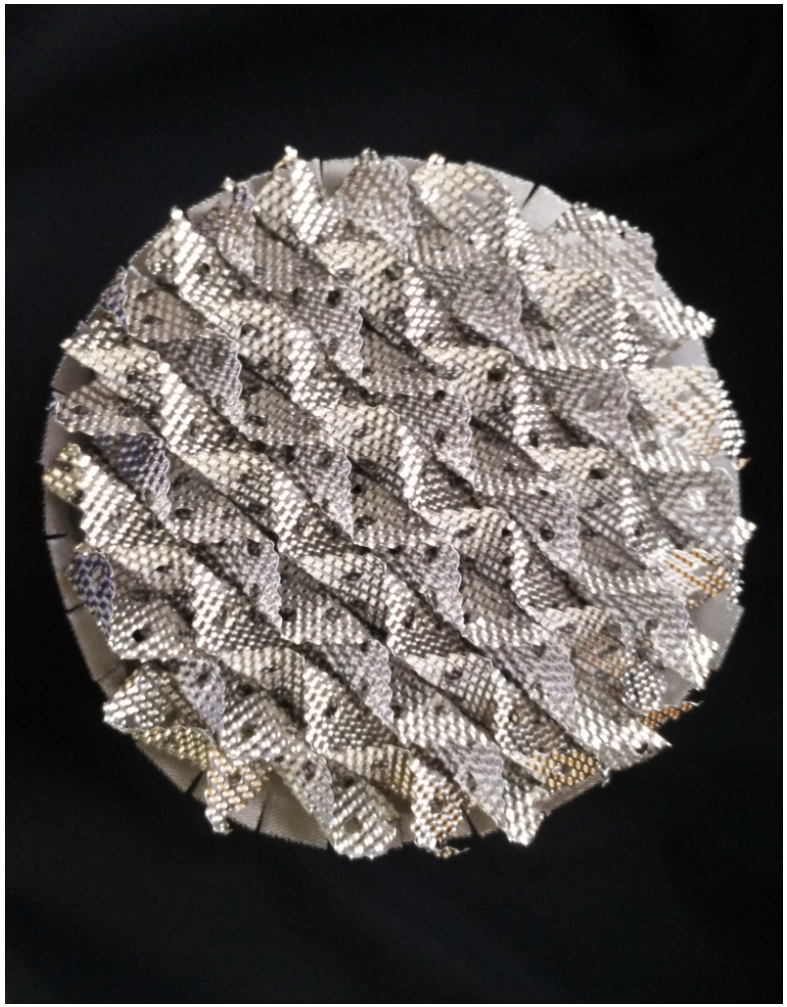
Heat transfer (no reaction), N phases

$$\frac{\partial}{\partial t} T_i + \nabla \cdot (u_i \, T_i) = \nabla \cdot \left(\Gamma_i^T \, \nabla T_i\right) + S_i^T, \qquad i = 1, \ldots, N$$

Structured packing

Curled, textured, perforated plate





UNIVERSITY OF CHEMISTRY AND TECHNOLOGY PRAGUE

POD & DEIM

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Problem setting

Modeling of large systems of differential equations

Original system

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

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

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

Reduced-order system

$$\dot{\eta}^l = A^l \eta^l + f^l(t, \eta^l), \quad \eta^l(t) \in \mathbb{R}^l, \quad \eta^l(0) = \eta^l_0,\, t \in [0, T],$$

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

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

gain $\quad \ldots \quad l \ll m$

Proper Orthogonal Decomposition

SVD based technique for model order reduction

Algorithm 1 POD basis of rank l with weighted inner product

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

1: Set $Y = [y_1, \ldots, y_n] \in \mathbb{R}^{m\times n}$;
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 $\varepsilon(l) = \sum_{i=1}^{l} \sigma_i / \sum_{i=1}^{d} \sigma_i$;
6: Truncate $\bar{\Psi} \leftarrow [\bar{\psi}_1, \ldots, \bar{\psi}_l] \in \mathbb{R}^{m\times l}$;
7: Compute $\Psi = W^{-1/2}\bar{\Psi} \in \mathbb{R}^{m\times l}$;
8: **return** POD basis, Ψ, and ratio $\varepsilon(l)$

Notes:

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

Discrete Empirical Interpolation Method

POD & Greedy algorithm based method for handling non-linearities

Algorithm 2 DEIM

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

1: Compute POD basis $\Phi = [\phi_1, \ldots, \phi_p]$ for F
2: $\mathrm{idx} \leftarrow \arg\max_{j=1,\ldots,m} |(\phi_1)_{\{j\}}|$;
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: $\quad \mathrm{idx} \leftarrow \arg\max_{j=1,\ldots,m} |(r)_{\{j\}}|$;
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.

Introduce the Galerking ansatz and Fourier modes

- Prerequisities: $\Psi \in \mathbb{R}^{m\times l}, \Phi \in \mathbb{R}^{m\times p}, W \in \mathbb{R}^{m\times m}, \vec{i} \in \mathbb{R}^p$
- Quick note,

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

$$y(t) = \sum_{j=1}^{d} \langle y(t), \psi_j \rangle_W \, \psi_j, \, \forall t \in [0,T], \quad d = \mathrm{rank}(Y)$$

- Ansatz for Galerkin projection, $l < d$

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

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

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

Introduce the reduced-order model

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

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

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

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

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

Deal with the non-linearities I

- Identify the problem,

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

- Approximate the non-linearities via the POD basis, Φ,

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

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

$$f^l(t,\eta^l) = \Psi^{\mathrm{T}} W f(t, \Psi\eta^l) = \Psi^{\mathrm{T}} W b(t) \approx \Psi^{\mathrm{T}} W \Phi c(t), \quad c(t) \in \mathbb{R}^p$$

- Plug-in the last output of the DEIM algorithm, $\vec{i}$

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

Deal with the non-linearities II (yes, almost done)

- Plug in the matrix P,

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

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

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

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

- Write the final ROM

$$\dot{\eta}^l = A^l\eta^l + f^l(t,\eta^l), \text{ for } t \in (0,T], \quad \eta^l(0) = \eta_0^l,$$

where

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

UNIVERSITY OF CHEMISTRY AND TECHNOLOGY PRAGUE

Link with OpenFOAM

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

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, $\{(y_i, A_i, b_i)\}_{i=1}^n$, $A_i \in \mathbb{R}^{m\times m}$, $i = 1, \ldots, m$ 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,

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

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

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

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

Example 1 – Passive scalar advection

Phase-volume fraction advection in multiphase flow

interFoam – Volume-of-Fluid model for multiphase flow

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

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

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

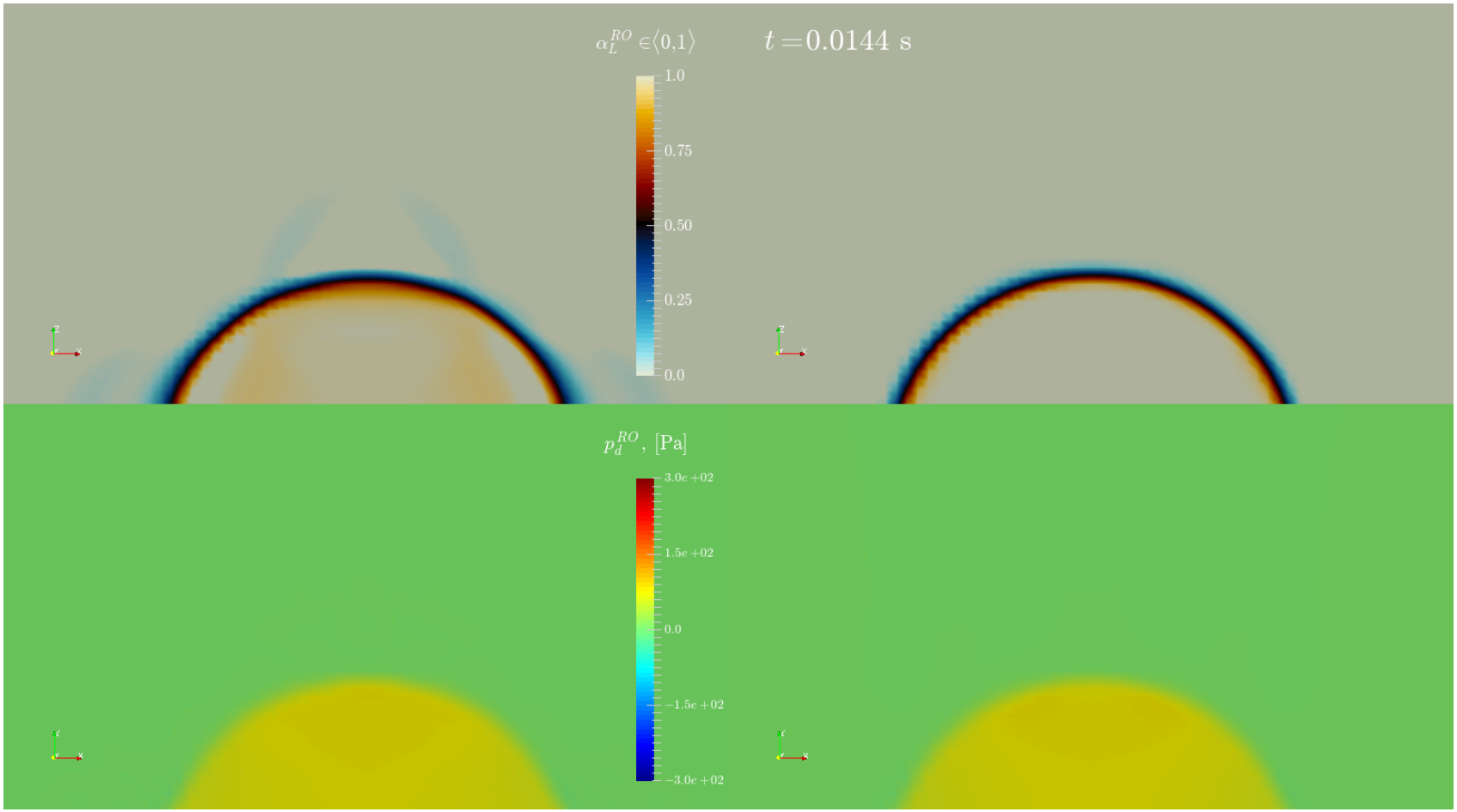
Example of implementation in OpenFOAM

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

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

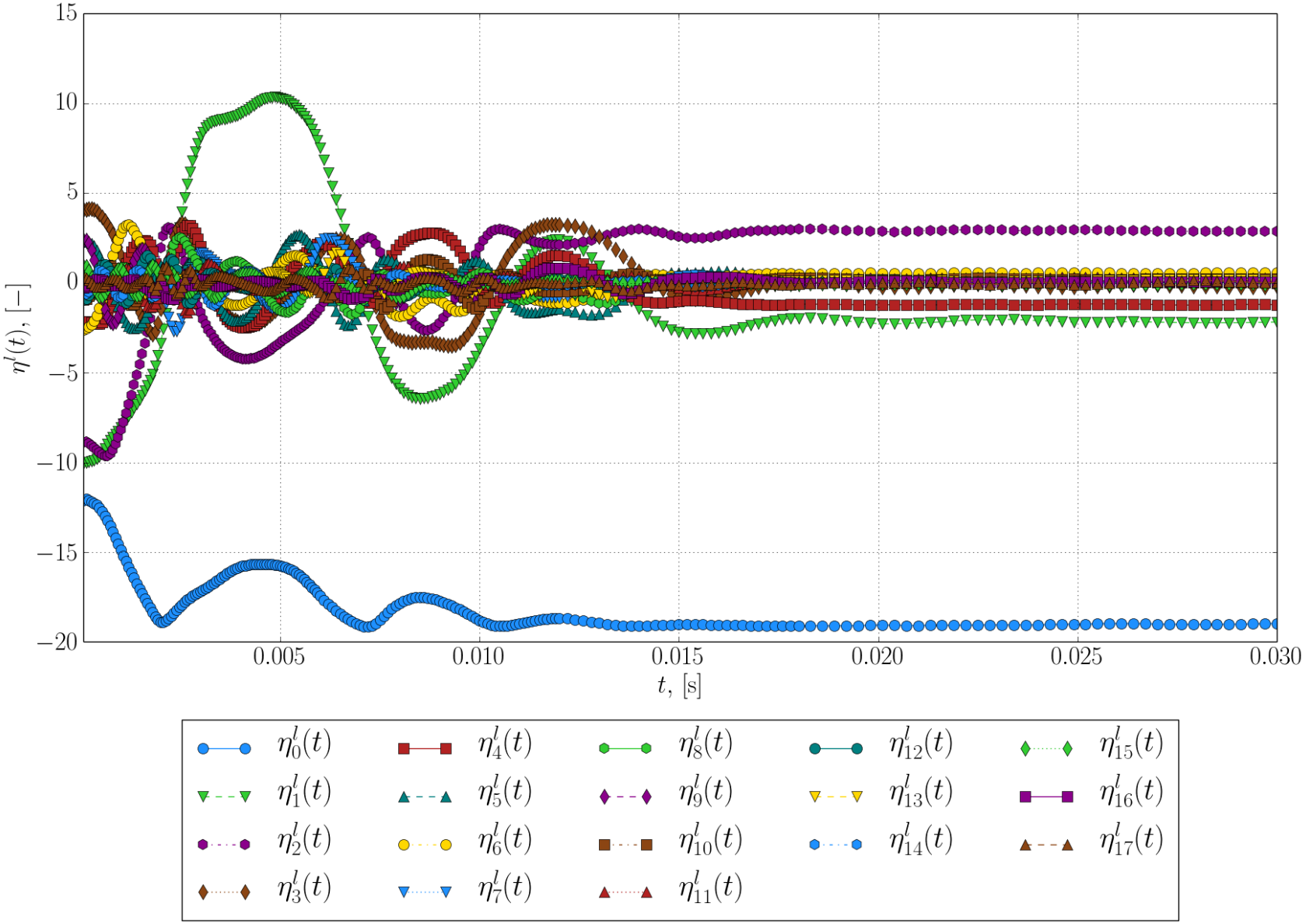
Example 1 – Passive scalar advection

Numerical results



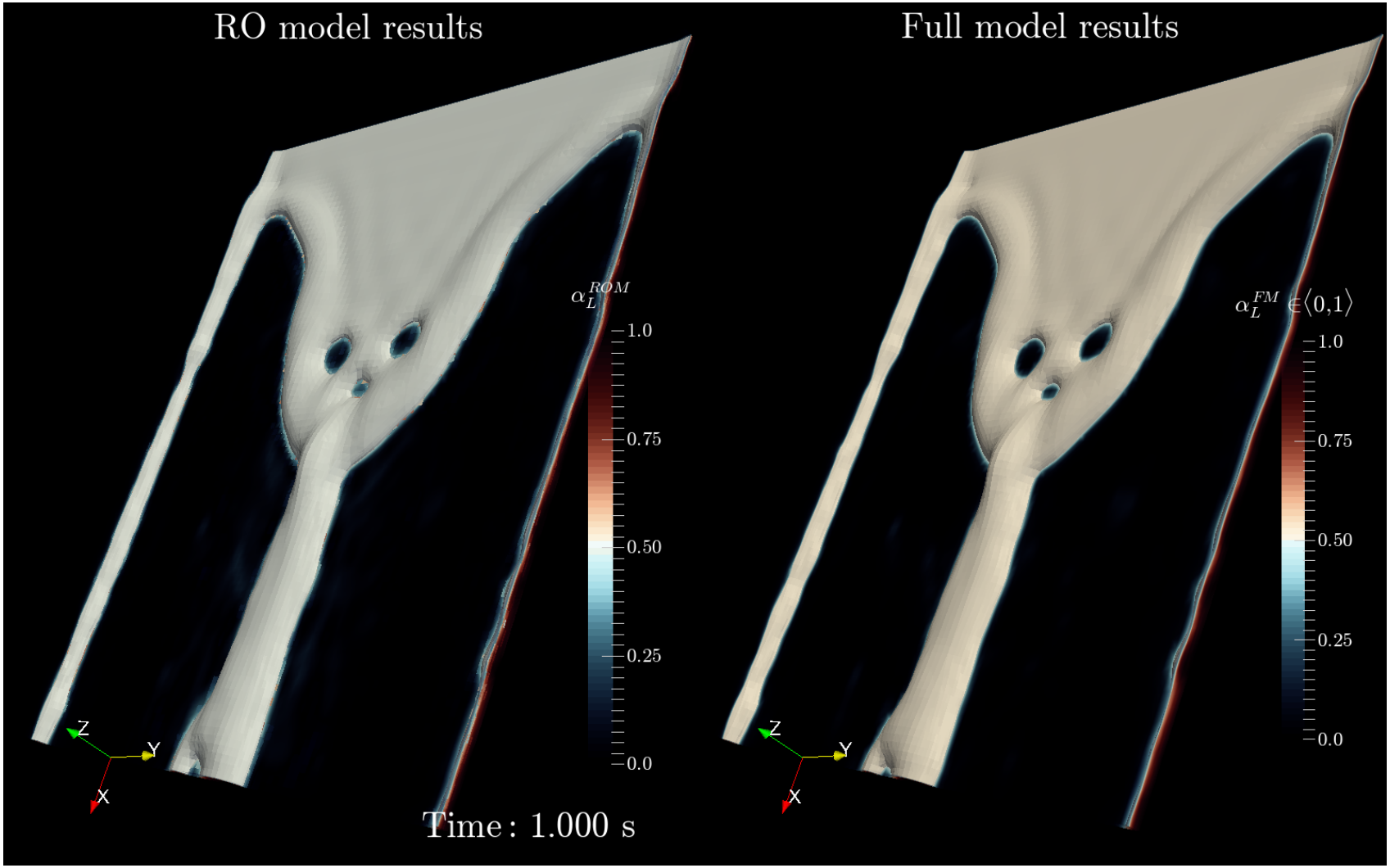
Example 1 – Passive scalar advection

Numerical results



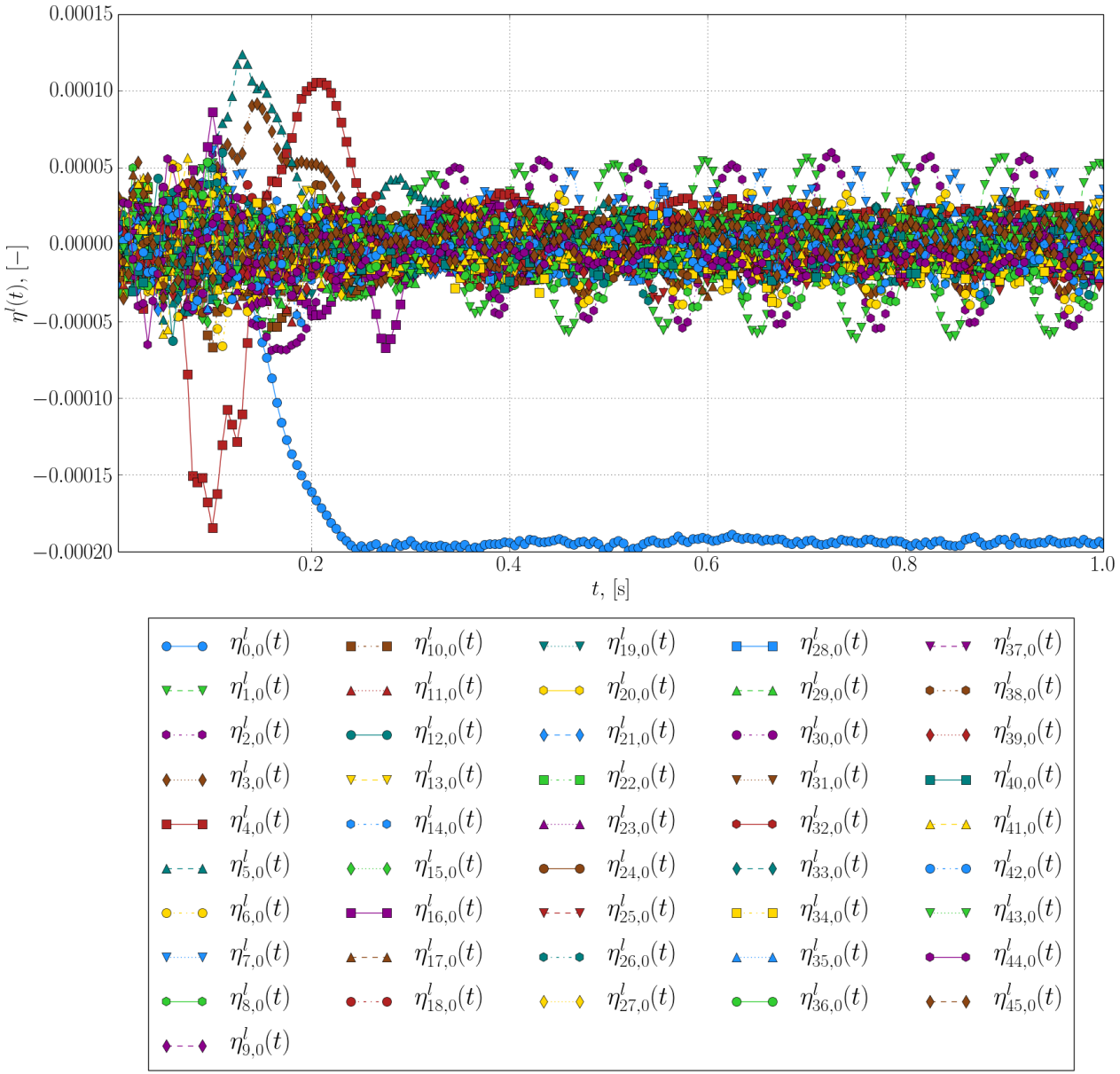
Example 1 – Passive scalar advection

Numerical results



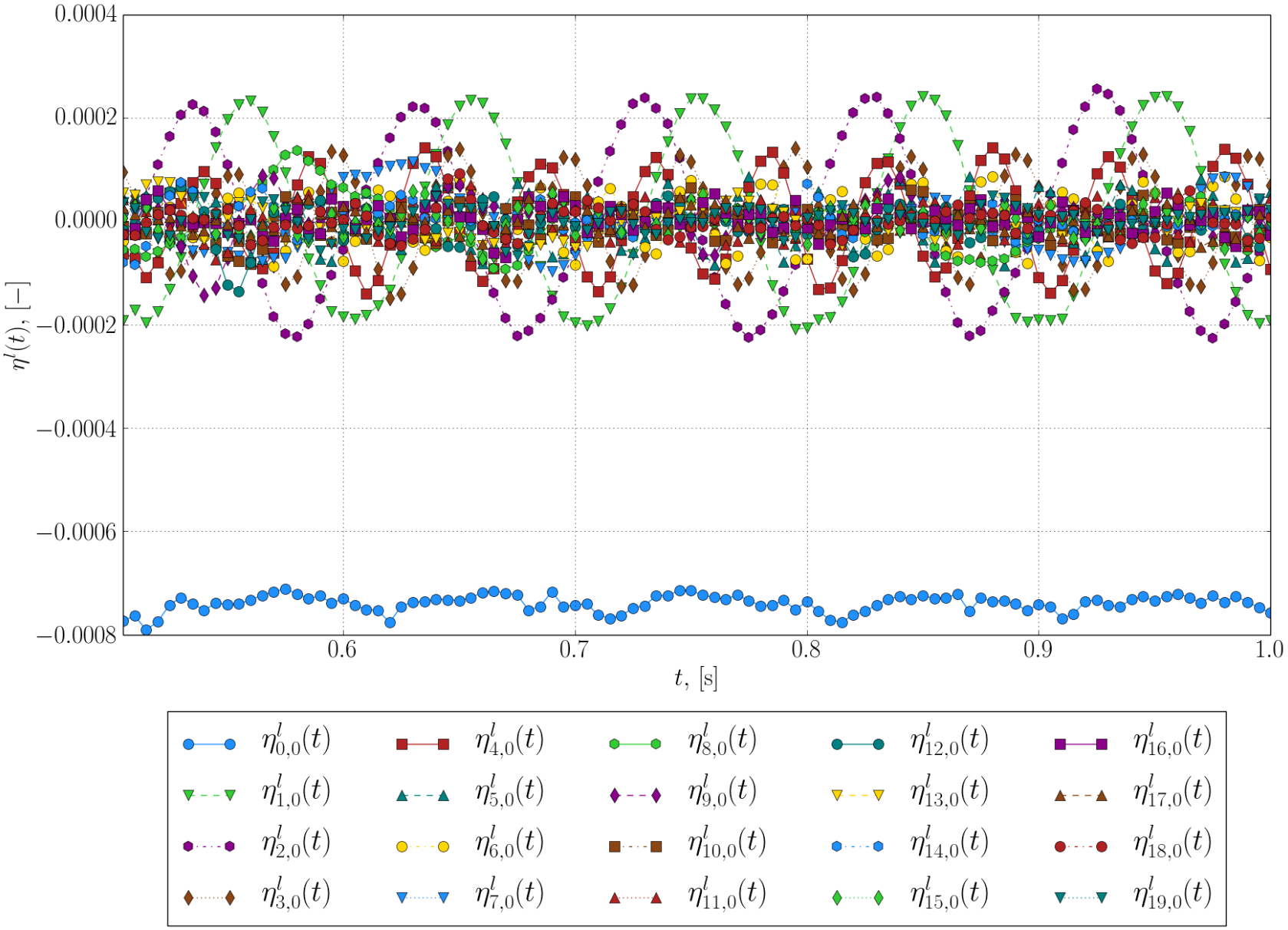
Example 1 – Passive scalar advection

Numerical results



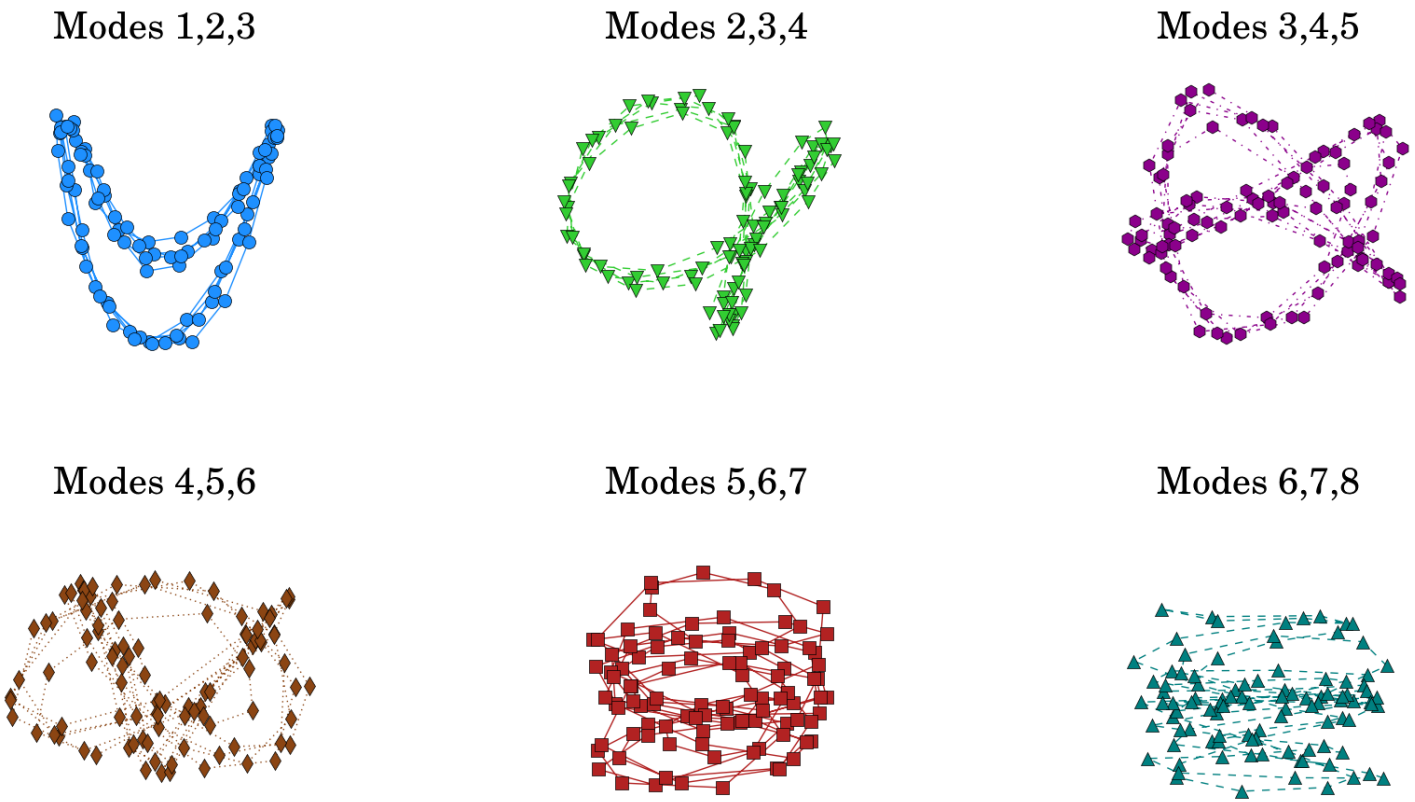
Example 1 – Passive scalar advection

Numerical results



Example 1 – Passive scalar advection

Numerical results



POD-DEIM ROMs for Navier-Stokes equations

A brief discussions of specifics linked to solving NS in OpenFOAM (or other FVM solver)

Navier-Stokes equations for incompressible fluids

$$
\begin{aligned}
u_t + \nabla \cdot (u \otimes u) - \nabla \cdot (\nu \nabla u) &= -\nabla \tilde{p} + \tilde{f} \\
\nabla \cdot u &= 0
\end{aligned}
$$

$$u = G(u),\, p = H(p) \quad \forall\, (t, x) \in I \times \partial\Omega$$

$$u = u_0,\, p = p_0 \quad \forall\, (0, x) \in \Omega$$

Wanted: $\dot{y}_u = A_u(t) y_u + b_u(t, y_u)$

Example of implementation in OpenFOAM

```
    fvm::div(phi, U)
  + turbulence->divDevReff(U)
 ==
    -fvc::grad(p) + fvOptions(U)
```

Link: `fvm` $\rightarrow A_u(t)$, `fvc` $\rightarrow b_u(t, y_u)$, `turbulence` $\rightarrow A_u(t),\, b_u(t, y_u)$

Specifics of p-U coupling in OpenFOAM

OpenFOAM uses a Schur complement based p-U coupling

Saddle-point problem

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

Jacobi iterations with Schur-complement based p-U coupling

$$u^* \leftarrow Au^* = f - B^{\mathrm{T}} p^{k-1}$$

$$p^k \leftarrow BD^{-1}B^{\mathrm{T}}p^k = BD^{-1}\left(f - (L+U)u^*\right)$$

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

At convergence

$$BD^{-1}B^{\mathrm{T}}p^k = BD^{-1}\left(f - (L+U)u^*\right) \approx BA^{-1}B^{\mathrm{T}}p = BA^{-1}f$$

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

Outcome for ROM

- ”Natural” is to construct ROM for p
- For the velocity, I can choose between computational cost and consistency and accuracy

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

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

```
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

$\mathcal{S} \ldots n\,[(1+3)m + (5+5)m + (1+3)m] \approx 15nm$ 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 $\Psi^{\mathrm{T}} W \texttt{rAUMORE}_i \Psi$,

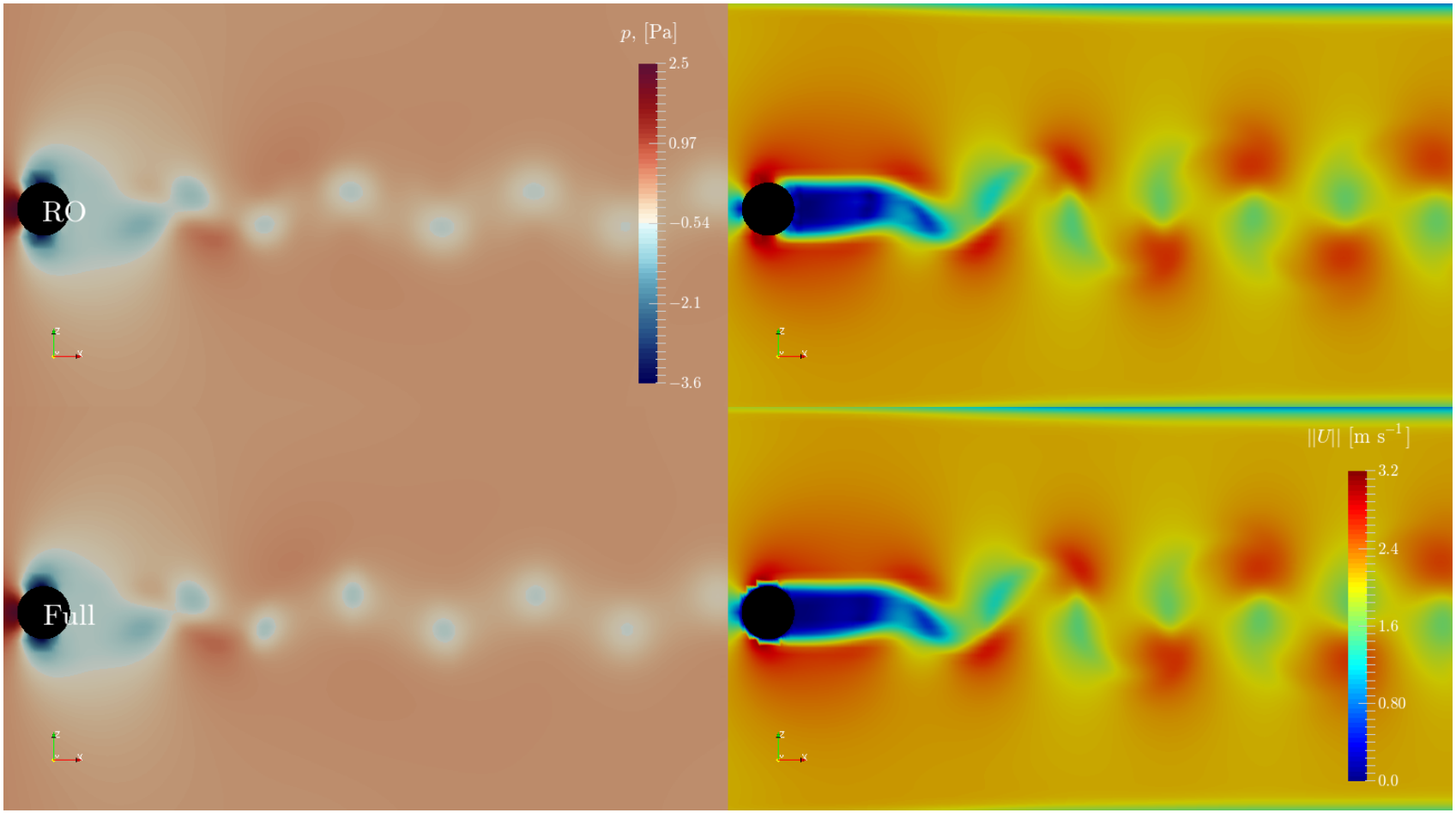
$\sim n$ calculations of $\Psi^{\mathrm{T}} W \texttt{HMOREbyAMORE}_i \Psi$,

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

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

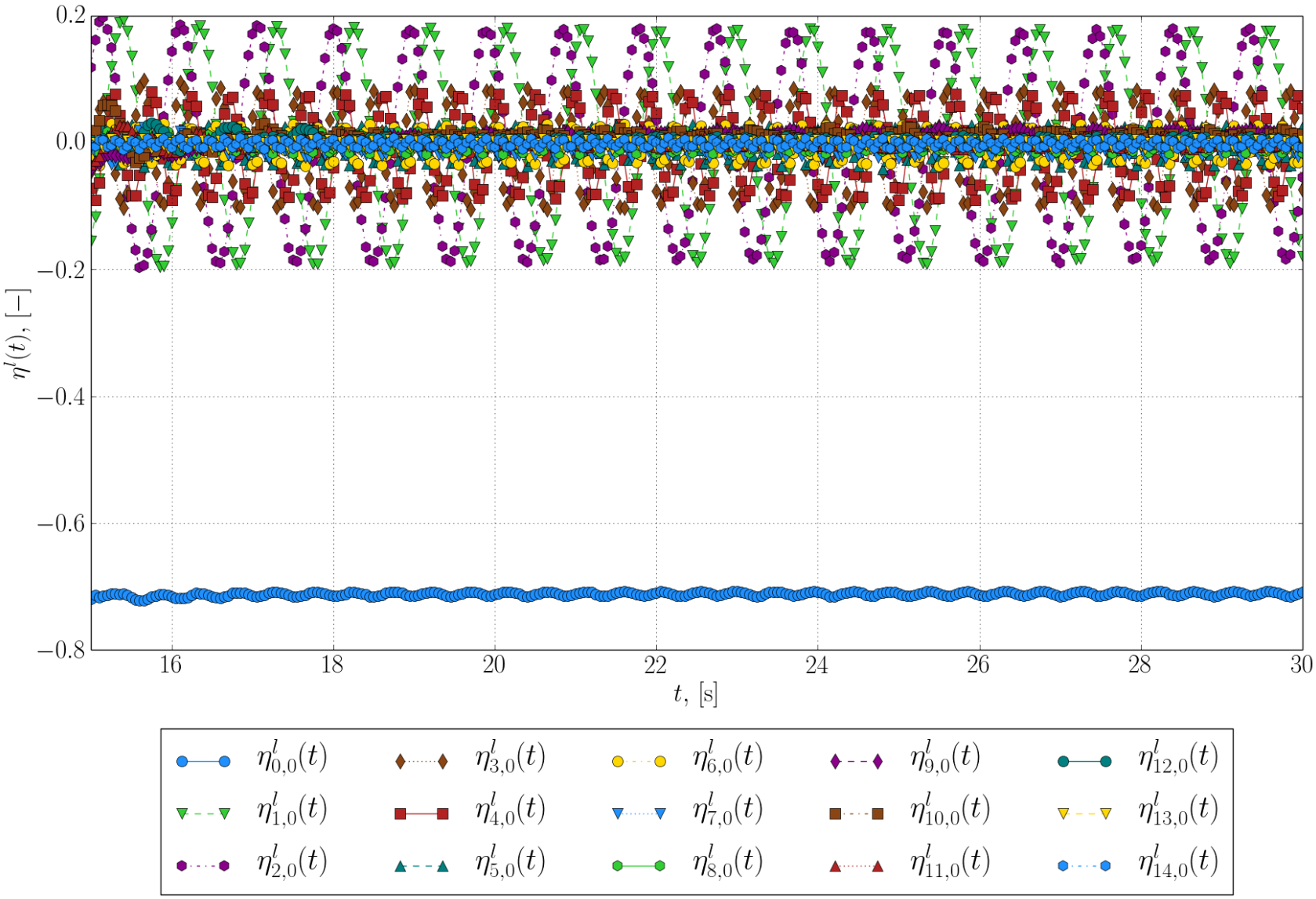
Example 2 – Von Karman vortex row

Validation of the approach



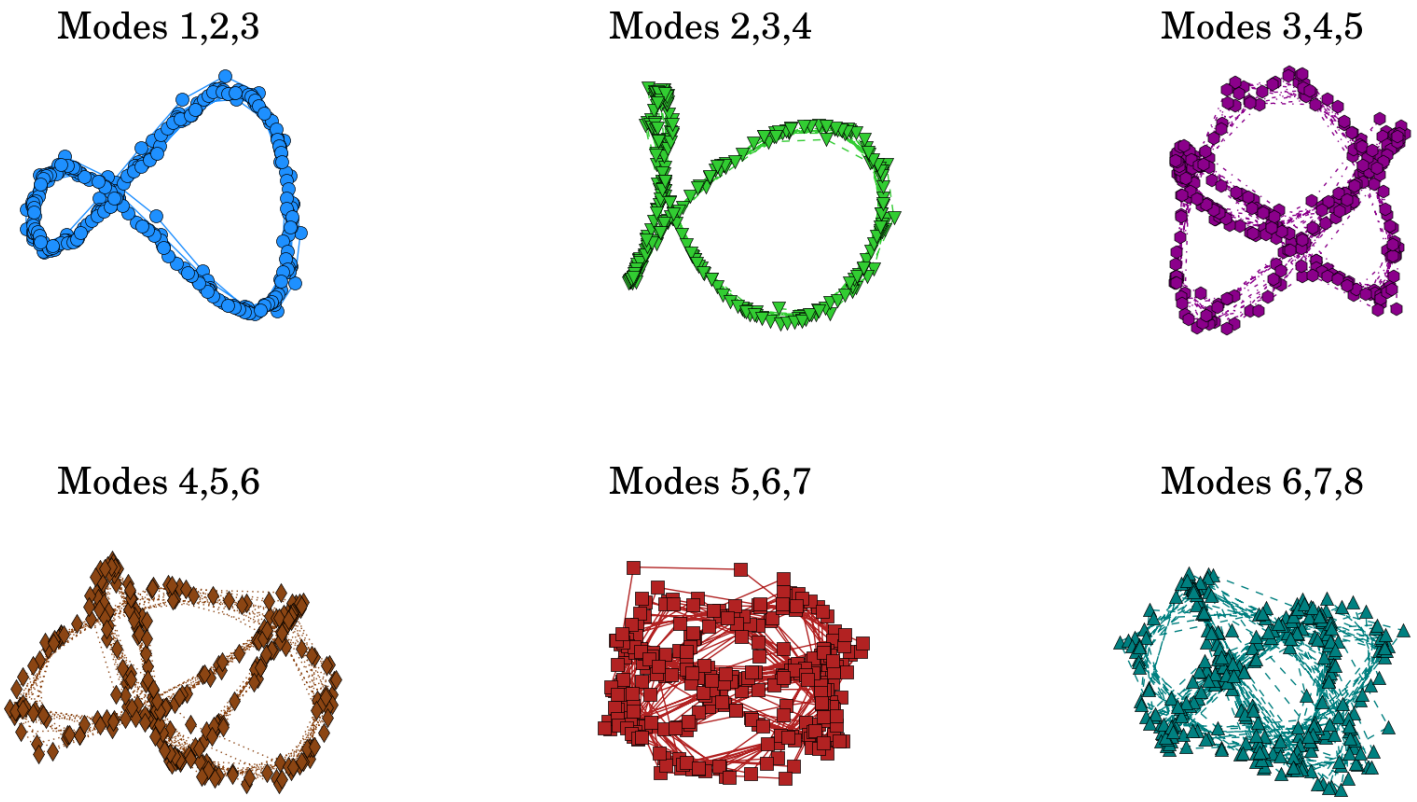
Example 2 – Von Karman vortex row

Validation of the approach



Example 2 – Von Karman vortex row

Validation of the approach



UNIVERSITY OF CHEMISTRY AND TECHNOLOGY PRAGUE

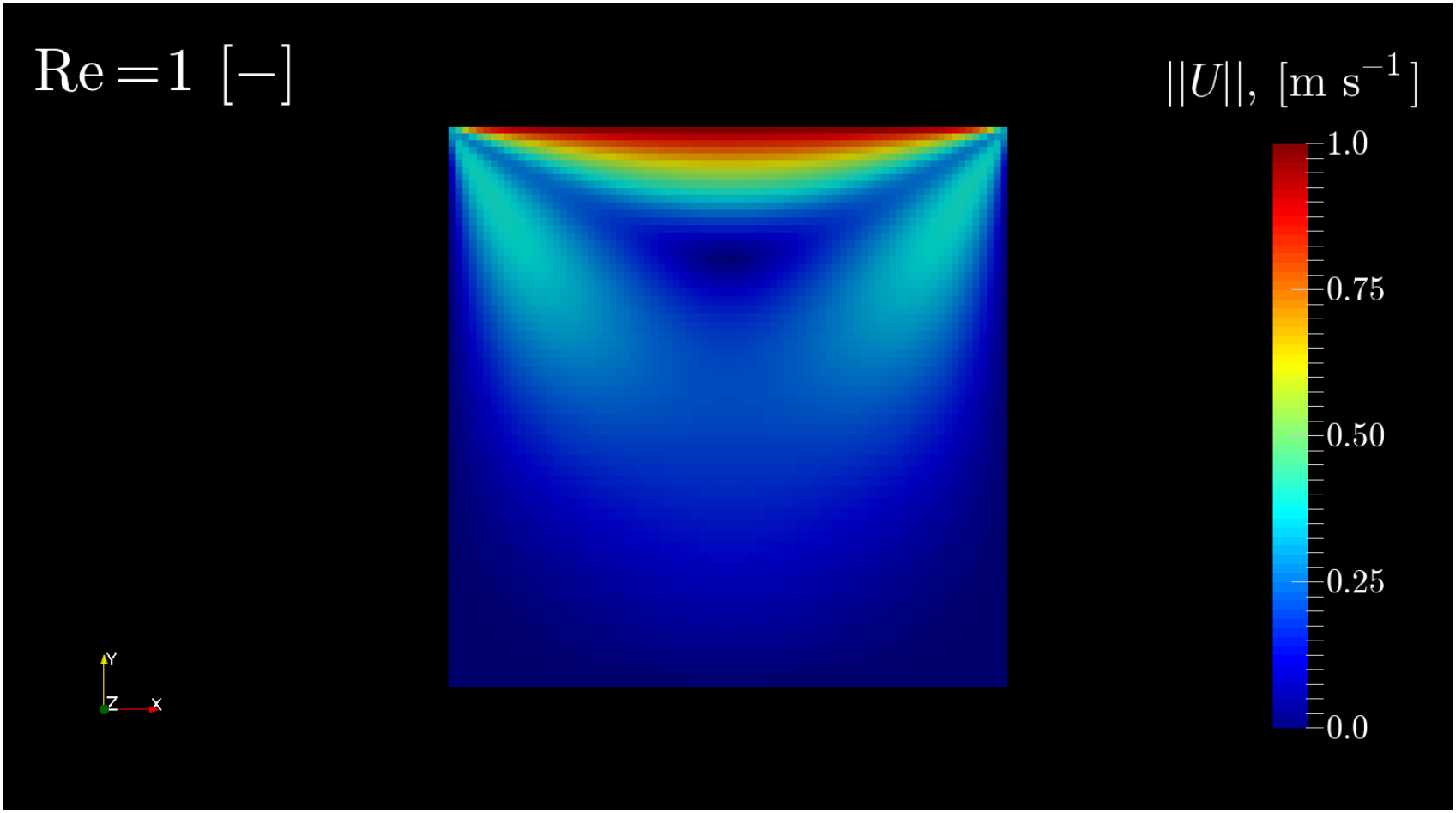
Applications

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Steady state simulations

ROM based initial guess prediction for full NS solver (simpleFoam)

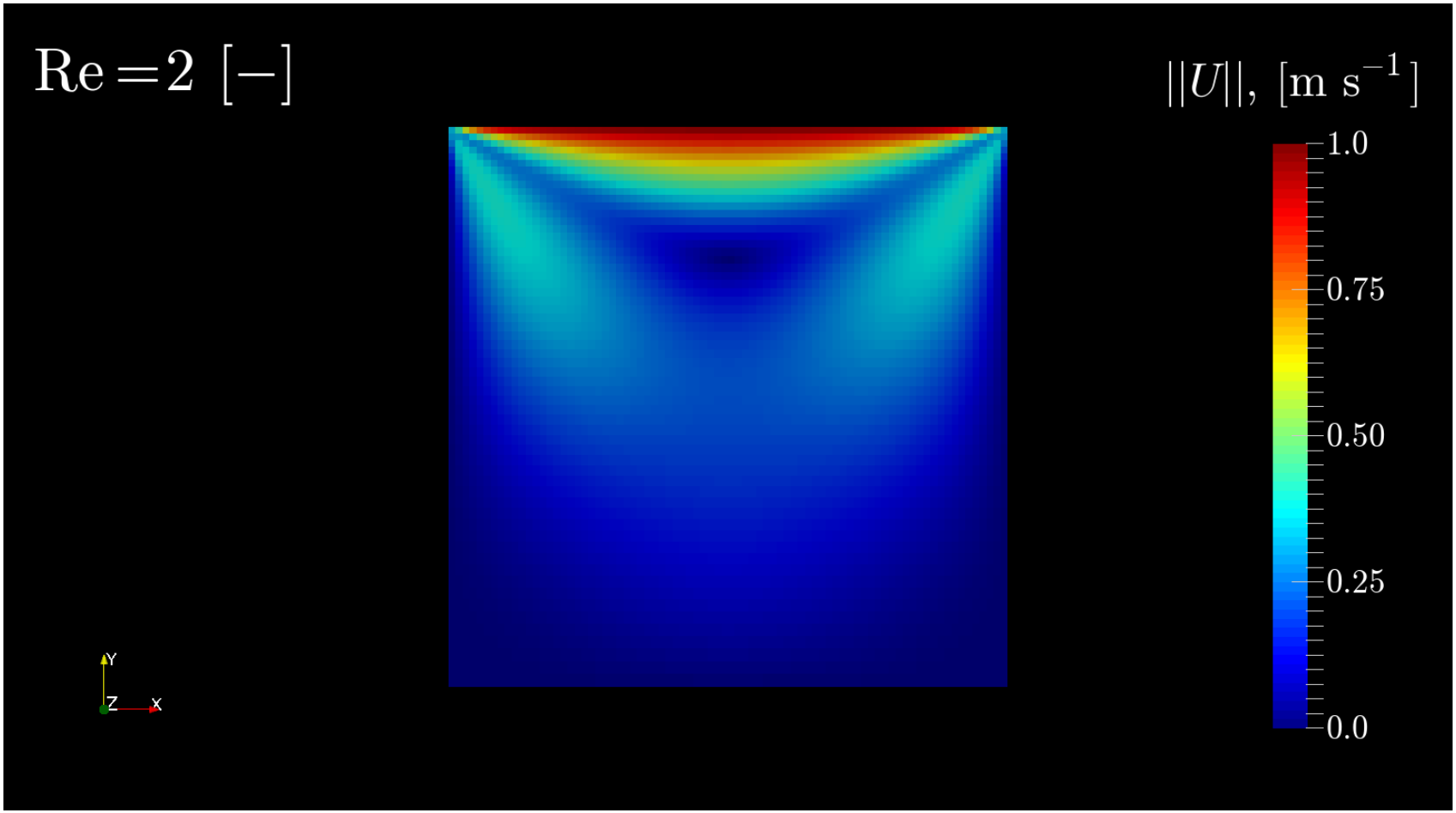
Test case: Standard OpenFOAM cavity case, variable $\mathrm{Re} = LU/\nu$



Steady state simulations

ROM based initial guess prediction for full NS solver (simpleFoam)

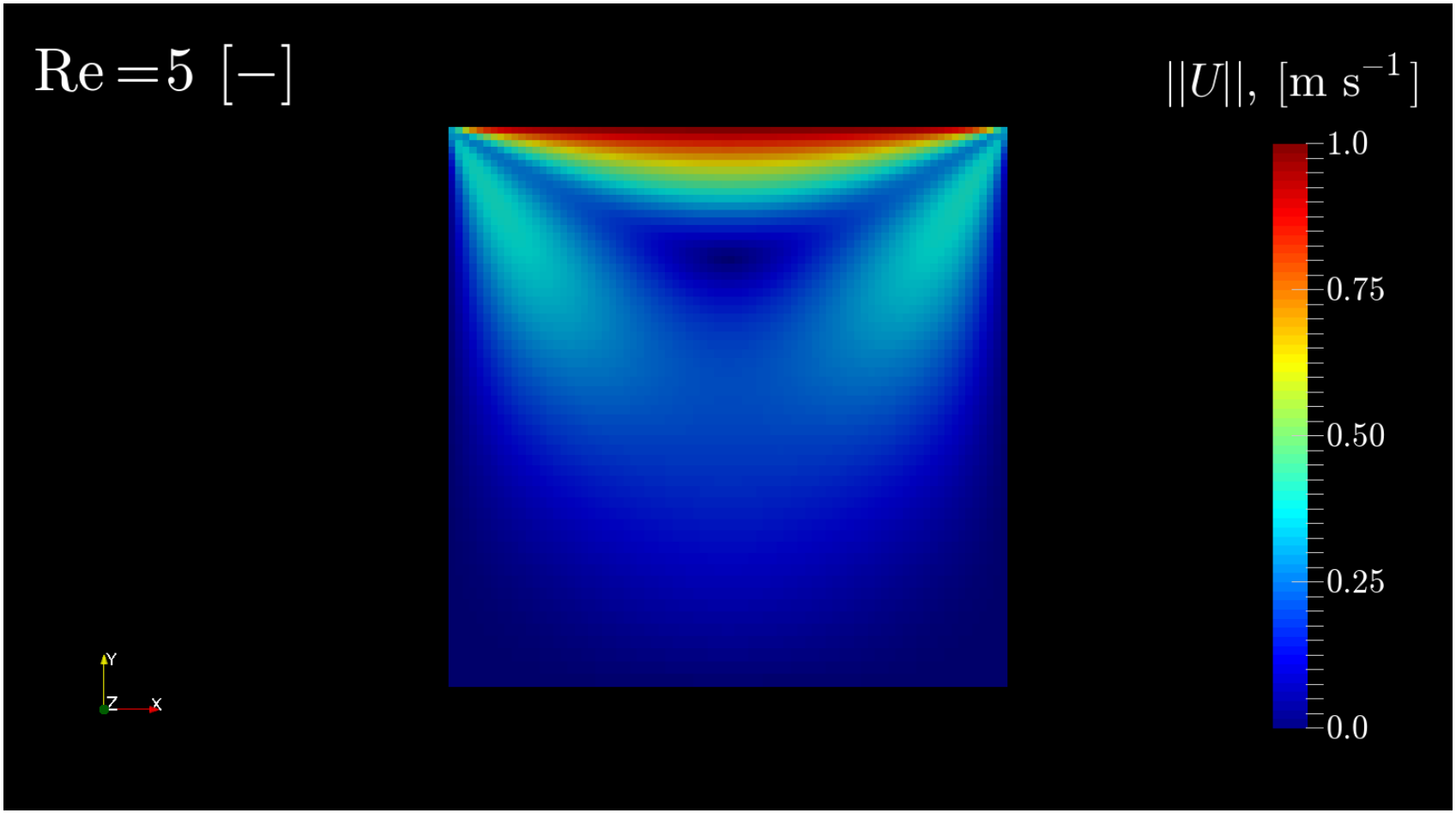
Test case: Standard OpenFOAM cavity case, variable $\mathrm{Re} = LU/\nu$



Steady state simulations

ROM based initial guess prediction for full NS solver (simpleFoam)

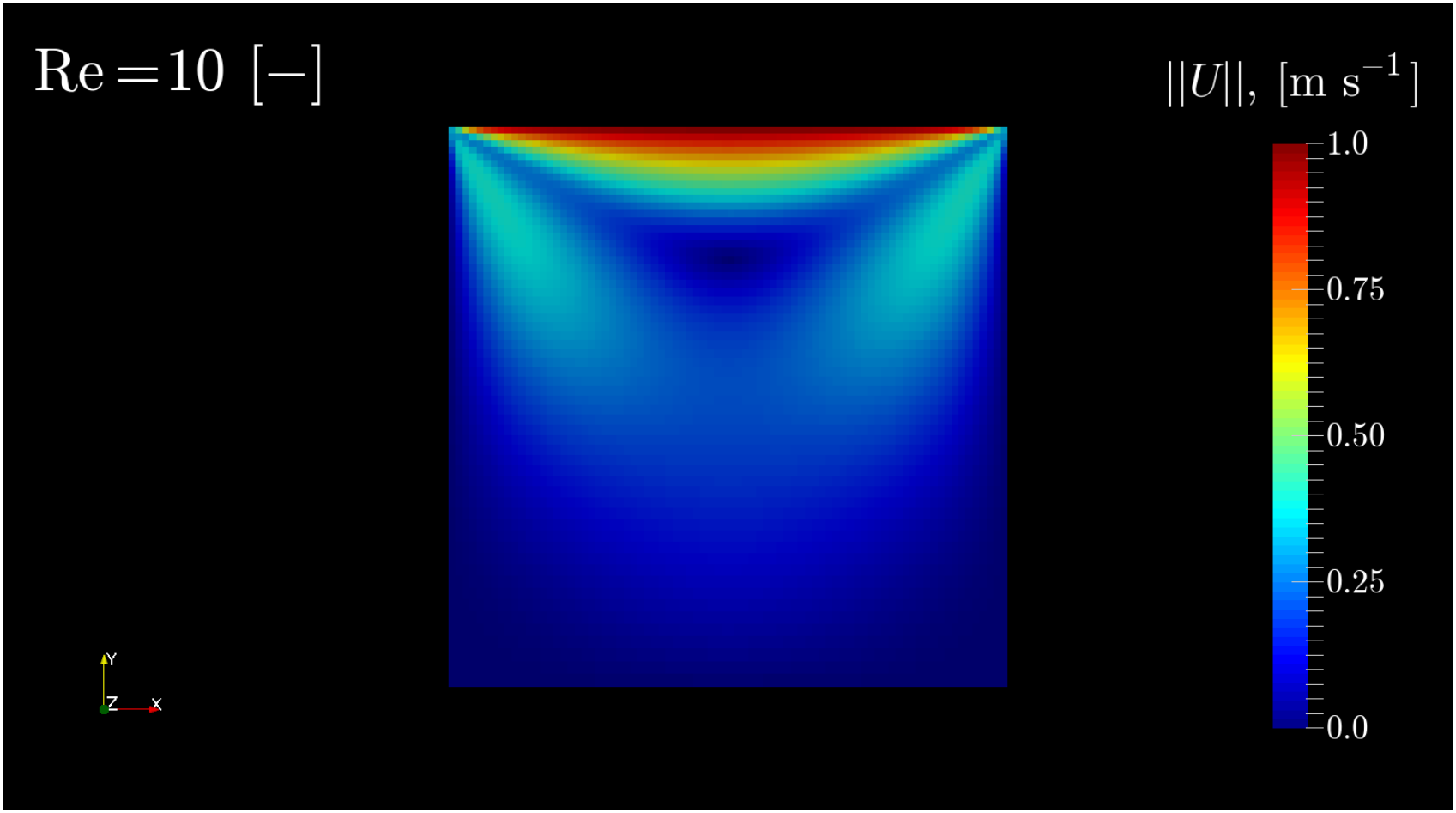
Test case: Standard OpenFOAM cavity case, variable $\mathrm{Re} = LU/\nu$



Steady state simulations

ROM based initial guess prediction for full NS solver (simpleFoam)

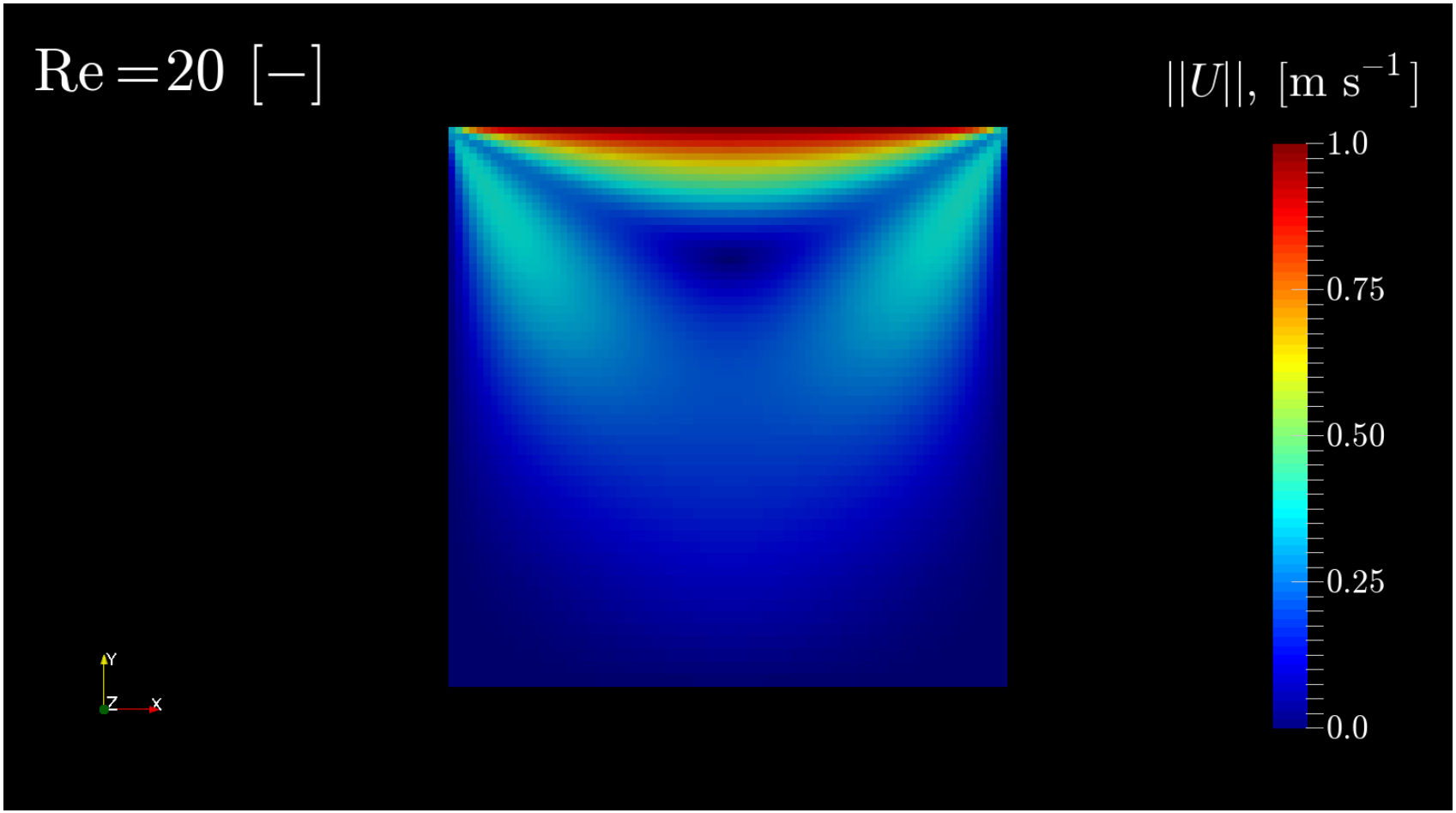
Test case: Standard OpenFOAM cavity case, variable $\mathrm{Re} = LU/\nu$



Steady state simulations

ROM based initial guess prediction for full NS solver (simpleFoam)

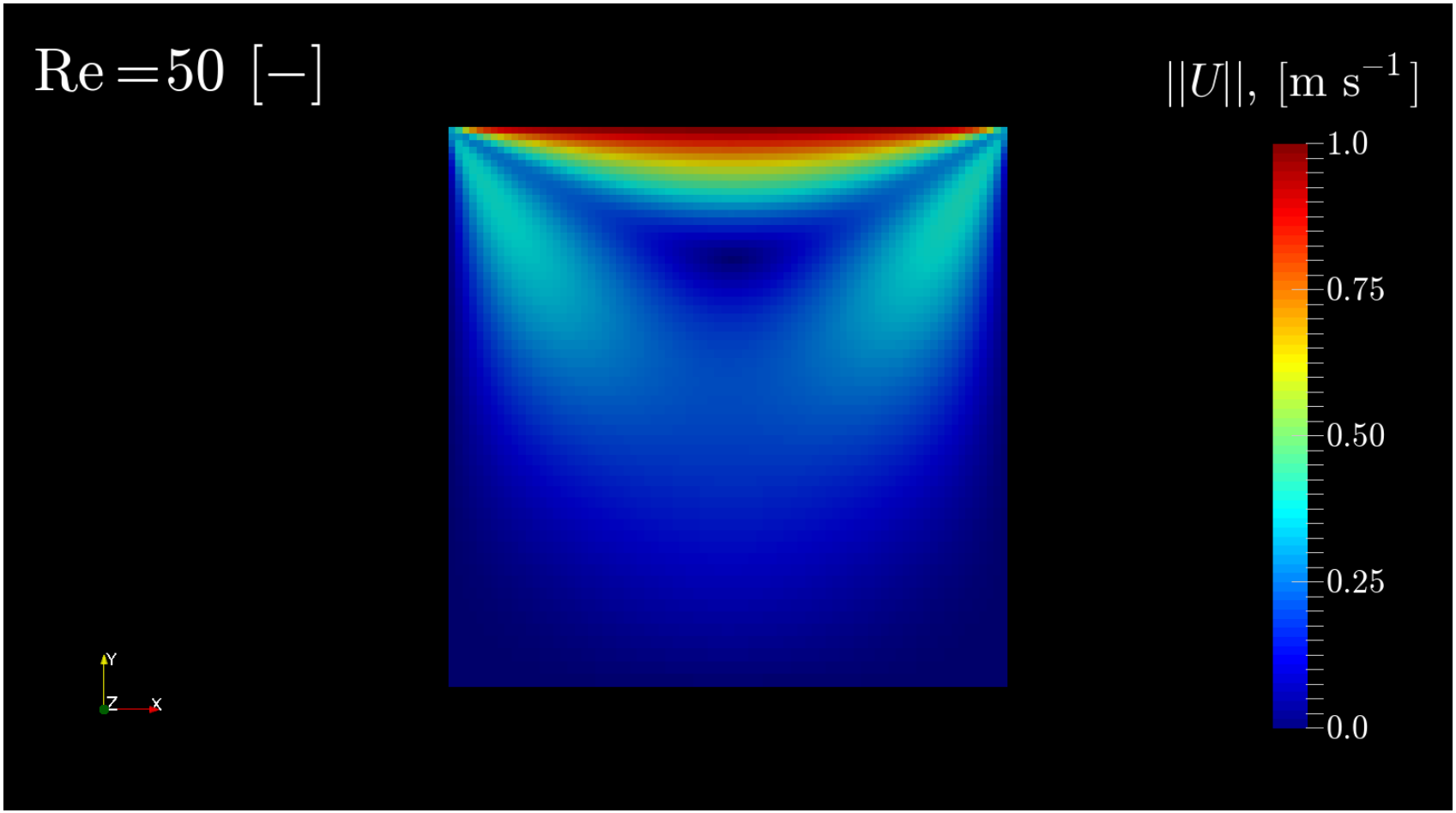
Test case: Standard OpenFOAM cavity case, variable $\mathrm{Re} = LU/\nu$



Steady state simulations

ROM based initial guess prediction for full NS solver (simpleFoam)

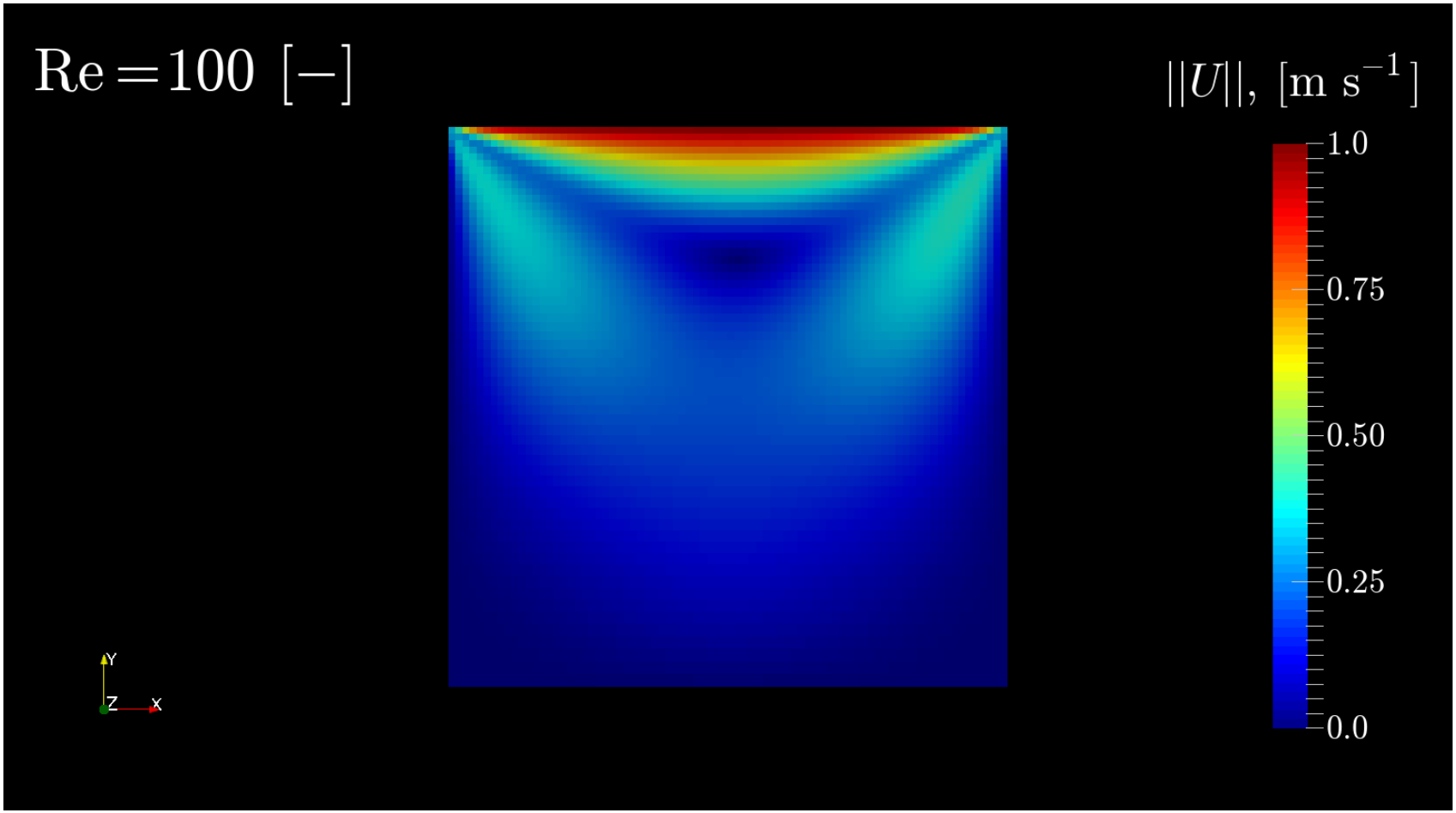
Test case: Standard OpenFOAM cavity case, variable $\mathrm{Re} = LU/\nu$



Steady state simulations

ROM based initial guess prediction for full NS solver (simpleFoam)

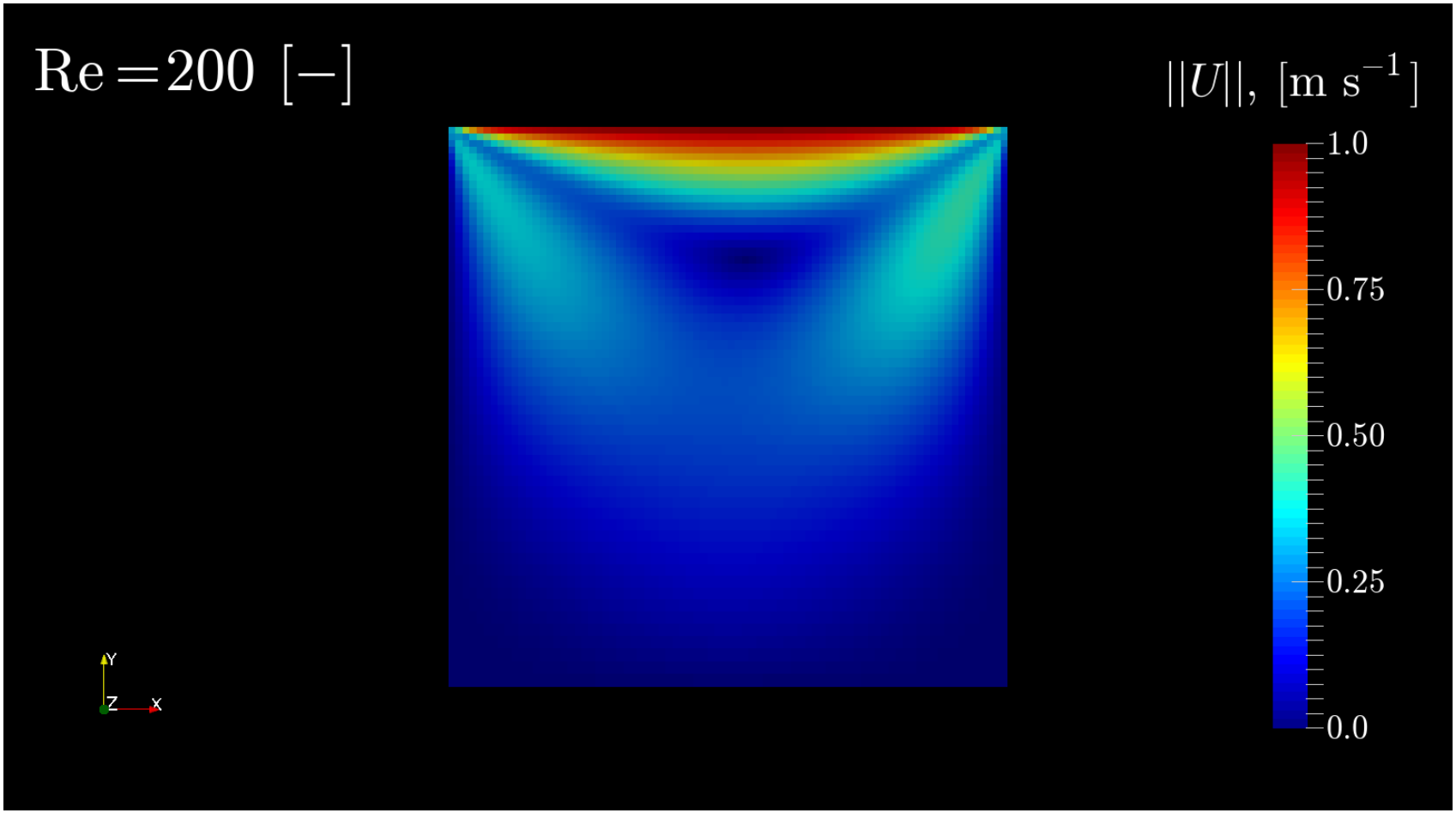
Test case: Standard OpenFOAM cavity case, variable $\mathrm{Re} = LU/\nu$



Steady state simulations

ROM based initial guess prediction for full NS solver (simpleFoam)

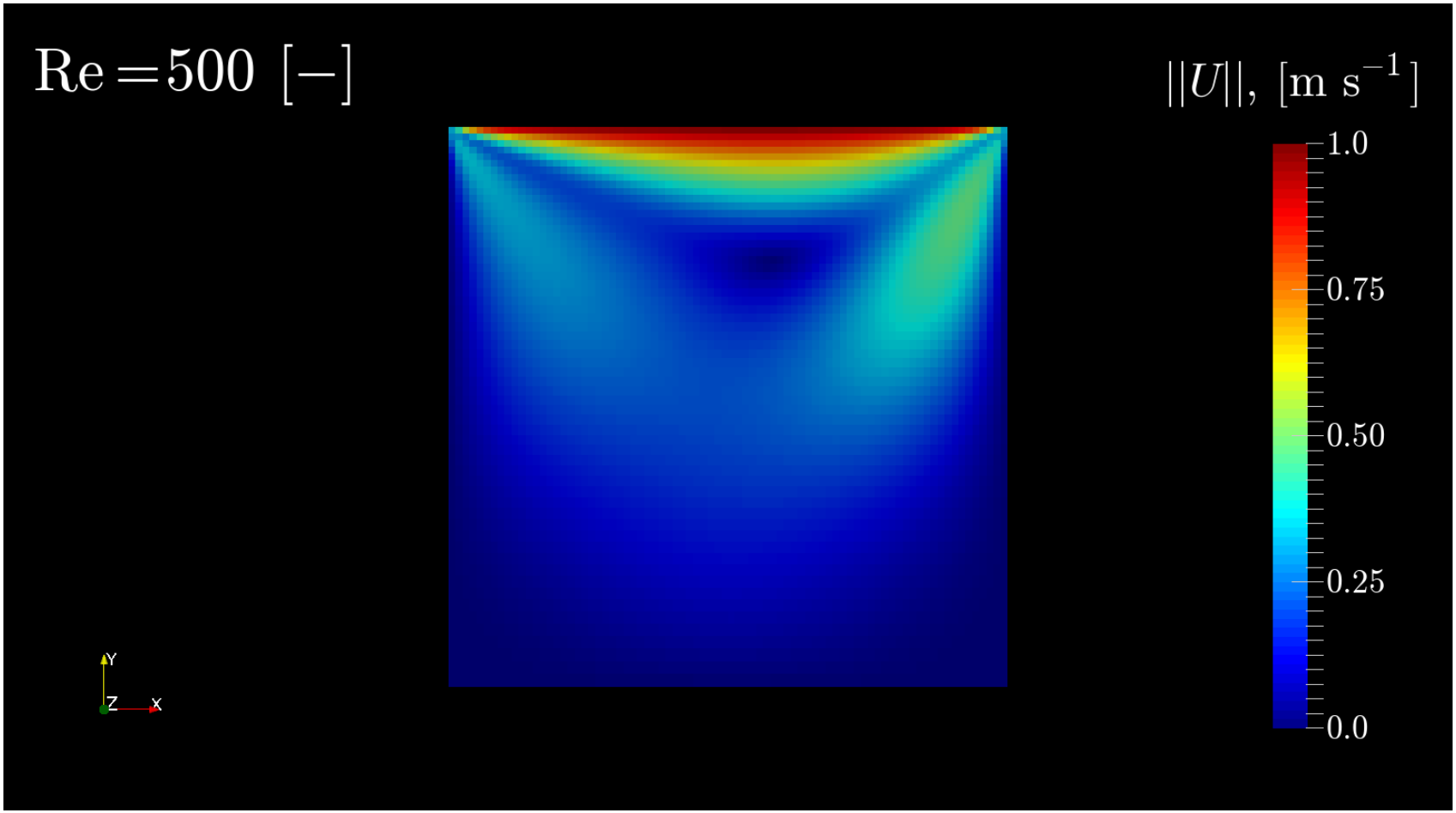
Test case: Standard OpenFOAM cavity case, variable $\mathrm{Re} = LU/\nu$



Steady state simulations

ROM based initial guess prediction for full NS solver (simpleFoam)

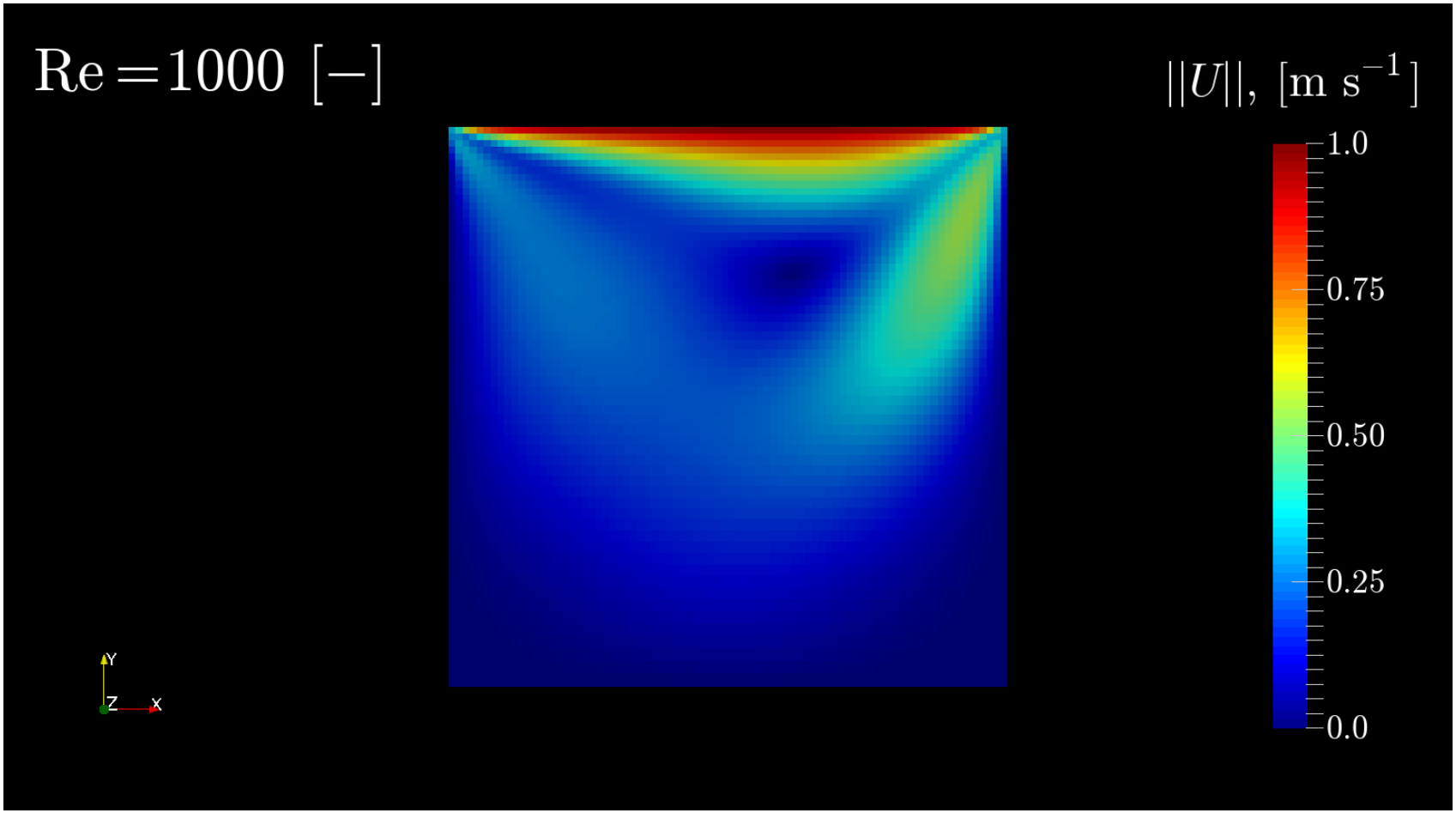
Test case: Standard OpenFOAM cavity case, variable $\mathrm{Re} = LU/\nu$



Steady state simulations

ROM based initial guess prediction for full NS solver (simpleFoam)

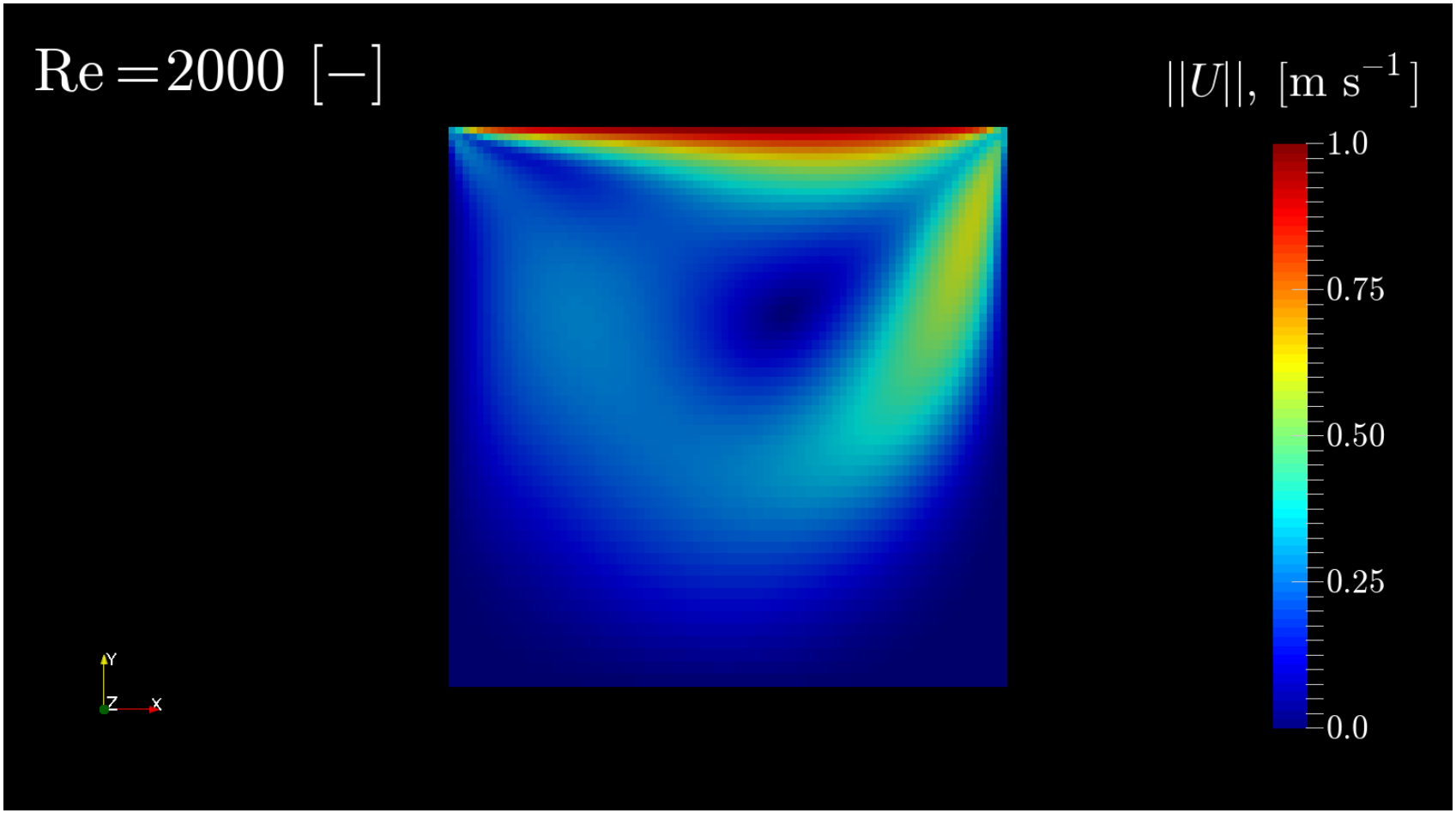
Test case: Standard OpenFOAM cavity case, variable $\mathrm{Re} = LU/\nu$



Steady state simulations

ROM based initial guess prediction for full NS solver (simpleFoam)

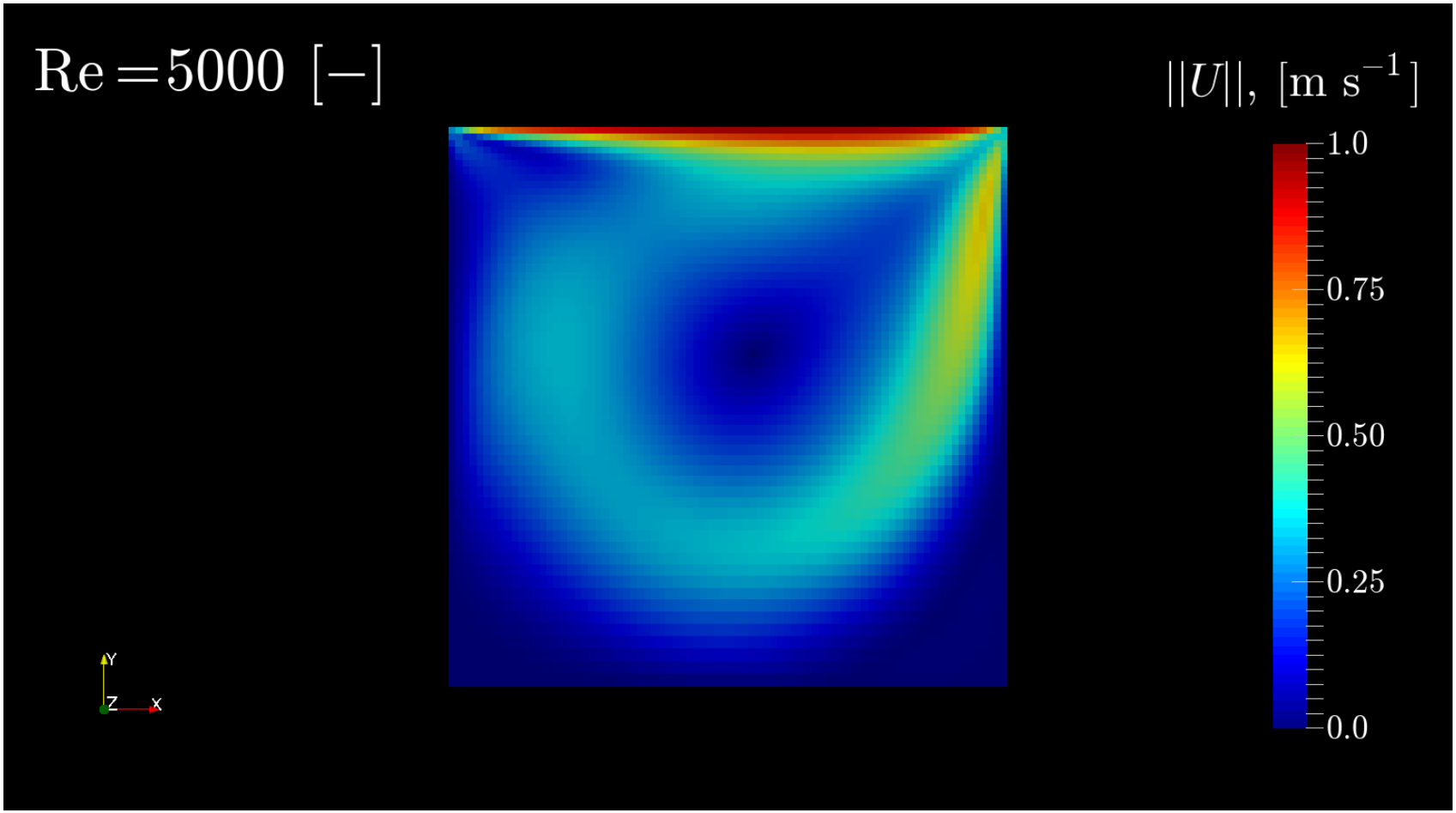
Test case: Standard OpenFOAM cavity case, variable $\mathrm{Re} = LU/\nu$



Steady state simulations

ROM based initial guess prediction for full NS solver (simpleFoam)

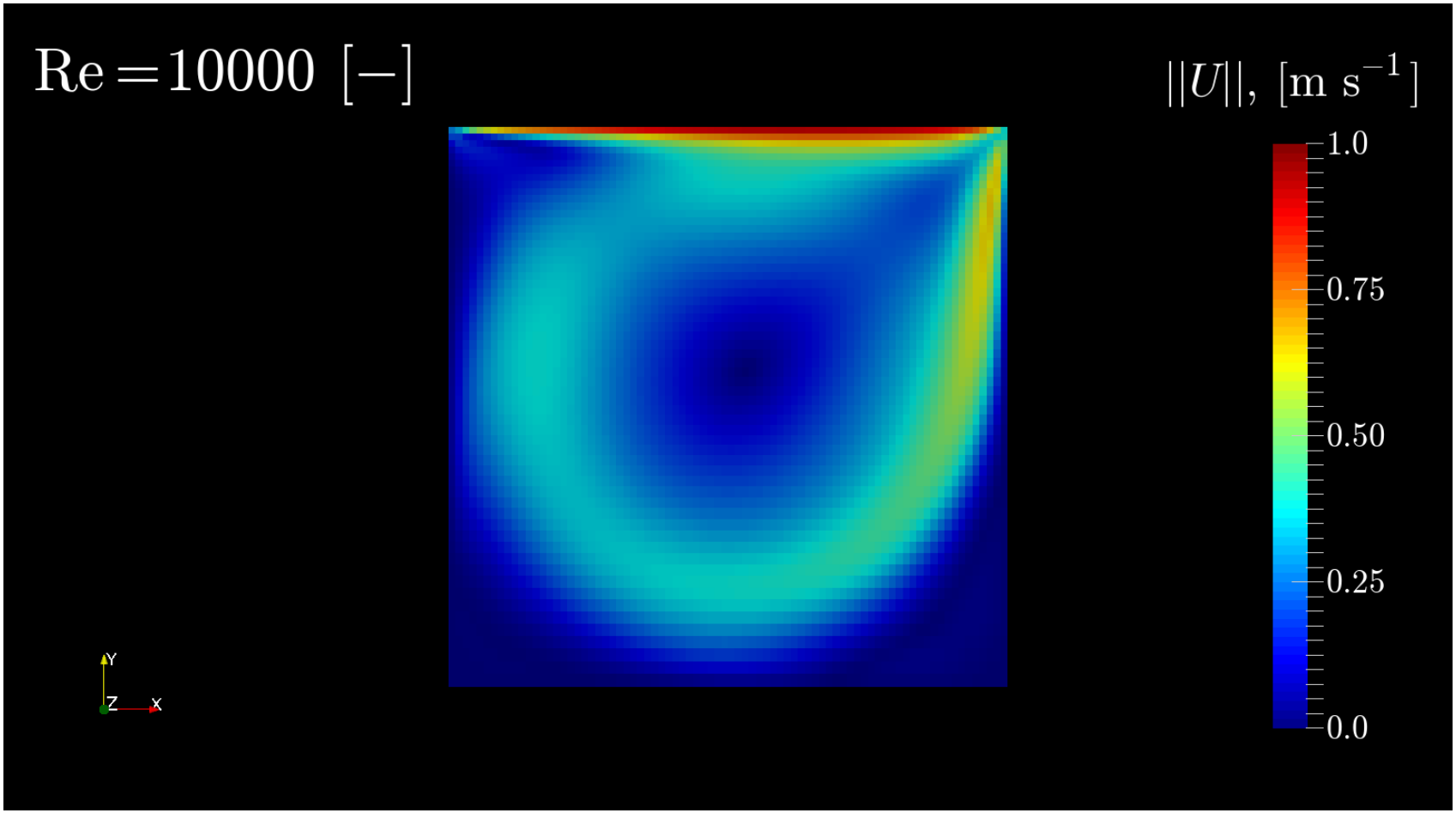
Test case: Standard OpenFOAM cavity case, variable $\mathrm{Re} = LU/\nu$



Steady state simulations

ROM based initial guess prediction for full NS solver (simpleFoam)

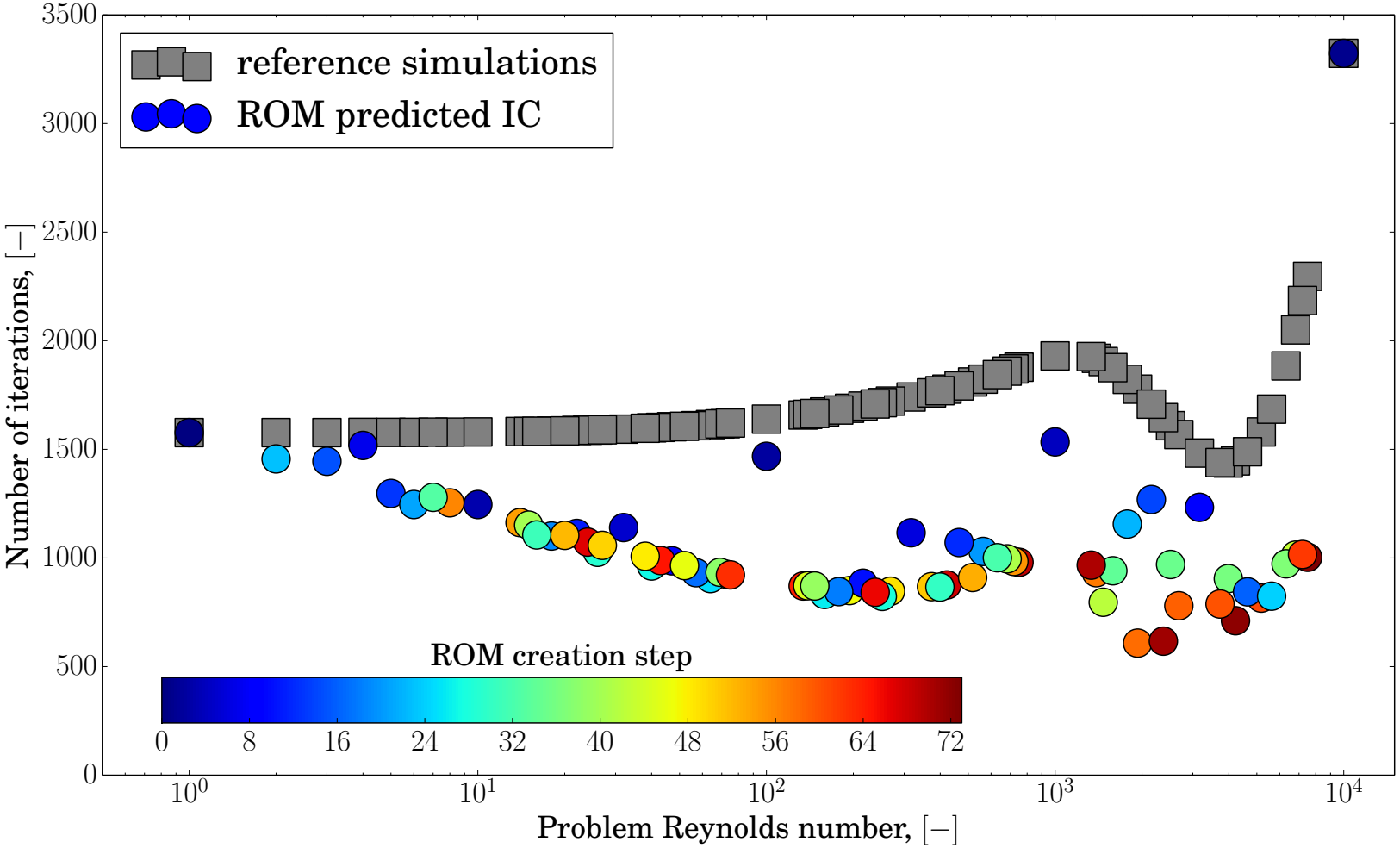
Test case: Standard OpenFOAM cavity case, variable $\mathrm{Re} = LU/\nu$



Steady state simulations

ROM based initial guess prediction for full NS solver (simpleFoam)

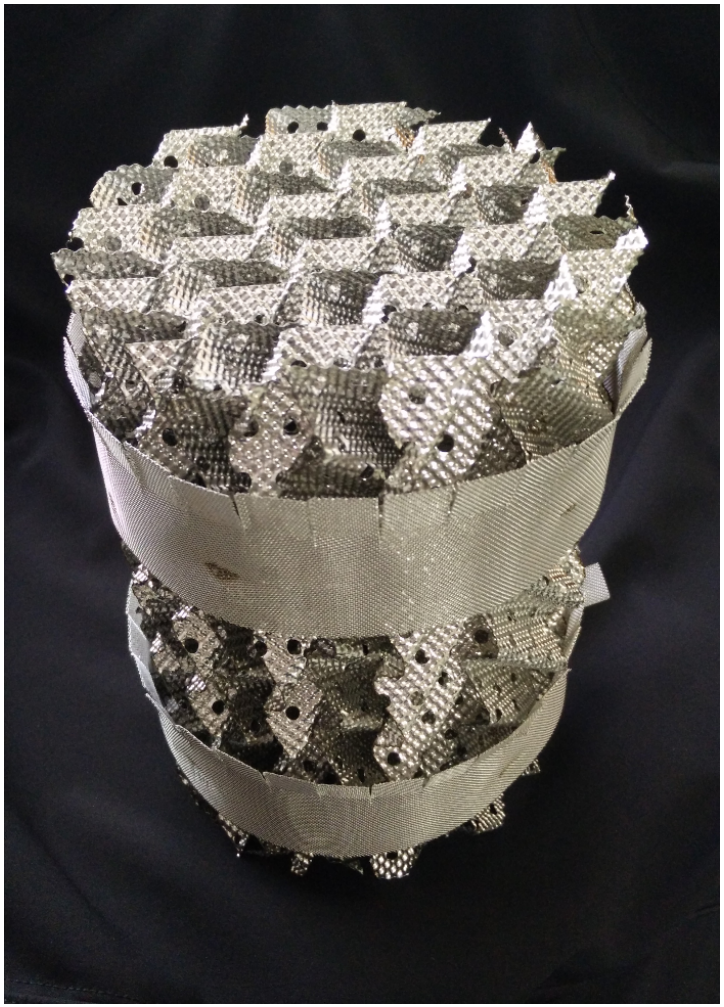
Test case: Iteration save-up for simpleFoam (serial run)

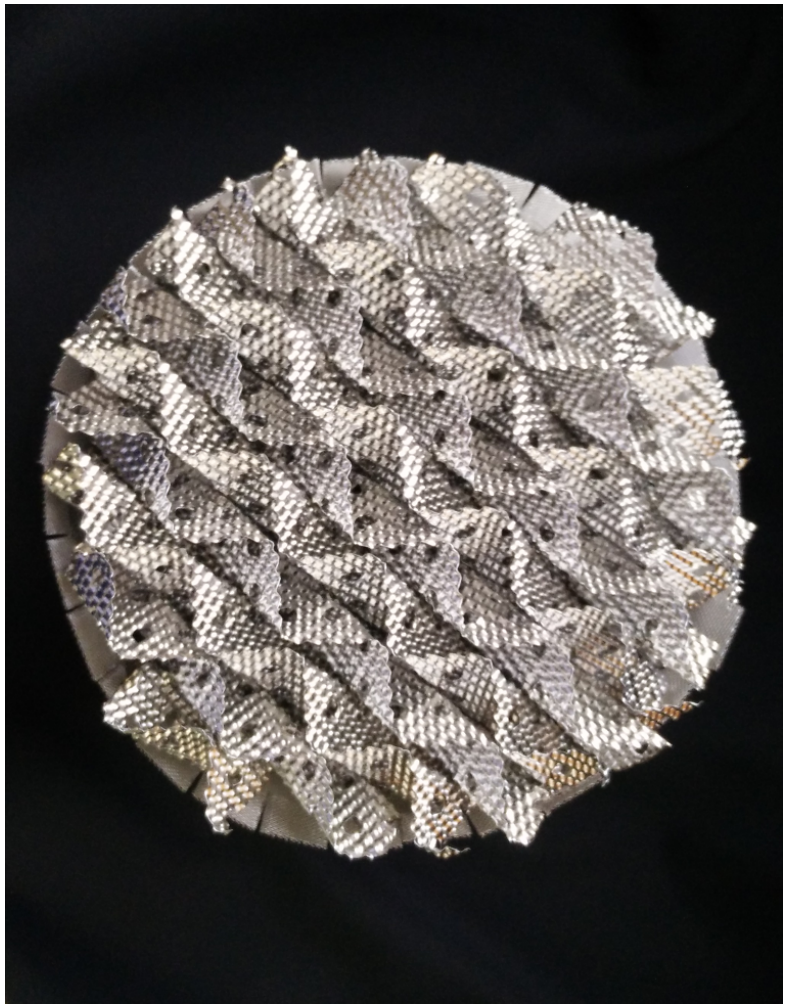


Semi-industrial scale CFD

Steady-state RANS simulation of flow in complex geometry

Remainder: Geometry of structured packing

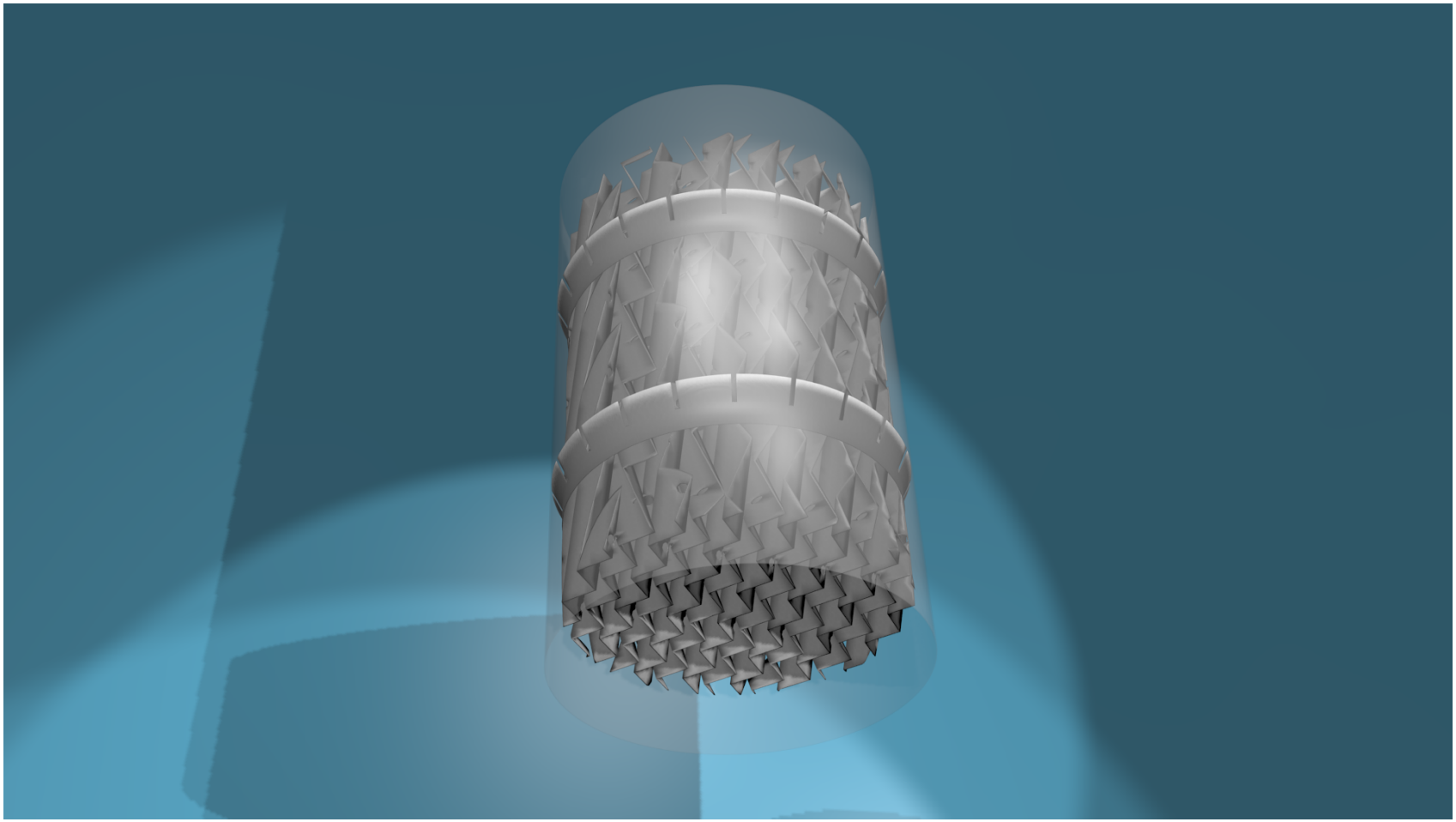




Semi-industrial scale CFD

Steady-state RANS simulation of flow in complex geometry

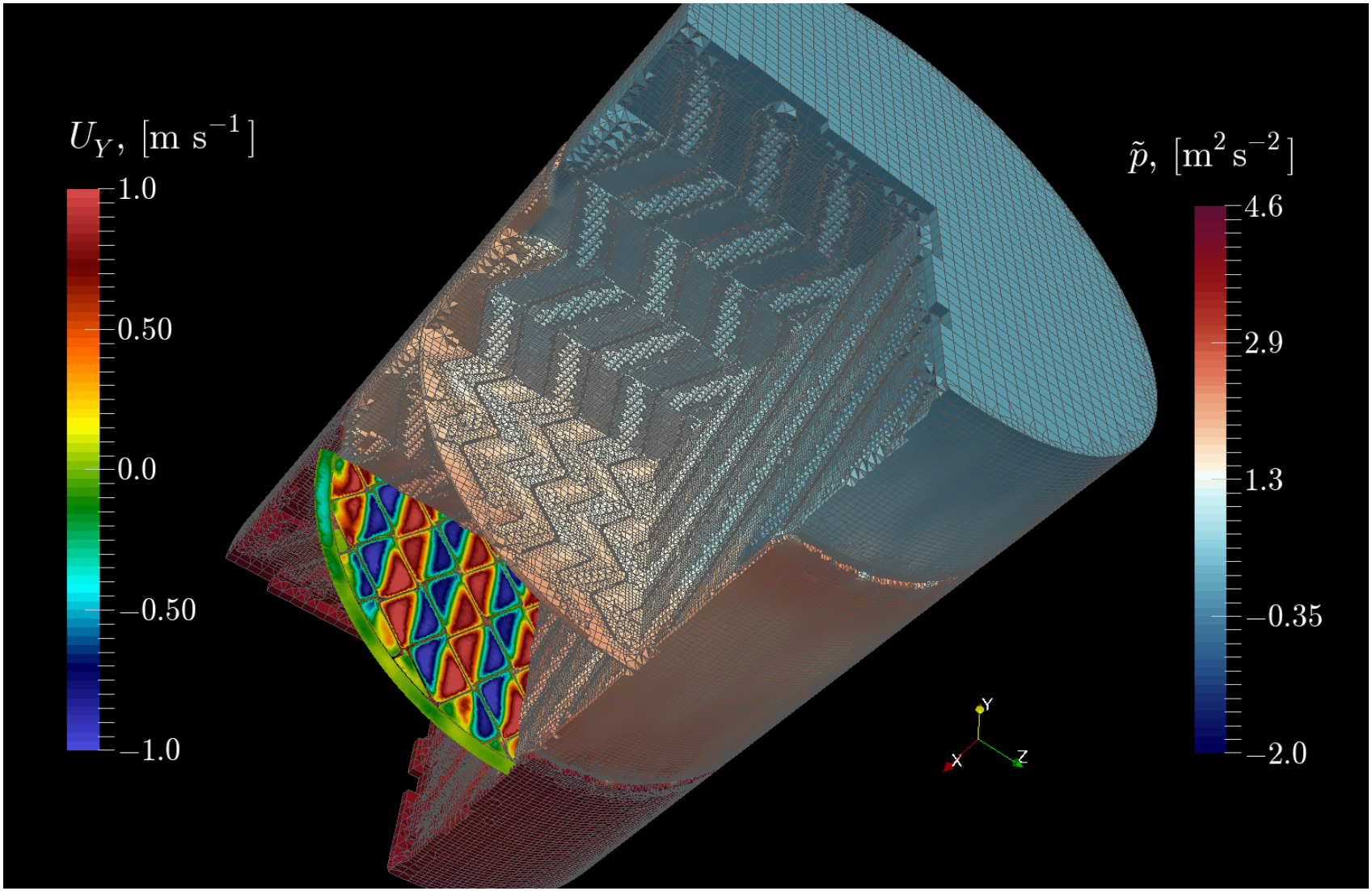
Geometry: Mellapak 250.X structured packing



Semi-industrial scale CFD

Steady-state RANS simulation of flow in complex geometry

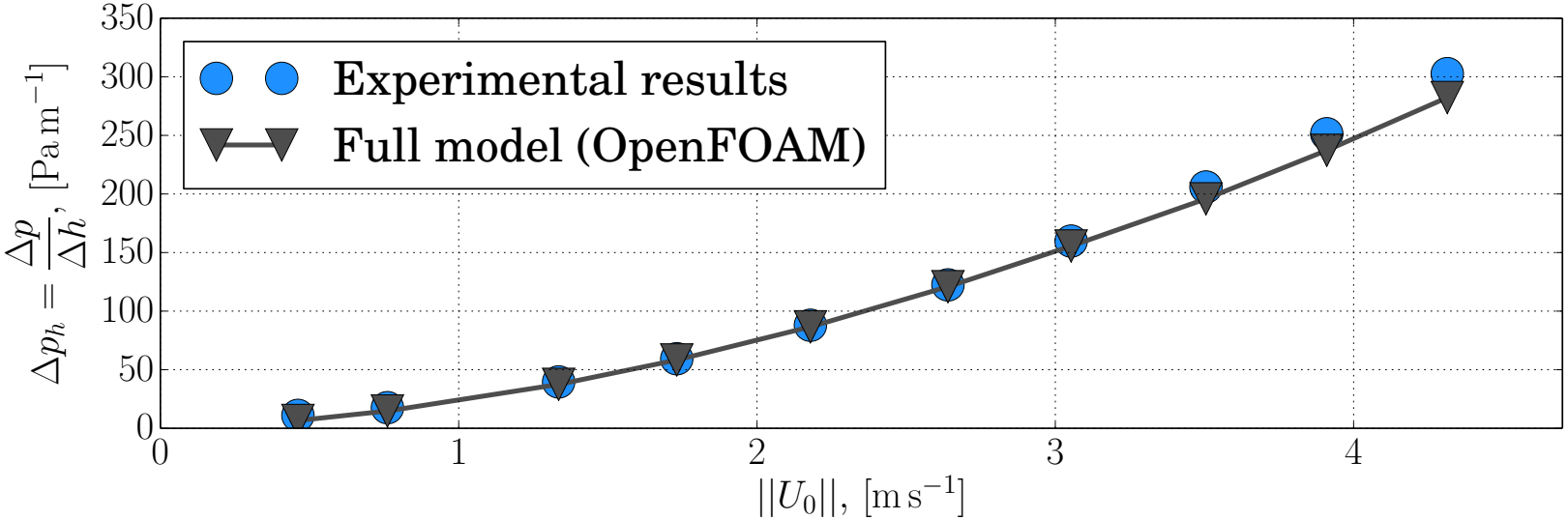
Gas flow simulation: Incompressible steady state RANS simulation

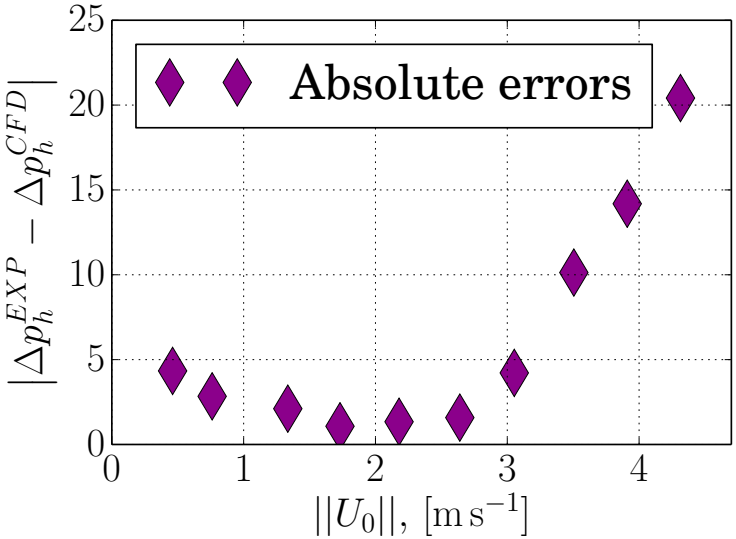


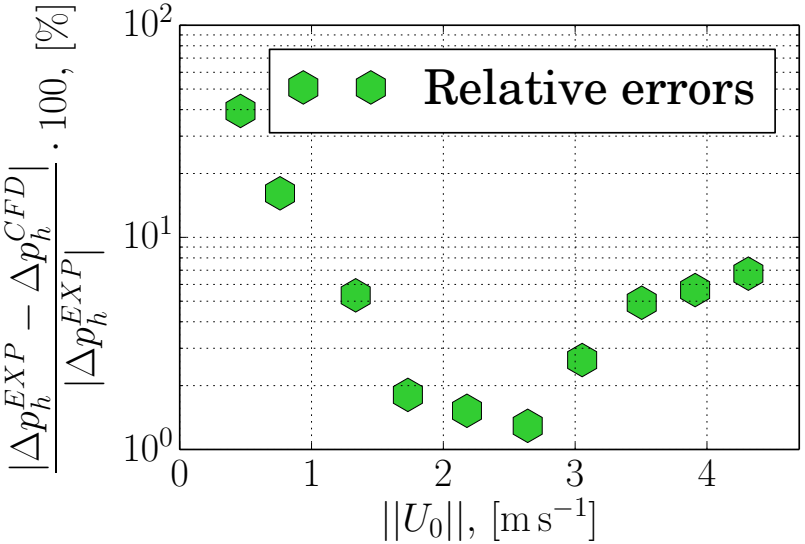
Semi-industrial scale CFD

Steady-state RANS simulation of flow in complex geometry

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



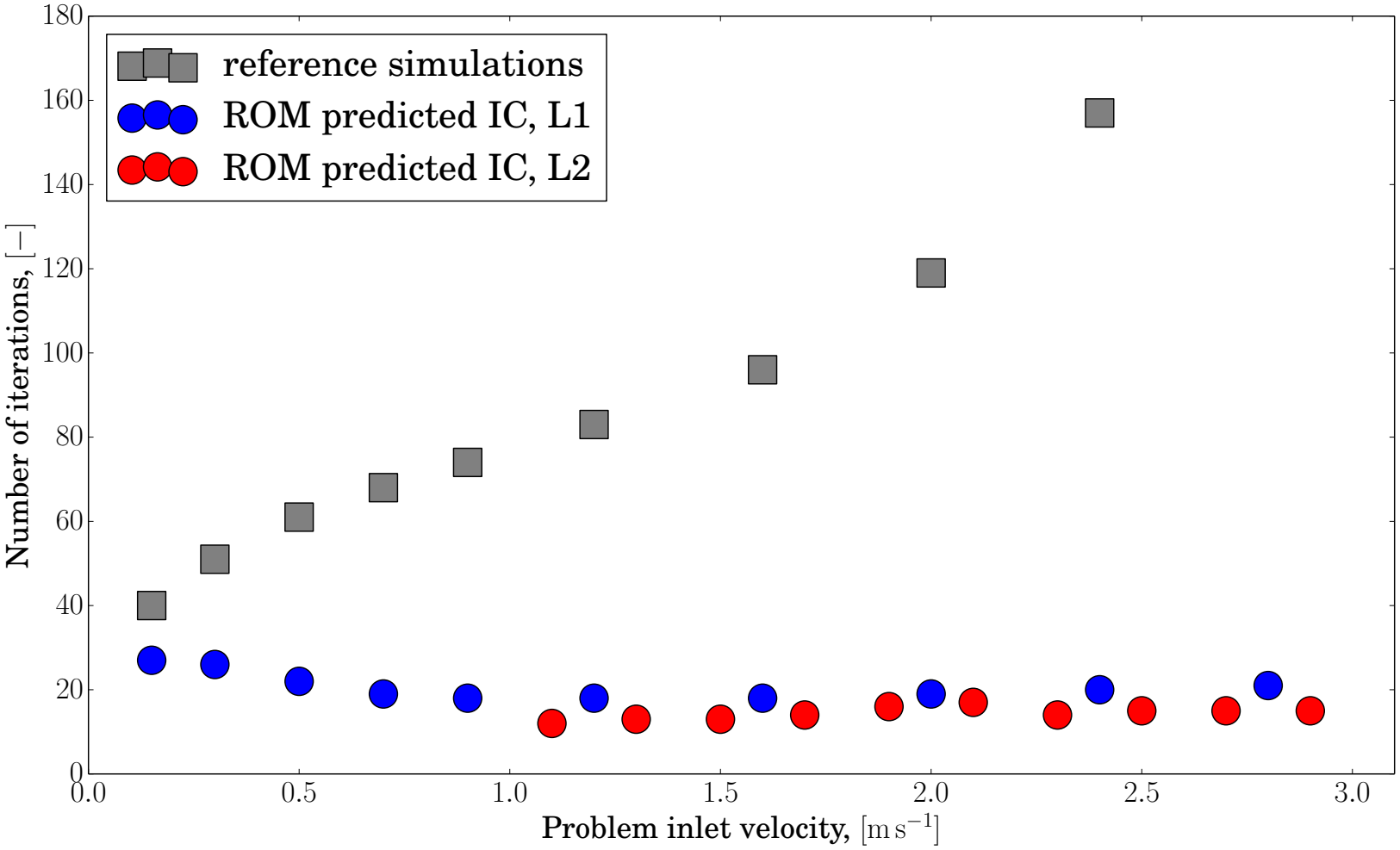




Semi-industrial scale application

ROM based initial guess prediction for full NS solver (simpleFoam)

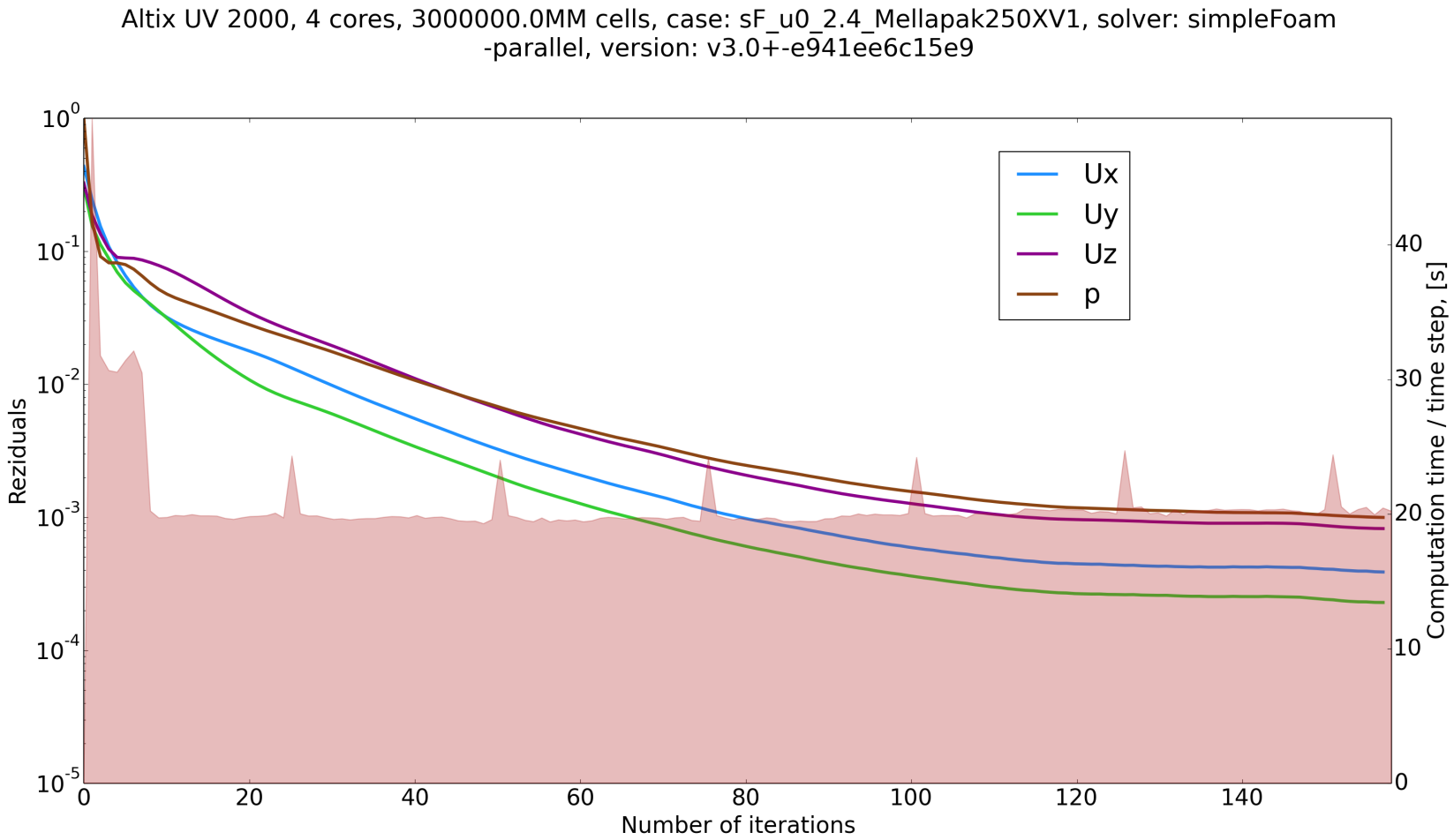
Full case: Flow through the Mellapak 250.X packing



Semi-industrial scale application

ROM based initial guess prediction for full NS solver (simpleFoam)

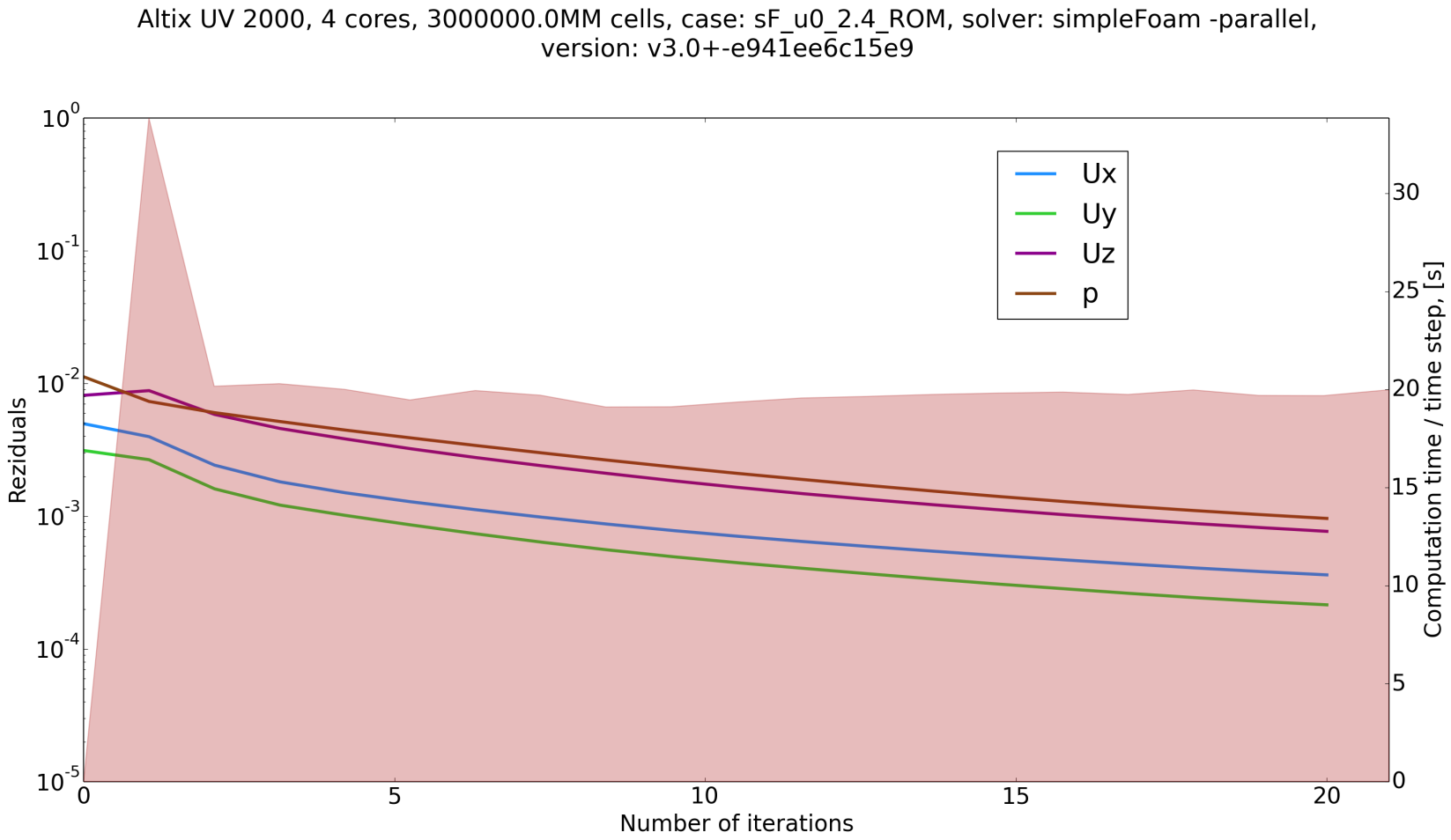
Full case: Residuals evolution, from potentialFoam initialized fields



Semi-industrial scale application

ROM based initial guess prediction for full NS solver (simpleFoam)

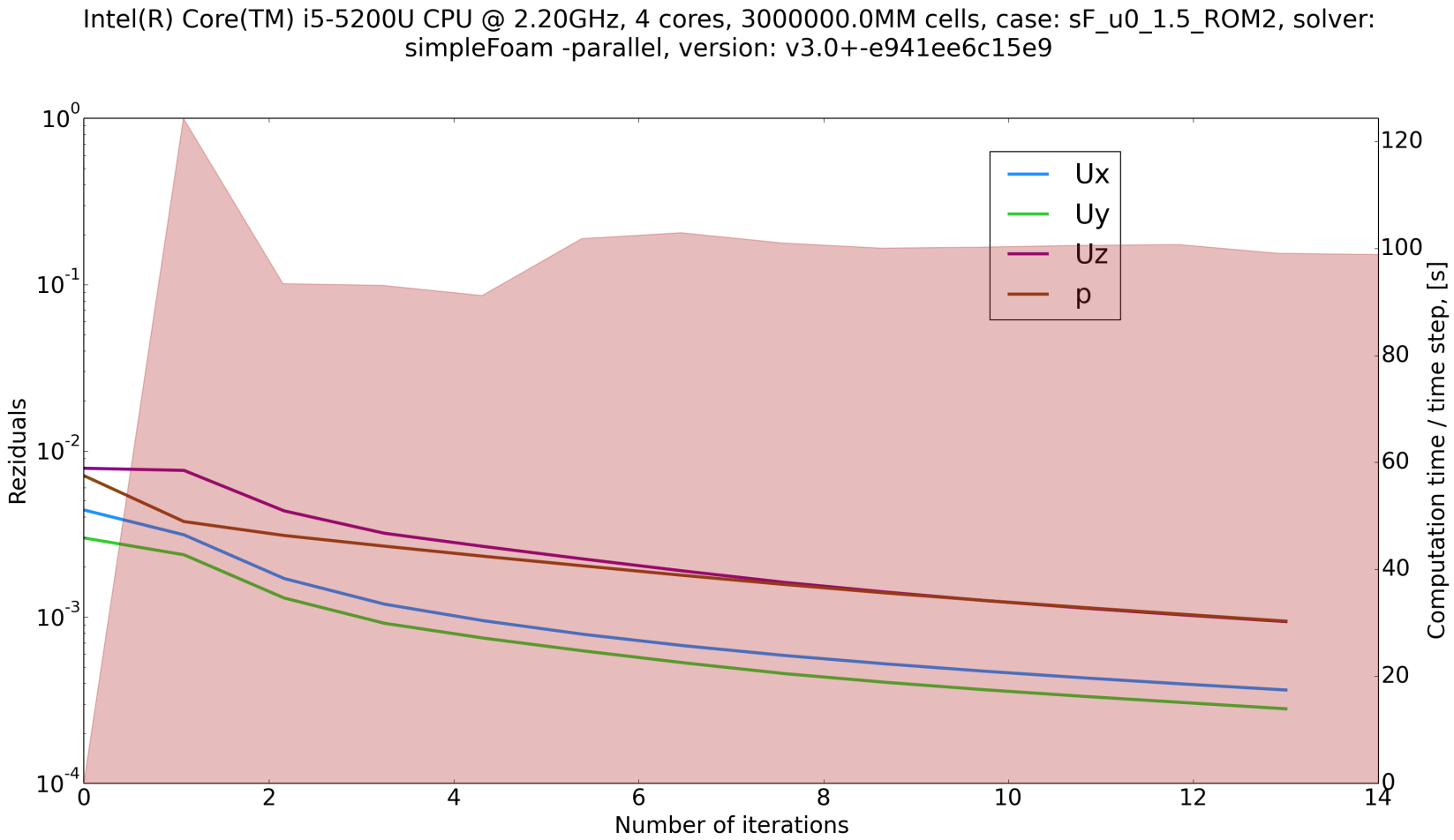
Full case: Residuals evolution, from ROM predicted fields, L1



Semi-industrial scale application

ROM based initial guess prediction for full NS solver (simpleFoam)

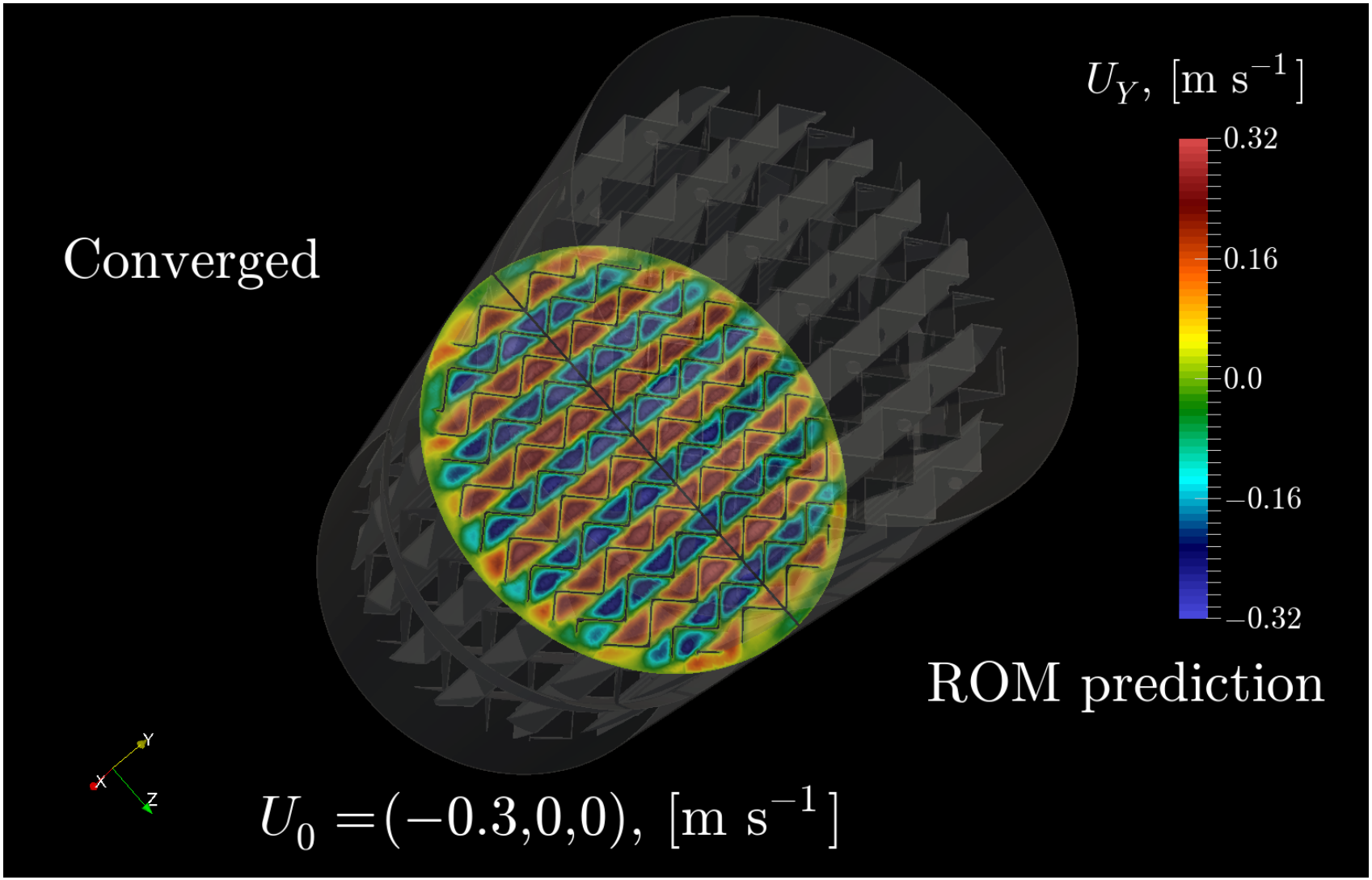
Full case: Residuals evolution, from ROM predicted fields, L2



Semi-industrial scale application

ROM based initial guess prediction for full NS solver (simpleFoam)

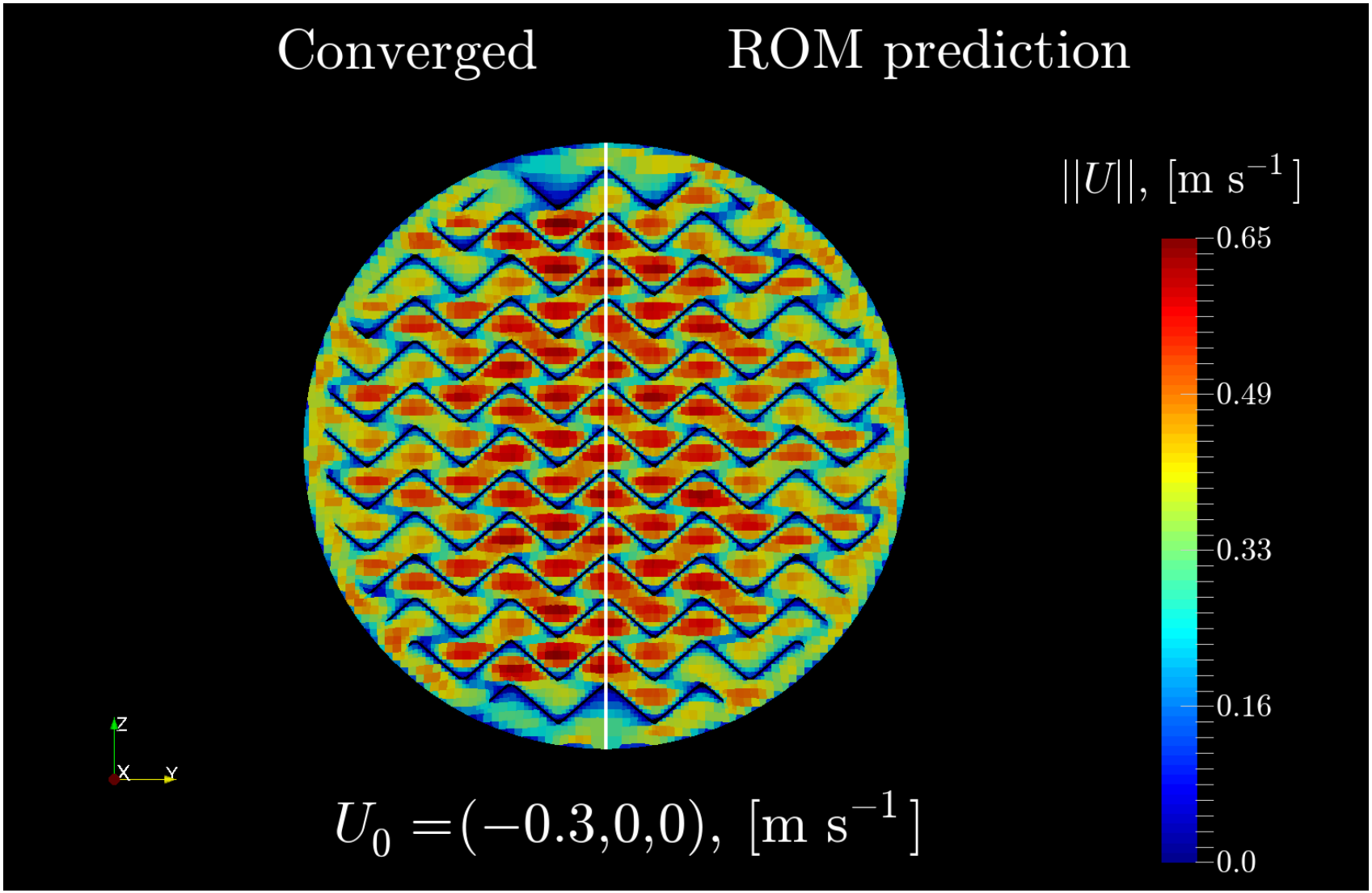
Full case: Predicted vs. converged solution in L1



Semi-industrial scale application

ROM based initial guess prediction for full NS solver (simpleFoam)

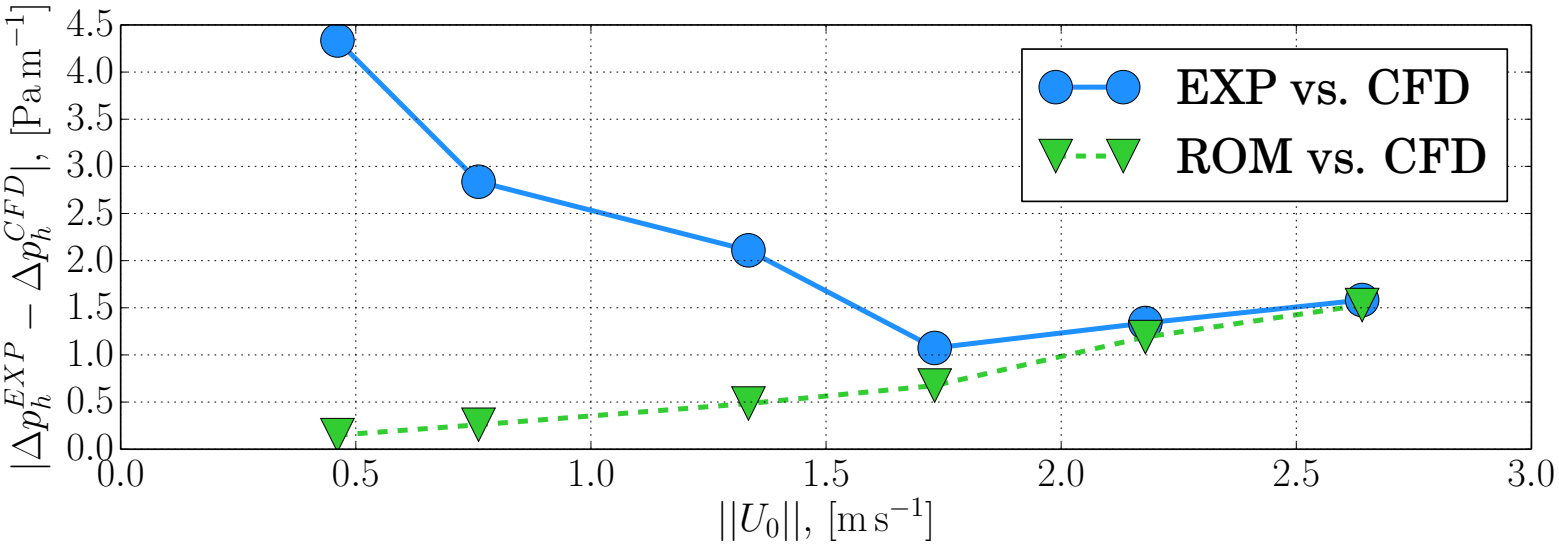
Full case: Predicted vs. converged solution in L1

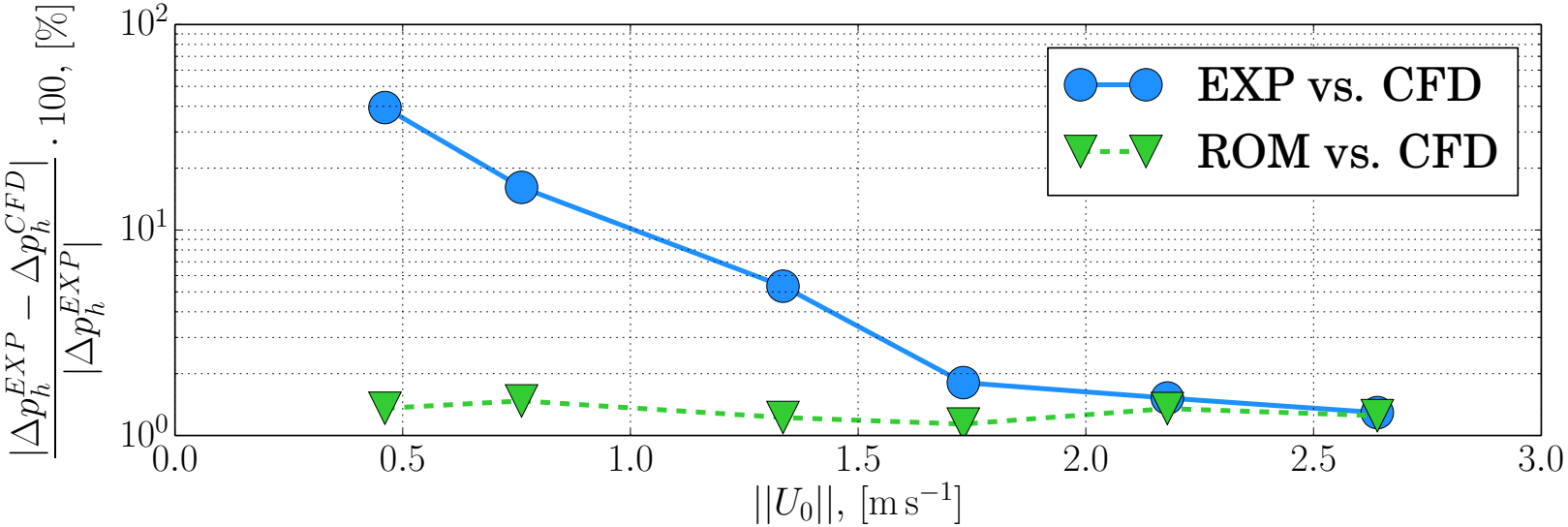


Semi-industrial scale application

ROM based initial guess prediction for full NS solver (simpleFoam)

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





Optimal operation conditions

Offline (ROM-based) operation conditions optimization, gas flow, Mellapak 250.X

Cost function: Single phase, toy problem

$$F(u_0) = \frac{\Delta\tilde{p} - \Delta\tilde{p}_{Max}}{\Delta\tilde{p}_{Max}} + K\frac{Q^2 - 2Q_{Max}Q + Q_{Min}(2Q_{Max} - Q_{Min})}{(Q_{Max} - Q_{Min})^2},$$

$$\Delta\tilde{p} = \Delta\tilde{p}(u_0), \quad Q = Q(u_0), \quad U_0 = (-u_0, 0, 0),$$

$\Delta\tilde{p}_{Max}$ ..... maximal allowable pressure loss

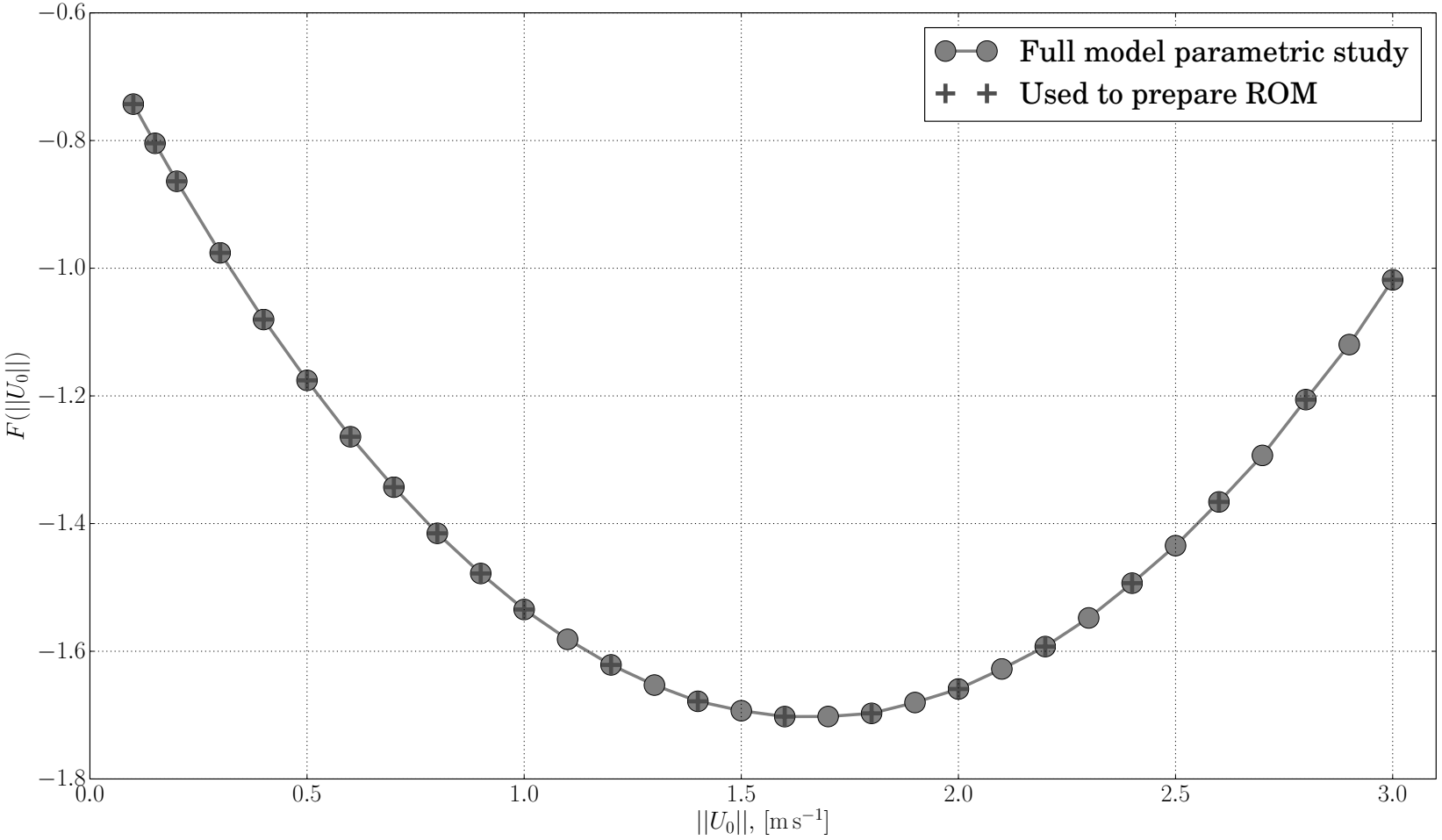
Q_{Max}, (Q_{Min}) ..... maximal, (minimal) allowable gas flow rate

K ..... relative importance of the two terms

Optimal operation conditions

Offline (ROM-based) operation conditions optimization, gas flow, Mellapak 250.X

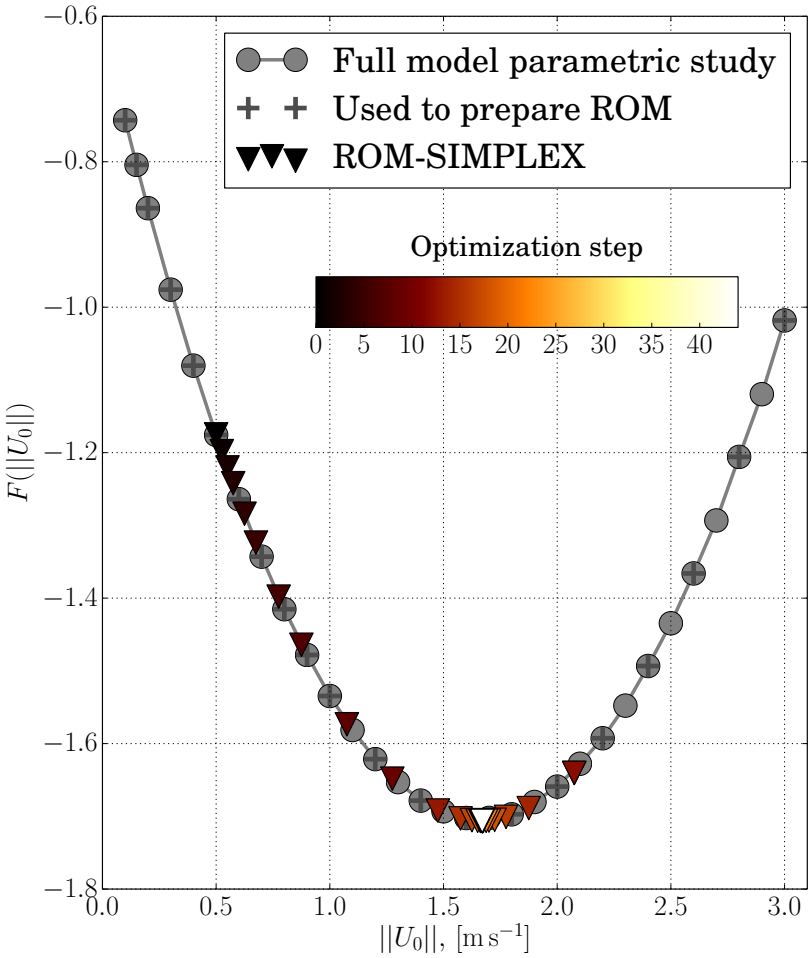
Available data: Cost function curve, $F(u_0)$, $u_0 \in \langle 0.1, 3.0 \rangle$

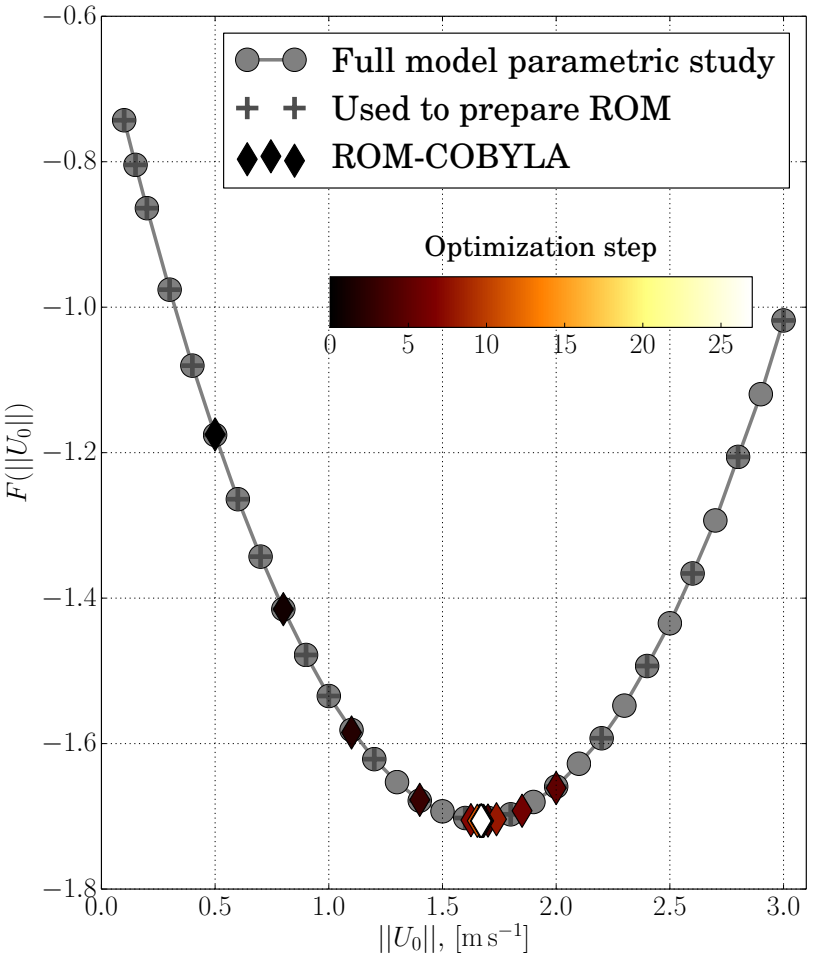


Optimal operation conditions

Offline (ROM-based) operation conditions optimization, gas flow, Mellapak 250.X

Cost function minimization: Results of SIMPLEX and COBYLA algorithms

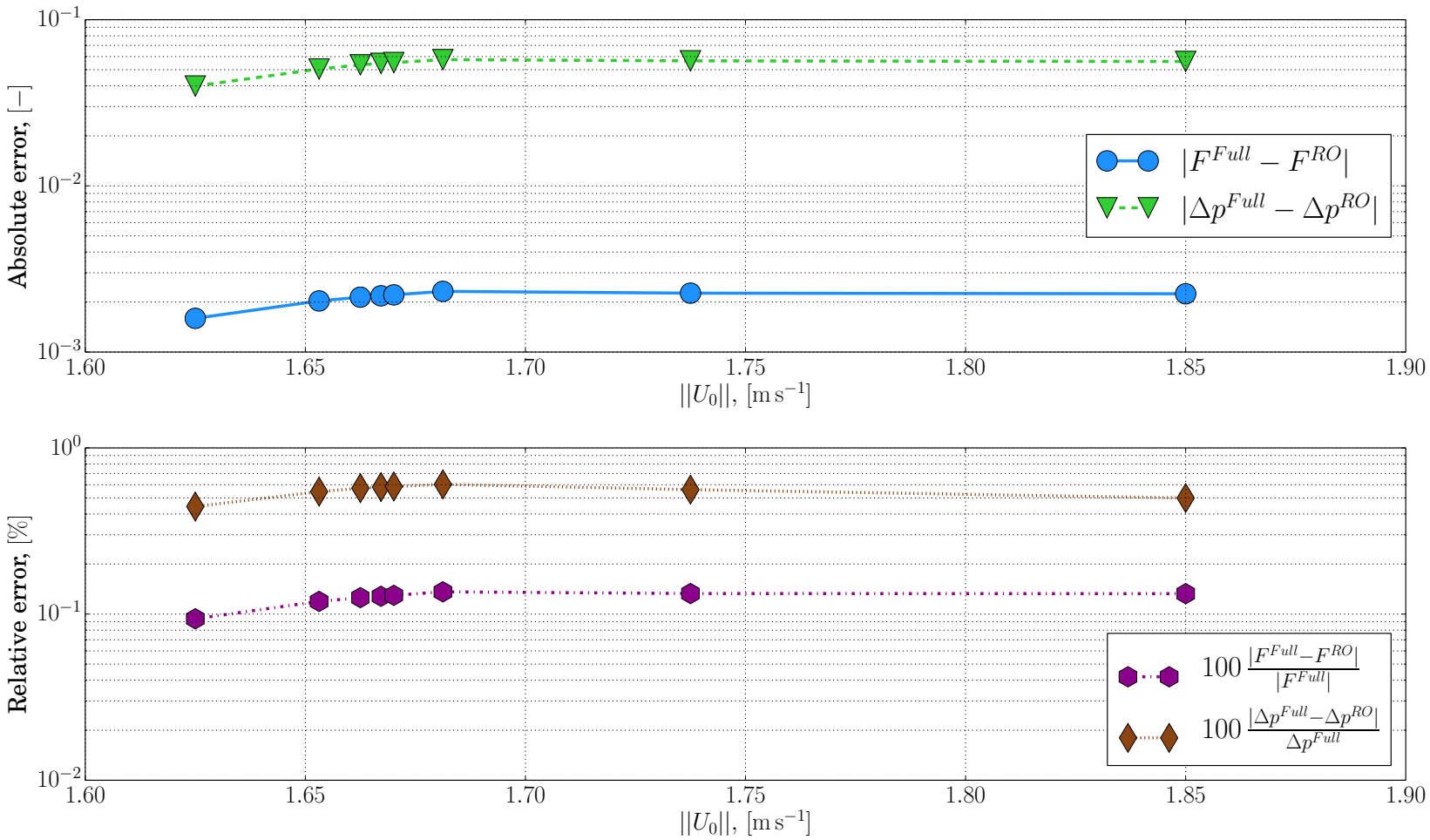




Optimal operation conditions

Offline (ROM-based) operation conditions optimization, gas flow, Mellapak 250.X

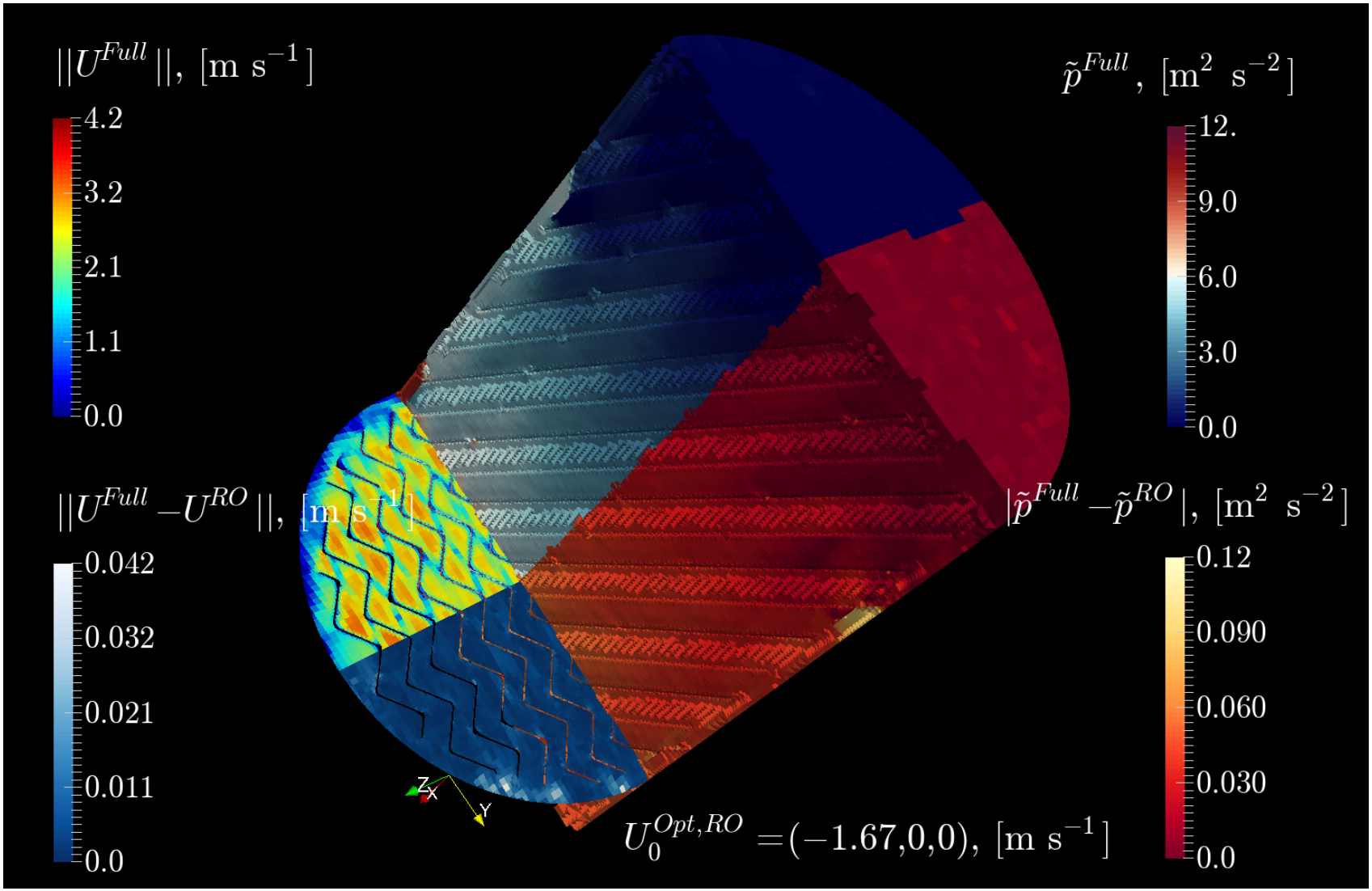
Solution quality: Comparison of ROM results with reference simulations (COBYLA)



Optimal operation conditions

Offline (ROM-based) operation conditions optimization, gas flow, Mellapak 250.X

Solution quality: Comparison of RO and Full models results



UNIVERSITY OF CHEMISTRY AND TECHNOLOGY PRAGUE

Conclusions

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

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)

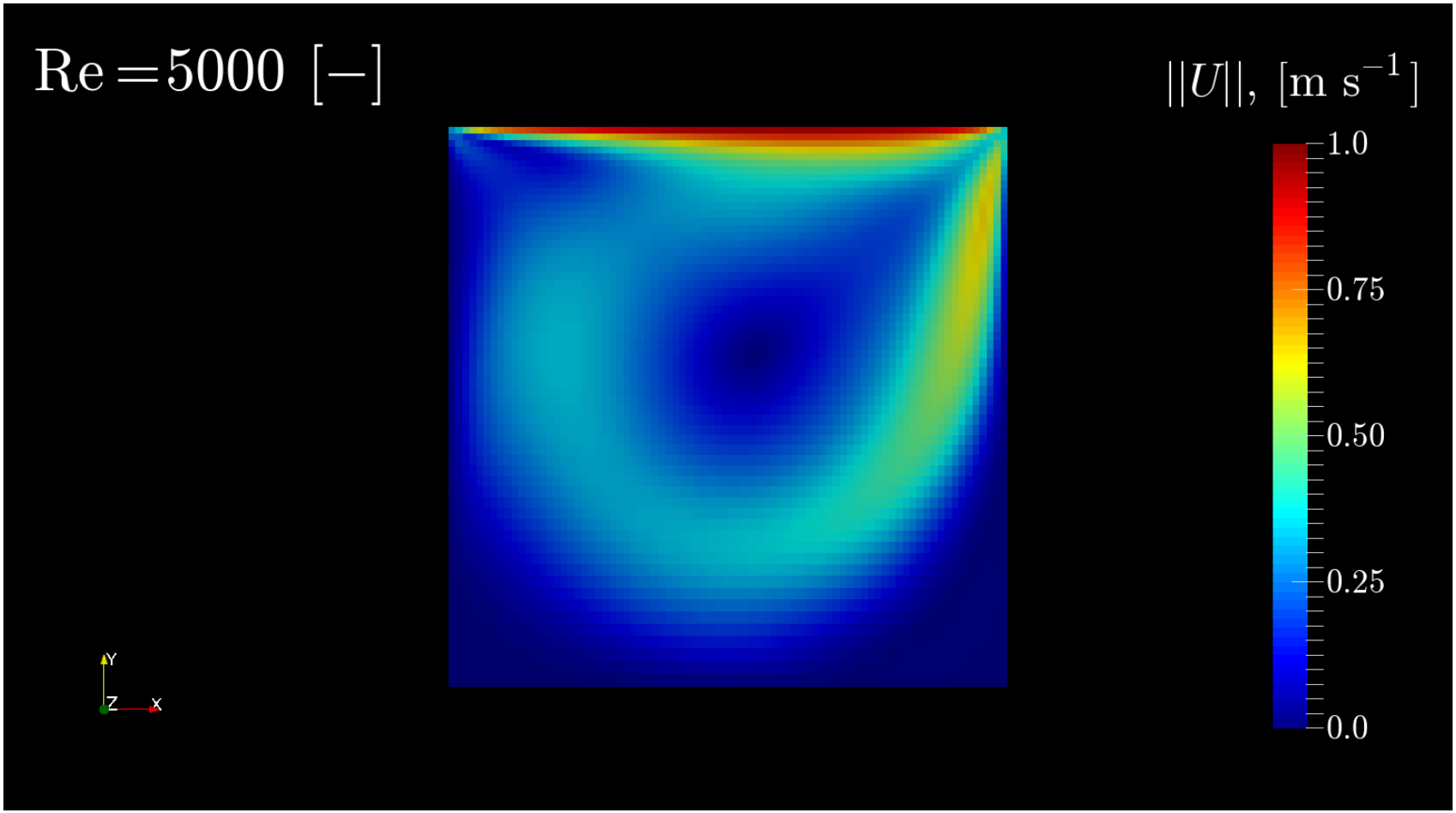
Disadvantages

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

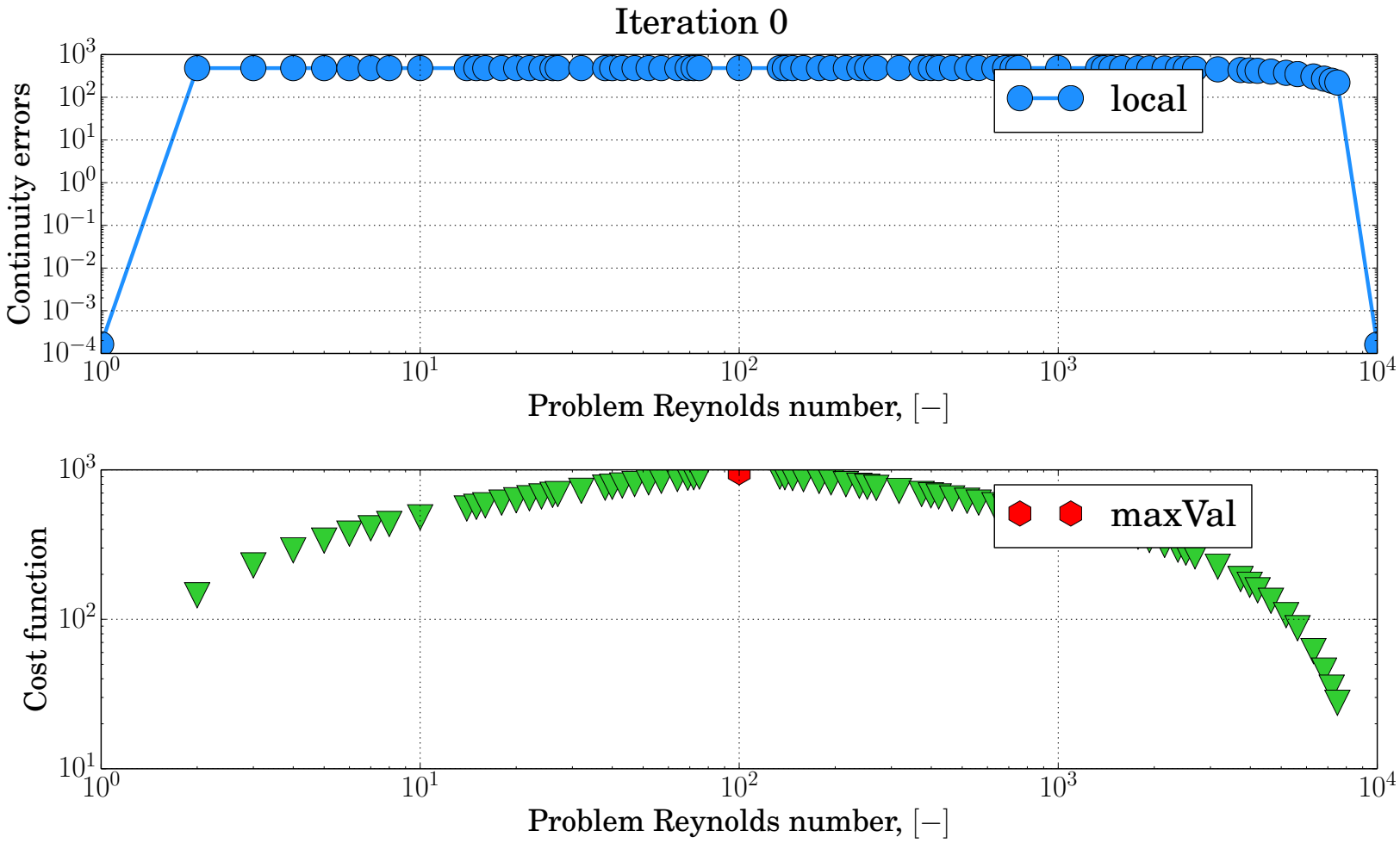
Next steps

Towards ROM size reduction and multiparametric systems

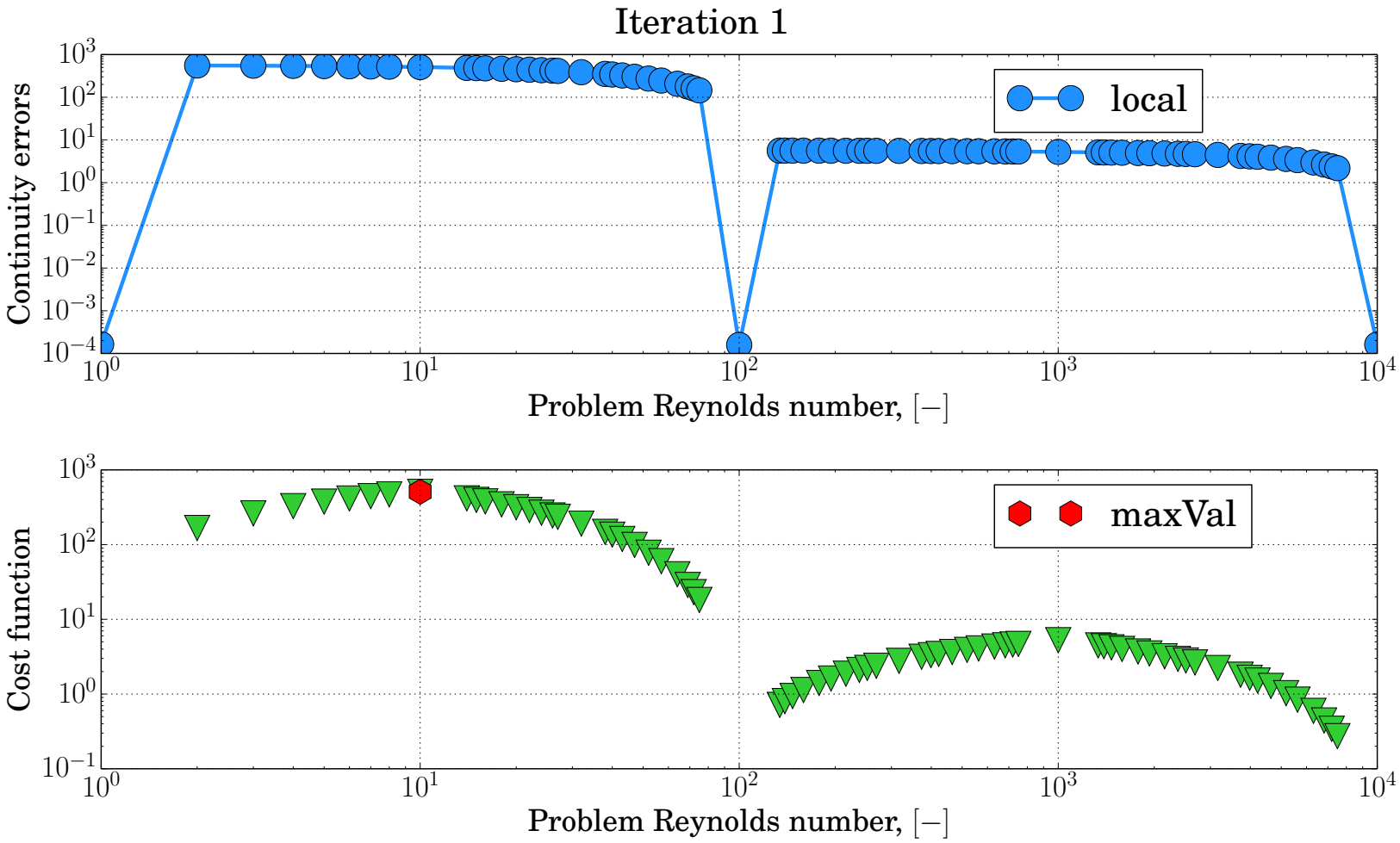
ROM size reduction: $\mathcal{S}^e$ selection based on greedy algorithm



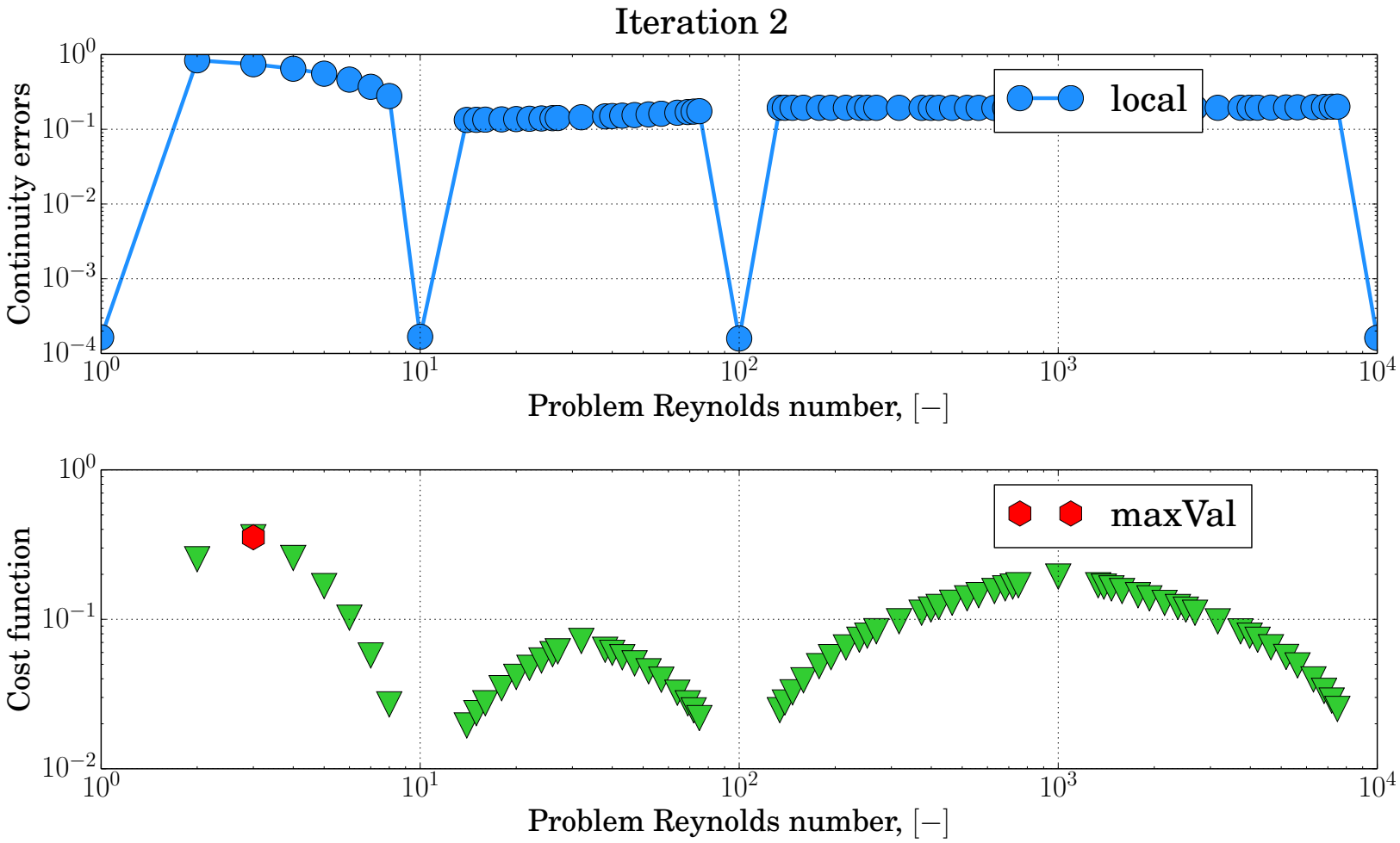
ROM size reduction: $\mathcal{S}^e$ selection based on greedy algorithm



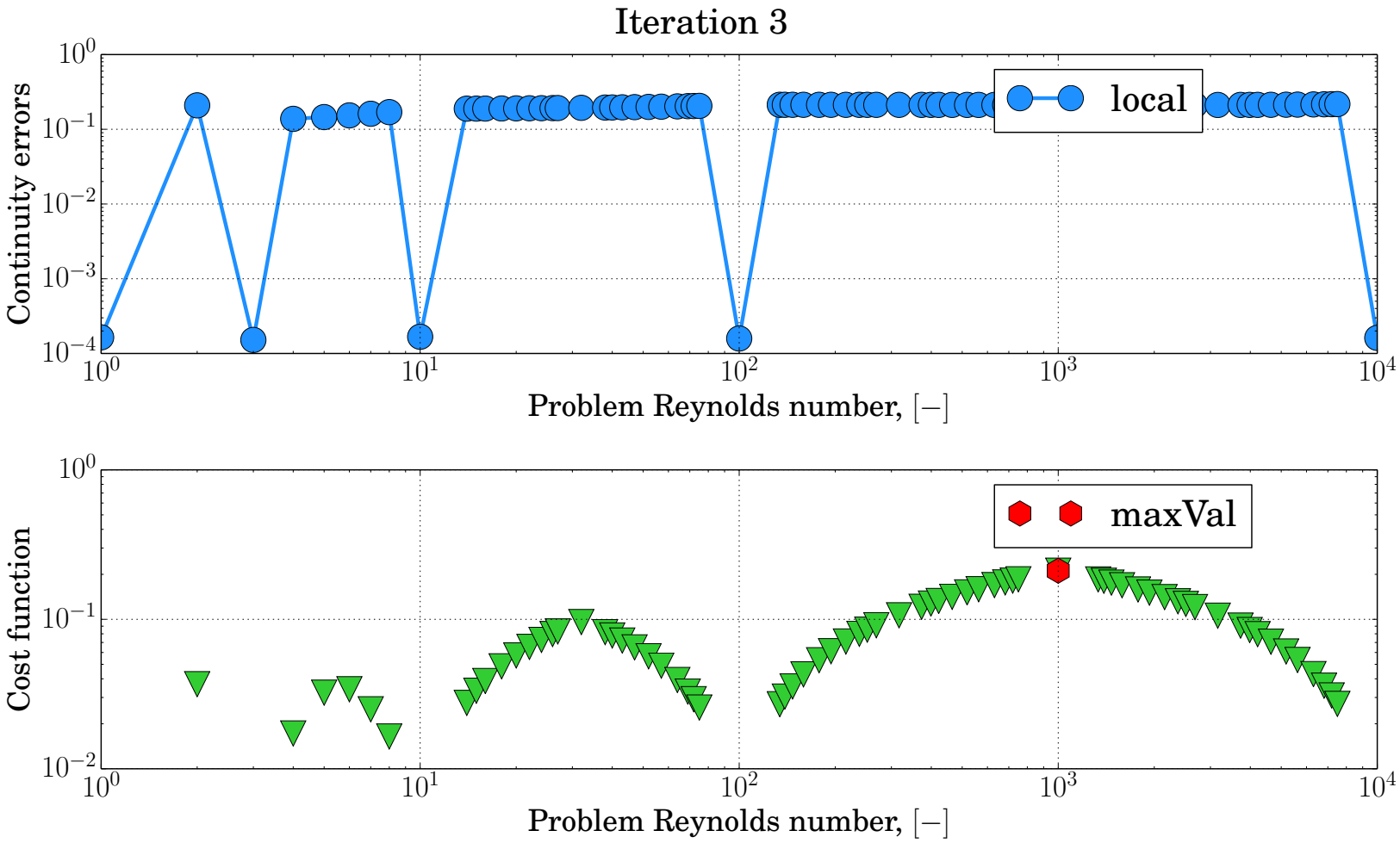
ROM size reduction: $\mathcal{S}^e$ selection based on greedy algorithm



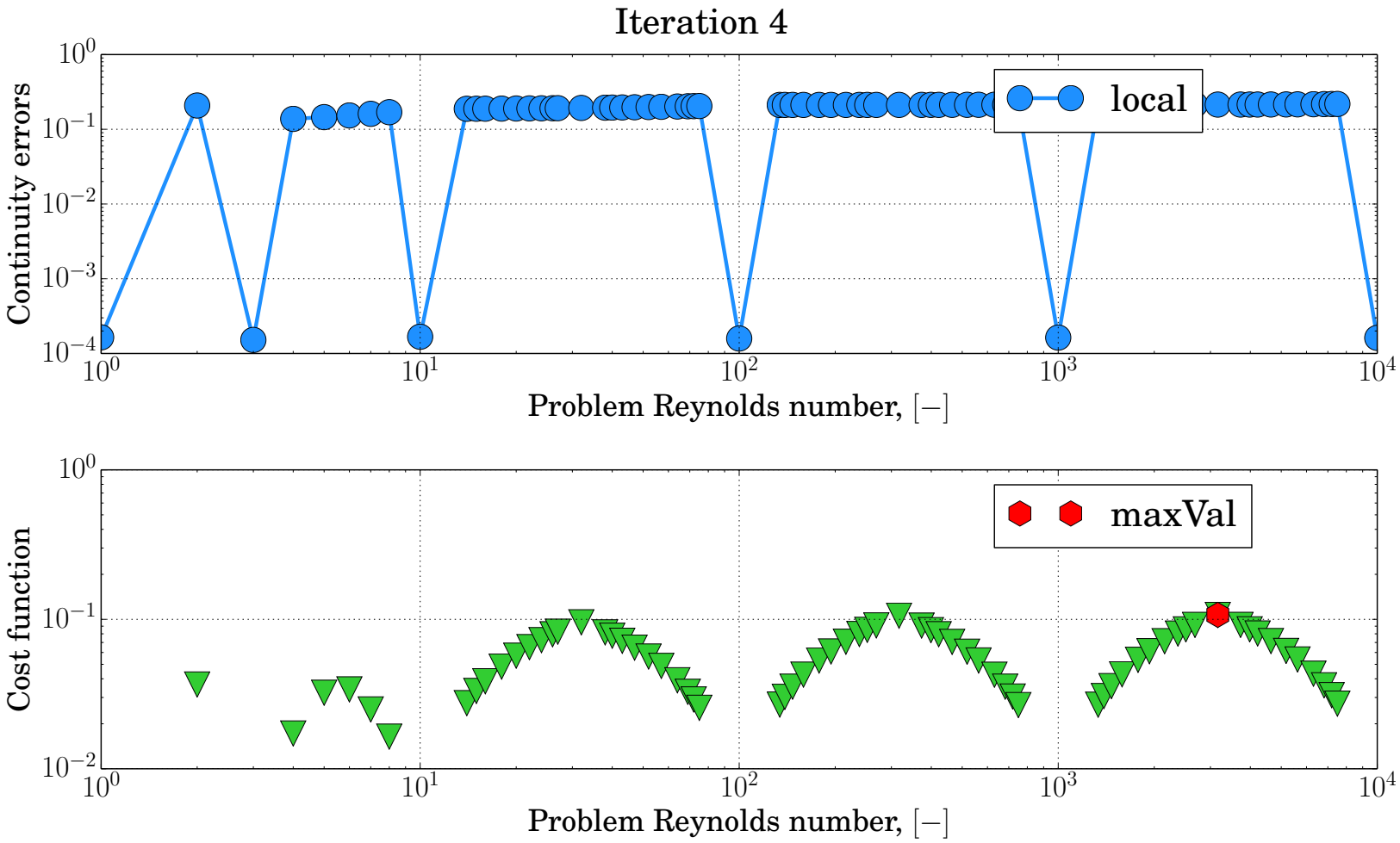
ROM size reduction: $\mathcal{S}^e$ selection based on greedy algorithm



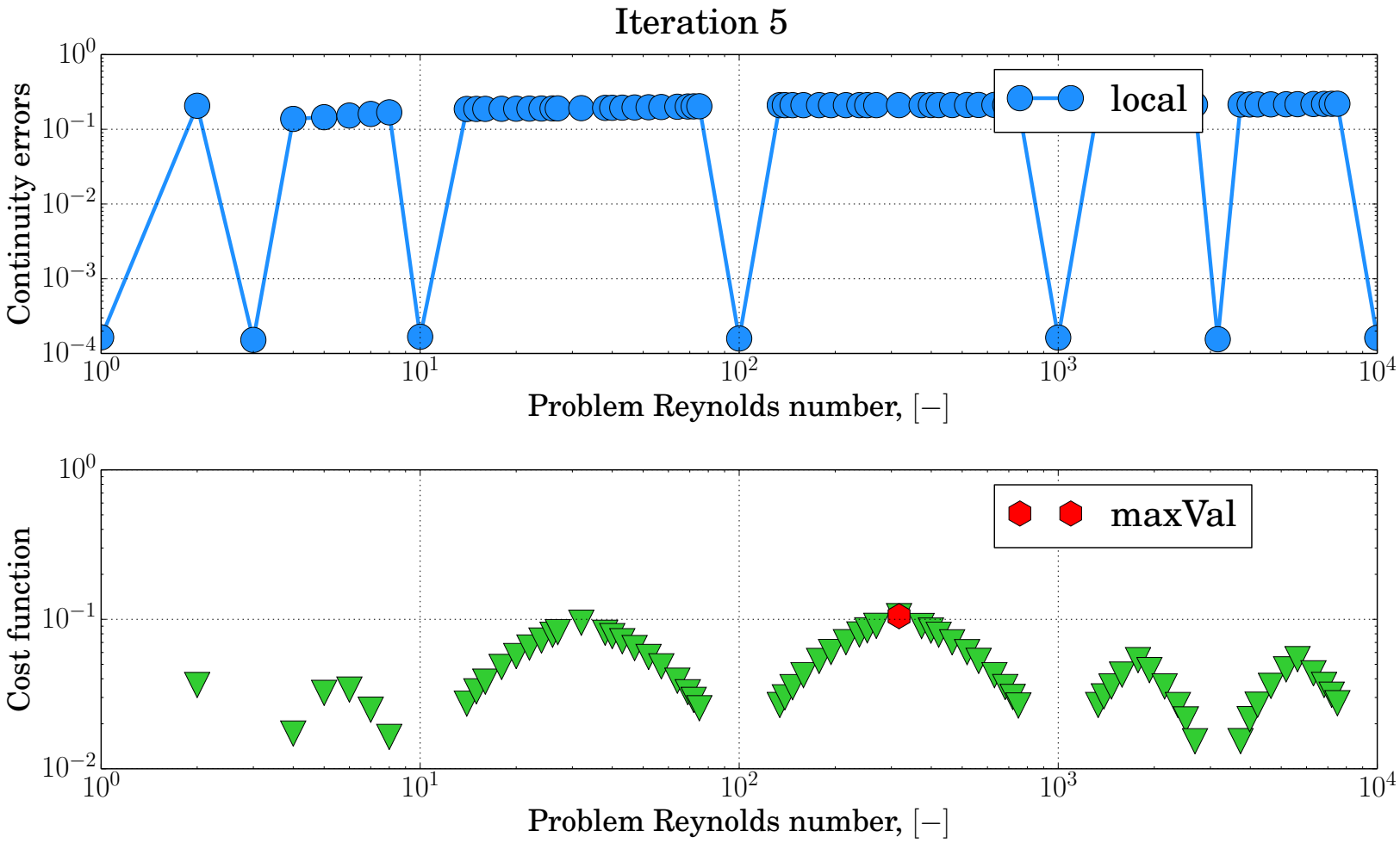
ROM size reduction: $\mathcal{S}^e$ selection based on greedy algorithm



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



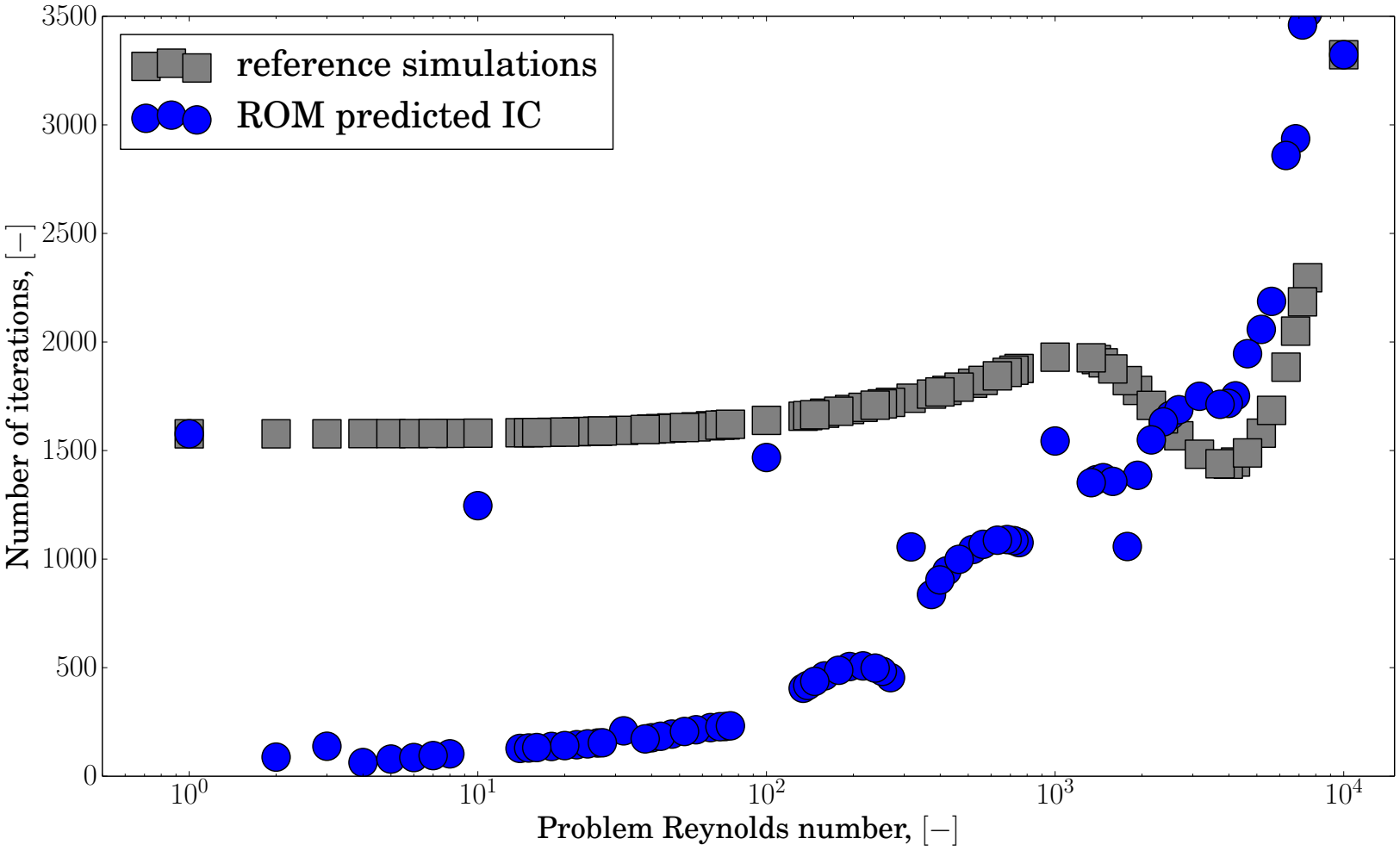
ROM size reduction: $\mathcal{S}^e$ selection based on greedy algorithm



Next steps

Towards ROM size reduction and multiparametric systems

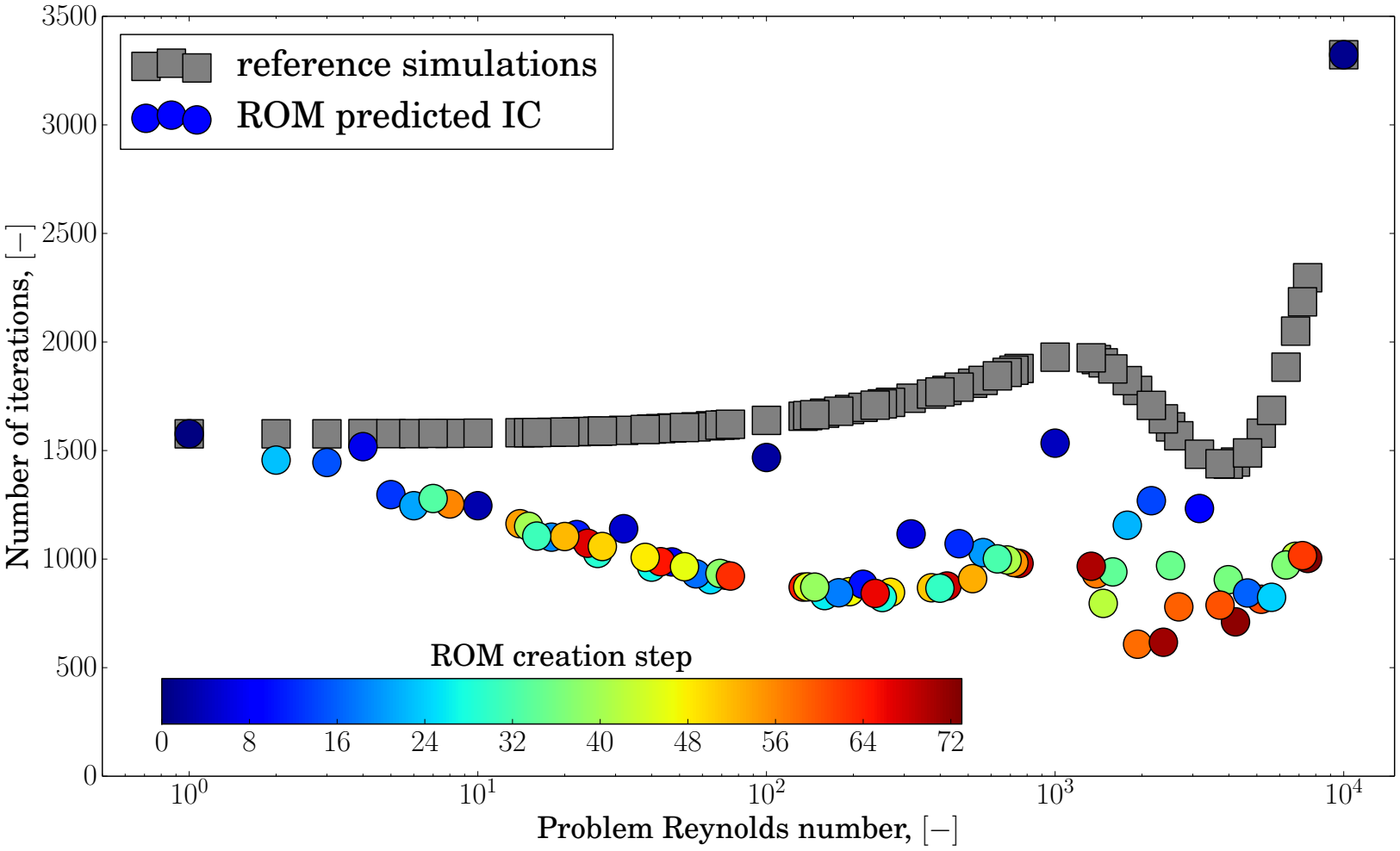
ROM size reduction: $\mathcal{S}^e$ selection based on greedy algorithm



Next steps

Towards ROM size reduction and multiparametric systems

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



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, *SIAM J. Sci. Comput.*, vol. 32, (2010) pp. 2737–2764.

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

UNIVERSITY OF
CHEMISTRY AND TECHNOLOGY
PRAGUE

Thank you for your attention

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

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, $\langle \varphi, \tilde{\varphi} \rangle_{L_h^2} = (\varphi^h)^{\mathrm{T}} W \varphi^h$.