Geometrically parametrized reduced-order models using OpenFOAM and ITHACA-FV

Giovanni Stabile

MathLab, Mathematics Area, SISSA International School for Advanced Studies, Trieste, Italy

Presentation at the 4th German OpenFoam User meetiNg
GOFUN - 2020
Braunschweig
22/04/2020

Outline

1. Introduction

- Overview of the physical problems

2. Intrusive Model Order Reduction - ITHACA-FV package

- A brief Introduction on POD-Galerkin methods
- Geometrically parametrized examples for heat transfer problems
- Geometrically parametrized examples for Navier-Stokes problems
- A short introduction to turbulent flows

3. Practical examples with NON-Intrusive model order reduction

- Non-intrusive python packages under development for POD-I, DMD and geometrical morphing
- Practical Examples with non-intrusive ROMs

4. Conclusions and Outlooks

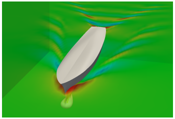
Team of **Prof. Gianluigi Rozza**[1] developing **Advanced Reduced Order Methods** with special focus on **Computational Fluid Dynamics**

[1] PI of the H2020 ERC Consolidar Grant AROMA-CFD, Grant Agreement 681447.

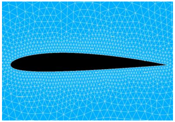
Overview of the physical problems

The interest is in **viscous parametrized incompressible flows**

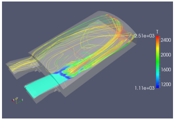
Industrial Flows



Naval Eng.

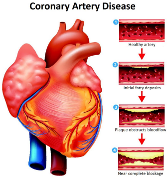


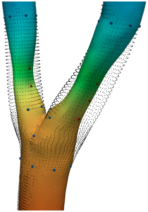
Aeronautics



Industrial App.

Biomedical Applications





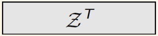
Possible applications can be found in **naval** and **nautical** engineering, **aeronautical** engineering and **industrial** engineering.

In general any application dealing with incompressible fluid dynamic problems that has the response depending on **parameter changes** (Reynolds Number, Grashof Number, Geometrical parameters ..)

Intrusive Reduced Order Methods in a nutshell

- $()^{\mathcal{N}}$: "truth" high order method (FEM, FV, FD, SEM) – to be accelerated
- $()_N$: reduced order method (ROM) – *the accelerator*

- **Offline**: very expensive preprocessing (high order): basis calculation (done *once*) after suitable parameters sampling (greedy, POD, ...)

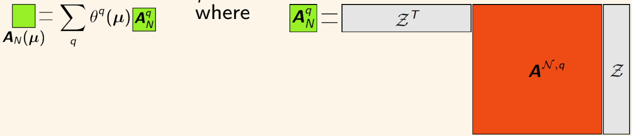


- **Online**: extremely fast (reduced order): real-time input-output evaluation

$$\boldsymbol{\mu} \quad \rightarrow \quad s_N(\boldsymbol{\mu})$$

thanks to an efficient assembly of problem operators

$$\boldsymbol{A}_N(\boldsymbol{\mu}) = \sum_q \theta^q(\boldsymbol{\mu}) \boldsymbol{A}_N^q, \text{ where } \boldsymbol{A}_N^q = \mathcal{Z}^T \boldsymbol{A}^{\mathcal{N},q} \mathcal{Z}$$



- Numerical issues: stability, error bounds, efficient parametrization, sampling, ...

J. S. Hesthaven, G. Rozza, B. Stamm. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations.* SpringerBriefs in Mathematics. Springer, 2015

Geometrically parametrized problems: The Idea

We have to rely on methods that work in the **physical domain** where the geometry is parametrized by a parameter vector $\boldsymbol{\mu}$. Let us take as example a simple Poisson problem:

$$\begin{cases} \int_{\Omega_i(\boldsymbol{\mu})} \mathrm{div}(\alpha_\theta \nabla\theta) \mathrm{d}\omega = \int_{\Omega_i(\boldsymbol{\mu})} f \mathrm{d}\omega & \forall \Omega_i(\boldsymbol{\mu}) \in \Omega(\mu), \\ \theta(x, \boldsymbol{\mu}) = \theta_D(x, \boldsymbol{\mu}) & \text{on } \Gamma_D(\boldsymbol{\mu}), \\ \theta_{\boldsymbol{n}}(x, \boldsymbol{\mu}) = \theta_N(\boldsymbol{\mu}, x) & \text{on } \Gamma_N(\boldsymbol{\mu}), \end{cases}$$

That after discretization becomes

$$\sum_{i=1}^{N_h} \int_{\partial\Omega_i(\boldsymbol{\mu})} \boldsymbol{n} \cdot (\alpha_\theta \nabla\theta) \mathrm{d}s = \sum_{i=1}^{N_h} \sum_{f=1}^{N_f} \alpha_{\theta_{if}} \boldsymbol{S_{if}} \cdot (\nabla\theta)_{if},$$

$$\sum_{i=1}^{N_h} \int_{\Omega_i(\boldsymbol{\mu})} f \mathrm{d}\omega = \sum_{i=1}^{N_h} f_i V_i.$$

and in Matrix Form:

$$\boldsymbol{A}(\boldsymbol{\mu})\boldsymbol{\theta}(\boldsymbol{\mu}) = \boldsymbol{f}(\boldsymbol{\mu}).$$

Geometrically parametrized problems: The Idea

It would be nice to perform an offline stage for some values of the parameter vector and then construct a reduced order model in a **straightforward way**:

$$\boldsymbol{L}^T \boldsymbol{A}(\boldsymbol{\mu}) \boldsymbol{L} \boldsymbol{a}^{\boldsymbol{\theta}} = \boldsymbol{L}^T \boldsymbol{f}(\boldsymbol{\mu}),$$

and to rewrite it as

$$\boldsymbol{A}^r(\boldsymbol{\mu}) \boldsymbol{a}^{\boldsymbol{\theta}} = \boldsymbol{f}^r(\boldsymbol{\mu}),$$

where:

$$\theta(x, \mu) \approx \sum_{i=1}^{N_\theta} (\mu) a_i^\theta(\mu) \phi_i^\theta(x),$$

However there are **issues** that need to be addressed:

- the FOM snapshots obtained during there training stage "live" in different domains. $\rightarrow$ **Obtain modes that consider this aspect.**
- the matrix $\boldsymbol{A}(\boldsymbol{\mu})$ needs to be assembled in the physical space and needs to be assembled also during the online stage. $\rightarrow$ **Rely on efficient hyper reduction techniques.**
- for a new value of the parameter one needs to deform the mesh also during the online stage. $\rightarrow$ **select a proper mesh motion strategy.**

Stabile-Zancanaro-Rozza, Efficient Geometrical parametrization for finite-volume based reduced order methods, *Int J Numer Methods Eng. 2020; 1– 28.* `https://doi.org/10.1002/nme.6324`

The Idea

To account that snapshots are defined in **different domains** a **modified POD with snapshots method** is used:

$$(\boldsymbol{C}^{\theta})_{ij} = \theta_i^T \boldsymbol{M}' \theta_j$$

with $\boldsymbol{M}'$ being the mass matrix for an average reference configuration.
To ensure offline-online decoupling we need to recover an **approximate affine expansion** of the differential operator $\boldsymbol{A}(\boldsymbol{\mu})$ and of the source term vector $\boldsymbol{f}(\boldsymbol{\mu})$.

$$\boldsymbol{A}(\boldsymbol{\mu}) = \sum_{k=1}^{N_A} b_k^A(\boldsymbol{\mu}) \boldsymbol{\chi}_k^A \ , \ \boldsymbol{f}(\boldsymbol{\mu}) = \sum_{k=1}^{N_f} c_k^f(\boldsymbol{\mu}) \boldsymbol{\chi}_k^f, \quad (1)$$

The matrix and vector basis functions $\boldsymbol{\chi}_k^A$ and $\boldsymbol{\chi}_k^f$ are computed using a Matrix and vector version of the **DEIM algorithm** with Frobenius inner product. The

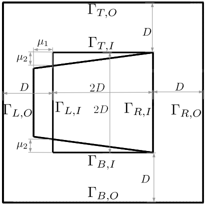
vectors of coefficients $b_k^A(\boldsymbol{\mu})$ and $c_k^f(\boldsymbol{\mu})$ are obtained during the online stage by point-wise evaluations of the Matrix A and the vector f.

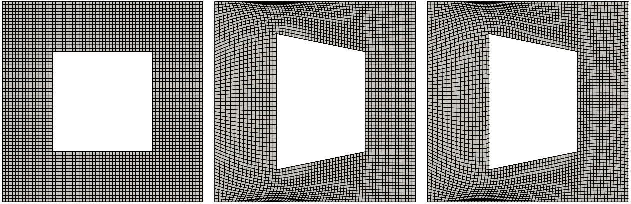
Stabile-Zancanaro-Rozza, Efficient Geometrical parametrization for finite-volume based reduced order methods, *Int J Numer Methods Eng. 2020; 1– 28.* https://doi.org/10.1002/nme.6324

Example for a Heat Exchange problem - **With M. Zancanaro**

Method tested on a **geometrically parametrized heat exchange problem**.

$\mathcal{P}^1_{train} = \{\boldsymbol{\mu}_{i_{train}}\}_{i=1}^{N_{train}} \in [-0.32, 0.32] \times [-0.32, 0.32]$. We set $N_{train} = 100$

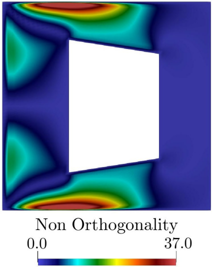




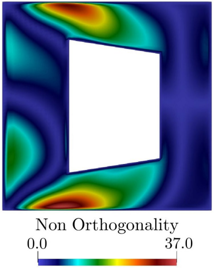
	$\Gamma_{L,O}$	$\Gamma_{T,O}$	$\Gamma_{R,O}$	$\Gamma_{B,O}$	$\Gamma_{L,I}$	$\Gamma_{T,I}$	$\Gamma_{R,I}$	$\Gamma_{B,I}$
B.C.	$\theta_{\boldsymbol{n}} = 10$	$\theta = 10$	$\theta_{\boldsymbol{n}} = -10$	$\theta = 10$	$\theta = -10$	$\theta = 0$	$\theta = 0$	$\theta = 0$

Comparison of the two mesh motion strategies

LAPL.



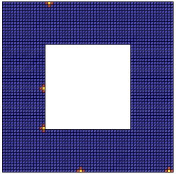
RBF



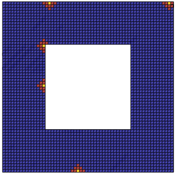
	Max Non-Ortho.	Average Non-Ortho.
LAPL.	29.774°	7.724°
RBF	25.518°	7.531°

Magic Points Location

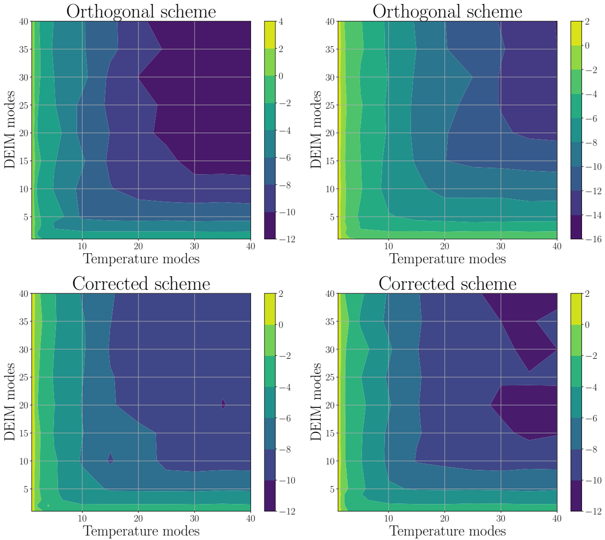
No-Correction



Correction



Numerical Example: Error computation



Speed-up between the ROM and FOM varies between 150 and 40 depending on number of POD and DEIM modes.

The ROM SIMPLE algorithm

Let us imagine to write down the two discretized **incompressible Navier-Stokes** equations of the **SIMPLE** algorithm (i.e. momentum and pressure ones) as follows:

$$\mathbf{A}_u\,\mathbf{u} = \mathbf{f}_u, \quad \mathbf{A}_P\,\mathbf{p} = \mathbf{f}_P$$

We can then **project** them into a reduced space by the use of the modes matrices $\mathbf{Z}_u$ and $\mathbf{Z}_P$:

$$\mathbf{Z}_u^T\mathbf{A}_u\mathbf{Z}_u\,\tilde{\mathbf{u}} = \mathbf{Z}_u^T\mathbf{f}_u, \quad \mathbf{Z}_P^T\mathbf{A}_P\mathbf{Z}_P\,\tilde{\mathbf{p}} = \mathbf{Z}_P^T\mathbf{f}_P$$

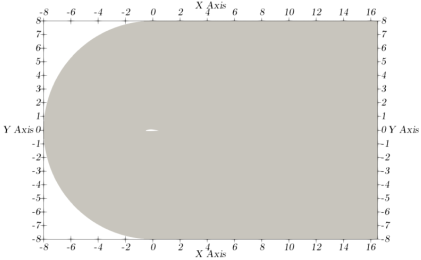
Where the velocity and pressure fields are approximated as:

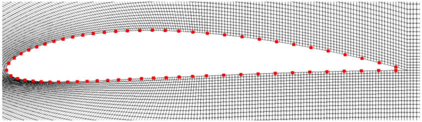
$$\boldsymbol{u} \approx \sum_{i=1}^{N_u} \tilde{u}_i\phi_i^u, \quad \boldsymbol{p} \approx \sum_{i=1}^{N_p} \tilde{p}_i\phi_i^p$$

And $\boldsymbol{Z_u}$ and $\boldsymbol{Z_p}$ are matrices which contain the velocity and pressure modes, respectively:

$$\boldsymbol{Z}_u = [\phi_1^u, \dots, \phi_{N_u}^u], \quad \boldsymbol{Z}_u = [\phi_1^u, \dots, \phi_{N_u}^u]$$

Geometrical Parametrization of a NACA 4412 profile

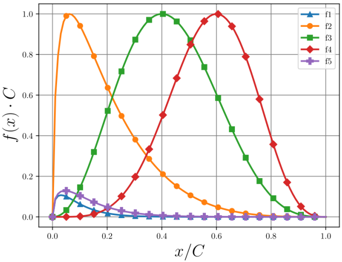




	Domain-X	Domain-Y	wing-chord
Dimension	24.5	16	1

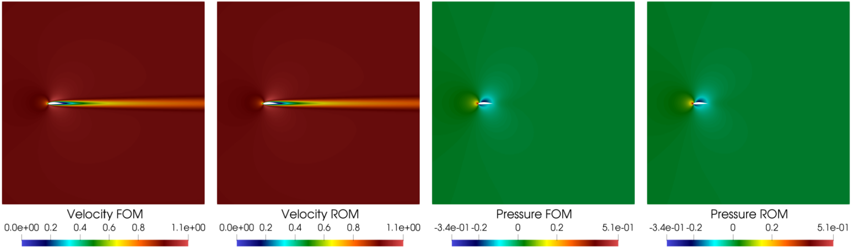
The mesh is deformed summing on the lower and upper part of the wing 5 bumb functions.

Therefore we have **10 different geometrical parameters**. The training set contains 100 training samples inside $\mathcal{K}_{train} = \{\boldsymbol{\kappa}_{i_{train}}\}_{i=1}^{N_{train}} \in [0, 0.02]^{10}$.

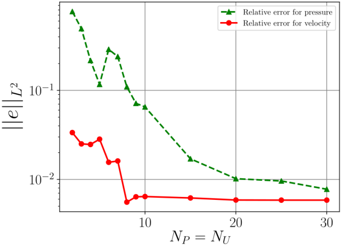


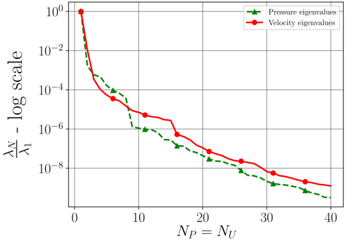
Geometrical Parametrization - Test case

Comparison of velocity and pressure snapshots



Quantitative error computation and eigenvalue decay





What about Turbulent Flows? - **With S. Hijazi**

- We use just the decomposition of **the eddy viscosity field**:

$$\nu_t(\boldsymbol{x}, t; \boldsymbol{\mu}) \approx \sum_{i=1}^{N_{\nu_t}} g_i(t, \boldsymbol{\mu})\eta_i(\boldsymbol{x}),$$

- The problem is now to **compute the coefficients $\boldsymbol{g}$** of the eddy viscosity equations **without relying on the projection** of the equations $\rightarrow$ **POD-I**.

$$\forall \boldsymbol{\mu}_k \in \mathcal{P}_{train}, \quad \boldsymbol{u}(\mu_k) \approx \boldsymbol{u}^N(\boldsymbol{\mu}_k) = \sum_{i=1}^{N} a_i(\boldsymbol{\mu}_k)\boldsymbol{\phi}_i,$$

$$\boldsymbol{u}^N_{NEW} = \sum_{i=1}^{N} a_i(\boldsymbol{\mu}_{NEW})\phi_i.$$

- Each function $a_i(\mu)$ is approximated using **approximated interpolant functions.**
- It relies only on the **snapshots**: it does not require any information about the system (**non-intrusive** approach).
- The interpolation is carried out using **Radial Basis Functions.**

S. Hijazi, G. Stabile, A. Mola, G. Rozza. *Data-driven POD-Galerkin reduced order model for turbulent flows.* 2019

The ITHACA-FV computational library

- **ITHACA-FV** (In real Time Highly Advanced Computational Applications for Finite Volumes) is a C++ implementation based on **OpenFOAM** (tested on v. 5-6, v. 1812-1906-1912) of several reduced order modeling techniques.
- It is mainly developed and maintained at **SISSA mathLab** but counts already several developers and users around the world.
- It is Open-Source and publicly available on **GitHub** (https://github.com/mathLab/ITHACA-FV).
- It has been successfully used to perform **intrusive** and **non-intrusive** model order reduction for stationary and unstationary fluid dynamic problems, heat transfer problems, coupled heat transfer and fluid dynamics problems.
- Dense Linear Algebra is based on the **Eigen** C++ library.

NonIntrusive ROMs with OF - N.Demo, M.Tezzele, F.Salmoiraghi

In the group during the years we have developed also several packages for Non-intrusive ROMs and geometrical morphing.

- The **proper orthogonal decomposition with interpolation**[2] is a method to approximate the numerical solution of a parametric partial differential equations as combination of few solutions computed for some properly chosen parameters.
- The **dynamic mode decomposition**[3] is an algorithm describing a nonlinear time-dependent problem as combination of few main structures that evolve linearly in time.
- For **geometrically parametrized problems** one needs a proper set of tools[4] to deform the different geometries.

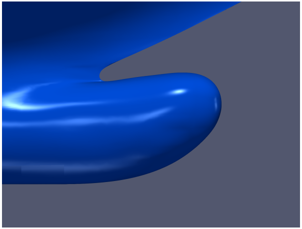
[2] EZyRB is available at: https://github.com/mathLab/EZyRB
[3] PyDMD is available at: https://github.com/mathLab/PyDMD.
[4] PyGeM is available at: https://github.com/mathLab/PyGeM

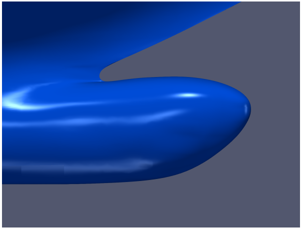
The shape optimization problem

Keywords: **#ShapeOptimization #ModelReduction #HullResistance**

A standard shape optimization system needs a **geometrical modeler** for shape morphing, an **high-fidelity solver** for evaluate the objective function and an **optimization algorithm**. The optimization cycle can last even **months**.

We introduce in the pipeline two different **model reduction** techniques in order to improve the performances.





N. Demo, M. Tezzele, G. Gustin, G. Lavini, G. Rozza, Shape optimization by means of proper orthogonal decomposition and dynamic mode decomposition, *Technology and Science for the Ships of the Future: Proceedings of NAV 2018: 19th International Conference on Ship & Maritime Research, IOS Press*, pp. 212–219, 2018.

Conclusion

What we have done...

- We developed a reduced order modeling pipeline based on OpenFOAM exploiting both intrusive and non-intrusive methods;
- All the developed methods are available as open source packages;
- The non-intrusive tools have presented for an OpenFOAM application but are completely independent from the full-order solver used (you can plug what you prefer) and the parametrization tools.

... and what else?

- Extensions to compressible flows.
- Development of tools for the reduction of the number of geometrical parameters.
- Better exploitation of machine learning tools for non-intrusive methods in the context of geometrically parametrized problems.