Research article

A Gaussian Process Regression approach within a data-driven POD framework for engineering problems in fluid dynamics

Giulio Ortali[1,2]**, Nicola Demo**[1,*] **and Gianluigi Rozza**[1]

[1] Mathematics Area, mathLab, SISSA, via Bonomea 265, I-34136 Trieste, Italy

[2] Department of Applied Physics, Eindhoven University of Technology, The Netherlands

* **Correspondence:** Email: nicola.demo@sissa.it.

Abstract: This work describes the implementation of a data-driven approach for the reduction of the complexity of parametrical partial differential equations (PDEs) employing Proper Orthogonal Decomposition (POD) and Gaussian Process Regression (GPR). This approach is applied initially to a literature case, the simulation of the Stokes problem, and in the following to a real-world industrial problem, within a shape optimization pipeline for a naval engineering problem.

Keywords: data-driven method; reduced order modeling; Gaussian Process Regression; parametric design problem

1. Introduction and motivations

Reduced Order Modeling (ROM) offers a simplification for very complex computational models [25], keeping by high accuracy for the solution we want to compute but, at the same time, reducing the computational cost to reach it. In a parametric context, this means that we are able to approximate the solution for any new unforseen parameter in a very efficient and rapid way. Especially for complex phenomena of which numerical solutions are expensive to compute — e.g., solve a Partial Differential Equations (PDEs) —, ROM makes a huge difference in terms of the global computational load, allowing for a quick (or even real-time) response and for a multiple queries approach, e.g., an optimization scenario, otherwise not viable.

In the Reduced Basis (RB) [13] method, a set of truth solutions for some selected parameters are computed using a numerical solver over the full-order model. Once these snapshots are collected, we combine them to extract the basis which spans the low-dimensional space, thus we can exploit it by using the basis as global test functions in a Galerkin method. Thanks to the low dimensionality of the reduced order model, the related system results smaller and faster to be solved. Despite this advantage, RB meets some difficulties for applications in industrial field: the necessity to extract the

discrete operators precludes the use of commercial software, as well as the effort required to derive the equations onto the reduced space discourages its exploitation.

Data-driven reduced order model are then gaining popularity to tackle industrial problems, mainly for its easiness of application and for the increasing quantity of available scientific data. Within the data-driven approach, it is possible avoiding to project equations and operators onto the reduced space — that can be, depending on the equations, very demanding and complex — and approximating the solution maninfold in the reduced space with an interpolation/regression. In this contribution, we coupled the Proper Orthogonal Decomposition (POD), a well-known technique for computation of reduced space basis [2], with the Gaussian Process Regression (GPR) for a probabilistic prediction of the new solutions. While the POD technique is pretty standard for projection based snapshots method in several fields of application — e.g., structural engineering [22], fluid dynamics [15, 29, 30], — the GPR has been adopted only in [12] for a structural problem. The POD is involved to reduce the large dimension of the input snapshots and the GPR is applied as a probabilistic response surface for the approximation of the parametric solution manifold. The novelty of this work is in fact the creation of a computational framework by combining model order reduction and probabilistic prediction, in order to increase the accuracy of the state-of-the-art (data-based) metodologies for a very limited amount of snapshots. This framework results then particularly suited for industrial setting, where simulations can last days or even weeks, making unfeasible the computation of large database of solutions. Similar methods have be proposed in the last year to data-based modeling of parametric problems, using interpolation to approximate the solution manifold. Among all contributions in literature, we cite [7,27,32] for a good overview. An alternative to interpolation or regression for the online prediction is constituted by the Artificial Neural Network (ANN), as described in [14]. Such approach seems very accurate, but previous works [21, 36] shown that typically a large dataset is required to properly reconstruct the solution manifold. Since our effort is dedicated in this work to reduce as possible the number of high-fidelity snapshots, we do not investigate the latter methodology, postponing to future works the possibility to better explore it. Approaches that exploit the Active Subspaces (AS) property for the reduction of the number of required input snapshots are presented in [9,33].

The paper is structured as follows: the POD and GPR methods are described more in details in Section 2 and Section 3, respectively, while in Section 4 we describe their integration into a single framework. The numerical results are presented in Section 5 and they are subdivided as following: first we applied the POD-GPR framework to a simply toy problem, a Stokes flow passing around a cylinder, then we move to an industrial problem where turbulent biphase Navier Stokes flow is simulated around a parametric cruise ship geometry. Finally, we summarize some conclusion and further future perspectives in Section 6.

2. Proper orthogonal decomposition for dimensionality reduction

POD is a technique used in the context of PDEs aiming at reducing the computational cost required to obtain their solution numerically. In this section we introduce the reader to the general idea behind POD and give some inshights on the mathematical theory supporting it.

We now begin with an introduction to the class of Reduced Basis (RB), presenting POD in the following as a special case. Finally, we discuss a modification of the standard POD formulation, which is equation free and it will be the one used in this work.

Reduced Basis. We begin with a formal definition of a system of discretized parametric PDEs in the form:

$$\text{find } u^{\mathcal{N}}(\boldsymbol{\mu}) \in \mathbb{X}^{\mathcal{N}} \text{ s.t.: } a(u^{\mathcal{N}}(\boldsymbol{\mu}), w; \boldsymbol{\mu}) = F(w; \boldsymbol{\mu}) \quad \forall w \in \mathbb{X}^{\mathcal{N}}, \tag{2.1}$$

where $\boldsymbol{\mu} \in \mathbf{P} \subset \mathbb{R}^p$ is the parameter, $u^{\mathcal{N}}$ is the truth solution to our problem, $a(\cdot, \cdot; \boldsymbol{\mu})$ is the parametric bilinear form and $\mathbb{X}^{\mathcal{N}}$ is a finite dimensional space of dimension $\mathcal{N}$.

We introduce the notion of the solution manifold, that is the set of all possible solutions of our parametric problem under the variation of the parameter:

$$\mathbb{M}^{\mathcal{N}} = \{u^{\mathcal{N}}(\boldsymbol{\mu}) \ \boldsymbol{\mu} \in \mathbf{P}\}. \tag{2.2}$$

The final goal of RB is to approximate any element of this manifold using a low number of basis functions, or modes, $\{\chi_i(x)\}_{i=1}^{N}$ by setting what we call the reduced basis. These N functions are globally defined over the computational domain and are obtained using some pre-computed truth solutions for particular parameter values.

The reduced solution $u_N^{\mathcal{N}} \approx u^{\mathcal{N}}$ is hence composed by a suitable linear combination of these modes:

$$u_N^{\mathcal{N}}(\boldsymbol{\mu}) = \sum_{i=1}^{N} \xi_i(\boldsymbol{\mu})\chi_i(x), \tag{2.3}$$

and the reduced formulation of the problem (2.1) becomes:

$$\text{find } u_N^{\mathcal{N}}(\boldsymbol{\mu}) \in \mathbb{X}_N^{\mathcal{N}} \text{ s.t.: } a(u_N^{\mathcal{N}}(\boldsymbol{\mu}), w; \boldsymbol{\mu}) = F(w; \boldsymbol{\mu}) \quad \forall w \in \mathbb{X}_N^{\mathcal{N}}, \tag{2.4}$$

where $\mathbb{X}_N^{\mathcal{N}} = span(\{\chi_i(x)\}_{i=1}^{N})$.

Hence, while the degrees of freedom associated with the truth solution $\mathcal{N}$ are typically high, the degrees of freedom associated with the RB approximation are only N, where $N \ll \mathcal{N}$.

Proper Orthogonal Decomposition. POD refers to a particular RB which considers a hierarchical orthonormal basis generated using energetic considerations, exploting them as global basis function for a Galerkin framework [2]. The mathematical tool behind POD is Singular Value Decomposition (SVD), as we will briefly discuss now.

Let us begin with some notation. We will call a possible outcome of our system $\mathbf{y}$ a snapshot and denote with $\mathcal{N}$ its dimensionality ($\mathbf{y} \in \mathbb{R}^{\mathcal{N}}$). We will then denote with n the number of snapshots collected, and gather them in the *snapshots' matrix* $\mathbf{Y} \in \mathbb{R}^{\mathcal{N} \times n}$:

$$\mathbf{Y} = \begin{bmatrix} | & | & & | \\ \mathbf{y}_1 & \mathbf{y}_2 & \cdots & \mathbf{y}_n \\ | & | & & | \end{bmatrix}. \tag{2.5}$$

We then call N the number of POD modes, i.e., the dimension of the reduced basis, and assume $N \le n < \mathcal{N}$.

A formal definition of the POD basis can be then given:

Theorem 1. *(POD basis) Given* $\mathbf{Y} \in \mathbb{R}^{N \times n}$, $\{\chi_i\}_{i=1}^N$, *for* $N \in \{1, .., n\}$ *is the POD basis of* $\mathbf{Y}$ *if and only if it is a solution of:*

$$\max_{\tilde{\psi}_1, \tilde{\psi}_2, .. \tilde{\psi}_N} \sum_{i=1}^N \sum_{j=1}^n | < \mathbf{y}_j, \tilde{\psi}_i >_{\mathbb{R}^N} |^2 \quad s.t. \ < \tilde{\psi}_i, \tilde{\psi}_i >_{\mathbb{R}^N} = \delta_{i,j}, \quad for \ 1 \le i, j \le N. \tag{2.6}$$

This can be read as: the POD basis is the one that maximizes the similarity (as measured by the square of the scalar product) between the snapshots matrix and its elements, under the constraint of orthonormality. In this sense, when we obtain the N-rank POD basis, we have the set of dimension N capable of optimally express the variance in the snapshots.

The link between POD and SVD is stated in following theorem, that can be proven using Lagrangian penalization techniques (see for example [35]):

Theorem 2. *Given* $\mathbf{Y} \in \mathbb{R}^{N \times n}$ *of rank* $d = n < \mathcal{N}$, *its N-rank POD basis is given by the set of the first N left singular vectors* $\{\psi_i\}_{i=1}^N$.

As a last remark on POD, we highlight the fact that, as already shown, to compute the reduced basis we need a set of what can be called, using machine learning nomenclature, training data, that is a set of truth solutions for particular values of the parameters computed using the high fidelity solver. This is usually the most expensive phase in the ROM pipeline, and the applicability of ROMs techniques is often related to the size of the sampling needed to obtain certain performances.

Connected to this last aspect, we mention here another important feature of ROMs, that is the offline-online decomposition. By this, we mean that the use of RB techniques can be split, as we have already seen, in two phases:

- an *offline-phase*, in which a set of high fidelity solutions are collected and the reduced basis is computed by combining them;
- an *online-phase*, in which we project the problem onto the reduced space and the solutions for new parameters are computed in an efficient manner.

The first phase requires typically more computational time, having to rely on the high fidelity solver, and is usually executed in high performance computing clusters using parallel programming techniques. The second phase, however, is quite inexpensive from the computational point of view and could be done in low-end terminal or even tablet/smartphones, widening the field of applicability of this method.

Data-driven Proper Orthogonal Decomposition. Data-driven POD is an alternative to standard POD that is more used in the industrial setting because of its ease of use and its flexibility. We define here the data-driven POD as the class of methods that exploit POD for dimensional reduction and any non-intrusive approximation for the reconstruction of the solution method. It includes, for example, the POD with interpolation (PODI) procedure [7], as well as the other possible techniques already discussed in Section 1. Within this method, the first phase, the generation of the reduced basis, is performed in the same way as the standard approach. A significant difference, however, is that while in intrusive POD-Galerkin we need to rely upon an open-source software to compute the truth solutions (since in the online phase we will need to have access to the source code in order to project the

equations). In the data-driven approach, this is not necessary, and we can use commercial software or even experimental data to train our model. The reason why in data-driven POD methods we have this freedom in the offline phase is that the online phase is completely different from the standard POD.

We recall here that the assumption of RB-ROMs is that the truth solution of our problem u^N can be approximated by the reduced solution u_N^N composed by linear combination of spatial modes $\chi_i(x)$ multiplied by coefficients $\xi_i(\boldsymbol{\mu})$, that is:

$$u^N(\boldsymbol{\mu}) \approx u_N^N(\boldsymbol{\mu}) = \sum_{i=1}^{N} \xi_i(\boldsymbol{\mu})\chi_i(x). \tag{2.7}$$

In the data-driven POD, the online phase consists of building an interpolation, or a regression, using high-fidelity data coming from offline phase, approximating the function that associates the value of the parameter $\boldsymbol{\mu}$ to the modal coefficients of the related solution $\{\xi_i(\boldsymbol{\mu})\}_{i=1}^{N}$. We can use the interpolator to infer the value of the coefficients associated with new parameters. The values of the coefficients are then used to reconstruct the approximated truth solution using (2.7).

This approach is entirely data-driven and is independent both on the equations and on the physics of the problem. This has its own advantages and disadvantages. The ease of use and the complete freedom in the generation of the snapshots, that are crucial in an industrial setting in which commercial software are widely spread, correspond to a lower accuracy associated with the reduced model.

3. Gaussian Process Regression model

We introduce in this section the Gaussian Process Regression (GPR), the supervised learning technique we adopt to approximate the modal coefficients. We provide in the following lines a summary of the method, in order to let the reader understand the entire pipeline, but avoiding to touch more details; for a complete description of such framework, we suggest [24].

A Gaussian Process (GP) is a stochastic process whose finite-dimensional distributions are Gaussian. In supervised learning, this process is usually adopted for regression manner, by constructing a stochastic model from a set of input data capable to predict quantities of interest. We define the set[*] $\mathcal{D} = \{(\bar{x}_i, \bar{y}_i)\}$ for $i = 1, 2, \ldots, M$ as the set containing all the input-output pairs, where $\bar{x}_i \in \mathcal{P} \subset \mathbb{R}^m$ is the input parameter and $\bar{y}_i \in \mathbb{R}$ is the corresponding output. We assume the output follows an unknown regression function $f : \mathcal{P} \to \mathbb{R}$ we want to approximate (we avoid in this simple overview to deal with noise), such that $\bar{y}_i = f(\bar{x}_i)$ for $i = 1, 2, \ldots, M$. Thus, we define a GP such that:

$$f(x) \sim \text{GP}(\mu(x), \mathbf{K}(x)), \tag{3.1}$$

where $\mu(x)$ refers to the mean of the distribution and $\mathbf{K}(x)$ to its covariance, defined as $\mathbf{K}_{ij} = \mathcal{K}(\bar{x}_i, \bar{x}_j)$. The covariance function, also called *kernel*, $\mathcal{K} : \mathcal{P} \times \mathcal{P} \to \mathbb{R}$ should be properly selected depending on the problem, for this work we use the squared exponential:

$$\mathcal{K}(\bar{x}_i, \bar{x}_j) = \sigma^2 \exp\left(-\frac{\|\bar{x}_i - \bar{x}_j\|^2}{2l}\right). \tag{3.2}$$

---

[*]we use $\bar{x}_i$ and $\bar{y}_i$ here for input and output data, in order to distinguish, respectively, from spatial coordinates and snapshots.

The prior joint Gaussian distribution for the outputs y results then[†]

$$\mathbf{y}|\mathbf{X} \sim \mathcal{N}(0, \mathbf{K_{XX}}),$$

where $\mathbf{y} \in \mathbb{R}^M$ is the output vector and $\mathbf{X} \in \mathbb{R}^{m \times M}$ is the matrix whose columns are the input parameters. We specify that, for sake of compactness, we adopt a compact notation for the covariance matrices, such that $\mathbf{K_{XX}} = (\mathcal{K}(\bar{x}_i, \bar{x}_j))$ with $i, j = 1, \ldots, M$, where $\bar{x}_i$ and $\bar{x}_j$ are the i-th and j-th columns of $\mathbf{X}$. We are now interested in predicting the output for new test input $\bar{\mathbf{X}}$ by exploiting the joint distribution based on the above prior, that is:

$$\begin{bmatrix} f(\mathbf{X}) \\ f(\bar{\mathbf{X}}) \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \mathbf{K_{XX}} & \mathbf{K_{X\bar{X}}} \\ \mathbf{K_{\bar{X}X}} & \mathbf{K_{\bar{X}\bar{X}}} \end{bmatrix} \right). \tag{3.3}$$

Now, the ouputs $\bar{\mathbf{y}} = f(\bar{\mathbf{X}})$ can be expressed in probabilistic terms by simply using the conditional Gaussian distribution:

$$\bar{\mathbf{y}}|\bar{\mathbf{X}}, \mathbf{X}, \mathbf{y} \sim \mathcal{N}(\mathbf{m}, \mathbf{C}), \quad \mathbf{m} = \mathbf{K_{\bar{X}X}}\mathbf{K_{XX}^{-1}}\mathbf{y}, \quad \mathbf{C} = \mathbf{K_{\bar{X}\bar{X}}} - \mathbf{K_{\bar{X}X}}\mathbf{K_{XX}^{-1}}\mathbf{K_{X\bar{X}}}. \tag{3.4}$$

It is important to remark that, for a good result regarding the prediction, we need to optimize the hyperparameters within the covariance function. In this work, since a squared exponential kernel (Eq 3.2) has been adopted, we optimize the hyperparameters σ and l, respectively the variance and the lenghtscale, by maximizing the likelihood through a standard multi-start grandient-based optimization algorithm.

In this work, concerning the GPR, we use the Python open source package called GPy [11].

4. The POD-GPR framework

In this section we integrate the POD and GPR methods for a data-driven probabilistic reduced order modeling framework. The general idea is to compute the low-dimensional representation of the input snapshots through the POD, then infer their distribution in the parameters space in order to approximate the solution manifold.

The only required ingredients for the method are then the snapshots matrix $\mathbf{Y} = \begin{bmatrix} \mathbf{y}_1 & \ldots & \mathbf{y}_n \end{bmatrix} \in \mathbb{R}^{N \times n}$ and the corresponding parameters matrix $\mathbf{X} = \begin{bmatrix} \boldsymbol{\mu}_1 & \ldots & \boldsymbol{\mu}_n \end{bmatrix} \in \mathbb{R}^{m \times n}$, where $\boldsymbol{\mu}_i$ refers to the parametric configuration of the system for the snapshots $\mathbf{y}_i$ for $i = 1, \ldots, n$. The parameter points are collected by randomly sampling the parameter space. As introduced in Section 2, we use the POD algorithm in order to obtain the modes and reduce the snapshots dimension such that $\boldsymbol{\Xi} = \begin{bmatrix} \boldsymbol{\xi}_1 & \ldots & \boldsymbol{\xi}_n \end{bmatrix} = \mathbf{U}^T \mathbf{Y} \in \mathbb{R}^{N \times n}$, where $\mathbf{U} = \begin{bmatrix} \chi_1 & \ldots & \chi_N \end{bmatrix}$ is the matrix whose columns are the POD modes. Simplifying, the columns of $\boldsymbol{\Xi}$ are the N-dimensional representation of the corresponding column of the snapshots matrix.

We can now exploit all this information by treating all the different components of the modal coefficients as indipendent scalar outputs. Formally, we define the i-th components of the j-th modal coefficient ξ_j^i, such that $\boldsymbol{\xi}_j = \begin{bmatrix} \xi_j^1 \ldots \xi_j^N \end{bmatrix}^T$ with $i = 1, \ldots, N$ and $j = 1, \ldots, n$. Analyzing the components independentely by considering each components as a scalar output we are able to construct

---

[†]for notational simplicity, we exclusively use zero-mean priors.

a probabilistic response surface for each component, then exploiting it to approximate the modal coefficients for unseen parameters. We define N train ouputs $\Xi_k = \begin{bmatrix} \xi_1^k & \dots & \xi_n^k \end{bmatrix}$ for $k = 1, \dots, N$, whereas the train and test parameters are defined $\mathbf{X}$ and $\bar{\mathbf{X}}$, respectively. Using Eq 3.4 we can approximate the outputs — the modal coefficient — for any new parameters:

$$\boldsymbol{\xi}_* = \begin{bmatrix} \xi_*^1 & \dots & \xi_*^N \end{bmatrix}^T = \begin{bmatrix} \bar{\Xi}_1 | \bar{\mathbf{X}}, \mathbf{X}, \Xi_1 & \dots & \bar{\Xi}_N | \bar{\mathbf{X}}, \mathbf{X}, \Xi_N \end{bmatrix}^T \in \mathbb{R}^N, \tag{4.1}$$

where the $\boldsymbol{\xi}_*$ indicates the test modal coefficient — with the j component that indicates the coefficient with respect to the j POD mode — and $\bar{\Xi}_k$ for $k = 1, \dots, N$ refers then to the predicted k coefficient. We remark here that any component of the modal coefficients has been treated in an independent way. We remark that in this work a response surface is constructed for each of the output components, treating these latter in an independent manner with N different GPs. The POD modes are finally exploited in order to return to the high-dimensional space and obtain the final approximation. In matrix form:

$$\mathbf{y}_* = \mathbf{U}\boldsymbol{\xi}_*, \tag{4.2}$$

where $\mathbf{y}_* \in \mathbb{X}^N$ is the solution (for the test parameter $\bar{\mathbf{X}}$).

5. Numerical results

We present in this section the numerical results obtained by applying the described method to two different computational fluid dynamics simulations. To emphasize the versatility of the proposed framework with respect to the high-fidelity solver, we firstly apply it to a finite element (FE) problem then to a finite volume (FV) one. The coupling between the original model and the reduced one is in fact based only on data, resulting in a complete modular pipeline. In the next sections, we briefly introduce both the full-order models before providing a discussion about the reduction accuracy, even if in this case the original model has not a strong impact in the generation of the reduced order space, in order to give to the reader the possibility to reproduce the results.

5.1. Parametric Stokes flow around cylinder

The first numerical experiment where we use the POD-GPR framework is the simulation of a parametric Stokes flow passing around a circular cylinder. We define $u(\boldsymbol{\mu}) : \Omega \to \mathbb{R}^2$ and $p(\boldsymbol{\mu}) : \Omega \to \mathbb{R}$ the velocity and the pressure parametric fields, respectively, such that:

$$\begin{aligned}
\mu(\mu_0)\Delta\mathbf{u}(\boldsymbol{\mu}) + \nabla p(\boldsymbol{\mu}) &= 0 && \text{in } \Omega, \\
\operatorname{div} u(\boldsymbol{\mu}) &= 0 && \text{in } \Omega, \\
u(\boldsymbol{\mu}) &= \mu_1 && \text{in } \Gamma_{\text{IN}}, \\
u(\boldsymbol{\mu}) &= 0 && \text{in } \Gamma_{\text{S}} \cup \Gamma_{\text{N}} \cup \Gamma_{\text{CYL}}, \\
p(\boldsymbol{\mu}) &= 0 && \text{in } \Gamma_{\text{OUT}}.
\end{aligned} \tag{5.1}$$

As we can see from Eq 5.1, we consider two physical parameters $\boldsymbol{\mu} = (\mu_0, \mu_1) \in \mathbf{P} \subset \mathbb{R}^2$ controlling the viscosity μ of the fluid and the velocity of the fluid at the inlet boundary Γ_{IN}. The 2-dimensional domain Ω (sketched in Figure 1) is a rectangle with a circular hole, on which we impose on the physical walls ($\Gamma_{\text{S}} \cup \Gamma_{\text{N}} \cup \Gamma_{\text{CYL}}$) no-slip boundary condition, on the inlet the parametric flow velocity μ_1 and on the outlet a null pressure.
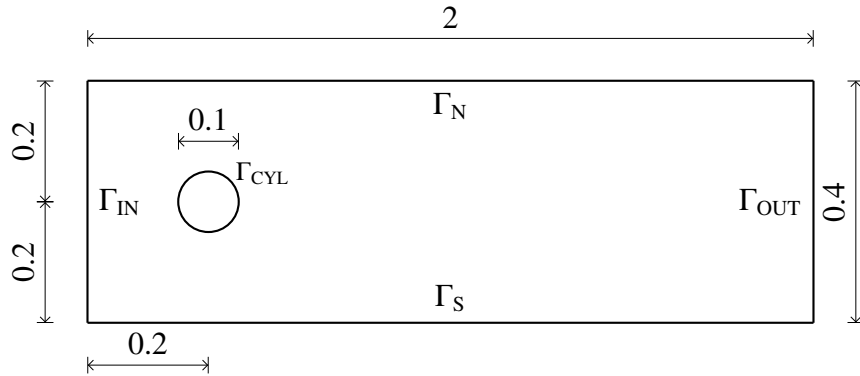
Figure 1. The domain Ω for the parametric Stokes flow simulation.

Starting from Eq 5.1, we can obtain the weak formulation of the Stokes equations by multiplying these equations to the test functions $v \in [H_0^1(\Omega)]^2 = V$ and $q \in L^2(\Omega) = Q$:

$$
\begin{aligned}
a(u(\mu), v; \mu) + b(v, p(\mu); \mu) &= 0 \qquad \forall v \in V, \\
b(u(\mu), q; \mu) &= 0 \qquad \forall q \in Q.
\end{aligned}
\tag{5.2}
$$

where $a(\cdot, \cdot; \mu)$ and $b(\cdot, \cdot; \mu)$ are the parametric bilinear forms.

To solve the Stokes problem described above, we adopt a finite element (FE) approach. We define the finite dimensional spaces $\mathbb{V}_h = (P_k(\mathcal{T}))^2 \cap V$ and $\mathbb{Q}_h = P_l(\mathcal{T}) \cap Q$, where $P_i(\mathcal{T}) = \{v \in C(\Omega) : v|_\tau \in P_i, \forall \tau \in \mathcal{T}\}$ for $k > 1$ indicates the piecewise polynomial space defined on the triangulation $\mathcal{T}$ of the domain Ω. For this work, we have chosen a second order polynomial for the velocity space and a first order one for the pressure space, i.e., (P_2, P_1). The use of the so-called Hood-Taylor scheme [31] is sufficient for ensuring the inf-sup condition; for the mathematical proof of the stability of such scheme, we refer to [3]. Moreover, the Reynolds number is very low (Re < 1) for all the combinations of parameters, making possible to avoid additional stabilization to numerically solve the problem.

The discretized parametric Stokes problem reads as follow:

$$
\begin{aligned}
a(u_h(\mu), v_h; \mu) + b(v_h, p_h(\mu); \mu) &= 0 \qquad \forall v_h \in \mathbb{V}_h, \\
b(u_h(\mu), q_h; \mu) &= 0 \qquad \forall q_h \in \mathbb{Q}_h,
\end{aligned}
\tag{5.3}
$$

where $u_h(\mu) \in \mathbb{V}_h$ and $p_h(\mu) \in \mathbb{Q}_h$ are the finite dimensional unknowns we want to compute. The discretized problem has been solved using the FEniCS framework [1].

We initially create the high-fidelity database by computing the pressure and velocity fields for 50 samples, randomly generated in the parameter space $\mathbf{P} = [.1, 10] \times [1e{-}4, 1e{-}6]$. We then apply the described method to these snapshots, using initially the POD algorithm to reduce the dimensionality of the solutions, aka extract the modal coefficients.

In this case, we use 12 modes for the construction of the reduced space and apply a GP — we repeat that the used kernel is the squared exponential one — to the distribution of the modal coefficients. To test the accuracy, we create a test dataset containing 20 random samples selected in the parameter space, that of course are different to the input database. For all these test parameters, we approximate the reduced solutions and compute the realtive error between the reduced test solutions and the truth ones. Figure 2 reports the average relative error for all the analized fields in relation to the number of input snapshots, starting from only 2 snapshots. We specify that, using less than 12 snapshots, the

number of POD modes is limited to the number of snapshots. As we can see, the error results very small with even few snapshots, especially for both the velocity components that are beyond the 5% with only 8 snapshots. The trend for the pressure error is less steep (we need 16 snapshots to reach an error of 5%), but it is able to reach at the end a precision of about 2.5%. A graphical visualization of the test solutions obtained using only 3 input samples is provided in Figure 3. We underline the error becomes pretty much constant using many snapshots, due to the Bayesian online phase, making the usage of GP during the online stage profitable especially in the cases where the offline stage has a limited computational budget.
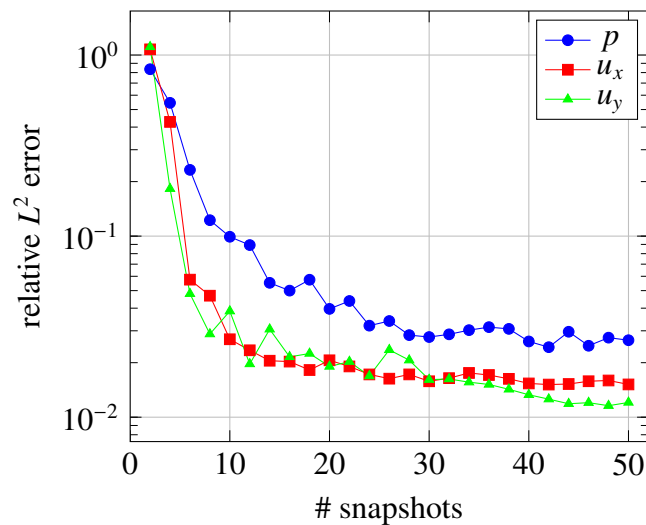
Figure 2. Relative average error between the truth solutions and the solutions obtained with the POD-GPR framework, evaluated in 20 test parameters, varying the number of snapshots.
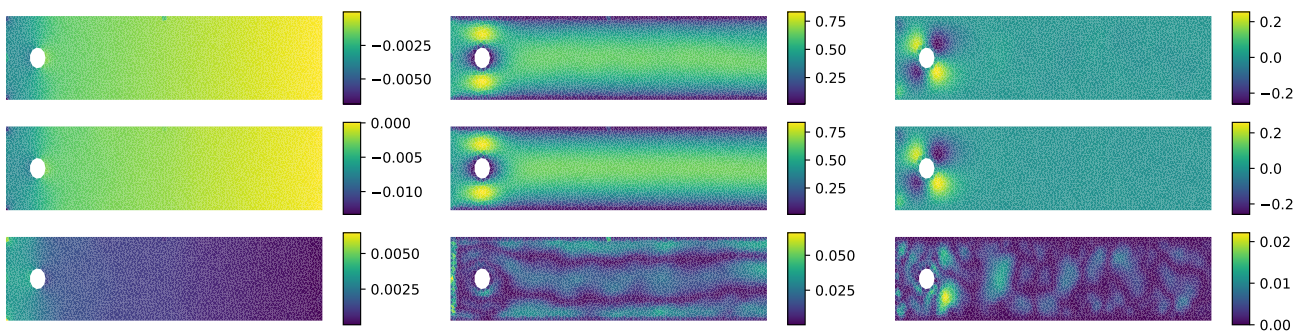

Figure 3. Graphical visualization of the reconstructed fields for a test parameter using only 3 input samples. The plots are organized as following: the three columns represent, from left to right, pressure, velocity among *x* direction and velocity among *y* direction. The first row shows the truth solutions, the second row the solutions approximated using the POD-GPR method and the third one represents the absolute error between them.

5.2. *Multiphase turbulent Navier-Stokes flow in an industrial parametric domain*

The second numerical experiment we set up to test the POD-GPR framework is the simulation of the flow around the hull of a cruise ship. The reduction is then employed inside a shape optimization pipeline, enabling the use of a global optimization procedure on the design space.

We now proceed to discuss the main ingredients of the optimization pipeline in order to understand the role of the POD-GPR reduction. We avoid going into many technical details and refer the reader to [6, 20] for a more extensive presentation of all the optimization pipeline.

The first step of the optimization pipeline consists of the generation of the parametric design space, where each possible shape is associated to a particular value of a pre-defined parameter. This is done by considering an initial undeformed configuration and applying to it a technique called Free Form Deformation (FFD) (for more details see [17, 26, 28]). FFD is a deformation technique in which the object to be deformed is initially put inside a lattice of points; then some of these points are moved, in a parametric way, and the space enclosed is deformed smoothly according to these motions.

In our case, we define the lattice shown in Figure 4. This lattice is enclosing the bottom-frontal part of the boat, being the part we want to deform, and is composed of a total of around 500 points, of which only a small part will be displaced.
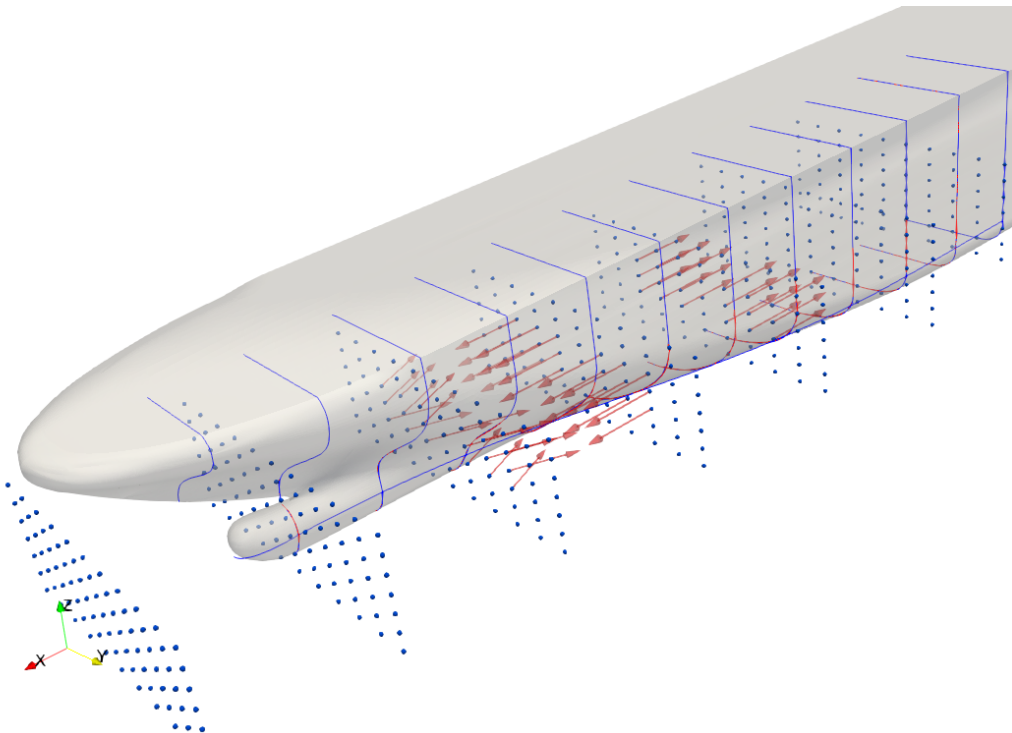

Figure 4. View of the the lattice of points over the undeformed hull: the blue dots are the FFD control points, while the red arrows represent the displacements.

In particular we define a set of movements connected to 6 independent parameters. The way in which these parameters are defined and the ranges they take values from are related to the kind of shapes we want to consider and the various constraints those shapes are subjected to. For example, the most simple and straightforward constraint we imposed is that of volume, i.e., we impose that the hulls

must not have a total volume that reaches too small values. Another constraint we impose is that the deformations applied on the hull needs to connect continuously and smoothly to the undeformed parts of the boat. This latter constraint can be eimposed by keeping fixed some points in the lattice.

The FFD step is performed using PyGEM, an open-source Python library (see [23]).

The next step consists of the evaluation of the performance of the newly generated hull. This is done via numerical simulation of the flow of water and air around it. In particular, we consider a 3D biphase (water and air) incompressible flow. The physical model is based on the well known Incompressible Navier-Stokes equations, with modellation of turbulence based on a RANS approach (for more details see [5]), Finite Volume discretization (see [18]) and using the Volume of Fluid (VOF) methodology (see [16]) to take into account the biphase nature of the fluid.

More specifically, the equations considered are the following:

$$\begin{cases} \frac{\partial(\rho u)}{\partial t} + \nabla \cdot (\rho u \otimes u) + \nabla p - \rho g - \nabla \cdot \nu \nabla u - \nabla \cdot R = 0, \\ \nabla \cdot u = 0, \\ \frac{\partial \alpha}{\partial t} + \nabla \cdot (u\alpha) = 0, \end{cases} \tag{5.4}$$

where u, p denote the fluid velocity and pressure fields, ρ, ν and g denote respectively density, dynamic viscosity and gravity acceleration, R is the Reynolds stress tensor and α is the fraction of water.

The first equation in Eq (5.4) expresses the momentum balance in the infinitesimal volume and is essentially a reformulation of Newton'st 2nd law of dynamics expressing the Eulerian fluid's acceleration ($\frac{\partial u}{\partial t}$) as a result of convection (($u \cdot \nabla)u$), a gradient of pressure ($\frac{1}{\rho}\nabla p$) and diffusion ($\nabla \cdot \nu \nabla u$). The second equation instead expresses the mass conservation, emerging as solenoidality of the velocity field because of the incompressibility hypothesis.

We note that both the velocity and pressure fields in this equations represent only the mean part, decoupled from the fluctuating part using a RANS approach. All the effects of the fluctuating part are contained in the Reynolds stress tensor R, which will be in then modeled as a function of the mean velocity field u in order to close the equations.

The third equation is coming from the VOF modelling of the biphase flow.

Volume of fluid is a free-surface modeling technique that allows to describe a multiphase fluid composed of two incompressible, isothermal immiscible fluids (water and air, in our case). For a more in-depth discussion on VOF, we refer to [16].

This method is based on a phase-fraction technique: a scalar variable, denoted by α, is added to N-S equations, representing the fraction of water contained in the infinitesimal volume (or in the finite volume, when we will discretize the equations). This variable belongs to the interval $[0, 1]$, where $\alpha = 1$ represents a point in which water is present, $\alpha = 0$ a point in which air is present and $\alpha \in (0, 1)$ represents interface points. By its nature, α is a discontinuous variable, and so the discretization method must ensure that the interface is captured a small number of cells.

In addition to the momentum and mass balance equations, we have two equations where the density ρ and the kinematic viscosity ν are defined using an algebraic formula expressing them as a convex combination of the corresponding water and air properties:

$$\rho = \alpha \rho_W + (1 - \alpha)\rho_A, \tag{5.5}$$

$$\nu = \alpha \nu_W + (1 - \alpha)\nu_A. \tag{5.6}$$

Finally, the third equation in the system (5.4) is the equation required to close the system after the addition of the variable α. In fact, this equation is simply a transport equation for the fraction of fluid in which only the convective term is considered.

As for the discretization and solution of the above equations, we used the implementation contained in the open-source CFD software OpenFOAM (see [19]).

The POD-GPR framework described above is then employed in order to reduce the computational time required for the resolution of the numerical problem associated to a new shape. Going into the details of the reduction, we consider as a snapshot, and hence as a solution, not the whole volumetric flow field but the the field of total resistance (viscous+pressure) over the hull.

The first step consists of the collection of the snapshots matrix, i.e., a set of solutions of the Full Order Model for selected parameter values. In particular we sampled the parameters space described above considering the vertices ($2^6 = 64$ points) and other 36 random points inside the domain, for a total of 100 snapsohts. Other 20 random snapshots have been generated for purpose of testing.

The extraction of the modes and approximation step have then been conducted using EZyRB [8], an open-source Python library for data-driven model order reduction. We now show some graphs reporting the relative L^2 error between FOM and ROM:

$$\left(\frac{\int_A (u^N - u_N^N)^2 dA}{\int_A u_N^{N2} dA}\right)^{\frac{1}{2}}. \tag{5.7}$$

In particular, in Figure 5 (left), the error is expressed as a function of the number of modes used and as a function of the number of snapshots used. In both the figures we report in different colors the different techniques used in the approximation of the modal coefficients. We consider the GPR approach, the classical n-dimensional linear interpolation and two interpolation techniques based on radial basis functions (RBF), implemented using the Python package Scipy (see [34]). We remark we use the classical RBF interpolation method [4]. The former two methods differ one from the other for the value of one particular parameter that defines the method, called "smoothness", which quantifies how close the approximation technique is to interpolation, where a value of 0 correspond to a plain interpolation while higher values correspond to more smooth approximations. In particular, we considered values of the smoothness equal to 0 and 0.01.

We can see from Figure 5 that, using the GPR for the regression of the modal coefficients, we are able to use a very low number of snapshots obtaining an error which is comparable to the performances of different regression techniques for higher number of snapshots. This is really interesting since, as argued in the previous section, the offline phase, i.e. the generation of the snapshots, is indeed the most expensive from the computational point of view. However, when the number of snapshots increases, the performances of GPR and the other methods tend to converge to a same value.

The final step of the shape optimization pipeline consists on the exploration of the design space in order to find the optimum. To perform this, we used a evolutionary optimization algorithm, implemented inside DEAP, an open-source Python library for evolutionary optimization (see [10]).

From the point of view of the speed-up in the computational time obtained with the use of the POD-GPR framework, we mention that while in order to run the above mentioned optimization procedure using the FOM we would have needed around 140 weeks of CPU time, the use of the ROM approach restricted this time to around 5 weeks. Of these 5 weeks, the biggest partition is associated to the offline phase, i.e., the generation of the 100 training simulations, being the time associated to the evaluation of

the online phase almost negligible. It is then clear that, in light of the considerations made regarding the error as a function of the number of snapshots for the POD-GPR framework, we could have augmented this speed-up of at least a factor 10 by generating fewer snapshots.
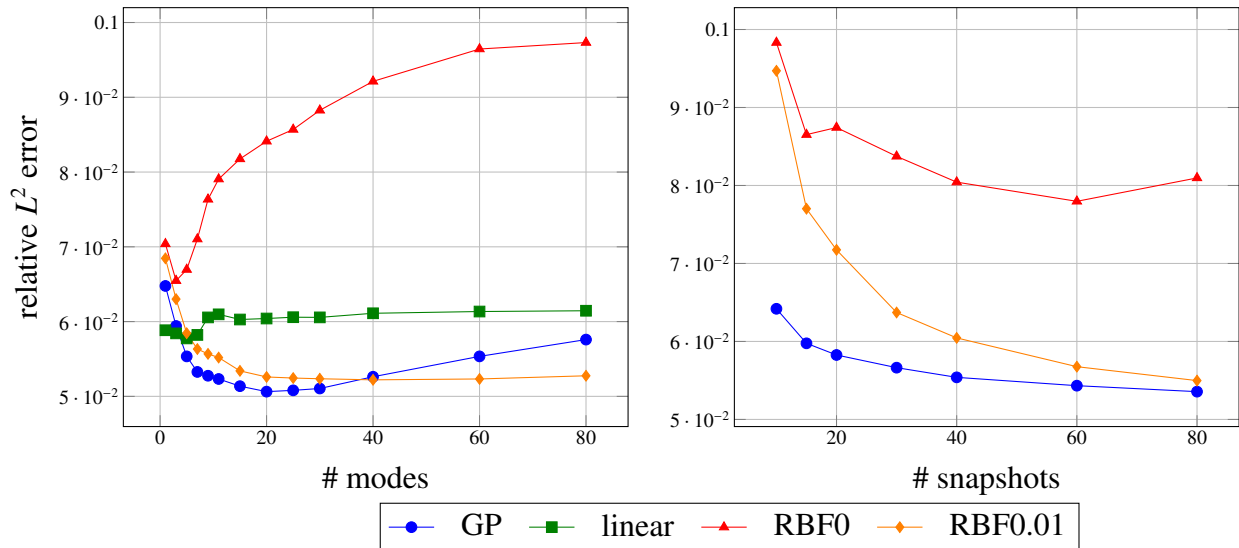

Figure 5. Average relative L^2 error between the FOM and the ROM as a function of the number of modes, keeping fixed the number of snapshots to 80, (left) and the number of snapshots used, keeping fixed the number of modes to 20 (right).

Finally we report, in Figure 6 the optimal shape found, which correspond to a reduction of drag of 3% from the initial hull. The figure represents sections of the hull on the longitudinal and transversal plane, in blue for the undeformed hull and red for the optimal deformed hull.
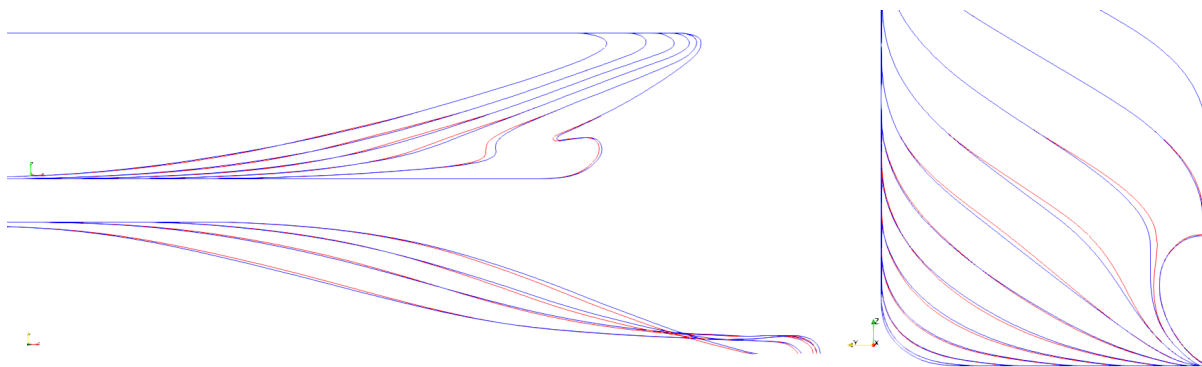

Figure 6. View of the x (right), y (upper left) and z (lower left) sections for the optimal hull shape; in blue the undeformed hull, in red the deformed hull.

6. Conclusions and perspectives

In this contribution, we applied the POD-GPR framework to two completely different parametric problems. To emphasize the equation-free nature of the algorithm, we demonstrate the applicability

to a Stokes problem discretized using finite element method, then to a turbulent multiphase industrial problem discretized using finite volume method. In both cases, it emerges that the GPR method for the online prediction of the modal coefficients (for untested parameters) is able to make a prediction with a relative small error by using very few input snapshots. The advantage of such method is then maximazed when the offline phase results quite expensive due to the high-fidelity model.

The adoption of this method open several possibilities for future developments, by using the GPR not only for the prediction of new modal coefficients, but also for the quantification of the error, or for a greedy approach for new snapshots selection by exploiting the variance of the Gaussian distribution. Additionaly, the use of different kernels can be investigated.

Acknowledgements

Conflict of interest

The authors declare no conflict of interest.

References

1. M. S. Alnæs, J. Blechta, J. Hake, A. Johansson, B. Kehlet, A. Logg, et al., The FEniCS project version 1.5, *Archive of Numerical Software*, **3** (2015), 9–23.

2. F. Ballarin, A. Manzoni, A. Quarteroni, G. Rozza, Supremizer stabilization of POD–Galerkin approximation of parametrized steady incompressible Navier–Stokes equations, *Int. J. Numer. Meth. Eng.*, **102** (2015), 1136–1161.

3. D. Boffi, F. Brezzi, M. Fortin, Finite elements for the stokes problem, In: *Mixed finite elements, compatibility conditions, and applications*, Springer, 2008, 45–100.

4. M. D. Buhmann, *Radial basis functions: theory and implementations*, Cambridge University Press, 2003.

5. P. Davidson, *Turbulence: an introduction for scientists and engineers*, Oxford University Press, 2015.

6. N. Demo, G. Ortali, G. Gustin, G. Rozza, G. Lavini, An efficient computational framework for naval shape design and optimization problems by means of data-driven reduced order modeling techniques. *Boll. Unione Mat. Ital.*, **14** (2021), 211–230.

7. N. Demo, M. Tezzele, G. Gustin, G. Lavini, G. Rozza, Shape optimization by means of proper orthogonal decomposition and dynamic mode decomposition, In: *Technology and science for the ships of the future: proceedings of NAV 2018: 19th international conference on ship & maritime research*, IOS Press, 2018, 212–219.

8. N. Demo, M. Tezzele, G. Rozza, EZyRB: Easy reduced basis method, *JOSS*, **3** (2018), 661.

9. N. Demo, M. Tezzele, G. Rozza, A non-intrusive approach for the reconstruction of POD modal coefficients through active subspaces, *CR. Mécanique*, **347** (2019), 873–881.

10. F. A. Fortin, F. M. De Rainville, M. A. Gardner, M. Parizeau, C. Gagné, DEAP: Evolutionary algorithms made easy, *J. Mach. Learn. Res.*, **13** (2012), 2171–2175.

11. GPy, GPy: A Gaussian process framework in Python, 2012. Available from: http://github.com/SheffieldML/GPy.

12. M. Guo, J. S. Hesthaven, Reduced order modeling for nonlinear structural analysis using Gaussian process regression, *Comput. Method. Appl. M.*, **341** (2018), 807–826.

13. J. S. Hesthaven, G. Rozza, B. Stamm, *Certified reduced basis methods for parametrized partial differential equations*, 1 Eds., Switzerland: Springer, 2016.

14. J. S. Hesthaven, S. Ubbiali, Non-intrusive reduced order modeling of nonlinear problems using neural networks, *J. Comput. Phys.*, **363** (2018), 55–78.

15. S. Hijazi, G. Stabile, A. Mola, G. Rozza, Data-driven pod-galerkin reduced order model for turbulent flows, *J. Comput. Phys.*, **416** (2020), 109513.

16. C. Hirt, B. Nichols, Volume of fluid (VOF) method for the dynamics of free boundaries, *J. Comput. Phys.*, **39** (1981), 201–225.

17. A. Koshakji, A. Quarteroni, G. Rozza, Free form deformation techniques applied to 3D shape optimization problems, *CAIM*, **4** (2013), 1–26.

18. F. Moukalled, L. Mangani, M. Darwish, *The finite volume method in computational fluid dynamics: an advanced introduction with OpenFOAM and Matlab*, Cham: Springer, 2015.

19. OpenCFD, OpenFOAM - The Open Source CFD Toolbox - User's Guide, 2018. Available from: https://www.openfoam.com/documentation/user-guide.

20. G. Ortali, A data-driven reduced order optimization approach for Cruise ship design, Master's thesis, Politecnico di Torino, 2019.

21. F. Pichi, F. Ballarin, G. Rozza, J. S. Hesthaven, Artificial neural network for bifurcating phenomena modelled by nonlinear parametrized PDEs, *PAMM*, **20** (2021), e202000350.

22. F. Pichi, G. Rozza, Reduced basis approaches for parametrized bifurcation problems held by nonlinear von kármán equations, *J. Sci. Comput.*, **81** (2019), 112–135.

23. PyGeM, PyGeM: Python geometrical morphing. Available from: https://github.com/mathLab/PyGeM.

24. J. Quiñonero-Candela, C. E. Rasmussen, A unifying view of sparse approximate gaussian process regression, *J. Mach. Learn. Res.*, **6** (2005), 1939–1959.

25. G. Rozza, M. H. Malik, N. Demo, M. Tezzele, M. Girfoglio, G. Stabile, et al., Advances in reduced order methods for parametric industrial problems in computational fluid dynamics, In: *Proceedings of the ECCOMAS Congress 2018*, 2018.

26. F. Salmoiraghi, F. Ballarin, L. Heltai, G. Rozza, Isogeometric analysis-based reduced order modelling for incompressible linear viscous flows in parametrized shapes, *Adv. Model. and Simul. in Eng. Sci.*, **3** (2016), 21.

27. F. Salmoiraghi, A. Scardigli, H. Telib, G. Rozza, Free-form deformation, mesh morphing and reduced-order methods: enablers for efficient aerodynamic shape optimisation, *Int. J. Comput. Fluid Dyn.*, **32** (2018), 233–247.

28. T. W. Sederberg, S. R. Parry, Free-form deformation of solid geometric models, In: *ACM SIGGRAPH Computer Graphics*, 1986, 151–160.

29. G. Stabile, G. Rozza, Finite volume POD-Galerkin stabilised reduced order methods for the parametrised incompressible Navier–Stokes equations, *Comput. Fluids*, **173** (2018), 273–284.

30. G. Stabile, M. Zancanaro, G. Rozza, Efficient Geometrical parametrization for finite-volume based reduced order methods. *Int. J. Numer. Meth. Eng.*, **121** (2020), 2655–2682.

31. C. Taylor, P. Hood, A numerical solution of the navier-stokes equations using the finite element technique, *Comput. Fluids*, **1** (1973), 73–100.

32. M. Tezzele, N. Demo, G. Rozza, Shape optimization through proper orthogonal decomposition with interpolation and dynamic mode decomposition enhanced by active subspaces, In: *Proceedings of MARINE 2019: VIII International Conference on Computational Methods in Marine Engineering*, 2019, 122–133.

33. M. Tezzele, N. Demo, G. Stabile, A. Mola, G. Rozza, Enhancing cfd predictions in shape design problems by model and parameter space reduction, *Adv. Model. and Simul. in Eng. Sci.*, **7** (2020), 40.

34. P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, et al., SciPy 1.0: Fundamental algorithms for scientific computing in Python, *Nat. Methods*, **17** (2020), 261–272.

35. S. Volkwein, *Proper orthogonal decomposition: theory and reduced-order modelling*, 2012.

36. Q. Wang, J. S. Hesthaven, D. Ray, Non-intrusive reduced order modeling of unsteady flows using artificial neural networks with application to a combustion problem, *J. Comput. Phys.*, **384** (2019), 289–307.