An optimisation–based domain–decomposition reduced order model for parameter-dependent non–stationary fluid dynamics problems

Ivan Prusak*[a], Davide Torlo[†a], Monica Nonino[‡b], and Gianluigi Rozza[§a]

[a]Mathematics Area, mathLab, SISSA, 34136 Trieste, Italy
[b]Fakultät für Mathematik, Universität Wien, 1090 Wien, Austria

Abstract

In this work, we address parametric non–stationary fluid dynamics problems within a model order reduction setting based on domain decomposition. Starting from the optimisation–based domain decomposition approach, we derive an optimal control problem, for which we present a convergence analysis in the case of non–stationary incompressible Navier–Stokes equations. We discretize the problem with the finite element method and we compare different model order reduction techniques: POD–Galerkin and a non–intrusive neural network procedures. We show that the classical POD–Galerkin is more robust and accurate also in transient areas, while the neural network can obtain simulations very quickly though being less precise in the presence of discontinuities in time or parameter domain. We test the proposed methodologies on two fluid dynamics benchmarks with physical parameters and time dependency: the non–stationary backward–facing step and lid–driven cavity flow.

1 Introduction

With the increase in the potential of high–performance computing in the last years, there is a significant demand for numerical methods and approximation techniques that can perform real–time simulations of Partial Differential Equation (PDEs). The applications vary from naval, aeronautical and biomedical engineering. There exist many techniques to achieve such a goal, including reduced–order modelling [28] and domain–decomposition (DD) methods [46].

---

*iprusak@sissa.it

†dtorlo@sissa.it

‡monica.nonino@univie.ac.at

§grozza@sissa.it

The DD methodology is a highly efficient tool in the framework of PDEs. Any DD algorithm is constructed by an effective splitting of the domain of interest into different subdomains (overlapping or not), and the original problem is then restricted to each of these subdomains with some coupling conditions on the intersections of the subdomains. The coupling conditions may be very different, they depend on the physical meaning of the problem at hand, and they are required to provide a certain degree of continuity among these subdomains (see, for example, [45, 46]). These methods are extremely important for multi–physics problems when efficient subcomponent numerical codes are already available, or when we do not have direct access to the numerical algorithms for some parts of the systems; see, for instance, [18, 22, 30, 32, 33, 34].

Model–order reduction methods are another set of methods that are extremely useful when dealing with real–time simulations or multi–query tasks. These methods are successfully employed in the settings of non–stationary and/or parameter–dependent PDEs. Reduced–order models (ROMs) are extremely effective due to the splitting of the computational effort into two stages: the offline stage, which contains the most expensive part of the computations, and the online stage, which allows performing fast computational queries using structures that are pre–computed in the offline stage; for more details, see [28]. ROMs have been successfully applied in different fields such as fluid dynamics [4, 14, 15, 17, 35, 49, 52, 53, 55, 57, 59, 61], structural mechanics [9, 10, 26, 50, 60, 62] and fluid–structure interaction problems [6, 9, 39, 38]. Lately, there have also been great advances in reduced–order modelling for optimal–control problems, [42, 54, 56].

The novelty of this paper lies in the study of an optimization-based approach for a domain-decomposition model addressing unsteady parametric fluid dynamics problems. We employ both intrusive and non-intrusive model-order reduction (MOR) techniques to achieve this, providing a comprehensive comparison between the two methodologies. Utilizing domain-decomposition methods, particularly an optimization approach ensuring coupling of interface conditions between subdomains as presented in [25, 23, 43], we develop various reduced-order models (ROMs). These include classical POD-Galerkin projection-based intrusive methods [28, 53] and neural network (NN) based non-intrusive ROMs [29, 48, 40, 41, 51]. Moreover, we perform a convergence analysis for the nonstationary fluid dynamics problems based on the *a priori* estimates by extending the results presented in [25] for the stationary case.

This work is structured as follows. In Section 2, we introduce the monolithic fluid dynamics problem and its time–discretisation scheme with the further derivation of the optimisation–based domain–decomposition formulation at each time step. In Section 3, we derive *a priori* estimates for the solutions to Navier–Stokes equations which are then used to prove the existence and uniqueness of the minimiser to the optimal–control problem derived in the previous section. Furthermore, in Section 3.5 we provide a closed–form expression for the gradient of the objective functional which allows for the use of gradient–based optimisation algorithms to decouple the subdomain solves. Section 4 contains the Finite Element discretisation of the problem of interest and the corresponding finite–dimensional high–fidelity optimisation problem. Section 5 deals with two ROM techniques: an intrusive Galerkin projection and a neural network (NN) algorithm, both based on a Proper Orthogonal Decomposition (POD) methodology. In Section 6, we show some numerical results for two toy problems: the backward–facing

2

step and the lid–driven cavity flow. Conclusions will follow in Section 7.

2 Problem formulation

In this section, starting with a monolithic formulation of the time–dependent incompressible Navier–Stokes equations, we first introduce a time discretisation on the continuous level employing the implicit Euler time–stepping scheme. Then, we will describe a two–domain optimisation-based domain–decomposition formulation at each time step in variational form in the end. Here and in the next few sections, the analysis is valid for any value of the physical parameter, so for the sake of simplicity, we postpone mentioning the parameter dependence of the problem until Section 5.

2.1 Monolithic formulation

Let Ω be a physical domain of interest: we assume Ω to be an open subset of $\mathbb{R}^2$ and Γ to be the boundary of Ω. We also consider a finite time interval $[0, T]$ with $T > 0$. Let $f : \Omega \times [0, T] \to \mathbb{R}^2$ be the forcing term, ν the kinematic viscosity, u_D a given Dirichlet datum and u_0 a given initial condition. The problem reads as follows: find the velocity field $u : \Omega \times [0, T] \to \mathbb{R}^2$ and the pressure $p : \Omega \times [0, T] \to \mathbb{R}$ s.t.

$$\frac{\partial u}{\partial t} - \nu \Delta u + (u \cdot \nabla) u + \nabla p = f \quad \text{in} \quad \Omega \times (0, T], \tag{1a}$$

$$-\mathrm{div} u = 0 \quad \text{in} \quad \Omega \times (0, T], \tag{1b}$$

$$u = u_D \quad \text{on} \quad \Gamma_D \times [0, T], \tag{1c}$$

$$\nu \frac{\partial u}{\partial \mathbf{n}} - p\mathbf{n} = 0 \quad \text{on} \quad \Gamma_N \times [0, T], \tag{1d}$$

$$u(t = 0) = u_0 \quad \text{in} \quad \Omega, \tag{1e}$$

where Γ_D and Γ_N are disjoint subsets of Γ (as it is shown in Figure 1a) and $\mathbf{n}$ is an outward unit normal vector to Γ_N.

2.2 Time discretisation

We will start with time discretisation of problem (1). Let $\Delta t > 0$, we assume the following time interval partition: $0 = t_0 < t_1 < \dots < t_M = T$, where $t_n = n\Delta t$ for $n = 0, \dots, M$. We employ the implicit Euler scheme for the incompressible Navier–Stokes equation which reads as follows: for $n \geq 1$ find $u^n : \Omega \to \mathbb{R}^2$, $p^n : \Omega \to \mathbb{R}$ s.t.

$$\frac{u^n - u^{n-1}}{\Delta t} - \nu \Delta u^n + (u^n \cdot \nabla) u^n + \nabla p^n = f^n \quad \text{in} \quad \Omega, \tag{2a}$$

$$-\mathrm{div} u^n = 0 \quad \text{in} \quad \Omega, \tag{2b}$$

$$u^n = u_D^n \quad \text{on} \quad \Gamma_D, \tag{2c}$$

$$\nu \frac{\partial u^n}{\partial \mathbf{n}} - p^n \mathbf{n} = 0 \quad \text{on} \quad \Gamma_N, \tag{2d}$$

3

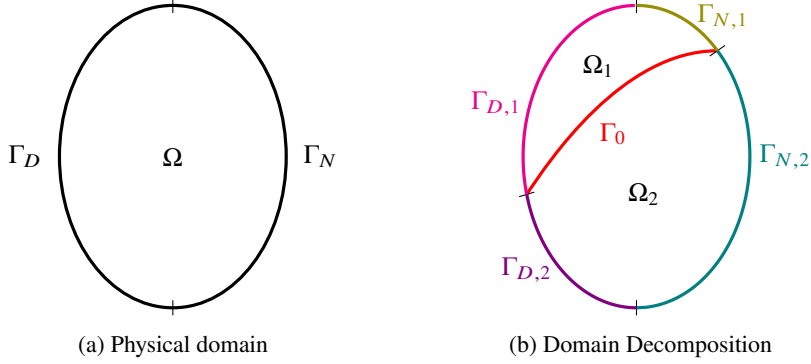(a) Physical domain   (b) Domain Decomposition

Figure 1: Domain and boundaries

and for $n = 0$

$$u^0 = u_0 \quad \text{in} \quad \Omega. \tag{2e}$$

Here we adopted the following notations: $f^n(\cdot) = f(\cdot, t_n)$ and $u_D^n(\cdot) = u_D(\cdot, t_n)$.

2.3 Domain Decomposition (DD) formulation

As mentioned in the introduction, we resort to the optimisation–based DD framework in [43]. Let Ω_i, $i = 1, 2$, be open subsets of Ω, such that $\overline{\Omega} = \overline{\Omega_1} \cup \overline{\Omega_2}$, $\Omega_1 \cap \Omega_2 = \emptyset$. Denote $\Gamma_i := \partial\Omega_i \cap \Gamma$, $i = 1, 2$, and $\Gamma_0 := \overline{\Omega_1} \cap \overline{\Omega_2}$. In the same way, we define the corresponding boundary subsets $\Gamma_{i,D}$ and $\Gamma_{i,N}$, $i = 1, 2$, see Figure 1b. Having at hand the time discretisation (2), we can cast the DD problem in the optimal control setting with the weak formulation of the DD state equations. For this purpose, we define the following variational spaces

- $V_i := \{u \in H^1(\Omega_i; \mathbb{R}^2)\}$,

- $V_{i,0} := \{u \in H^1(\Omega_i; \mathbb{R}^2) : u|_{\Gamma_{i,D}} = 0\}$,

- $Q_i := \{p \in L^2(\Omega_i; \mathbb{R})\}$,

together with the following bilinear and trilinear forms

- $m_i : V_i \times V_{i,0} \to \mathbb{R}, \quad m_i(u_i, v_i) := (u_i, v_i)_{\Omega_i}$,

- $a_i : V_i \times V_{i,0} \to \mathbb{R}, \quad a_i(u_i, v_i) := \nu(\nabla u_i, \nabla v_i)_{\Omega_i}$,

- $b_i : V_i \times Q_i \to \mathbb{R}, \quad b_i(v_i, q_i) := -(\text{div} v_i, q_i)_{\Omega_i}$,

- $c_i : V_i \times V_i \times V_{i,0} \to \mathbb{R}, \quad c_i(u_i, w_i, v_i) := ((u_i \cdot \nabla)w_i, v_i)_{\Omega_i}$,

for $i = 1, 2$. In the definitions above, $(\cdot, \cdot)_\omega$ indicates the $L^2(\omega)$ inner product. The spaces V_i are endowed with the $H^1(\Omega_i)$–norm for $i = 1, 2$, the spaces $V_{i,0}$ with the $H_0^1(\Omega_i)$–norm and the spaces Q_i with the $L^2(\Omega_i)$–norm for $i = 1, 2$.

4

The DD formulation then reads as follows. For $n \geq 1$ solve the following optimal control problem:

minimise over $g \in L^2(\Gamma_0)$ *the functional*

$$\mathcal{J}_\gamma(u_1^n, u_2^n; g) := \frac{1}{2} \int_{\Gamma_0} |u_1^n - u_2^n|^2 \, d\Gamma + \frac{\gamma}{2} \int_{\Gamma_0} |g|^2 \, d\Gamma, \tag{3}$$

subject to the variational problem:

$$\begin{cases} \frac{1}{\Delta t} m_i \left(u_i^n - u_i^{n-1}, v_i \right) + a_i(u_i^n, v_i) + c_i(u_i^n, u_i^n, v_i) + b_i(v_i, p_i^n) \\ \qquad\qquad = (f_i^n, v_i)_{\Omega_i} + \left((-1)^{i+1} g, v_i \right)_{\Gamma_0} & \forall v_i \in V_{i,0}, \\ b_i(u_i^n, q_i) = 0 & \forall q_i \in Q_i, \\ u_i^n = u_{i,D}^n & \text{on } \Gamma_{i,D}. \end{cases} \tag{4}$$

The presence of the regularisation term $\frac{\gamma}{2} \int_{\Gamma_0} |g|^2 \, d\Gamma$ with $\gamma > 0$ in the definition of the objective functional (3) is essential for the well–posedness analysis of the optimal control problem (3)– (4) carried out in the next section. Nevertheless, our goal is to find a DD solution which is as close as possible to the monolithic one; for this reason we will provide the analysis when the regularisation parameter $\gamma \to 0$.

3 Analysis of the optimal control problem

In this section, we will give an overview of the existence of local minima of the optimal–control problem described above. It will rely on the a priori estimates for the solutions to the Navier–Stokes equations. As is evidenced, for instance, in [47], due to the presence of the Neumann boundary condition, the analysis of the state problem is not possible in the general case, so we will modify the problem in the framework where the problem is well posed and give some indication about the original problem later in the section. Finally, in the end of the section, we will list the objective functional gradient of the original DD optimal control problem.

3.1 A modified Navier–Stokes problem

First, without loss of generality, we assume that the Dirichlet data $u_{i,D}$ is homogeneous. Otherwise, it is possible to recast the inhomogeneous problem into a homogeneous one with various techniques, e.g., lifting functions as described in Appendix A. As mentioned in the preface to this section, it is hard to prove the well–posedness of the solution to the Navier–Stokes equation in the form (4). The main problem arises from the nonlinear term $c_i(u_i^n, u_i^n, v_i) = \left((u_i^n \cdot \nabla) u_i^n, v_i \right)_{\Omega_i^n}$. Indeed, by integration by parts

and the incompressibility conditions in (4), we can see

$$
\begin{aligned}
c_i(u_i^n, w_i, v_i) \quad &= \int_{\Omega_i} (u_i^n \cdot \nabla) w_i \cdot v_i d\Omega = \int_{\partial\Omega_i} (w_i \cdot v_i)(u_i^n \cdot \mathbf{n}_i) d\Gamma \\
&\quad - \int_{\Omega_i} (w_i \cdot v_i) \operatorname{div} u_i^n d\Omega - \int_{\Omega_i} (u_i^n \cdot \nabla) v_i \cdot w_i d\Omega \\
&= \int_{\partial\Omega_i} (w_i \cdot v_i)(u_i^n \cdot \mathbf{n}_i) d\Gamma - \int_{\Omega_i} (u_i^n \cdot \nabla) v_i \cdot w_i d\Omega \\
&= \int_{\Gamma_{i,N} \cup \Gamma_0} (w_i \cdot v_i)(u_i^n \cdot \mathbf{n}_i) d\Gamma - c_i(u_i^n, v_i, w_i),
\end{aligned}
$$

which leads to the following expression

$$
c_i(u_i^n, u_i^n, u_i^n) = \frac{1}{2} \int_{\Gamma_{i,N} \cup \Gamma_0} |u_i^n|^2 (u_i^n \cdot \mathbf{n}_i) d\Gamma. \tag{5}
$$

The issue here is that this boundary term is of third order and has an unknown sign since the solution on the interface Γ_0 is undetermined. This complicates the analysis, which relies on the *a priori* bounds of the weak solutions. On the other hand, it gives us an idea of how to redefine the problem at hand in order to obtain well–posedness (see, e.g. [12, 47]). Indeed, the trilinear forms $c_i(\cdot, \cdot, \cdot), i = 1, 2$ can be rewritten as

$$
\begin{aligned}
c_i(u_i, w_i, v_i) \quad &= \quad \frac{1}{2} c_i(u_i, w_i, v_i) + \frac{1}{2} c_i(u_i, w_i, v_i) \\
&= \quad \frac{1}{2} c_i(u_i, w_i, v_i) - \frac{1}{2} c_i(u_i, v_i, w_i) + \frac{1}{2} \int_{\Gamma_{i,N} \cup \Gamma_0} (w_i \cdot v_i)(u_i \cdot \mathbf{n}_i) d\Gamma \\
&= \quad \tilde{c}_i(u_i, w_i, v_i) + \frac{1}{2} \int_{\Gamma_{i,N} \cup \Gamma_0} (w_i \cdot v_i)(u_i \cdot \mathbf{n}_i) d\Gamma,
\end{aligned}
$$

where the trilinear form $\tilde{c}(\cdot, \cdot, \cdot)$ is defined as

$$
\tilde{c}_i(u_i, w_i, v_i) = \frac{1}{2} ((u_i \cdot \nabla) w_i, v_i)_{\Omega_i} - \frac{1}{2} ((u_i \cdot \nabla) v_i, w_i)_{\Omega_i} \tag{6}
$$

and it has the following remarkable property

$$
\tilde{c}_i(u_i, v_i, v_i) = 0 \quad \forall u_i, v_i \in V_{i,0}. \tag{7}
$$

We thus consider a new variational formulation of the state equations (4): for $n \geq 1$ and $i = 1, 2$, find $u_i^n \in V_{i,0}$ and $p_i^n \in Q_i$ s.t.

$$
\frac{1}{\Delta t} m_i \left(u_i^n - u_i^{n-1}, v_i \right) + a_i(u_i^n, v_i) + \tilde{c}_i(u_i^n, u_i^n, v_i) + b_i(v_i, p_i^n)
$$
$$
= (f_i^n, v_i)_{\Omega_i} + \left((-1)^{i+1} g, v_i \right)_{\Gamma_0} \qquad \forall v_i \in V_{i,0}, \tag{8a}
$$
$$
b_i(u_i^n, q_i) = 0 \qquad \forall q_i \in Q_i, \tag{8b}
$$

6

which corresponds to imposing the following modified Neumann outlet condition

$$\nu\frac{\partial u_i^n}{\partial \mathbf{n}_i} - p_i^n \mathbf{n}_i - \frac{1}{2}(u_i^n \cdot \mathbf{n}_i)u_i^n = 0 \quad \text{on} \quad \Gamma_{i,N}. \tag{9}$$

The boundary condition (9) no longer arises from the original physical problem (1), but, on the other hand, the condition (7) allows us to derive the *a priori* estimates in the next section. It is also worth mentioning that in the case of DD optimal control problem (3)–(8) the optimum g is the approximation of the modified normal stress on the interface, i.e., $\nu\frac{\partial u_i^n}{\partial \mathbf{n}_i} - p_i^n \mathbf{n}_i - \frac{1}{2}(u_i^n \cdot \mathbf{n}_i)u_i^n$, $i = 1, 2$, whereas in the case of the original problem (3)–(4) the optimum g is the approximation of the physical normal stress $\nu\frac{\partial u_i^n}{\partial \mathbf{n}_i} - p_i^n \mathbf{n}_i, i = 1, 2$.

3.2 *A priori* estimates

The key element for the analysis of the optimal control problem is the derivation of the *a priori* estimates of the solutions to the state equations (8). We will follow the idea introduced in [25] for the stationary Navier–Stokes equations and extend the results to the non–stationary case. We first introduce the various well–known properties of the different terms in the weak formulation (8), see, for instance, [25, 47].

- The forms $m_i(\cdot, \cdot)$, $a_i(\cdot, \cdot)$ and $\tilde{c}_i(\cdot, \cdot, \cdot)$ are continuous: there exist positive constants C_m, C_a and C_c such that

$$|m_i(u_i, v_i)| \leq C_m ||u_i||_{V_{i,0}} ||v_i||_{V_{i,0}} \qquad\qquad \forall u_i, v_i \in V_{i,0}, \tag{10}$$
$$|a_i(u_i, v_i)| \leq C_a ||u_i||_{V_{i,0}} ||v_i||_{V_{i,0}} \qquad\qquad \forall u_i, v_i \in V_{i,0}, \tag{11}$$
$$|\tilde{c}_i(u_i, w_i, v_i)| \leq C_c ||u_i||_{V_{i,0}} ||w_i||_{V_{i,0}} ||v_i||_{V_{i,0}} \qquad \forall u_i, w_i, v_i \in V_{i,0}, \tag{12}$$

- the bilinear form $a_i(\cdot, \cdot)$ is coercive: there exists a positive constant $\alpha > 0$ such that

$$a_i(v_i, v_i) \geq \alpha ||v_i||_{V_{i,0}}^2 \quad \forall v_i \in V_{i,0}, \tag{13}$$

- the bilinear form $b_i(\cdot, \cdot)$ satisfies inf–sup condition: there exists a positive constant $\beta > 0$ such that

$$\sup_{v_i \in V_{i,0}\setminus\{0\}} \frac{b_i(v_i, q_i)}{||v_i||_{V_{i,0}}} \geq \beta ||q_i||_{Q_i} \quad \forall q_i \in Q_i, \tag{14}$$

- the bilinear form $m_i(\cdot, \cdot)$ is non–negative definite, i.e.

$$m_i(v_i, v_i) = ||v_i||_{L^2(\Omega_i)}^2 \geq 0 \quad \forall v_i \in V_{i,0}. \tag{15}$$

By using the properties (10), (13), (7), (15), the trace theorem and equations (8),

we are able to write the following estimate for the solution u_i^n and p_i^n to (8)

$$
\begin{aligned}
||u_i^n||_{V_{i,0}}^2 &\leq \frac{1}{\alpha} a_i(u_i^n, u_i^n) \leq \frac{1}{\alpha} a_i(u_i^n, u_i^n) + \frac{1}{\alpha \Delta t} m_i(u_i^n, u_i^n) \\
&= \frac{1}{\alpha} \left(\frac{1}{\Delta t} m_i(u_i^{n-1}, u_i^n) - \tilde{c}_i(u_i^n, u_i^n, u_i^n) - b_i(u_i^n, p_i^n) \right. \\
&\quad + \left. (f_i^n, u_i^n)_{\Omega_i} + (-1)^{i+1}(g, u_i^n)_{\Gamma_0} \right) \\
&\leq \frac{1}{\alpha} \left(\frac{C_m}{\Delta t} ||u_i^{n-1}||_{V_{i,0}} + ||f_i^n||_{L^2(\Omega_i)} + ||g||_{L^2(\Gamma_0)} \right) ||u_i^n||_{V_{i,0}}.
\end{aligned}
$$

This leads to the following estimate

$$
||u_i^n||_{V_{i,0}} \leq \frac{1}{\alpha} \left(\frac{C_m}{\Delta t} ||u_i^{n-1}||_{V_{i,0}} + ||f_i^n||_{L^2(\Omega_i)} + ||g||_{L^2(\Gamma_0)} \right). \tag{16}
$$

Similarly, by using (14), (10), (11), (12) and equations (8), we obtain

$$
\begin{aligned}
||p_i^n||_{Q_i} &\leq \frac{1}{\beta} \sup_{v_i \in V_{i,0}\setminus\{0\}} \frac{b_i(v_i, p_i^n)}{||v_i||_{V_{i,0}}} \leq \frac{1}{\beta} \sup_{v_i \in V_{i,0}\setminus\{0\}} \frac{\frac{1}{\Delta t}|m_i(u_i^n - u_i^{n-1}, v_i)|}{||v_i||_{V_{i,0}}} \\
&\quad + \frac{1}{\beta} \sup_{v_i \in V_{i,0}\setminus\{0\}} \frac{|a_i(u_i^n, v_i)| + |c_i(u_i^n, u_i^n, v_i)| + |(f_i^n, v_i)_{\Omega_i}|}{||v_i||_{V_{i,0}}} \\
&\quad + \frac{1}{\beta} \sup_{v_i \in V_{i,0}\setminus\{0\}} \frac{|(g, v_i)_{\Gamma_0}|}{||v_i||_{V_{i,0}}} \leq \frac{1}{\beta} \left(\frac{C_m}{\Delta t} + C_a + C_c ||u_i^n||_{V_{i,0}} \right) ||u_i^n||_{V_{i,0}} \\
&\quad + \frac{1}{\beta} \left(\frac{C_m}{\Delta t} ||u_i^{n-1}||_{V_{i,0}} + ||f_i^n||_{L^2(\Omega_i)} + ||g||_{L^2(\Gamma_0)} \right),
\end{aligned}
$$

which together with the estimate (16) leads to

$$
\begin{aligned}
||p_i^n||_{Q_i} &\leq \frac{1}{\beta} \left[\left(1 + \frac{1}{\alpha} \left(\frac{C_m}{\Delta t} + C_a \right) \right) \left(\frac{C_m}{\Delta t} ||u_i^{n-1}||_{V_{i,0}} + ||f_i^n||_{L^2(\Omega_i)} \right. \right. \\
&\quad + \left. \left. ||g||_{L^2(\Gamma_0)} \right) + \frac{C_c}{\alpha^2} \left(\frac{C_m}{\Delta t} ||u_i^{n-1}||_{V_{i,0}} + ||f_i^n||_{L^2(\Omega_i)} + ||g||_{L^2(\Gamma_0)} \right)^2 \right].
\end{aligned} \tag{17}
$$

3.3 Existence of optimal solutions

In this subsection, we prove the existence of optimal solutions for the regularised functional (3). The proof follows the methodology presented by Gunzburger et al. [24]. Firstly, we define the admissibility set as follows:

$$
\begin{aligned}
\mathcal{U}_{ad} = \big\{ (u_1^n, p_1^n, u_2^n, p_2^n, g) &\in V_{1,0} \times Q_1 \times V_{2,0} \times Q_2 \times L^2(\Gamma_0) \text{ such that} \\
&\text{equations (8) are satisfied and } \mathcal{J}_\gamma(u_1^n, u_2^n; g) < \infty \big\}.
\end{aligned}
$$

The admissibility set is clearly non–empty, since, as it was pointed out above, the restrictions to subdomains of the monolithic solution (2) and its corresponding flux on the interface belong to the set.

8

Let $\left\{\left(u_1^{n,(j)}, p_1^{n,(j)}, u_2^{n,(j)}, p_2^{n,(j)}, g^{(j)}\right)\right\}$ be a minimizing sequence in $\mathcal{U}_{ad}$, i.e.,

$$\lim_{j\to\infty} \mathcal{J}_\gamma\left(u_1^{n,(j)}, u_2^{n,(j)}, g^{(j)}\right) = \inf_{(u_1^n, p_1^n, u_2^n, p_2^n, g)\in\mathcal{U}_{ad}} \mathcal{J}_\gamma(u_1^n, u_2^n, g).$$

From the definition of the admissible set $\mathcal{U}_{ad}$ and the functional $\mathcal{J}_\gamma$, it is evident that the set $\{g^{(j)}\}$ is uniformly bounded in $L^2(\Gamma_0)$, which in turn, due to the a priori estimates (16) and (17), implies that the sets $\left\{\left(u_i^{n,(j)}\right)\right\}$ are uniformly bounded in $V_{i,0}$ and the sets $\left\{\left(p_i^{n,(j)}\right)\right\}$ are uniformly bounded in Q_i for $i = 1, 2$. Thus there exists a point $(\hat{u}_1^n, \hat{p}_1^n, \hat{u}_2^n, \hat{p}_2^n, \hat{g}) \in \mathcal{U}_{ad}$ and a subsequence $\left\{\left(u_1^{n,(j_k)}, p_1^{n,(j_k)}, u_2^{n,(j_k)}, p_2^{n,(j_k)}, g^{(j_k)}\right)\right\}$ of the minimising sequence such that for $i = 1, 2$

$$u_i^{n,(j_k)} \rightharpoonup \hat{u}_i^n \quad \text{in} \quad V_{i,0}, \tag{18}$$

$$p_i^{n,(j_k)} \rightharpoonup \hat{p}_i^n \quad \text{in} \quad Q_i, \tag{19}$$

$$g^{(j_k)} \rightharpoonup \hat{g} \quad \text{in} \quad L^2(\Gamma_0), \tag{20}$$

$$u_i^{n,(j_k)} \to \hat{u}_i^n \quad \text{in} \quad L^2(\Omega_i), \tag{21}$$

$$u_i^{n,(j_k)}|_{\Gamma_0} \to \hat{u}_i^n|_{\Gamma_0} \quad \text{in} \quad L^2(\Gamma_0). \tag{22}$$

The last two results are obtained by the trace theorem and compact embedding results in Sobolev spaces, see for example [19, 36].

Since the forms $m_i(\cdot, \cdot)$, $a_i(\cdot, \cdot)$ and $b_i(\cdot, \cdot)$ are bilinear and continuous by (18), (19) and (20) we obtain the following convergence results:

$$m_i(u_i^{n,(j_k)}, v_i) \to m_i(\hat{u}_i^n, v_i) \quad \forall v_i \in V_{i,0},$$

$$a_i(u_i^{n,(j_k)}, v_i) \to a_i(\hat{u}_i^n, v_i) \quad \forall v_i \in V_{i,0},$$

$$b_i(u_i^{n,(j_k)}, q_i) \to b_i(\hat{u}_i^n, q_i) \quad \forall q_i \in Q_i,$$

$$b_i(v_i, p_i^{n,(j_k)}) \to b_i(v_i, \hat{p}_i^n) \quad \forall v_i \in V_{i,0},$$

$$(g^{(j_k)}, v_i)_{\Gamma_0} \to (\hat{g}, v_i)_{\Gamma_0} \quad \forall v_i \in V_{i,0}.$$

Concerning the trilinear form $\tilde{c}_i(\cdot, \cdot, \cdot)$, we exploit integration by part twice, divergence–free conditions for $u_i^{n,(j_k)}$ and $\hat{u}_i^n$, and the strong convergence results (21)–(22). We obtain $\forall v_i \in V_{i,0}$

$$\lim_{k\to\infty} \frac{1}{2} \int_{\Omega_i} (u_i^{n,(j_k)} \cdot \nabla) v_i \cdot u_i^{n,(j_k)} d\Omega = \frac{1}{2} \int_{\Omega_i} (\hat{u}_i^n \cdot \nabla) v \cdot \hat{u}_i^n d\Omega,$$

$$\lim_{k\to\infty} \frac{1}{2} \int_{\Omega_i} (u_i^{n,(j_k)} \cdot \nabla) u_i^{n,(j_k)} \cdot v_i d\Omega = \lim_{k\to\infty} \frac{1}{2} \int_{\Gamma_0} \left(u_i^{n,(j_k)} \cdot v_i\right)\left(u_i^{n,(j_k)} \cdot \mathbf{n}_i\right) d\Gamma$$

$$- \lim_{k\to\infty} \frac{1}{2} \int_{\Omega_i} (u_i^{n,(j_k)} \cdot \nabla) v_i \cdot u_i^{n,(j_k)} d\Omega = \frac{1}{2} \int_{\Gamma_0} \left(\hat{u}_i^n \cdot v_i\right)\left(\hat{u}_i^n \cdot \mathbf{n}_i\right) d\Gamma$$

$$- \frac{1}{2} \int_{\Omega_i} (\hat{u}_i^n \cdot \nabla) v_i \cdot \hat{u}_i^n d\Omega = \frac{1}{2} \int_{\Omega_i} (\hat{u}_i^n \cdot \nabla) \hat{u}_i^n \cdot v_i d\Omega,$$

which leads to

$$\lim_{k \to \infty} \tilde{c}_i(u_i^{n,(j_k)}, u_i^{n,(j_k)}, v_i) = \tilde{c}_i(\hat{u}_i^n, \hat{u}_i^n, v_i) \quad \forall v_i \in V_{i,0}.$$

These convergence results mean that the functions $\hat{u}_1^n, \hat{p}_1^n, \hat{u}_2^n, \hat{p}_2^n, \hat{g}$ satisfy the state equations (8). We also note that the functional $\mathcal{J}_\gamma$ is lower–semicontinuous, i.e

$$\liminf_{j \to \infty} \mathcal{J}_\gamma(u_1^{n,(j_k)}, u_2^{n,(j_k)}, g^{(j_k)}) \geq \mathcal{J}_\gamma(\hat{u}_1^n, \hat{u}_2^n, \hat{g}),$$

which implies that

$$\inf_{(u_1^n, p_1^n, u_2^n, p_2^n, g) \in \mathcal{U}_{ad}} \mathcal{J}_\gamma(u_1^n, u_2^n, g) = \mathcal{J}_\gamma(\hat{u}_1^n, \hat{u}_2^n, \hat{g}).$$

Hence, we have proved the existence of optimal solutions.

3.4   Convergence with vanishing penalty parameter

In the previous section, we have proved the existence of optimal solutions of the regularised function $\mathcal{J}_\gamma$ for any $\gamma > 0$, where the parameter γ indicates the relative importance of the two terms entering the definition of the functional. This poses an issue in our domain–decomposition setting since the optimal solution does not satisfy the coupling condition $u_1^n|_{\Gamma_0} = u_2^n|_{\Gamma_0}$. In this section, we prove the existence of an optimal solution to the unregularised functional $\mathcal{J}$ with corresponds to the functional $\mathcal{J}_\gamma$ with $\gamma = 0$.

Let $(u^{n,mon}, p^{n,mon})$ be a weak solution to the monolithic equations (2), and for each $\gamma > 0$ we denote by $(u_1^{n,\gamma}, p_1^{n,\gamma}, u_2^{n,\gamma}, p_2^{n,\gamma}, g^\gamma)$ an optimum of $\mathcal{J}_\gamma$ under the constraints (8). We define the following functions for $i = 1, 2$:

$$
\begin{aligned}
u_i^{n,mon} &:= u^{n,mon}|_{\Omega_i}, \\
p_i^{n,mon} &:= p^{n,mon}|_{\Omega_i}, \\
g^{mon} &:= \nu \frac{\partial u_1^{n,mon}}{\partial \mathbf{n}_1} - p_1^{n,mon}\mathbf{n}_1 - \frac{1}{2}(u_1^{n,mon} \cdot \mathbf{n}_1)u_1^{n,mon} \quad \text{on } \Gamma_0.
\end{aligned}
$$

Due to optimality of the point $(u_1^{n,\gamma}, p_1^{n,\gamma}, u_2^{n,\gamma}, p_2^{n,\gamma}, g^\gamma)$, we obtain that $\forall \gamma > 0$

$$\mathcal{J}_\gamma(u_1^{n,\gamma}, p_1^{n,\gamma}, u_2^{n,\gamma}, p_2^{n,\gamma}, g^\gamma) \leq \mathcal{J}_\gamma(u_1^{n,mon}, p_1^{n,mon}, u_2^{n,mon}, p_2^{n,mon}, g^{mon}),$$

which due to the definition of $u_1^{n,mon}$ and $u_2^{n,mon}$ gives us the following bound:

$$\frac{1}{2}\int_{\Gamma_0} |u_1^{n,\gamma} - u_2^{n,\gamma}|^2 \, d\Gamma + \frac{\gamma}{2}\int_{\Gamma_0} |g^\gamma|^2 \, d\Gamma \leq \frac{\gamma}{2}\int_{\Gamma_0} |g^{mon}|^2 \, d\Gamma \quad \forall \gamma > 0.$$

The last inequality tells us that the sequence $\{g^\gamma : \gamma > 0\}$ is bounded in $L^2(\Gamma_0)$. Following the exact same lines of arguments as in the previous section, we are able to deduce that there is a subsequence of the original sequence (we will keep the same notation for the sake of simplicity) that converges to $(u_1^{n,*}, p_1^{n,*}, u_2^{n,*}, p_2^{n,*}, g^*) \in$

10

$V_{1,0} \times Q_1 \times V_{2,0} \times Q_2 \times L^2(\Gamma_0)$ in the sense of (18) – (22). In addition to this, the inequality above tells us that $||u_1^{n,\gamma} - u_2^{n,\gamma}||_{L^2(\Gamma_0)} \to 0$ as $\gamma \to 0$, which in turn yields $u_1^{1,*} = u_2^{1,*}$ a.e. on Γ_0. The non–negativity of $\mathcal{J}$ leads to the fact that $(u_1^{n,*}, p_1^{n,*}, u_2^{n,*}, p_2^{n,*}, g^*)$ is a global minimum of $\mathcal{J}$. Also, it is easy to see that the following functions $u^{n,*} \in H_{0,\Gamma_D}^1(\Omega), p^{n,*} \in L^2(\Omega)$, defined as

$$u^{n,*} := \begin{cases} u_1^{n,*}, & \text{in } \Omega_1 \cup \Gamma_0, \\ u_2^{n,*}, & \text{in } \Omega_2 \cup \Gamma_0, \end{cases} \qquad p^{n,*} := \begin{cases} p_1^{n,*}, & \text{in } \Omega_1 \cup \Gamma_0, \\ p_2^{n,*}, & \text{in } \Omega_2 \cup \Gamma_0, \end{cases}$$

satisfy the monolithic equations (2) in the weak sense.

Remark (Uniqueness of optimal solutions). It is well–known that the solution to the non–stationary incompressible Navier–Stokes equation in 2D is unique [47], and it can be proved that uniqueness transfers to the implicit–Euler time–discretisation scheme with a good choice of a time–step parameter (see, for instance, [27]). This, together with the convexity of the objective functional, leads to the uniqueness of the optimal solution discussed above.

Remark (Weak formulation with "non–symmetric" trilinear form). The condition (7) was crucial for analyzing the optimal-control problem. Regarding the problem formulated in the weak form (4), numerical experiments yield similar convergence results when employing the trilinear form (6). We believe that this similarity arises because we have tested scenarios where the monolithic solution (u^n, p^n) is well-posed, either satisfying the condition $\int_{\Gamma_N} |u^n|^2 u^n \cdot \mathbf{n} \, d\Gamma \geq 0$ or considering only Dirichlet boundary conditions (with the pressure variational space comprising solely zero-mean functions to ensure uniqueness).

3.5   DD optimal control problem gradient

Resorting to the Lagrangian functional and the sensitivity approaches described in [43], it is easy to show that the gradient of the original DD optimal control problem (3)– (4) at a point $g \in L^2(\Gamma_0)$ is given by

$$\frac{d\mathcal{J}_\gamma}{dg}(u_1^n, u_2^n; g) = \gamma g + \xi_1|_{\Gamma_0} - \xi_2|_{\Gamma_0}, \tag{23}$$

where ξ_1 and ξ_2 are the solutions to the following adjoint problems for $i = 1, 2$: given $u_1^n \in V_1$ and $u_2^n \in V_2$ solutions to the state problem (4), find $\xi \in V_{i,0}$ and $\lambda_i \in Q_i$ solving

$$\frac{1}{\Delta t} m_i(\eta_i, \xi_i) + a_i(\eta_i, \xi_i) + c_i\left(\eta_i, u_i^n, \xi_i\right) + c_i\left(u_i^n, \eta_i, \xi_i\right)$$
$$+ b_i(\eta_i, \lambda_i) = ((-1)^{i+1} \eta_i, u_1^n - u_2^n)_{\Gamma_0}, \qquad \forall \eta_i \in V_{i,0}, \quad \text{(24a)}$$
$$b_i(\xi_i, \mu_i) = 0, \qquad \forall \mu_i \in Q_i. \quad \text{(24b)}$$

The closed–form formula for the gradient of the objective functional $\mathcal{J}_\gamma$ is an essential element to tackle the optimal control problem utilizing gradient–based optimisation

algorithms as the one presented in [43] or more effective algorithms such as Broyden–Fletcher–Goldfarb–Shanno (BFGS) [16, 64] and Newton Conjugate Gradient (CG) [37, 5] algorithms. This approach leads to an iterative procedure in which both the state and the adjoint subcomponents of the DD system can be solved independently at each optimisation iteration on each subdomain and then updated through the gradient. In particular, in the numerical tests we will perform, we will use the limited–memory Broyden–Fletcher–Goldfarb–Shanno (L–BFGS–B) optimisation algorithm [13], which shows faster convergence and higher efficiency with respect to other gradient–based algorithms.

4   Finite Element Discretisation

In this section, we present the Finite Element spatial discretisation for the optimal control problem previously introduced. We assume to have at hand two well–defined triangulations $\mathcal{T}_1$ and $\mathcal{T}_2$ over the domains Ω_1 and Ω_2, respectively, and a one–dimensional discretisation $\mathcal{T}_0$ of the interface Γ_0. We can then define usual Lagrangian FE spaces $V_{i,h} \subset V_i$, $V_{i,0,h} \subset V_{i,0}$, $Q_{i,h} \subset Q_i$, for $i = 1, 2$, and $X_h \subset L^2(\Gamma_0)$ endowed with $L^2(\Gamma_0)$–norm. Since the problems at hand have a saddle–point structure, to guarantee the well–posedness of the discretised problem, we require the FE spaces to satisfy the following inf–sup conditions: there exist $c_1, c_2, c_3, c_4 \in \mathbb{R}^+$ s.t.

$$\inf_{q_{i,h} \in Q_{i,h} \setminus \{0\}} \sup_{v_{i,h} \in V_{i,h} \setminus \{0\}} \frac{b_i(v_{i,h}, q_{i,h})}{||v_{i,h}||_{V_{i,h}} ||q_{i,h}||_{Q_{i,h}}} \geq c_i, \quad i = 1, 2, \tag{25}$$

$$\inf_{q_{i,h} \in Q_{i,h} \setminus \{0\}} \sup_{v_{i,h} \in V_{i,0,h} \setminus \{0\}} \frac{b_i(v_{i,h}, q_{i,h})}{||v_{i,h}||_{V_{i,0,h}} ||q_{i,h}||_{Q_{i,h}}} \geq c_{i+2}, \quad i = 1, 2. \tag{26}$$

A very common choice in this framework is to use the so–called Taylor–Hood finite element spaces, namely the Lagrange polynomial approximation of the second–order for velocity and of the first–order for pressure. We point out that the order of the polynomial space X_h will not lead to big computational efforts as it is defined on the 1–dimensional curve Γ_0 by the restriction of the second–order Lagrangian polynomial approximations defined by the velocity spaces. For this reason, we have chosen conformal spaces that share the degrees of freedom on Γ_0.

Using the Galerkin projection, we can derive the following discretised optimisation problem. Minimise over $g_h \in X_h$ the functional

$$\mathcal{J}_{\gamma,h}(u_{1,h}^n, u_{2,h}^n; g_h) := \frac{1}{2} \int_{\Gamma_0} \left| u_{1,h}^n - u_{2,h}^n \right|^2 d\Gamma + \frac{\gamma}{2} \int_{\Gamma_0} |g_h|^2 d\Gamma \tag{27}$$

under the constraints that $u_{i,h}^n \in V_{i,h}$, $p_{i,h}^n \in Q_{i,h}$ satisfy the following variational

equations for $i = 1, 2$

$$\frac{m_i(u_{i,h}^n - u_{i,h}^{n-1}, v_{i,h})}{\Delta t} + a_i(u_{i,h}^n, v_{i,h}) + c_i(u_{i,h}^n, u_{i,h}^n, v_i)$$

$$+ b_i(v_{i,h}, p_{i,h}^n) = (f_i^n, v_{i,h})_{\Omega_i} + \left((-1)^{i+1} g_h, v_{i,h}\right)_{\Gamma_0}, \qquad \forall v_i \in V_{i,0,h}, \quad (28a)$$

$$b_i(u_{i,h}^n, q_{i,h}) = 0, \qquad \forall q_{i,h} \in Q_{i,h}, \quad (28b)$$

$$u_i^n = u_{i,D,h}^n, \qquad \text{on } \Gamma_{i,D}, \quad (28c)$$

where $u_{i,D,h}^n$ is the Galerkin projection of $u_{i,D}$ onto the trace–space $V_{i,h}|_{\Gamma_{i,D}}$. Notice that the structure of the equations (28) and of the functional (27) is the same as the one of the continuous case. This allows us to provide the following expression of the gradient of the discretised functional (27):

$$\frac{d\mathcal{J}_{\gamma,h}}{dg_h}(u_{1,h}^n, u_{2,h}^n; g_h) = \gamma g_h + \xi_{1,h}|_{\Gamma_0} - \xi_{2,h}|_{\Gamma_0}, \qquad (29)$$

where $\xi_{1,h}$ and $\xi_{2,h}$ are the solutions to the discretised adjoint problem: for $i = 1, 2$ find $\xi_{i,h} \in V_{i,0,h}$ and $\lambda_{i,h} \in Q_{i,h}$ that satisfy

$$\frac{m_i(\eta_{i,h}, \xi_{i,h})}{\Delta t} + a_i(\eta_{i,h}, \xi_{i,h}) + c_i(\eta_{i,h}, u_{i,h}^n, \xi_i) + c_i(u_{i,h}^n, \eta_{i,h}, \xi_{i,h})$$

$$+ b_i(\eta_{i,h}, \lambda_{i,h}) = ((-1)^{i+1} \eta_{i,h}, u_{1,h}^n - u_{2,h}^n)_{\Gamma_0}, \qquad \forall \eta_{i,h} \in V_{i,0,h},$$

$$(30a)$$

$$b_i(\xi_{i,h}, \mu_{i,h}) = 0, \forall \mu_{i,h} \in Q_{i,h}. \qquad (30b)$$

In (29), the restriction $\cdot|_{\Gamma_0}$ is meant as an $L^2(\Gamma_0)$–projection onto space X_h. We would also like to stress that at the algebraic level, the discretised minimisation problem acts only on the finite–dimensional space $\mathbb{R}^p$ of the variable g_h, where p is the number of Finite Element degrees of freedom that belong to the interface Γ_0.

Remark (The choice of discrete space X_h). The choice for space X_h might have an important impact on the convergence properties of the iterative optimisation algorithms. For the numerical tests in this work, since the objective functional and its gradient are defined by the state or adjoint solutions, our choice is to use the FE space X_h defined by the degrees of freedom shared by the FE spaces $V_{1,h}$ and $V_{2,h}$. Nevertheless, it is of much interest to conduct the convergence analysis of the FE DD control problem which will be the subject of future work.

5   Reduced–Order Model

As highlighted in Section 1, reduced–order methods are efficient tools for significant reduction of the computational time for parameter–dependent PDEs. This section deals with the ROM for the problem obtained in the previous section, where the state equations, namely Navier–Stokes equations, are assumed to be dependent on a set of physical parameters. We rely on the classical Proper Orthogonal Decomposition (POD)

13

technique for generating reduced spaces for each subcomponent of the problem locally. In this section, we describe two online phases based on a Galerkin projection onto the reduced spaces and on a multilayer perceptron neural network.

5.1 POD–Galerkin

We assume to have at hand local reduced spaces constructed by the classical POD compression technique [28, 43, 11, 26] coupled with the velocity supremiser technique as described in Appendix A to ensure the inf–sup stability of the reduced velocity–pressure spaces. For the interested reader, we describe the POD algorithm adopted in our case in Appendix B. We can then define the reduced functions expansions

$$(u_{1,0,N}^n, p_{1,N}^n, u_{2,0,N}^n, p_{2,N}^n, g_N) \in V_{rb}^{u_1} \times V_{rb}^{p_1} \times V_{rb}^{u_2} \times V_{rb}^{p_2} \times V_{rb}^g$$

as

$$u_{i,0,N}^n := \sum_{k=1}^{N_{u_i}} \underline{u}_{i,0,k}^n \Phi_k^{u_i}, \quad p_{i,N} := \sum_{k=1}^{N_{p_i}} \underline{p}_{i,k}^n \Phi_k^{p_i}, \ i = 1, 2, \quad g_N := \sum_{k=1}^{N_g} \underline{g}_k \Phi_k^g.$$

In the previous equations, the underlined variables denote the coefficients of the basis expansion of the reduced solution. Then, the online reduced problem reads as follows: minimise over $g_N \in V_{rb}^g$ the functional

$$\mathcal{J}_{\gamma,N}(u_{1,N}^n, u_{2,N}^n; g_N) := \frac{1}{2} \int_{\Gamma_0} \left| u_{1,N}^n - u_{2,N}^n \right|^2 d\Gamma + \frac{\gamma}{2} \int_{\Gamma_0} |g_N|^2 d\Gamma \qquad (31)$$

where $u_{1,N}^n = u_{1,0,N}^n + l_{1,N}^n, u_{2,N} = u_{2,0,N}^n + l_{2,N}^n$ and $(u_{1,0,N}, p_{1,N}, u_{2,0,N}, p_{2,N})$ satisfy the following reduced equations:

$$
\begin{aligned}
\frac{1}{\Delta t} m_i(u_{i,0,N}^n, v_{i,N}) \quad + \quad & a_i(u_{i,0,N}^n, v_{i,N}) + c_i(u_{i,0,N}^n, u_{i,0,N}^n, v_{i,N}) \\
+ \quad & c_i(u_{i,0,N}^n, l_{i,N}^n, v_{i,N}) + c_i(l_{i,N}^n, u_{i,0,N}^n, v_{i,N}) \qquad (32\text{a}) \\
+ \quad & b_i(v_{i,N}, p_{i,N}^n) = (f_i^n, v_{i,N})_{\Omega_i} + ((-1)^{i+1} g_N, v_{i,N})_{\Gamma_0} \\
+ \quad & \frac{1}{\Delta t} m_i(u_{i,0,N}^{n-1}, v_{i,N}) - \frac{1}{\Delta t} m_i(l_{i,N}^n, v_{i,N}) \\
- \quad & a_i(l_{i,N}^n, v_{i,N}^n) - c_i(l_{i,N}, l_{i,N}, v_{i,N}), \quad \forall v_{i,N} \in V_{rb}^{u_i}, \\
b_i(u_{i,0,N}^n, q_{i,N}) \quad = \quad & -b_i(l_{i,N}^n, q_{i,N}), \quad \forall q_{i,N} \in V_{rb}^{p_i}, \qquad (32\text{b})
\end{aligned}
$$

where $l_{i,N}^n$ is the Galerkin projection of the lifting function $l_{i,h}^n$ (see Appendix A) for more details) to the finite dimensional vector space $V_{rb}^{u_i}$ and $i = 1, 2$.

Similarly to the offline phase, we notice that the structure of the equations (32) and the functional (31) are the same as the ones of the continuous case, so this enables us to provide the following expression of the gradient of the reduced functional (31)

$$\frac{d\mathcal{J}_{\gamma,N}}{dg_N}(u_{1,N}^n, u_{2,N}^n; g_N) = \gamma g_N + \xi_{1,N}|_{\Gamma_0} - \xi_{2,N}|_{\Gamma_0}, \qquad (33)$$

where $(\xi_{1,N}, \xi_{2,N})$ are the solutions to the reduced adjoint problem: find $(\xi_{1,N}, \lambda_{1,N}, \xi_{2,N}, \lambda_{2,N}) \in V_{rb}^{u_1} \times V_{rb}^{p_1} \times V_{rb}^{u_2} \times V_{rb}^{p_2}$ such that it satisfies, for $i = 1, 2$,

$$\frac{1}{\Delta t} m_i(\eta_{i,N}, \xi_{i,N}) + a_i(\eta_{i,N}, \xi_{i,N}) + c_i\left(\eta_{i,N}, u_{i,N}^n, \xi_i\right) + c_i\left(u_{i,N}^n, \eta_{i,N}, \xi_{i,N}\right) \quad (34a)$$

$$+ b_i(\eta_{i,N}, \lambda_{i,h}) = ((-1)^{i+1}\eta_{i,N}, u_{1,N}^n - u_{2,N}^n)_{\Gamma_0}, \quad \forall \eta_{i,N} \in V_{i,N}^{u_i},$$

$$b_i(\xi_{i,N}, \mu_{i,N}) = 0, \qquad \forall \mu_{i,N} \in V_{i,N}^{p_i}. \quad (34b)$$

Above, the restriction $\cdot|_{\Gamma_0}$ is meant as an $L^2(\Gamma_0)$–projection onto space V_{rb}^g. At the algebraic level, the reduced minimisation problem can be recast in the setting of the finite–dimensional space $\mathbb{R}^p$, where p is the number of reduced basis functions used for the control variable g_N in the online phase, that is $p = N_g$. We would like also to highlight the difference in the spaces we use to approximate the adjoint equations (34) with respect to the stationary case described in [43]. Here, instead of creating separate spaces for adjoint velocities, we used the same reduced spaces as for the state equations (32) and this renders the offline stage less computationally expensive, it requires less storage and, in addition, the numerical simulations proved to be more stable.

Here, we would like to also highlight the possibility of treating the different subdomains and control variables in the FOM or the ROM independently [44], obtaining various combinations of ROM/FOM models.

5.2 POD–NN

In this section, we would like to give a quick overview of the POD–NN method [29]. After the construction of the POD reduced spaces as described in Section B, the POD–NN tries to learn the map that, given the physical parameters and time, returns the reduced coefficients of the POD projection. To learn this map, we consider a training set of parameters $\Xi := \{(\mu_z, t_z)\}_{z=1}^{KM}$ defined by a tensor product of K physical parameters and M time steps. We compute the FOM solutions for these parameters, $U_z = (u_{1,h}(\mu_z, t_z), u_{2,h}(\mu_z, t_z), p_{1,h}(\mu_z, t_z), p_{2,h}(\mu_z, t_z))$ for $z = 1, \ldots, KM$ and we project the snapshots onto each reduced space to obtain the reduced coefficients, for $i = 1, 2$

$$\underline{u}_{i,0}(\mu_z, t_z) := \Pi_{rb,u_i}(u_{i,h}(\mu_z, t_z)), \qquad \underline{p}_i(\mu_z, t_z) := \Pi_{rb,p_i}(p_{i,h}(\mu_z, t_z)).$$

Now, for each component we consider as input the parameters (μ, t) and as output the reduced coefficients $\underline{u}_{i,0}$ or $\underline{p}_i$ for $i = 1, 2$: we therefore define $U_{u_i,\text{output}} = \{\underline{u}_{i,0}(\mu_z, t_z)\}$ and $U_{p_i,\text{output}} = \{\underline{p}_i(\mu_z, t_z)\}$.

Once defined the input and output training sets, we build an artificial neural network (ANN) for each component $* \in \{u_1, p_1, u_2, p_2\}$ that approximates $\Xi \to U_{*,\text{output}}$. Then, the POD–NN reduced solutions are defined by recovering the predicted values by these ANN in the corresponding FEM space. Notice that this approach does not require any optimisation algorithm in the online phase, just the evaluation of the ANN at the required parameter value and time step.

The ANN used in this algorithm is a simple dense multilayer perceptron that consists of repeated compositions of affine operations and nonlinear activation functions
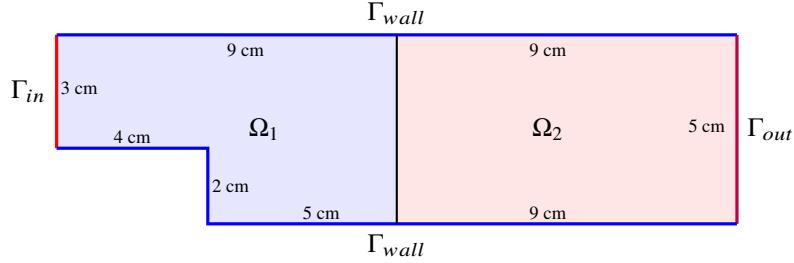
15

Figure 2: Physical domain and domain decomposition for the backward–facing step problem

[21]. The chosen architecture contains 3 hidden layers with 40, 60 and 100 neurons, respectively. This means that there are 4 affine mappings between the input, hidden and output layers, and at each layer, we use the hyperbolic tangent as an activation function. The learning of the weights and biases of the NN is optimised using the Adam algorithm [31], a variation of the stochastic gradient descent. In both test cases of the numerical result section, we used 5000 as the maximum number of optimisation iterations (epochs) and 10^{-5} as target for the loss functional.

The hyperparameters are the result of a quick optimization process. We observed that a lower number of layers/neurons were less accurate in representing the map of interest, while more layers were too expensive to be trained in terms of necessary epochs without resulting in more accurate networks.

6 Numerical Results

We now present some numerical results obtained by applying the two–domain decomposition optimisation algorithm to the backward–facing step and the lid–driven cavity flow benchmarks.

All the numerical simulations for the offline phase were obtained using the software FEniCS [1], whereas the online phase simulations were carried out using RBniCS [2] and EZyRB [3].
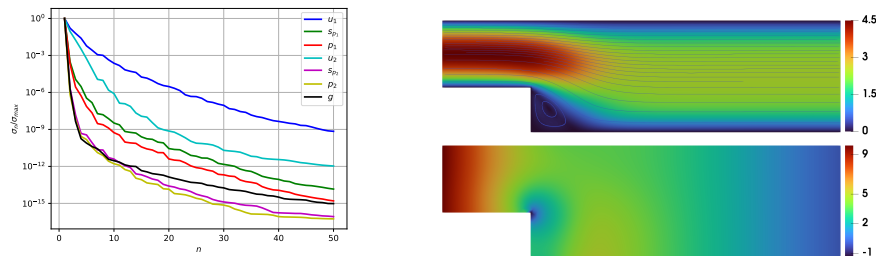
6.1 Backward–facing step test case

We start with the backward–facing step flow test case. Figure 2 represents the physical domain of interest, the dimensional lengths and the boundary conditions. The splitting into two domains is performed by dissecting the domain by a vertical segment at the distance 9 cm from the left end of the channel, as shown in Figure 2.

We consider zero initial velocity condition, homogeneous Dirichlet boundary conditions on walls Γ_{wall} for the fluid velocity, and homogeneous Neumann conditions on the outlet Γ_{out}, meaning that we assume free outflow on this portion of the boundary.

We impose a parabolic profile u_{in} on the inlet boundary Γ_{in}, where $u_{in}(x, y) = (w(y), 0)^T$ with $w(y) = \bar{U} \cdot \frac{4}{9}(y - 2)(5 - y)$, $y \in [2, 5]$; the range of $\bar{U}$ is reported

16

Physical parameters		FE parameters	
Range ν	[0.4, 2]	Velocity–pressure space in a cell	$\mathbb{P}^2 \times \mathbb{P}^1$
Range $\bar{U}$	[0.5, 4.5]	Total dofs	27,890
Final time T	1	Dofs at interface	130
		Time step Δt	0.01
Optimization		Snapshots training set parameters	
Algorithm	L–BFGS–B	Timestep number M	100
$\mathrm{it_{max}}$	1000	Parameters training set size K	62
$\mathrm{tol_{opt}}$	10^{-9}	Maximum retained modes $N_{\max}$	100

Table 1: Backward–facing step: computational details of the offline stage.

(a) The singular values as a function of the number of POD modes (log scale in y–direction)

(b) Monolithic model fluid velocity and pressure

Figure 3: Backward–facing step: POD singular eigenvalue decay of the first 50 POD modes (a) and the monolithic solution for a parameter $(\bar{U}, \nu) = (4.5, 0.4)$ at the final time step (b)

in Table 1. Two physical parameters are considered: the viscosity ν and the maximal magnitude $\bar{U}$ of the inlet velocity profile u_{in}. Details of the offline stage and the finite–element discretisation are summarised in Table 1. High–fidelity solutions are obtained by carrying out the minimisation in the space of dimension equal to the number of degrees of freedom at the interface, which is 130 for our test case. The best performance has been achieved by using the limited–memory Broyden–Fletcher–Goldfarb–Shanno (L–BFGS–B) optimisation algorithm [13], where the following stopping criteria were applied: either the maximal number of iteration $\mathrm{it_{max}}$ is reached or the gradient norm of the target functional is less than the given tolerance $\mathrm{tol_{opt}}$ or the relative reduction of the functional value is less than the tolerance that is automatically chosen by the `scipy` library [63].

Snapshots are sampled from a training set of K parameters randomly sampled from the 2–dimensional parameter space for each time–step t_i, $i = 1, ..., M$, and the first $N_{\max}$ POD modes have been retained for each component. Figure 3a shows the POD singular values for all the state and the control variables; we can see an evident exponential decay of the singular values. Figure 3b shows an example of a monolithic (whole–domain)
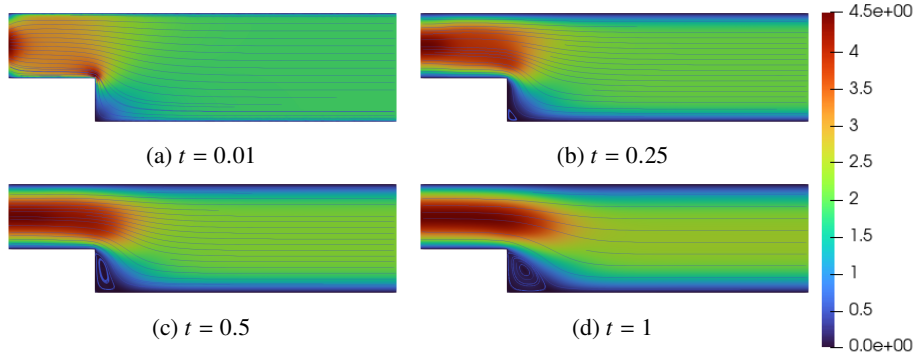
17

Figure 4: Backward–facing step: high–fidelity solution for the velocities u_1 and u_2 at 4 different time instances
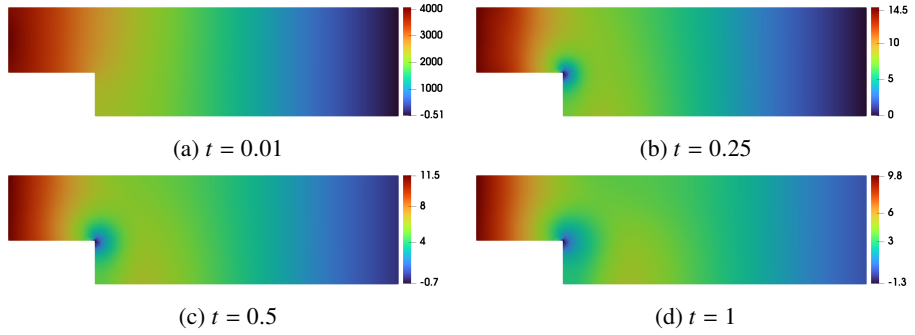

Figure 5: Backward–facing step: high–fidelity solution for the pressures p_1 and p_2 at 4 different time instances

solution that will be the benchmark solution for the numerical and error analysis of the DD–FOM and the ROM.

In Table 2, we list the values of the parameters for which we conduct a numerical test of the ROM and the number of POD modes for each component of the problem. The number of reduced bases is chosen so that the discarded energy for each of the components is less than 10^{-6}. Reduced–order solutions are obtained by carrying out the minimisation in the space of dimension equal to the number of POD modes for the control g, which is 5 for our test case. Clearly, the minimisation in this space of dimension 5 is much simpler than in the FOM one. The optimisation algorithm used in this test case is the same as in the FOM case described above.

Figures 4–5 represent the high–fidelity solutions for a value of the parameters $(\bar{U}, \nu) = (4.5, 0.4)$ at 4 different time instances. Visually, we can see a great degree of continuity on the interface, which will be highlighted below. Figure 6 shows the spatial distribution of the error with respect to the monolithic solution at the final time step for both the FOM and ROM solutions. As expected, the error of the FOM solution is

18

Parameter		POD modes					
ν	0.4	velocity u_1	30	pressure p_1	5	supremiser s_1	5
$\bar{U}$	4.5	velocity u_2	12	pressure p_2	5	supremiser s_2	5
		control g	5				

Table 2: Backward–facing step: computational details of the online stage.

(a) FOM velocity (magnitude)

(b) FOM pressure

(c) ROM velocity (magnitude)

(d) ROM pressure
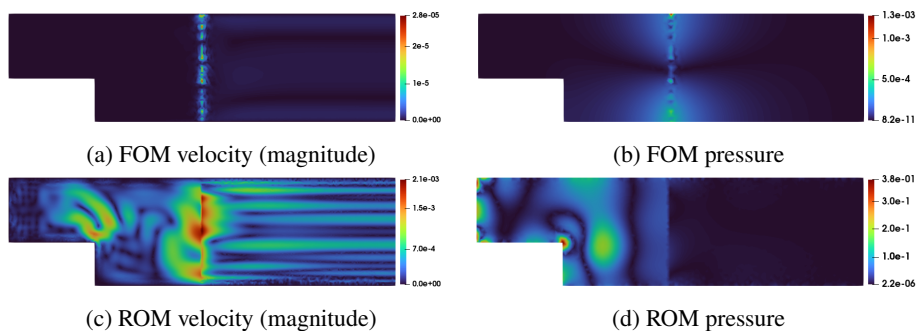
Figure 6: Backward–facing step: absolute errors of DD–FOM and ROM solutions w.r.t. the monolithic solution at the final time step

mostly concentrated at the interface, while the ROM solutions show some extra noise due to the POD reduction.

Figure 7 shows the number of iterations for both FOM and ROM optimisation processes for value of the parameters $(\bar{U}, \nu) = (4.5, 0.4)$. We plot this quantity for one parameter, instead of a statistic over the parameter space, to better grasp the variability of the quantity in time, while its averages is a good representative for all other parameters. Each iteration of the optimisation algorithm requires at least one computation of the state and the adjoint solvers. Therefore, we can see that we have managed to obtain a significant reduction in terms of computational efforts: the average number of iterations over all time steps in the case of the FOM solver is 170, while it is 24 in the case of the ROM solver. Additionally, each solver at the reduced level is of much smaller dimension (see Table 2), and with good use of hyper–reduction techniques (see, for instance, [28]), it will allow to obtain very efficient solvers in terms of computational time.

We would like also to provide a comparison of the full–order and the reduced–order models with non–intrusive POD–NN model. Due to the discontinuity given by the initial and the boundary conditions, the first time step was excluded from the training set in order to achieve better performance. In practice, the first few simulations can be computed with a Galerkin projection or some FOM steps. Figure 8 shows the relative errors with respect to the monolithic solution for the FOM, ROMs and POD–NN model for value of the parameters $(\bar{U}, \nu) = (4.5, 0.4)$, again, to show the variability in time. As we can see, both FOM and ROM give us very good convergence results, i.e., the relative error does not exceed 1% in either case. Regarding the POD–NN, in terms of computational time, it is very effective, but the approximation can be very poor,
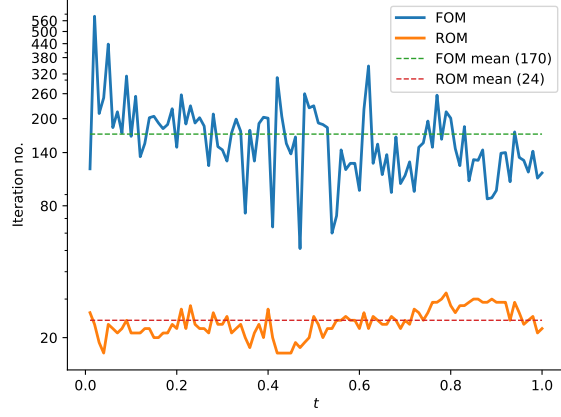
Figure 7: Backward–facing step at $\nu = 0.4$ and $\bar{U} = 4.5$: number of optimisation iterations of FOM and ROM solvers
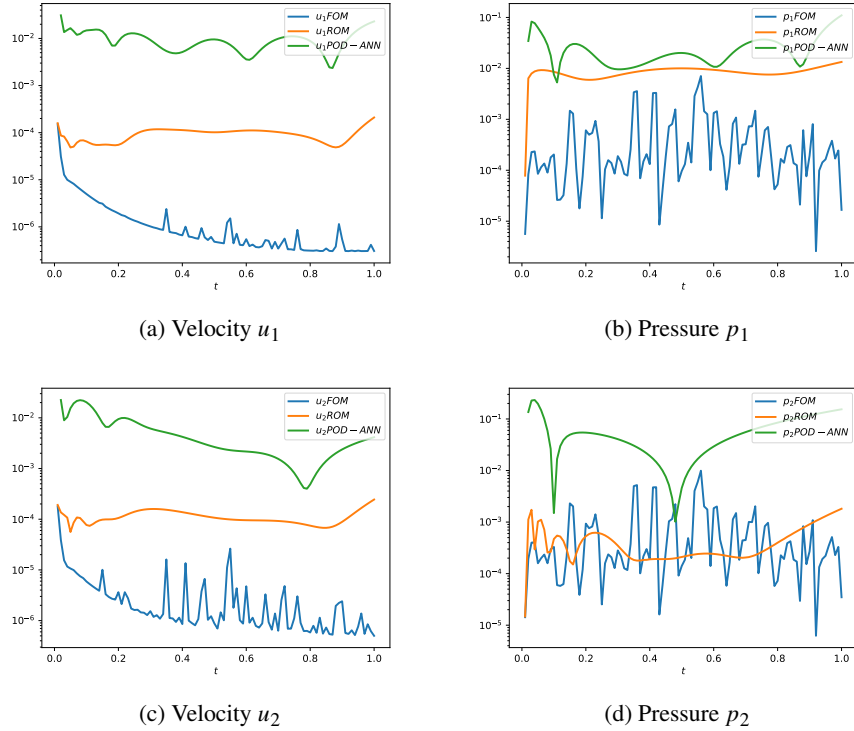

(a) Velocity u_1

(b) Pressure p_1

(c) Velocity u_2

(d) Pressure p_2

Figure 8: Backward–facing step at $\nu = 0.4$ and $\bar{U} = 4.5$: relative errors of FOM, ROM and POD–NN solutions w.r.t. the monolithic solution
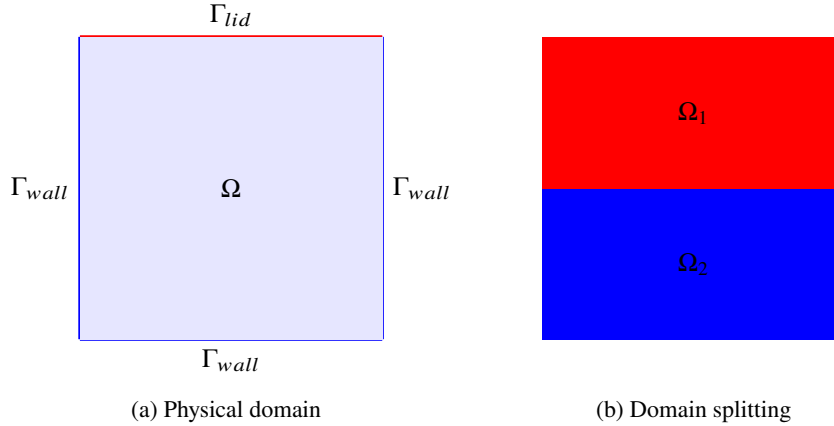
20

(a) Physical domain   (b) Domain splitting

Figure 9: Lid–driven cavity flow geometry and domain decomposition

especially in the initial and final time steps. Just to give an idea of the differences in the computational times, one time step of the FOM takes between 30 and 60 minutes, one time step of the ROM (without hyper–reduction) takes around 5 minutes, while a POD–NN prediction needs around 0.003 seconds. One of the possible scenarios could be a combination of the ROM and the POD–NN model based on the *a posteriori* error estimates, so that the time steps in which the much more computationally effective ANN model fails to produce a sufficiently accurate approximation, the Galerkin projection ROM is applied. Similar ideas can be found *inter alia* in [7]. This will be the subject of future works.

6.2 Lid–driven cavity flow test case

In this section, we provide the numerical simulation for the lid–driven cavity flow test case. Figure 9a represents the physical domain of interest – the unit square. The division into two subdomains is performed by dissecting the domain by a median horizontal line, as shown in Figure 9b.

We consider zero initial velocity condition, homogeneous Dirichlet boundary conditions on the boundary Γ_{wall} for the fluid velocity and the nonzero horizontal constant velocity on the lid boundary Γ_{lid}: $u_{lid} = (\bar{U}, 0)$; the values of $\bar{U}$ are reported in Table 3. We consider one physical parameter – the magnitude $\bar{U}$ of the lid velocity profile u_{in}. Details of the offline stage and the FE discretisation are summarised in Table 3. High–fidelity solutions are obtained by carrying out the minimisation in the space of dimension equal to the number of degrees of freedom at the interface, which is 294 for our test case. Snapshots are derived from a training set of K values uniformly sampled from the 1–dimensional parameter space for each time–step t_i, $i = 1, ..., M$, and the first N_{max} POD modes have been retained for each component. In Figure 10a, we see that the POD singular values decay even faster than in the previous test for all the state and the control variables. As before, we show in Figure 10b the monolithic (whole–domain) solution related to the parameter ($\bar{U} = 3$) on which we will test the

Physical parameters		FE parameters	
Range ν	[1, 1]	Velocity–pressure space in a cell	$\mathbb{P}^2 \times \mathbb{P}^1$
Range $\bar{U}$	[0.5, 5]	Total dofs	58,056
Final time T	0.4	Dofs at interface	294
		Time step Δt	0.01
Optimization		Snapshots training set parameters	
Algorithm	L–BFGS–B	Timestep number M	40
it_{max}	300	Parameters training set size K	10
tol_{opt}	10^{-7}	Maximum retained modes N_{max}	100

Table 3: Lid–driven cavity flow: computational details of the offline stage.

(a) The singular values as a function of the number of POD modes (log scale in y–direction)

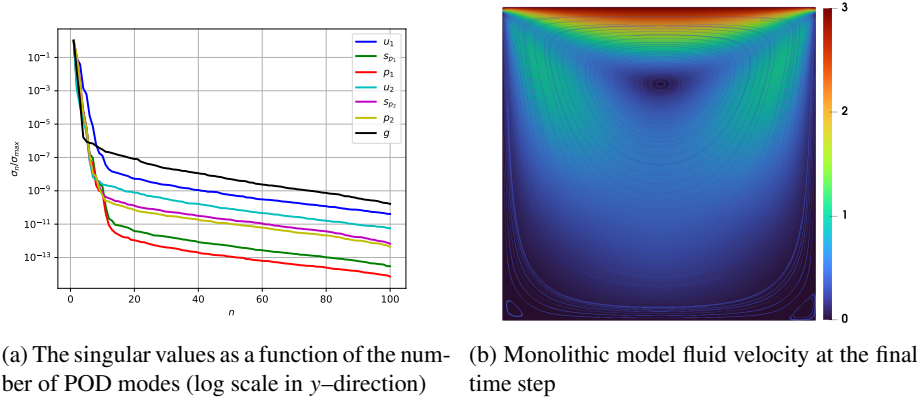(b) Monolithic model fluid velocity at the final time step

Figure 10: Lid–driven cavity flow: POD singular eigenvalue decay of POD modes (a) and the monolithic solution for a parameter $\bar{U} = 3$ at the final time step (b)
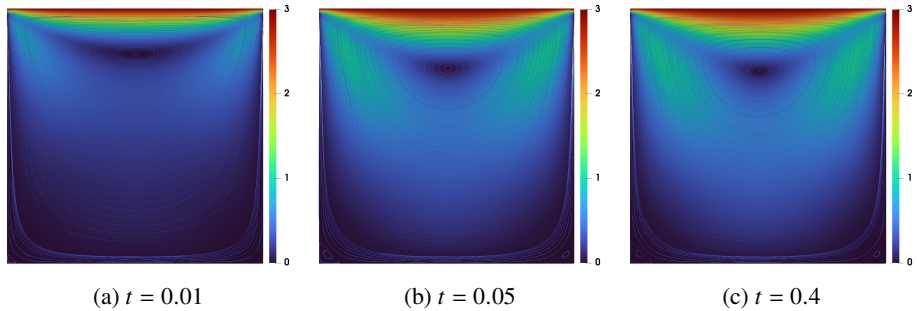

(a) $t = 0.01$ (b) $t = 0.05$ (c) $t = 0.4$

Figure 11: Lid–driven cavity flow: FOM velocity solution at 3 different time instances

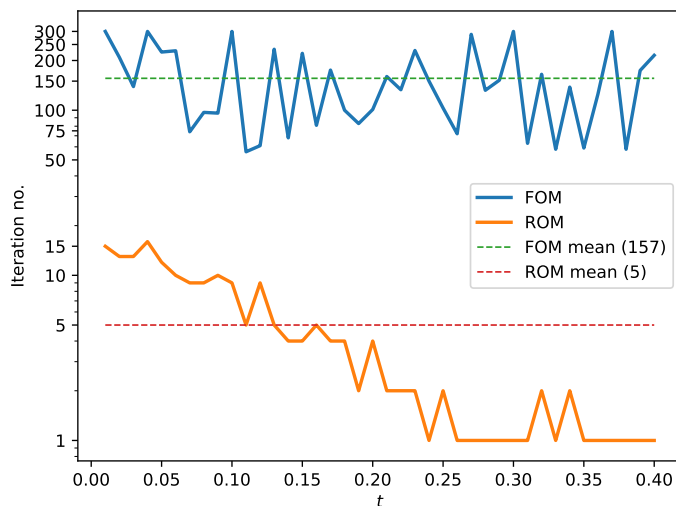| Parameter		POD modes						
ν	1	velocity u_1	15	pressure p_1	10	supremiser s_1	10
$\bar{U}$	3	velocity u_2	10	pressure p_2	10	supremiser s_2	10
		control g	5				

Table 4: Lid–driven cavity flow: Computational details of the online stage.

Figure 12: Lid–driven cavity flow at $\nu = 1$ and $\bar{U} = 3$: number of optimisation iterations of FOM and ROM solvers

DD–FOM and the ROM. In Table 4, we report the number of POD modes we use to obtain the ROM. The number of reduced bases is chosen so that the discarded energy for each of the components is less than 10^{-6}. As before, the ROM optimization is the same used in the FOM, but on a smaller space with dimension 5 instead of 294. As optimisation algorithm, we still use the L–FBGS–B, but, in this case, we use a smaller value for tol_{opt} of 10^{-6}. Figures 11 represent the DD–FOM solutions for $\bar{U} = 3$ at 3 different time instances, where we see a qualitative agreement with the monolithic solution in Figure 10b.

Again, in Figure 12 we observe that the number of optimization iterations for FOM is between 10 and 100 times larger than the ROM ones for the parameter $(\bar{U}, \nu) = (3, 1)$. Recalling that each iteration requires at least one computation of the state and the adjoint solvers, we obtain a great computational advantage. For the test with $\bar{U} = 3$, the average number of the iteration over all time steps in the case of the FOM solver is 170 while it is 24 in the case of the ROM solver. Additionally, each solver at the reduced level is of a much smaller dimension (see Table 4). As in the previous test case, we would
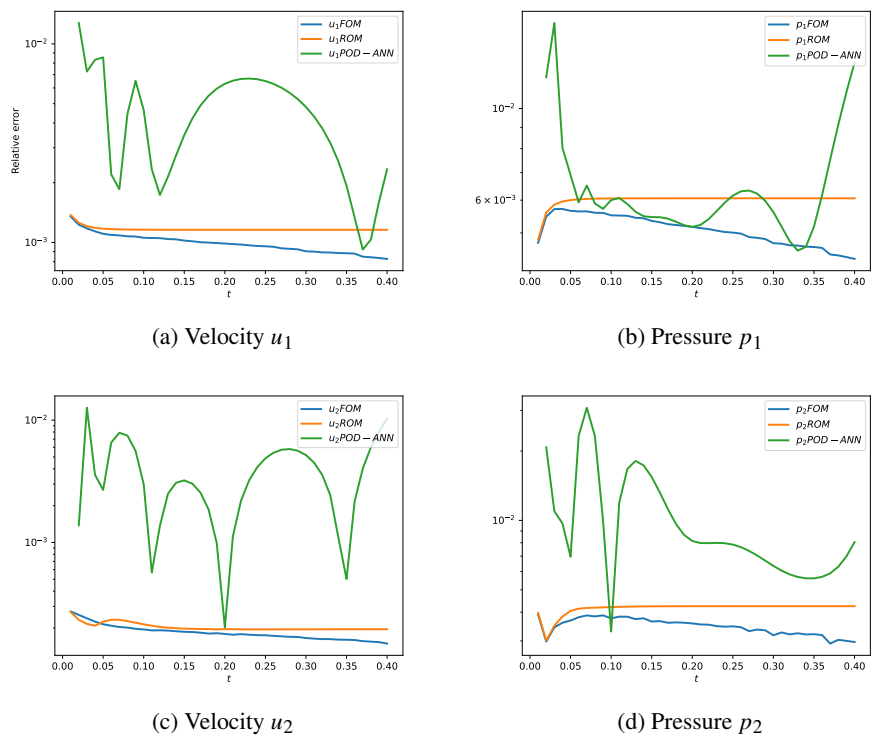
(a) Velocity u_1

(b) Pressure p_1

(c) Velocity u_2

(d) Pressure p_2

Figure 13: Lid–driven cavity flow at $\nu = 1$ and $\bar{U} = 3$: relative errors of FOM, ROM and POD–NN solutions w.r.t. the monolithic solution

24

like to provide a comparison of the full–order and the reduced–order models with non–intrusive POD–NN model. The architecture is still the one reported in Section 5.2 Again, the initial condition leads to a discontinuity in time at the starting timestep, hence, we exclude it from the training set in order to achieve better performance. Figure 13 shows the relative errors with respect to the monolithic solution for the FOM, ROMs and POD–NN model for the parameter $(\bar{U}, \nu) = (3, 1)$. As we can see, both FOM and ROM give us very good convergence results, i.e., the relative error does not exceed 1% in either case; but, in this case, also POD–NN gives quite good results, indeed, for each variable the relative error does not exceed 3%. Computational times for each method, FOM, ROM and POD–NN, are comparable with those of the backward–facing step flow, that is, one time step of the FOM takes between 15 and 45 minutes, one time step of the ROM (without hyper–reduction) takes on average 5 minutes, while a POD–NN prediction needs around 0.003 seconds.

Remark (Sensitivity to the domain splitting). Similarly to the stationary case analyzed in [43], we encounter the issue of choosing the optimal domain splitting. Also in the non–stationary lid–driven cavity flow case, we observe a significantly slower convergence when the domain is split by a vertical segment instead of a horizontal one.

6.3 Validation of the POD–NN method

As we have seen in the numerical experiments above, unlike the POD–Galerkin, the POD–NN ROM seems to be very sensitive to the presence of discontinuities in time. In this section, we would like to investigate under which circumstances the POD–NN technique is able to produce a good approximation by considering various scenarios and performing a statistical analysis in terms of physical parameters. The first necessary step in all the approaches below is to provide a larger set of snapshots for generating RB spaces and for the training artificial neural network, as is described in Section 5.2.

Regarding the backward–facing step test case presented in Section 6.1, we have seen that the constructed NN–ROM provides very poor approximations, most probably due to the presence of a discontinuity between the inlet boundary and the initial conditions. We restrict the analysis by considering only one physical parameter, $\bar{U}$ to generate new POD–based reduced spaces and to train the ANN. In this case, we draw 101 parameters uniformly sampled from the parameter space reported in Table 1, we randomly shuffle the parameter set and divide it into a training set $\mathbb{P}_{\text{train}}$, containing 75 parameters, and a testing set $\mathbb{P}_{\text{test}}$, composed of remaining 26 parameters.

We consider two scenarios: in the first one we construct the parameter space for POD compression and the ANN training by using the training parameter set and over all the time steps, i.e, the set $\mathcal{P}_{\text{train}}^{\text{full}} = \left\{ (\bar{U}, t^n) : \bar{U} \in \mathbb{P}_{\text{train}}, \ n \in \{1, ..., 100\} \right\}$ whereas in the other one we discard the first 20 time steps and consider only the training parameter set and the time steps t^n for $n \geq 21$, i.e., the set $\mathcal{P}_{\text{train}}^{\text{restricted}} = \left\{ (\bar{U}, t^n) : \bar{U} \in \mathbb{P}_{\text{train}}, \ n \in \{21, ..., 100\} \right\}$. We perform the POD compression using the snapshots corresponding to parameter values in $\mathcal{P}_{\text{train}}^{*}$ for $* \in \{\text{full}, \text{restricted}\}$ to generate the RB spaces in each case by retaining $50, 30, 35$ and 15 bases functions for components u_1, p_1, u_2 and p_2, respectively. Then, in both cases, the ANN with 5 hidden layers with $16, 64, 128, 128$ and 128 neurons, respectively, is constructed for

(a) Mean relative errors over the entire time interval $[0, 1]$

(b) Mean relative errors over the restricted time interval $[0.21, 1]$
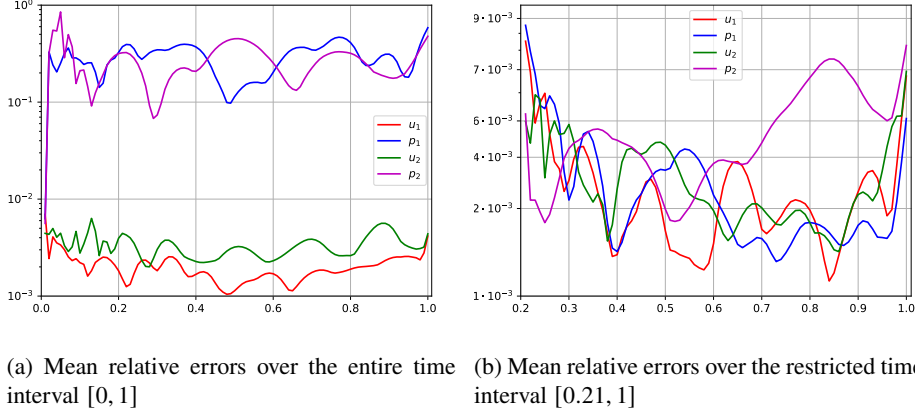
Figure 14: Mean over the parameter testing set of POD–NN relative errors for the discontinuous in time backward–facing step test case with respect to $\bar{U}$ over the entire (a) and restricted (b) time intervals

each state subcomponent, using the hyperbolic tangent as an activation function. We use the same sets of snapshots used for the RB spaces construction to train the ANN and then validate the performance of the POD–NN method on the testing set of 26 physical parameters as described above. Figures 14a and 14b show the relative errors of the POD–NN method in time, averaged over the training set of physical parameters, for each state subcomponent of the problem over the full and restricted time intervals, respectively. It can be easily seen that in both cases the POD–NN is able to provide very good approximation for the velocity fields u_1 an u_2. On the other hand, the time discontinuity at the initial time step pervades in time and compromises the POD–NN approximation for the pressure fields p_1 and p_2, which does not happen if we restrict the analysis over the time interval where the discontinuity vanishes.

Given that the POD–NN approach seems to be sensitive to the time discontinuity, as evidenced above, we perform another test case on the same domain where we have a smooth transition from the zero initial condition by considering the inlet boundary condition $u_{in}(x, y) = (w(y, t), 0)^T$ on Γ_{in}, where the function w is given by

$$w(y, t) = \begin{cases} \bar{U} \cdot \frac{1}{2} \left(1 - \cos\left(\frac{2\pi t}{0.4}\right)\right) \frac{4}{9}(y - 2)(5 - y), \ y \in [2, 5], \ t \leq 0.4, \\ \bar{U} \cdot \frac{4}{9}(y - 2)(5 - y), \ y \in [2, 5], \ t > 0.4. \end{cases}$$

As in the previous test case, we consider one physical parameter $\bar{U}$ to generate new POD–based reduced spaces and we train the ANN. Again, we consider 101 parameters, 75 in the training set and 26 in the test set, selected uniformly in the parameter space, see Table 1. The dimensions of the RB spaces and the architecture of the ANN are the same as in the discontinuous test case above. We again consider two scenarios by using use the sets $\mathcal{P}_{\text{train}}^{\text{full}}$ and $\mathcal{P}_{\text{train}}^{\text{restricted}}$ to perform POD compression and to train the ANN. Here, we discover another complication in the POD–NN approach: the sensitivity to

(a) Mean relative errors over the entire time interval $[0, 1]$

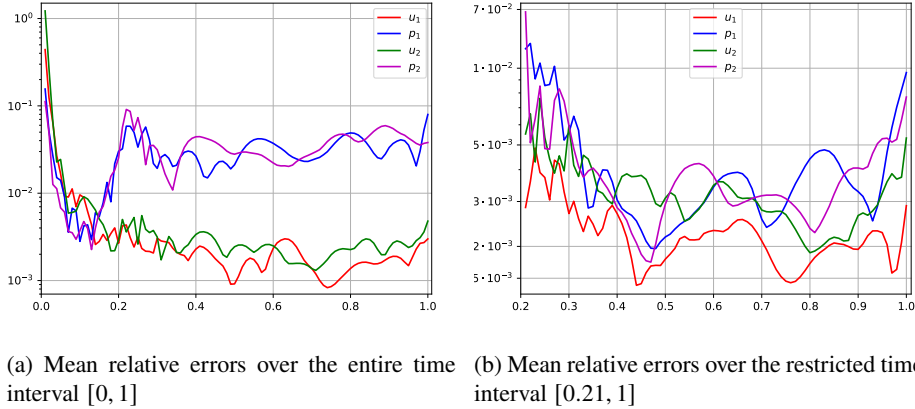(b) Mean relative errors over the restricted time interval $[0.21, 1]$

Figure 15: Mean over the parameter testing set of POD–NN relative errors for the continuous in time backward–facing step test case with respect to $\bar{U}$ over the entire (a) and restricted (b) time intervals

small data, i.e., the ANN struggles to predict the data at the initial time steps as the norm is very close to zero. In Figures 15a and 15b, we show the relative errors of the POD–NN method in time, averaged over the training set of physical parameters, for each state subcomponent of the problem over the full and restricted time intervals, respectively. The picture here is very similar to the one in the discontinuous case, i.e. the velocity fields u_1 and u_2 show very good approximation properties (except for the first few time steps where the relative errors are expected to be bigger since the solution norm is very close to zero) while the POD–NN method struggles to learn the pressure field over the whole time intervals while the situation is improved by restricting the analysis to the time interval where the velocity fields norms are big enough.

From the analysis carried out before, we can see that POD–NN approach provides excellent results as soon as we are far away enough from the "problematic" regions: either the regions that contain discontinuities or where the solution has a small norm. The error analysis reported in Figures 14 and 15 shows that the relative error does not exceed the 1%–threshold if we exclude from the POD–NN model set–up the mentioned "problematic" regions, where the solution can be computed by FOM or POD–Galerkin methods with more reliability.

Concerning the lid–driven cavity flow test case presented in 6.2, we were able to achieve a good approximation by NN–ROM method even in the presence of discontinuities, by merely extending the training set. As in the previous test cases, we draw 101 uniformly sampled parameters from the parameter space reported in Table 3, 75 for the training set and 26 for the testing set. We perform the POD compression using the training set to generate RB spaces by retaining $25, 15, 25$ and 15 bases functions for components u_1, p_1, u_2 and p_2, respectively. Then, the ANN with 5 hidden layers with $16, 64, 128, 128$ and 128 neurons, respectively, is constructed for each state subcomponent, using the hyperbolic tangent as an activation function. We use the same 75
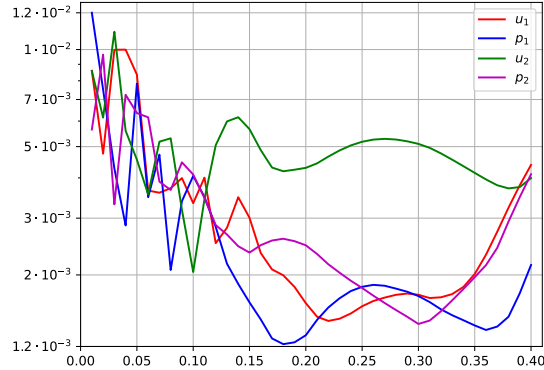
27

Figure 16: Mean over the parameter testing set of POD–NN relative errors for the discontinuous lid–driven cavity test case with respect to $\bar{U}$ over the time interval [0,0.4]

snapshots used for the RB spaces construction to train the ANN and then validate the performance of the POD–NN method on the testing set of 26 parameters as described above. In this case, the statistical analysis of the relative errors shows good behaviour on the entire time interval, as is evidenced in Figure 16.

The tests carried out in this section suggest the necessity to set up a proper sensitivity analysis with respect to model parameters and the problem geometry which will be the subject of future work.

7   Conclusions

In this work, we described and conducted the convergence analysis of an optimisation–based domain decomposition algorithm for nonstationary Navier–Stokes equations.

The original problem cast into the optimisation–based domain–decomposition framework leads to the optimal control problem aimed at minimising the coupling error at the interface; the problem, then, has been tackled using an iterative gradient–based optimisation algorithm, which allowed us to obtain a complete separation of the solvers on different subdomains.

At the reduced–order level, we provided two techniques: a POD–Galerkin projection and a data–driven POD–NN, both of them on separate domains. In the Galerkin projection, the optimal–control problem was solved much faster, not only because of the reduced dimensions but also because of the smaller number of iterations. In the POD–NN results are less accurate, but the computational time is way cheaper with respect to the other methods. Additional statistical analysis has been carried out to investigate the approximation properties of POD–NN methodology which confirms the sensitivity of the method to discontinuities in time and small data. In future works, we want to perform a sensitivity analysis of the method to the model parameters and the

problem geometry.

As has been mentioned in the paper, the aforementioned techniques could be promising for various areas of computational physics. First of all, these algorithms can be used when complex time–dependent problems arise and domain decomposition becomes necessary due to the number of degrees of freedom. We plan to extend this algorithm also to several subdomains for more complex engineering problems as similarly done in [58]. Moreover, in the context of multi–physics or fluid–structure interaction, the coupling of pre–existing solvers on each subcomponent can be exploited in this framework, with the additional benefit of the reduction for parametric problems, which guarantees high adherence with respect to the full order solutions. Finally, in case the codes are not directly accessible, the presented non–intrusive approach can be used to highly speed up the simulations while still obtaining meaningful results.

Acknowledgements

A Lifting Function and Velocity Supremiser Enrichment

The use of lifting functions is quite common in the reduced basis method (RBM) framework; see, for example, [8, 28]. It is motivated by the fact that, in the chosen model, we tackle a problem with non–homogeneous Dirichlet boundary conditions on the parts of the boundaries $\Gamma_{i,D}$, $i = 1, 2$. From the implementation point of view, this does not present any problem when dealing with the high–fidelity model, since there are several well–known techniques for non–homogeneous essential conditions, in particular at the algebraic level. However, these boundary conditions create some problems when dealing with the reduced basis methods. Indeed, we seek to generate a linear vector space that is obtained by the compression of the set of snapshots, and this clearly cannot be achieved by using snapshots that satisfy different Dirichlet conditions, as the resulting space would not be linear. This problem is solved by introducing a lifting function $l_{i,h}^n \in V_{i,h}$, for $i = 1, 2$, during the offline stage, such that $l_{i,h}^n = u_{i,D,h}^n$ on $\Gamma_{i,D}$. We define two new variables $u_{i,0,h}^n := u_{i,h}^n - l_{i,h}^n \in V_{i,0,h}$, for $i = 1, 2$, which satisfy the homogeneous condition $u_{i,0,h}^n = 0$ on $\Gamma_{i,D}$. So, they can be used to generate the reduced basis linear space. We remark that the lifting function is needed only in the domain where the Dirichlet boundary is non–empty, i.e., where $\Gamma_{i,D} \neq \emptyset$ for $i = 1, 2$. It is important to point out that the choice of lifting functions is not unique; in our work, we chose to use the solution of the incompressible Stokes problem in one of the domains Ω, Ω_1 or Ω_2 (depending on the particular model we are investigating) with the velocity equal to u_D on the corresponding parts of the boundaries and the homogeneous Neumann conditions analogous to the original problem setting.

Velocity supremiser enrichment is a very important technique to ensure that the reduced spaces are inf–sup stable in the context of saddle–point problems. We recall that each velocity snapshot, which is a solution to the incompressible Navier–Stokes equation, is divergence–free. Hence, the term $b_i(\cdot, \cdot)$ for $i = 1, 2$ applied to any pair of functions in the span of the snapshots will be zero. This does not allow us to fulfil the inf–sup condition of the type (26). For this reason, there is a need to enrich the reduced velocity spaces with extra functions, which are called supremisers, that will make the pairs of velocity–pressure reduced spaces inf–sup stable. The supremiser variables $s_{i,h}^n$, for $i = 1, 2$, are defined as the solution to the following problem: find $s_{i,h}^n \in V_{i,0,h}$ such that

$$\left(\nabla v_{i,h}, \nabla s_{i,h}^n\right) = b_{i,h}\left(v_{i,h}, p_{i,h}^n\right) \quad \forall v_{i,h} \in V_{i,0,h}, \tag{35}$$

where $p_{i,h}^n$, for $i = 1, 2$, are the finite–element pressure solutions of the Navier–Stokes problem and the left–hand side is the scalar product that defines the norm with which the variational spaces $V_{i,0,h}$ are endowed. For more details, we refer to [8, 20].

B Reduced Basis Generation

Once we obtain the homogenised snapshots $u_{i,0,h}$ as described in Appendix A and the velocity supremisers $s_{i,h}$ for $i = 1, 2$, we are ready to construct a set of reduced basis functions. A very common choice when dealing with Navier–Stokes equations is to use

the POD technique; see, for instance, [28]. In order to implement this technique, we will need two main ingredients: the matrices of the inner products and the snapshot matrices, obtained by a full–order model (FOM) discretization as the one presented in the previous sections. First, we define the basis functions for the FE element spaces used in the weak formulation (27), (28) and (30): $V_{i,0,h} = \text{span}\{\phi_j^{u_i}\}_{j=1}^{\mathcal{N}_h^{u_i}}$, $Q_{i,h} = \text{span}\{\phi_j^{p_i}\}_{j=1}^{\mathcal{N}_h^{p_i}}$ and $X_h = \text{span}\{\phi_j^g\}_{j=1}^{\mathcal{N}_h^g}$, where $\mathcal{N}_h^*$, for $* \in \{u_1, p_1, u_2, p_2, g\}$, denotes the dimension of the corresponding FE space.

We proceed by building the snapshot matrices. First, we sample the parameter space and draw a discrete set of K parameter values. Then, the snapshots, i.e., the high–fidelity FE solutions at each parameter value in the sampling set and at each time–step $t_1, ..., t_M$, are collected into snapshot matrices $\mathcal{S}_{u_i} \in \mathbb{R}^{\mathcal{N}_h^{u_i} \times MK}$, $\mathcal{S}_{s_i} \in \mathbb{R}^{\mathcal{N}_h^{u_i} \times MK}$, $\mathcal{S}_{p_i} \in \mathbb{R}^{\mathcal{N}_h^{p_i} \times MK}$, for $i = 1, 2$ and $\mathcal{S}_g \in \mathbb{R}^{\mathcal{N}_h^g \times MK}$ for the corresponding values.

The next step is to define the inner–product matrices X_{u_i}, X_{p_i}, for $i = 1, 2$, and X_g:

$$(X_{s_i})_{jk} = (X_{u_i})_{jk} = \left(\nabla \phi_k^{u_i}, \nabla \phi_j^{u_i}\right)_{\Omega_i}, \qquad \text{for } j, k = 1, ..., \mathcal{N}_h^{u_i}, \ i = 1, 2,$$

$$(X_{p_i})_{jk} = \left(\phi_k^{p_i}, \phi_j^{p_i}\right)_{\Omega_i}, \qquad \text{for } j, k = 1, ..., \mathcal{N}_h^{p_i}, \ i = 1, 2,$$

$$(X_g)_{jk} = \left(\phi_k^g, \phi_j^g\right)_{\Gamma_0}, \qquad \text{for } j, k = 1, ..., \mathcal{N}_h^g.$$

We are now ready to introduce the correlation matrices C_{u_i}, C_{s_i}, C_{p_i} for $i = 1, 2$ and C_g, all of dimension $MK \times MK$, as:

$$C_* := \mathcal{S}_*^T X_* \mathcal{S}_*$$

for every $* \in \{u_1, p_1, u_2, p_2, s_1, s_2, g\}$.

Once we have built the correlation matrices, we are able to carry out a POD compression on the sets of snapshots. This can be achieved by solving the following eigenvalue problems:

$$C_* Q_* = Q_* \Lambda_* \tag{36}$$

where $* \in \{u_1, s_1, p_1, u_2, s_2, p_2, g\}$, Q_* is the eigenvectors matrix and Λ_* is the diagonal eigenvalues matrix with eigenvalues ordered by decreasing order of their magnitude. The k–th reduced basis function for the component $*$ is then obtained by applying the matrix $\mathcal{S}_*$ to $\underline{v}_k^*$, the k–th column vector of the matrix Q_*:

$$\Phi_k^* := \frac{1}{\sqrt{\lambda_k^*}} \mathcal{S}_* \underline{v}_k^*,$$

where λ_k^* is the k–th eigenvalue from (36). Therefore, we are able to form the set of reduced basis as

$$\mathcal{A}^* := \left\{\Phi_1^*, ..., \Phi_{N_*}^*\right\},$$

where the integer numbers N_* indicate the number of the basis functions used for each component for $* \in \{u_1, p_1, u_2, p_2, s_1, s_2, g\}$. Now, it is time to include the supremiser

enrichment of the velocities spaces discussed at the beginning of this section. We provide the following renumbering of the functions for further simplicity:

$$\Phi^{u_i}_{N_{u_i}+k} := \Phi^{s_i}_k, \quad \text{for } k = 1, ..., N_{s_i}, \ i = 1, 2,$$

and we redefine $N_{u_i} := N_{u_i} + N_{s_i}$, and new basis functions sets

$$\mathcal{A}^{u_i} := \left\{ \Phi^{u_i}_1, ..., \Phi^{u_i}_{N_{u_i}} \right\},$$

for $i = 1, 2$ and these new sets are now including extra basis functions obtained from the corresponding supremiser. Finally, we introduce three separate reduced basis spaces – for the state and the control variables, respectively:

$$V^*_{rb} = \text{span}(\mathcal{A}^*), \qquad \dim(V^*_{rb}) = N_*,$$

for $* \in \{u_1, p_1, u_2, p_2, g\}$. Now, due to the supremiser enrichment the spaces $V^{u_i}_{rb}$ and $V^{p_i}_{rb}$ are inf–sup stable in the sense (26) for $i = 1, 2$; the proof can be found in [8].

References

[1] FEniCS - the FEniCSx computing platform, https://fenicsproject.org/, 2019.

[2] RBniCS - reduced order modelling in FEniCS, http://mathlab.sissa.it/rbnics, 2015.

[3] EZyRB - Easy Reduced Basis method, https://mathlab.sissa.it/ezyrb, 2015.

[4] S. Ali, F. Ballarin, and G. Rozza. Stabilized reduced basis methods for parametrized steady Stokes and Navier–Stokes equations. *Computers & Mathematics with Applications*, 80(11):2399–2416, 2020. High-Order Finite Element and Isogeometric Methods 2019.

[5] N. Andrei. Acceleration of conjugate gradient algorithms for unconstrained optimization. *Applied Mathematics and Computation*, 213(2):361–369, 2009.

[6] M. Astorino, F. Chouly, and M. A. Fernández. Robin Based Semi-Implicit Coupling in Fluid-Structure Interaction: Stability Analysis and Numerics. *SIAM Journal on Scientific Computing*, 31(6):4041–4065, 2010.

[7] F. Bai and Y. Wang. A reduced order modeling method based on GNAT-embedded hybrid snapshot simulation. *Mathematics and Computers in Simulation*, 199:100–132, 2022.

[8] F. Ballarin, A. Manzoni, A. Quarteroni, and G. Rozza. Supremizer stabilization of POD–Galerkin approximation of parametrized steady incompressible Navier–Stokes equations. *International Journal for Numerical Methods in Engineering*, 102(5):1136–1161, 2015.

[9] F. Ballarin and G. Rozza. POD–Galerkin monolithic reduced order models for parametrized fluid-structure interaction problems. *International Journal for Numerical Methods in Fluids*, 82(12):1010–1034, 2016.

[10] F. Ballarin, G. Rozza, and Y. Maday. Reduced-order semi-implicit schemes for fluid-structure interaction problems. In *Model Reduction of Parametrized Systems*, pages 149–167. Springer, 2017.

[11] P. Benner, M. Ohlberger, A. Cohen, and K. Willcox. *Model reduction and approximation: theory and algorithms*. SIAM, 2017.

[12] C.-H. Bruneau and P. Fabrie. New efficient boundary conditions for incompressible Navier-Stokes equations: a well-posedness result. *M2AN - Modélisation mathématique et analyse numérique*, 30(7):815–840, 1996.

[13] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on scientific computing*, 16(5):1190–1208, 1995.

[14] G. Carere, M. Strazzullo, F. Ballarin, G. Rozza, and R. Stevenson. A weighted POD-reduction approach for parametrized PDE-constrained Optimal Control Problems with random inputs and applications to environmental sciences. *Computers & Mathematics with Applications*, 102:261–276, 2021.

[15] R. Crisovan, D. Torlo, R. Abgrall, and S. Tokareva. Model order reduction for parametrized nonlinear hyperbolic problems as an application to uncertainty quantification. *Journal of Computational and Applied Mathematics*, 348:466–489, 2019.

[16] Y.-H. Dai. Convergence properties of the bfgs algoritm. *SIAM Journal on Optimization*, 13(3):693–701, 2002.

[17] S. Deparis and G. Rozza. Reduced basis method for multi-parameter-dependent steady Navier-Stokes equations: Applications to natural convection in a cavity. *Journal of Computational Physics*, 228(12):4359–4378, 2009.

[18] V. Ervin, E. Jenkins, and H. Lee. Approximation of the Stokes–Darcy system by optimization. *Journal of Scientific Computing*, 59, 06 2014.

[19] L. Evans. *Partial Differential Equations*. AMS, 1997.

[20] A.-L. Gerner and K. Veroy. Certified reduced basis methods for parametrized saddle point problems. *SIAM Journal on Scientific Computing*, 34(5):A2812–A2836, 2012.

[21] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

[22] P. Gosselet, V. Chiaruttini, C. Rey, and F. Feyel. A monolithic strategy based on an hybrid domain decomposition method for multiphysic problems. application to poroelasticity. *Revue Européenne des Éléments Finis*, 13, 04 2012.

[23] M. Gunzburger and H. K. Lee. A domain decomposition method for optimization problems for partial differential equations. *Computers & Mathematics with Applications*, 40(2):177–192, 2000.

[24] M. Gunzburger, J. Peterson, and H. Kwon. An optimization based domain decomposition method for partial differential equations. *Computers & Mathematics with Applications*, 37(10):77–93, 1999.

[25] M. D. Gunzburger and H. K. Lee. An optimization-based domain decomposition method for the Navier–Stokes equations. *SIAM Journal on Numerical Analysis*, 37(5):1455–1480, 2000.

[26] B. Haasdonk. *Reduced Basis Methods for Parametrized PDEs—A Tutorial Introduction for Stationary and Instationary Problems*, chapter 2, pages 65–136. Society for Industrial and Applied Mathematics, 2017.

[27] E. Hairer, S. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*. Springer-Verlag, Berlin, 1987.

[28] J. S. Hesthaven, G. Rozza, and B. Stamm. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. Springer Briefs in Mathematics. Springer, Switzerland, 1 edition, 2015.

[29] J. S. Hesthaven and S. Ubbiali. Non-intrusive reduced order modeling of nonlinear problems using neural networks. *Journal of Computational Physics*, 363:55–78, 2018.

[30] T. T. P. Hoang and H. Lee. A global-in-time domain decomposition method for the coupled nonlinear Stokes and Darcy flows. *Journal of Scientific Computing*, 87, 04 2021.

[31] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[32] P. Kuberry and H. K. Lee. A decoupling algorithm for fluid-structure interaction problems based on optimization. *Computer Methods in Applied Mechanics and Engineering*, 267:594–605, 2013.

[33] P. Kuberry and H. K. Lee. Analysis of a fluid-structure interaction problem recast in an optimal control setting. *SIAM Journal on Numerical Analysis*, 53(3):1464–1487, 2015.

[34] J. E. Lagnese, G. Leugering, and G. Leugering. *Domain in Decomposition Methods in Optimal Control of Partial Differential Equations*. Number 148 in International Series of Numerical Mathematics. Springer Science & Business Media, 2004.

[35] T. Lassila, A. Manzoni, A. Quarteroni, and G. Rozza. Model order reduction in fluid dynamics: challenges and perspectives. In A. Quarteroni and G. Rozza, editors, *Reduced Order Methods for Modeling and Computational Reduction*, volume 9, pages 235–274. Springer MS&A Series, 2014.

[36] J. L. Lions and E. Magenes. *Non-Homogeneous Boundary Value Problems and Applications*. Springer, 1972.

[37] J. L. Nazareth. Conjugate gradient method. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1(3):348–353, 2009.

[38] M. Nonino, F. Ballarin, and G. Rozza. A monolithic and a partitioned reduced basis method for fluid–structure interaction problems. *Fluids*, 6(6), 2021.

[39] M. Nonino, F. Ballarin, G. Rozza, and Y. Maday. Projection based semi–implicit partitioned reduced basis method for non parametrized and parametrized Fluid–Structure Interaction problems. *Journal of Scientific Computing*, 94(4), 2022.

[40] D. Papapicco, N. Demo, M. Girfoglio, G. Stabile, and G. Rozza. The Neural Network shifted-proper orthogonal decomposition: A machine learning approach for non-linear reduction of hyperbolic equations. *Computer Methods in Applied Mechanics and Engineering*, 392:114687, 2022.

[41] F. Pichi, F. Ballarin, G. Rozza, and J. S. Hesthaven. Artificial neural network for bifurcating phenomena modelled by nonlinear parametrized PDEs. *PAMM*, 20(S1):e202000350, 2021.

[42] F. Pichi, M. Strazzullo, F. Ballarin, and G. Rozza. Driving bifurcating parametrized nonlinear PDEs by optimal control strategies: application to Navier–Stokes equations with model order reduction. *ESAIM: Mathematical Modelling and Numerical Analysis*, 56(4):1361–1400, 2022.

[43] I. Prusak, M. Nonino, D. Torlo, F. Ballarin, and G. Rozza. An optimisation–based domain–decomposition reduced order model for the incompressible Navier-Stokes equations. *Computers & Mathematics with Applications*, 151:172–189, 2023.

[44] I. Prusak, D. Torlo, M. Nonino, and G. Rozza. Optimisation–based coupling of finite element model and reduced order model for computational fluid dynamics. *arXiv preprint arXiv:2402.10570*, 2024.

[45] A. Quarteroni and A. Valli. *Numerical Approximation of Partial Differential Equations*. Springer Series in Computational Mathematics. 23. Springer Berlin Heidelberg, Heidelberg, DE, 1994. Written for: Numerical analysts, applied mathematicians.

[46] A. Quarteroni and A. Valli. *Domain decomposition Methods for Partial Differential Equations*. Oxford University Press, Oxford, UK, 1999.

[47] T. Richter. *Fluid-structure interactions*. Lecture Notes in Computational Science and Engineering. Springer International Publishing, Basel, Switzerland, 1 edition, Aug. 2017.

[48] F. Romor, G. Stabile, and G. Rozza. Non-linear manifold reduced-order models with convolutional autoencoders and reduced over-collocation method. *Journal of Scientific Computing*, 94(3):74, 2023.

[49] G. Rozza. Reduced basis methods for stokes equations in domains with non-affine parameter dependence. *Computing and Visualization in Science*, 12:23–35, 2009.

[50] G. Rozza, D. Huynh, and A. Patera. Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations. *Archives of Computational Methods in Engineering*, 15:1–47, 09 2007.

[51] P. Siena, M. Girfoglio, and G. Rozza. Fast and accurate numerical simulations for the study of coronary artery bypass grafts by artificial neural networks. In *Reduced Order Models for the Biomechanics of Living Organs*, pages 167–183. Elsevier, 2023.

[52] G. Stabile, F. Ballarin, G. Zuccarino, and G. Rozza. A reduced order variational multiscale approach for turbulent flows. *Advances in Computational Mathematics*, 45(5):2349–2368, 2019.

[53] G. Stabile and G. Rozza. Finite volume POD-Galerkin stabilised reduced order methods for the parametrised incompressible Navier–Stokes equations. *Computers & Fluids*, 173:273–284, 2018.

[54] M. Strazzullo, F. Ballarin, R. Mosetti, and G. Rozza. Model reduction for parametrized optimal control problems in environmental marine sciences and engineering. *SIAM Journal on Scientific Computing*, 40(4):B1055–B1079, 2018.

[55] M. Strazzullo, F. Ballarin, and G. Rozza. POD–Galerkin model order reduction for parametrized time dependent linear quadratic optimal control problems in saddle point formulation. *Journal of Scientific Computing*, 83(3):1–35, 2020.

[56] M. Strazzullo, F. Ballarin, and G. Rozza. POD-Galerkin model order reduction for parametrized nonlinear time-dependent optimal flow control: an application to shallow water equations. *Journal of Numerical Mathematics*, 30(1):63–84, 2022.

[57] M. Strazzullo, M. Girfoglio, F. Ballarin, T. Iliescu, and G. Rozza. Consistency of the full and reduced order models for evolve-filter-relax regularization of convection-dominated, marginally-resolved flows. *International Journal for Numerical Methods in Engineering*, 123(14):3148–3178, 2022.

[58] T. Taddei, X. Xu, and L. Zhang. A non-overlapping optimization-based domain decomposition approach to component-based model reduction of incompressible flows. *arXiv preprint arXiv:2310.20267*, 2023.

[59] M. Tezzele, N. Demo, G. Stabile, A. Mola, and G. Rozza. Enhancing CFD predictions in shape design problems by model and parameter space reduction. *Advanced Modeling and Simulation in Engineering Sciences*, 7(1):1–19, 2020.

[60] D. Torlo, F. Ballarin, and G. Rozza. Stabilized weighted reduced basis methods for parametrized advection dominated problems with random inputs. *SIAM/ASA Journal on Uncertainty Quantification*, 6(4):1475–1502, 2018.

[61] D. Torlo and M. Ricchiuto. Model order reduction strategies for weakly dispersive waves. *Mathematics and Computers in Simulation*, 205:997–1028, 2023.

[62] L. Venturi, F. Ballarin, and G. Rozza. A weighted POD method for elliptic PDEs with random inputs. *Journal of Scientific Computing*, 81(1):136–153, 2019.

[63] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature methods*, 17(3):261–272, 2020.

[64] Y.-x. Yuan. A modified bfgs algorithm for unconstrained optimization. *IMA Journal of Numerical Analysis*, 11(3):325–332, 1991.