SISSA
Scuola
Internazionale
Superiore di
Studi Avanzati

Mathematics Area - PhD course in
Mathematical Analysis, Modelling, and Applications

Application of optimisation–based domain–decomposition reduced order models to parameter–dependent fluid dynamics and multiphysics problems

Candidate:
Ivan Prusak

Advisor:
Prof. Gianluigi Rozza
Co-advisors:
Dr. Monica Nonino
Dr. Davide Torlo

Academic Year 2022-23

To all those who stand in the defence of Ukraine and to my family

Abstract

With the increase in the potential of high–performance computing in the last years, there is an immense necessity for numerical methods and approximation techniques that can perform real–time simulations of Partial Differential Equations (PDEs). The applications range from naval to aeronautical, to biomedical engineering ones, to name a few. There exist many techniques that might help in achieving such a goal, among which reduced–order models (ROMs) and domain–decomposition (DD) algorithms, namely those employed in this manuscript.

The DD methodology is a very 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 must render a certain degree of continuity among these subdomains. These methods are extremely important for multiphysics 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.

Model–order reduction methods are another set of methods mentioned before, which 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. ROMs are extremely effective thanks 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.

This thesis aims to introduce a framework where both aforementioned techniques, namely DD algorithms and ROMs are combined in order to achieve better performance of numerical simulations. We choose to model the DD using an optimisation approach to ensure the coupling of the interface conditions among subdomains. Starting from the domain decomposition approach, we derive an optimal control problem, for which we present the convergence analysis. The snapshots for the high–fidelity model are obtained with the Finite Element discretisation, and the model order reduction is then proposed both in terms of time and/or physical parameters, with a standard Proper Orthogonal Decomposition (POD)–Galerkin projection or with non–intrusive methods, such as

POD–neural network (NN). The methodology has been tested on a couple of Computational Fluid Dynamics (CFD) benchmark problems.

The final aim of the thesis is to produce a fully–segregated method for multiphysics problems using the aforementioned techniques. We have managed successfully to build a model for a non–stationary Fluid–Structure Interaction (FSI) problem. The resulting numerical method shows an extremely important feature — it is stable under the assumption of the "added mass" effect, which causes instabilities of many partitioned approaches to FSI problems. It has been evidenced by the numerical experiments of the model presented for a two—dimensional haemodynamics benchmark FSI problem.

Contents

List of Figures

List of Tables

Introduction

Partial differential equations (PDEs) lie at the foundation of many mathematical models for real–life applications. The applications of such models vary from naval [176, 174, 47] and aeronautical [110, 154] to cardiovascular engineering [173, 147, 121], to name a few. In practice, for most of the problems, there is no way to find an analytical closed solution, which forces one to resort to surrogate models for different phenomena, namely numerical methods such as Finite Element (FE), Finite Volume (FV), Discontinuous Galerkin (DG) or other methods.

The FE method is among the most used discretisation techniques for engineering design and analysis. It is based on an efficient tessellation of the domain of interest and the introduction of a finite number of basis functions with local supports. The span of these basis functions is supposed to provide a good approximation of the continuous space where the original problem is defined. The assumption that the basis functions have local supports leads to sparse algebraic systems that can be efficiently tackled by well–developed fields of linear and non–linear algebra.

Even though modern computer architectures make it possible to solve a wide range of problems, there still exists a class of problems of large size that needs substantial attention. In particular, cutting–edge research has put its effort in improving existing numerical algorithms as well as in developing new ones, that are able to tackle those problems or provide computationally cheap real–time computations. Two of the most known techniques devised to address these problems are Domain Decomposition (DD) algorithms and Reduced Order Models (ROMs).

The DD methodology is a very efficient tool in the framework of PDEs. Any DD algorithm [151, 179, 40, 50, 165], is based on an effective splitting of the domain of interest into different subdomains (overlapping [27, 30, 62, 151, 179] or not [162, 166, 151, 179]). The original problem is then restricted to each of these subdomains, with the addition of 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 must render a certain degree of continuity among these subdomains. Then, the interface coupling conditions may be fulfilled by iterative procedures among subdomains, which can be solved independently, exploiting the efficiency of multiprocessor computer architectures. These methods are extremely important for multiphysics 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, [54, 69, 84, 98, 99, 103].

On the other hand, model–order reduction (MOR) methods [44, 23, 24, 78, 81, 132, 158, 25]

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. ROMs are incredibly effective thanks 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 phase. ROMs have been successfully applied over the years in different fields such as fluid dynamics [8, 45, 49, 105, 123, 157, 90, 33, 167, 168, 170, 175, 66, 178, 155], structural mechanics [78, 177, 181, 91, 28] and fluid–structure interaction problems [13, 18, 19, 127, 126]. Lately, there have also been great advances in reduced–order modelling for optimal–control problems, [36, 139, 172, 169, 171].

With these ideas in mind, the goal of the thesis is to give an extensive overview of the application of the domain–decomposition reduced–order models in the context of the parameter–dependent Fluid Dynamics and Fluid–Structure Interaction (FSI) problems. We bring our attention to domain–decomposition methods using an optimisation approach to ensure the coupling of the interface conditions between different subdomains and using a complete separation of the solvers on the subdomains. In particular, in order to exploit both aforementioned techniques, in this thesis the optimisation–based domain–decomposition algorithm is combined with projection–based reduced–order models.

As mentioned above, one of the problems addressed in this thesis is FSI. Even though there has been extensive research in computational methods for solving FSI problems in the last decades, a comprehensive presentation from a mathematical point of view is still missing nowadays: one of the reasons for this is the fact that the two subproblems, namely the Navier–Stokes equation and the elastic solid equation, are two big mathematical challenges on their own, see for example [153, 70]. FSI problems describe the dynamic interplay between a fluid and a solid. This interplay takes place because of the coupling of the two different physics at the FSI interface, namely the part of the physical domain that is common to the fluid subdomain and the solid subdomain. The FSI interface profile is unknown *a priori* and depends on the dynamics of the fluid and the structure problem.

Classically, there are two different approaches to solving an FSI problem: a partitioned (or segregated) procedure [127, 38, 19, 15, 39, 57, 58, 153, 14], and a monolithic procedure [18, 184, 153]. The idea behind partitioned algorithms is to try to combine available well–developed computational tools for fluid dynamics and structural dynamics and couple them with some iterative procedure. On the other hand, in monolithic algorithms, the fluid and the solid problem are solved simultaneously.

Even though the partitioned approach is more attractive because of its computational efficiency, it might lead to unstable algorithms, under some physical and geometrical conditions, while the monolithic approach does not suffer from this issue. This happens, for example, if the physical domain has a slender shape, or, as in our numerical test, if the fluid density is close to the solid density, and this is usually the case in haemodynamics applications [144], where the density of the blood is quite close to the density of the walls of the vessel. This phenomenon is the so–called "added mass" effect; see, for instance, [38] for a detailed derivation of the "added mass" effect and related consequences. The approach undertaken in this thesis leads to a fully–segregated algorithm

which, as will be evidenced by numerical experiments, is nevertheless stable under the "added mass" effect.

Thesis overview

This thesis is structured as follows. The first two chapters are devoted to an overview of two key techniques used throughout the thesis, namely domain decomposition algorithms and reduced order models, whereas, the rest of the thesis describes three main applications: fluid dynamics problem in both stationary and non–stationary regimes, and an FSI problem.

In Section 1.1, we give the basic motivations for DD methods. Section 1.2 describes the general mathematical setting of any DD method, and Section 1.3 lists a few classical DD algorithms. In Section 2.1, we provide a quick motivation and historical background for ROMs. Section 2.2 sets up a general mathematical framework for ROMs. Sections 2.3 and 2.4 describe the methods for reduced bases generation and efficient offline–online decomposition, respectively.

In Section 3.1, we introduce the monolithic and the optimisation–based domain–decomposition formulations of the stationary incompressible Navier–Stokes equations in both strong and weak forms. In Section 3.2, we derive the optimality condition for the resulting optimal control problem together with the expression for the gradient of the objective functional and we introduce a gradient–based optimisation algorithm to tackle the optimisation problem. In section 3.3, we describe the Finite Element discretisation of the problem of interest and provide a finite–dimensional high–fidelity optimisation problem. Section 3.4 deals with the reduced–order model that is based on a reduced basis generation by Proper Orthogonal Decomposition methodology and the Galerkin projection of the high–fidelity problem onto the lower–dimensional reduced spaces. In Section 3.5, we show some numerical results for two toy problems: the backward–facing step and the lid–driven cavity flows. Conclusions and perspectives follow in Section 3.6.

In Section 4.1, we introduce the monolithic formulation of the non–stationary fluid dynamics problem and its time–discretisation scheme with the further derivation of the optimisation–based domain–decomposition formulation at each time step in both strong and weak forms. In Section 4.2, 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 4.3 we derive the optimality condition for the resulting optimal control problem and the expression for the gradient of the objective functional with the following listing of the gradient–based optimisation algorithm. Section 4.4 contains the Finite Element discretisation of the problem of interest and the corresponding finite–dimensional high–fidelity optimisation problem. Section 4.5 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 4.6, we show some numerical results for two toy problems: the backward–facing step and the lid–driven cavity flows. Conclusions and perspectives follow in Section 4.7.

In Section 5.1, we describe the general dynamics of an FSI problem, we introduce the notion of

Arbitrary Lagrangian–Eulerian map and we provide a monolithic formulation for the FSI problem in the reference configuration. In Section 5.2, starting with the introduction of the variational formulation in space, we provide a time discretisation scheme for the problem of interest and we derive a DD optimal control problem. In Section 5.3, we undertake two approaches to tackle the optimisation–problem obtained in the previous section. The first approach is obtained by deriving the optimality condition for the optimal control problem and the expression for the gradient of the objective functional with the following description of the gradient–based optimisation algorithm. In the second approach, we recast the objective functional in the framework of non–linear least squares optimisation and describe an efficient Gauss–Newton algorithm to tackle the problem. Section 5.4 contains the Finite Element discretisation of the problem of interest and the corresponding finite–dimensional high–fidelity optimisation problem for both settings described in the previous section. Section 5.5 deals with the reduced–order model based on a reduced basis generation by Proper Orthogonal Decomposition methodology and on the Galerkin projection of the high–fidelity problem onto the lower–dimensional reduced spaces. In Section 5.6, we conduct some numerical results on the two–dimensional haemodynamics benchmark FSI problem. Conclusions and perspectives follow in Section 5.7.

Chapter 6 summarises the achieved goals and lists the future perspectives of the work carried out throughout this thesis.

Chapter 1

An Overview of Domain Decomposition Methods

In this section, we will give motivation for Domain Decomposition (DD) methods, provide a general mathematical setting and list a few classical techniques to address the problem.

1.1 Motivation

Very often numerical solvers of differential problems of interest lead to large–scale algebraic systems that are very hard (or even sometimes impossible) to tackle due to limited computational resources. Therefore, there is a need to improve existing numerical algorithms and develop new, more efficient ones.

The first ideas for DD methods date back to the eighties and they are one of the most important tools for devising parallel algorithms for large–scale problems arising in different areas such as physics, biomedicine and engineering.

Any domain decomposition method [151, 179, 40, 50, 165] is based on the assumption that the domain of interest is partitioned into subdomains, overlapping [27, 30, 62, 151, 179] or not [162, 166, 151, 179]; the original problem is then reformulated in each subdomain, and each subproblem is then coupled through the values of the unknown solutions at the subdomain interfaces. Then, the interface coupling conditions may be removed by introducing iterative procedures among subdomains, which can be solved independently, and, thus, benefit from the efficiency of multiprocessor computer architectures.

DD algorithms have been successfully applied with different numerical schemes, such as Finite Element method (FEM) [7, 124, 151, 179], Finite Volume (FV) method [26, 131], spectral methods [146, 187, 179], Discontinuous Galerkin (DG) methods [12, 56, 65] and others.

These methods are extremely important for multiphysics 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, [54, 69, 84, 98, 99, 103].

1.2 General setting

Let us first introduce the general notation and ideas of DD. We assume $\Omega \subset \mathbb{R}^d$ to be an d–dimensional domain with a Lipschitz boundary Γ, $\mathcal{L}$ is a second–order differential operator and f a given data on Ω. We consider the following problem: find $u : \Omega \to \mathbb{R}$ such that

$$\mathcal{L}u = f \quad \text{in} \quad \Omega, \tag{1.1a}$$
$$u = 0 \quad \text{on} \quad \Gamma. \tag{1.1b}$$

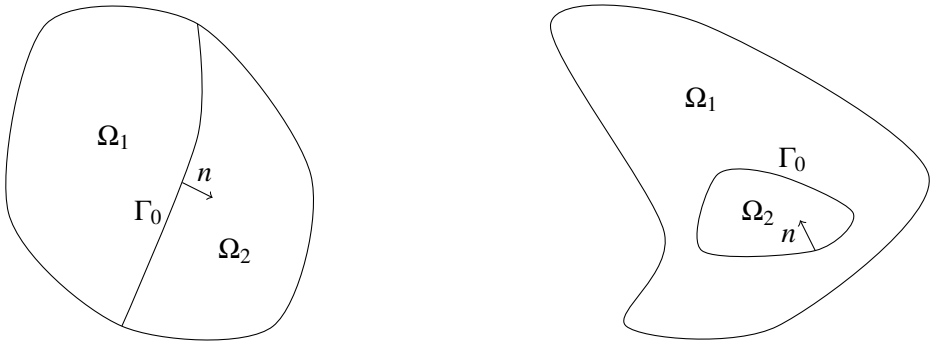


Figure 1.1: Non–overlapping partition of the domain Ω into two subdomains

To keep the exposition simple, we will consider in the rest of the work the two–domain non–overlapping methods: for more details and generalisations we refer to [54, 69, 74, 75, 77, 84, 98, 99, 103, 150, 151, 179]. To begin with, we assume that the domain Ω is partitioned into two non–overlapping subdomains Ω_1 and Ω_2, and we denote by Γ_0 the interface between them: $\Gamma_0 := \overline{\Omega_1} \cap \overline{\Omega_2}$ (see Figure 1.1); we assume that Γ_0 is a Lipschitz $(d-1)$–dimensional manifold. We denote by $\Gamma_i = \Gamma \cap \overline{\Omega_i}$ for $i = 1, 2$.

We denote by u_i the restriction to subdomain Ω_i, $i = 1, 2$, of the solution u to (1.1) and by $\mathbf{n}_i$ the normal outward unit vector on $\partial\Omega_i \cap \Gamma_0$. For the sake of the notation, we also set $\mathbf{n} := \mathbf{n}_1$. Then, the domain decomposition problem reads: for $i = 1, 2$, solve

$$\mathcal{L}u_i = f \quad \text{in} \quad \Omega_i, \tag{1.2a}$$
$$u_i = 0 \quad \text{on} \quad \Gamma_i. \tag{1.2b}$$

To guarantee the equivalence with (1.1), we need to enforce interface conditions between u_1 and u_2 across Γ_0. Such condition can be expressed by two relationships:

$$\Phi(u_1) = \Phi(u_2) \quad \text{on} \quad \Gamma_0, \tag{1.3a}$$
$$\Psi(u_1) = \Psi(u_2) \quad \text{on} \quad \Gamma_0, \tag{1.3b}$$

where the functions Φ and Ψ depend upon the nature of the problem at hand. In general, these interface conditions are being chosen in such a way that the following two requirements are satisfied:

- The monolithic solution $u : \Omega \to \mathbb{R}$ belongs to some space of functions defined on the domain Ω. This, in turn, implies that the restriction u_i has a certain degree of regularity on the corresponding domain Ω_i for $i = 1, 2$ and, moreover, they satisfy a suitable matching condition on the interface Γ_0.

- The restriction u_i, for $i = 1, 2$, is a distributional solution of problem (1.2). Another interface condition arises from the fact that the monolithic solution u is a distributional solution of problem (1.1), also through the interface Γ_0 and not only inside the domains Ω_1 and Ω_2.

Typically, for second–order elliptic operators, the condition (1.3a) expresses the continuity between u_1 and u_2 on Γ_0, whereas (1.3b) indicates the equality of normal fluxes or normal stresses through the interface Γ_0. We will provide two examples:

- Laplace operator:

$$\mathcal{L}u = -\Delta u, \quad \Phi(u) = u, \quad \Psi(u) = \frac{\partial u}{\partial \mathbf{n}},$$

- Stokes operator:

$$\mathcal{L}\begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} -\nu\Delta u + \nabla p \\ -\mathrm{div} u \end{pmatrix}, \quad \Phi\begin{pmatrix} u \\ p \end{pmatrix} = u, \quad \Psi\begin{pmatrix} u \\ p \end{pmatrix} = \nu\frac{\partial u}{\partial \mathbf{n}} - p\mathbf{n}.$$

In Chapter 5, we will also introduce a more complex multiphysics problem, where the dynamics on the subdomains Ω_1 and Ω_2 differ from one another, and we will describe other types of coupling conditions coming from the physical meaning of a problem.

1.3 Iterative substructuring algorithms

In this section, we will give a brief introduction of the methods to tackle the multi–domain problem described in the previous section. These methods are well–known in the literature under the name of iterative substructuring algorithms; see, for instance, [27, 30, 62, 151, 179]. Here, we will consider the classical Dirichlet–Neumann, Neumann–Neumann and Robin methods, originally introduced for the Laplace operator (where their names stem from) and later generalised for a family of boundary value problems.

Starting from an initial guess u_1^0 and u_2^0, each method generates two sequences $\{u_1^k\}$ and $\{u_2^k\}$, which can be found independently, and which will eventually converge to u_1 and u_2, respectively. We will keep the general notation introduced in the previous section and provide references on the specific applications and their convergence analysis in each case.

1. The Dirichlet–Neumann method [40, 50, 165, 151, 179]

Given λ^0, solve for each $k \geq 0$:

$$\begin{cases} \mathcal{L}u_1^{k+1} = f & \text{in} \quad \Omega_1, \\ u_1^{k+1} = 0 & \text{on} \quad \Gamma_1, \\ \Phi(u_1^{k+1}) = \lambda^k & \text{on} \quad \Gamma_0, \end{cases}$$

then

$$\begin{cases} \mathcal{L}u_2^{k+1} = f & \text{in} \quad \Omega_2, \\ u_2^{k+1} = 0 & \text{on} \quad \Gamma_2, \\ \Psi(u_2^{k+1}) = \Psi(u_1^{k+1}) & \text{on} \quad \Gamma_0, \end{cases}$$

with

$$\lambda^{k+1} := \theta \, u_2^{k+1}\big|_{\Gamma_0} + (1-\theta)\lambda^k,$$

where θ is a positive acceleration parameter.

2. The Neumann–Neumann method [6, 29, 151, 179]

Given λ^0, solve for each $k \geq 0$:

$$\begin{cases} \mathcal{L}u_i^{k+1} = f & \text{in} \quad \Omega_i, \\ u_i^{k+1} = 0 & \text{on} \quad \Gamma_i, \\ \Phi(u_i^{k+1}) = \lambda^k & \text{on} \quad \Gamma_0, \end{cases}$$

for $i = 1, 2$, and then solve an auxiliary problem in ψ_i

$$\begin{cases} \mathcal{L}\psi_i^{k+1} = 0 & \text{in} \quad \Omega_i, \\ \psi_i^{k+1} = 0 & \text{on} \quad \Gamma_i, \\ \Psi(\psi_i^{k+1}) = \Psi(u_1^{k+1}) - \Psi(u_2^{k+1}) & \text{on} \quad \Gamma_0, \end{cases}$$

for $i = 1, 2$, with

$$\lambda^{k+1} := \lambda^k - \theta \left(\sigma_1 \, \psi_1^{k+1}\big|_{\Gamma_0} - \sigma_2 \, \psi_2^{k+1}\big|_{\Gamma_0} \right),$$

where θ is a positive acceleration parameter, σ_1 and σ_2 are two positive averaging coefficients.

3. The Robin method [5, 113, 151, 179]

Given u_2^0 solve for each $k \geq 0$:

$$\begin{cases} \mathcal{L}u_1^{k+1} = f & \text{in} \quad \Omega_1, \\ u_1^{k+1} = 0 & \text{on} \quad \Gamma_1, \\ \Psi(u_1^{k+1}) + \gamma_1 \Phi(u_1^{k+1}) = \Psi(u_2^k) + \gamma_1 \Phi(u_2^k) & \text{on} \quad \Gamma_0, \end{cases}$$

then

$$\begin{cases} \mathcal{L}u_2^{k+1} = f & \text{in} \quad \Omega_2, \\ u_2^{k+1} = 0 & \text{on} \quad \Gamma_2, \\ \Psi(u_2^{k+1}) - \gamma_2\Phi(u_2^{k+1}) = \Psi(u_1^{k+1}) - \gamma_2\Phi(u_1^{k+1}) & \text{on} \quad \Gamma_0, \end{cases}$$

where γ_1 and γ_2 are non–negative acceleration parameters satisfying $\gamma_1 + \gamma_2 > 0$. For the sake of parallelisation, in the latter problem, we could also consider u_1^k instead of u_1^{k+1}, initialising, in that case, also u_1^0.

Our approach to domain decomposition takes advantage of optimal control techniques, as it is done, for instance, in [74, 75, 77, 98, 99]. In this case, we consider the following optimisation problem: minimise over $g \in G$, where G is some suitable Banach space of functions defined on Γ_0 equipped with a norm $||\cdot||_G$, the functional

$$\mathcal{J}(u_1, u_2; g) := \frac{1}{2}||\Phi(u_1) - \Phi(u_2)||_G^2$$

subject to

$$\begin{cases} \mathcal{L}u_i = f \quad \text{in} \quad \Omega_i, \\ u_i = 0 \quad \text{on} \quad \Gamma_i, \\ \Psi(u_i) = g \quad \text{on} \quad \Gamma_0, \end{cases}$$

for $i = 1, 2$.

This approach allows us to rely on the fully–fledged field of optimal control [37, 180, 111, 119, 136, 109], for the development of various methods, using iterative minimisation algorithms and proof of their convergence.

Chapter 2

An Overview of Reduced Order Models

In this chapter, we will provide some motivation and historical background on the Reduced order models (ROMs) followed by the general mathematical setting. Then, we will discuss the classical techniques for the reduced basis generation and the efficient offline–online splitting in this context.

2.1 Motivation and historical background

In the last decades, there has been a growing interest in approximation techniques for partial differential equations (PDEs) that exploit high–performance computing within different fields of applications: naval engineering [176, 174, 47], aeronautical engineering [110, 154], biomedical engineering [173, 147, 121], etc. Very often these problems have prohibitively high computational costs, and there is always the need for much more effective algorithms in order to alleviate the complexities of such numerical models. These models are often expressed as parametrised PDEs and input parameters are used to characterize a particular problem and possible variations in its geometric configuration, physical properties, boundary conditions or source terms. The parametrised model implicitly connects these input parameters to outputs of interest, e.g., a maximum system temperature, an added mass coefficient, a crack stress intensity factor, an effective constitutive property, a waveguide transmission loss, a channel flow rate or a pressure drop.

Reduced basis methods come into play when there is a need to tackle problems in which the solution is sought for a large number of different parameter values or for real–time evaluation of the solution for onsite parameter collection. Examples of typical applications of relevance are optimisation [36, 94, 169, 172, 171, 186], design [11, 63, 47], uncertainty quantification [85, 120, 117] and real–time query [10, 61, 185]. In this case, the aim is to obtain models which can be treated more efficiently, from the computational point of view, while keeping the accuracy of the underlying numerical methods. We will call a high–fidelity method the more expensive numerical discretisation that gives us a very accurate numerical solution to the problem of interest. On the contrary, we will denote by a low–fidelity model a reduced–order model that is much more computationally efficient but possibly less accurate.

The core idea of the reduced basis approach is the identification of a suitable problem–dependent basis to effectively represent parametrised solutions to PDEs. The methodology dates back to the 1970s and it was initially presented in [9, 59, 128, 129, 130, 164]. In those early works, the approximation spaces were local and typically low–dimensional in the number of parameters. Furthermore, the absence of *a posteriori* error estimators left open questions about the accuracy of the reduced model. Consequently, substantial efforts have been recently devoted to the formulation of *a posteriori* error estimators and rigorous error bounds for outputs of interest, as well as to the design of more effective sampling techniques, see, for instance, [118, 125, 142, 156, 182].

Recent work has focused on achieving a full decoupling of the expensive discretisation scheme and the reduced order model through an offline–online procedure. In this approach, the complexity of the offline stage depends on the complexity of the underlying high–fidelity approximation of the parametrised PDE, while the complexity of the online stage depends solely on the complexity of the reduced model. When combined with the *a posteriori* error estimation, the online stage guarantees the accuracy of the high–fidelity numerical approximation at the low cost of a reduced–order model.

The full separation is achievable in the case of a so–called “affine parameter dependence": this offline–online decomposition is quite natural and has been described, for instance, in [92, 135, 159]. However, the combination of affine parameter dependence and the rigorous *a posteriori* error estimates is more involved and more recent [88, 142]. In the case of non–affine parameter dependence, the development of offline–online strategies is much less obvious but still can be achieved with the use of the empirical interpolation method [20, 72]. This last development, essential for the overall efficiency of the offline–online decomposition, has brought significant progress into the development of reduced basis methods and their use in real–time and many-query contexts for complex applications, including nonlinear problems. Reduced basis methods can be effectively applied also to nonlinear problems [71, 35, 95], although this typically introduces both numerical and theoretical complications, and many questions remain open. For classical problems with a quadratic nonlinearity, there has been substantial progress, e.g., Navier–Stokes/Boussinesq and Burgers' equations in fluid mechanics [135, 125, 156, 182, 148, 49, 160] and nonlinear elasticity in solid mechanics [91, 28].

In the last decade, great attention has been brought to the application of ROMs in the field of multiphysics problems, especially when dealing with domain decomposition techniques. A domain decomposition approach [151, 150], combined with a reduced basis method, has been successfully applied in [160, 116, 115] and further extensions are discussed in [93, 87, 89]. A coupled multiphysics setting has been proposed for simple fluid–structure interaction problems in [18, 19, 104, 106, 107, 126, 127], and for Stokes–Darcy problem in [122].

In the last decade, novel non–linear ROM approaches have been proposed, mainly following the trend topic of machine learning and artificial intelligence. The new techniques have been used to take care of different aspects of the ROMs. Autoencoders, in particular convolutional ones, [108, 60] and graph neural networks (NN) [137, 155] have been used as a surrogate nonlinear model to reduce the solution manifold, hence, substituting the classical POD and reduced basis algorithms in the offline phase. They are typically able to reduce the dimension of the reduced space in problems where there is slow decay of the Kolmogorov n–width and where classical linear

ROMs fail. Another task often delegated to neural networks is the offline phase, where simple NN [82] or radial basis function interpolation [175, 48] can speed–up the prediction phase. They are extremely useful when the problem at hand is complicated, nonlinear or its code is not accessible. Finally, we need to mention further approximation techniques that aim at minimising a functional that depends on the residual of the underlying PDEs as physics–informed neural networks (PINNs) and therein references [152, 114].

After the construction of the reduced spaces by classical techniques discussed above, the RBM–NN tries to learn the map that, given a parameter, returns the reduced coefficients of the reduced solution. In many contexts, the use of ANN brings a lot of advantages in terms of computational time in the online stage, since it only needs to solve an optimisation problem with already existing, extremely effective methods. On the other hand, using ANNs can have drawbacks when dealing with complex problems and problems with discontinuities.

2.2 Mathematical setting

The aim of this chapter is to give an overview of the general setting of reduced–order models. To keep the exposition simple we will limit ourselves to linear coercive scalar problems and then give some insights for tackling more complex problems. For more details, we suggest consulting [81, 149, 161]. The extension to vector differential problems is quite straightforward.

Let us consider a physical domain $\Omega \subset \mathbb{R}^d$, $d = 1, 2, 3$, with Lipschitz boundary Γ and let us denote by Γ_D and Γ_N the portions of the boundary where we impose Dirichlet and Neumann boundary conditions, respectively.

We assume the problem to be parameter–dependent: let $\mu \in \mathcal{P}$ be a parameter, with $\mathcal{P} \in \mathbb{R}^p$, $p \geq 1$ the parameter space. We want to solve the following problem: for every $\mu \in \mathcal{P}$, find $u(\mu) : \Omega \to \mathbb{R}$ such that:

$$\begin{cases} \mathcal{L}(\mu)u(\mu) = f(\mu) & \text{in} \quad \Omega, \\ u(\mu) = 0 & \text{on} \quad \Gamma_D, \\ \Psi(u(\mu); \mu) = g(\mu) & \text{on} \quad \Gamma_N, \end{cases}$$

where $\mathcal{L}(\cdot)$ is a linear second–order differential operator with a corresponding Neumann operator $\Psi(\cdot;\cdot)$, and $f(\cdot)$ and $g(\cdot)$ are some prescribed data. For the sake of simplicity, we consider here homogeneous Dirichlet condition; in Chapters 3 and 4 we also consider non–homogeneous Dirichlet problems and we describe how to tackle them.

Let $(V, ||\cdot||_V)$ be a function space endowed with the $H_1(\Omega)$–norm, where $V := \{u \in V : u = 0 \text{ on } \Gamma_D\}$. The variational counterpart of the problem then reads as follows: for every $\mu \in \mathcal{P}$, find $u(\mu) \in V$ such that:

$$a(u(\mu), v; \mu) = l(v; \mu) \quad \forall v \in V, \tag{2.1}$$

where $a(\cdot, \cdot; \cdot)$ is a bilinear coercive continuous form with respect to the first two parameters and the $H^1(\Omega)$–norm, while $l(\cdot; \cdot)$ is a linear and continuous form with respect to the first parameter

and the $H^1(\Omega)$–norm, that takes into account the data of the problem. Under these constraints, it is well–known that problem (2.1) is well–posed for each $\mu \in \mathcal{P}$ due to the Lax–Milgram lemma.

The next step towards the construction of the reduced space for efficient solving problem (2.1) for any input parameter $\mu \in \mathcal{P}$ is to provide a suitable spatial discretisation of the variational space V. Throughout this thesis, we will focus on Finite Element (FE) discretisation, but the methodology works just as well for other discretisation techniques, such as Finite Volumes or Discontinuous Galerkin methods.

In the framework of FE discretisation, under the assumption that the domain Ω is polygonal, we first provide a well–defined triangulation of Ω, and then we build a finite–dimensional linear subspace $V_h \subset V$ spanned by piecewise continuous polynomials of degree less or equal to $k \in \mathbb{N}$, with local supports. We denote by N_h the dimension of the space V_h, and by φ_i, $i = 1, ..., N_h$ its basis. Then, the discretised version of (2.1) reads: for every $\mu \in \mathcal{P}$, find $u_h(\mu) \in V_h$ such that:

$$a(u_h(\mu), v_h; \mu) = l(v_h; \mu) \quad \forall v_h \in V_h. \tag{2.2}$$

This problem will be denoted as a truth or high–fidelity problem. This is because of the assumption that the solution needs to be computed for one parameter value only and this task can be achieved with as high accuracy as desired. A unique solver is potentially very expensive due to the fact that for high accuracy we might need to use a high number of basis functions N_h. This implies that linear and nonlinear solvers could be extremely expensive in terms of computational time and resources, as they might scale as $\mathcal{O}\left(N_h^2\right)$ or $\mathcal{O}\left(N_h^3\right)$, depending on the solver.

Problem (2.2) can be rewritten in the algebraic form: find a vector $\underline{u_h}(\mu) = [u_h^1(\mu), .., u_h^{N_h}(\mu)] \in \mathbb{R}^{N_h}$ solving the following linear system:

$$A_h(\mu)\underline{u_h}(\mu) = F_h(\mu), \tag{2.3}$$

where $A_h(\mu)$ is a matrix of dimension $N_h \times N_h$ with $[A_h(\mu)]_{i,j} = a(\varphi_j, \varphi_i; \mu)$ and $F_h(\mu)$ is a vector of dimension N_h with $[F_h(\mu)]_i = l(\varphi_i; \mu)$. Here $\underline{u_h}(\mu)$ is a vector containing the coefficients of the expansion of the solution $u_h(\mu)$ with respect to the basis $\left\{\varphi_1, ..., \varphi_{N_h}\right\}$, i.e.,

$$u_h(\mu) = \sum_{i=1}^{N_h} u_h^i(\mu)\varphi_i.$$

We introduce now the notion of discrete solution manifold, defined as the manifold of the high–fidelity solution varying with the parameter μ, i.e.,

$$\mathcal{M}_h := \{u_h(\mu) : \mu \in \mathcal{P} \text{ and } u_h(\mu) \text{ is a solution to } (2.2)\}. \tag{2.4}$$

A fundamental assumption in the development of any (linear) reduced model is that the solution manifold is of low dimension, i.e., that the span of a low number of appropriately chosen basis functions (which we will call reduced basis functions) represents $\mathcal{M}_h$ with a small error. In the following section, we will describe the techniques for building such basis functions. Let us assume

for the moment that we have already computed a set of N reduced basis functions $\{\Phi_1, ..., \Phi_N\}$, and we generate a reduced basis space V_N by spanning over those functions, that is

$$V_N = \text{span}\,\{\Phi_1, ..., \Phi_N\} \subset V_h, \quad \dim V_N = N. \tag{2.5}$$

The above assumption on the low dimensionality of the reduced spaces implies that $N \ll N_h$. Now, given the reduced space V_N, we define a reduced problem as the Galerkin projection of the problem (2.1) onto the reduced space V_N: for every $\mu \in \mathcal{P}$, find $u_N(\mu) \in V_N$ such that

$$a(u_N(\mu), v_N; \mu) = l(v_N; \mu) \quad \forall v_N \in V_N. \tag{2.6}$$

At the algebraic level, we have the following linear problem: find a vector $\underline{u_N}(\mu) = [u_N^1(\mu), .., u_N^N(\mu)]$, $\underline{u_N}(\mu) \in \mathbb{R}^N$ solving the following linear system:

$$A_N(\mu)\underline{u_N}(\mu) = F_N(\mu), \tag{2.7}$$

where $A_N(\mu)$ is a matrix of dimension $N \times N$ with $[A_N(\mu)]_{i,j} = a(\Phi_j, \Phi_i; \mu)$ and $F_N(\mu)$ is a vector of dimension N with $[F_N(\mu)]_i = l(\Phi_i; \mu)$. Here $\underline{u_N}(\mu)$ is a vector containing the coefficients of the expansion of the solution $u_N(\mu)$ with respect to the basis $\{\Phi_1, ..., \Phi_N\}$, i.e.

$$u_N(\mu) = \sum_{i=1}^{N} u_N^i(\mu)\Phi_i.$$

There is a direct connection between the matrices $A_h(\cdot)$ and $A_N(\cdot)$ and between the vectors $F_h(\mu)$ and $F_N(\mu)$. We define a reduced basis matrix Z of dimension $N_h \times N$ as follows: each j–th column of Z contains the expansion coefficients of the reduced basis function Φ_j with respect to the basis $\{\varphi_i\}_{i=1}^{N_h}$ of the high–fidelity space V_h:

$$\Phi_j = \sum_{i=1}^{N_h} \alpha_{i,j}\varphi_i, \quad j = 1, ..., N.$$

With these notations, Z is then defined as

$$[Z]_{i,j} := \alpha_{i,j}, \quad i = 1, ..., N_h, \; j = 1, ..., N.$$

Then, we have the following relations:

$$A_N(\cdot) = Z^T A_h(\cdot) Z, \quad F_N(\cdot) = Z^T F_h(\cdot). \tag{2.8}$$

2.3 Reduced basis generation

In this section, we will give a quick overview of the two most classical ways of generating the reduced basis functions, namely the greedy basis generation [81, 158, 142] and the Proper Orthogonal

Decomposition (POD) [81, 100, 101, 96, 41, 140, 158]. In both cases, we begin by introducing a finite training set $\mathcal{P}_m \subset \mathcal{P}$ of cardinality M and we define an approximation of the discrete solution manifold:

$$\mathcal{M}_{h,M} = \{u_h(\mu) : \mu \in \mathcal{P}_M \text{ and } u_h(\mu) \text{ is a solution to } (2.2)\} \subset \mathcal{M}_h. \tag{2.9}$$

We assume $\mathcal{P}_M$ is fine enough, so that $\mathcal{M}_{h,M}$ is a good representation of $\mathcal{M}_h$.

Greedy Basis Generation

The greedy generation of the reduced basis space is an iterative procedure, where at each iteration one new basis function is added in order to improve the overall accuracy of the reduced space. It requires only one truth solve of (2.2) at each iteration and a total of N truth solves to generate an N–dimensional reduced basis space. An essential component of the greedy algorithm is the availability of a cheap *a posteriori* error estimator $\Delta(\cdot)$ that predicts the error between the high–fidelity and the reduced–order solution, that is:

$$||u_h(\mu) - u_N(\mu)||_\mu \leq \Delta(\mu), \quad \forall \mu \in \mathcal{P},$$

where $||\cdot||_\mu = \sqrt{a(\cdot,\cdot;\mu)}$ under the assumption of coercivity of the bilinear form a.

If at the n–th iteration an n–dimensional reduced basis space V_n is given, the next basis function is the truth solution $u_h(\mu_{n+1})$, where

$$\mu_{n+1} = \text{argmax}_{\mu \in \mathcal{P}_M} \Delta(\mu),$$

and the reduced basis space is defined as $V_{n+1} = \text{span}\{u_h(\mu_1), ..., u_h(\mu_n), u_h(\mu_{n+1})\}$. This is repeated until $\Delta(\mu_{n+1}) < \varepsilon$, with ε to be a chosen tolerance. We also note that this procedure produces hierarchical bases, that is $V_1 \subset ... \subset V_n \subset V_{n+1}$. Algorithm 2.1 lists a pseudo–code of the greedy procedure.

The greedy algorithm is quite cheap in terms of computational time and resources since it requires only one solution of the high–fidelity problem (2.2) per iteration. On the other hand, the error estimator $\Delta(\cdot)$ is an indispensable element of the procedure, but it is available for a very limited class of differential problems and not, for example, for the problems discussed in this thesis. Hence, we will need another method for reduced basis generation, that is, the Proper Orthogonal Decomposition.

Proper Orthogonal Decomposition

In contrast to the greedy procedure, the Proper Orthogonal Decomposition (POD) relies on an explore–and–compress strategy, in which the truth solution is computed for each parameter in the training set $\mathcal{P}_M$. The POD then compresses the constructed set retaining only the essential information. The N–dimensional POD–space is the space that minimizes the following quantity:

$$\sqrt{\frac{1}{M} \sum_{\mu \in \mathcal{P}_M} \inf_{v_N \in V_N} ||u_h(\mu) - v_N||_V^2}.$$

Algorithm 2.1 The greedy algorithm

Input: $\varepsilon > 0$ and μ_n
$n := 1$
while $\Delta(\mu_n) > \varepsilon$ **do**
  Compute $u_h(\mu_n)$ solution to (2.2) for μ_1
  Set $V_n := \text{span}\{u_h(\mu_1), ..., u_h(\mu_n)\}$
  for each $\mu \in \mathcal{P}_M$ **do**
    Compute the reduced basis approximation $u_n(\mu) \in V_n$ solution to (2.6) with $V_N = V_n$
    Evaluate the error estimator $\Delta(\mu)$
  end for
  $n := n + 1$
end while
Set $N := n$, $V_N := V_n$.
return V_N

over all N–dimensional subspaces V_N of the span of the manifold $\mathcal{M}_{h,M}$. This problem can be solved exactly [53]. For this, we need to construct a correlation matrix C of dimension $M \times M$ defined as

$$[C]_{ij} = \frac{1}{M}\left(u_h(\mu_i), u_h(\mu_j)\right)_V \quad \text{for } i, j = 1, ..., M,$$

where $(\cdot, \cdot)_V$ is $H^1(\Omega)$ inner product. Then, we solve an eigenvalue–eigenvector problem

$$Cv_k = \lambda_k v_k, \quad 1 \leq k \leq N,$$

where $\lambda_1 \geq ... \geq \lambda_N \geq 0$. Finally, the reduced basis function $\Phi_k, k = 1, ..., N$ is defined as:

$$\Phi_k = \frac{1}{M}\sum_{j=1}^{M}(v_k)_j u_h(\mu_j),$$

where $(v_k)_j$ denotes the j–th component of the eigenvector v_k.

As it can be seen, POD is a much more expensive procedure, since it requires the computation of the solution for each parameter in the set $\mathcal{P}_M$, which might be of large cardinality. On the other hand, unlike the greedy procedure, the POD does not require any special error estimator, which is usually not available for complex non–coercive and non–linear problems. For this reason, taking into account the complexity of the problems which are dealt with within this thesis, we will rely on the POD generation method.

2.4 Offline–online decomposition

Reduced–order models are incredibly effective thanks 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, such as reduced basis generation discussed in the previous section, that are pre–computed in the offline phase.
In the ideal setting, the computational costs involved in the online stage should be independent of the complexity of the truth problem, measured by N_h, and should depend solely on the dimension $N \ll N_h$ of the reduced basis space. This is not always the case, as we can see from equation (2.8), when, for instance, the bilinear form $a(\cdot,\cdot;\mu)$ and the linear form $l(\cdot;\mu)$ depend on the parameter μ. However, this restriction can be overcome if we assume that the forms $a(\cdot,\cdot;\mu)$, $l(\cdot;\mu)$ satisfy the so–called affine decomposition property:

$$\begin{aligned} a(u,v;\mu) &= \sum_{q=1}^{Q_a} \theta_a^q(\mu) a_q(u,v), \\ l(v;\mu) &= \sum_{q=1}^{Q_l} \theta_l^q(\mu) l_q(v), \end{aligned}$$

where each form $a_q : V \times V \to \mathbb{R}$, $q = 1, ..., Q_a$, and $l_q : V \to \mathbb{R}$, $q = 1, ..., Q_l$, is parameter–independent, and the coefficients $\theta_a^q : \mathcal{P} \to \mathbb{R}$, $q = 1, ..., Q_a$ and $\theta_l^q : \mathcal{P} \to \mathbb{R}$, $q = 1, ..., Q_l$ depend solely on the parameter values. In this case, the matrix $A_h(\mu)$ and the vector $F_h(\mu)$ can be decomposed as follows:

$$A_h(\mu) = \sum_{q=1}^{Q_a} \theta_a^q(\mu) A_h^q, \quad F(\mu) = \sum_{q=1}^{Q_l} \theta_l^q(\mu) F_h^q,$$

where

$$\begin{aligned} [A_h^q]_{ij} &= a_q(\varphi_j, \varphi_i), \quad i,j = 1, ..., N_h,\ q = 1, ..., Q_a, \\ [F_h^q]_i &= l_q(\varphi_i), \quad i = 1, ..., N_h,\ q = 1, ..., Q_l. \end{aligned}$$

and their reduced counterparts can be computed by

$$A_N^q = Z^T A_h^q Z,\ q = 1, ..., Q_a, \quad F_N^q = Z^T F_h^q,\ q = 1, ..., Q_l.$$

All these quantities can be pre–computed in the offline stage and stored for future use in the online stage. Therefore, during the online stage, we will need to compute only parameter–dependent coefficients $\theta_a^q(\cdot)$, $q = 1, ..., Q_a$ and $\theta_l^q(\cdot)$, $q = 1, ..., Q_l$, instead of parameter–dependent matrices and vectors.

The affine decomposition assumption is satisfied only for a certain class of problems, but, in general, many problems fail to satisfy this property; for example, we do not have an explicit affine decomposition when the problem is non–linear. For this reason, a lot of research has been conducted on how to approximate functions and operators, in order to obtain the desired decomposition. Some well known methods are, for example, the Empirical Interpolation Method (EIM) [81, 42, 158, 52]

the Discrete Empirical Interpolation Method (DEIM) [81, 43, 158] and the Empirical Quadrature Method (EQM) [134]. We will not make use of these techniques in this thesis, but it is very important to highlight that their introduction in the developed algorithms would improve the overall effectiveness of the method.

We should also mention the application of Artificial Neural Networks (ANN) to Reduced–Order Modelling. In particular, we focus on surrogate procedures to the Galerkin projection in the online phase. After the construction of the reduced spaces as described in Section 2.3, the RBM–neural network (RBM–NN) tries to learn the map that, given a parameter, returns the reduced coefficients of the reduced solution [82]. This procedure starts from a training set composed of parameters μ_j (including times) and reduced coefficients of the solutions $u_N^i(\mu_j)$. Then, different techniques like radial basis function interpolation [48, 175] or ANN [82] can be trained to learn the map from the parameters to the reduced coefficients. Finally, in the online stage, the map is quickly evaluated and a very fast reconstruction of the solutions can be performed.

In many contexts, the use of ANN brings a lot of advantages in terms of computational time in the online stage, since it only needs to solve in the offline phase an optimisation task with extremely effective methods, e.g. stochastic gradient descent method. On the other hand, using ANNs can have drawbacks when dealing with complex problems and problems with discontinuities in the parameter to reduced coefficient map. We will discuss this in more detail in Section 4.5.3.

Chapter 3

Stationary Incompressible Navier–Stokes Equations

In this chapter, we introduce an optimisation–based domain-decomposition formulation of the incompressible Navier–Stokes equations and provide a gradient–based optimisation algorithm for the resulting optimal control problem. Then, we build a reduced–order model based on the Proper Orthogonal Decomposition methodology for the parameter–dependent Navier–Stokes equations. At the end of the chapter, we show some numerical results for two toy problems: the backward–facing step and the lid–driven cavity flows. The results of this section have been published in the paper [143].

3.1 Problem formulation

In this section, starting from a monolithic formulation of the incompressible Navier–Stokes equations, we introduce a two–domain optimisation–based domain–decomposition formulation in both strong and weak form.

3.1.1 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. Let $f : \Omega \to \mathbb{R}^2$ be the forcing term, ν the kinematic viscosity, u_D a given Dirichlet datum. The problem reads as follows: find the velocity field $u : \Omega \to \mathbb{R}^2$ and the pressure $p : \Omega \to \mathbb{R}$ s.t.

$$-\nu\Delta u + (u \cdot \nabla)\, u + \nabla p = f \quad \text{in} \quad \Omega, \tag{3.1a}$$

$$-\mathrm{div} u = 0 \quad \text{in} \quad \Omega, \tag{3.1b}$$

$$u = u_D \quad \text{on} \quad \Gamma_D, \tag{3.1c}$$

$$\nu\frac{\partial u}{\partial \mathbf{n}} - p\mathbf{n} = 0 \quad \text{on} \quad \Gamma_N, \tag{3.1d}$$

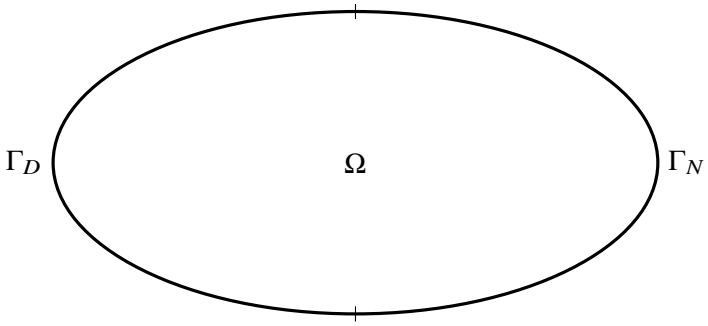


Figure 3.1: Physical domain

where Γ_D and Γ_N are disjoint subsets of Γ (as it is shown in Figure 3.1) and $\mathbf{n}$ is an outward unit normal vector to Γ_N.

3.1.2 Domain Decomposition (DD) formulation

Let Ω_i, $i = 1, 2$, be open subsets of Ω, such that $\overline{\Omega} = \overline{\Omega_1 \cup \Omega_2}$, $\Omega_1 \cap \Omega_2 = \emptyset$. 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 3.2.
The DD formulation reads as follows: for $i = 1, 2$, given $f_i : \Omega_i \to \mathbb{R}^2$ and $u_{i,D} : \Gamma_{i,D} \to \mathbb{R}^2$, find $u_i : \Omega_i \to \mathbb{R}^2$, $p_i : \Omega_i \to \mathbb{R}$ s.t.

$$-\nu\Delta u_i + (u_i \cdot \nabla)\, u_i + \nabla p_i = f_i \quad \text{in} \quad \Omega_i, \tag{3.2a}$$
$$-\mathrm{div} u_i = 0 \quad \text{in} \quad \Omega_i, \tag{3.2b}$$
$$u_i = u_{i,D} \quad \text{on} \quad \Gamma_{i,D}, \tag{3.2c}$$
$$\nu\frac{\partial u_i}{\partial \mathbf{n}_i} - p_i\mathbf{n}_i = 0 \quad \text{on} \quad \Gamma_{i,N}, \tag{3.2d}$$
$$\nu\frac{\partial u_i}{\partial \mathbf{n}_i} - p_i\mathbf{n}_i = (-1)^{i+1} g \quad \text{on} \quad \Gamma_0, \tag{3.2e}$$

for some $g : \Gamma_0 \to \mathbb{R}^2$ such that the functions defined in the following way

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

satisfy the monolithic equations (3.1).

Even though in the numerical simulations we will focus on the cases where $f_i = f|_{\Omega_i}$, $u_{i,D} = u_D|_{\Gamma_{i,D}}$ for $i = 1, 2$, the whole theoretical exposition in this thesis works just as well for more general functions $f_1, f_2, u_{1,D}$ and $u_{2,D}$.

For any g the solution to the problem (3.2) is not the same as the solution to the problem (3.1), that is $u_1 \neq u|_{\Omega_1}$, $p_1 \neq p|_{\Omega_1}$, $u_2 \neq u|_{\Omega_2}$ and $p_2 \neq p|_{\Omega_2}$. On the other hand, there exists a choice for

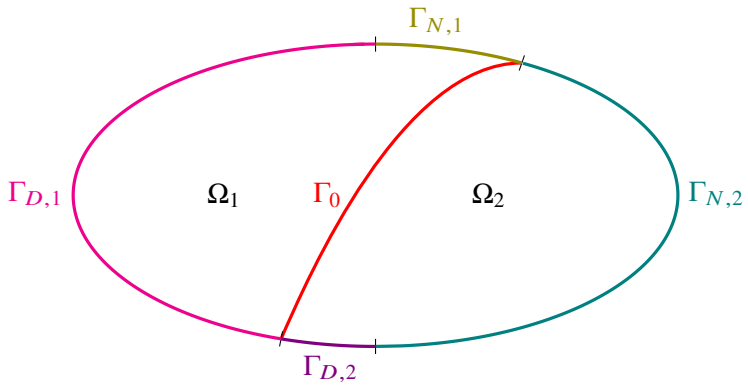


Figure 3.2: Domain Decomposition of the fluid domain

g, $g = \left(\nu \frac{\partial u_1}{\partial \mathbf{n}_1} - p_1 \mathbf{n}_1\right)|_{\Gamma_0} = -\left(\nu \frac{\partial u_2}{\partial \mathbf{n}_2} - p_2 \mathbf{n}_2\right)|_{\Gamma_0}$, such that the solutions to (3.2) coincide with the solution to (3.1) on the corresponding subdomains. Therefore, we must find a g, such that u_1 is as close as possible to u_2 at the interface Γ_0. One way to accomplish this is to minimise the functional

$$\mathcal{J}(u_1, u_2) := \frac{1}{2} \int_{\Gamma_0} |u_1 - u_2|^2 \, d\Gamma. \tag{3.3}$$

Instead of (3.3) we can also consider the penalised or regularised functional

$$\mathcal{J}_\gamma(u_1, u_2; g) := \frac{1}{2} \int_{\Gamma_0} |u_1 - u_2|^2 \, d\Gamma + \frac{\gamma}{2} \int_{\Gamma_0} |g|^2 \, d\Gamma, \tag{3.4}$$

where γ is a constant that can be chosen to change the relative importance of the terms in (3.4). Thus we face an optimisation problem under PDE constraints: minimise the functional (3.3) (or (3.4)) over a suitable function g, subject to (3.2).

3.1.3 Variational Formulation of the PDE constraints

For $i = 1, 2$, define the following spaces and the norms with which each of them is endowed:

- $V_i := \left\{u \in H^1(\Omega_i; \mathbb{R}^2)\right\}, \quad ||\cdot||_{V_i} = ||\cdot||_{H^1(\Omega_i)}$,
- $V_{i,0} := \left\{u \in H^1(\Omega_i; \mathbb{R}^2) : \ u|_{\Gamma_{i,D}} = 0\right\}, \quad ||\cdot||_{V_{i,0}} = ||\cdot||_{H^1_0(\Omega_i)}$,
- $Q_i := \left\{p \in L^2(\Omega_i; \mathbb{R})\right\}, \quad ||\cdot||_{Q_i} = ||\cdot||_{L^2(\Omega_i)}$.

Then, we define the following bilinear and trilinear forms: for i=1,2

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

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

where $(\cdot, \cdot)_\omega$ indicates the $L^2(\omega)$ inner product.

Consequently, the variational counterpart of (3.2) reads as follows: for $i = 1, 2$, find $u_i \in V_i$ and $p_i \in Q_i$ s.t.

$$a_i(u_i, v_i) + c_i(u_i, u_i, v_i) + b_i(v_i, p_i) = (f_i, v_i)_{\Omega_i} + \left((-1)^{i+1} g, v_i\right)_{\Gamma_0} \qquad \forall v_i \in V_{i,0}, \tag{3.5a}$$

$$b_i(u_i, q_i) = 0 \qquad \forall q_i \in Q_i, \tag{3.5b}$$

$$u_i = u_{i,D} \qquad \text{on } \Gamma_{i,D}. \tag{3.5c}$$

Remark. In general, the fluxes through an interface Γ_0 for the weak formulation of Navier–Stokes equation live in the space $H^{-\frac{1}{2}}(\Gamma_0)$ so that, in theory, the definition (3.4) of functional $\mathcal{J}_\gamma$ is not justified, since it includes the $L^2(\Gamma_0)$–norm of the function g. Nevertheless, as it will be evident in Section 3.2.3, the family of optimisation algorithms that are used to tackle the optimal–control problem in hand, in fact, define the respective approximation of g that belongs to the space $H^{\frac{1}{2}}(\Gamma_0)$.

3.2 Optimality system and minimisation algorithms

In this section, we derive the optimality conditions of the optimal control problem obtained at the end of the previous section together with the expression of the gradient of the objective functional obtained by sensitivity analysis. Then, we list a minimisation gradient–based algorithm for the problem at hand.

3.2.1 Optimality system

One of the ways to address the constrained optimisation problem is to reformulate the initial problem in terms of a Lagrangian functional by introducing the so–called adjoint variables. In this way, the optimal solution to the original problem is sought among the stationary points of the Lagrangian, see, for instance, [76, 83].

We define the Lagrangian functional as follows:

$$\begin{aligned} \mathcal{L}(u_1, p_1, u_2, p_2, \xi_1, \xi_2, \lambda_1, \lambda_2; g) := \mathcal{J}_\gamma(u_1, u_2; g) - \sum_{i=1}^{2} [a_i(u_i, \xi_i) \\ + c_i(u_i, u_i, \xi_i) + b_i(\xi_i, p_i) + b_i(u_i, \lambda_i)] + \sum_{i=1}^{2} (f_i, \xi_i)_{\Omega_i} + \sum_{i=1}^{2} ((-1)^{i+1} g, \xi_i)_{\Gamma_0}. \end{aligned} \tag{3.6}$$

Notice that, technically, we should have also included Lagrange multipliers corresponding to the non–homogeneous Dirichlet boundary conditions (3.5c) in the definition of the functional $\mathcal{L}$; however, since the functional $\mathcal{J}_\gamma$ does not explicitly depend on $u_{1,D}$ and $u_{2,D}$, the corresponding

Dirichlet boundary conditions for the adjoint equation that we are going to derive below will be homogeneous on $\Gamma_{1,D}$ and $\Gamma_{2,D}$.

We now apply the necessary conditions for finding stationary points of $\mathcal{L}$. Setting to zero the first variations w.r.t. ξ_i and λ_i, for $i = 1, 2$, yields the state equations (3.5a)– (3.5b). Setting to zero the first variations w.r.t. u_1, p_1, u_2 and p_2 yields the adjoint equations:

$$\begin{aligned} a_i(\eta_i, \xi_i) + c_i(\eta_i, u_i, \xi_i) + c_i(u_i, \eta_i, \xi_i) + b_i(\eta_i, \lambda_i) \\ = ((-1)^{i+1}\eta_i, u_1 - u_2)_{\Gamma_0} \end{aligned} \qquad \forall \eta_i \in V_{i,0}, \tag{3.7a}$$

$$b_i(\xi_i, \mu_i) = 0 \qquad \forall \mu_i \in Q_i. \tag{3.7b}$$

Finally, setting to zero the first variations w.r.t. g yields the optimality condition:

$$\gamma(h, g)_{\Gamma_0} + (h, \xi_1 - \xi_2)_{\Gamma_0} = 0 \quad \forall h \in L^2(\Gamma_0). \tag{3.8}$$

3.2.2 Sensitivity derivatives

In order to obtain the expression for the gradient of the optimisation problem at hand, we will resort to the sensitivity approach, see, for instance, [76, 83]. The approach consists of finding equations for direction derivatives of the state variable with respect to the control, called sensitivities.

The first derivative $\frac{d\mathcal{J}_\gamma}{dg}$ of $\mathcal{J}_\gamma$ is defined through its action on variation $\tilde{g}$ as follows:

$$\left\langle \frac{d\mathcal{J}_\gamma}{dg}, \tilde{g} \right\rangle = (u_1 - u_2, \tilde{u}_1 - \tilde{u}_2)_{\Gamma_0} + \gamma(g, \tilde{g})_{\Gamma_0}, \tag{3.9}$$

where $\tilde{u}_1 \in V_{1,0}$, $\tilde{u}_2 \in V_{2,0}$ are the solutions to:

$$a_i(\tilde{u}_i, v_i) + c_i(\tilde{u}_i, u_i, v_i) + c_i(u_i, \tilde{u}_i, v_i) + b_i(v_i, \tilde{p}_i) = ((-1)^{i+1}\tilde{g}, v_i)_{\Gamma_0} \qquad \forall v_i \in V_{i,0}, \tag{3.10a}$$

$$b_i(\tilde{u}_i, q_i) = 0 \qquad \forall q_i \in Q_i. \tag{3.10b}$$

We can make use of the adjoint equations (3.7) in order to find the representation of the gradient of the functional $\mathcal{J}_\gamma$. Let ξ_1 and ξ_2 be the solutions to (3.7), $\tilde{u}_1$ and $\tilde{u}_2$ be the solutions to (3.10). By setting $\eta_i = \tilde{u}_i$ in (3.7a), $\mu_i = \tilde{p}_i$ in (3.7b), $v_i = \xi_i$ in (3.10a) and $q_i = \lambda_i$ in (3.10b) we obtain:

$$(u_1 - u_2, \tilde{u}_1 - \tilde{u}_2)_{\Gamma_0} = (\tilde{g}, \xi_1 - \xi_2)_{\Gamma_0},$$

so that it yields the explicit formula for the gradient of $\mathcal{J}_\gamma$:

$$\frac{d\mathcal{J}_\gamma}{dg}(u_1, u_2; g) = \gamma g + (\xi_1 - \xi_2)|_{\Gamma_0}, \tag{3.11}$$

where ξ_1 and ξ_2 are determined from g through (3.7). Notice that the gradient expression (3.11) is consistent with the optimality condition (3.8) derived in the previous section.

3.2.3 Gradient–based algorithm for PDE–constraint optimisation problem

In view of being able to provide a closed–form formula for the gradient for the objective functional $\mathcal{J}_\gamma$, the natural way to proceed is to resort to a gradient–based iterative optimisation algorithm.

In order to keep the exposition simple, we consider the following simple gradient method with a constant step size $\alpha > 0$: given a starting guess $g^{(0)}$, let

$$g^{(n+1)} := g^{(n)} - \alpha \frac{d\mathcal{J}_\gamma}{dg}\left(u_1^{(n)}, u_2^{(n)}; g^{(n)}\right). \tag{3.12}$$

Combining this with (3.11) we obtain:

$$g^{(n+1)} = g^{(n)} - \alpha\left(\gamma g^{(n)} + (\xi_1^{(n)} - \xi_2^{(n)})|_{\Gamma_0}\right), \tag{3.13}$$

or

$$g^{(n+1)} = (1 - \alpha\gamma)\, g^{(n)} - \alpha(\xi_1^{(n)} - \xi_2^{(n)})|_{\Gamma_0}, \tag{3.14}$$

where $\xi_1^{(n)}$ and $\xi_2^{(n)}$ are determined from (3.7) with g replaced by $g^{(n)}$.
In summary, we have Algorithm 3.1.

Algorithm 3.1 Gradient method with a fixed step

Input: $g^{(0)}$, $\alpha > 0$
$n := 0$
while Convergence criteria are not met **do**
  Solve (3.5) for $u_1^{(n)} \in V_1$, $u_2^{(n)} \in V_2$ with $g = g^{(n)}$
  Solve (3.7) for $\xi_1^{(n)} \in V_{1,0}$, $\xi_2^{(n)} \in V_{2,0}$ with $u_1 = u_1^{(n)}$, $u_2 = u_2^{(n)}$
  Set $g^{(n+1)} := (1 - \alpha\gamma)\, g^{(n)} - \alpha\left(\xi_1^{(n)} - \xi_2^{(n)}\right)\Big|_{\Gamma_0}$
  $n := n + 1$
end while
Set $u_1 := u_1^{(n)}$, $p_1 := p_1^{(n)}$, $u_2 := u_2^{(n)}$, $p_2 := p_2^{(n)}$ and $g := g^{(n)}$
return u_1, p_1, u_2, p_2, g

In practice, the typical methods used to solve problems like the one considered in this thesis are Broyden–Fletcher–Goldfarb–Shanno (BFGS) and Newton Conjugate Gradient (CG) algorithms, which tend to show much faster convergence and higher efficiency with respect to the steepest–descent algorithm.

3.3 Finite Element discretisation

In this section, we present the Finite Element spatial discretisation for the optimal control problem previously introduced. In order to be able to apply FE discretisation, the domains Ω_i, $i = 1, 2$,

and the interface Γ_0 are assumed to be polygonal. We consider two well–defined triangulations $\mathcal{T}_1$ and $\mathcal{T}_2$ over the domains Ω_1 and Ω_2 respectively, and an extra lower–dimensional triangulation $\mathcal{T}_0$ of the interface Γ_0; additionally, we assume that $\mathcal{T}_1$, $\mathcal{T}_2$ and $\mathcal{T}_0$ share the same degrees on freedom relative to the interface Γ_0. We can then define usual Lagrangian FE spaces $V_{i,h} \subset V_i$, $V_{i,0,h} \subset V_{i,0}$, $Q_{i,h} \subset Q_i$, $i = 1, 2$ and $X_h \subset L^2(\Gamma_0)$ endowed with $L^2(\Gamma_0)$–norm; the spaces $V_{i,h}$, $V_{i,0,h}$ and $Q_{i,h}$ for $i = 1, 2$ are endowed with the same norms as their continuous counterparts. Since the problems at hand have a saddle–point structure, in order to guarantee the well–posedness of the discretised problem, we require the FE spaces to satisfy the following inf–sup conditions: there exist positive constants c_1, c_2, c_3 and c_4 s.t.

$$\inf_{q_{i,h} \in Q_{i,h} \setminus \{0\}} \sup_{v_{i,h} \in V_{i,h} \setminus \{0\}} \frac{b_i(v_{i,h}, q_{i,h})}{||v_{i,h}||_{V_{i,h}} ||q_{i,h}||_{Q_{i,h}}} \geq c_i, \quad i = 1, 2, \tag{3.15}$$

$$\inf_{q_{i,h} \in Q_{i,h} \setminus \{0\}} \sup_{v_{i,h} \in V_{i,0,h} \setminus \{0\}} \frac{b_i(v_{i,h}, q_{i,h})}{||v_{i,h}||_{V_{i,0,h}} ||q_{i,h}||_{Q_{i,h}}} \geq c_{i+2}, \quad i = 1, 2. \tag{3.16}$$

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.

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

$$\mathcal{J}_{\gamma,h}(u_{1,h}, u_{2,h}; g_h) := \frac{1}{2} \int_{\Gamma_0} \left| u_{1,h} - u_{2,h} \right|^2 d\Gamma + \frac{\gamma}{2} \int_{\Gamma_0} |g_h|^2 \, d\Gamma \tag{3.17}$$

under the constraints that $u_{i,h} \in V_{i,h}$, $p_{i,h} \in Q_{i,h}$ satisfy the following variational equations for $i = 1, 2$

$$\begin{aligned} a_i(u_{i,h}, v_{i,h}) + c_i(u_{i,h}, u_{i,h}, v_{i,h}) + b_i(v_{i,h}, p_{i,h}) & \\ = (f_i, v_{i,h})_{\Omega_i} + ((-1)^{i+1} g_h, v_{i,h})_{\Gamma_0} & \quad \forall v_{i,h} \in V_{i,0,h}, \qquad (3.18a) \\ b_i(u_{i,h}, q_{i,h}) = 0 & \quad \forall q_{i,h} \in Q_{i,h}, \qquad (3.18b) \\ u_{i,h} = u_{i,D,h} & \quad \text{on } \Gamma_{i,D}, \qquad (3.18c) \end{aligned}$$

where $u_{i,D,h}$ 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 (3.18) and of the functional (3.17) is the same as the one of the continuous case so that it enables us to provide the following expression of the gradient of the discretised functional (3.17):

$$\frac{d\mathcal{J}_{\gamma,h}}{dg_h}(u_{1,h}, u_{2,h}; g_h) = \gamma g_h + (\xi_{1,h} - \xi_{2,h})|_{\Gamma_0}, \tag{3.19}$$

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

$$\begin{aligned} a_i(\eta_{i,h}, \xi_{i,h}) + c_i\left(\eta_{i,h}, u_{i,h}, \xi_{i,h}\right) + c_i\left(u_{i,h}, \eta_{i,h}, \xi_{i,h}\right) & \\ + b_i(\eta_{i,h}, \lambda_{i,h}) = ((-1)^{i+1}\eta_{i,h}, u_{1,h} - u_{2,h})_{\Gamma_0} & \quad \forall \eta_{i,h} \in V_{i,0,h}, \qquad (3.20a) \\ b_i(\xi_{i,h}, \mu_{i,h}) = 0 & \quad \forall \mu_{i,h} \in Q_{i,h}. \qquad (3.20b) \end{aligned}$$

We would also like to stress that at the algebraic level, the discretised minimisation problem can be recast in the setting of the finite–dimensional space $\mathbb{R}^p$, where p is the number of Finite Element degrees of freedom which belong to the interface Γ_0.

3.4 Reduced–Order Model

As was highlighted in Section 2, Reduced–Order methods are efficient tools for significant reduction of the parameter–dependent PDEs. This section deals with the reduced–order model 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. First, we introduce two practical ingredients we will be using in the course of the reduced–basis generation, namely a lifting function and the pressure supremiser enrichment. Then, we describe the offline phase based on the Proper Orthogonal Decomposition technique, which is followed by the online phase based on a Galerkin projection onto the reduced spaces.

3.4.1 Lifting Function and Velocity Supremiser Enrichment

In the following, we are going to discuss a snapshot compression technique for the generation of reduced basis functions. In order to do so we need to introduce two important ingredients in this context, namely the lifting function technique and the supremiser enrichment of the velocity space.

The use of lifting functions is quite common in the reduced basis method (RBM) framework; see, for example, [81, 17]. It is motivated by the fact that in the chosen model we are supposed to tackle the non–homogeneous Dirichlet boundary condition 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, particularly 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 which is obtained by the compression of the set of snapshots, and this clearly cannot be achieved by using snapshots that satisfy different Dirichlet conditions – the resulting space would not be linear. This problem is solved by introducing lifting functions $l_{i,h} \in V_{i,h}$, $i = 1, 2$, during the offline stage, such that $l_{i,h} = u_{i,D,h}$ on $\Gamma_{i,D}$. We define two new variables $u_{i,0,h} \in V_{i,0,h}$, $i = 1, 2$, by setting $u_{i,0,h} := u_{i,h} - l_{i,h}$. Clearly, the variables $u_{i,0,h}$, $i = 1, 2$, satisfy the homogeneous condition $u_{i,0,h} = 0$ on $\Gamma_{i,D}$ and 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 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.

The other ingredient we will use in the following exposition is the so-called velocity supremiser. This is necessary to obtain a stable approximation of the saddle–point problem at the reduced level discussed in the following subsections. 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 (3.16). 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 $s_{i,h}$, $i = 1, 2$, are defined as the solution to the following problem: find $s_{i,h} \in V_{i,0,h}$ such that

$$\left(\nabla v_{i,h}, \nabla s_{i,h}\right) = b_{i,h}\left(v_{i,h}, p_{i,h}\right) \quad \forall v_{i,h} \in V_{i,0,h}, \tag{3.21}$$

where $p_{i,h}$, $i = 1, 2$, are the finite–element pressure solutions of the Navier–Stokes problem and the left–hand side is the scalar product which defines a norm on the space $V_{i,0,h}$. For more details, we refer to [17, 64].

3.4.2 Reduced Basis Generation

Once we obtain the homogenised snapshots $u_{i,0,h}$ and the pressure 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 Proper Orthogonal Decomposition (POD) technique, which is based on the Singular Value Decomposition of the snapshot matrices; see, for instance, [81]. In order to implement this technique we will need two main ingredients: the matrices of the inner products and the snapshot matrices. First, we define the basis functions for the FE element spaces used in the weak formulation (3.17), (3.18) and (3.20) as follows:

$$\begin{aligned}
\mathcal{U}_{i,0,h} &= \left\{\phi_1^{u_i}, ..., \phi_{\mathcal{N}_h^{u_i}}^{u_i}\right\} - \text{the FE basis of the space } V_{i,0,h}, i = 1, 2, \\
\mathcal{P}_{i,h} &= \left\{\phi_1^{p_i}, ..., \phi_{\mathcal{N}_h^{p_i}}^{p_i}\right\} - \text{the FE basis of the space } Q_{i,h}, i = 1, 2, \\
\Xi_{i,0,h} &:= \mathcal{U}_{i,0,h}, \quad \mathcal{N}_h^{\xi_i} := \mathcal{N}_h^{u_i}, i = 1, 2, \\
\mathcal{G}_{i,h} &= \left\{\phi_1^{g}, ..., \phi_{\mathcal{N}_h^{g}}^{g}\right\} - \text{the FE basis of the space } X_h,
\end{aligned}$$

where $\mathcal{N}_h^*, * \in \{u_1, p_1, u_2, p_2, g\}$ denotes the dimension of the corresponding FE space.

We proceed by building the snapshot matrices. In doing so, we sample a parameter space and draw a discrete set of M parameter values; there are various sampling techniques, among which we

point out the uniform sampling. Then, the snapshots are taken as high–fidelity, i.e. Finite Element, solutions at each parameter value in the sampling set.

We proceed by building the snapshot matrices $\mathcal{S}_{u_i} \in \mathbb{R}^{\mathcal{N}_h^s \times 4M}$, $\mathcal{S}_{s_i} \in \mathbb{R}^{\mathcal{N}_h^s \times 4M}$, $\mathcal{S}_{p_i} \in \mathbb{R}^{\mathcal{N}_h^s \times 4M}$, $\mathcal{S}_{\xi_i} \in \mathbb{R}^{\mathcal{N}_h^a \times 2M}$ for $i = 1, 2$ and $\mathcal{S}_g \in \mathbb{R}^{\mathcal{N}_h^g \times M}$ defined as follows:

$$
\begin{aligned}
\mathcal{S}_{u_1} &= [u^1_{1,0,h}, ..., u^M_{1,0,h}, 0, ..., 0, 0, ..., 0, 0, ..., 0], \\
\mathcal{S}_{s_1} &= [s^1_{1,h}, ..., s^M_{1,h}, 0, ..., 0, 0, ..., 0, 0, ..., 0], \\
\mathcal{S}_{p_1} &= [0, ..., 0, p^1_{1,h}, ..., p^M_{1,h}, 0, ..., 0, 0, ..., 0], \\
\mathcal{S}_{u_2} &= [0, ..., 0, 0, ..., 0, u^1_{2,0,h}, ..., u^M_{2,0,h}, 0, ..., 0], \\
\mathcal{S}_{s_2} &= [0, ..., 0, 0, ..., 0, s^1_{2,h}, ..., s^M_{2,h}, 0, ..., 0], \\
\mathcal{S}_{p_2} &= [0, ..., 0, 0, ..., 0, 0, ..., 0, p^1_{2,h}, ..., p^M_{2,h}], \\
\mathcal{S}_{\xi_1} &= [\xi^1_{1,h}, ..., \xi^M_{1,h}, 0, ..., 0], \quad \mathcal{S}_{\xi_2} = [0, ..., 0, \xi^1_{2,h}, ..., \xi^M_{2,h}], \\
\mathcal{S}_g &= [g^1_h, ..., g^M_h],
\end{aligned}
$$

where $\mathcal{N}_h^s = \mathcal{N}_h^{u_1} + \mathcal{N}_h^{p_1} + \mathcal{N}_h^{u_2} + \mathcal{N}_h^{p_2}$, $\mathcal{N}_h^a = \mathcal{N}_h^{\xi_1} + \mathcal{N}_h^{\xi_2}$ and M is the number of snapshots.

Notice that since all the snapshots of the variables $\xi_{1,h}$ and $\xi_{2,h}$ are divergence–free on the domain of definition, the reduced spaces constructed for those variables will already contain this information, so that it allows us not to store the snapshots of the variables $\lambda_{1,h}$ and $\lambda_{2,h}$, which are playing the role of the Lagrange multipliers relative to the divergence free–conditions, as they do not contain any important information.

The next step is to define the inner–product matrices X_{u_i}, X_{p_i}, X_{ξ_i} for $i = 1, 2$ and X_g. These matrices have the block diagonal structure as follows:

$$
\begin{aligned}
X_{u_1} &= \operatorname{diag}\left(x_{u_1}, 0_{p_1}, 0_{u_2}, 0_{p_2}\right), \\
X_{p_1} &= \operatorname{diag}\left(0_{u_1}, x_{p_1}, 0_{u_2}, 0_{p_2}\right), \\
X_{u_2} &= \operatorname{diag}\left(0_{u_1}, 0_{p_1}, x_{u_2}, 0_{p_2}\right), \\
X_{p_2} &= \operatorname{diag}\left(0_{u_1}, 0_{p_1}, 0_{u_2}, x_{p_2}\right), \\
X_{\xi_1} &= \operatorname{diag}\left(x_{u_1}, 0_{\xi_2}\right), \\
X_{\xi_2} &= \operatorname{diag}\left(0_{\xi_1}, x_{u_2}\right), \\
X_g &= x_g.
\end{aligned}
$$

Above, we used the following notations: $0_* \in \mathbb{R}^{\mathcal{N}_h^* \times \mathcal{N}_h^*}$ is a zero square matrix of dimension $N_h^* \times \mathcal{N}_h^*$, where $* \in \{u_1, p_1, u_2, p_2, \xi_1, \xi_2, g\}$ and

$$
\begin{aligned}
(x_{u_i})_{jk} &= \left(\nabla\phi_k^{u_i}, \nabla\phi_j^{u_i}\right)_{\Omega_i}, \quad \text{for } j, k = 1, ..., \mathcal{N}_h^{u_i},\ i = 1, 2, \\
(x_{p_i})_{jk} &= \left(\phi_k^{p_i}, \phi_j^{p_i}\right)_{\Omega_i}, \quad \text{for } j, k = 1, ..., \mathcal{N}_h^{p_i},\ i = 1, 2, \\
(x_g)_{jk} &= \left(\phi_k^g, \phi_j^g\right)_{\Gamma_0}, \quad \text{for } j, k = 1, ..., \mathcal{N}_h^g.
\end{aligned}
$$

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

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

for every $* \in \{u_1, p_1, u_2, p_2, \xi_1, \xi_2, g\}$ and

$$C_{s_i} := \mathcal{S}_{s_i}^T X_{u_i} S_{s_i},\ i = 1, 2.$$

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:

$$C_* \mathcal{Q}_* = \mathcal{Q}_* \Lambda_*, \tag{3.22}$$

where $* \in \{u_1, s_1, p_1, u_2, s_2, p_2, \xi_1, \xi_2, g\}$, $\mathcal{Q}_*$ 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 $\mathcal{Q}_*$:

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

where λ_k^* is the k–th eigenvalue from (3.22). Therefore, we are able to form the set of reduced bases as

$$\mathcal{A}^s := \bigcup_{*\in\{u_1,s_1,p_1,u_2,s_2,p_2\}} \left\{\Psi_1^*, ..., \Psi_{N_*}^*\right\},$$

$$\mathcal{A}^a := \bigcup_{*\in\{\xi_1,\xi_2\}} \left\{\Psi_1^*, ..., \Psi_{N_*}^*\right\},$$

$$\mathcal{A}^g := \left\{\Phi_1^g, ..., \Phi_{N_g}^g\right\},$$

where the integer numbers N_* indicate the number of the basis functions used for each component and

$$\Psi_k^{u_1} = \begin{pmatrix} \Phi_k^{u_1} \\ 0 \\ 0 \\ 0 \end{pmatrix},\ \Psi_k^{s_1} = \begin{pmatrix} \Phi_k^{s_1} \\ 0 \\ 0 \\ 0 \end{pmatrix},\ \Psi_k^{p_1} = \begin{pmatrix} 0 \\ \Phi_k^{p_1} \\ 0 \\ 0 \end{pmatrix},\ \Psi_k^{u_2} = \begin{pmatrix} 0 \\ 0 \\ \Phi_k^{u_2} \\ 0 \end{pmatrix},$$

$$\Psi_k^{s_2} = \begin{pmatrix} 0 \\ 0 \\ \Phi_k^{s_2} \\ 0 \end{pmatrix},\ \Psi_k^{p_2} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \Phi_k^{p_2} \end{pmatrix},\ \Psi_k^{\xi_1} = \begin{pmatrix} \Phi_k^{\xi_1} \\ 0 \end{pmatrix},\ \Psi_k^{\xi_2} = \begin{pmatrix} 0 \\ \Phi_k^{\xi_2} \end{pmatrix}.$$

We note that the first and the third blocks include both the u_1, s_1 and the u_2, s_2 basis functions – it is here that we use the pressure supremiser enrichment of the velocities spaces discussed at

the beginning of this section. We provide the following renumbering of the functions for further simplicity:

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

and we redefine $N_{u_i} := N_{u_i} + N_{s_i},\ i = 1, 2$.

Finally, we introduce three separate reduced basis spaces – for the state, the adjoint and the control variables, respectively:

$$\begin{aligned} V^s_N &= \text{span}\mathcal{A}^s, \qquad \text{dim}V^s_N = N_{u_1} + N_{p_1} + N_{u_2} + N_{p_2}, \\ V^a_N &= \text{span}\mathcal{A}^a, \qquad \text{dim}V^s_N = N_{\xi_1} + N_{\xi_2}, \\ V^g_N &= \text{span}\mathcal{A}^g, \qquad \text{dim}V^s_N = N_g. \end{aligned}$$

3.4.3 Online Phase

Once we have introduced the reduced basis spaces, we can define the reduced function expansions

$$U_N = (u_{1,0,N}, p_{1,N}, u_{2,0,N}, p_{2,N}) \in V^s_N, \Xi_N = (\xi_{1,N}, \xi_{2,N}) \in V^a_N, g_N \in V^g_N$$

as

$$u_{i,0,N} := \sum_{k=1}^{N_{u_i}} \underline{u}_{i,0,k}\Phi^{u_i}_k,\ i = 1, 2, \qquad \xi_{i,N} := \sum_{k=1}^{N_{\xi_i}} \underline{\xi}_{i,k}\Phi^{\xi_i}_k,\ i = 1, 2,$$

$$p_{i,N} := \sum_{k=1}^{N_{p_i}} \underline{p}_{i,k}\Phi^{p_i}_k,\ i = 1, 2, \qquad g_N := \sum_{k=1}^{N_g} \underline{g}_k\Phi^g_k.$$

In the previous equations, the underscore indicates the coefficients of the basis expansion of the reduced solution. Then, the online reduced problem reads as follows: minimise over $g_N \in V^g_N$ the functional

$$\mathcal{J}_{\gamma,N}(u_{1,N}, u_{2,N}; g_N) := \frac{1}{2}\int_{\Gamma_0} \left|u_{1,N} - u_{2,N}\right|^2 d\Gamma + \frac{\gamma}{2}\int_{\Gamma_0} |g_N|^2\, d\Gamma, \tag{3.23}$$

where $u_{1,N} = u_{1,0,N} + l_{1,N}$, $u_{2,N} = u_{2,0,N} + l_{2,N}$ for $(u_{1,0,N}, p_{1,N}, u_{2,0,N}, p_{2,N}) \in V^s_N$ satisfy the following reduced equations $\forall v_N = (v_{1,N}, q_{1,N}, v_{2,N}, q_{2,N}) \in V^s_N$

$$\begin{aligned} a_i(u_{i,0,N}, v_{i,N}) \;&+\; c_i(u_{i,0,N}, u_{i,0,N}, v_{i,N}) + c_i(u_{i,0,N}, l_{i,N}, v_{i,N}) \\ &+\; c_i(l_{i,N}, u_{i,0,N}, v_{i,N}) + b_i(v_{i,N}, p_{i,N}) \\ &=\; (f_i, v_{i,N})_{\Omega_i} + ((-1)^{i+1} g_N, v_{i,N})_{\Gamma_0} \qquad (3.24a) \\ &\quad\; -a_i(l_{i,N}, v_{i,N}) - c_i(l_{i,N}, l_{i,N}, v_{i,N}), \\ b_i(u_{i,0,N}, q_{i,N}) \;&=\; -b_i(l_{i,N}, q_{i,N}), \qquad (3.24b) \end{aligned}$$

where $l_{i,N}$ is the Galerkin projection of the lifting function $l_{i,h}$ to the finite dimensional vector space spanned by the i–th velocity basis functions and $i = 1, 2$.

Similarly to the offline phase, we notice that the structure of the equations (3.24) and the functional (3.23) 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 (3.23), i.e.,

$$\frac{d\mathcal{J}_{\gamma,N}}{dg_N}(u_{1,N}, u_{2,N}; g_N) = \gamma g_N + (\xi_{1,N} - \xi_{2,N})|_{\Gamma_0}, \tag{3.25}$$

where $(\xi_{1,N}, \xi_{2,N}) \in V_N^a$ are the solutions to the reduced adjoint problem: find $(\xi_{1,N}, \xi_{2,N}) \in V_N^a$ such that it satisfies, for each pair of test functions $(\eta_{1,N}, \eta_{2,N}) \in V_N^a$ and $i = 1, 2$,

$$\begin{aligned} a_i(\eta_{i,N}, \xi_{i,N}) + c_i\left(\eta_{i,N}, u_{i,N}, \xi_{i,N}\right) + c_i\left(u_{i,N}, \eta_{i,N}, \xi_{i,N}\right) \\ = ((-1)^{i+1}\eta_{i,N}, u_{1,N} - u_{2,N})_{\Gamma_0}. \end{aligned} \tag{3.26}$$

Notice that the reduced adjoint equations no longer contain any terms corresponding to the bilinear forms $b_i(\cdot,\cdot)$, $i = 1, 2$. Indeed, as was previously mentioned, all the functions belonging to the reduced space V_N^a are already divergence–free by construction, so the aforementioned terms are automatically satisfied.

We would also like to stress that from the numerical implementation point of view 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 function used for the control variable g_N in the online phase, that is $p = N_g$.

3.5 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 multiphenics [2], whereas the online phase simulations were carried out using RBniCS [3].

3.5.1 Backward–facing step test case

We start with introducing the backward–facing step flow test case. Figure 3.3 represents the physical domain of interest. The upper part of the channel has a length of 18 cm, the lower part 14 cm; the height of the left chamber is 3 cm, and the height of the right one is 5 cm. The splitting into two domains is performed by dissecting the domain by a vertical segment at the distance $\frac{26}{3}$ cm from the beginning of the channel as shown in Figure 3.4.

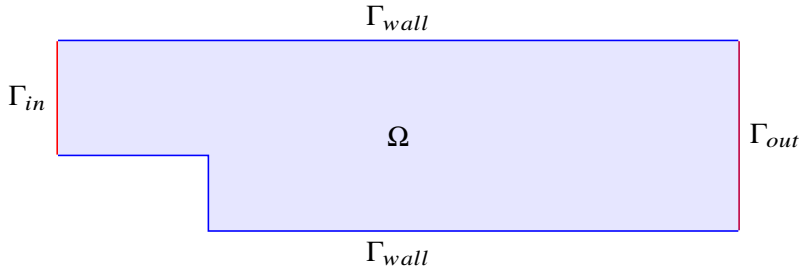


Figure 3.3: Physical domain for the backward–facing step problem

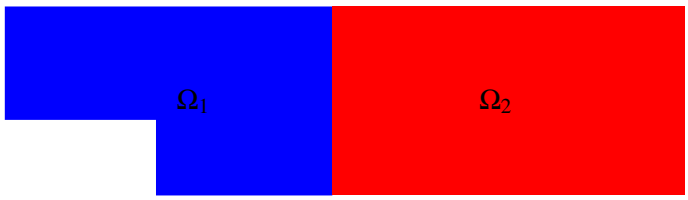


Figure 3.4: Domain decomposition for the backward–facing step problem domain

We impose homogeneous Dirichlet boundary conditions on the top and the bottom walls of the boundary 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

$$u_{in}(x, y) = \begin{pmatrix} w(y) \\ 0 \end{pmatrix} \tag{3.27}$$

with $w(y) = \bar{U} \times \frac{4}{9}(y-2)(5-y)$, $y \in [2, 5]$; values of $\bar{U}$ are reported in Table 3.1. Two physical parameters are considered: the viscosity ν and the maximal magnitude $\bar{U}$ of the inlet velocity profile u_{in}. Both parameters concur to the definition of the only physically relevant parameter, the Reynolds number $Re = L\frac{\bar{U}}{\nu}$, where L is the characteristic length. Hence, we indicate for all tests also the corresponding Re. Details of the offline stage and the finite–element discretisation are summarised in Table 3.1. 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 in our test case. The best performance has been achieved by using the limited–memory Broyden–Fletcher–Goldfarb–Shanno (L–BFGS–B) optimisation algorithm, and two stopping criteria were applied: either the maximal number of iteration $\text{it}_{\max}$ is reached or the gradient norm of the target functional is less than the given tolerance tol_{opt}.

Snapshots are sampled from a training set of M parameters uniformly distributed in the two–dimensional parameter space, and the first $N_{\max}$ POD modes have been retained. Figure 3.5a shows the POD singular values for all the state, the adjoint and the control variables. As it can be seen,

Physical parameters	$2 : \nu, \bar{U}$
Range ν	[0.5, 2]
Range $\bar{U}$	[0.5, 6.5]
Resulting Re number	[0.75, 40]
FE velocity order	2
FE pressure order	1
Total number of FE dofs	27,890
Number of FE dofs at the interface	130
Optimisation algorithm	L-BFGS-B
it_{max}	40
tol_{opt}	10^{-5}
M	900
N_{max}	50

Table 3.1: Computational details of the offline stage

the POD singular values corresponding to the adjoint velocities ξ_1 and ξ_2 feature a slower decay compared to the one for the other variables. In Figure 3.5b, we can see the behaviour of the energy E_n retained by the first N modes for different components of the solution. Here, the retained energy for the component $* \in \{u_1, s_1, p_1, u_2, s_2, p_2, \xi_1, \xi_2, g\}$ is defined as

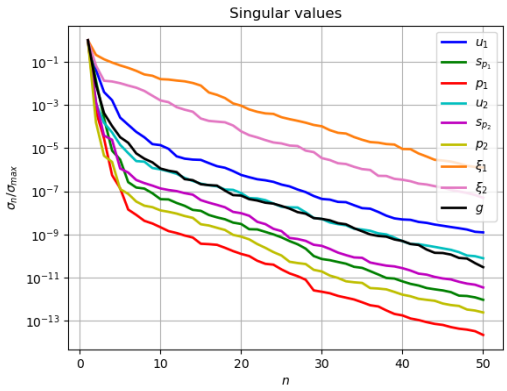
$$E_n^* := \frac{\sum_{k=1}^{n} |\lambda_k^*|}{\sum_{k=1}^{N_*} |\lambda_k^*|}.$$

The retained energy gives us an idea of the number of modes we would need to choose to preserve all the necessary physical information in the reduced model. In particular, we can see that a higher number of modes is needed to correctly represent the adjoint variables ξ_1 and ξ_2.

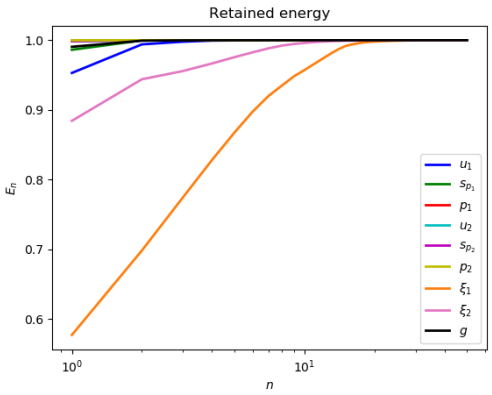
Figures 3.6– 3.9 represent the first four POD modes for each of the variables u_1, u_2, s_1, s_2, p_1, p_2, ξ_1 and ξ_2. We stress that the POD modes were obtained separately for each component and the resulting figures are obtained by gluing the subdomain function just for the sake of visualisation.

Figure 3.6 shows the first modes for the fluid velocities u_1 and u_2: in particular, notice that the modes corresponding to u_1 (on the left section of the domain) are zero at the inlet boundary due to the use of lifting function.

In Figure 3.7, we can see the first four modes for s_1 and s_2: here, the corresponding functions are mostly localised inside the domains Ω_1 and Ω_2 thanks to the homogeneous conditions at the boundaries and the non–zero forcing term coming from the pressure.

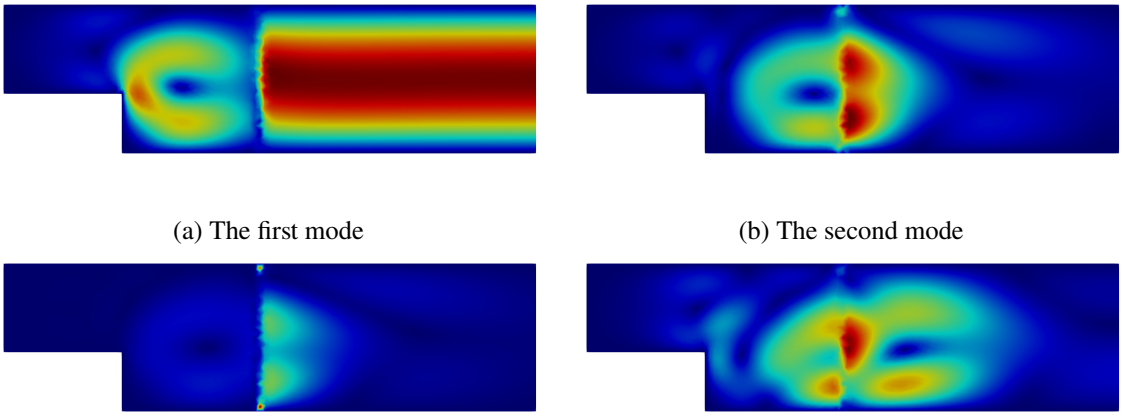


(a) POD singular values as a function of number n of POD modes (log scaling in y–direction)



(b) Energy retained by the first $N_{\max}$ POD modes (log scaling in x–direction)

Figure 3.5: Results of the offline stage: POD singular eigenvalue decay (a) and retained energy (b) of the first $N_{\max}$ POD modes.



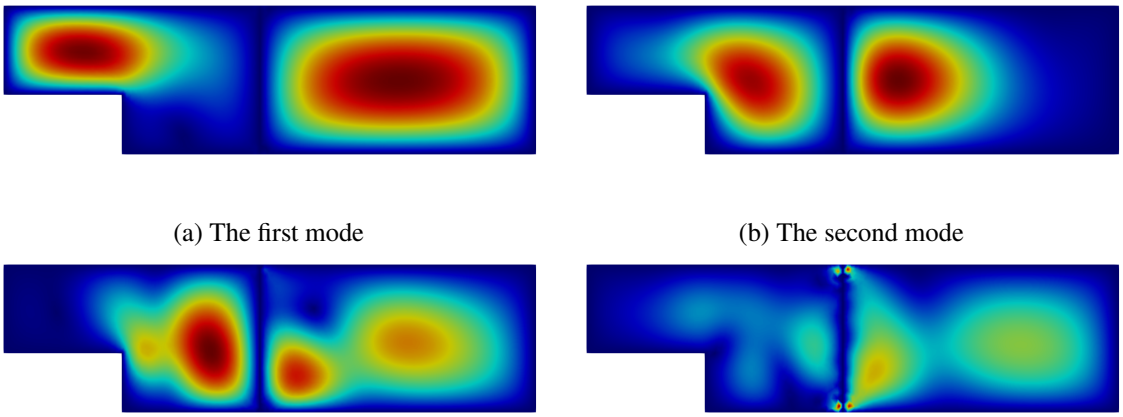
(a) The first mode

(b) The second mode

(c) The third mode

(d) The fourth mode

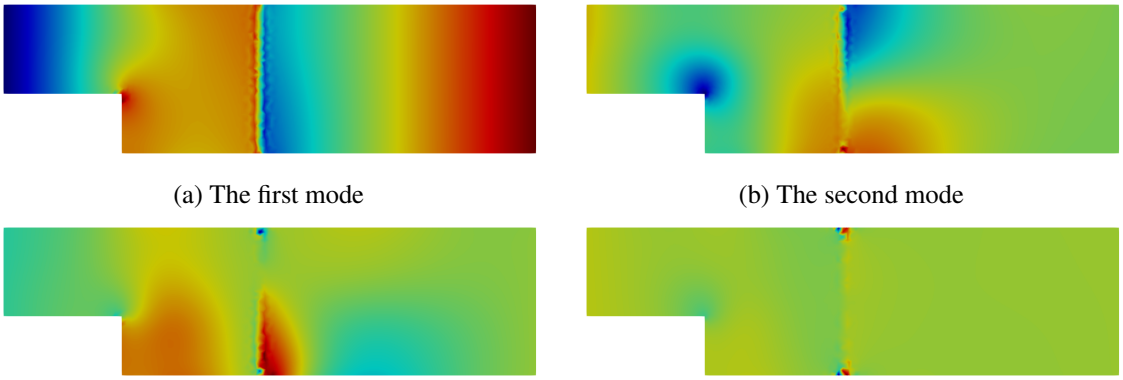
Figure 3.6: The first POD modes for the velocities u_1 and u_2 (subdomain functions are glued together for visualisation purposes).



(a) The first mode (b) The second mode

(c) The third mode (d) The fourth mode

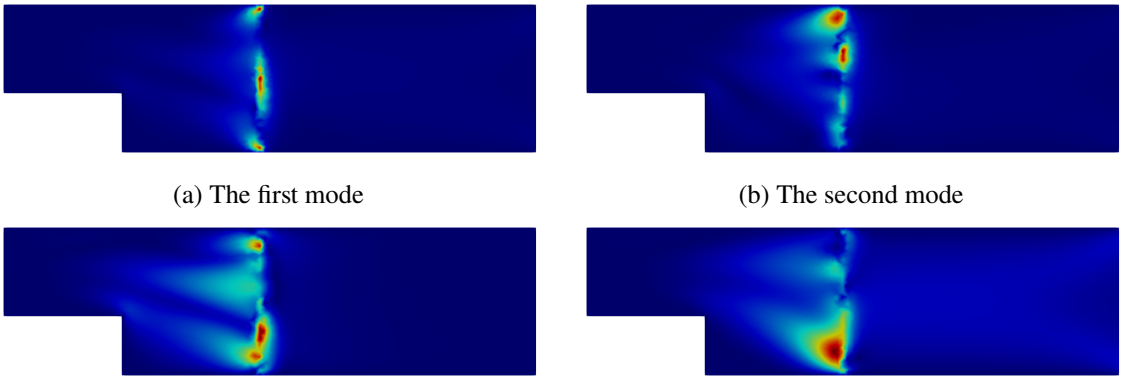
Figure 3.7: The first POD modes for the pressure supremisers s_1 and s_2 (subdomain functions are glued together for visualisation purposes).



(a) The first mode (b) The second mode

(c) The third mode (d) The fourth mode

Figure 3.8: The first POD modes for the pressures p_1 and p_2 (subdomain functions are glued together for visualisation purposes).



(a) The first mode (b) The second mode

(c) The third mode (d) The fourth mode

Figure 3.9: The first POD modes for the adjoint velocities ξ_1 and ξ_2 (subdomain functions are glued together for visualisation purposes).

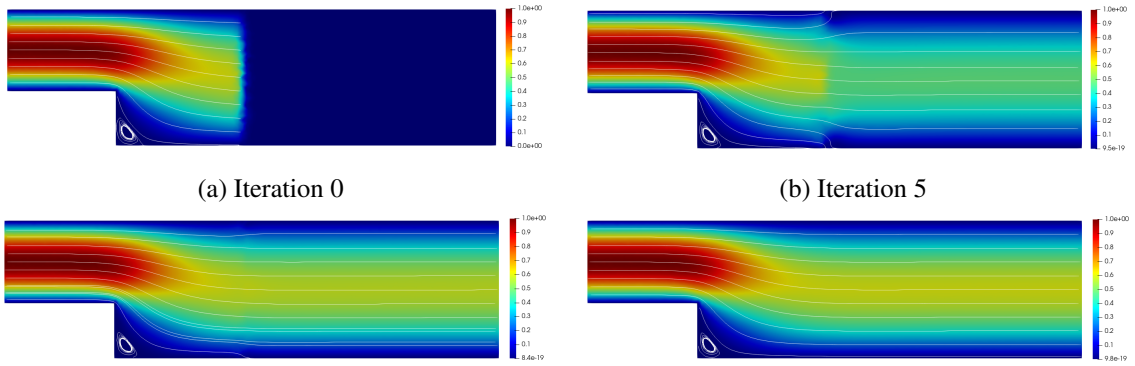
Iteration	Functional Value	Gradient norm
0	$4.8 \cdot 10^{-1}$	$4.1 \cdot 10^{-1}$
5	$6.0 \cdot 10^{-2}$	$2.2 \cdot 10^{-1}$
10	$5.0 \cdot 10^{-3}$	$3.3 \cdot 10^{-2}$
40	$1.7 \cdot 10^{-4}$	$2.4 \cdot 10^{-3}$

Table 3.2: Functional values and the gradient norm for the FOM optimisation solution at the parameter values $\bar{U} = 1$, $\nu = 1$ and $Re = 3$

Figure 3.8 represents the first modes for the pressures p_1 and p_2: we point out the signs of the oscillation behaviour, which suggests that the supremiser enrichment might be needed to assure stability of the reduced–order solution.

Finally, Figure 3.9 shows the first four modes for the adjoint variables ξ_1 and ξ_2: note that they are concentrated only around the interface Γ_0 because the only nonzero contribution in the adjoint equations is coming from the source terms, which are defined solely on the interface Γ_0.

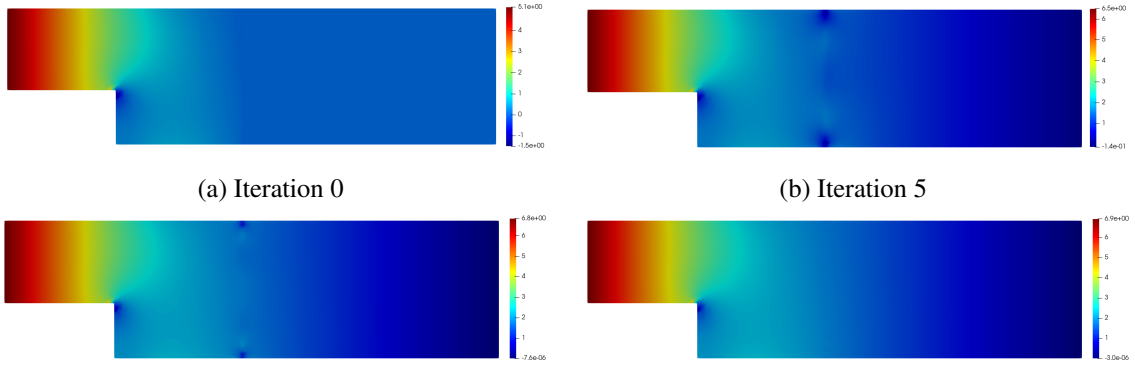
Figures 3.10– 3.13 represent the high–fidelity solutions for two different values of the parameters $(\bar{U}, \nu) = (1, 1)$, resulting in $Re = 3$, and $(\bar{U}, \nu) = (4.5, 0.7)$ with $Re \approx 19$. The solutions were obtained by carrying out 40 optimisation iterations via the L–BFGS–B algorithm. Figures 3.10 and 3.12 show the intermediate solutions at iteration 0, 5, 10 and 40 for the fluid velocities u_1 and u_2, whereas Figures 3.11 and 3.13 show the corresponding pressures p_1 and p_2. The final solution is taken to be the 40th iteration optimisation solution in which we can observe a continuity between subdomain solutions at the interface Γ_0. Moreover, it can be noticed that the solution for parameters $(\bar{U}, \nu) = (1, 1)$ looks continuous already at iteration 10, which suggests that the convergence of the optimisation algorithm might depend on the Reynolds number.



(a) Iteration 0 (b) Iteration 5

(c) Iteration 10 (d) Iteration 40

Figure 3.10: High–fidelity solution for the velocities u_1 and u_2. Values of the parameters $\bar{U} = 1$, $\nu = 1$ and $Re = 3$.



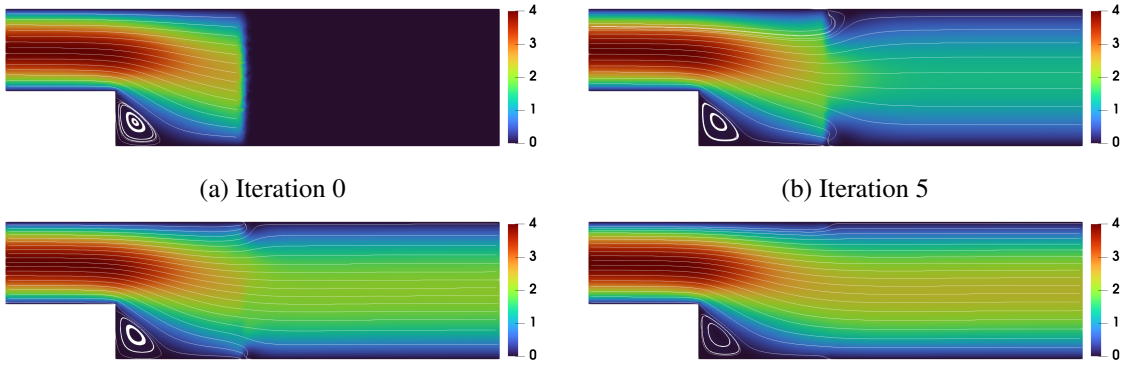
(a) Iteration 0 (b) Iteration 5

(c) Iteration 10 (d) Iteration 40

Figure 3.11: High–fidelity solution for the pressures p_1 and p_2. Values of the parameters $\bar{U} = 1$, $\nu = 1$ and $Re = 3$.

Iteration	**Abs. error** u_h		**Rel. error** u_h		**Abs. error** p_h		**Rel. error** p_h	
	Ω_1	Ω_2	Ω_1	Ω_2	Ω_1	Ω_2	Ω_1	Ω_2
0	0.0302	2.9935	0.0088	1.0000	10.6515	7.0679	0.5056	1.0000
5	0.1020	0.6279	0.0297	0.2098	2.4520	1.5317	0.1164	0.2167
10	0.0384	0.1355	0.0112	0.0453	0.5807	0.3793	0.0276	0.0537
40	0.0184	0.0583	0.0053	0.0195	0.2670	0.1827	0.0127	0.0259

Table 3.3: Absolute and relative errors of the FOM optimisation solution with respect to the monolithic solution at the parameter values $\bar{U} = 1$, $\nu = 1$ and $Re = 3$



(a) Iteration 0 (b) Iteration 5

(c) Iteration 10 (d) Iteration 40

Figure 3.12: High–fidelity solution for the velocities u_1 and u_2. Values of the parameters $\bar{U} = 4$, $\nu = 0.75$ and $Re \approx 19$.

Iteration	Functional Value	Gradient norm
0	7.902	2.213
5	1.956	1.210
10	0.403	2.132
40	0.007	0.069

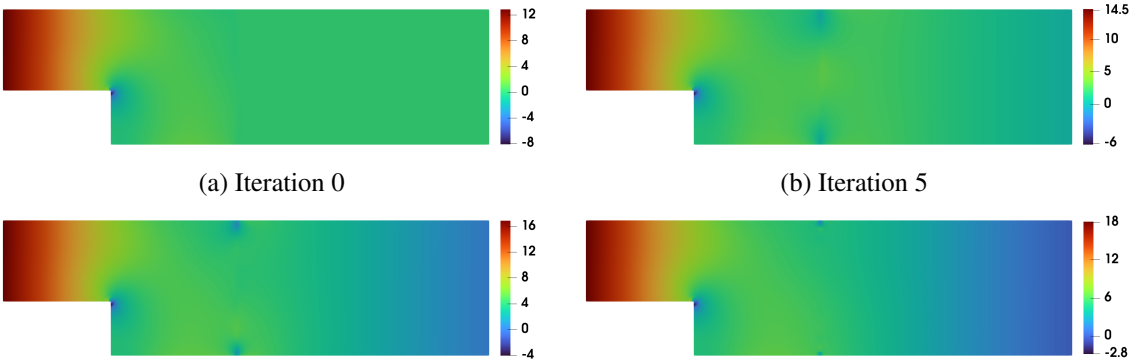
Table 3.4: Functional values and the gradient norm for the FOM optimisation solution at parameter values $\bar{U} = 4$, $\nu = 0.75$ and $Re \approx 19$

We present additional details in Tables 3.2 – 3.5. In particular, in Tables 3.2 and 3.4, we list the values for the functional $\mathcal{J}_\gamma$ and the $L^2(\Gamma_0)$–norm of the gradient $\frac{d\mathcal{J}_\gamma}{dg}$ at the different iteration of the optimisation procedure, while Table 3.3 contains the absolute and relative errors with respect to the monolithic (entire–domain) solutions u_h, p_h, i.e.,

- Abs. error $u_h := ||u_{i,h} - u_h||_{L^2(\Omega_i)}$ on domain Ω_i,
- Rel. error $u_h := \frac{||u_{i,h} - u_h||_{L^2(\Omega_i)}}{||u_h||_{L^2(\Omega_i)}}$ on domain Ω_i,
- Abs. error $p_h := ||p_{i,h} - p_h||_{L^2(\Omega_i)}$ on domain Ω_i,
- Rel. error $p_h := \frac{||p_{i,h} - p_h||_{L^2(\Omega_i)}}{||p_h||_{L^2(\Omega_i)}}$ on domain Ω_i,

for $i = 1, 2$.

Figures 3.14– 3.17 represent the reduced–order solutions for two different values of the parameters $(\bar{U}, \nu) = (1, 1)$ and $Re = 3$ and $(\bar{U}, \nu) = (4, 0.75)$ and $Re \approx 19$. In each of the cases, we choose the following number of the reduced basis functions: $N_{u_1} = N_{s_1} = N_{p_1} = N_{u_2} = N_{s_2} = N_{p_2} = N_g =$



(a) Iteration 0

(b) Iteration 5

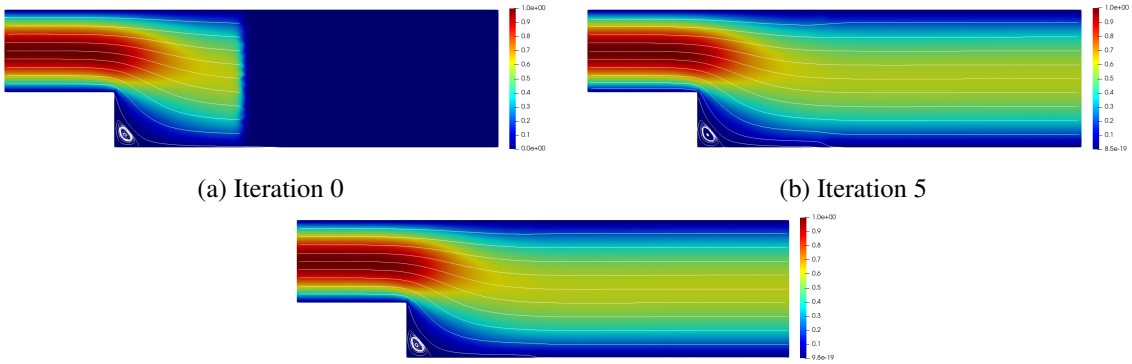
(c) Iteration 10

(d) Iteration 40

Figure 3.13: High–fidelity solution for the pressures p_1 and p_2. Values of the parameters $\bar{U} = 4$, $\nu = 0.75$ and $Re \approx 19$.

Iteration	**Abs. error** u_h		**Rel. error** u_h		**Abs. error** p_h		**Rel. error** p_h	
	Ω_1	Ω_2	Ω_1	Ω_2	Ω_1	Ω_2	Ω_1	Ω_2
0	0.2520	11.9830	0.0181	1.0000	31.6121	21.1630	0.5859	1.0000
5	0.6639	5.0075	0.0478	0.4179	20.7060	10.2359	0.3838	0.4837
10	0.2704	1.3722	0.0195	0.1145	6.7317	2.8262	0.1248	0.1335
40	0.0865	0.2566	0.0062	0.0214	1.4498	0.6443	0.0269	0.0304

Table 3.5: Absolute and relative errors of the FOM optimisation solution with respect to the monolithic solution at the parameter values $\bar{U} = 4$, $\nu = 0.75$ and $Re \approx 19$



(a) Iteration 0

(b) Iteration 5

(c) Iteration 10

Figure 3.14: Reduced order solution for the velocities u_1 and u_2. Values of the parameters $\bar{U} = 1$, $\nu = 1$ and $Re = 3$. Number of POD modes: 10 – for each state variable, each supremiser and the control, 30 – for both adjoint velocities.

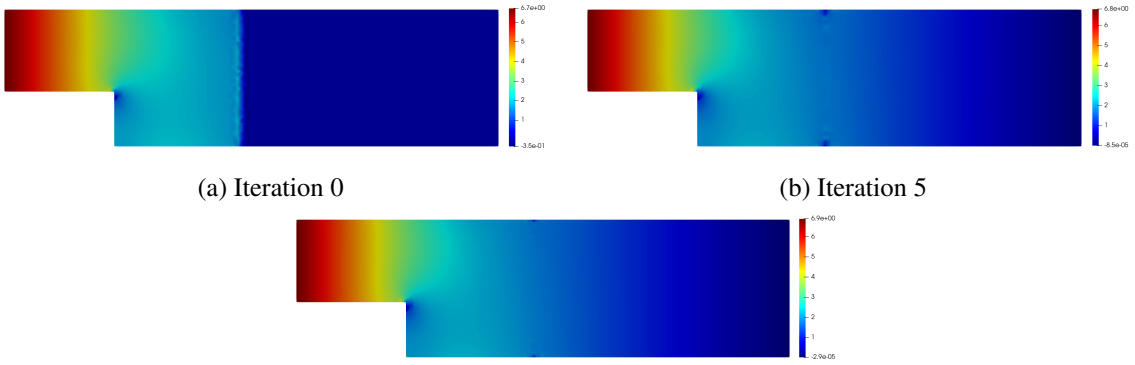
10 and $N_{\xi_1} = N_{\xi_2} = 30$. As was previously anticipated, we use a higher number for the adjoint variables ξ_1 and ξ_2 since they show much slower decay of the singular values (see Figure 3.5a). The solutions were obtained by carrying out 10 optimisation iterations of the L–BFGS–B algorithm. Figures 3.14 and 3.16 show the intermediate solutions at iteration 0, 5 and 10 for the fluid velocities u_1 and u_2, whereas Figures 3.15 and 3.17 show the corresponding pressures p_1 and p_2. The final solution, at the 10th iteration, shows continuity between subdomain solutions at the interface Γ_0.

We present additional details in Tables 3.6– 3.9. In particular, in Tables 3.6 and 3.8, we list the values for the functional $\mathcal{J}_\gamma$ and the $L^2(\Gamma_0)$–norm of the gradient $\frac{d\mathcal{J}_\gamma}{dg}$ at the different iteration of the optimisation procedure, while Table 3.7 and Table 3.9 contain the absolute and relative errors with respect to the monolithic (entire–domain) solutions u_h, p_h, i.e.,

- Abs. error $u_N := ||u_{i,N} - u_h||_{L^2(\Omega_i)}$ on domain Ω_i,
- Rel. error $u_N := \frac{||u_{i,N} - u_h||_{L^2(\Omega_i)}}{||u_h||_{L^2(\Omega_i)}}$ on domain Ω_i,
- Abs. error $p_N := ||p_{i,N} - p_h||_{L^2(\Omega_i)}$ on domain Ω_i,
- Rel. error $p_N := \frac{||p_{i,N} - p_h||_{L^2(\Omega_i)}}{||p_h||_{L^2(\Omega_i)}}$ on domain Ω_i,

for $i = 1, 2$.

Analysing the results, we are able to see that the reduced basis method gives us a solution as accurate as the high–fidelity one. The reduced–order approximation of the optimisation problem at hand allowed us to reduce the dimension of the high–fidelity optimisation functional by more than 10 times and enabled us to use 4 times fewer iterations in the optimisation algorithm (each



(a) Iteration 0

(b) Iteration 5

(c) Iteration 10

Figure 3.15: Reduced order solution for the pressures p_1 and p_2. Values of the parameters $\bar{U} = 1$, $\nu = 1$ and $Re = 3$. Number of POD modes: 10 – for each state variable, each supremiser and the control, 30 – for both adjoint velocities.

Iteration	**Functional Value**	**Gradient norm**
0	$4.8 \cdot 10^{-1}$	0.391
5	$5.4 \cdot 10^{-3}$	0.047
10	$3.6 \cdot 10^{-4}$	0.015

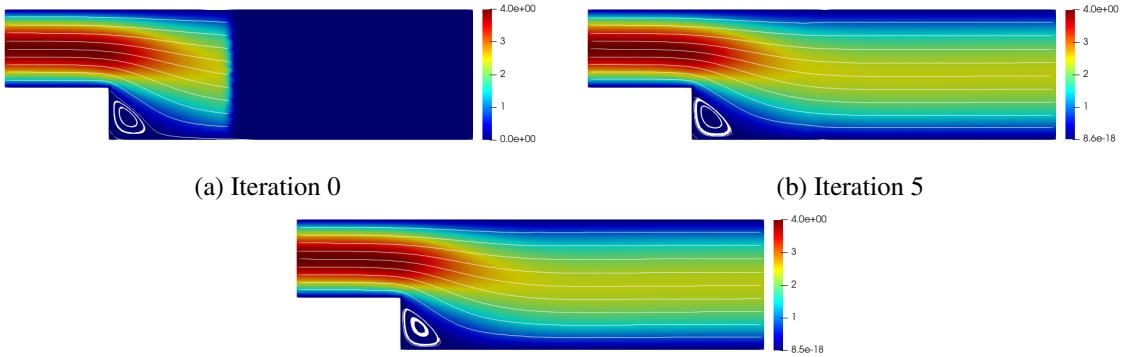
Table 3.6: Functional values and the gradient norm for the ROM optimisation solution at parameter values $\bar{U} = 1$, $\nu = 1$ and $Re = 3$

Iteration	**Abs. error u_N**		**Rel. error u_N**		**Abs. error p_N**		**Rel. error p_N**	
	Ω_1	Ω_2	Ω_1	Ω_2	Ω_1	Ω_2	Ω_1	Ω_2
0	0.0284	2.9935	0.0083	1.0000	10.9522	7.0679	0.5198	1.0000
5	0.0746	0.1956	0.0217	0.0653	0.8548	0.5672	0.0406	0.0803
10	0.0135	0.0357	0.0039	0.0119	0.1714	0.1186	0.0081	0.0168

Table 3.7: Absolute and relative errors of the ROM optimisation solution with respect to the monolithic solution at the parameter values $\bar{U} = 1$, $\nu = 1$ and $Re = 3$

Iteration	**Functional Value**	**Gradient norm**
0	7.869	2.120
5	0.107	0.401
10	0.060	0.555

Table 3.8: Functional values and the gradient norm for the ROM optimisation solution at parameter values $\bar{U} = 4$, $\nu = 0.75$ and $Re \approx 19$

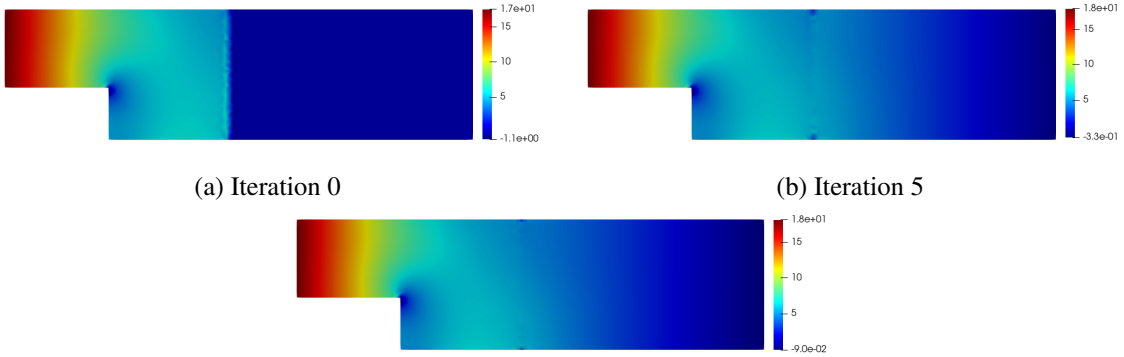


(a) Iteration 0

(b) Iteration 5

(c) Iteration 10

Figure 3.16: Reduced order solution for the velocities u_1 and u_2. Values of the parameters $\bar{U} = 4$, $\nu = 0.75$ and $Re \approx 19$. Number of POD modes: 10 – for each state variable, each supremiser and the control, 30 – for both adjoint velocities.



(a) Iteration 0

(b) Iteration 5

(c) Iteration 10

Figure 3.17: Reduced order solution for the pressures p_1 and p_2. Values of the parameters $\bar{U} = 4$, $\nu = 0.75$ and $Re \approx 19$. Number of POD modes: 10 – for each state variable, each supremiser and the control, 30 – for both adjoint velocities.

Iteration	**Abs. error** u_N		**Rel. error** u_N		**Abs. error** p_N		**Rel. error** p_N	
	Ω_1	Ω_2	Ω_1	Ω_2	Ω_1	Ω_2	Ω_1	Ω_2
0	0.1782	11.9830	0.0128	1.0000	32.5149	21.1630	0.6026	1.0000
5	0.2826	0.8724	0.0204	0.0728	4.1633	1.9392	0.0772	0.0916
10	0.1910	0.3826	0.0138	0.0319	0.6725	0.7453	0.0125	0.0352

Table 3.9: Absolute and relative errors of the ROM optimisation solution with respect to the monolithic solution at the parameter values $\bar{U} = 4$, $\nu = 0.75$ and $Re \approx 19$

Parameter value		**Velocity relative error**		**Pressure relative error**	
$\bar{U}$	ν	Ω_1	Ω_2	Ω_1	Ω_2
1	1	0.024	0.032	0.005	0.012
4	0.75	0.019	0.059	0.021	0.046

Table 3.10: Relative errors between FOM and ROM solutions (in terms of H^1–norm for the velocity fields and L^2–norm for the pressure fields)

optimisation step requires at least one solve of the state and the adjoint equations). We also note that the fact that we chose a bigger number of the reduced basis functions for the adjoint variables ξ_1 and ξ_2 is not supposed to affect the computational costs much since the adjoint problem is linear and does not require multiple Newton iteration to be solved so that the biggest computational effort still lies in the nonlinear Navier–Stokes equations and the optimisation process.

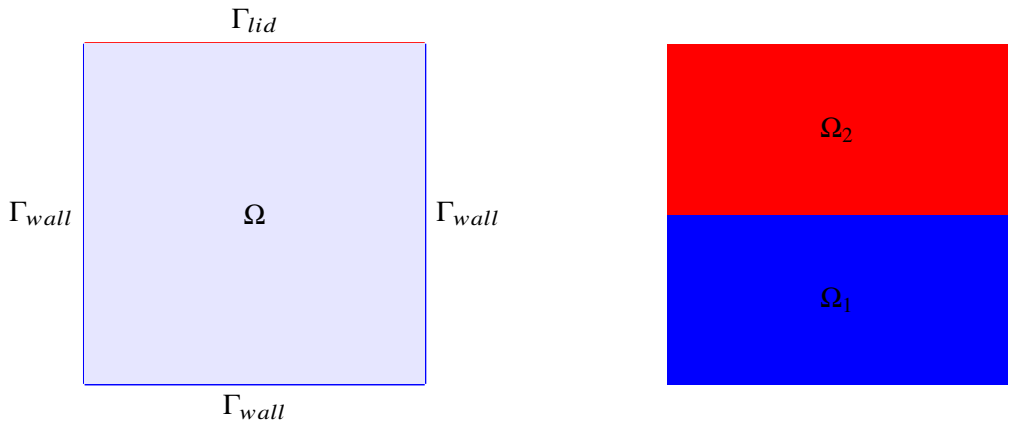
Additionally, in Table 3.10 we provide a comparison between full–order and reduced–order models in terms of the relative errors between ROM solutions with respect to the corresponding FOM solutions. Comparing the convergence results for different models – monolithic vs. DD–FOM, monolithic vs. DD–ROM, and DD–FOM vs. DD–ROM – it can be seen that the DD–ROM method gives a more accurate solution with respect to DD–FOM. We believe that this is due to the optimisation process: the DD–ROM is much less sensitive to the initial guess in the optimisation procedure and much fewer iterations are needed for the optimisation algorithm to converge. Nevertheless, errors between DD–FOM and DD–ROM are comparable to the ones with respect to the monolithic solution.

Remark (High Reynolds and uniqueness of the solution). As it is evident from Table 3.1, the Reynolds number reported for this test case is quite small. This is due to the fact that the optimisation solver diverges for higher Reynolds numbers. The authors suspect that this issue is mostly due to the bifurcation effect (known as the "Coanda effect" or "wall hugging effect" of these types of simulations). One of the reasons to support this argument is that the range of Reynolds numbers for which the optimisation solver converges changes (though not very significantly) when the interface is moved closer to the beginning or the end of the channel. This problem is very complicated in itself and is addressed, for instance, in [163, 102, 34, 80, 138, 139, 141, 145, 133]. In particular, in [139], it is shown that for a similar test already for $Re \approx 78$ there is non–uniqueness of the solution.

3.5.2 Lid–driven cavity flow test case

In this section, we provide the numerical simulation for the lid–driven cavity flow test case. Figure 3.18a represents the physical domain of interest – the unit square. The split into two domains is performed by dissecting the domain by a median horizontal line as shown in Figure 3.18b.

We impose homogeneous Dirichlet boundary conditions on the part of 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 3.11.

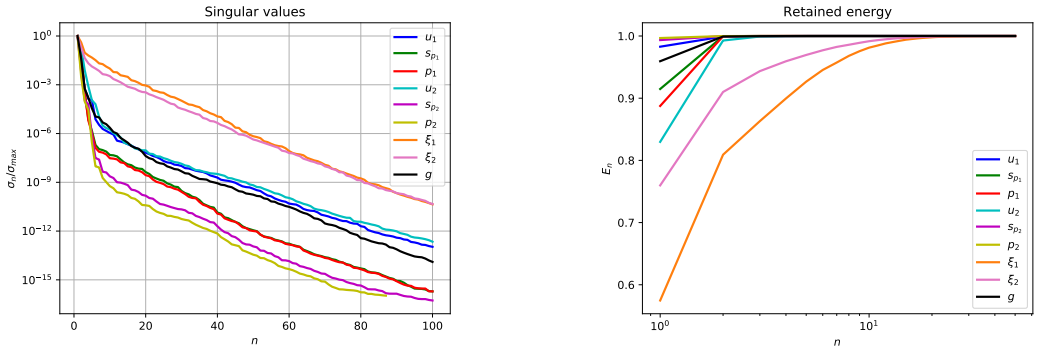


(a) Physical domain (b) Domain splitting

Figure 3.18: Lid–driven cavity flow geometry.

Physical parameters	$2 : \nu, \bar{U}$
Range ν	[0.05, 2]
Range $\bar{U}$	[0.5, 10]
Resulting Re number	[0.25, 200]
FE velocity order	2
FE pressure order	1
Total number of FE dofs	14,867
Number of FE dofs at the interface	138
Optimisation algorithm	L-BFGS-B
it_{max}	100
tol_{opt}	10^{-6}
M	300
N_{max}	100

Table 3.11: Computational details of the offline stage



(a) POD singular values as a function of number n of POD modes (log scaling in y–direction)

(b) Energy retained by the first $N_{\max}$ POD modes (log scaling in x–direction)

Figure 3.19: Results of the offline stage: POD singular eigenvalue decay (a) and retained energy (b) of the first $N_{\max}$ POD modes.

Two physical parameters are considered: viscosity ν and the magnitude $\bar{U}$ of the lid velocity profile u_{in}. Details of the offline stage and the finite–element discretisation are summarised in Table 3.11. 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 138 in our test case. The best performance has been achieved by using the limited–memory Broyden–Fletcher–Goldfarb–Shanno (L–BFGS–B) optimisation algorithm, and two stopping criteria are applied: either the maximal number of iteration $\text{it}_{\max}$ is reached or the gradient norm of the target functional is less than the given tolerance tol_{opt}.

Snapshots are sampled from a training set of M parameters uniformly distributed in the two–dimensional parameter space, and the first $N_{\max}$ POD modes have been retained. Figure 3.19a shows POD singular values for all the state, the adjoint and the control variables. As it can be seen, the POD singular values corresponding to the adjoint velocities ξ_1 and ξ_2 feature a slower decay compared to the one for the other variables. In Figure 3.19b, we can see the behaviour of the energy E_n retained by the first N modes for different components of the solution. Note that, as it was in the previous numerical case, a higher number of modes is needed to correctly represent the adjoint variables ξ_1 and ξ_2.

Figures 3.20– 3.23 represent first three POD modes for the variables $u_1, u_2, s_1, s_2, p_1, p_2$ and ξ_1, ξ_2. We stress that the POD modes were obtained separately for each component and the resulting figures are obtained by gluing the subdomain functions just for the sake of visualisation. Figure 3.20 shows the first modes for the fluid velocities u_1 and u_2. In particular, we notice that the modes corresponding to u_2 (on the upper section of the domain) are zero at the lid boundary due to the use of the lifting function. Figure 3.23 shows the first three modes for the adjoint variables ξ_1 and ξ_2: note that they are concentrated only around the interface Γ_0 because the only nonzero contribution in the adjoint equations is coming from the source terms defined solely on the interface Γ_0.

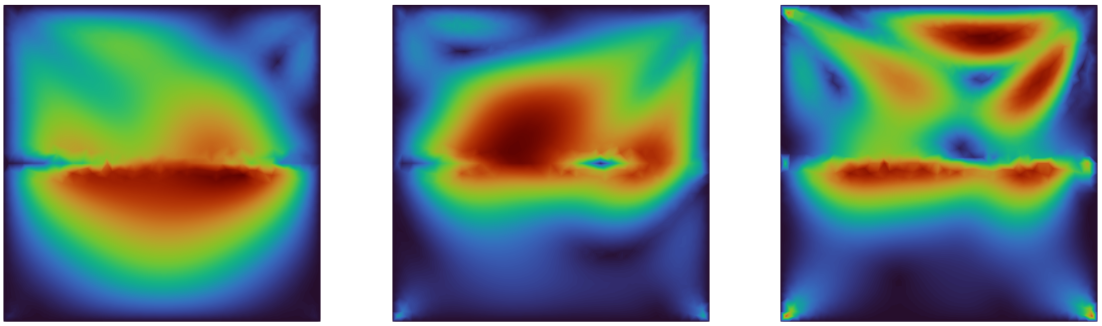


Figure 3.20: The first POD modes for the velocities u_1 and u_2 (subdomain functions are glued together for visualisation purposes)

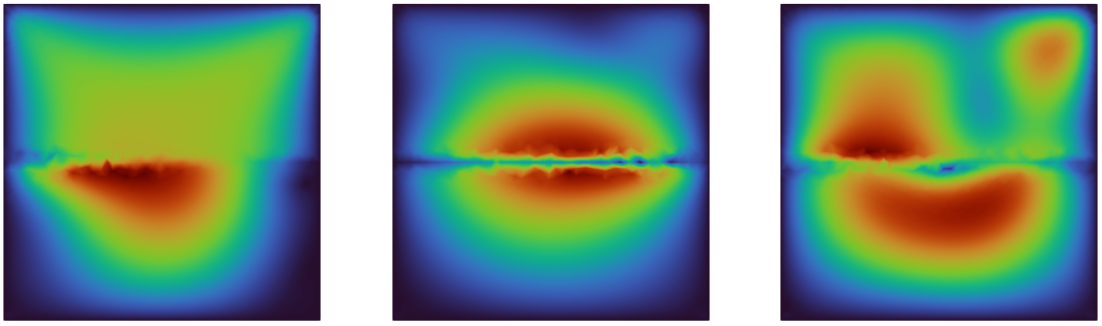


Figure 3.21: The first POD modes for the supremiser variables s_1 and s_2 (subdomain functions are glued together for visualisation purposes)

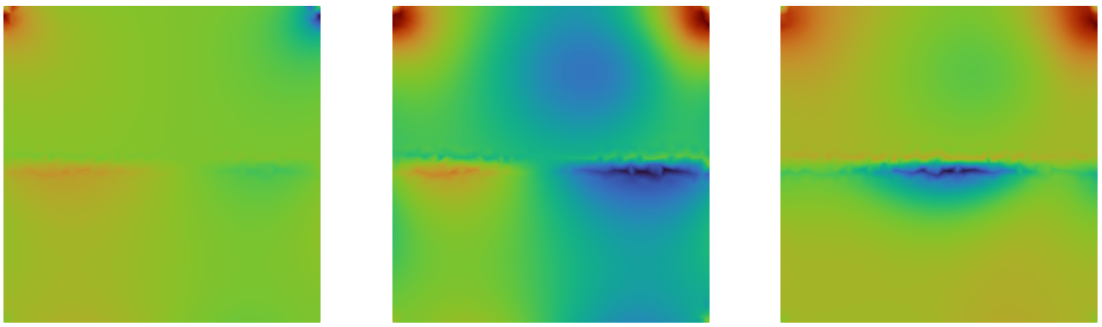


Figure 3.22: The first POD modes for the pressures p_1 and p_2 (subdomain functions are glued together for visualisation purposes)

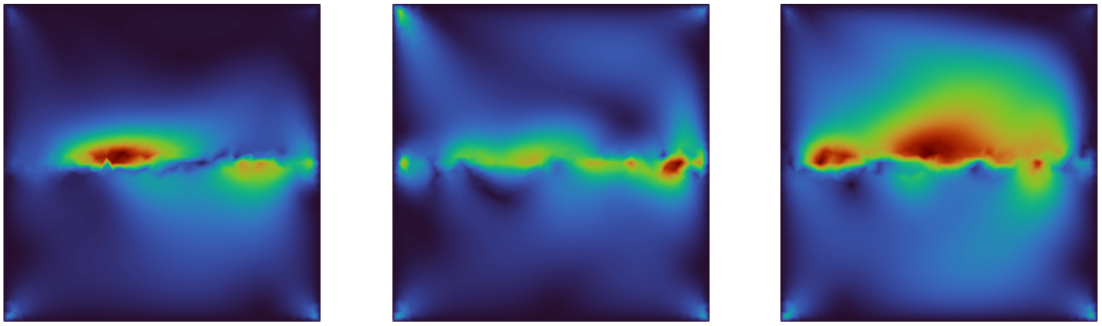
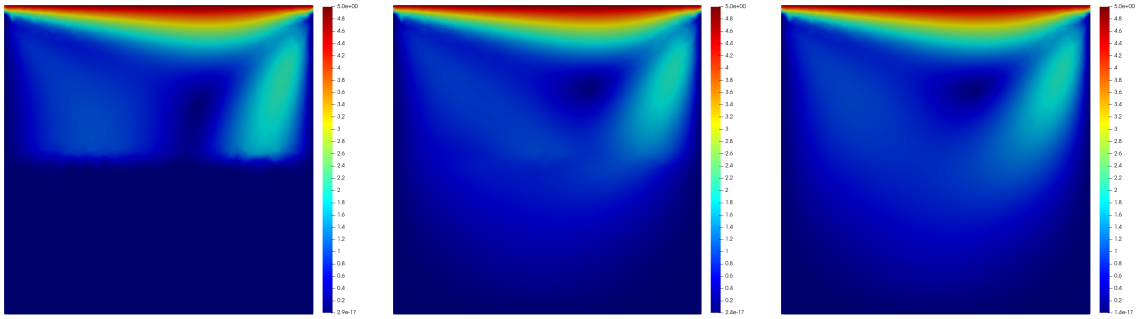
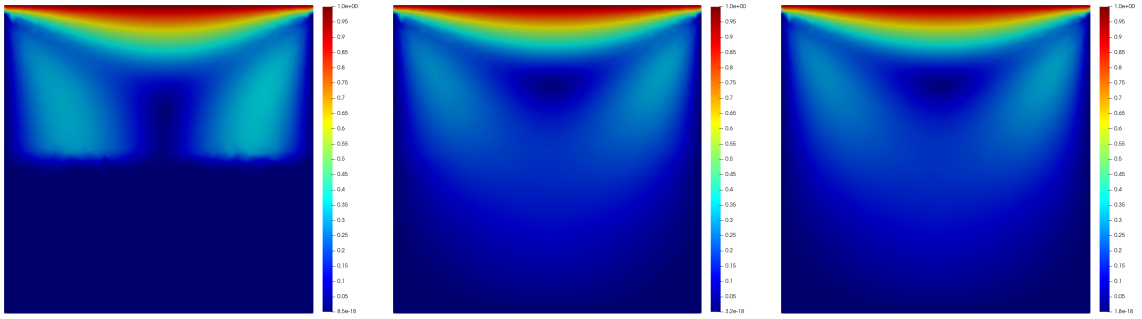


Figure 3.23: The first POD modes for the adjoint velocities ξ_1 and ξ_2 (subdomain functions are glued together for visualisation purposes)



(a) Iteration 0 (b) Iteration 5 (c) Iteration 25

Figure 3.24: High–fidelity solution for the velocities u_1 and u_2. Values of the parameters $\bar{U} = 5$ and $\nu = 0.05$ and with $Re = 100$.



(a) Iteration 0 (b) Iteration 5 (c) Iteration 25

Figure 3.25: High–fidelity FOM solution for the velocities u_1 and u_2. Values of the parameters $\bar{U} = 1$, $\nu = 0.1$ and with $Re = 10$.

Iteration	Functional Value	Gradient norm
0	$4.4 \cdot 10^{-1}$	3.398
5	$3.0 \cdot 10^{-2}$	1.001
10	$3.5 \cdot 10^{-3}$	0.171
25	$8.7 \cdot 10^{-5}$	0.016

Table 3.12: Functional values and the gradient norm for the FOM optimisation solution at parameter values $\bar{U} = 5$, $\nu = 0.05$ and with $Re = 100$

Iteration	Abs. error u_N		Rel. error u_N		Abs. error p_N		Rel. error p_N	
	Ω_1	Ω_2	Ω_1	Ω_2	Ω_1	Ω_2	Ω_1	Ω_2
0	0.3411	0.1949	1.0000	0.1653	0.2689	0.3149	1.0000	0.2330
5	0.0623	0.0613	0.1826	0.0520	0.0531	0.0575	0.3633	0.0426
10	0.0114	0.0136	0.0334	0.0116	0.0184	0.0206	0.1256	0.0153
25	0.0051	0.0062	0.0151	0.0053	0.0143	0.0147	0.0980	0.0109

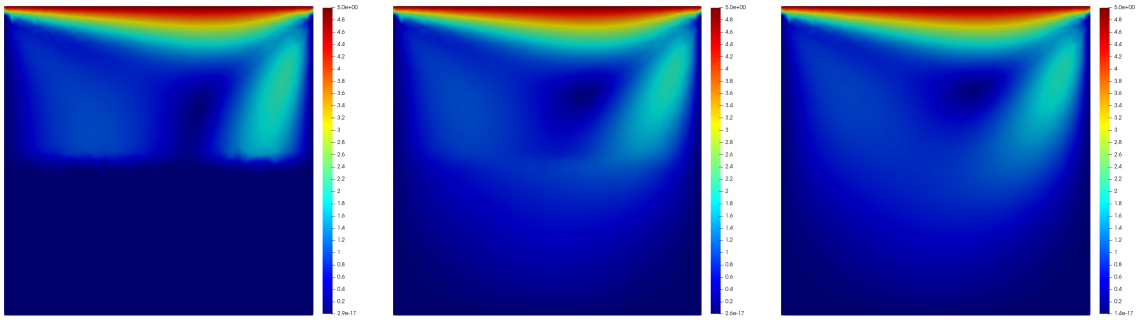
Table 3.13: Absolute and relative errors of the FOM optimisation solution with respect to the monolithic solution at the parameter value $\bar{U} = 5$, $\nu = 0.05$ and with $Re = 100$

Iteration	Functional Value	Gradient norm
0	$2.5 \cdot 10^{-2}$	$4.1 \cdot 10^{-1}$
5	$7.4 \cdot 10^{-5}$	$1.4 \cdot 10^{-2}$
10	$3.3 \cdot 10^{-6}$	$9.1 \cdot 10^{-4}$
25	$7.0 \cdot 10^{-7}$	$3.9 \cdot 10^{-4}$

Table 3.14: Functional values and the gradient norm for the FOM optimisation solution at the parameter values $\bar{U} = 1$, $\nu = 0.1$ and with $Re = 10$

Iteration	Abs. error u_N		Rel. error u_N		Abs. error p_N		Rel. error p_N	
	Ω_1	Ω_2	Ω_1	Ω_2	Ω_1	Ω_2	Ω_1	Ω_2
0	0.0668	0.0589	1.0000	0.2416	0.0349	0.0411	1.0000	0.0956
5	0.0032	0.0028	0.0483	0.0114	0.0036	0.0036	0.1100	0.0084
10	0.0006	0.0006	0.0095	0.0027	0.0024	0.0023	0.0733	0.0054
25	0.0005	0.0005	0.0069	0.0019	0.0021	0.0021	0.0663	0.0048

Table 3.15: Absolute and relative errors of the optimisation FOM solution with respect to the monolithic solution at the parameter value $\bar{U} = 1$, $\nu = 0.1$ and with $Re = 10$



(a) Iteration 0 (b) Iteration 3 (c) Iteration 10

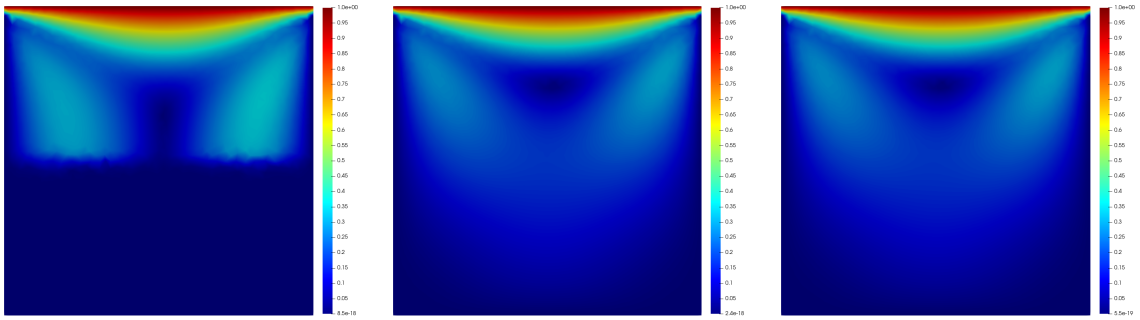
Figure 3.26: Reduced–order solution for the velocities u_1 and u_2. Values of the parameters $\bar{U} = 5$, $\nu = 0.05$ and with $Re = 100$. Number of POD modes: 10 – for each state variable, each supremiser and the control, 15 – for both adjoint velocities.

Figures 3.24 and 3.25 represent the high–fidelity solutions for two different values of the parameters $(\bar{U}, \nu) = (5, 0.05)$, with $Re = 100$, and $(\bar{U}, \nu) = (1, 0.1)$, with $Re = 10$. The solutions were obtained by carrying out 25 optimisation iterations via the L–BFGS–B algorithm. Figures 3.24 and 3.25 show the intermediate solutions at iteration 0, 5 and 25 for the fluid velocities u_1 and u_2. The final solution is taken to be the 25–iteration optimisation solution as we can observe a continuity between subdomain solutions at the interface Γ_0. We present additional details in Tables 3.12 – 3.15. In particular, in Tables 3.12 and 3.14, we list the values for the functional $\mathcal{J}_\gamma$ and the $L^2(\Gamma_0)$–norm of the gradient $\frac{d\mathcal{J}_\gamma}{dg}$ at the different iteration of the optimisation procedure, while Table 3.13 and Table 3.15 contain the absolute and relative errors with respect to the monolithic(entire–domain) solutions u_h, p_h.

Figures 3.26 – 3.27 represent the reduced–order solutions for two different values of the parameters $(\bar{U}, \nu) = (5, 0.05)$ and $(\bar{U}, \nu) = (1, 0.1)$. For both cases, we choose the following number of the reduced basis functions: $N_{u_1} = N_{s_1} = N_{p_1} = N_{u_2} = N_{s_2} = N_{p_2} = N_g = 10$, whereas for the adjoint velocities, we choose $N_{\xi_1} = N_{\xi_2} = 15$. As it was mentioned before we use a higher number for the adjoint variables ξ_1 and ξ_2 since they show much slower decay of the singular values (see Figure 3.19a). Figure 3.26 shows the intermediate solutions at iteration 0, 3 and 15 for the fluid velocities u_1 and u_2 corresponding to the parameter value $(\bar{U}, \nu) = (5, 0.05)$, and Figure 3.27 shows the velocities u_1 and u_2 for the parameter value $(\bar{U}, \nu) = (1, 0.1)$. The final solution is taken to be the 10–iteration optimisation solution.

We present additional details in Tables 3.16 – 3.19. In particular, in Tables 3.16 and 3.18 we list the values for the functional $\mathcal{J}_\gamma$ and the $L^2(\Gamma_0)$–norm of the gradient $\frac{d\mathcal{J}_\gamma}{dg}$ at the different iteration of the optimisation procedure, while Table 3.17 and Table 3.19 contain the L^2–relative errors with respect to the monolithic (the entire–domain) solutions u_h, p_h.

Analysing the results, we are able to see that the reduced basis method gives us a solution as accurate as the high–fidelity model. The reduced–order approximation of the optimisation problem



(a) Iteration 0 (b) Iteration 3 (c) Iteration 10

Figure 3.27: Reduced–order solution for the velocities u_1 and u_2. Values of the parameters $\bar{U} = 1$, $\nu = 0.1$ and with $Re = 10$. Number of POD modes: 10 – for each state variable, each supremiser and the control, 15 – for both adjoint velocities.

Iteration	**Functional Value**	**Gradient norm**
0	$4.8 \cdot 10^{-1}$	3.153
3	$2.4 \cdot 10^{-2}$	1.634
10	$7.2 \cdot 10^{-5}$	0.023

Table 3.16: Functional values and the gradient norm for the ROM optimisation solution at parameter values $\bar{U} = 5$, $\nu = 0.05$ and with $Re = 100$

Iteration	**Abs. error u_N**		**Rel. error u_N**		**Abs. error p_N**		**Rel. error p_N**	
	Ω_1	Ω_2	Ω_1	Ω_2	Ω_1	Ω_2	Ω_1	Ω_2
0	0.3411	0.1796	1.0000	0.1523	0.2431	0.2519	1.0000	0.1864
3	0.0512	0.0552	0.1501	0.0468	0.0531	0.0646	0.3634	0.0478
10	0.0050	0.0056	0.0147	0.0047	0.0139	0.0139	0.0956	0.0103

Table 3.17: Absolute and relative errors of the ROM optimisation solution with respect to the monolithic solution at the parameter values $\bar{U} = 5$, $\nu = 0.05$ and with $Re = 100$

Iteration	**Functional Value**	**Gradient norm**
0	$2.6 \cdot 10^{-2}$	$2.6 \cdot 10^{-1}$
3	$1.5 \cdot 10^{-5}$	$1.0 \cdot 10^{-2}$
10	$7.1 \cdot 10^{-7}$	$1.2 \cdot 10^{-3}$

Table 3.18: Functional values and the gradient norm for the ROM optimisation solution at the parameter values $\bar{U} = 1$, $\nu = 0.1$ and with $Re = 10$

Iteration	**Abs. error u_N**		**Rel. error u_N**		**Abs. error p_N**		**Rel. error p_N**	
	Ω_1	Ω_2	Ω_1	Ω_2	Ω_1	Ω_2	Ω_1	Ω_2
0	0.0668	0.0591	1.0000	0.2424	0.0349	0.0403	1.0000	0.0936
3	0.0010	0.0019	0.0155	0.0076	0.0024	0.0020	0.0752	0.0047
10	0.0004	0.0004	0.0066	0.0017	0.0020	0.0019	0.0621	0.0045

Table 3.19: Absolute and relative errors of the ROM optimisation solution with respect to the monolithic solution at the parameter values $\bar{U} = 1$, $\nu = 0.1$ and with $Re = 10$

Parameter value		**Velocity relative error**		**Pressure relative error**	
$\bar{U}$	ν	Ω_1	Ω_2	Ω_1	Ω_2
1	0.1	0.020	0.003	0.014	0.0007
5	0.05	0.040	0.005	0.013	0.002

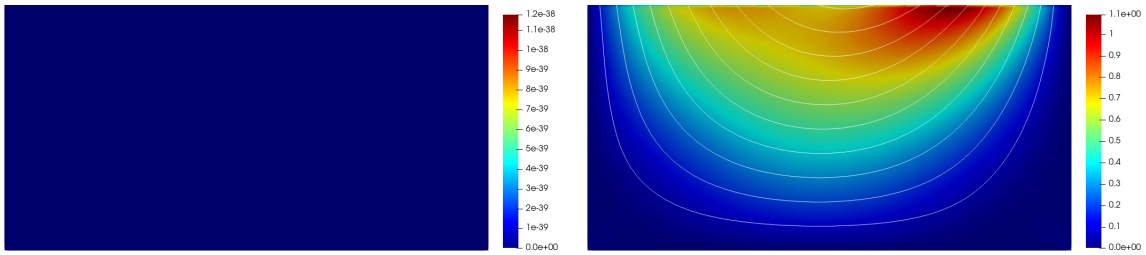
Table 3.20: Relative errors between FOM and ROM solutions (in terms of H^1–norm for the velocity fields and L^2–norm for the pressure fields)

at hand allowed us to reduce the dimension of the high–fidelity optimisation functional by more than 10–20 times and enabled us to use half optimisation algorithm iterations (each optimisation step requires at least one solve of the state and the adjoint equations).

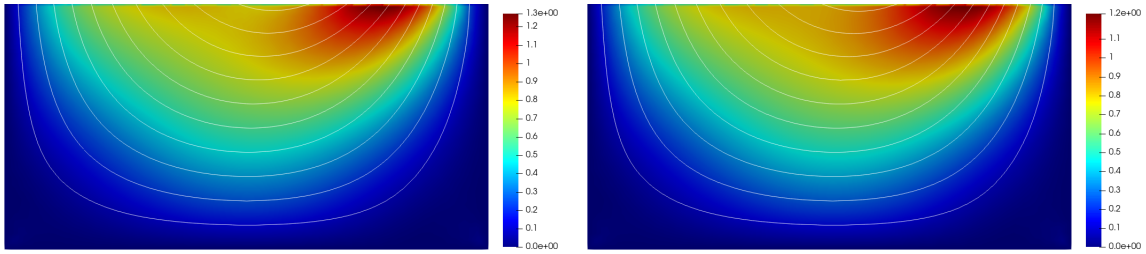
In order to provide more visually representable results (the scale of the solution on the subdomains Ω_1 and Ω_2 has a few orders of the difference in the magnitude), we provide the graphs of the velocities u_1 and u_2 separately in Figures 3.28 and 3.29. Additionally, in Table 3.20 we provide a comparison between full–order and reduced–order models in terms of the relative errors between ROM solutions with respect to the corresponding FOM solutions. The considerations drawn in the previous section are valid also for this test case.

Remark. In both numerical cases presented above, it might seem that due to the fact that the non–homogeneous Dirichlet boundary condition is present only on the boundary of one of the subdomains only a few corrections are needed on this subdomain. On the other hand, this is true only for the velocity field, as it can be seen in the tables listing the errors (for instance in Table 3.3). Indeed, the errors for the pressure on those subdomains are higher than on the other one. Regarding the cavity flow, our original idea was to split the domain vertically, but in that case, the convergence even at the full–order level was much slower, hence, we opted for the horizontal split.

Remark (High Reynolds simulations). Also for this test case, the range of Reynolds number for which the DD solver converges is stricter than the one where the monolithic solver provides a solution. The reason is that the optimisation algorithms are very sensitive to the initial guess, and the authors suspect that some further stabilisation techniques should be used.

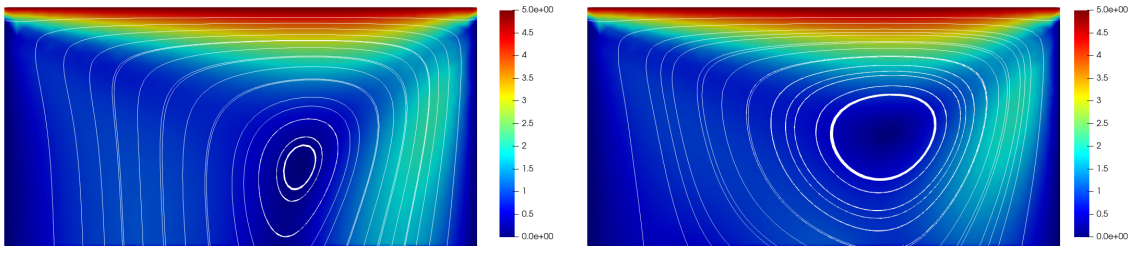


(a) Iteration 0 (b) Iteration 5

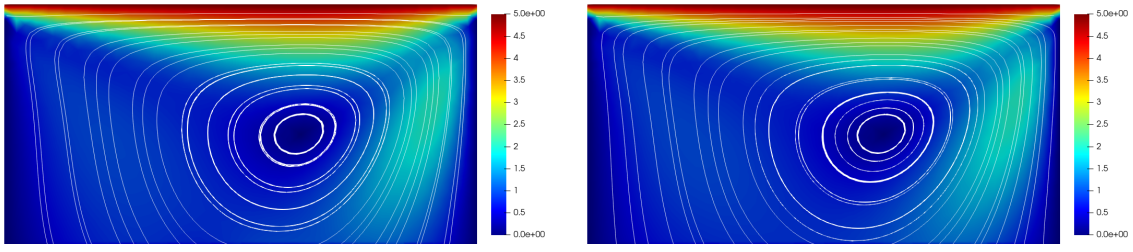


(c) Iteration 10 (d) Iteration 25

Figure 3.28: Reduced–order solution for the velocity u_1. Values of the parameters $\bar{U} = 5$, $\nu = 0.05$ and with $Re = 100$.



(a) Iteration 0 (b) Iteration 5



(c) Iteration 10 (d) Iteration 25

Figure 3.29: Reduced–order solution for the velocity u_2. Values of the parameters $\bar{U} = 5$, $\nu = 0.05$ and with $Re = 100$.

3.6 Conclusions and perspectives

In this chapter, we proposed a reduced–order model for the optimisation–based domain decomposition formulation of the parameter–dependent stationary incompressible 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 have managed to build a model for which the generation of the reduced basis spaces is carried out separately in each subdomain and for each component of the problem solution. Furthermore, as the numerical results show, the reduction of the optimal–control problem can be observed not only in the dimensions of the different components of the problem, i.e., of the functional, the state and the adjoint equations but also in the number of the iterations of the optimisation algorithm. The aforementioned techniques could be promising in the context of more complex time–dependent problems and, more importantly, multiphysics problems, where either pre–existing solvers can be used on each subcomponent or we do not have direct access to the codes. In particular, this methodology can be extended to problems with several sub–domains, to nonstationary fluid–dynamics problems, see Chapter 4, and to multiphysics problems, for instance, Fluid–Structure interaction problems in Chapter 5. Moreover, this approach can be applied also to more complicated problems, where different types of numerical models are used in different subdomains.

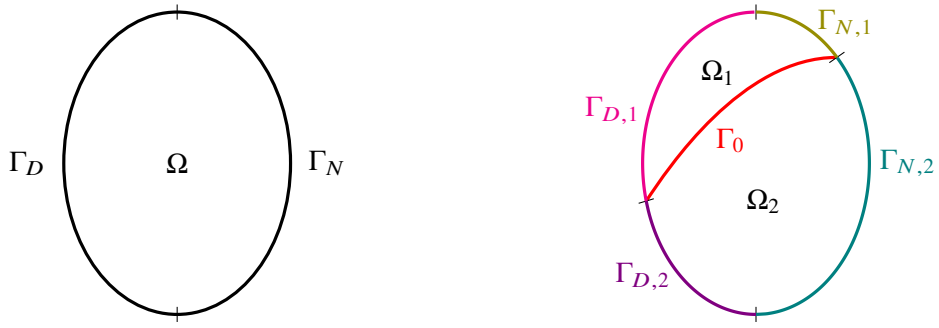
Chapter 4

Non–stationary Fluid Dynamics Problems

In this chapter, we introduce the monolithic formulation of non–stationary incompressible Navier–Stokes equations and its time–discretisation scheme with the further derivation of the optimisation–based domain–decomposition formulation at each time step. Then, we derive *a priori* estimates for the solutions of Navier–Stokes equations that are then used to prove the existence and uniqueness of the minimiser to the optimal–control problem at hand. Furthermore, we obtain the optimality condition for the resulting optimal control problem and the expression for the gradient of the objective functional with the following listing of the gradient–based optimisation algorithm. Later in the chapter, we present two ROM techniques: an intrusive Galerkin projection and a neural network (NN) algorithm, both based on a Proper Orthogonal Decomposition (POD) methodology. At the end of the section, we show some numerical results for two toy problems: the backward–facing step and the lid–driven cavity flows.

4.1 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 by employing the implicit Euler time–stepping scheme. We then describe a two–domain optimisation–based domain–decomposition formulation at each time step, and introduce its variational formulation 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 4.5.



(a) Physical domain (b) Domain Decomposition

Figure 4.1: Domain and boundaries.

4.1.1 Monolithic formulation

Let Ω be a physical domain of interest and we assume it 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.

$$\frac{\partial u}{\partial t} - \nu \Delta u + (u \cdot \nabla)\, u + \nabla p = f \quad \text{in} \quad \Omega \times (0,T], \tag{4.1a}$$

$$-\text{div} u = 0 \quad \text{in} \quad \Omega \times (0,T], \tag{4.1b}$$

$$u = u_D \quad \text{on} \quad \Gamma_D \times [0,T], \tag{4.1c}$$

$$\nu \frac{\partial u}{\partial \mathbf{n}} - p\mathbf{n} = 0 \quad \text{on} \quad \Gamma_N \times [0,T], \tag{4.1d}$$

$$u(t=0) = u_0 \quad \text{in} \quad \Omega, \tag{4.1e}$$

where Γ_D and Γ_N are disjoint subsets of Γ (as it is shown in Figure 4.1a) and $\mathbf{n}$ is an outward unit normal vector to Γ_N.

4.1.2 Time discretisation

We will start with the time discretisation of problem (4.1). Let $\Delta t > 0$, we assume the following time interval partition: $0 = t_0 < t_1 < .... < t_M = T$, where $t_n = n\Delta t$ for $n = 0, ..., M$. 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.

$$\frac{u^n - u^{n-1}}{\Delta t} - \nu \Delta u^n + (u^n \cdot \nabla)\, u^n + \nabla p^n = f^n \quad \text{in} \quad \Omega, \tag{4.2a}$$

$$-\text{div} u^n = 0 \quad \text{in} \quad \Omega, \tag{4.2b}$$

$$u^n = u_D^n \quad \text{on} \quad \Gamma_D, \tag{4.2c}$$

$$\nu \frac{\partial u^n}{\partial \mathbf{n}} - p^n \mathbf{n} = 0 \quad \text{on} \quad \Gamma_N, \tag{4.2d}$$

and for $n = 0$

$$u^0 = u_0 \quad \text{in} \quad \Omega. \tag{4.2e}$$

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

4.1.3 Domain Decomposition (DD) formulation

Let Ω_i, $i = 1, 2$, be open subsets of Ω, such that $\overline{\Omega} = \overline{\Omega_1 \cup \Omega_2}$, $\Omega_1 \cap \Omega_2 = \emptyset$. 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 4.1b.
The DD formulation reads as follows: for $n \geq 1$ and $i = 1, 2$, given $f_i : \Omega_i \times [0, T] \to \mathbb{R}^2$ and $u_{i,D} : \Gamma_{i,D} \times [0, T] \to \mathbb{R}^2$, find $u_i^n : \Omega_i \to \mathbb{R}^2$ and $p_i^n : \Omega_i \to \mathbb{R}$ s.t.

$$\frac{u_i^n - u_i^{n-1}}{\Delta t} - \nu \Delta u_i^n + \left(u_i^n \cdot \nabla\right) u_i^n + \nabla p_i^n = f_i^n \quad \text{in} \quad \Omega_i, \tag{4.3a}$$

$$-\text{div} u_i^n = 0 \quad \text{in} \quad \Omega_i, \tag{4.3b}$$

$$u_i^n = u_{i,D}^n \quad \text{on} \quad \Gamma_{i,D}, \tag{4.3c}$$

$$\nu \frac{\partial u_i^n}{\partial \mathbf{n}} - p_i^n \mathbf{n} = 0 \quad \text{on} \quad \Gamma_{i,N}, \tag{4.3d}$$

$$\nu \frac{\partial u_i^n}{\partial \mathbf{n}_i} - p_i^n \mathbf{n}_i = (-1)^{i+1} g \quad \text{on} \quad \Gamma_0, \tag{4.3e}$$

for some $g : \Gamma_0 \to \mathbb{R}^2$, where by $\mathbf{n}_i$ we denote an outward unit normal vector with respect to the domain Ω_i and

$$u_i^0 = u_0 \quad \text{in} \quad \Omega_i, \tag{4.3f}$$

for $n = 0$.

For a given g, the solution to problem (4.3) might not be the same as the solution to problem (4.2), that is $u_1^n \neq u^n|_{\Omega_1}$, $p_1^n \neq p^n|_{\Omega_1}$, $u_2^n \neq u^n|_{\Omega_2}$ and $p_2^n \neq p^n|_{\Omega_2}$. On the other hand, there exists a choice for g, $g = \left(\nu \frac{\partial u_1^n}{\partial \mathbf{n}_1} - p_1^n \mathbf{n}_1\right)\Big|_{\Gamma_0} = -\left(\nu \frac{\partial u_2^n}{\partial \mathbf{n}_2} - p_2^n \mathbf{n}_2\right)\Big|_{\Gamma_0}$, such that the solution to (4.3) coincides with the solution to (4.2) on the corresponding subdomains. Therefore, we must find such a g, so that u_1^n is as close as possible to u_2^n at the interface Γ_0. One way to accomplish this is to minimise the functional

$$\mathcal{J}(u_1^n, u_2^n) := \frac{1}{2} \int_{\Gamma_0} \left|u_1^n - u_2^n\right|^2 d\Gamma. \tag{4.4}$$

Instead of (4.4), we can also consider the penalised or regularised functional

$$\mathcal{J}_\gamma(u_1^n, u_2^n; g) := \frac{1}{2}\int_{\Gamma_0} \left|u_1^n - u_2^n\right|^2 d\Gamma + \frac{\gamma}{2}\int_{\Gamma_0} |g|^2 \, d\Gamma, \tag{4.5}$$

where γ is a constant that can be chosen to change the relative importance of the terms in (4.5). Thus, we face an optimisation problem under PDE constraints: minimise the functional (4.4) (or (4.5)) over a suitable function g, subject to (4.3).

4.1.4 Variational Formulation of the PDE constraints

For $i = 1, 2$ define the following spaces:

- $V_i := \left\{u \in H^1(\Omega_i; \mathbb{R}^2)\right\}$,
- $V_{i,0} := \left\{u \in H^1(\Omega_i; \mathbb{R}^2) : \ u|_{\Gamma_{i,D}} = 0\right\}$,
- $Q_i := \left\{p \in L^2(\Omega_i; \mathbb{R})\right\}$.

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 $i = 1, 2$.

Then, we define the following bilinear and trilinear forms: for i=1,2

- $m_i : V_i \times V_{i,0} \to \mathbb{R}, \quad m_i(u_i, v_i) := (u_i, v_i)_{\Omega_i}$,
- $a_i : V_i \times V_{i,0} \to \mathbb{R}, \quad a_i(u_i, v_i) := \nu(\nabla u_i, \nabla v_i)_{\Omega_i}$,
- $b_i : V_i \times Q_i \to \mathbb{R}, \quad b_i(v_i, q_i) := -(\mathrm{div} v_i, q_i)_{\Omega_i}$,
- $c_i : V_i \times V_i \times V_{i,0} \to \mathbb{R}, \quad c_i(u_i, w_i, v_i) := ((u_i \cdot \nabla) w_i, v_i)_{\Omega_i}$,

where $(\cdot, \cdot)_\omega$ indicates the $L^2(\omega)$ inner product.

Consequently, the variational counterpart of (4.3) reads as follows: for $n \geq 1$ and $i = 1, 2$, find $u_i \in V_i$ and $p_i \in Q_i$ s.t.

$$\begin{aligned} \frac{1}{\Delta t} m_i\left(u_i^n - u_i^{n-1}, v_i\right) + a_i(u_i^n, v_i) + c_i(u_i^n, u_i^n, v_i) + b_i(v_i, p_i^n) \\ = (f_i^n, v_i)_{\Omega_i} + \left((-1)^{i+1} g, v_i\right)_{\Gamma_0} \quad \forall v_i \in V_{i,0}, \end{aligned} \tag{4.6a}$$

$$b_i(u_i^n, q_i) = 0 \qquad \forall q_i \in Q_i, \tag{4.6b}$$

$$u_i^n = u_{i,D}^n \qquad \text{on } \Gamma_{i,D}. \tag{4.6c}$$

4.2 Analysis of the optimal control problem

In this section, we would like to 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. Due to the presence of the Neumann boundary condition, the analysis of the Navier–Stokes equations is not possible, so we will recast the problem in the framework where it is possible to prove the well–posedness and we give some indication about the original problem.

4.2.1 A modified Navier–Stokes problem

First, without loss of generality, we assume that the Dirichlet data $u_{i,D}$ is homogeneous. Otherwise, we can use various techniques, e.g., lifting functions as will be discussed later on, to obtain a problem with homogeneous Dirichlet boundary conditions on $\Gamma_{i,D}$. 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 (4.6). 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 (4.6b), we can see

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

which leads to the following expression

$$c_i(u_i^n, u_i^n, u_i^n) = \frac{1}{2} \int_{\Gamma_{i,N}\cup\Gamma_0} |u_i^n|^2 (u_i^n \cdot \mathbf{n}) d\Gamma. \tag{4.7}$$

As we can see, the problem here is due to the fact that this boundary term has an unknown sign, which complicates further analysis. 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. [31]). We rewrite the Neumann–type outlet conditions and the coupling stress conditions between subdomains in the following way:

$$\nu \frac{\partial u_i^n}{\partial \mathbf{n}} - p_i^n \mathbf{n} - \frac{1}{2}(u_i^n \cdot \mathbf{n})u_i^n = 0 \quad \text{on} \quad \Gamma_{i,N}, \tag{4.8}$$

$$\nu \frac{\partial u_i^n}{\partial \mathbf{n}_i} - p_i^n \mathbf{n}_i - \frac{1}{2}(u_i^n \cdot \mathbf{n}_i)u_i^n = (-1)^{i+1} g \quad \text{on} \quad \Gamma_0. \tag{4.9}$$

This, in turn, leads to a new variational formulation of the state equations (4.3): 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.

$$\begin{aligned} \frac{1}{\Delta t} m_i \left(u_i^n - u_i^{n-1}, v_i \right) + a_i(u_i^n, v_i) + \tilde{c}_i(u_i^n, u_i^n, v_i) + b_i(v_i, p_i^n) \\ = (f_i^n, v_i)_{\Omega_i} + \left((-1)^{i+1} g, v_i \right)_{\Gamma_0} \qquad \forall v_i \in V_{i,0}, \end{aligned} \tag{4.10a}$$

$$b_i(u_i^n, q_i) = 0 \qquad \forall q_i \in Q_i, \tag{4.10b}$$

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

$$\tilde{c}_i(u_i, w_i, v_i) = \frac{1}{2} \left((u_i \cdot \nabla) w_i, v_i \right)_{\Omega_i} - \frac{1}{2} \left((u_i \cdot \nabla) v_i, w_i \right)_{\Omega_i} \tag{4.11}$$

and it has the following remarkable property

$$\tilde{c}_i(u_i, v_i, v_i) = 0 \quad \forall u_i, v_i \in V_{i,0}. \tag{4.12}$$

4.2.2 *A priori* estimates

We now first introduce the various well–known properties of the different terms entering the weak formulation (4.10), i.e.,

- 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

$$|m_i(u_i, v_i)| \leq C_m ||u_i||_{V_{i,0}} ||v_i||_{V_{i,0}} \qquad \forall u_i, v_i \in V_{i,0}, \tag{4.13}$$

$$|a_i(u_i, v_i)| \leq C_a ||u_i||_{V_{i,0}} ||v_i||_{V_{i,0}} \qquad \forall u_i, v_i \in V_{i,0}, \tag{4.14}$$

$$|\tilde{c}_i(u_i, w_i, v_i)| \leq C_c ||u_i||_{V_{i,0}} ||w_i||_{V_{i,0}} ||v_i||_{V_{i,0}} \qquad \forall u_i, w_i, v_i \in V_{i,0}, \tag{4.15}$$

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

$$a_i(v_i, v_i) \geq \alpha ||v_i||_{V_{i,0}}^2 \quad \forall v_i \in V_{i,0}, \tag{4.16}$$

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

$$\sup_{v_i \in V_{i,0} \setminus \{0\}} \frac{b_i(v_i, q_i)}{||v_i||_{V_{i,0}}} \geq \beta ||q_i||_{Q_i} \quad \forall q_i \in Q_i, \tag{4.17}$$

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

$$m_i(v_i, v_i) = ||v_i||_{L^2(\Omega_i)}^2 \geq 0 \quad \forall v_i \in V_{i,0}. \tag{4.18}$$

By using the properties (4.13), (4.16), (4.12), (4.18), the trace theorem and equations (4.10), we are able to write the following estimate for the solution u_i^n and p_i^n to (4.10)

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

which leads to the following estimate:

$$||u_i^n||_{V_{i,0}} \leq \frac{1}{\alpha}\left(\frac{C_m}{\Delta t}||u_i^{n-1}||_{V_{i,0}} + ||f_i^n||_{L^2(\Omega_i)} + ||g||_{L^2(\Gamma_0)}\right). \tag{4.19}$$

Similarly, by using (4.17), (4.13), (4.14), (4.15) and equations (4.10), we obtain

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

which together with the estimate (4.19) leads to

$$\begin{aligned}
||p_i^n||_{Q_i} &\leq \frac{1}{\beta}\left[\left(1+\frac{1}{\alpha}\left(\frac{C_m}{\Delta t}+C_a\right)\right)\left(\frac{C_m}{\Delta t}||u_i^{n-1}||_{V_{i,0}} + ||f_i^n||_{L^2(\Omega_i)}\right.\right. \\
&+ \left.\left.||g||_{L^2(\Gamma_0)}\right) + \frac{C_c}{\alpha^2}\left(\frac{C_m}{\Delta t}||u_i^{n-1}||_{V_{i,0}} + ||f_i^n||_{L^2(\Omega_i)} + ||g||_{L^2(\Gamma_0)}\right)^2\right].
\end{aligned} \tag{4.20}$$

4.2.3 Existence of optimal solutions

In this subsection, we prove the existence of optimal solutions for the regularised functional (4.5). The proof follows the methodology presented by Gunzburger et al. [75]. Firstly, we define the admissibility set as follows:

$$\begin{aligned}
\mathcal{U}_{ad} = \big\{&(u_1^n,p_1^n,u_2^n,p_2^n,g) \in V_{1,0}\times Q_1\times V_{2,0}\times Q_2\times L^2(\Gamma_0) \text{ such that} \\
&\text{equations (4.10) are satisfied and } \mathcal{J}_\gamma(u_1^n,u_2^n;g) < \infty\big\}.
\end{aligned}$$

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

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 $\left\{g^{(j)}\right\}$ is uniformly bounded in $L^2(\Gamma_0)$, which, in turn, due to the *a priori* estimates (4.19) and (4.20), 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 $\left(\hat{u}_1^n, \hat{p}_1^n, \hat{u}_2^n, \hat{p}_2^n, \hat{g}\right) \in \mathcal{U}_{ad}$ 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 $i = 1, 2$

$$u_i^{n,(j_k)} \rightharpoonup \hat{u}_i^n \quad \text{in} \quad V_{i,0}, \tag{4.21}$$
$$p_i^{n,(j_k)} \rightharpoonup \hat{p}_i^n \quad \text{in} \quad Q_i, \tag{4.22}$$
$$g^{(j_k)} \rightharpoonup \hat{g} \quad \text{in} \quad L^2(\Gamma_0), \tag{4.23}$$
$$u_i^{n,(j_k)} \to \hat{u}_i^n \quad \text{in} \quad L^2(\Omega_i), \tag{4.24}$$
$$u_i^{n,(j_k)}|_{\Gamma_0} \to \hat{u}_i^n|_{\Gamma_0} \quad \text{in} \quad L^2(\Gamma_0). \tag{4.25}$$

The last two results are obtained by the trace theorem and compact embedding results in Sobolev spaces, see for example [55, 112].

Since the forms $m_i(\cdot,\cdot)$, $a_i(\cdot,\cdot)$ and $b_i(\cdot,\cdot)$ are bilinear and continuous by (4.21), (4.22) and (4.23), we obtain the following convergence results:

$$\begin{aligned}
m_i(u_i^{n,(j_k)}, v_i) &\to m_i(\hat{u}_i^n, v_i) \quad \forall v_i \in V_{i,0},\\
a_i(u_i^{n,(j_k)}, v_i) &\to a_i(\hat{u}_i^n, v_i) \quad \forall v_i \in V_{i,0},\\
b_i(u_i^{n,(j_k)}, q_i) &\to b_i(\hat{u}_i^n, q_i) \quad \forall q_i \in Q_i,\\
b_i(v_i, p_i^{n,(j_k)}) &\to b_i(v_i, \hat{p}_i^n) \quad \forall v_i \in V_{i,0},\\
(g^{(j_k)}, v_i)_{\Gamma_0} &\to (\hat{g}, v_i)_{\Gamma_0} \quad \forall v_i \in V_{i,0}.
\end{aligned}$$

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 (4.24)– (4.25). 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 (4.10). 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.

4.2.4 Convergence with vanishing penalty parameter

In the previous section, we have proved the existence of optimal solutions of the regularised functional $\mathcal{J}_\gamma$ 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 (4.2), 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 (4.10). 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:

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

The last inequality tells as 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 $(u_1^{n,*},p_1^{n,*},u_2^{n,*},p_2^{n,*},g^*)\in V_{1,0}\times Q_1\times V_{2,0}\times Q_2\times L^2(\Gamma_0)$ in the sense of (4.21) – (4.25). 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 $u^{n,*}\in H^1_{0,\Gamma_D}(\Omega)$, $p^{n,*}\in L^2(\Omega)$, 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 (4.2) 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 [153], 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, [79]). 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). As we pointed out the condition (4.12) was essential in order to conduct the analysis of the optimal–control problem. Concerning the problem posed in the weak from (4.6), the numerical experiments show the same convergence results as in the case where the trilinear form (4.11) is adopted.

4.3 Optimality system and optimisation algorithms

In this section, we will provide the tools to tackle the optimal–control problem that arises in Section 4.1.3. First, we will derive the optimality system by means of the Lagrangian functional. Then, we will use the sensitivity derivatives technique to obtain the representation for the gradient of the objective functional, which will allow us to define an optimisation–based minimisation algorithm for the optimal–control problem at hand.

4.3.1 Optimality system

One of the ways to address the constrained optimisation problem is to reformulate the initial problem in terms of a Lagrangian functional by introducing the so–called adjoint variables. In this way, the

optimal solution to the original problem is sought among the stationary points of the Lagrangian, see, for instance, [76, 83].

We define the Lagrangian functional as follows:

$$\begin{aligned}&\mathcal{L}(u_1^n, p_1^n, u_2^n, p_2^n, \xi_1, \xi_2, \lambda_1, \lambda_2; g) := \mathcal{J}_\gamma(u_1^n, u_2^n; g)\\ &-\sum_{i=1}^{2}\left[\frac{m_i(u_i^n - u_i^{n-1}, \xi_i)}{\Delta t} + a_i(u_i^n, \xi_i) + c_i(u_i^n, u_i^n, \xi_i) + b_i(\xi_i, p_i^n) + b_i(u_i^n, \lambda_i)\right]\\ &+\sum_{i=1}^{2}(f_i^n, \xi_i)_{\Omega_i} + \sum_{i=1}^{2}((-1)^{i+1}g, \xi_i)_{\Gamma_0}.\end{aligned} \tag{4.26}$$

The Lagrangian functional above takes into account our objective functional (4.5) and the PDE constraints (4.6) multiplied by additional so–called adjoint variables $\xi_1, \xi_2, \lambda_1, \lambda_2 \in V_{1,0} \times Q_1 \times V_{2,0} \times Q_2$. It is proved (see, e.g. [83]) that the optimal solution to the constrained optimisation problem (4.5)– (4.6) coincides with stationary points of the higher–dimensional functional (4.26) that, in turn, gives us an easy way to obtain the optimality conditions.

Notice that, technically, we should also have included Lagrange multipliers corresponding to the non–homogeneous Dirichlet boundary conditions (4.6c) in the definition of the functional $\mathcal{L}$. However, since the functional $\mathcal{J}_\gamma$ (4.5) does not explicitly depend on $u_{1,D}^n$ and $u_{2,D}^n$, the corresponding adjoint Dirichlet boundary conditions will be homogeneous on these parts of the boundaries.

We now apply the necessary conditions for finding stationary points of $\mathcal{L}$. Setting to zero the first variations w.r.t. $\xi_i \in V_{i,0}$ and $\lambda_i \in Q_i$, for $i = 1, 2$, yields the state equations (4.6a)– (4.6b). Setting to zero the first variations w.r.t. u_1^n, p_1^n, u_2^n and p_2^n yields the adjoint equations:

$$\begin{aligned}\frac{1}{\Delta t}m_i(\eta_i, \xi_i) + a_i(\eta_i, \xi_i) + c_i\left(\eta_i, u_i^n, \xi_i\right) + c_i\left(u_i^n, \eta_i, \xi_i\right) & \\ + b_i(\eta_i, \lambda_i) = ((-1)^{i+1}\eta_i, u_1^n - u_2^n)_{\Gamma_0} & \quad \forall \eta_i \in V_{i,0}, \qquad (4.27a)\\ b_i(\xi_i, \mu_i) = 0 & \quad \forall \mu_i \in Q_i. \qquad (4.27b)\end{aligned}$$

Finally, setting to zero the first variations w.r.t. $g \in L^2(\Gamma_0)$ yields the optimality condition:

$$\gamma(h, g)_{\Gamma_0} + (h, \xi_1 - \xi_2)_{\Gamma_0} = 0 \quad \forall h \in L^2(\Gamma_0). \tag{4.28}$$

4.3.2 Sensitivity derivatives

In order to obtain the expression for the gradient of the optimisation problem at hand, we will resort to the sensitivity approach; see, for instance, [76, 83]. The approach consists of finding equations for directional derivatives of the state variables with respect to the control, called sensitivities. The first derivative $\frac{d\mathcal{J}_\gamma}{dg}$ of $\mathcal{J}_\gamma$ is defined through its action on the variation $\tilde{g}$ as follows:

$$\left\langle \frac{d\mathcal{J}_\gamma}{dg}, \tilde{g} \right\rangle = (u_1^n - u_2^n, \tilde{u}_1 - \tilde{u}_2)_{\Gamma_0} + \gamma(g, \tilde{g})_{\Gamma_0}, \tag{4.29}$$

where $\tilde{u}_1 \in V_{1,0}$, $\tilde{u}_2 \in V_{2,0}$ are the solutions to:

$$\frac{1}{\Delta t} m_i(\tilde{u}_i, v_i) + a_i(\tilde{u}_i, v_i) + c_i(\tilde{u}_i, u_i^n, v_i) + c_i(u_i^n, \tilde{u}_i, v_i) + b_i(v_i, \tilde{p}_i) = ((-1)^{i+1}\tilde{g}, v_i)_{\Gamma_0} \qquad \forall v_i \in V_{i,0}, \tag{4.30a}$$

$$b_i(\tilde{u}_i, q_i) = 0 \qquad \forall q_i \in Q_i. \tag{4.30b}$$

We can make use of the adjoint equations (4.27) in order to find the representation of the gradient of the functional $\mathcal{J}_\gamma$. Let ξ_1 and ξ_2 be the solutions to (4.27) and $\tilde{u}_1$ and $\tilde{u}_2$ be the solutions to (4.30). By setting $\eta_i = \tilde{u}_i$ in (4.27a), $\mu_i = \tilde{p}_i$ in (4.27b), $v_i = \xi_i$ in (4.30a) and $q_i = \lambda_i$ in (4.30b) we obtain:

$$(u_1^n - u_2^n, \tilde{u}_1 - \tilde{u}_2)_{\Gamma_0} = (\tilde{g}, \xi_1 - \xi_2)_{\Gamma_0},$$

so that it yields the explicit formula for the gradient of $\mathcal{J}_\gamma$:

$$\frac{d\mathcal{J}_\gamma}{dg}(u_1^n, u_2^n; g) = \gamma g + (\xi_1 - \xi_2)|_{\Gamma_0}, \tag{4.31}$$

where ξ_1 and ξ_2 are determined from g through (4.27). Notice that the gradient expression (4.31) is consistent with the optimality condition (4.28) derived in the previous section.

4.3.3 Gradient–based algorithm for the optimisation problem

In view of being able to provide a closed–form formula for the gradient for the objective functional $\mathcal{J}_\gamma$, the natural way to proceed is to resort to a gradient–based iterative optimisation algorithm.
In order to keep the exposition simple, we will describe the idea using a simple gradient method, while, in practice, we will use more sophisticated gradient–based methods. For every time step t^n, given an initial guess $g^{(0)}$, which we set from the previous time step, we update successive values of $g^{(j)}$ with

$$g^{(j+1)} = g^{(j)} - \alpha \frac{d\mathcal{J}_\gamma}{dg}\left(u_1^{n,(j)}, u_2^{n,(j)}; g^{(j)}\right). \tag{4.32}$$

Combining this with (4.31) we obtain:

$$g^{(j+1)} = g^{(j)} - \alpha\left(\gamma g^{(j)} + (\xi_1^{(j)} - \xi_2^{(j)})|_{\Gamma_0}\right), \tag{4.33}$$

where $\xi_1^{(j)}$ and $\xi_2^{(j)}$ are determined from (4.27) with g replaced by $g^{(j)}$.
To summarise, we have Algorithm 4.1 to find g^n at every time step t^n. Some of the commonly used convergence criteria of the algorithm are: the value of the functional or of the gradient norm is less than a certain given tolerance and the maximum number of optimisation iterations. Most commonly, a couple of them are used together. In practice, the methods we will use to solve such problems are Broyden–Fletcher–Goldfarb–Shanno (BFGS) and Newton Conjugate Gradient (CG) algorithms, which show faster convergence and higher efficiency with respect to the steepest–descent algorithm.

Algorithm 4.1 Gradient method with a fixed step

Input: $g^{(0)} := g^{n-1}$ and a step size $\alpha > 0$
$n := 0$
while Convergence criteria are not met **do**
  Solve (4.6) for $u_1^{n,(j)} \in V_1$, $u_2^{n,(j)} \in V_2$ with $g = g^{(j)}$
  Solve (4.27) for $\xi_1^{(j)} \in V_{1,0}$, $\xi_2^{(j)} \in V_{2,0}$ with $u_1^n = u_1^{n,(j)}$, $u_2^n = u_2^{n,(j)}$
  Set $g^{(j+1)} := (1 - \alpha\gamma)\, g^{(j)} - \alpha \left(\xi_1^{(j)} - \xi_2^{(j)}\right)\Big|_{\Gamma_0}$
  $n := n + 1$
end while
Set $u_1^n := u_1^{n(j)}$, $p_1^n := p_1^{n,(j)}$, $u_2^n := u_2^{n,(j)}$, $p_2^n := p_2^{n,(j)}$ and $g^n := g^{(j)}$
return u_1^n, p_1^n, u_2^n, p_2^n, g^n

4.4 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 an extra lower–dimensional triangulation $\mathcal{T}_0$ of the interface Γ_0. In theory, there is no requirement for the meshes $\mathcal{T}_1$ and $\mathcal{T}_2$ to be conforming on the interface Γ_0, but in the numerical examples listed later in the paper, this limitation was imposed by the software used by the authors. 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, in order 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.

$$\inf_{q_{i,h} \in Q_{i,h} \setminus \{0\}} \sup_{v_{i,h} \in V_{i,h} \setminus \{0\}} \frac{b_i(v_{i,h}, q_{i,h})}{||v_{i,h}||_{V_{i,h}} ||q_{i,h}||_{Q_{i,h}}} \geq c_i, \quad i = 1, 2, \tag{4.34}$$

$$\inf_{q_{i,h} \in Q_{i,h} \setminus \{0\}} \sup_{v_{i,h} \in V_{i,0,h} \setminus \{0\}} \frac{b_i(v_{i,h}, q_{i,h})}{||v_{i,h}||_{V_{i,0,h}} ||q_{i,h}||_{Q_{i,h}}} \geq c_{i+2}, \quad i = 1, 2. \tag{4.35}$$

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.

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

$$\mathcal{J}_{\gamma,h}(u_{1,h}^n, u_{2,h}^n; g_h) := \frac{1}{2} \int_{\Gamma_0} \left| u_{1,h}^n - u_{2,h}^n \right|^2 d\Gamma + \frac{\gamma}{2} \int_{\Gamma_0} |g_h|^2 \, d\Gamma \tag{4.36}$$

under the constraints that $u^n_{i,h} \in V_{i,h}$, $p^n_{i,h} \in Q_{i,h}$ satisfy the following variational equations for $i = 1, 2$

$$\begin{aligned} &\frac{m_i(u^n_{i,h} - u^{n-1}_{i,h}, v_{i,h})}{\Delta t} + a_i(u^n_{i,h}, v_{i,h}) + c_i(u^n_{i,h}, u^n_{i,h}, v_i) && \\ &\quad + b_i(v_{i,h}, p^n_{i,h}) = (f^n_i, v_{i,h})_{\Omega_i} + \left((-1)^{i+1} g_h, v_{i,h}\right)_{\Gamma_0} && \forall v_i \in V_{i,0,h}, \qquad (4.37a)\\ &\qquad b_i(u^n_{i,h}, q_{i,h}) = 0 && \forall q_{i,h} \in Q_{i,h}, \qquad (4.37b)\\ &\qquad u^n_i = u^n_{i,D,h} && \text{on } \Gamma_{i,D}, \qquad (4.37c) \end{aligned}$$

where $u^n_{i,D,h}$ 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 (4.37) and of the functional (4.36) 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 (4.36):

$$\frac{d\mathcal{J}_{\gamma,h}}{dg_h}(u^n_{1,h}, u^n_{2,h}; g_h) = \gamma g_h + (\xi_{1,h} - \xi_{2,h})|_{\Gamma_0}, \qquad (4.38)$$

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

$$\begin{aligned} &\frac{1}{\Delta t} m_i(\eta_{i,h}, \xi_{i,h}) + a_i(\eta_{i,h}, \xi_{i,h}) + c_i\left(\eta_{i,h}, u^n_{i,h}, \xi_{i,h}\right) + c_i\left(u^n_{i,h}, \eta_{i,h}, \xi_{i,h}\right) && (4.39a)\\ &\qquad + b_i(\eta_{i,h}, \lambda_{i,h}) = ((-1)^{i+1}\eta_{i,h}, u^n_{1,h} - u^n_{2,h})_{\Gamma_0} \quad \forall \eta_{i,h} \in V_{i,0,h},\\ &\qquad b_i(\xi_{i,h}, \mu_{i,h}) = 0 \qquad \forall \mu_{i,h} \in Q_{i,h}. && (4.39b) \end{aligned}$$

In (4.38), 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.

4.5 Reduced–Order Model

As highlighted in Section 2, 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 start with describing the reduced basis generation based on the Proper Orthogonal Decomposition (POD) technique, which is followed by two online phases based on a Galerkin projection onto the reduced spaces and on a multilayer perceptron neural network.

4.5.1 Reduced Basis Generation

Firstly, we will mention again the need for a supremiser and a lifting function as it is described in Section 3.4.1 to render the further stability of the reduced–order problem. Once we obtain the homogenised snapshots $u^n_{i,0,h}$ defined as $u^n_{i,0,h} := u^n_{i,h} - l^n_{i,h}$ (see Section 3.4.1) and the pressure supremisers $s^n_{i,h}$ 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, [81]. 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 (4.36), (4.37) and (4.39): $V_{i,0,h} = \text{span}\{\phi^{u_i}_j\}_{j=1}^{\mathcal{N}^{u_i}_h}$, $Q_{i,h} = \text{span}\{\phi^{p_i}_j\}_{j=1}^{\mathcal{N}^{p_i}_h}$ and $X_h = \text{span}\{\phi^g_j\}_{j=1}^{\mathcal{N}^g_h}$, 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 $\mathcal{S}_{u_i} \in \mathbb{R}^{\mathcal{N}^{u_i}_h \times MK}$, $\mathcal{S}_{s_i} \in \mathbb{R}^{\mathcal{N}^{u_i}_h \times MK}$, $\mathcal{S}_{p_i} \in \mathbb{R}^{\mathcal{N}^{p_i}_h \times MK}$, for $i = 1, 2$, and $\mathcal{S}_g \in \mathbb{R}^{\mathcal{N}^g_h \times MK}$ 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:

$$
\begin{aligned}
(X_{s_i})_{jk} = (X_{u_i})_{jk} &= \left(\nabla\phi^{u_i}_k, \nabla\phi^{u_i}_j\right)_{\Omega_i}, && \text{for } j,k = 1, ..., \mathcal{N}^{u_i}_h,\ i = 1, 2,\\
(X_{p_i})_{jk} &= \left(\phi^{p_i}_k, \phi^{p_i}_j\right)_{\Omega_i}, && \text{for } j,k = 1, ..., \mathcal{N}^{p_i}_h,\ i = 1, 2,\\
(X_g)_{jk} &= \left(\phi^g_k, \phi^g_j\right)_{\Gamma_0}, && \text{for } j,k = 1, ..., \mathcal{N}^g_h.
\end{aligned}
$$

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:

$$C_* := \mathcal{S}^T_* X_* S_*$$

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:

$$C_* \mathcal{Q}_* = \mathcal{Q}_* \Lambda_* \tag{4.40}$$

where $* \in \{u_1, s_1, p_1, u_2, s_2, p_2, g\}$, $\mathcal{Q}_*$ 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 $\mathcal{Q}_*$:

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

where λ_k^* is the k–th eigenvalue from (4.40). Therefore, we are able to form the set of reduced bases 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:

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

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

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

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:

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

for $* \in \{u_1, p_1, u_2, p_2, g\}$. Now, due to the supremiser enrichment the spaces $V_N^{u_i}$ and $V_N^{p_i}$ are inf–sup stable in the sense (4.35) for $i = 1, 2$; the proof can be found in [17].

4.5.2 Online Phase

Once we have introduced the reduced basis spaces, we can define the reduced function expansions

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

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 $g_N \in V_N^g$ the functional

$$\mathcal{J}_{\gamma,N}(u_{1,N}^n, u_{2,N}^n; g_N) := \frac{1}{2} \int_{\Gamma_0} \left| u_{1,N}^n - u_{2,N}^n \right|^2 d\Gamma + \frac{\gamma}{2} \int_{\Gamma_0} |g_N|^2 \, d\Gamma, \tag{4.41}$$

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

$$\begin{aligned}
\frac{1}{\Delta t} m_i(u^n_{i,0,N}, v_{i,N}) \;&+\; a_i(u^n_{i,0,N}, v_{i,N}) + c_i(u^n_{i,0,N}, u^n_{i,0,N}, v_{i,N}) \\
&+\; c_i(u^n_{i,0,N}, l^n_{i,N}, v_{i,N}) + c_i(l^n_{i,N}, u^n_{i,0,N}, v_{i,N}) \qquad (4.42a)\\
&+\; b_i(v_{i,N}, p^n_{i,N}) = (f^n_i, v_{i,N})_{\Omega_i} + ((-1)^{i+1} g_N, v_{i,N})_{\Gamma_0} \\
&+\; \frac{1}{\Delta t} m_i(u^{n-1}_{i,0,N}, v_{i,N}) - \frac{1}{\Delta t} m_i(l^n_{i,N}, v_{i,N}) \\
&-\; a_i(l^n_{i,N}, v^n_{i,N}) - c_i(l_{i,N}, l_{i,N}, v_{i,N}) \quad \forall v_{i,N} \in V^{u_i}_N, \\
b_i(u^n_{i,0,N}, q_{i,N}) \;&=\; -b_i(l^n_{i,N}, q_{i,N}) \quad \forall q_{i,N} \in V^{p_i}_N, \qquad (4.42b)
\end{aligned}$$

where $l^n_{i,N}$ is the Galerkin projection of the lifting function $l^n_{i,h}$ to the finite dimensional vector space $V^{u_i}_N$ and $i = 1, 2$.

Similarly to the offline phase, we notice that the structure of the equations (4.42) and the functional (4.41) 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 (4.41)

$$\frac{d\mathcal{J}_{\gamma,N}}{dg_N}(u^n_{1,N}, u^n_{2,N}; g_N) = \gamma g_N + (\xi_{1,N} - \xi_{2,N})|_{\Gamma_0}, \qquad (4.43)$$

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

$$\begin{aligned}
&\frac{1}{\Delta t} m_i(\eta_{i,N}, \xi_{i,N}) + a_i(\eta_{i,N}, \xi_{i,N}) + c_i\left(\eta_{i,N}, u^n_{i,N}, \xi_{i,N}\right) + c_i\left(u^n_{i,N}, \eta_{i,N}, \xi_{i,N}\right) \qquad (4.44a)\\
&\qquad + b_i(\eta_{i,N}, \lambda_{i,h}) = ((-1)^{i+1}\eta_{i,N}, u^n_{1,N} - u^n_{2,N})_{\Gamma_0} \quad \forall \eta_{i,N} \in V^{u_i}_{i,N}, \\
&\qquad b_i(\xi_{i,N}, \mu_{i,N}) = 0 \qquad\qquad \forall \mu_{i,N} \in V^{p_i}_{i,N}. \qquad (4.44b)
\end{aligned}$$

Above, the restriction $\cdot|_{\Gamma_0}$ is meant as an $L^2(\Gamma_0)$–projection onto space V^g_N. We would also like to stress that from the numerical implementation point of view 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 $p = N_g$.

4.5.3 POD–NN

In this section, we would like to give a quick overview of the POD–NN method [82]. After the construction of the POD reduced spaces as described in Section 4.5.1, 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 form a training set by the projection of each snapshot for variables u_1, p_1, u_2, p_2 onto the corresponding reduced space – $U_{*,\text{output}}, * \in \{u_1, p_1, u_2, p_2\}$. The input set is composed of the tuples U_{input} that contain K sets of physical parameters (the same ones that have

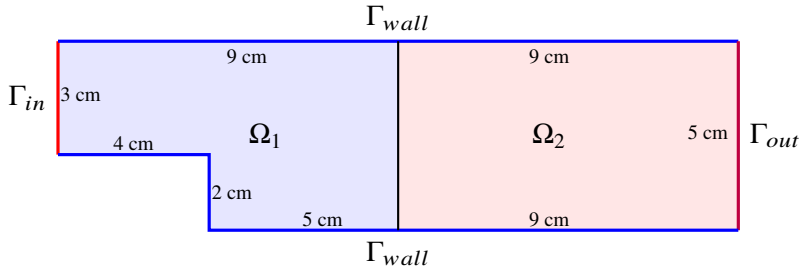


Figure 4.2: Physical domain and domain decomposition for the backward–facing step problem

been sampled for snapshot construction) and M time–steps $t_i, i = 1, .., M$; this results in KM tuples of dimension $(L + 1)$, where L is the number of considered physical parameters.
Having built the input and output training sets, we build an artificial neural network (ANN) for each component $* \in \{u_1, p_1, u_2, p_2\}$ which approximates $U_{\text{input}} \to U_{*,\text{output}}$. 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, just the prediction by ANN at the required parameter value and time step.

The ANN used in this algorithm is a simple dense multilayer perceptron that consists of a repeated composition of affine operations and nonlinear activation functions [68]. 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 [97], 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 a 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.

4.6 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.

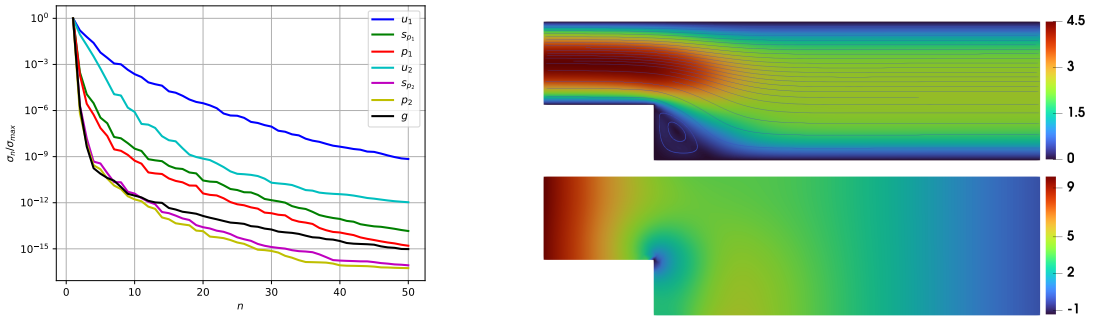
All the numerical simulations for the offline phase were obtained using the software FEniCS [1], whereas the online phase simulations were carried out using RBniCS [3] and EZyRB [4].

4.6.1 Backward–facing step test case

We start with introducing the backward–facing step flow test case. Figure 4.2 represents the physical domain of interest, the dimensional lengths and the boundary conditions. The splitting into two

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
$\text{it}_{\max}$	1000	Parameters training set size K	62
tol_{opt}	10^{-9}	Maximum retained modes $N_{\max}$	100

Table 4.1: 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 solutions at the final time step

Figure 4.3: 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).

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 4.2.

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 in Table 4.1. 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 4.1. 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–

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 4.2: Backward–facing step: computational details of the online stage

Goldfarb—Shanno (L–BFGS–B) optimisation algorithm [32], where the following stopping criteria were applied: either the maximal number of iteration $\text{it}_{\max}$ is reached or the gradient norm of the target functional is less than the given tolerance tol_{opt} or the relative reduction of the functional value is less than the tolerance that is automatically chosen by the `scipy` library [183].

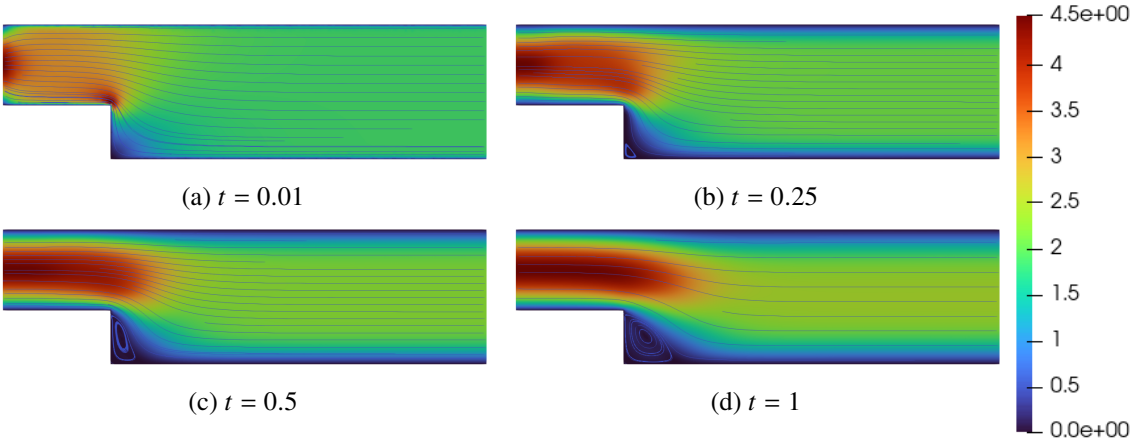
Snapshots are sampled from a training set of K parameters randomly sampled from the two–dimensional parameter space for each time–step t_i, $i = 1, ..., M$ and the first $N_{\max}$ POD modes have been retained for each component. Figure 4.3a shows the POD singular values for all the state and the control variables; we can see an evident exponential decay of the singular values. Figure 4.3b shows an example of a monolithic (whole–domain) solution with which we would conduct a comparison and the numerical analysis of the DD–FOM and the ROM.

In Table 4.2, 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 4.4– 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 4.6 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 mostly concentrated at the interface, while ROM contains some extra noise due to the POD reduction.

Figure 4.7 shows the number of iterations for both FOM and ROM optimisation processes. 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 4.2), and with good use of hyper–reduction techniques (see, for instance, [81]), it will allow to obtain very efficient solvers in terms of computational time.

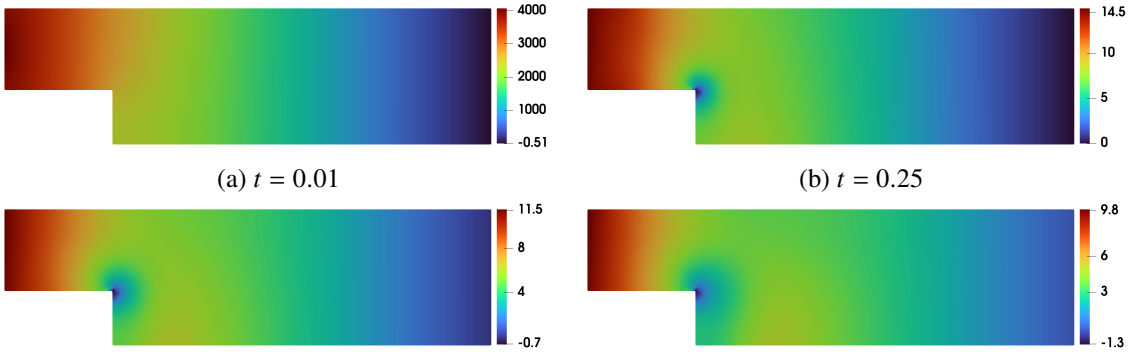
We would like also to provide a comparison of the full–order and the reduced–order models with non–intrusive POD–NN model. Due to the discontinuity with the initial condition, the first



(a) $t = 0.01$ (b) $t = 0.25$

(c) $t = 0.5$ (d) $t = 1$

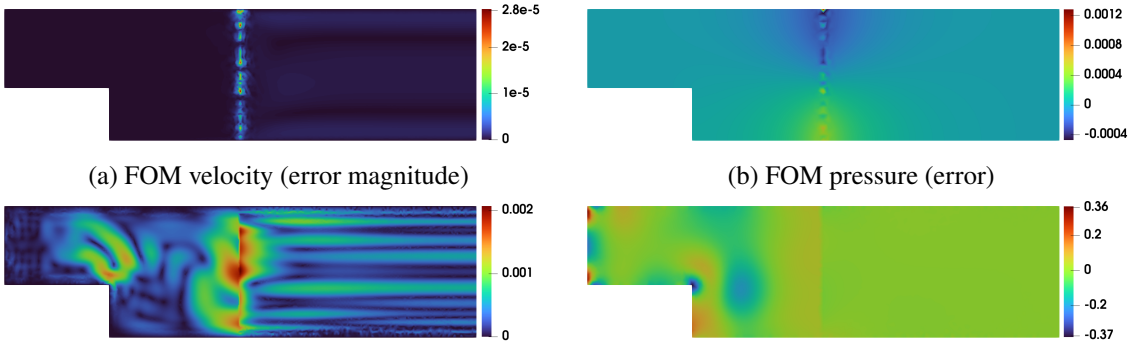
Figure 4.4: Backward–facing step: high–fidelity solution for the velocities u_1 and u_2 at 4 different time instances.



(a) $t = 0.01$ (b) $t = 0.25$

(c) $t = 0.5$ (d) $t = 1$

Figure 4.5: Backward–facing step: high–fidelity solution for the pressures p_1 and p_2 at 4 different time instances.



(a) FOM velocity (error magnitude)

(b) FOM pressure (error)

(c) ROM velocity (error magnitude)

(d) ROM pressure (error)

Figure 4.6: Backward–facing step: absolute errors of DD–FOM and ROM solutions w.r.t. the monolithic solution at the final time step.

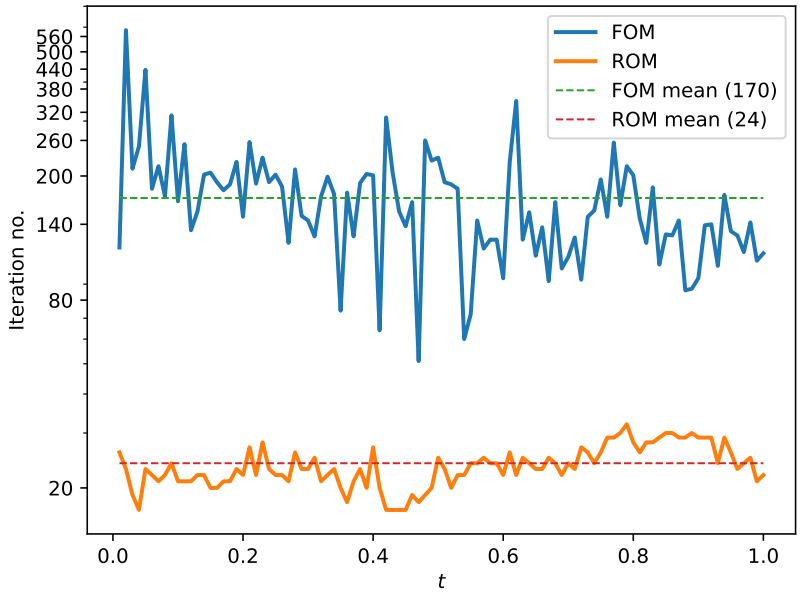
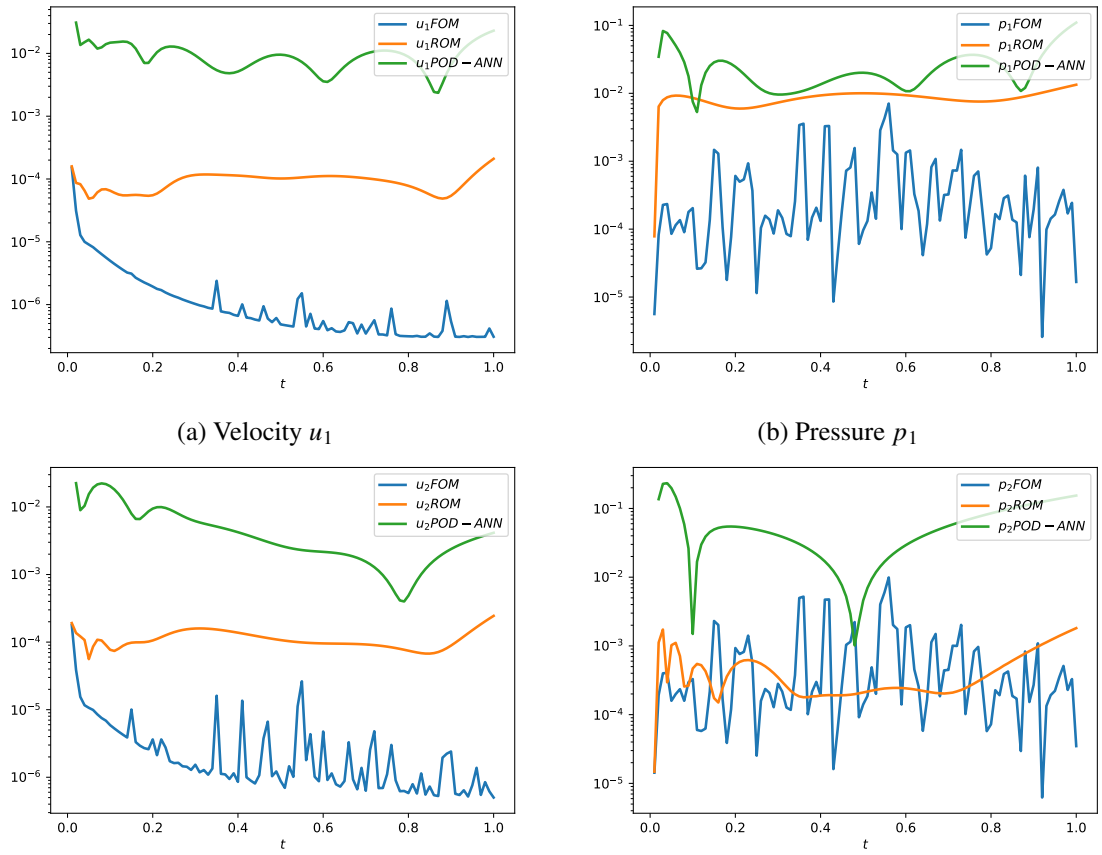


Figure 4.7: Backward–facing step: number of optimisation iterations of FOM and ROM solvers



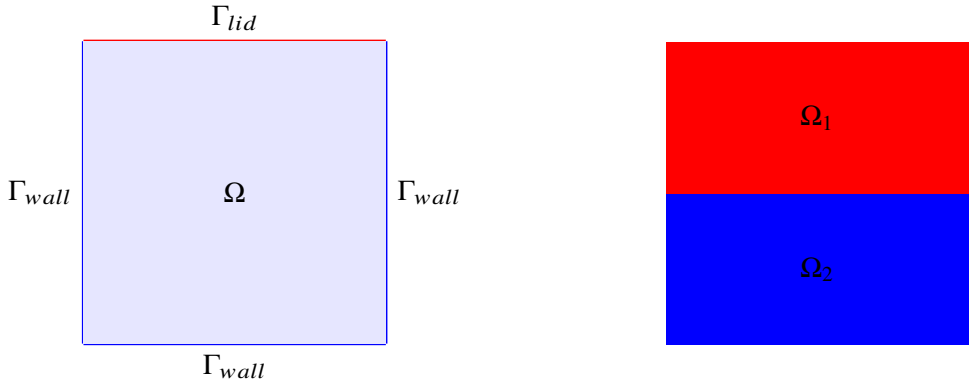
(a) Velocity u_1

(b) Pressure p_1

(c) Velocity u_2

(d) Pressure p_2

Figure 4.8: Backward–facing step: relative errors of FOM, ROM and POD–NN solutions w.r.t. the monolithic solution.



(a) Physical domain (b) Domain splitting

Figure 4.9: Lid–driven cavity flow geometry and domain decomposition.

time step was excluded from the training set in order to achieve better performance. In practice, this first step can be computed with a Galerkin projection or a FOM step. Figure 4.8 shows the relative errors with respect to the monolithic solution for the FOM, ROMs and POD–NN model. As we can see, both FOM and ROM give us very good convergence results – 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, 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 a much more computationally effective ANN model fails to produce a sufficient approximation, the ROM is applied. Similar ideas can be found *inter alia* in [16].

4.6.2 Lid–driven cavity flow test case

In this section, we provide the numerical simulation for the lid–driven cavity flow test case. Figure 4.9a represents the physical domain of interest – the unit square. The split into two domains is performed by dissecting the domain by a median horizontal line as shown in Figure 4.9b.

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}: $u_{lid} = \left(\bar{U}, 0\right)$; the values of $\bar{U}$ are reported in Table 4.3. We consider one physical parameter – the magnitude $\bar{U}$ of the lid velocity profile u_{lid}. Details of the offline stage and the FE discretisation are summarised in Table 4.3. 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

Physical parameters		FE parameters	
ν	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
$\text{it}_{\max}$	300	Parameters training set size K	10
tol_{opt}	10^{-7}	Maximum retained modes $N_{\max}$	100

Table 4.3: Lid–driven cavity flow: computational details of the offline stage

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

Table 4.4: Lid–driven cavity flow: Computational details of the online stage

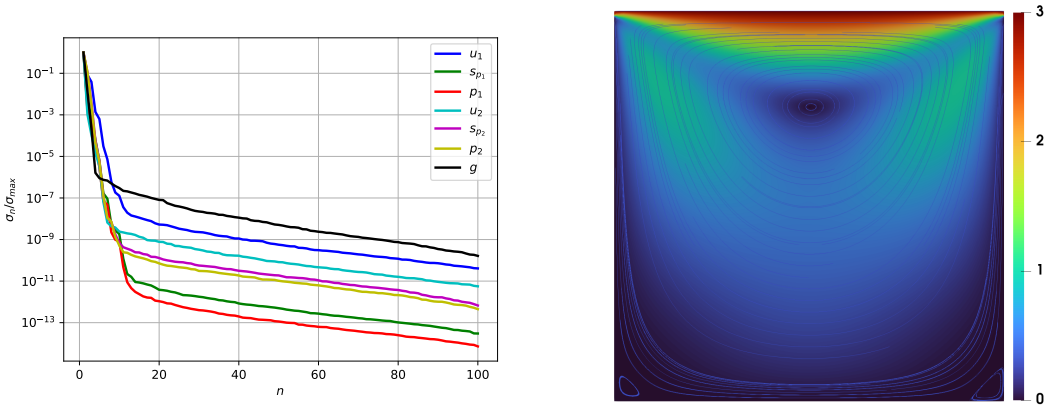
the first $N_{\max}$ POD modes have been retained for each component. In Figure 4.10a, we see that the POD singular values decay even faster than in the previous test for all the state and the control variables. As before, we show in Figure 4.10b the monolithic (whole–domain) solution related to the parameter ($\bar{U} = 3$) on which we will test the DD–FOM and the ROM.

In Table 4.4, 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 as used in the FOM, but on a smaller space with dimension 5 instead of 294. As an optimisation algorithm in this case we use the L–FBGS–B, but in this case, we use a smaller value for tol_{opt} of 10^{-6}.

Figures 4.11 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 4.10b.

Again, in Figure 4.12 we observe that the number of optimization iterations for FOM is between 10 and 100 times larger than the ROM ones. 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 4.4).

As in the previous test case, we would 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

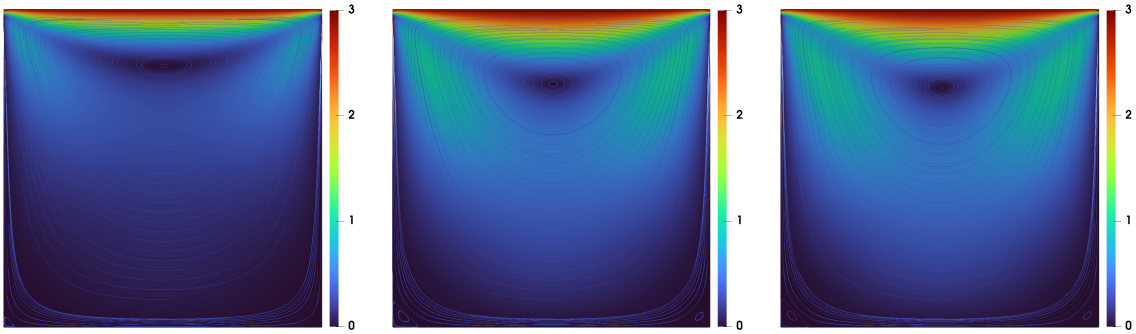


(a) The singular values as a function of the number of POD modes (log scale in y–direction)

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

Figure 4.10: 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).

in Section 4.5.3 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 4.13 shows the relative errors with respect to the monolithic solution for the FOM, ROMs and POD–NN model. 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



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

Figure 4.11: Lid–driven cavity flow: FOM velocity solution at 3 different time instances.

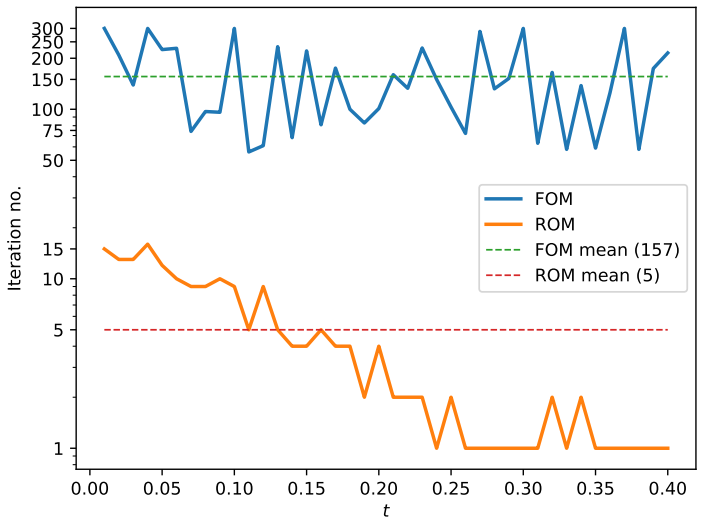


Figure 4.12: The number of optimisation iterations of FOM and ROM solvers

seconds.

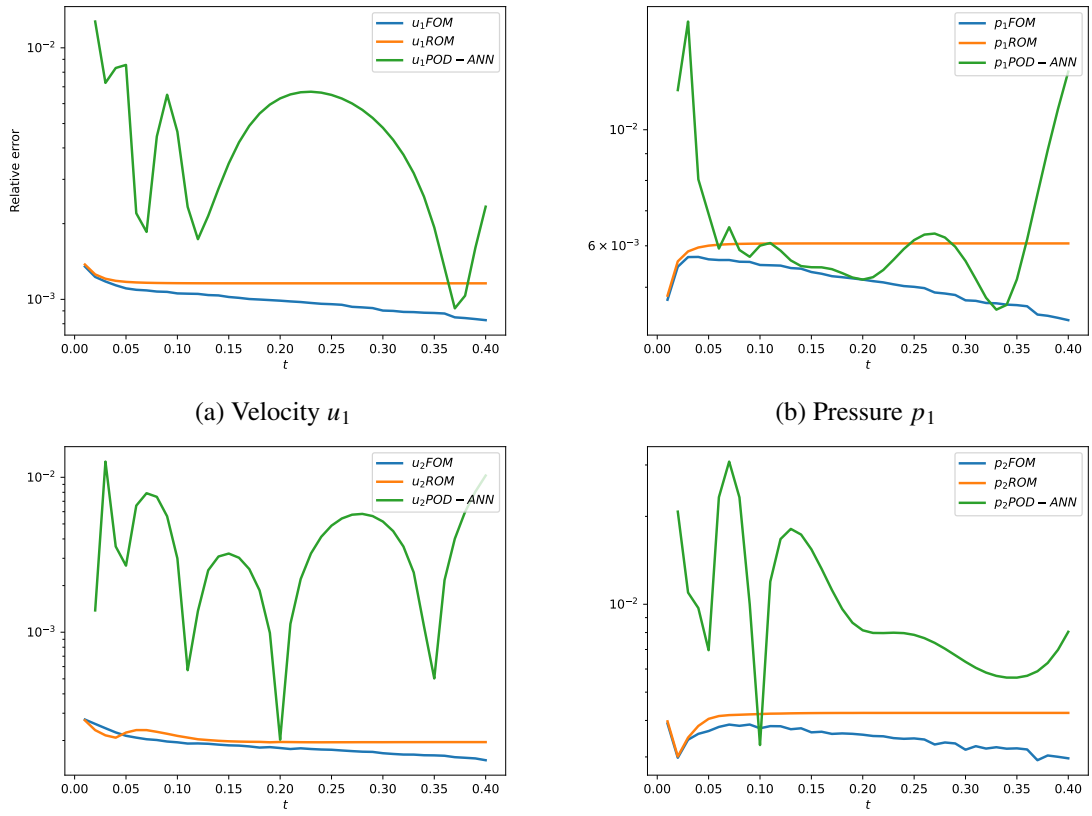
4.7 Conclusions and perspectives

In this chapter, we described and conducted the convergence analysis of an optimisation–based domain decomposition algorithm for the nonstationary incompressible 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 time required to solve the optimal–control problem was much shorter, not only because of the reduced dimensions but also because of the smaller number of iterations. The POD–NN results are less accurate, but the computational time is way shorter than the other methods.

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 a large number of degrees of freedom. Moreover, in the context of multiphysics, for instance, for fluid–structure interaction (see Chapter 5), 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



(a) Velocity u_1

(b) Pressure p_1

(c) Velocity u_2

(d) Pressure p_2

Figure 4.13: Lid–driven cavity flow: relative errors of FOM, ROM and POD–NN solutions w.r.t. the monolithic solution.

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.

Chapter 5

Fluid–Structure Interaction Problem

In this chapter, we introduce an optimisation–based domain–decomposition formulation of the non–stationary FSI problem with the incompressible Navier–Stokes equations and the linear elasticity model. Firstly, we provide a monolithic formulation and its time–discretisation scheme with the further derivation of the fully decoupled optimisation–based domain–decomposition formulation at each time step. Then, we obtain the optimality condition for the resulting optimal control problem and the expression for the gradient of the objective functional with the following listing of the gradient–based optimisation algorithm and the Gauss–Newton algorithm for the nonlinear least–squares minimisation problem. Furthermore, we develop a reduced–order model based on the Proper Orthogonal Decomposition methodology for the parameter–dependent FSI problem. At the end of the chapter, we show some numerical results on a two–dimensional haemodynamics benchmark FSI problem.

5.1 Problem formulation

In this section, we describe a formulation of the time–dependent Fluid–Structure Interaction (FSI) problem with incompressible Navier–Stokes equations for the fluid model and the linear elastic structure model. We start by introducing a general description of the FSI model with its corresponding coupling conditions between fluid and structure subcomponents. Then, we introduce a monolithic formulation of the problem and a notion of Arbitrary Lagrangian–Eulerian (ALE) formulation that will allow us to describe the fully coupled FSI problem in the reference frame of coordinates. 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 5.5.

5.1.1 FSI problem: general setting and coupling conditions

FSI problems describe the dynamical interplay between a fluid and a solid. This interplay takes place because of the coupling of the two different physics at the FSI interface, namely the part of the

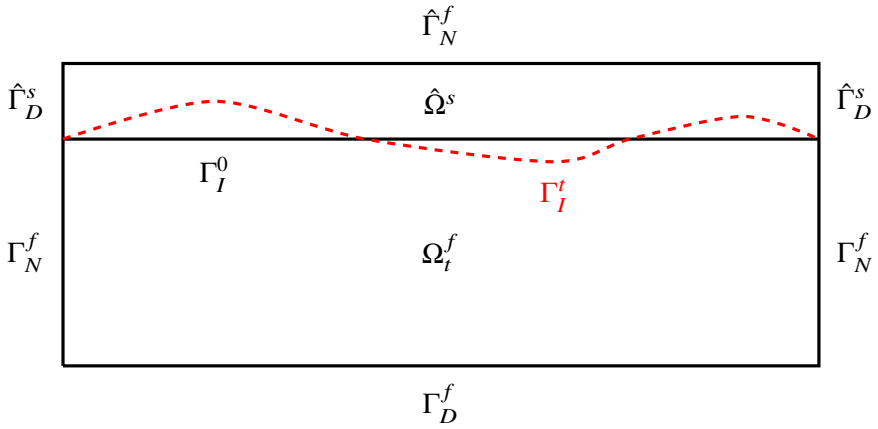


Figure 5.1: Domains and boundaries: Ω_t^f – moving fluid domain, $\hat{\Omega}^s$ – reference solid domain, Γ_I^0 and Γ_I^t (in red) – the fluid–structure interface in its reference and moving configuration, respectively

physical domain that is common to the fluid subdomain and the solid subdomain. The FSI interface profile is unknown *a priori* and depends on the dynamics of the fluid and of the structure problem. The coupling conditions are the result of three principles:

- Continuity of the velocities: this is a kinematic condition, that represents the hypothesis that the fluid sticks to the moving FSI interface;
- Balance of stresses: this is a classical action–reaction principle, that imposes the balance between the fluid and the solid stresses at the FSI interface;
- Continuity of the displacements: this geometrical condition imposes the continuity of the solid displacement and the fluid displacement (which will be described later via ALE map) at the FSI interface to guarantee that the fluid and the solid domains do not overlap.

5.1.2 Monolithic formulation

Let $\Omega_t^f \subset \mathbb{R}^2$ be a bounded moving domain at time $t \geq 0$ for the fluid, with the boundary $\Gamma_t^f = \Gamma_N^f \cup \Gamma_D^f \cup \Gamma_I^t$, and let $\hat{\Omega}^s \subset \mathbb{R}^2$ be a physical structure domain in its reference configuration, with the boundary $\hat{\Gamma}^s = \hat{\Gamma}_N^s \cup \hat{\Gamma}_D^s \cup \Gamma_I^0$; see Figure 5.1 for the details. We have this separation between the reference and current configurations of the domain of interest due to the fact that to describe the behaviour of a solid it is common practice to use the so–called Lagrangian formalism: all the quantities and the conservation laws are formulated on the reference configuration $\hat{\Omega}^s = \Omega^s(t = 0)$, whereas when describing the behaviour of a fluid, the Eulerian formalism is used instead: all the quantities and the conservation laws are formulated on the configuration Ω_t^f at the current time t. We consider a finite time interval $[0, T]$ with $T > 0$. Let $b_f : \Omega_t^f \times [0, T] \to \mathbb{R}^2$ and

$b_s : \hat{\Omega}^s \times [0,T] \to \mathbb{R}^2$ be the forcing terms for the fluid and structure subproblem, respectively; let u_N^f and $\hat{u}_N^s$ be a given Neumann data on Γ_N^f and $\hat{\Gamma}_N^s$, respectively. We further introduce u_f^0, the initial fluid velocity on Ω_0^f, $\hat{d}_0^s$ and $\hat{d}_0^{s,\partial t}$ are the initial condition for the structure displacement its first time derivative on $\hat{\Omega}^s$.
The FSI problem then reads as follows: find the velocity field $u_f \colon \Omega_t^f \times [0,T] \to \mathbb{R}^2$, the pressure $p_f \colon \Omega_t^f \times [0,T] \to \mathbb{R}$ and the displacement $\hat{d}_s \colon \hat{\Omega}^s \times [0,T] \to \mathbb{R}^2$ s.t.

$$\begin{aligned} \rho_f \left[\partial_t u_f + \left(u_f \cdot \nabla\right) u_f\right] - \mathrm{div}\sigma_f(u_f, p_f) &= b_f \quad \text{in } \Omega_t^f \times (0,T], && (5.1a)\\ -\mathrm{div} u_f &= 0 \quad \text{in } \Omega_t^f \times (0,T], && (5.1b)\\ \rho_s \partial_t^2 \hat{d}_s - \widehat{\mathrm{div}}\hat{P}(\hat{d}_s) &= \hat{b}_s \quad \text{in } \hat{\Omega}^s \times (0,T], && (5.1c) \end{aligned}$$

with the following boundary conditions:

$$\begin{aligned} u_f &= 0 \quad \text{on } \Gamma_D^f \times [0,T], && (5.2a)\\ \sigma_f(u_f, p_f)\mathbf{n}_f &= u_N^f \quad \text{on } \Gamma_N^f \times [0,T], && (5.2b)\\ \hat{d}_s &= 0 \quad \text{on } \hat{\Gamma}_D^s \times [0,T], && (5.2c)\\ \hat{P}(\hat{d}_s)\hat{\mathbf{n}}_s &= \hat{d}_N^s \quad \text{on } \hat{\Gamma}_N^s \times [0,T], && (5.2d) \end{aligned}$$

and initial conditions

$$\begin{aligned} u_f(t=0) &= u_0 \quad \text{in } \Omega_0^f, && (5.3a)\\ \hat{d}_s(t=0) &= \hat{d}_0^s \quad \text{in } \hat{\Omega}^s, && (5.3b)\\ \partial_t \hat{d}_s(t=0) &= \hat{d}_0^{s,\partial t} \quad \text{in } \hat{\Omega}^s, && (5.3c) \end{aligned}$$

as well as the following constitutive laws

- symmetric gradients:

$$\varepsilon(v) = \frac{1}{2}\left(\nabla v + \nabla^T v\right), \tag{5.4}$$

$$\hat{\varepsilon}(v) = \frac{1}{2}\left(\hat{\nabla} v + \hat{\nabla}^T v\right), \tag{5.5}$$

- second Piola–Kirchoff tensor for compressible linear elastic material:

$$\hat{P}(\hat{d}_s) = \lambda_s \mathrm{tr}\hat{\varepsilon}(\hat{d}_s) I + 2\mu_s \hat{\varepsilon}(\hat{d}_s), \tag{5.6}$$

- Cauchy stress tensor for the incompressible Newtonian fluid:

$$\sigma_f(u_f, p_f) = -p_f I + 2\rho_f \nu_f \varepsilon(u_f). \tag{5.7}$$

Remark (Non–homogeneous Dirichlet conditions). Notice that above, for the sake of simplicity, we chose to consider only homogeneous Dirichlet conditions on the fixed parts of the boundaries. This is due to the fact, that we do not want to complicate the exposition of Section 5.5 below. Nevertheless, the problem can be very easily adapted to the case with the non–homogeneous data in a very similar way as it is done in Chapters 3 and 4.

In the equations above, ρ_f and ρ_s are fluid and structure densities, respectively, ν_f – is the fluid viscosity whereas λ_s and μ_s are the Lamé coefficients for the solid. Due to the difference in the formalisms of description of the fluid and structure subsystems we have the following notation above: the gradients ∇ and $\hat{\nabla}$ indicate over which domain the differentiation takes place, that is, ∇ refers to the moving domain Ω_t^f and $\hat{\nabla}$ to the reference structure domain.

Due to the fact that fluid and structure equations are defined on different domains, in order to formalise the coupling conditions listed in the previous section, there is a need for a mapping between those subdomains. The most common technique used in such a framework is the Arbitrary Lagrangian–Eulerian (ALE) formulation. The aim of the following section is to describe how the ALE formalism works.

5.1.3 The Arbitrary Lagrangian–Eulerian formulation

The Arbitrary Lagrangian–Eulerian (ALE) [51, 153, 86, 22] method is one of the most widely used methods for the simulation of fluid flows in moving domains. In the ALE formulation, a one–to–one coordinate transformation is introduced for the fluid domain, and the fluid equations can be rewritten with respect to a fixed reference configuration. Specifically, we define the time–dependent bijective mapping $\mathcal{A}_t$ (see Figure 5.2) that maps the reference domain $\hat{\Omega}^f := \Omega_0^f$ to the physical domain Ω_t^f as follows: for $t \in [0, T]$

$$\begin{aligned} \mathcal{A}_t : \hat{\Omega}^f &\rightarrow \Omega_t^f, \\ \hat{x} &\mapsto x = \hat{x} + \hat{d}_f(\hat{x}, t), \end{aligned}$$

where $\hat{d}_f(\cdot, t) : \hat{\Omega}^f \rightarrow \Omega_t^f$ is the so called mesh displacement.

The definition of d_f usually depends on the model with a moving domain to be tackled. For FSI problems, the mesh displacement is defined as an extension of the solid displacement to the whole fluid domain. In this work, we will use a harmonic extension: find $\hat{d}_f : \hat{\Omega}^f \rightarrow \mathbb{R}^2$ such that

$$-\hat{\Delta}\hat{d}_f = 0 \quad \text{in} \quad \hat{\Omega}^f, \tag{5.8a}$$

$$\hat{d}_f = \hat{d}_s \quad \text{on} \quad \Gamma_I^0, \tag{5.8b}$$

where $\hat{d}_s$ is the structure displacement.

It should be highlighted that $\hat{d}_f$ is merely a geometrical quantity that takes into account the displacement of the mesh points and does not carry any physical meaning.

Great attention should be paid to the definition of fluid displacement, as different definitions of $\hat{d}_f$ may lead to different levels of continuity of the whole FSI problem. In the context of the linear

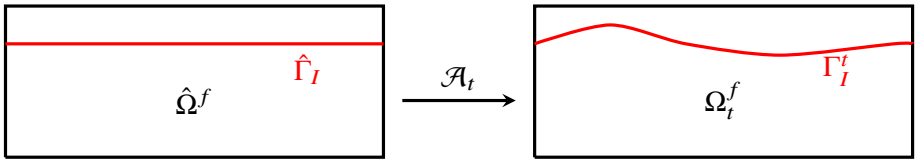


Figure 5.2: ALE map

elastic compressible structure model, the harmonic extension (5.8) of the structure displacement $\hat{d}_s$ usually does a very good job as we do not expect huge displacement of the interface; in case of large structure displacement, see, for example, [21]. For more details, we refer to [153].

5.1.4 ALE formulation of the FSI problem

With the formalism introduced in the previous section, we are now able to perform a pullback of the fluid equations with the moving domain onto the fluid reference configuration Ω^f, so that we are able to obtain a formulation of the FSI problem on the coherent subdomains. In order to perform a pullback of the Navier–Stokes equation, we define:

$$\hat{F} := \hat{\nabla}\mathcal{A}_t, \quad \hat{J} := \det\hat{F},$$

the deformation gradient and its determinant, respectively. These quantities allow us to rewrite the FSI dynamics problem in the reference configuration: for every $t \in [0,T]$, find $\hat{u}_f(t) : \hat{\Omega}^f \to \mathbb{R}^2$, $\hat{p}_f(t) : \hat{\Omega}^f \to \mathbb{R}$, $\hat{d}_f(t) : \hat{\Omega}^f \to \mathbb{R}^2$ and $\hat{d}_s(t) : \hat{\Omega}^s \to \mathbb{R}^2$ satisfying the following equations

$$\begin{aligned}
\rho_f \hat{J}\left[\partial_t \hat{u}_f + \hat{\nabla}\hat{u}_f \hat{F}^{-1}\left(\hat{u}_f - \partial_t \hat{d}_f\right)\right] - \widehat{\mathrm{div}}\left(\hat{J}\hat{\sigma}_f(\hat{u}_f,\hat{p}_f)\hat{F}^{-T}\right) &= b_f \quad \text{in } \hat{\Omega}^f \times (0,T], && \text{(5.9a)}\\
-\widehat{\mathrm{div}}\left(\hat{J}\hat{F}^{-1}\hat{u}_f\right) &= 0 \quad \text{in } \hat{\Omega}^f \times (0,T], && \text{(5.9b)}\\
-\hat{\Delta}\hat{d}_f &= 0 \quad \text{in } \hat{\Omega}^f \times (0,T], && \text{(5.9c)}\\
\rho_s \partial_t^2 \hat{d}_s - \widehat{\mathrm{div}}\hat{P}(\hat{d}_s) &= \hat{b}_s \quad \text{in } \hat{\Omega}_s \times (0,T], && \text{(5.9d)}
\end{aligned}$$

with the following boundary

$$\begin{aligned}
\hat{u}_f &= 0 \quad \text{on } \Gamma_D^f \times [0,T], && \text{(5.10a)}\\
\hat{\sigma}_f(\hat{u}_f,\hat{p}_f)\hat{\mathbf{n}}_f &= u_N^f \quad \text{on } \Gamma_N^f \times [0,T], && \text{(5.10b)}\\
\hat{d}_f &= 0 \quad \text{on } \left(\Gamma_D^f \cup \Gamma_N^f\right) \times [0,T], && \text{(5.10c)}\\
\hat{d}_s &= 0 \quad \text{on } \hat{\Gamma}_D^s \times [0,T], && \text{(5.10d)}\\
\hat{P}(\hat{d}_s)\hat{\mathbf{n}}_s &= \hat{d}_N^s \quad \text{on } \hat{\Gamma}_N^s \times [0,T], && \text{(5.10e)}
\end{aligned}$$

and initial conditions

$$\hat{u}_f(t=0) = u_0 \quad \text{in } \hat{\Omega}^f, \tag{5.11a}$$
$$\hat{d}_s(t=0) = \hat{d}_0^s \quad \text{in } \hat{\Omega}^s, \tag{5.11b}$$
$$\partial_t \hat{d}_s(t=0) = \hat{d}_0^{s,\partial t} \quad \text{in } \hat{\Omega}^s. \tag{5.11c}$$

In equation (5.9a), the tensor $\hat{\sigma}_f$ is the reference representation of the Cauchy stress tensor (5.7):

$$\hat{\sigma}_f(\hat{u}_f, \hat{p}_f) = \hat{\sigma}_f^{du}(\hat{d}_f, \hat{u}_f) + \hat{\sigma}_f^{p}(\hat{p}_f), \tag{5.12}$$

where

$$\hat{\sigma}_f^{du}(\hat{d}_f, \hat{u}_f) = \nu_f \left(\hat{\nabla}\hat{u}_f \hat{F}^{-1} + \hat{F}^{-T}\hat{\nabla}^T \hat{u}_f \right), \quad \hat{\sigma}_f^{p}(\hat{p}_f) = -\hat{p}_f I.$$

At this point, we are able to write down the coupling conditions described in Section 5.1.1. The system (5.9) – (5.11) is completed by the following conditions at the interface $\hat{\Gamma}_I$:

$$\hat{u}_f = \partial_t \hat{d}_s \quad \text{on } \hat{\Gamma}_I \times [0,T], \tag{5.13a}$$
$$\hat{J}\hat{\sigma}_f(\hat{u}_f, \hat{p}_f)\hat{F}^{-T}\hat{\mathbf{n}}_f = -\hat{P}(\hat{d}_s)\hat{\mathbf{n}}_s \quad \text{on } \hat{\Gamma}_I \times [0,T], \tag{5.13b}$$
$$\hat{d}_f = \hat{d}_s \quad \text{on } \hat{\Gamma}_I \times [0,T]. \tag{5.13c}$$

In the notations above, $\hat{\mathbf{n}}_f$ and $\hat{\mathbf{n}}_s$ denote the outward unit normal vectors on the boundaries of the fluid and the structure reference domains, respectively. From now on, we will solely work in the reference frame of coordinates, so that, in order to facilitate the description, we will drop the $\hat{}$ notation.

5.2 Domain Decomposition formulation

The scope of this section is to describe the Domain Decomposition (DD) formulation of the FSI problem obtained at the end of the previous section. We will start by considering two separate problems, namely, the fluid dynamics problem with the prescribed moving boundary and the structure problem with prescribed stresses on the boundary. These are followed by variational formulation and time–discretisation of both problems at the continuous level. At the end of the section, we present an optimisation–based domain–decomposition formulation at each time step.

5.2.1 Fluid and structure equations with given domain and stresses

Since we aim at completely decoupling the fluid and structure subsystems, we will describe two separate problems: one is the fluid problem with a prescribed moving domain, where the displacement of the moving interface Γ_I is prescribed by d_s^* and where the stresses on Γ_I are given by g_1, whereas the other is the structure problem with the stresses given by g_2.

Fluid problem:

- fluid momentum equation

$$\rho_f J \left[\partial_t u_f + \nabla u_f F^{-1} \left(u_f - \partial_t d_f\right)\right] - \text{div}\left(J\sigma_f(u_f, p_f)F^{-T}\right) = b_f \text{ in } \Omega^f \times (0,T], \quad (5.14)$$

- fluid incompressibility equation

$$-\text{div}\left(JF^{-1}u_f\right) = 0 \quad \text{in } \Omega^f \times (0,T], \quad (5.15)$$

- extension equation

$$-\Delta d_f = 0 \quad \text{in } \Omega^f \times (0,T], \quad (5.16)$$

- boundary conditions on the moving interface

$$J\sigma_f(u_f, p_f)F^{-T}\mathbf{n}_f = g_1 \quad \text{on } \Gamma_I \times [0,T], \quad (5.17a)$$
$$d_f = d_s^* \quad \text{on } \Gamma_I \times [0,T], \quad (5.17b)$$

- boundary conditions on the static boundary

$$u_f = 0 \quad \text{on } \Gamma_D^f \times [0,T], \quad (5.18a)$$
$$\sigma_f(u_f, p_f)\mathbf{n}_f = u_N^f \quad \text{on } \Gamma_N^f \times [0,T], \quad (5.18b)$$
$$d_f = 0 \quad \text{on } \left(\Gamma_D^f \cup \Gamma_N^f\right) \times [0,T], \quad (5.18c)$$

- initial condition

$$u_f(t=0) = u_0 \quad \text{in } \Omega^f. \quad (5.19)$$

Structure problem:

- extension equation

$$\rho_s \partial_t^2 d_s - \text{div}P(d_s) = b_s \quad \text{in } \Omega_s \times (0,T], \quad (5.20)$$

- boundary conditions on the moving interface

$$P(d_s)\mathbf{n}_s = g_2 \quad \text{on } \Gamma_I \times [0,T], \quad (5.21)$$

- boundary conditions on the static boundary

$$d_s = 0 \quad \text{on } \Gamma_D^s \times [0,T], \quad (5.22a)$$
$$P(d_s)\mathbf{n}_s = d_N^s \quad \text{on } \Gamma_N^s \times [0,T], \quad (5.22b)$$

- initial conditions

$$d_s(t=0) = d_0^s \quad \text{in } \Omega^s, \quad (5.23a)$$
$$\partial_t d_s(t=0) = d_0^{s,\partial t} \quad \text{in } \Omega^s. \quad (5.23b)$$

5.2.2 Variational (in space) formulation

In this section, we will provide variational formulations in space of the fluid and structure problems described in the previous section. Firstly, we define the following spaces:

$$\begin{aligned} V_f &= \left\{ v_f \in \left[H^1(\Omega^f)\right]^2 : \ v_f = 0 \text{ on } \Gamma_{f,D} \right\}, \\ Q_f &= L^2(\Omega^f), \\ E_f &= \left\{ v_f \in \left[H^1(\Omega^f)\right]^2 : \ v_f = 0 \text{ on } \Gamma_{f,D} \right\}, \\ E_{f,0} &= \left\{ v_f \in \left[H^1(\Omega^f)\right]^2 : \ v_f = 0 \text{ on } \Gamma_{f,D} \cup \Gamma_I \right\}, \\ E_s &= \left\{ e_s \in \left[H^1(\Omega^s)\right]^2 : \ e_s = 0 \text{ on } \Gamma_{s,D} \right\}. \end{aligned}$$

We equip the spaces V_f, E_f and $E_{f,0}$ with $H_0^1(\Omega^f)$–norm, the space Q_f with $L^2(\Omega^f)$–norm and the space E_s with $H_0^1(\Omega^s)$–norm. We also introduce the following notation: by $(\cdot,\cdot)_\omega$ we denote $L^2(\omega)$–inner product.

The variational counterparts of the equations (5.14)– (5.19) and (5.20)– (5.23) reads as follows: for every $t \in [0,T]$ find $u_f(t) \in V_f$, $p_f(t) \in Q_f$, $d_f(t) \in E_f$ and $d_s(t) \in E_s$ satisfying the following equations:

- fluid momentum equation

$$\begin{aligned} m_f(\partial_t u_f, v_f; d_f) &+ a_f(u_f, v_f; d_f) + c_f^{ALE}(\partial_t d_f, v_f, u_f; d_f) \\ &+ b_f^A(p_f, v_f; d_f) + c_f(u_f, u_f, v_f; d_f) \\ &= f_f(v_f; d_f) + \left(u_N^f, v_f\right)_{\Gamma_N^f} + (g, v_f)_{\Gamma_I} \qquad \forall v_f \in V_f, \end{aligned} \tag{5.24}$$

- incompressibility equation

$$b_f^B(u_f, q_f; d_f) = 0 \quad \forall q_f \in Q_f, \tag{5.25}$$

- extension equation

$$a_f^e(d_f, e_f) = 0 \quad \forall e_f \in E_{f,0}, \tag{5.26a}$$

$$d_f(t) = d_s^*(t) \quad \text{on } \Gamma_I, \tag{5.26b}$$

- structure equation

$$m_s(\partial_t^2 d_s, e_s) + a_s(d_s, e_s) = f_s(e_s) + \left(d_N^s, e_s\right)_{\Gamma_N^s} + (g_2, e_s)_{\Gamma_I} \quad \forall e_s \in E_s, \tag{5.27}$$

together with initial conditions (5.19) and (5.23). Above, by $d_s^*(t)$ we denote some extrapolation of $d_s(t)$ using previous time instances and the forms present in the equations are defined as follows:

- fluid velocity mass form

$$m_f(w, v_f; d_f) = \int_{\Omega^f} J\rho_f w \cdot v_f d\Omega, \tag{5.28}$$

- fluid stiffness form

$$a_f(u_f, v_f; d_f) = \int_{\Omega^f} J\sigma_f^{du}(d_f, u_f)F^{-T} : \nabla v_f d\Omega, \tag{5.29}$$

- fluid ALE form

$$c_f^{ALE}(w, v_f, u_f; d_f) = -\int_{\Omega^f} J\rho_f \left[\nabla u_f F^{-1}\right] w \cdot v_f d\Omega, \tag{5.30}$$

- fluid incompressibility forms

$$b_f^A(p_f, v_f; d_f) = \int_{\Omega^f} J\sigma_f^p(p_f)F^{-T} : \nabla v_f d\Omega, \tag{5.31}$$

$$b_f^B(u_f, q_f; d_f) = -\int_{\Omega^f} \text{div}\left(JF^{-1}u_f\right) q_f d\Omega, \tag{5.32}$$

- fluid advection form

$$c_f(u_f, w_f, v_f; d_f) = \int_{\Omega^f} J\rho_f \left[\nabla w_f F^{-1}\right] u_f \cdot v_f d\Omega, \tag{5.33}$$

- fluid forcing form

$$f_f(v_f; d_f) = \int_{\Omega^f} Jb_f \cdot v_f d\Omega, \tag{5.34}$$

- extension fluid stiffness form

$$a_f^e(d_f, e_f) = \int_{\Omega^f} \nabla d_f : \nabla e_f d\Omega, \tag{5.35}$$

- structure displacement mass form

$$m_s(w, e_s) = \int_{\Omega^s} \rho_s w \cdot e_s d\Omega, \tag{5.36}$$

- structure stiffness form

$$a_s(d_s, e_s) = \int_{\Omega^s} P(d_s) : \nabla e_s d\Omega, \tag{5.37}$$

- structure forcing form

$$f_s(e_s) = \int_{\Omega^s} b_s \cdot e_s d\Omega. \tag{5.38}$$

5.2.3 Discretisation in time

We will proceed with the time discretisation of the variational equation obtained in the previous section. The discretisation of the fluid equations will be carried on by the implicit Euler scheme, while the time discretisation of the solid equation will use the central second–order difference scheme. Let us introduce the following notation: we denote by ℓ^n the function $\ell(t^n)$ at time t^n. The resulting time discretisation is then given as follows: for $n \geq 1$ find $u_f^n \in V_f, p_f^n \in Q_f, d_f^n \in E_f$ and $e_s^n \in E_s$ satisfying the following equations:

- fluid momentum equation

$$\begin{aligned} \frac{1}{\Delta t} m_f(u_f^n, v_f; d_f) &+ a_f(u_f^n, v_f; d_f^n) + \frac{1}{\Delta t} c_f^{ALE}(d_f^n, v_f, u_f^n; d_f^n) \\ &+ b_f^A(p_f^n, v_f; d_f^n) + c_f(u_f^n, u_f^n, v_f; d_f^n) \\ &= \frac{1}{\Delta t} m_f(u_f^{n-1}, v_f; d_f) + f_f(v_f; d_f^n) + \left(u_N^f, v_f\right)_{\Gamma_N^f} \\ &+ \frac{1}{\Delta t} c_f^{ALE}(d_f^{n-1}, v_f, u_f^n; d_f^n) + (g_1^n, v_f)_{\Gamma_I} \qquad \forall v_f \in V_f, \end{aligned} \tag{5.39}$$

- incompressibility equation

$$b_f^B(u_f^n, q_f; d_f^n) = 0 \quad \forall q_f \in Q_f, \tag{5.40}$$

- extension equation

$$a_f^e(d_f^n, e_f) = 0 \quad \forall e_f \in E_{f,0}, \tag{5.41a}$$

$$d_f^n = d_s^{*,n} \quad \text{on } \Gamma_I, \tag{5.41b}$$

- structure equation

$$\begin{aligned} \frac{1}{\Delta t^2} m_s(d_s^n, e_s) + a_s(d_s^n, e_s) &= \frac{2}{\Delta t^2} m_s(d_s^{n-1}, e_s) + f_s(e_s) + \left(d_N^s, e_s\right)_{\Gamma_N^s} \\ &- \frac{1}{\Delta t^2} m_s(d_s^{n-2}, e_s) + (g_2^n, e_s)_{\Gamma_I} \qquad \forall e_s \in E_s, \end{aligned} \tag{5.42}$$

and

$$u_f^0 = u_0, \quad d_s^{-1} = d_0 - \Delta t d_s^{0,\partial t}, \quad d_s^0 = d_0. \tag{5.43}$$

As before, $d_s^{*,n}$ is an extrapolation of the quantity d_s^n obtained using previous time–step values, for example, $d_s^{*,n}$ can be defined as

$$d_s^{*,n} := 2d_s^{n-1} - d_s^{n-2}. \tag{5.44}$$

5.2.4 Domain Decomposition formulation

Now, having in hand all the necessary ingredients, we are able to write down an optimisation–based domain–decomposition algorithm for the FSI problem (5.9) – (5.13). It reads as follows: minimise over $g \in L^2(\Gamma_I)$ the functional

$$J_\gamma(u_f^n, d_s^n; g) = \frac{1}{2}\int_{\Gamma_I} \left| u_f^n - \frac{3d_s^n - 4d_s^{n-1} + d_s^{n-2}}{2\Delta t} \right|^2 d\Gamma + \frac{\gamma}{2}\int_{\Gamma_I} |g|^2 \, d\Gamma \tag{5.45}$$

subject to the following state equations for $n \geq 1$:

- fluid momentum equation

$$\begin{aligned} \frac{1}{\Delta t} m_f(u_f^n, v_f; d_f^n) &+ a_f(u_f^n, v_f; d_f^n) + \frac{1}{\Delta t} c_f^{ALE}(d_f^n, v_f, u_f^n; d_f^n) \\ &+ b_f^A(p_f^n, v_f; d_f^n) + c_f(u_f^n, u_f^n, v_f; d_f^n) \\ &= \frac{1}{\Delta t} m_f(u_f^{n-1}, v_f; d_f^n) + f_f(v_f; d_f^n) + \left(u_N^f, v_f\right)_{\Gamma_N^f} \\ &+ \frac{1}{\Delta t} c_f^{ALE}(d_f^{n-1}, v_f, u_f^n; d_f^n) + (g, v_f)_{\Gamma_I} \qquad \forall v_f \in V_f, \end{aligned} \tag{5.46}$$

- incompressibility equation

$$b_f^B(u_f^n, q_f; d_f^n) = 0 \quad \forall q_f \in Q_f, \tag{5.47}$$

- extension equation

$$a_f^e(d_f^n, e_f) = 0 \quad \forall e_f \in E_{f,0}, \tag{5.48a}$$

$$d_f^n = d_s^{*,n} \quad \text{on } \Gamma_I, \tag{5.48b}$$

- structure equation

$$\begin{aligned} \frac{1}{\Delta t^2} m_s(d_s^n, e_s) + a_s(d_s^n, e_s) &= \frac{2}{\Delta t^2} m_s(d_s^{n-1}, e_s) + f_s(e_s) + \left(d_N^s, e_s\right)_{\Gamma_N^s} \\ &- \frac{1}{\Delta t^2} m_s(d_s^{n-2}, e_s) - (g, e_s)_{\Gamma_I} \qquad \forall e_s \in E_s, \end{aligned} \tag{5.49}$$

and

$$u_f^0 = u_0, \quad d_s^{-1} = d_0 - \Delta t d_s^{0,\partial t}, \quad d_s^0 = d_0. \tag{5.50}$$

It can be easily seen that thanks to the fact that we use an extrapolation for the fluid–displacement problem (5.48), the variable d_f^n can be computed at the beginning of the time step and then substituted into the fluid momentum and incompressibility equations.

5.3 Optimality system and optimisation algorithms

In this section, we will present some techniques for dealing with the optimal control problem introduced at the end of Section 5.2. In the first part, we will derive an optimality system via Lagrange multiplier techniques. This will allow us to find the formula of the objective functional gradient that will open many ways of tackling a minimisation problem by gradient–based optimisation algorithm. In the second part, we will rewrite the problem in the context of non–linear least square minimisation problem and describe the Gauss–Newton optimisation technique.

5.3.1 Optimality system

One of the ways to address the constrained optimisation problem is to reformulate the initial problem in terms of a Lagrangian functional by introducing the so–called adjoint variables. In this way, the optimal solution to the original problem is sought among the stationary points of the Lagrangian, see, for instance, [76, 83].

We define two sets of variable: the state variables $V^n = \left\{u_f^n, p_f^n, d_s^n\right\} \in V_f \times Q_f \times E_s$ and the adjoint variables $\Theta = \left\{\xi_f, \eta_f, \xi_s\right\} \in V_f \times Q_f \times E_s$. The Lagrange functional for our optimal control problem is defined as follows:

$$\begin{aligned}
\mathcal{L}\left(V^n, \Theta, d_f^n, g\right) &= J_\gamma(u_f^n, d_s^n; g) - \left[\frac{1}{\Delta t} m_f(u_f^n, \xi_f; d_f^n) + a_f(u_f^n, \xi_f; d_f^n)\right. \\
&+ \frac{1}{\Delta t} c_f^{ALE}(d_f^n, \xi_f, u_f^n; d_f^n) + b_f^A(p_f^n, \xi_f; d_f^n) + c_f(u_f^n, u_f^n, \xi_f; d_f^n) \\
&- \left.\frac{1}{\Delta t} m_f(u_f^{n-1}, \xi_f; d_f^n) - f_f^n(\xi_f; d_f^n) - \frac{1}{\Delta t} c_f^{ALE}(d_f^{n-1}, \xi_f, u_f^n; d_f^n)\right] \\
&- b_f^B(u_f^n, \eta_f; d_f^n) - \left[\frac{1}{\Delta t^2} m_s(d_s^n, \xi_s) + a_s(d_s^n, \xi_s) - f_s^n(\xi_s)\right] \\
&+ \left(u_N^f, \xi_f\right)_{\Gamma_N^f} + \left(d_N^s, \xi_s\right)_{\Gamma_N^s} + (g, \xi_f)_{\Gamma_I} - (g, \xi_s)_{\Gamma_I}.
\end{aligned} \tag{5.51}$$

As has been pointed out before, the fluid displacement d_f^n can be precomputed before, so that we do not need to introduce a Lagrange multiplier or a lifting function for this variable.

We now apply the necessary conditions for finding stationary points of $\mathcal{L}$. Setting to zero the first variations w.r.t. $\xi_f \in V_f$ and $\eta_f \in Q_f$ and $\xi_s \in E_s$ yields the state equations (5.46)– (5.47) and (5.49). Setting to zero the first variations w.r.t. u_f^n, p_f^n and e_s^n yields the adjoint equations:

- adjoint fluid momentum equation

$$\begin{aligned}\frac{1}{\Delta t}m_f(\delta v_f,\xi_f;d_f^n) &+ a_f(\delta v_f,\xi_f;d_f^n)+c_f(\delta v_f,u_f^n,\xi_f;d_f^n)\\ &+ c_f(u_f^n,\delta v_f,\xi_f;d_f^n)+b_f^B(\delta v_f,\eta_f;d_f^n)\\ &+ c_f^{ALE}\left(\frac{d_f^n-d_f^{n-1}}{\Delta t},\xi_f,\delta v_f;d_f^n\right)\\ &= \left(u_f^n-\frac{3d_s^n-4d_s^{n-1}+d_s^{n-2}}{2\Delta t},\delta v_f\right)_{\Gamma_I}\quad \forall \delta v_f\in V_f,\end{aligned} \tag{5.52}$$

- adjoint incompressibility equation

$$b_f^A(\xi_f,\delta q_f;d_f^n)=0\quad\forall\delta q_f\in Q_f, \tag{5.53}$$

- adjoint structure equation

$$\begin{aligned}&\frac{1}{\Delta t^2}m_s(\delta e_s,\xi_s)+a_s(\delta e_s,\xi_s)\\ &\qquad=\left(u_f^n-\frac{3d_s^n-4d_s^{n-1}+d_s^{n-2}}{2\Delta t},-\frac{3}{2\Delta t}\delta e_s\right)_{\Gamma_I}\quad\forall\delta e_s\in E_s.\end{aligned} \tag{5.54}$$

Finally, setting to zero the first variations w.r.t. $g\in L^2(\Gamma_I)$ yields the optimality condition:

$$\gamma\,(h,g)_{\Gamma_I}+\left(h,\xi_f\right)_{\Gamma_I}-(h,\xi_s)_{\Gamma_I}=0\quad\forall h\in L^2(\Gamma_I). \tag{5.55}$$

5.3.2 Sensitivity derivatives

In order to obtain the expression for the gradient of the optimisation problem at hand, we will resort to the sensitivity derivatives approach; see, for instance, [76, 83]. The approach consists of finding equations for directional derivatives of the state variables with respect to the control, called sensitivities.
The first derivative $\frac{d\mathcal{J}_\gamma}{dg}$ of $\mathcal{J}_\gamma$ is defined through its action on the variation $\tilde{g}$ as follows:

$$\left\langle\frac{dJ_\gamma}{dg},\tilde{g}\right\rangle=\left(u_f^n-\frac{3d_s^n-4d_s^{n-1}+d_s^{n-2}}{2\Delta t},\tilde{u}_f-\frac{3}{2\Delta t}\tilde{d}_s\right)_{\Gamma_I}+\gamma(g,\tilde{g})_{\Gamma_I}, \tag{5.56}$$

where $\tilde{u}_f\in V_f$, $\tilde{p}_f\in Q_f$ and $\tilde{d}_s\in E_s$ are the solutions to:

- sensitivity fluid momentum equation

$$\begin{aligned}\frac{1}{\Delta t}m_f(\tilde{u}_f,v_f;d_f^n) &+ a_f(\tilde{u}_f,v_f;d_f^n)+c_f^{ALE}\left(\frac{d_f^n-d_f^{n-1}}{\Delta t},v_f,\tilde{u}_f;d_f^n\right)\\ &+ b_f^A(\tilde{p}_f,v_f;d_f^n)+c_f(\tilde{u}_f,u_f^n,v_f;d_f^n)\\ &+ c_f(u_f^n,\tilde{u}_f,v_f;d_f^n)=(\tilde{g},v_f)_{\Gamma_I}\quad\forall v_f\in V_f,\end{aligned} \tag{5.57}$$

- sensitivity incompressibility equation

$$b_f^B(\tilde{u}_f, q_f; d_f^n) = 0 \quad \forall q_f \in Q_f, \tag{5.58}$$

- sensitivity structure equation

$$\frac{1}{\Delta t^2} m_s(\tilde{d}_s, e_s) + a_s(\tilde{d}_s, e_s) = -(\tilde{g}, e_s)_{\Gamma_I} \quad \forall e_s \in E_s. \tag{5.59}$$

We can make use of the adjoint equations (5.52)– (5.54) in order to find the representation of the gradient of the functional $\mathcal{J}_\gamma$. Let ξ_f, η_f and ξ_s be the solutions to (5.52)– (5.54) and $\tilde{u}_f$, $\tilde{p}_f$ and $\tilde{d}_s$ be the solutions to (5.57)– (5.59). By setting $\delta v_f := \tilde{u}_f$ in (5.52), $\delta q_f := \tilde{p}_f$ in (5.53), $\delta e_s := \tilde{d}_s$ in (5.54), and $v_f := \xi_f$ in (5.57) and $q_f := \eta_f$ in (5.58), $e_s := \xi_s$ in (5.59), we obtain:

$$\left(u_f^n - \frac{3d_s^n - 4d_s^{n-1} + d_s^{n-2}}{2\Delta t}, \tilde{u}_f - \frac{3}{2\Delta t}\tilde{d}_s \right)_{\Gamma_I} = (\tilde{g}, \xi_f)_{\Gamma_I} - (\tilde{g}, \xi_s)_{\Gamma_I},$$

so that it yields the explicit formula for the gradient of $\mathcal{J}_\gamma$:

$$\frac{d\mathcal{J}_\gamma}{dg}(u_f^n, d_s^n; g) = \gamma g + \xi_f|_{\Gamma_I} - \xi_s|_{\Gamma_I}, \tag{5.60}$$

where ξ_f and ξ_s are determined from g through (5.52)– (5.54). Notice that the gradient expression (5.60) is consistent with the optimality condition (5.55) derived in the previous section.

5.3.3 Gradient–based algorithm for the optimisation problem

In view of being able to provide a closed–form formula for the gradient for the objective functional $\mathcal{J}_\gamma$, the natural way to proceed is to resort to a gradient–based iterative optimisation algorithm. In order to keep the exposition simple, we will describe the idea using a simple gradient method with a fixed step, while, in practice, we will use more sophisticated gradient–based methods. For every time step t^n, given an initial guess $g^{(0)}$, which we set from the previous time step, we update successive values of $g^{(j)}$ with

$$g^{(j+1)} = g^{(j)} - \alpha \frac{d\mathcal{J}_\gamma}{dg}\left(u_f^{n,(j)}, d_s^{n,(j)}; g^{(j)} \right). \tag{5.61}$$

Combining this with (5.60) we obtain:

$$g^{(j+1)} = g^{(j)} - \alpha \left(\gamma g^{(j)} + \xi_f^{(j)}\Big|_{\Gamma_I} - \xi_s^{(j)}\Big|_{\Gamma_I} \right), \tag{5.62}$$

where $\xi_f^{(j)}$ and $\xi_s^{(j)}$ are determined from (5.52)– (5.54) with g replaced by $g^{(j)}$.

To summarise, we describe in Algorithm 5.1 the procedure used to find g^n at every time step t^n. Some of the common convergence criteria for Algorithm 5.1 are values of the functional or

Algorithm 5.1 Gradient–based algorithm for optimisation

Input: $g^{(0)} := g^{n-1}$, $\alpha > 0$
$j := 0$
while Convergence criteria are not met **do**
  Solve (5.46)– (5.49) for $u_f^{n,(j)} \in V_f$, $d_s^{n,(j)} \in E_s$ with $g = g^{(j)}$
  Solve (5.52)– (5.54) for $\xi_f^{(j)} \in V_f$, $\xi_s^{(j)} \in E_s$ with $u_f^n = u_f^{n,(j)}$, $d_s^n = d_s^{n,(j)}$
  Update $g^{(j+1)} := (1 - \alpha\gamma)\, g^{(j)} - \alpha \left(\xi_f^{(j)}\Big|_{\Gamma_I} - \xi_s^{(j)}\Big|_{\Gamma_I} \right)$
  $j := j + 1$
end while
return $g^n := g^{(j)}$

of the gradient norm smaller than a given tolerance and the maximum number of optimisation iterations. Most commonly, a couple of them are used together. In practice, the methods we will use to solve such problems are Broyden–Fletcher–Goldfarb–Shanno (BFGS) and Newton Conjugate Gradient (CG) algorithms, which show faster convergence and higher efficiency with respect to the steepest–descent algorithm.

Remark (Sensitivity of "black–box" gradient optimisation algorithms to small data). In the numerical results presented at the end of this chapter, the structure (and consequently fluid) displacement has a very small magnitude. In this case, the "black–box" gradient optimisation algorithm as, for example, the algorithms from the `scipy` library [183] used for the simulations in Chapters 3 and 4, sometimes show very slow convergence. This prompted us to search for different optimisation methods, and one of these will be described in the following section. Nevertheless, we believe that in other FSI settings, the procedure above might be very effective.

5.3.4 Non–linear least squares setting

We use non–linear least squares to develop a computational algorithm for the constrained optimal control problem.

We define the non–linear operator $N_n : L^2(\Gamma_I) \to L^2(\Gamma_I) \times L^2(\Gamma_I)$ by

$$N_n(g) = \begin{pmatrix} u_f^n\Big|_{\Gamma_I} - \frac{3d_s^n - 4d_s^{n-1} + d_s^{n-2}}{2\Delta t}\Big|_{\Gamma_I} \\ \sqrt{\gamma} g \end{pmatrix}, \tag{5.63}$$

where u_f^n and d_s^n are solutions to the state equations (5.46)– (5.49) with g as a stress function on the interface Γ_I. Then, the functional $\mathcal{J}_\gamma$ can be rewritten as

$$J_\gamma(u_f^n, d_s^n; g) = \frac{1}{2} ||N_n(g)||^2_{L^2(\Gamma_I)\times L^2(\Gamma_I)}. \tag{5.64}$$

The non–linear least square optimisation problem is the following: minimise the functional (5.64) over $g \in L^2(\Gamma_I)$ subject to (5.46)– (5.49).

To tackle this problem, we undertake the strategy described in [77, 98]. The functional N_n can be linearised at a point $\bar{g} \in L^2(\Gamma_I)$ by using its Fréchet derivative $N_n'(\bar{g}) : L^2(\Gamma_I) \to L^2(\Gamma_I) \times L^2(\Gamma_I)$ by

$$N_n(g) = N_n(\bar{g}) + N_n'(\bar{g})[g - \bar{g}] + O\left(||g - \bar{g}||^2_{L^2(\Gamma_I)\times L^2(\Gamma_I)}\right).$$

Then, solutions of the nonlinear least squares problem can be obtained by repeatedly solving the linear least squares problem: minimise over $h \in L^2(\Gamma_I)$ the following

$$\frac{1}{2}||N(\bar{g}) + N_n{}'(\bar{g})[h]||^2_{L^2(\Gamma_I)\times L^2(\Gamma_I)}, \tag{5.65}$$

where $h = g - \bar{g}$. Hence, starting with arbitrary $g^{(0)}$, we can find a sequence $\{g^{(k)}\}$ obtained by $g^{(k)} = g^{(k-1)} + h^{(k)}$, where $h^{(k)}$ is the solution of the linear least square problem above. We will make use of the conjugate gradient algorithm.

5.3.5 Fréchet derivative and its conjugate

For $\bar{g} \in L^2(\Gamma_I)$, the Fréchet derivative $N_n{}'(\bar{g})[\cdot] : L^2(\Gamma_I) \to L^2(\Gamma_I) \times L^2(\Gamma_I)$ is defined by

$$N_n{}'(\bar{g})[h] = \begin{pmatrix} w_f\big|_{\Gamma_I} - \frac{3}{2\Delta t}\ \phi_s|_{\Gamma_I} \\ \sqrt{\gamma} h \end{pmatrix}, \tag{5.66}$$

where w_f and ϕ_s are the solutions to

$$\begin{aligned} \frac{1}{\Delta t} m_f(w_f, v_f; d_f^n) &+ a_f(w_f, v_f; d_f^n) + c_f^{ALE}\left(\frac{d_f^n - d_f^{n-1}}{\Delta t}, v_f, w_f; d_f^n\right) \\ &+ b_f^A(\psi_f, v_f; d_f^n) + c_f(w_f, \bar{u}_f, v_f; d_f^n) \\ &+ c_f(\bar{u}_f, w_f, v_f; d_f^n) = (h, v_f)_{\Gamma_I} \quad \forall v_f \in V_f, \qquad (5.67) \\ b_f^B(w_f, q_f; d_f^n) &= 0 \quad \forall q_f \in Q_f, \qquad (5.68) \end{aligned}$$

and

$$\frac{1}{\Delta t^2} m_s(\phi_s, e_s) + a_s(\phi_s, e_s) = -(h, e_s)_{\Gamma_I} \quad \forall e_s \in E_s, \tag{5.69}$$

where $\bar{u}_f$ is the solution to the fluid state equations (5.46)– (5.49) with g replaced by $\bar{g}$.

The adjoint of $N_n{}'(\bar{g})[\cdot]$ is $(N_n{}'(\bar{g}))^*\,[\cdot] : L^2(\Gamma_I) \times L^2(\Gamma_I) \to L^2(\Gamma_I)$, given by

$$(N_n{}'(\bar{g}))^* \begin{bmatrix} r \\ s \end{bmatrix} = \beta_f\big|_{\Gamma_I} - \frac{3}{2\Delta t}\ \varphi_s|_{\Gamma_I} + \sqrt{\gamma} s, \tag{5.70}$$

where β_f and φ_s are solutions of

$$\begin{aligned}\frac{1}{\Delta t}m_f(v_f,\beta_f;d_f^n) &+ a_f(v_f,\beta_f;d_f^n) + c_f^{ALE}\left(\frac{d_f^n-d_f^{n-1}}{\Delta t},\beta_f,v_f;d_f^n\right)\\ &+ b_f^B(v_f,\alpha_f;d_f^n) + c_f(v_f,\bar{u}_f,\beta_f;d_f^n)\\ &+ c_f(\bar{u}_f,v_f,\beta_f;d_f^n) = (r,v_f)_{\Gamma_I} \quad \forall v_f\in V_f, \qquad (5.71)\\ b_f^A(q_f,\beta_f;d_f^n) &= 0 \quad \forall q_f\in Q_f, \qquad (5.72)\end{aligned}$$

and

$$\frac{1}{\Delta t^2}m_s(e_s,\varphi_s)+a_s(e_s,\varphi_s) = -(r,e_s)_{\Gamma_I} \quad \forall e_s\in E_s. \qquad (5.73)$$

Indeed, by setting $v_f := w_f$ in (5.71), $q_f := \psi_f$ in (5.72) and $e_s := \phi_s$ in (5.73), we get

$$\begin{aligned}(w_f,r)_{\Gamma_I} &= \frac{1}{\Delta t}m_f(w_f,\beta_f;d_f^n) + a_f(w_f,\beta_f;d_f^n) + c_f^{ALE}\left(\frac{d_f^n-d_f^{n-1}}{\Delta t},\beta_f,w_f;d_f^n\right)\\ &+ b_f^B(w_f,\alpha_f;d_f^n) + c_u(\bar{u}_f,w_f,\beta_f;d_f^n) + c_u(w_f,\bar{u}_f,\beta_f;d_n^f)\\ &+ b_f^A(\psi_f,\beta_f;d_f^n) = (h,\beta_f)_{\Gamma_I}\end{aligned}$$

and

$$(-\phi_s,r)_{\Gamma_I} = \frac{1}{\Delta t^2}m_s(\phi_s,\varphi_s)+a_s(\phi_s,\varphi_s) = -(h,\varphi_s)_{\Gamma_I}.$$

This leads to

$$\left(w_f-\frac{3}{2\Delta t}\phi_s,r\right)_{\Gamma_I} = (\beta_f-\frac{3}{2\Delta t}\varphi_s,h)_{\Gamma_I}$$

that, in turn, gives us

$$\begin{aligned}\left\langle (N_n{}'(\bar{g})[h],\begin{pmatrix}r\\s\end{pmatrix}\right\rangle &= \left(w_f-\frac{3}{2\Delta t}\phi_s,r\right)_{\Gamma_I}+\sqrt{\gamma}(h,s)_{\Gamma_I}\\ &= \left(\beta_f-\frac{3}{2\Delta t}\varphi_s,h\right)_{\Gamma_I}+\sqrt{\gamma}(h,s)_{\Gamma_I}\\ &= \left\langle h,(N_n{}'(\bar{g}))^*\begin{bmatrix}r\\s\end{bmatrix}\right\rangle.\end{aligned}$$

The Fréchet derivative (5.66) and its conjugate (5.70) derived here are the key elements of the Gauss–Newton optimisation algorithm that will be described in the next section.

Algorithm 5.2 Gauss–Newton (GN) algorithm

Input: $g^{(0)} := g^{n-1}$, It_{max} and ε_{tol}
for $j = 1, \dots, It_{max}$ **do**
  Solve (5.46)– (5.47) for $u_f^{n,(j)} \in V_f$ and $p_f^{n,(j)} \in Q_f$ with $g = g^{(j-1)}$
  Solve (5.49) for $d_s^{n,(j)} \in E_s$ with $g = g^{(j-1)}$
  Solve (5.48) for $d_f^{n,(j)}$ with $d_s^{*,n} = d_s^{n,(j)}$
  if $\frac{1}{2}\int_{\Gamma_I} \left| u_f^{n,(j)} - \frac{3d_s^{n,(j)} - 2d_s^{n-1} + d_s^{n-2}}{2\Delta t} \right|^2 d\Gamma < \varepsilon_{tol}$ **then**
    `break`
  end if
  Compute $h^{(j)}$ by the Conjugate Gradient Algorithm 5.3 with $A = N_n'\left(g^{(j-1)}\right)$, $b = -N_n\left(g^{(j-1)}\right)$ and $A^* = \left(N_n'\left(g^{(j-1)}\right)\right)^*$
  Set $g^{(j)} := g^{(j-1)} + h^{(j)}$.
end for
Set $u_f^n := u_f^{n,(j)}$, $p_f^n := p_f^{n,(j)}$, $d_f^n := d_f^{n,(j)}$, $d_s^n := d_s^{n,(j)}$, $g^n := g^{(j)}$
return u_f^n, p_f^n, d_f^n, d_s^n, g^n

5.3.6 The Gauss–Newton optimisation algorithm

The nonlinear least squares problem described in Section 5.3.4 can be solved using the Gauss–Newton Algorithm 5.2.

In Algorithm 5.3, we show the conjugate gradient algorithm used in the Gauss–Newton optimisation process. More details can be found in [77, 73, 67].

Now that we have described the whole setting on the continuous level, we can move onto a spatial discretisation and a reduced–order model setting. This will be the topic of the following sections.

5.4 Finite Element discretisation

In this section, we present the Finite Element spatial discretisation for the previously introduced optimal control problem. We assume to have at hand two well–defined triangulations $\mathcal{T}_f$ and $\mathcal{T}_s$ over the domains Ω^f and Ω^s, respectively, and an extra lower–dimensional triangulation $\mathcal{T}_I$ of the interface Γ_I. In theory, there is no requirement for the meshes $\mathcal{T}_f$ and $\mathcal{T}_s$ to be conforming on the interface Γ_I, but in the numerical examples listed later in the chapter, this limitation was imposed by the software. We can then define usual Lagrangian FE spaces $V_{f,h} \subset V_f$, $Q_{f,h} \subset Q_i$,$E_{f,h} \subset E_f$, $E_{f,0,h} \subset E_{f,0}$ and $X_h \subset L^2(\Gamma_I)$ endowed with $L^2(\Gamma_I)$–norm. The spaces $V_{f,h}$, $E_{f,h}$ and $E_{f,0,h}$ are endowed with $H_0^1(\Omega^f)$–norm, the space Q_f with $L^2(\Omega^f)$–norm and the space $E_{s,h}$ with $H_0^1(\Omega^s)$–norm.

As it has been highlighted in Chapter 4, the discretisation of the fluid subsystem leads to a saddle–

Algorithm 5.3 Conjugate gradient (CG) algorithm for the least squares problem

Input: A, A^*, b, $h^{(0)}$, $\text{it}_{\max}$, ε_{tol}
$r^{(0)} := b - A[h^{(0)}]$
$p^{(0)} := A^*[r^{(0)}]$
for $j = 0, \dots, \text{it}_{\max}$ **do**
  if $||A^*[r^{(j)}]||_{L^2(\Gamma_I)} < \varepsilon_{tol}$ **then**
    return $h^{(j)}$
  end if
  Compute

$$\sigma^{(j)} := \frac{||A^*[r^{(j)}]||^2_{L^2(\Gamma_I)}}{||A[p^{(j)}]||^2_{L^2(\Gamma_I)\times L^2(\Gamma_I)}}$$

  Set $h^{(j+1)} := h^{(j)} + \sigma^{(j)} p^{(j)}$
  Set $r^{(j+1)} := r^{(j)} - \sigma^{(j)} A[p^{(j)}]$
  Compute

$$\tau^{(j)} := \frac{||A^*[r^{(j+1)}]||^2_{L^2(\Gamma_I)}}{||A^*[r^{(j)}]||^2_{L^2(\Gamma_I)}}$$

  Set $p^{(j+1)} := A^*[r^{(j+1)}] + \tau^{(j)} p^{(j)}$
end for
return $h^{(j)}$

point problem. In order to guarantee the well–posedness of the discretised problem, we require the FE spaces $V_{f,h}$ and $Q_{f,h}$ to satisfy the following inf–sup condition in the ALE–coordinates, see, for example, [153]: there exists $c \in \mathbb{R}^+$ s.t.

$$\inf_{q_{f,h}\in Q_{f,h}\setminus\{0\}} \sup_{v_{f,h}\in V_{f,h}\setminus\{0\}} \frac{b^B_f(v_{f,h}, q_{f,h}; d_{f,h})}{||J^{\frac{1}{2}} v_{f,h} F^{-T}||_{V_{f,h}} ||J^{\frac{1}{2}} q_{f,h}||_{Q_{f,h}}} \geq c, \tag{5.74}$$

for every $d_{f,h} \in E_{f,h}$. A very common choice in this framework is to use the so–called generalised Taylor–Hood finite element spaces, namely the Lagrange polynomial approximation of the second–order for the fluid velocity and displacement, 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_I.

5.4.1 Discrete optimal control problem for domain decomposition formulation

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

$$J_{\gamma,h}(u^n_{f,h}, d^n_{s,h}; g_h) = \frac{1}{2}\int_{\Gamma_I} \left| u^n_{f,h} - \frac{3d^n_{s,h} - 4d^{n-1}_{s,h} + d^{n-2}_{s,h}}{2\Delta t} \right|^2 d\Gamma + \frac{\gamma}{2}\int_{\Gamma_I} |g_h|^2 \, d\Gamma \tag{5.75}$$

under the constraints that $u^n_{f,h} \in V_{f,h}$, $p^n_{f,h} \in Q_{f,h}$, $d^n_{f,h}$, $d^n_{s,h} \in E_{s,h}$ satisfy the following variational equations subject to the following state equations for $n \geq 1$:

- FEM fluid momentum equation

$$\begin{aligned} \frac{1}{\Delta t} m_f(u^n_{f,h}, v_{f,h}; d^n_{f,h}) &+ a_f(u^n_{f,h}, v_{f,h}; d^n_{f,h}) + b^A_f(p^n_{f,h}, v_{f,h}; d^n_{f,h}) \\ &+ c^{ALE}_f\left(\frac{d^n_{f,h} - d^{n-1}_{f,h}}{\Delta t}, v_{f,h}, u^n_{f,h}; d^n_{f,h}\right) \\ &+ c_f(u^n_{f,h}, u^n_{f,h}, v_{f,h}; d^n_{f,h}) \\ &= \frac{1}{\Delta t} m_f(u^{n-1}_{f,h}, v_{f,h}; d^n_{f,h}) + f_f(v_{f,h}; d^n_{f,h}) \\ &+ \left(u^f_N, v_{f,h}\right)_{\Gamma^f_N} + (g_h, v_{f,h})_{\Gamma_I} \quad \forall v_{f,h} \in V_{f,h}, \end{aligned} \tag{5.76}$$

- FEM incompressibility equation

$$b^B_f(u^n_{f,h}, q_{f,h}; d^n_{f,h}) = 0 \quad \forall q_{f,h} \in Q_{f,h}, \tag{5.77}$$

- FEM extension equation

$$a^e_f(d^n_{f,h}, e_{f,h}) = 0 \quad \forall e_{f,h} \in E_{f,0,h}, \tag{5.78a}$$

$$d^n_{f,h} = d^{*,n}_{s,h} \quad \text{on } \Gamma_I, \tag{5.78b}$$

- FEM structure equation

$$\begin{aligned} \frac{1}{\Delta t^2} m_s(d^n_{s,h}, e_{s,h}) &+ a_s(d^n_{s,h}, e_{s,h}) = \frac{2}{\Delta t^2} m_s(d^{n-1}_{s,h}, e_{s,h}) + f_s(e_{s,h}) \\ &+ \left(d^s_N, e_{s,h}\right)_{\Gamma^s_N} - \frac{1}{\Delta t^2} m_s(d^{n-2}_{s,h}, e_{s,h}) - (g_h, e_{s,h})_{\Gamma_I} \quad \forall e_{s,h} \in E_{s,h}, \end{aligned} \tag{5.79}$$

and

$$u^0_{f,h} = u_{0,h}, \quad d^{-1}_{s,h} = d^s_{0,h} - \Delta t d^{s,\partial t}_{0,h}, \quad d^0_{s,h} = d^s_{h,0}, \tag{5.80}$$

where $u^n_{0,h}$ is the Galerkin projection of u_0 onto $V_{f,h}$, $d^{*,n}_{s,h}$ is the Galerkin projection of $d^{*,n}_s$ onto $E_{s,h}$, $d^s_{0,h}$ and $d^{s,\partial t}_{0,h}$ are the Galerkin projection of d^s_0 and $d^{s,\partial t}_0$ onto $E_{s,h}$, respectively. Notice that the structure of the equations (5.76) – (5.79) and of the functional (5.75) 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 (5.75):

$$\frac{d\mathcal{J}_{\gamma,h}}{dg_h}(u^n_{f,h}, d^n_{s,h}; g_h) = \gamma g_h + \xi_{f,h}|_{\Gamma_I} - \xi_{s,h}|_{\Gamma_I}, \tag{5.81}$$

where $\xi_{f,h}$ and $\xi_{s,h}$ are the solutions to the discretised adjoint problem: find $\xi_{f,h} \in V_{f,h}$, $\eta_{f,h} \in Q_{f,h}$ and $\xi_{s,h} \in E_{s,h}$ that satisfy

- FEM adjoint fluid momentum equation

$$\begin{aligned}\frac{1}{\Delta t} m_f(\delta v_{f,h}, \xi_{f,h}; d^n_{f,h}) \quad &+ \quad a_f(\delta v_{f,h}, \xi_{f,h}; d^n_{f,h}) + c_f(\delta v_{f,h}, u^n_{f,h}, \xi_{f,h}; d^n_{f,h}) \\ &+ \quad c_f(u^n_{f,h}, \delta v_{f,h}, \xi_{f,h}; d^n_{f,h}) + b^B_f(\delta v_{f,h}, \eta_{f,h}; d^n_{f,h}) \\ &+ \quad c^{ALE}_f\left(\frac{d^n_{f,h} - d^{n-1}_{f,h}}{\Delta t}, \xi_{f,h}, \delta v_{f,h}; d^n_{f,h}\right) \\ &= \quad \left(u^n_{f,h} - \frac{3d^n_{s,h} - 4d^{n-1}_{s,h} + d^{n-2}_{s,h}}{2\Delta t}, \delta v_{f,h}\right)_{\Gamma_I} \qquad \forall \delta v_{f,h} \in V_{f,h},\end{aligned} \tag{5.82}$$

- FEM adjoint incompressibility equation

$$b^A_f(\xi_{f,h}, \delta q_{f,h}; d^n_{f,h}) = 0 \quad \forall \delta q_{f,h} \in Q_f, \tag{5.83}$$

- FEM adjoint structure equation

$$\begin{aligned}&\frac{1}{\Delta t^2} m_s(\delta e_{s,h}, \xi_{s,h}) + a_s(\delta e_{s,h}, \xi_{s,h}) \\ &\qquad = \left(u^n_{f,h} - \frac{3d^n_{s,h} - 4d^{n-1}_{s,h} + d^{n-2}_{s,h}}{2\Delta t}, -\frac{3}{2\Delta t}\delta e_{s,h}\right)_{\Gamma_I} \qquad \forall \delta e_{s,h} \in E_{s,h}.\end{aligned} \tag{5.84}$$

In (5.81), the restriction $\cdot|_{\Gamma_I}$ is meant as an $L^2(\Gamma_I)$–projection onto space X_h. The above formulae allow us to write down the gradient–based algorithm similar to Algorithm 5.1. 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_I.

5.4.2 Non–linear least squares setting

We use non–linear least squares to develop a computational algorithm for the constrained optimal control problem. We introduce a notation that will be used throughout the section: the restriction $\cdot|_{\Gamma_I}$ is meant as an $L^2(\Gamma_I)$–projection onto space X_h. We define the non–linear operator $N_{n,h} : X_h \to X_h \times X_h$ by

$$N_{n,h}(g_h) = \begin{pmatrix} u^n_{f,h}\Big|_{\Gamma_I} - \frac{3d^n_{s,h} - 4d^{n-1}_{s,h} + d^{n-2}_{s,h}}{2\Delta t}\Big|_{\Gamma_I} \\ \sqrt{\gamma} g_h \end{pmatrix}, \tag{5.85}$$

where $u^n_{f,h}$ and $d^n_{s,h}$ are solutions to the state equations (5.76)– (5.79) with $g_h \in X_h$ as a disctretised stress function on the interface Γ_I. Then, the functional $\mathcal{J}_{\gamma,h}$ can be rewritten as

$$J_{\gamma,h}(u^n_{f,h}, d^n_{s,h}; g_h) = \frac{1}{2}||N_{n,h}(g_h)||^2_{X_h \times X_h}. \tag{5.86}$$

The non–linear least square optimisation problem is the following: minimise the functional (5.86) over $g_h \in X_h$ subject to (5.76)– (5.79). This problem can be tackled in a similar fashion as on the continuous level, namely using the Gauss–Newton Algorithm 5.2. Considering that the discrete non–linear least squares optimisation problem described above has the same structure as the continuous one, it is straightforward to derive the Fréchet derivative and its conjugate for the functional $N_{n,h}$.

For $\bar{g}_h \in X_h$, the Fréchet derivative $N'_{n,h}(\bar{g}_h)[\cdot] : X_h \to X_h \times X_h$ is defined by

$$N'_{n,h}(\bar{g}_h)[l_h] = \begin{pmatrix} w_{f,h}\big|_{\Gamma_I} - \frac{3}{2\Delta t}\, \phi_{s,h}\big|_{\Gamma_I} \\ \sqrt{\gamma} l_h \end{pmatrix}, \tag{5.87}$$

where $w_{f,h}$ and $\phi_{s,h}$ are the solutions to

$$\begin{aligned} \frac{1}{\Delta t} m_f(w_{f,h}, v_{f,h}; d^n_{f,h}) &+ a_f(w_{f,h}, v_{f,h}; d^n_{f,h}) + c^{ALE}_f\left(\frac{d^n_{f,h} - d^{n-1}_{f,h}}{\Delta t}, v_{f,h}, w_{f,h}; d^n_{f,h}\right) \\ &+ b^A_f(\psi_{f,h}, v_{f,h}; d^n_{f,h}) + c_f(w_{f,h}, \bar{u}_{f,h}, v_{f,h}; d^n_{f,h}) \\ &+ c_f(\bar{u}_{f,h}, w_{f,h}, v_{f,h}; d^n_{f,h}) = (l_h, v_{f,h})_{\Gamma_I} \quad \forall v_{f,h} \in V_{f,h}, \end{aligned} \tag{5.88}$$

$$b^B_f(w_{f,h}, q_{f,h}; d^n_{f,h}) = 0 \quad \forall q_{f,h} \in Q_{f,h}, \tag{5.89}$$

and

$$\frac{1}{\Delta t^2} m_s(\phi_{s,h}, e_{s,h}) + a_s(\phi_{s,h}, e_{s,h}) = -(l_h, e_{s,h})_{\Gamma_I} \quad \forall e_{s,h} \in E_{s,h}, \tag{5.90}$$

where $\bar{u}_{s,h}$ is the solution to the fluid state equations (5.82)– (5.83) with g_h replaced by $\bar{g}_h$.

The adjoint of $N'_{n,h}(\bar{g}_h)[\cdot]$ is $\left(N'_{n,h}(\bar{g}_h)\right)^* [\cdot] : X_h \times X_h \to X_h$, given by

$$\left(N'_{n,h}(\bar{g}_h)\right)^* \begin{bmatrix} r_h \\ s_h \end{bmatrix} = \beta_{f,h}\big|_{L^2(\Gamma_I)} - \frac{3}{2\Delta t}\, \varphi_{s,h}\big|_{L^2(\Gamma_I)} + \sqrt{\gamma} s_h, \tag{5.91}$$

where $\beta_{f,h}$ and $\varphi_{s,h}$ are solutions of

$$\begin{aligned}\frac{1}{\Delta t}m_f(v_{f,h},\beta_{f,h};d^n_{f,h}) &+ a_f(v_{f,h},\beta_{f,h};d^n_{f,h}) + c_f^{ALE}\left(\frac{d^n_{f,h}-d^{n-1}_{f,h}}{\Delta t},\beta_{f,h},v_{f,h};d^n_{f,h}\right)\\ &+ b_f^B(v_{f,h},\alpha_{f,h};d^n_{f,h}) + c_f(v_{f,h},\bar{u}_{f,h},\beta_{f,h};d^n_{f,h}) \qquad (5.92)\\ &+ c_f(\bar{u}_{f,h},v_{f,h},\beta_{f,h};d^n_{f,h}) = (r_h,v_{f,h})_{\Gamma_I} \quad \forall v_{f,h}\in V_{f,h},\\ b_f^A(q_{f,h},\beta_{f,h};d^n_{f,h}) &= 0 \quad \forall q_{f,h}\in Q_{f,h}, \qquad (5.93)\end{aligned}$$

and

$$\frac{1}{\Delta t^2}m_s(e_{s,h},\varphi_{s,h}) + a_s(e_{s,h},\varphi_{s,h}) = -(r_h,e_{s,h})_{\Gamma_I} \quad \forall e_{s,h}\in E_{s,h}. \qquad (5.94)$$

Now, having at hand the high–fidelity FEM model, we are able to derive a reduced–order model: this will be the topic of the following section.

5.5 Reduced–Order Model

As highlighted in Section 2, reduced–order methods are efficient tools for significant reduction of the computational time for parameter–dependent PDEs. We will now deal with the ROM for the problem obtained in the previous section, where the state FSI problem is assumed to be dependent on a set of physical parameters. We start with describing the reduced basis generation based on the Proper Orthogonal Decomposition (POD) technique, which is followed by an online phase based on a Galerkin projection onto the reduced spaces.

5.5.1 Reduced Basis Generation

Firstly, as it has been highlighted in Section 5.4, the discretised fluid subsystem has the saddle–point structure. We therefore mention again the need for a supremiser, as it is described in Section 3.4.1, to stabilise the reduced–order problem. In the case of the fluid dynamics problem with a moving domain, having at our disposal an ALE map, we are able to perform a pullback of the equation (3.21) onto the reference domain Ω^f and obtain the following problem for a supremiser variable: find $s_p \in V_f$ satisfying

$$\left(J\nabla s^n_{p,h}F^{-1}F^{-T},\nabla v_{f,h}\right)_{\Omega^f} = b_f^B\left(v_{f,h},p^n_{f,h};d^n_{f,h}\right) \quad \forall v_{f,h}\in V_{f,h}. \qquad (5.95)$$

Once we have at our disposal the snapshots for variables $u^n_{f,h}$, $p^n_{f,h}$, $d^n_{f,h}$, $d^n_{s,h}$, g^n_h and the supremisers $s^n_{p,h}$, we are ready to construct a set of reduced basis functions. Again our choice is to use the POD as it is described in Section 2.3. 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 (5.75) and (5.76) – (5.79): $V_{f,h} = \text{span}\{\phi_j^{u_f}\}_{j=1}^{\mathcal{N}_h^{u_f}}$, $Q_{f,h} = \text{span}\{\phi_j^{p_f}\}_{j=1}^{\mathcal{N}_h^{p_f}}$, $E_{s,h} = \text{span}\{\phi_j^{d_s}\}_{j=1}^{\mathcal{N}_h^{d_s}}$, and $X_h = \text{span}\{\phi_j^{g}\}_{j=1}^{\mathcal{N}_h^{g}}$, where $\mathcal{N}_h^*$, for $* \in \{u_f, p_f, d_s, g\}$, denotes the dimension of the corresponding FE space. Notice that we do not require a basis for a fluid displacement variable d_f^n: we will deal with it in a simplified manner that will be described later in the section.

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 $\mathcal{S}_{u_f} \in \mathbb{R}^{\mathcal{N}_h^{u_f} \times MK}$, $\mathcal{S}_{s_p} \in \mathbb{R}^{\mathcal{N}_h^{u_f} \times MK}$, $\mathcal{S}_{p_f} \in \mathbb{R}^{\mathcal{N}_h^{p_f} \times MK}$, $\mathcal{S}_{d_s} \in \mathbb{R}^{\mathcal{N}_h^{d_s} \times MK}$ and $\mathcal{S}_g \in \mathbb{R}^{\mathcal{N}_h^{g} \times MK}$.

The next step is to define the inner–product matrices X_{u_f}, X_{s_p}, X_{p_f}, X_{d_s} and X_g:

$$
\begin{aligned}
(X_{s_p})_{jk} = (X_{u_f})_{jk} &= \left(\nabla\phi_k^{u_f}, \nabla\phi_j^{u_f}\right)_{\Omega^f}, && \text{for } j,k = 1, ..., \mathcal{N}_h^{u_f},\\
(X_{p_f})_{jk} &= \left(\phi_k^{p_f}, \phi_j^{p_f}\right)_{\Omega^f}, && \text{for } j,k = 1, ..., \mathcal{N}_h^{p_f},\\
(X_{d_s})_{jk} &= \left(\nabla\phi_k^{d_s}, \nabla\phi_j^{d_s}\right)_{\Omega^s}, && \text{for } j,k = 1, ..., \mathcal{N}_h^{d_s},\\
(X_g)_{jk} &= \left(\phi_k^{g}, \phi_j^{g}\right)_{\Gamma_I}, && \text{for } j,k = 1, ..., \mathcal{N}_h^{g}.
\end{aligned}
$$

We are now ready to introduce the correlation matrices C_{u_f}, C_{s_p}, C_{p_f}, C_{d_s} and C_g, all of dimension $MK \times MK$, as

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

for every $* \in \{u_f, s_p, p_f, d_s, g\}$. 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:

$$C_* Q_* = Q_* \Lambda_* \tag{5.96}$$

where $* \in \{u_f, s_p, p_f, d_s, g\}$, 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 (5.96). Therefore, we are able to form the set of reduced bases 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 $* \in \{u_f, s_p, p_f, d_s, g\}$. 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 the velocities for simplicity:

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

and we redefine $N_{u_f} := N_{u_f} + N_{s_p}$. Hence, the new basis function sets are

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

including the extra basis functions obtained from the corresponding supremiser. Finally, we introduce separate reduced basis spaces – for the state and the control variables, respectively:

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

for $* \in \{u_f, p_f, d_s, g\}$. Now, due to the supremiser enrichment the spaces $V^{u_f}_N$ and $V^{p_f}_N$ are inf–sup stable in the sense (5.74); the proof can be found in [18].

Harmonic extension of the fluid displacement

In order to generate a reduced basis for the fluid displacement, we resort to the idea described in [19]. Having at hand a reduced basis for the structure displacement $\mathcal{A}^{d_s}$, we can build reduced basis functions for the variable d_f by performing a harmonic extension of each basis function $\Phi^{d_s}_i, i = 1, ..., N_{d_s}$ to the fluid domain. The i–th reduced basis function $\Phi^{d_f}_i$ is therefore defined by

$$-\Delta\Phi^{d_f}_i = 0 \quad \text{in } \Omega^f, \tag{5.97a}$$

$$\Phi^{d_f}_i = \Phi^{d_s}_i \quad \text{on } \Gamma_I, \tag{5.97b}$$

for $i = 1, ..., N_{d_s}$. We can then define the fluid displacement reduced space $V^{d_f}_N$ as

$$V^{d_f}_N := \text{span} \left\{ \Phi^{d_f}_i \right\}^{N_{d_s}}_{i=1}.$$

The reasons for defining the basis functions for d_f in such a way are two: the POD compression is quite expensive and, moreover, since we impose the non–homogeneous Dirichlet condition in the equations (5.78) for $d^n_{f,h}$, we would need to introduce the Lagrange multiplier or a lifting function as it is done in Section 3.4.

Then, during the online phase, the reduced fluid displacement will be computed just as a linear combination of the basis $\Phi^{d_f}_i$, with coefficients that are the coefficients of the reduced solid displacement. We will see in the next section the final formulation of the online phase of the algorithm.

5.5.2 Online phase

Following the description of two techniques for dealing with the optimal control problem at the FOM level in Section 5.4, we will provide a reduced–order online phase procedure for each of them.

Online phase for the gradient–based optimisation algorithms

Once we have introduced the reduced basis spaces, we can define the reduced function expansions

$$(u^n_{f,N}, p^n_{f,N}, d^n_{s,N}, g_N) \in V_N^{u_f} \times V_N^{p_f} \times V_N^{d_s} \times V_N^{g},$$

as

$$u^n_{f,N} := \sum_{k=1}^{N_{u_f}} \underline{u}^N_{f,k} \Phi^{u_f}_k, \quad p_{f,N} := \sum_{k=1}^{N_{p_f}} \underline{p}^N_{f,k} \Phi^{p_f}_k,$$
$$d^n_{s,N} := \sum_{k=1}^{N_{d_s}} \underline{d}^N_{s,k} \Phi^{d_s}_k, \quad g_N := \sum_{k=1}^{N_g} \underline{g}_k \Phi^{g}_k.$$

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

$$J_{\gamma,N}(u^n_{f,N}, d^n_{s,N}; g_N) = \frac{1}{2} \int_{\Gamma_I} \left| u^n_{f,N} - \frac{3d^n_{s,N} - 4d^{n-1}_{s,N} + d^{n-2}_{s,N}}{2\Delta t} \right|^2 d\Gamma + \frac{\gamma}{2} \int_{\Gamma_I} |g_N|^2 \, d\Gamma, \tag{5.98}$$

where $(u^n_{f,N}, p_{f,N}, d_{s,N})$ satisfy the following reduced state equations: for $n \geq 1$:

- ROM fluid momentum equation

$$\begin{aligned} \frac{1}{\Delta t} m_f(u^n_{f,N}, v_{f,N}; d^n_{f,N}) \quad &+ \quad a_f(u^n_{f,N}, v_{f,N}; d^n_{f,N}) + b^A_f(p^n_{f,N}, v_{f,N}; d^n_{f,N}) \\ &+ \quad c^{ALE}_f \left(\frac{d^n_{f,N} - d^{n-1}_{f,N}}{\Delta t}, v_{f,N}, u^n_{f,N}; d^n_{f,N} \right) \\ &+ \quad c_f(u^n_{f,N}, u^n_{f,N}, v_{f,N}; d^n_{f,N}) \\ &= \quad \frac{1}{\Delta t} m_f(u^{n-1}_{f,N}, v_{f,N}; d^n_{f,N}) + f_f(v_{f,N}; d^n_{f,N}) \\ &+ \quad \left(u^f_N, v_{f,N} \right)_{\Gamma^f_N} + (g_N, v_{f,N})_{\Gamma_I} \quad \forall v_{f,N} \in V_{f,N}, \end{aligned} \tag{5.99}$$

- ROM incompressibility equation

$$b^B_f(u^n_{f,N}, q_{f,N}; d^n_{f,N}) = 0 \quad \forall q_{f,N} \in Q_{f,N}, \tag{5.100}$$

- ROM structure equation

$$\frac{1}{\Delta t^2} m_s(d^n_{s,N}, e_{s,N}) + a_s(d^n_{s,N}, e_{s,N}) = \frac{2}{\Delta t^2} m_s(d^{n-1}_{s,N}, e_{s,N}) + f_s(e_{s,N}) \\ + \left(d^s_N, e_{s,N}\right)_{\Gamma^s_N} - \frac{1}{\Delta t^2} m_s(d^{n-2}_{s,N}, e_{s,N}) - (g_N, e_{s,N})_{\Gamma_I} \quad \forall e_{s,N} \in E_{s,N}, \tag{5.101}$$

with

$$d^n_{f,N} := \sum_{k=1}^{N_{d_s}} \underline{d}^{n,*}_{s,k} \Phi^{d_f}_k, \tag{5.102}$$

where $\underline{d}^{n,*}_{s,k}$ are some extrapolation of reduced coefficients from previous time steps $\underline{d}^{n-1}_{s,k}, \underline{d}^{n-2}_{s,k}, \ldots$ and

$$u^0_{f,N} = u_{0,N}, \quad d^{-1}_{s,N} = d^s_{0,N} - \Delta t d^{s,\partial t}_{0,N}, \quad d^0_{s,N} = d^s_{N,0}. \tag{5.103}$$

where $u^n_{0,N}$ is the Galerkin projection of u_0 onto $V^{u_f}_N$, $d^s_{0,N}$ and $d^{s,\partial t}_{0,N}$ are the Galerkin projection of d^s_0 and $d^{s,\partial t}_0$ onto $V^{d_s}_N$, respectively.

Similarly to the offline phase, we notice that the structure of the equations (5.99)– (5.101) and the functional (5.98) 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 (5.98):

$$\frac{d\mathcal{J}_{\gamma,N}}{dg_N}(u^n_{f,N}, d^n_{s,N}; g_N) = \gamma g_N + \xi_{f,N}|_{\Gamma_I} - \xi_{s,N}|_{\Gamma_I}, \tag{5.104}$$

where $\xi_{f,N}$ and $\xi_{s,N}$ are the solutions to the discretised adjoint problem: find $\xi_{f,N} \in V^{u_f}_N$, $\eta_{f,N} \in V^{p_f}_N$ and $\xi_{s,N} \in V^{d_s}_N$ that satisfy

- ROM adjoint fluid momentum equation

$$\begin{aligned} \frac{1}{\Delta t} m_f(\delta v_{f,N}, \xi_{f,N}; d^n_{f,N}) \quad &+ \quad a_f(\delta v_{f,N}, \xi_{f,N}; d^n_{f,N}) + c_f(\delta v_{f,N}, u^n_{f,N}, \xi_{f,N}; d^n_{f,N}) \\ &+ \quad c_f(u^n_{f,N}, \delta v_{f,N}, \xi_{f,N}; d^n_{f,N}) + b^B_f(\delta v_{f,N}, \eta_{f,N}; d^n_{f,N}) \\ &+ \quad c^{ALE}_f\left(\frac{d^n_{f,N} - d^{n-1}_{f,N}}{\Delta t}, \xi_{f,N}, \delta v_{f,N}; d^n_{f,N}\right) \\ &= \quad \left(u^n_{f,N} - \frac{3d^n_{s,N} - 4d^{n-1}_{s,N} + d^{n-2}_{s,N}}{2\Delta t}, \delta v_{f,N}\right)_{\Gamma_I} \quad \forall \delta v_{f,N} \in V^{u_f}_N, \end{aligned} \tag{5.105}$$

- ROM adjoint incompressibility equation

$$b^A_f(\xi_{f,N}, \delta q_{f,N}; d^n_{f,N}) = 0 \quad \forall \delta q_{f,N} \in V^{p_f}_N, \tag{5.106}$$

- ROM adjoint structure equation

$$\frac{1}{\Delta t^2} m_s(\delta e_{s,N}, \xi_{s,N}) + a_s(\delta e_{s,N}, \xi_{s,N}) \tag{5.107}$$
$$= \left(u^n_{f,N} - \frac{3d^n_{s,N} - 4d^{n-1}_{s,N} + d^{n-2}_{s,N}}{2\Delta t}, -\frac{3}{2\Delta t}\delta e_{s,N} \right)_{\Gamma_I} \quad \forall \delta e_{s,N} \in V^{d_s}_N.$$

Above, the restriction $\cdot|_{\Gamma_I}$ is meant as an $L^2(\Gamma_I)$–projection onto space V^g_N. The above formulae allow us to write down the gradient–based algorithm similar to Algorithm 5.1. We would also like to stress that from the numerical implementation point of view 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 $p = N_g$.

Online phase in the non–linear least squares optimisation setting

We use non–linear least squares to develop a computational algorithm for the constrained optimal control problem. We introduce a notation that will be used throughout the section: the restriction $\cdot|_{\Gamma_I}$ is meant as an $L^2(\Gamma_I)$–projection onto space V^g_N. We define the non–linear operator $N_{n,N} : V^g_N \to V^g_N \times V^g_N$ by

$$N_{n,N}(g_N) = \begin{pmatrix} u^n_{f,N}\Big|_{\Gamma_I} - \frac{3d^n_{s,N} - 4d^{n-1}_{s,N} + d^{n-2}_{s,N}}{2\Delta t}\Big|_{\Gamma_I} \\ \sqrt{\gamma} g_N \end{pmatrix}, \tag{5.108}$$

where $u^n_{f,N}$ and $d^n_{s,N}$ are solutions to the state equations (5.99)– (5.101) with $g_N \in V^g_N$ as a disctretised stress function on the interface Γ_I. Then the functional $\mathcal{J}_{\gamma,N}$ can be rewritten as

$$J_{\gamma,N}(u^n_{f,N}, d^n_{s,N}; g_N) = \frac{1}{2} ||N_{n,N}(g_N)||^2_{V^g_N \times V^g_N}. \tag{5.109}$$

The non–linear least square optimisation problem is the following: minimise the functional (5.109) over $g_N \in V^g_N$ subject to (5.99)– (5.101). This problem can be tackled in a similar fashion as on the continuous level, namely by the use of Gauss–Newton Algortihm 5.2. Considering that the reduced non–linear least squares optimisation problem described above has the same structure as the continuous one, it is straightforward to derive the Fréchet derivative and its conjugate for the functional $N_{n,N}$.

For $\bar{g}_N \in V^g_N$, the Fréchet derivative $N'_{n,N}(\bar{g}_N)[\cdot] : V^g_N \to V^g_N \times V^g_N$ is defined by

$$N'_{n,N}(\bar{g}_N)[h_N] = \begin{pmatrix} w_{f,N}\big|_{\Gamma_I} - \frac{3}{2\Delta t}\, \phi_{s,N}\big|_{\Gamma_I} \\ \sqrt{\gamma} h_N \end{pmatrix}, \tag{5.110}$$

where $w_{f,N}$ and $\phi_{s,N}$ are the solutions to

$$\begin{aligned}\frac{1}{\Delta t}m_f(w_{f,N},v_{f,N};d^n_{f,N}) &+ a_f(w_{f,N},v_{f,N};d^n_{f,N}) + c_f^{ALE}\left(\frac{d^n_{f,N}-d^{n-1}_{f,N}}{\Delta t},v_{f,N},w_{f,N};d^n_{f,N}\right)\\ &+ b_f^A(\psi_{f,N},v_{f,N};d^n_{f,N}) + c_f(w_{f,N},\bar{u}_{f,N},v_{f,N};d^n_{f,N}) \qquad (5.111)\\ &+ c_f(\bar{u}_{f,N},w_{f,N},v_{f,N};d^n_{f,N}) = (h_N,v_{f,N})_{\Gamma_I} \quad \forall v_{f,N}\in V_N^{u_f},\\ b_f^B(w_{f,N},q_{f,N};d^n_{f,N}) &= 0 \quad \forall q_{f,N}\in V_N^{p_f}, \qquad (5.112)\end{aligned}$$

and

$$\frac{1}{\Delta t^2}m_s(\phi_{s,N},e_{s,N}) + a_s(\phi_{s,N},e_{s,N}) = -(h_N,e_{s,N})_{\Gamma_I} \quad \forall e_{s,N}\in V_N^{d_s}, \qquad (5.113)$$

where $\bar{u}_{s,N}$ is the solution to the fluid state equations (5.105)– (5.106) with g_N replaced by $\bar{g}_N$.

The adjoint of $N'_{n,N}(\bar{g}_N)[\cdot]$ is $\left(N'_{n,N}(\bar{g}_N)\right)^*[\cdot] : V_N^g \times V_N^g \to V_N^g$, given by

$$\left(N'_{n,N}(\bar{g}_N)\right)^*\begin{bmatrix}r_N\\ s_N\end{bmatrix} = \beta_{f,N}\big|_{L^2(\Gamma_I)} - \frac{3}{2\Delta t}\,\varphi_{s,N}\big|_{L^2(\Gamma_I)} + \sqrt{\gamma}s_N, \qquad (5.114)$$

where $\beta_{f,N}$ and $\varphi_{s,N}$ are solutions of

$$\begin{aligned}\frac{1}{\Delta t}m_f(v_{f,N},\beta_{f,N};d^n_{f,N}) &+ a_f(v_{f,N},\beta_{f,N};d^n_{f,N}) + c_f^{ALE}\left(\frac{d^n_{f,N}-d^{n-1}_{f,N}}{\Delta t},\beta_{f,N},v_{f,N};d^n_{f,N}\right)\\ &+ b_f^B(v_{f,N},\alpha_{f,N};d^n_{f,N}) + c_f(v_{f,N},\bar{u}_{f,N},\beta_{f,N};d^n_{f,N}) \qquad (5.115)\\ &+ c_f(\bar{u}_{f,N},v_{f,N},\beta_{f,N};d^n_{f,N}) = (r_N,v_{f,N})_{\Gamma_I} \quad \forall v_{f,N}\in V_N^{u_f},\\ b_f^A(q_{f,N},\beta_{f,N};d^n_{f,N}) &= 0 \quad \forall q_{f,N}\in V_N^{p_f}, \qquad (5.116)\end{aligned}$$

and

$$\frac{1}{\Delta t^2}m_s(e_{s,N},\varphi_{s,N}) + a_s(e_{s,N},\varphi_{s,N}) = -(r_N,e_{s,N})_{\Gamma_I} \quad \forall e_{s,N}\in V_N^{d_s}. \qquad (5.117)$$

With all this, we are able to write down the ROM version of the Gauss–Newton algorithm in Algorithm 5.4.

At this point, we conclude the theoretical exposition and we present numerical experiments on a two–dimensional haemodynamic benchmark FSI problem in the following section.

5.6 Numerical results

We now present some numerical results obtained with the Gauss–Newton optimisation algorithm described in the previous section on a haemodynamics benchmark FSI problem. All the numerical

Algorithm 5.4 ROM Gauss–Newton (ROM–GN) algorithm

Input: $g_N^{(0)} := g_N^{n-1}$, It_{max} and ε_{tol}
for $j = 1, 2, ..., It_{max}$ **do**
Solve (5.99)– (5.100) for $u_{f,N}^{n,(j)} \in V_N^{u_f}$ and $p_{f,N}^{n,(j)} \in V_N^{p_f}$ with $g_N = g_N^{(j-1)}$
Solve (5.101) for $d_{s,N}^{n,(j)} \in V_N^{d_s}$ with $g_N = g_N^{(j-1)}$
Set $d_{f,N}^{n,(j)} := \sum_{k=1}^{N_{ds}} \underline{d}_{s,k}^{n,(j)} \Phi_k^{d_f}$, where $\underline{d}_{s,k}^{n,(j)}$ are the reduced coefficients of $d_{s,N}^{n,(j)}$
if $\frac{1}{2}\int_{\Gamma_I} \left| u_{f,N}^{n,(j)} - \frac{3d_{s,N}^{n,(j)} - 2d_{s,N}^{n-1} + d_{s,N}^{n-2}}{2\Delta t} \right|^2 d\Gamma < \varepsilon_{tol}$ **then**
`break`
end if
Compute $h_N^{(j)}$ by the Conjugate Gradient Algorithm 5.3 with $A = N'_{n,N}\left(g_N^{(j-1)}\right)$, $b = -N_{n,N}\left(g_N^{(j-1)}\right)$, $A^* = \left(N'_{n,N}\left(g^{(j-1)}\right)\right)^*$ and $h = h_N$.
Set $g_N^{(j)} := g_N^{(j-1)} + h_N^{(j)}$
end for
Set $u_{f,N}^n := u_{f,N}^{n,(j)}$, $p_{f,N}^n := p_{f,N}^{n,(j)}$, $d_{f,N}^n := d_{f,N}^{n,(j)}$, $d_{s,N}^n := d_{s,N}^{n,(j)}$ and $g_N^n := g_N^{(j)}$
return $u_{f,N}^n$, $p_{f,N}^n$, $d_{f,N}^n$, $d_{s,N}^n$, g_N^n

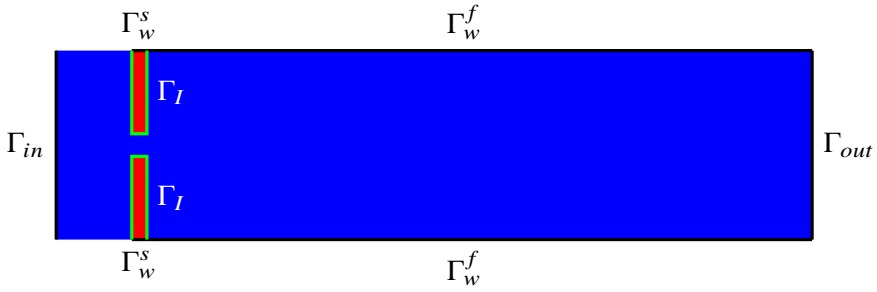


Figure 5.3: Physical reference domain. In blue: the reference fluid domain Ω^f. In red: the reference solid domain Ω^s. In green: the fluid–structure interface Γ_I.

simulations for the offline phase were obtained using the software FEniCS [1], whereas the online phase simulations were carried out using RBniCS [3].

The reference domains Ω^f and Ω^s are presented in Figure 5.3: the fluid domain is $2.5cm$ in height $10cm$ long; the leaflets are situated $1cm$ downstream the inlet boundary Γ_{in}, they are $0.2cm$ thick and $1.1cm$ in height. The values of physical parameters are listed in Table 5.1. We consider zero initial conditions for fluid velocity and the structure displacement, homogeneous Dirichlet boundary conditions on walls Γ_w^f for the fluid velocity and on Γ_w^s for the structure displacement, and homogeneous Neumann conditions on the outlet Γ_{out}, meaning that we assume free outflow on this portion of the boundary. A pressure impulse p_{in} is applied at the inlet boundary Γ_{in}:

$$p_{in}(t) = \bar{P}\left(1 - \cos\left(\frac{2\pi t}{0.2}\right)\right), \quad t \in [0, 0.01]$$

and the range of $\bar{P}$ is reported in Table 5.1. On average, the Gauss–Newton algorithm converges in 2–3 iterations and we present additional details in Tables 5.2– 5.3. In particular, in Table 5.2, we list the values for the functional $\mathcal{J}_{\gamma,h}$ and the corresponding number of nested CG iterations at each GN iteration, while Table 5.3 contains the absolute and relative errors with respect to the monolithic (entire–domain) solutions u_h^n, p_h^n, d_h^n, obtained by the reliable method described in [18] i.e.,

- Absolute error $u_{f,h}^n := ||u_h^n - u_{f,h}^n||_{L^2(\Omega^f)}$, Relative error $u_{f,h} := \frac{||u_{f,h}^n - u_h^n||_{L^2(\Omega^f)}}{||u_h^n||_{L^2(\Omega^f)}}$,
- Absolute error $p_{f,h}^n := ||p_h^n - p_{f,h}^n||_{L^2(\Omega^f)}$, Relative error $p_{f,h}^n := \frac{||p_{f,h}^n - p_h^n||_{L^2(\Omega^f)}}{||p_h^n||_{L^2(\Omega^f)}}$,
- Absolute error $d_{s,h}^n := ||d_h^n - d_{s,h}^n||_{L^2(\Omega^s)}$, Relative error $d_{s,h}^n := \frac{||d_{s,h}^n - d_h^n||_{L^2(\Omega^s)}}{||d_h^n||_{L^2(\Omega^s)}}$.

Three physical parameters are considered: the Lamè coefficients λ_s and μ_s for the structure and $\bar{P}$ – the intensity parameter of the pressure impulse p_{in}. Details of the offline stage and the finite–element discretisation are summarised in Table 5.1. High–fidelity solutions are obtained by carrying out the minimisation by the Gauss–Newton algorithm with the maximum number of iterations 5 for the algorithm itself and 30 for the conjugate gradient algorithm, and the tolerance in both cases has been chosen as a minimum between 10^{-6} and the first step approximation divided by 100 (the choice is due to the fact that at initial time steps the problem variables have small magnitude).

Figure 5.4 shows the fluid velocity and pressure at the final time $t = 0.01$ computed at the parameter value $(\bar{P}, \lambda_s, \mu_s) = (110, 80000, 40000)$ and Figure 5.5 shows the structure displacement for the same parameter computed at three different time instances $t = 0.005$, $t = 0.0075$ and $t = 0.01$.

Snapshots are sampled from a training set of K parameters uniformly sampled from the 3–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 5.6a shows the POD singular values for all the state and the control variables. It can be noticed that the rates of decay of the eigenvalues for the fluid velocity and structure displacement are slightly slower than for other variables, but overall we can

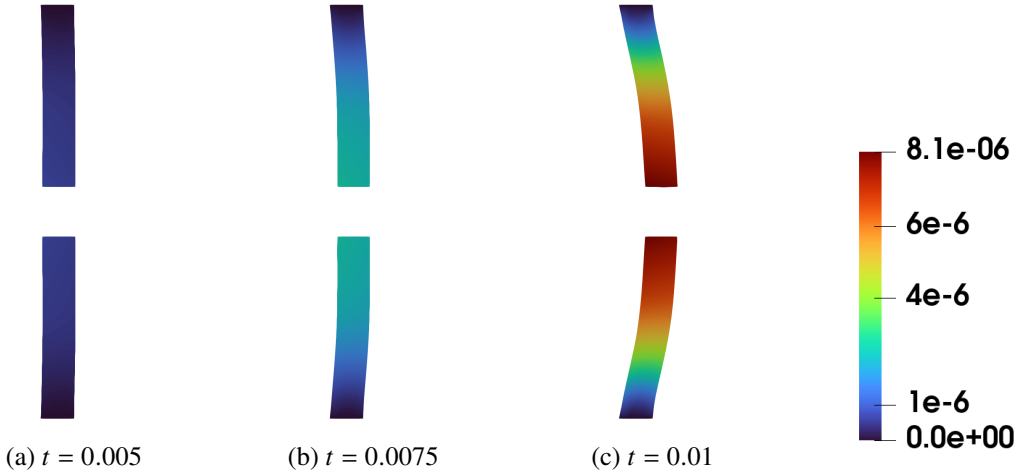
Physical parameters		FE parameters	
ν_f	0.035	Velocity–pressure space in a cell	$\mathbb{P}^2 \times \mathbb{P}^1$
ρ_f	1	Fluid displacement space in a cell	$\mathbb{P}^2$
ρ_s	1.1	Structure displacement space in a cell	$\mathbb{P}^2$
Range $\bar{P}$	[90, 120]	Total velocity–pressure dofs	66,171
Range λ_s	[800000, 1600000]	Fluid displacement dofs	58,720
Range μ_s	[400000, 1000000]	Structure displacement dofs	1,220
Final time T	0.01	Dofs at interface	292
		Time step Δt	10^{-4}
Optimization		**Snapshots training set parameters**	
Algorithm	Gauss–Newton(GN)	Timestep number M	100
$\text{it}_{\max}$ for CG alg.	30	Parameters training set size K	64
$\text{it}_{\max}$ for GN alg.	5	Maximum retained modes $N_{\max}$	100

Table 5.1: Computational details of the offline stage

(a) The fluid velocity at the final time

(b) The fluid pressure at the final time

Figure 5.4: The high–fidelity fluid velocity (a) and pressure (b) at the final time $t = 0.01$ (the mesh displacement has been magnified for visualisation purposes).



(a) $t = 0.005$ (b) $t = 0.0075$ (c) $t = 0.01$

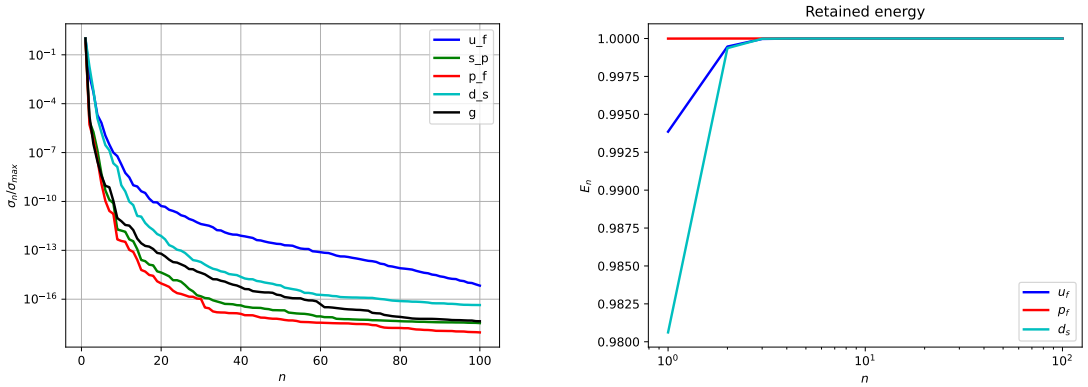
Figure 5.5: The high–fidelity structure displacement at 3 different times (the mesh displacement has been magnified for visualisation purposes).

GN Itearation	**Functional value $\mathcal{J}_{\gamma,h}$**	**CG iterations**
1	1.42×10^{-8}	30
2	1.27×10^{-10}	30
3	2.14×10^{-12}	27

Table 5.2: The value of the FOM functional and the corresponding number of nested CG iterations at each GN iteration at the final time step

GN Itearation	**Absolute error**			**Relative error**		
	$u^M_{f,h}$	$p^M_{f,h}$	$d^M_{s,h}$	$u^M_{f,h}$	$p^M_{f,h}$	$d^M_{s,h}$
1	2.14×10^{-5}	0.258	5.94×10^{-8}	0.002	0.017	0.01658
2	9.51×10^{-6}	0.003	5.93×10^{-8}	0.001	0.0002	0.01653
3	9.23×10^{-6}	0.005	5.93×10^{-8}	0.001	0.0004	0.01654

Table 5.3: L^2 absolute and relative errors between the high–fidelity solution and the monolithic solution at each GN iteration at the final time step



(a) POD singular values as a function of number n of POD modes (log scaling in y–direction)

(b) Energy retained by the first $N_{\max}$ POD modes (log scaling in x–direction)

Figure 5.6: Results of the offline stage: POD singular eigenvalue decay (a) and retained energy (b) of the first $N_{\max}$ POD modes.

Parameter		POD modes			
$\bar{P}$	110	velocity u_f	30	pressure p_f	20
λ_s	80000	displacement d_s	20	supremiser s_p	20
μ_s	40000	control g	20		

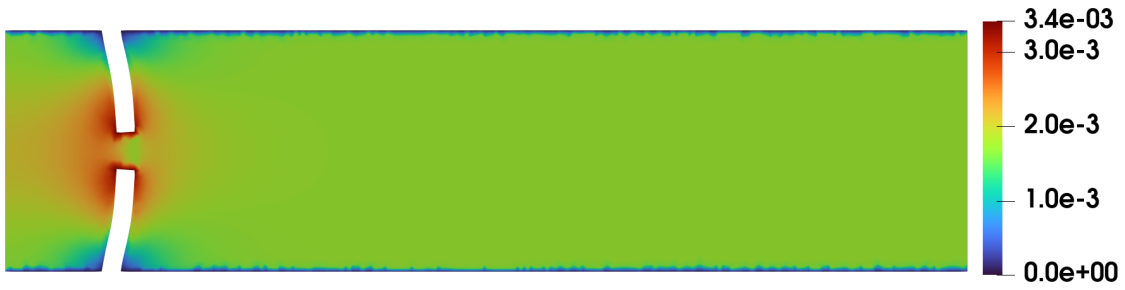
Table 5.4: Computational details of the online stage

see an evident exponential decay of the singular values for each component. In Figure 5.6b, we can see the behaviour of the energy E_n retained by the first N modes for the components u_f, p_f and d_s. Here, the retained energy for the component $* \in \{u_f, p_f, d_s\}$ is defined as

$$E_n^* := \frac{\sum_{k=1}^{n} |\lambda_k^*|}{\sum_{k=1}^{N_*} |\lambda_k^*|}.$$

The retained energy for the components s_p and g show the same behaviour as for the component p_f. We can notice that the first mode of the fluid pressure retains around 0.75% more energy with respect to the first mode of the fluid velocity and 2% more energy compared to the first mode of the structure displacement, which is the one that retains less energy. The retained energy gives us an idea of the number of modes we would need to choose to preserve all the necessary physical information in the reduced model.

In Table 5.4, 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 reduced–order solutions are obtained by carrying out the minimisation by the Gauss–Newton algorithm with the



(a) The fluid velocity at the final time

(b) The fluid pressure at the final time

Figure 5.7: The reduced–order fluid velocity (a) and pressure (b) at the final time $t = 0.01$ (the mesh displacement has been magnified for visualisation purposes).

(a) $t = 0.005$ (b) $t = 0.0075$ (c) $t = 0.01$

Figure 5.8: The reduced–order structure displacement at 3 different times (the mesh displacement has been magnified for visualisation purposes).

GN Itearation	Functional value $\mathcal{J}_{\gamma,N}$	CG iterations
1	1.42×10^{-8}	15
2	2.20×10^{-10}	15
3	1.94×10^{-10}	15
4	1.75×10^{-10}	14

Table 5.5: The value of the ROM functional and the corresponding number of nested CG iterations at each ROM-GN iteration at the final time step

GN Iteration	Absolute error			Relative error		
	$u_{f,N}$	$p_{f,N}$	$d_{s,N}$	$u_{f,N}$	$p_{f,N}$	$d_{s,N}$
1	4.9034e-05	0.3847	5.4018e-08	0.0056	0.0262	0.0151
2	4.5359e-05	0.1619	5.3899e-08	0.0052	0.0110	0.0150
3	4.4968e-05	0.1606	5.3918e-08	0.0051	0.0109	0.0150
4	4.4609e-05	0.1614	5.3902e-08	0.0051	0.0110	0.0150

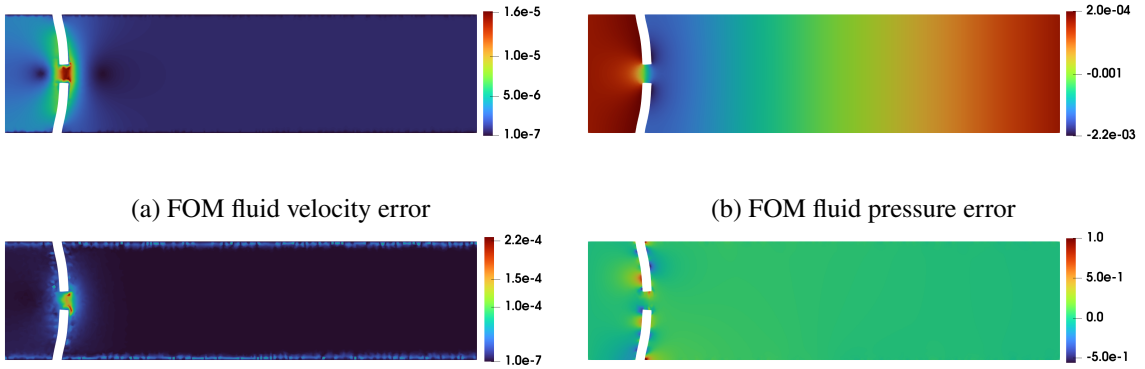
Table 5.6: L^2 absolute and relative errors between the reduced–order solution and the monolithic solution at each ROM-GN iteration at the final time step

maximum 5 iterations for the algorithm itself and 15 for the conjugate gradient algorithm, and the tolerance in both cases has been chosen the same as in the full–order model – a minimum between 10^{-6} and the first step approximation divided by 100 (the choice is again due to the fact that at initial time steps the problem variables have small magnitude). Figure 5.7 shows the reduced–order fluid velocity and pressure at the final time $t = 0.01$ computed at the parameter value $(\bar{P}, \lambda_s, \mu_s) = (110, 80000, 40000)$ and Figure 5.8 shows the reduced–order structure displacement for the same parameter computed at three different time instances $t = 0.005$, $t = 0.0075$ and $t = 0.01$.

On average, the reduced Gauss–Newton algorithm converges in 2–4 iterations and we present additional details in Tables 5.5– 5.6. In Table 5.5, we list the values for the functional $\mathcal{J}_{\gamma,N}$ and the corresponding number of nested CG iterations at each GN iteration, while Table 5.6 contains the absolute and relative errors with respect to the monolithic (entire–domain) solutions u_h^n, p_h^n, d_h^n, obtained using the method described in [18]. The errors we will consider are the following

- Absolute error $u_{f,N}^n := ||u_h^n - u_{f,N}^n||_{L^2(\Omega^f)}$, Relative error $u_{f,N} := \frac{||u_{f,N}^n - u_h^n||_{L^2(\Omega^f)}}{||u_h^n||_{L^2(\Omega^f)}}$,
- Absolute error $p_{f,N}^n := ||p_h^n - p_{f,N}^n||_{L^2(\Omega^f)}$, Relative error $p_{f,N}^n := \frac{||p_{f,N}^n - p_h^n||_{L^2(\Omega^f)}}{||p_h^n||_{L^2(\Omega^f)}}$,
- Absolute error $d_{s,N}^n := ||d_h^n - d_{s,N}^n||_{L^2(\Omega^s)}$, Relative error $d_{s,N}^n := \frac{||d_{s,N}^n - d_h^n||_{L^2(\Omega^s)}}{||d_h^n||_{L^2(\Omega^s)}}$.

Analysing the results, we are able to see that the reduced basis method gives us a solution as accurate as the high–fidelity one. In Figure 5.9, we can notice that the errors for the ROM



(a) FOM fluid velocity error

(b) FOM fluid pressure error

(c) ROM fluid velocity error

(d) ROM fluid pressure error

Figure 5.9: The error plots of FOM and ROM fluid solutions w.r.t the monolithic solutions at the final time $t = 0.01$ (the mesh displacement has been magnified for visualisation purposes).

solutions (especially, for the fluid pressure) are slightly bigger in the ROM case, but it is mostly concentrated on the fluid–structure interface; in the rest of the domain it is still quite small as evidenced in Table 5.6. This could be due to the fact that quite a small number of time steps have been considered and the fluid velocity and the structure displacement do not develop a significant magnitude. The reduced–order approximation of the optimisation problem at hand allowed us to reduce the dimensions of the high–fidelity optimisation functional by more than 10 times and enabled us to use 2 times fewer iterations in the CG while maintaining a very small number of GN iterations, which, in turn, requires a small number of nonlinear solves of the fluid subsystem. We also remark that the use of good hyper–reduction techniques (see, for instance, [81]), will allow us to obtain very efficient solvers in terms of computational time.

Regarding the computational time, one time step of the FOM takes between 7 and 9 minutes, whereas one time step of the ROM (without hyper–reduction) takes around 1–3 minutes.

Remark (Stability in the case of "added mass" effect). As it has been highlighted in the introduction partitioned approaches to FSI problems might, under some physical and geometrical conditions, be unstable: this happens, for example, if the physical domain has a slender shape, or, as in our numerical test, if the fluid density ρ_f is close to the solid density ρ_s and this is usually the case in haemodynamics applications, where the density of the blood is quite close to the density of the walls of the vessel. The reason for this instability is the so–called "added mass" effect: the fluid acts like an added mass to the solid, thus changing its natural behaviour; we refer to [38] for a detailed derivation of the "added mass" effect and relative consequences. Nevertheless, numerical tests presented in this section show that our partitioned algorithm is stable under the assumption of the "added mass" effect in both FOM and ROM settings.

5.7 Conclusions and perspectives

In this chapter, we proposed a reduced–order model for the optimisation–based domain decomposition formulation of the parameter–dependent non–stationary FSI problem with incompressible Navier–Stokes equations and linear elasticity model.

The original problem cast into the optimisation–based domain–decomposition framework leads to the optimal control problem aiming at minimising the velocity coupling error at the fluid–structure interface; the problem, then, has been tackled using either an iterative gradient–based optimisation algorithm or by the Gauss–Newton algorithm in the non–linear least squares optimisation framework. These allowed us to obtain a complete separation of the solvers on different subdomains.

On the reduced–order level, we have built a model for which the generation of the reduced basis spaces is carried out separately in each subdomain and for each component of the problem. 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 optimisation iterations.

Another extremely important feature of the algorithm presented here is the fact that they are stable under the assumption of the "added mass" effect, which is the cause of numerical instabilities in many partitioned approaches for FSI problems.

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 a large number of degrees of freedom and complex geometries. The future extensions of this methodology, just to name a few, may include combining this segregated approach with further domain–decomposition methods used for separate components of the multiphysics problem, compressible or non–Newtonian fluids and nonlinear elasticity models and combining the presented technique with optimal control problems.

Chapter 6

Conclusions and perspectives

In this final chapter, I provide a summary of the work presented throughout the thesis and suggest possible future perspectives.

6.1 Summary

The goal of this thesis has been to provide an extensive overview of optimisation–based domain decomposition methods and reduced order models, and their applications to fluid dynamics and multiphysics problems. In the course of the work, we have managed to develop reliable domain–decomposition ROM methods for parameter–dependent incompressible Navier–Stokes equations in both stationary and non–stationary regimes, and for Fluid–Structure interaction problems. The techniques used here rely on the highly–evolved fields of optimal control and reduced basis methods.

In the first two chapters, we provided a general setup of any domain–decomposition algorithm (Chapter 1) and a reduced–order model (Chapter 2). We introduced basic notations and provided an overview of classical approaches for each setting. In the rest of the thesis, we provided an extensive description and analysis of applications to various problems.

In Chapter 3, we proposed a reduced–order model for the optimisation–based domain decomposition formulation of the parameter–dependent stationary incompressible 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 have managed to build a model for which the generation of the reduced basis spaces is carried out separately in each subdomain and for each component of the problem solution. Furthermore, as the numerical results show, the reduction of the optimal–control problem can be observed not only in the dimensions of the different components of the problem, i.e., the functional, the state and the adjoint equations but also in the number of the iterations of the optimisation algorithm. The presented model has been tested on two CFD benchmark problems: backward–facing step and the

lid–driven cavity flows.

Chapter 4 deals with non–stationary fluid dynamics problems. We described and conducted, based on the *a priori* estimates, 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. The presented model has been tested on two non–stationary CFD problems, namely backward–facing step and the lid–driven cavity flows.

In Chapter 5, we focused on the application of the methodology in the context of multi-physics problems. In particular, we considered a non–stationary FSI problem with the incompressible Navier–Stokes equations and the linear elasticity model. The original problem cast into the optimisation–based domain–decomposition framework leads to the optimal control problem aiming at minimising the velocity coupling error at the fluid–structure interface; the problem, then, has been tackled using either an iterative gradient–based optimisation algorithm or by the Gauss–Newton algorithm in the non–linear least squares optimisation framework. These allowed us to obtain a complete separation of the solvers on different subdomains. At the reduced–order level, we have built a model for which the generation of the reduced basis spaces is carried out separately in each subdomain and for each component of the problem. 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 optimisation iterations. Another extremely important feature of the algorithms presented in Chapter 5 is the fact that they are stable under the assumption of the "added mass" effect, which causes numerical instabilities in many partitioned approaches to FSI problems: this has been confirmed by the numerical experiments of the model presented for a two–dimensional haemodynamics benchmark FSI problem.

6.2 Perspectives

In this last section, I would like to provide some ideas for possible improvements for the techniques developed in this thesis and future perspectives.

In Chapter 3, in the numerical examples presented, we faced a problem of sensitivity of gradient–based optimisation algorithms to the initial guess. As mentioned in that section, this could be due to the non–uniqueness of the solution because of possible bifurcation phenomena present in many stationary fluid dynamics problems. In this context, one direction of improvement could be changing the perspective of how to address the resulting optimal–control problem or, in

the cases where bifurcation phenomena are proved to develop, account for them in the objective functional, as it is done, for instance, in [139].

In Chapter 4, we conducted a comparison between classical POD–Galerkin ROM and the POD–NN model based on the use of artificial neural networks. In terms of computational time, the POD–NN is very effective, but the approximation can be very poor when dealing with discontinuities, especially in the initial and final time steps. 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 a much more computationally effective ANN model fails to produce a sufficient approximation, the ROM is applied. Similar ideas can be found *inter alia* in [16].

In the course of the thesis, we considered only the two–domain decomposition and the obvious extension could be considering many–domain decomposition models that could greatly alleviate computational costs. In addition to this, we might also consider the models where different discretisation techniques, such as FEM, FV or DG, are used on the different subdomains. Regarding the Galerkin–POD ROM, as it has been mentioned a few times throughout the thesis, the online stage can be made much less expensive by ensuring an affine–decomposition assumption of the nonlinear problems via various techniques, such as EIM, DEIM or EQM.

In Chapter 5, we developed a stable fully–segregated model for an FSI problem. The future extensions of this methodology, just to name a few, may include combining this segregated approach with further domain–decomposition methods used for separate components of the multiphysics problem, compressible or non–Newtonian fluids and nonlinear elasticity models. In addition, this technique can be applied in a similar fashion to other multiphysics problems, and not only FSI.

Recently, the authors of the paper [46] have introduced a novel partitioned approach for ROMs, where they couple either two different reduced–order models on each subdomain or a reduced–order model on one subdomain and a full–order (Finite Element) model on the other for the case of nonstationary diffusion–advection problems. In this context, the construction followed in this thesis could be also applicable to the coupling presented in the paper, as long as there is a way of casting functions defined on the subdomain interface onto the approximation spaces used on the corresponding subdomains.

Finally, I would like to mention the combination of domain–decomposition methods with optimal control or inverse problems. An optimisation–based domain approach to domain–decomposition described here together with an optimal control problem leads to multi–objective optimisation set up on each subdomain, as it is done, for instance, in [74].

Acknowledgements

I am not going to lie. This has been a difficult journey. Probably the hardest 4 years of my life. Pandemics and the war enacted by the terrorist state of russia on Ukraine had brought rough times. But these were also among the most exciting years of my life. If it wasn't for the support of my loved ones, I wouldn't have arrived at this point.

First of all, I would like to say thank you to my supervisor Gianluigi Rozza, for not only being a valuable scientific advisor but also for showing his human face, being conscious and supportive in difficult moments and reminding me why this path of PhD is so important to me. I thank Monica Nonino for being with me from the very beginning of my path and for the valuable contribution and support I have received from her over these years. I thank Davide Torlo for taking me in at the time when I needed it the most, for his constant presence (even whenever I came knocking on his office door at SISSA), for invaluable scientific discussions and for all the help I got from him to fulfil my work. I also express my gratitude to Francesco Ballarin for his indispensable help at the beginning of my PhD and valuable contribution to our first published paper.

SISSA and, in particular, the mathLab group have been extremely comfortable and stimulating places to do my thesis, to spend all days studying and programming and mainly for people who have been around me during these years: Francesco, Simone, Romina, Sasha, Martina, Fabio, D.M., P.V., Riccardo, Warren, Matteo, Laura, Pasquale and many others. Many thanks also to the Student Council for giving me an opportunity to discuss the issues common for international students and the school in general and give my contribution to make the experiences of my peers at SISSA (hopefully) easier. I also thank Gabriella and Marco for supporting my ideas and contributing to their accomplishments; and also just for pleasant cigarette breaks (disclaimer: smoking is bad for your health).

Now, it's time for Trieste. I could have never imagined what a life I was going to have here. First of all, I would like to express my immense gratitude to Smarza Pride and all the affine collectives for giving meaning to my life in Trieste outside the scientific world; for the possibility to participate in creating a safe queer political space, for giving me a strong sense of belonging, for organising many events and discussions on various topics concerning all the axes of our identity, for creating the first two non–institutional horizontal prides in Trieste with taking over the streets, for overcoming my stage fright and insecurities about my Italian when speaking in front of an immense crowd

of people during the last pride. Many thanks to the friends I have acquired here during these years — Elizaveta, Diletta, Berenice, Chiara, Silvia, Emma, David, Andrea M., Janiki, Jessica, Lucio, Andrea, Francesco, Barbara, Martina, Anna, Tiziana, Arved — for extremely interesting philosophical discussions, your love and friendship and just for all the pleasant time we spend at various events organised by you and over aperitivo and beers (yet another disclaimer: alcohol is bad for your health). Also, I want to say thank you and express my love to three very special people scattered all over Italy — Michi, AnarchOtter and π.

I would like to mention specifically my passion - cinema, which helped me to survive the pandemic. Especially, I would like to thank Ukrainian *cinema chat (*гей клуб*)* for a great cinephilic community — among which mostly Den. S, Alin and Dmytro — and Trieste Film Festival for the amazing possibility to enter the world of the cinema, especially, Nicoletta Romeo, the artistic director of TSFF.

Much of my gratitude and worries go to my long–time friends in Ukraine: Nina, Ostap, Marta, Viktoriia, Ivan, Nazar, Sergii, Viktoriia, Orysia, Hannusia. Thank you for the years we stayed in touch and I hope all of you are safe wherever you are. I love all of you very much.

Last, but not least, I express my deep gratitude to my parents. Despite many disagreements we have, I love you and I owe you for many things I have achieved so far. I hope you will stay safe and know that I will be back at some point.

Bibliography

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

[2] multiphenics - easy prototyping of multiphysics problems in FEniCS, http://mathlab.sissa.it/multiphenics, 2016.

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

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

[5] V. I. Agoshkov. Poincaré-Steklov operators and domain decomposition methods in finite dimensional spaces. In *First International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pages 73–112, 1988.

[6] V. I. Agoshkov and V. I. Lebedev. Poincaré-Steklov operators and methods of partition of the domain in variational problems. In *Computational processes and systems, No. 2*, pages 173–227. "Nauka", Moscow, 1985.

[7] M. Al-Nasra and D. Nguyen. An algorithm for domain decomposition in finite element analysis. *Computers & Structures*, 39(3):277–289, 1991.

[8] 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.

[9] B. O. Almroth, P. Stern, and F. A. Brogan. Automatic choice of global shape functions in structural analysis. *Aiaa Journal*, 16(5):525–528, 1978.

[10] D. Amsallem, J. Cortial, and C. Farhat. Towards real-time computational-fluid-dynamics-based aeroelastic computations using a database of reduced-order information. *AIAA journal*, 48(9):2029–2037, 2010.

[11] D. Amsallem, M. Zahr, Y. Choi, and C. Farhat. Design optimization using hyper-reduced-order models. *Structural and Multidisciplinary Optimization*, 51(4):919–940, 2015.

[12] P. F. Antonietti, S. Giani, and P. Houston. Domain decomposition preconditioners for discontinuous Galerkin methods for elliptic problems on complicated domains. *Journal of Scientific Computing*, 60:203–227, 2014.

[13] 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.

[14] S. Badia, A. Quaini, and A. Quarteroni. Modular vs. non-modular preconditioners for fluid–structure systems with large added-mass effect. *Computer Methods in Applied Mechanics and Engineering*, 197(49-50):4216–4232, 2008.

[15] S. Badia, A. Quaini, and A. Quarteroni. Splitting methods based on algebraic factorization for fluid-structure interaction. *SIAM Journal on Scientific Computing*, 30(4):1778–1805, 2008.

[16] 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.

[17] 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.

[18] 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.

[19] 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 International Publishing, 2017.

[20] M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera. An 'empirical interpolation' method: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematique*, 339(9):667–672, 2004.

[21] S. Basting, A. Quaini, S. Čanić, and R. Glowinski. Extended ALE method for fluid–structure interaction problems with large structural displacements. *Journal of Computational Physics*, 331:312–336, 2017.

[22] S. Basting, A. Quaini, S. Čanić, and R. Glowinski. *1. On the implementation and benchmarking of an extended ALE method for FSI problems*, pages 3–40. De Gruyter, Berlin, Boston, 2018.

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

[24] P. Benner, M. Ohlberger, A. Patera, G. Rozza, and K. Urban. Model reduction of parametrized systems. In *MODEL REDUCTION OF PARAMETRIZED SYSTEMS*, volume 17 of *MS&A-Modeling Simulation and Applications*, pages 1–504. Springer Cham, 2017.

[25] P. Benner, W. Schilders, S. Grivet-Talocia, A. Quarteroni, G. Rozza, and L. Miguel Silveira. *Model order reduction: volume 3 applications*. De Gruyter, 2020.

[26] P.-M. Berthe, C. Japhet, and P. Omnes. Space–time domain decomposition with Finite Volumes for porous media applications. *Lecture Notes in Computational Science and Engineering*, pages 567–575, 2014.

[27] P. Bjorstad and O. Widlund. Iterative Methods for the Solution of Elliptic Problems on Regions Partitioned into Substructures. *SIAM J. Numer. Anal.*, 23:1097–1120, 12 1986.

[28] D. Bonomi, A. Manzoni, and A. Quarteroni. A matrix discrete empirical interpolation method for the efficient model reduction of parametrized nonlinear PDEs: Application to nonlinear elasticity problems. *MATHICSE, Ecole Polytechn. Fédérale de Lausanne, Lausanne, Switzerland, Tech. Rep*, 14:2016, 2016.

[29] J.-F. Bourgat, R. Glowinski, and P. Le Tallec. *Variational formulation and algorithm for trace operator in domain decomposition calculations*. INRIA, 1988.

[30] J. Bramble, J. Pasciak, and A. Schatz. An iterative method for elliptic problems on regions partitioned into substructures. *Mathematics of Computation*, 46:361–369, 04 1986.

[31] 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.

[32] 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.

[33] A. Caiazzo, T. Iliescu, V. John, and S. Schyschlowa. A numerical investigation of velocity–pressure reduced order models for incompressible flows. *Journal of Computational Physics*, 259:598–616, 2014.

[34] G. Caloz and J. Rappaz. Numerical analysis for nonlinear and bifurcation problems. *Handbook of numerical analysis*, 5:487–637, 1997.

[35] C. Canuto, T. Tonn, and K. Urban. A posteriori error analysis of the reduced basis method for nonaffine parametrized nonlinear PDEs. *SIAM Journal on Numerical Analysis*, 47(3):2001–2022, 2009.

[36] 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.

[37] E. Casas and M. Mateos. Optimal control of partial differential equations. *Computational Mathematics, Numerical Analysis and Applications: Lecture Notes of the XVII'Jacques-Louis Lions' Spanish-French School*, pages 3–59, 2017.

[38] P. Causin, J. Gerbeau, and F. Nobile. Added-mass effect in the design of partitioned algorithms for fluid–structure problems. *Computer Methods in Applied Mechanics and Engineering*, 194(42):4506–4527, 2005.

[39] A. Cesmelioglu, H. Lee, A. Quaini, K. Wang, and S.-Y. Yi. Optimization-based decoupling algorithms for a fluid-poroelastic system. *Topics in numerical partial differential equations and scientific computing*, pages 79–98, 2016.

[40] T. F. Chan and T. P. Mathew. Domain decomposition algorithms. *Acta numerica*, 3:61–143, 1994.

[41] D. Chapelle, A. Gariah, and J. Sainte-Marie. Galerkin approximation with proper orthogonal decomposition: new error estimates and illustrative examples. *ESAIM: Mathematical Modelling and Numerical Analysis*, 46(4):731–757, 2012.

[42] S. Chaturantabut. *Dimension reduction for unsteady nonlinear partial differential equations via empirical interpolation methods*. Rice university, 2009.

[43] S. Chaturantabut and D. C. Sorensen. Nonlinear model reduction via discrete empirical interpolation. *SIAM Journal on Scientific Computing*, 32(5):2737–2764, 2010.

[44] F. Chinesta, A. Huerta, G. Rozza, and K. Willcox. *Model Reduction Methods*, pages 1–36. John Wiley & Sons, Ltd, 2017.

[45] 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.

[46] A. de Castro, P. Kuberry, I. Tezaur, and P. Bochev. A novel partitioned approach for reduced order model—finite element model (ROM-FEM) and ROM-ROM coupling. In *Earth and Space 2022*, pages 475–489. American Society of Civil Engineers, 2022.

[47] N. Demo, M. Tezzele, A. Mola, G. Rozza, et al. A complete data-driven framework for the efficient solution of parametric shape design and optimisation in naval engineering problems. In *8th International Conference on Computational Methods in Marine Engineering, MARINE 2019*, pages 111–121. International Center for Numerical Methods in Engineering, 2019.

[48] N. Demo, M. Tezzele, and G. Rozza. A non-intrusive approach for the reconstruction of POD modal coefficients through active subspaces. *Comptes Rendus Mécanique*, 347(11):873–881, 2019.

[49] 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.

[50] V. Dolean, P. Jolivet, and F. Nataf. *An introduction to domain decomposition methods: algorithms, theory, and parallel implementation*. SIAM, 2015.

[51] J. Donea, S. Giuliani, and J. Halleux. An arbitrary lagrangian-eulerian finite element method for transient dynamic fluid-structure interactions. *Computer Methods in Applied Mechanics and Engineering*, 33(1):689–723, 1982.

[52] M. Drohmann, B. Haasdonk, and M. Ohlberger. Reduced basis approximation for nonlinear parametrized evolution equations based on empirical operator interpolation. *SIAM Journal on Scientific Computing*, 34(2):A937–A969, 2012.

[53] C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

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

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

[56] C. Farhat, R. Tezaur, and J. Toivanen. A domain decomposition method for discontinuous Galerkin discretizations of Helmholtz problems with plane waves and lagrange multipliers. *International journal for numerical methods in engineering*, 78(13):1513–1531, 2009.

[57] M. A. Fernández, J.-F. Gerbeau, and C. Grandmont. A projection semi-implicit scheme for the coupling of an elastic structure with an incompressible fluid. *International Journal for Numerical Methods in Engineering*, 69(4):794–821, 2007.

[58] M. A. Fernández, M. Landajuela, and M. Vidrascu. Fully decoupled time-marching schemes for incompressible fluid/thin-walled structure interaction. *Journal of Computational Physics*, 297:156–181, 2015.

[59] R. L. Fox and H. Miura. An approximate analysis technique for design calculations. *AIAA Journal*, 9(1):177–179, 1971.

[60] S. Fresca, L. Dedè, and A. Manzoni. A comprehensive deep learning-based approach to reduced order modeling of nonlinear time-dependent parametrized PDEs. *Journal of Scientific Computing*, 87:1–36, 2021.

[61] S. Fresca, A. Manzoni, L. Dedè, and A. Quarteroni. POD-enhanced deep learning-based reduced order models for the real-time simulation of cardiac electrophysiology in the left atrium. *Frontiers in physiology*, 12:679076, 2021.

[62] D. Funaro, A. Quarteroni, and P. Zanolli. An iterative procedure with interface relaxation for domain decomposition methods. *SIAM Journal on Numerical Analysis*, 25:1213–1236, 1988.

[63] S. Gaggero, G. Vernengo, D. Villa, and L. Bonfiglio. A reduced order approach for optimal design of efficient marine propellers. *Ships and Offshore Structures*, 15(2):200–214, 2020.

[64] 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.

[65] S. Giani and P. Houston. Domain decomposition preconditioners for discontinuous Galerkin discretizations of compressible fluid flows. *Numerical Mathematics: Theory, Methods and Applications*, 7(2):123–148, 2014.

[66] M. Girfoglio, A. Quaini, and G. Rozza. A pod-galerkin reduced order model for a LES filtering approach. *Journal of Computational Physics*, 436:110260, 2021.

[67] G. H. Golub and C. F. Van Loan. *Matrix computations*. JHU press, 2013.

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

[69] 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, 2012.

[70] C. Grandmont and Y. Maday. Fluid-structure interaction: a theoretical point of view. *Revue européenne des éléments finis*, 9(6-7):633–653, 2000.

[71] M. A. Grepl, Y. Maday, N. C. Nguyen, and A. T. Patera. Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations. *ESAIM: Mathematical Modelling and Numerical Analysis*, 41(3):575–605, 2007.

[72] M. A. Grepl and A. T. Patera. A posteriori error bounds for reduced-basis approximations of parametrized parabolic partial differential equations. *ESAIM: Mathematical Modelling and Numerical Analysis*, 39(1):157–181, 2005.

[73] C. Groetsch. *Generalized Inverses of Linear Operators: Representation and Approximation*. Dekker, 1977.

[74] 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.

[75] 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.

[76] M. D. Gunzburger. *Perspectives in Flow Control and Optimization*. Society for Industrial and Applied Mathematics, 2002.

[77] 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.

[78] 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.

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

[80] M. Hess, A. Alla, A. Quaini, G. Rozza, and M. Gunzburger. A localized reduced-order modeling approach for PDEs with bifurcating solutions. *Computer Methods in Applied Mechanics and Engineering*, 351:379–403, 2019.

[81] 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.

[82] 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.

[83] M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich. *Optimization with PDE Constraints*, volume 23 of *Mathematical modelling*. Springer, 2009.

[84] 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.

[85] R. Hu, F. Fang, C. Pain, and I. Navon. Rapid spatio-temporal flood prediction and uncertainty quantification using a deep learning method. *Journal of Hydrology*, 575:911–920, 2019.

[86] T. J. Hughes, W. K. Liu, and T. K. Zimmermann. Lagrangian-Eulerian finite element formulation for incompressible viscous flows. *Computer Methods in Applied Mechanics and Engineering*, 29(3):329–349, 1981.

[87] D. B. P. Huynh, D. J. Knezevic, and A. T. Patera. A static condensation reduced basis element method: approximation and a posteriori error estimation. *ESAIM: Mathematical Modelling and Numerical Analysis*, 47(1):213–251, 2013.

[88] D. B. P. Huynh, G. Rozza, S. Sen, and A. T. Patera. A successive constraint linear optimization method for lower bounds of parametric coercivity and inf–sup stability constants. *Comptes Rendus. Mathématique*, 345(8):473–478, 2007.

[89] L. Iapichino, A. Quarteroni, and G. Rozza. A reduced basis hybrid method for the coupling of parametrized domains represented by fluidic networks. *Computer Methods in Applied Mechanics and Engineering*, 221:63–82, 2012.

[90] T. Iliescu and Z. Wang. Variational multiscale proper orthogonal decomposition: Navier-stokes equations. *Numerical Methods for Partial Differential Equations*, 30(2):641–663, 2014.

[91] A. Iollo, G. Sambataro, and T. Taddei. A one-shot overlapping Schwarz method for component-based model reduction: application to nonlinear elasticity. *Computer Methods in Applied Mechanics and Engineering*, 404:115786, 2023.

[92] K. Ito and S. S. Ravindran. A reduced basis method for control problems governed by PDEs. In *Control and Estimation of Distributed Parameter Systems: International Conference in Vorau, Austria, July 14-20, 1996*, pages 153–168. Springer, 1998.

[93] C. Jäggli, L. Iapichino, and G. Rozza. An improvement on geometrical parameterizations by transfinite maps. *Comptes Rendus. Mathématique*, 352(3):263–268, 2014.

[94] J. D. Jansen and L. J. Durlofsky. Use of reduced-order models in well control optimization. *Optimization and Engineering*, 18:105–132, 2017.

[95] N. Jung, B. Haasdonk, and D. Kroner. Reduced basis method for quadratically nonlinear transport equations. *International Journal of Computing Science and Mathematics*, 2(4):334–353, 2009.

[96] M. Kahlbacher and S. Volkwein. Galerkin proper orthogonal decomposition methods for parameter dependent elliptic systems. *Discussiones Mathematicae, Differential Inclusions, Control and Optimization*, 27(1):95–117, 2007.

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

[98] 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.

[99] 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.

[100] K. Kunisch and S. Volkwein. Galerkin proper orthogonal decomposition methods for parabolic problems. *Numerische mathematik*, 90:117–148, 2001.

[101] K. Kunisch and S. Volkwein. Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics. *SIAM Journal on Numerical analysis*, 40(2):492–515, 2002.

[102] Y. A. Kuznetsov, I. A. Kuznetsov, and Y. Kuznetsov. *Elements of applied bifurcation theory*, volume 112. Springer, 1998.

[103] J. E. Lagnese 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.

[104] T. Lassila, A. Manzoni, A. Quarteroni, and G. Rozza. A reduced computational and geometrical framework for inverse problems in hemodynamics. *International journal for numerical methods in biomedical engineering*, 29(7):741–776, 2013.

[105] 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.

[106] T. Lassila, A. Quarteroni, and G. Rozza. A reduced basis model with parametric coupling for fluid-structure interaction problems. *SIAM Journal on Scientific Computing*, 34(2):A1187–A1213, 2012.

[107] T. Lassila and G. Rozza. Model reduction of steady fluid-structure interaction problems with reduced basis methods and free-form deformations. *Proceedings of 10th Finnish Mechanics Days (Jyvaskyla, Finland)*, 2009.

[108] K. Lee and K. T. Carlberg. Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. *Journal of Computational Physics*, 404:108973, 2020.

[109] G. Leugering, S. Engell, A. Griewank, M. Hinze, R. Rannacher, V. Schulz, M. Ulbrich, and S. Ulbrich. *Constrained optimization and optimal control for partial differential equations*, volume 160. Springer Science & Business Media, 2012.

[110] T. Lieu, C. Farhat, and M. Lesoinne. POD-based aeroelastic analysis of a complete F-16 configuration: ROM adaptation and demonstration. In *46th AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics and materials conference*, page 2295, 2005.

[111] J. L. Lions. *Optimal control of systems governed by partial differential equations*, volume 170. Springer, 1971.

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

[113] P. L. Lions. On the Schwarz alternating method III: A Variant for Nonoverlapping Subdomains. In T. F. Chan, R. Glowinski, J. Périaux, and O. B. Widlund, editors, *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pages 202–223. Society for Industrial and Applied Mathematics, 1990.

[114] M. Liu and Z. Cai. Adaptive two-layer ReLU neural network: II. Ritz approximation to elliptic PDEs. *Computers & Mathematics with Applications*, 113:103–116, 2022.

[115] A. Løvgren, Y. Maday, and E. Rønquist. The reduced basis element method: Offline-online decomposition in the nonconforming, nonaffine case. In *Spectral and High Order Methods for Partial Differential Equations: Selected papers from the ICOSAHOM'09 conference, June 22-26, Trondheim, Norway*, pages 247–254. Springer, 2011.

[116] A. E. Løvgren, Y. Maday, and E. M. Rønquist. The reduced basis element method for fluid flows. In *Analysis and simulation of fluid dynamics*, pages 129–154. Springer, 2006.

[117] A. Manzoni, S. Pagani, and T. Lassila. Accurate solution of Bayesian inverse uncertainty quantification problems combining reduced basis methods and reduction error models. *SIAM/ASA Journal on Uncertainty Quantification*, 4(1):380–412, 2016.

[118] A. Manzoni, A. Quarteroni, and G. Rozza. Computational reduction for parametrized PDEs: strategies and applications. *Milan Journal of Mathematics*, 80:283–309, 2012.

[119] A. Manzoni, A. Quarteroni, and S. Salsa. *Optimal control of partial differential equations*. Springer, 2021.

[120] K. Marepally, Y. S. Jung, J. Baeder, and G. Vijayakumar. Uncertainty quantification of wind turbine airfoil aerodynamics with geometric uncertainty. In *Journal of Physics: Conference Series*, volume 2265, page 042041. IOP Publishing, 2022.

[121] A. L. Marsden and M. Esmaily-Moghadam. Multiscale modeling of cardiovascular flows for clinical decision support. *Applied Mechanics Reviews*, 67(3):030804, 2015.

[122] I. Martini, G. Rozza, and B. Haasdonk. Reduced basis approximation and a-posteriori error estimation for the coupled Stokes-Darcy system. *Advances in Computational Mathematics*, 41:1131–1157, 2015.

[123] M. Mohebujjaman, L. G. Rebholz, and T. Iliescu. Physically constrained data-driven correction for reduced-order modeling of fluid flows. *International Journal for Numerical Methods in Fluids*, 89(3):103–122, 2019.

[124] I. Navon and Y. Cai. Domain decomposition and parallel processing of a finite element model of the shallow water equations. *Computer Methods in Applied Mechanics and Engineering*, 106(1):179–212, 1993.

[125] N. C. Nguyen, K. Veroy, and A. T. Patera. Certified real-time solution of parametrized partial differential equations. *Handbook of Materials Modeling: Methods*, pages 1529–1564, 2005.

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

[127] 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.

[128] A. K. Noor. Recent advances in reduction methods for nonlinear problems. *Computational methods in nonlinear structural and solid Mechanics*, pages 31–44, 1981.

[129] A. K. Noor. On making large nonlinear problems small. *Computer methods in applied mechanics and engineering*, 34(1-3):955–985, 1982.

[130] A. K. Noor and J. M. Peters. Reduced basis technique for nonlinear analysis of structures. *Aiaa journal*, 18(4):455–462, 1980.

[131] J. M. Nordbotten and P. E. Bjørstad. On the relationship between the multiscale finite-volume method and domain decomposition preconditioners. *Computational Geosciences*, 12:367–376, 2008.

[132] M. Ohlberger and S. Rave. Reduced basis methods: Success, limitations and future challenges. *Proceedings of the Conference Algoritmy*, pages 1–12, 2016.

[133] T. Panitz and D. Wasan. Flow attachment to solid surfaces: the Coanda effect. *AIChE Journal*, 18(1):51–57, 1972.

[134] A. T. Patera and M. Yano. An LP empirical quadrature procedure for parametrized functions. *Comptes Rendus. Mathématique*, 355(11):1161–1167, 2017.

[135] J. S. Peterson. The reduced basis method for incompressible viscous flow calculations. *SIAM Journal on Scientific and Statistical Computing*, 10(4):777–786, 1989.

[136] P. Philip. Optimal control of partial differential equations. *Lecture Notes, Ludwig-Maximilians-Universität, Germany*, 2009.

[137] F. Pichi, B. Moya, and J. S. Hesthaven. A graph convolutional autoencoder approach to model order reduction for parametrized PDEs. *arXiv preprint arXiv:2305.08573*, 2023.

[138] F. Pichi and G. Rozza. Reduced basis approaches for parametrized bifurcation problems held by non-linear Von Kármán equations. *Journal of Scientific Computing*, 81:112–135, 2019.

[139] 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.

[140] R. Pinnau. Model reduction via proper orthogonal decomposition. In *Model order reduction: theory, research aspects and applications*, pages 95–109. Springer, 2008.

[141] M. Pintore, F. Pichi, M. Hess, G. Rozza, and C. Canuto. Efficient computation of bifurcation diagrams with a deflated approach to reduced basis spectral element method. *Advances in Computational Mathematics*, 47:1–39, 2021.

[142] C. Prud'homme, D. V. Rovas, K. Veroy, L. Machiels, Y. Maday, A. T. Patera, and G. Turinici. Reliable real-time solution of parametrized partial differential equations: Reduced-basis output bound methods. *Journal of Fluids Engineering*, 124(1):70–80, 11 2002.

[143] 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.

[144] A. Quaini. Algorithms for fluid-structure interaction problems arising in hemodynamics. Technical report, EPFL, 2009.

[145] A. Quaini, R. Glowinski, and S. Čanić. Symmetry breaking and preliminary results about a Hopf bifurcation for incompressible viscous flow in an expansion channel. *International Journal of Computational Fluid Dynamics*, 30(1):7–19, 2016.

[146] A. Quarteroni. Domain decomposition techniques using spectral methods. *Calcolo*, 24(2):141–177, 1987.

[147] A. Quarteroni, A. Manzoni, and C. Vergara. The cardiovascular system: mathematical modelling, numerical algorithms and clinical applications. *Acta Numerica*, 26:365–590, 2017.

[148] A. Quarteroni and G. Rozza. Numerical solution of parametrized Navier–Stokes equations by reduced basis methods. *Numerical methods for partial differential equations: an international journal*, 23(4):923–948, 2007.

[149] A. Quarteroni, G. Rozza, and A. Manzoni. Certified reduced basis approximation for parametrized partial differential equations and applications. *Journal of Mathematics in Industry*, 1, 12 2011.

[150] 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.

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

[152] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

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

[154] M. Ripepi, M. J. Verveld, N. Karcher, T. Franz, M. Abu-Zurayk, S. Görtz, and T. M. Kier. Reduced-order models for aerodynamic applications, loads and MDO. *CEAS Aeronautical Journal*, 9(1):171–193, 2018.

[155] F. Romor, D. Torlo, and G. Rozza. Friedrichs' systems discretized with the Discontinuous Galerkin method: domain decomposable model order reduction and Graph Neural Networks approximating vanishing viscosity solutions. *arXiv preprint arXiv:2308.03378*, 2023.

[156] G. Rozza. Reduced-basis methods for elliptic equations in sub-domains with a posteriori error bounds and adaptivity. *Applied Numerical Mathematics*, 55(4):403–424, 2005.

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

[158] G. Rozza, M. Hess, G. Stabile, M. Tezzele, and F. Ballarin. *1 Basic ideas and tools for projection-based model reduction of parametric partial differential equations*, pages 1–47. De Gruyter, Berlin, Boston, 2021.

[159] G. Rozza, D. B. P. Huynh, and A. T. Patera. Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations: application to transport and continuum mechanics. *Archives of Computational Methods in Engineering*, 15(3):229–275, 2008.

[160] G. Rozza, C. Nguyen, A. T. Patera, and S. Deparis. Reduced basis methods and a posteriori error estimators for heat transfer problems. In *Heat Transfer Summer Conference*, volume 43574, pages 753–762, 2009.

[161] G. Rozza, G. Stabile, and F. Ballarin. *Advanced Reduced Order Methods and Applications in Computational Fluid Dynamics*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2022.

[162] H. A. Schwarz. Ueber einige Abbildungsaufgaben. *J. Reine Angew. Math.*, 70:105–112, 1869.

[163] R. Seydel. *Practical bifurcation and stability analysis*, volume 5. Springer Science & Business Media, 2009.

[164] P. Siena, M. Girfoglio, and G. Rozza. Chapter 6 - an introduction to POD-greedy-Galerkin reduced basis method. In F. Chinesta, E. Cueto, Y. Payan, and J. Ohayon, editors, *Reduced Order Models for the Biomechanics of Living Organs*, Biomechanics of Living Organs, pages 127–145. Academic Press, 2023.

[165] B. F. Smith. Domain decomposition methods for partial differential equations. In *Parallel Numerical Algorithms*, pages 225–243. Springer, 1997.

[166] S. Sobolev. Schwarz's algorithm in elasticity theory. *Selected Works of SL Sobolev: Volume I: Mathematical Physics, Computational Mathematics, and Cubature Formulas*, pages 399–403, 2006.

[167] 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.

[168] 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.

[169] 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.

[170] 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.

[171] 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.

[172] 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.

[173] M. Tezzele, F. Ballarin, and G. Rozza. Combined parameter and model reduction of cardiovascular problems by means of active subspaces and POD-Galerkin methods. *Mathematical and numerical modeling of the cardiovascular system and applications*, pages 185–207, 2018.

[174] M. Tezzele, N. Demo, and G. Rozza. Shape optimization through proper orthogonal decomposition with interpolation and dynamic mode decomposition enhanced by active subspaces.

In *MARINE VIII: proceedings of the VIII International Conference on Computational Methods in Marine Engineering*, pages 122–133. CIMNE, 2019.

[175] 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.

[176] M. Tezzele, F. Salmoiraghi, A. Mola, and G. Rozza. Dimension reduction in heterogeneous parametric spaces with application to naval engineering shape design problems. *Advanced Modeling and Simulation in Engineering Sciences*, 5(1):1–19, 2018.

[177] 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.

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

[179] A. Toselli and O. Widlund. *Domain decomposition methods-algorithms and theory*, volume 34. Springer Science & Business Media, 2004.

[180] F. Tröltzsch. *Optimal control of partial differential equations: theory, methods, and applications*, volume 112. American Mathematical Soc., 2010.

[181] 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.

[182] K. Veroy, C. Prud'Homme, D. Rovas, and A. Patera. A posteriori error bounds for reduced-basis approximation of parametrized noncoercive and nonlinear elliptic partial differential equations. In *16th AIAA Computational Fluid Dynamics Conference*, page 3847, 2003.

[183] 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.

[184] Y. Wu and X.-C. Cai. A fully implicit domain decomposition based ALE framework for three-dimensional fluid–structure interaction with application in blood flow computation. *Journal of Computational Physics*, 258:524–537, 2014.

[185] R. Yondo, E. Andrés, and E. Valero. A review on design of experiments and surrogate models in aircraft real-time and many-query aerodynamic analyses. *Progress in aerospace sciences*, 96:23–61, 2018.

[186] M. J. Zahr and C. Farhat. Progressive construction of a parametric reduced-order model for PDE-constrained optimization. *International Journal for Numerical Methods in Engineering*, 102(5):1111–1135, 2015.

[187] P. Zanolli. Domain decomposition algorithms for spectral methods. *Calcolo*, 24(3-4):201–240, 1987.