Polytechnic University of Turin

Master's degree in Mathematical Engineering

Master thesis

Efficient Computation of Bifurcation Diagrams with Spectral Element Method and Reduced Order Models

Author:
Moreno Pintore

Advisors:
Prof. Claudio Canuto
Prof. Gianluigi Rozza

Co-Advisors:
Dr. Martin Hess
Dr. Federico Pichi

Academic year: 2018/2019

Contents

Introduction

Many physical phenomena can be described using partial differential equations (PDEs), they usually are non-linear and, due to the non-linearity, multiple different solutions can exist. Understanding this property and the relation between the different solutions and some parameters is crucial to predict the evolution of a system when the parameters can vary. Even if these phenomena can be represented with bifurcation diagrams, obtaining them analytically or theoretically is impossible for almost every interesting problem. For this reason, one would like to compute them numerically, but, due to the complexity of the task, several different techniques must be used together to perform it in an accurate and efficient way. Firstly, one has to be able to compute a solution of the problem for many values of the parameters, therefore the solver should be as accurate and fast as possible because it will be used many times. Secondly, a suitable continuation method is needed to efficiently follow a single branch of the bifurcation diagram and, lastly, it is important to use another method to detect the bifurcation points or discover new branches. In particular, the deflation method has been chosen to find them. The problem is that, even if all these elements are available and optimized, the computation of a bifurcation diagram is a very expensive operation; indeed the described methods have to be performed a huge number of times and the associated computational cost can easily become prohibitive if more than one parameter is involved.
In this work an efficient way to compute bifurcation diagrams with one or more parameters is proposed and all the different required techniques are explained with a particular focus on their implementation and the obtained results. We highlight that the described method can be used in several different scenarios, in fact the *spectral element method* [42] (SEM) has been used to compute the full order solutions required to generate the reduced space with the *reduced basis* [38] (RB) method. These two techniques are very general because they simply consider the variational formulation of an arbitrary equation. Moreover, we implemented an advanced *deflated continuation* method [28] to efficiently compute the set of solutions that discretizes the bifurcation diagram. In such a technique we alternate the continuation method [25], implemented in two different ways in order to be able to accurately track all the branches during each phase of the computation, and the deflation one, that we paired with a novel heuristic to increase its effectiveness. Since a reduced order model (ROM) is used, the discussed methods can be divided in two phases: the first one, named *offline phase*, is the most expensive one and is responsible for the computation of the so called *full order solutions* or *snapshots*. Subsequently, in the *online phase*, the solution of the discrete problem is sought in a low-dimensional space and all the matrices and vectors are simply assembled [46] with objects created during the offline phase and, for this reason, the computation is much faster.
Initially, we highlight that, in appendix A, a short description of the *Navier-Stokes* [73] (NS) equations and an introduction to the computational fluid dynamics (CFD) will be presented, this will be a brief overview of some fundamental concepts that will be frequently used throughout

the whole text. In particular, in section A.1 the equations that will be used in the subsequent chapters will be derived adding some physical assumptions to the complete NS equations system in order to simplify it. Then, in section A.2, the most important non-dimensional number in CFD [16] will be presented, it is named *Reynolds number* and its variation can imply qualitative and quantitative changes in the solutions. Furthermore, if such a number is significantly increased, the computation of the solutions becomes a more complex task.
In Chapter 1 the SEM will be analyzed, focusing on the variational formulation and on the efficiency [33] and accuracy issues. In section 1.1 the method will be presented in one dimension. This is important to discuss, in an easier context, the Galerkin formulation [62] (section 1.1.1) and the different kinds of expansion bases that can be used to discretize the problem (section 1.1.2). These topics will be used and generalized to multidimensional problems in section 1.2, where the concept of local and global operations will be presented. Moreover, the *static condensation method* [42] will be analyzed in section 1.2.3.1, it is a technique used to increase the efficiency of the SEM exploiting the fact that the expansion bases can be divided in two different groups. Finally, in section 1.3, we will discuss how to apply the SEM to the CFD underlying the required variational formulation, the iterative methods that can be used (section 1.3.1) and the generalization of the static condensation method to consider both the velocity and the pressure degrees of freedom [78] (section 1.3.2).
Then, in Chapter 2, the ROM will be accurately described with concentration on the different structure of the formulation [64] with respect to SEM one (section 2.1) and on the method used for constructing the reduced basis [32] (section 2.2). Moreover, the affine decomposition required to ensure the efficiency of the method will be presented in section 2.3, while the extension of such topics to non-coercive problems [38] will be discussed in section 2.4. This is important because the variational problem associated to the NS equations is non-coercive and, therefore, in order to consider a well-posed problem, one has to rely on different theorems and conditions.
Later, in Chapter 3, all the methods and the theory required to obtain a bifurcation diagram will be presented. In section 3.1 the theory will be introduced [3] and the main ideas behind the numerical computation of such diagrams will be presented. Subsequently, in section 3.2, the *continuation method* [67] will be introduced and two different approaches will be discussed, focusing on their advantages and disadvantages. On the other hand, the second required technique, the *deflation method* [29], will be analyzed in section 3.3. Here we will focus on its interpretation in section 3.3.1 and on its efficiency in section 3.3.2.1.
Then, in Chapter 4, the numerical results that can be obtained with this method will be shown, first with only a single varying parameter and then with two. In such a chapter we will prove that, using the continuation and the deflation together, it is possible to obtain a bifurcation diagram including more bifurcation points and that the singular values decay involved in the POD method is only weakly influenced by the different branches or by the techniques used to obtain them. Moreover, we will highlight the stability issues that can imply several negative consequences. For instance, in section 4.2.2.2, we will show that the online solver could converge to unphysical solutions if the reduced basis is too noisy while, on the other hand, it may not converge if two parameters are involved and the tolerance of the POD is not properly selected (section 4.3.2.2). We highlight that the software used to solve the full order problem is *Nektar++* version 4.4.0 [68], that is an open source spectral/hp software, while

ITHACA-SEM (https://github.com/mathLab/ITHACA-SEM) has been used to perform the online phase. Moreover, the methods associated to the continuation method and to the deflation method have been implemented in *ITHACA-SEM*.
Finally the conclusion will be presented in Chapter 5. Here we will shortly summarize the content and the motivation of the thesis, furthermore, we will present some extensions and methods that could improve the results shown in Chapter 4.
Lastly, we want to remark that this thesis has been developed thanks to the *MathLab* team of SISSA (Trieste, Italy), which members consistently helped us to understand and implement the numerical methods, and to the European project *"H2020 ERC CoG AROMA-CFD"* that supported the entire work.

September 2019, Torino and Trieste

1. Spectral Element Method

1.1. Introduction and one-dimensional formulation

In many different fields of engineering and physics, problems governed by systems of PDEs arise; such systems are usually too complex to be analytically solved and specific numerical methods are required to obtain approximated solutions. A very common method is the finite element method (FEM) [62], it is based upon a variational formulation named *Galerkin formulation* and its mathematical foundations have been deeply investigated in the last century. One of the features of such a method is that it relies on a mesh [21]: a conforming partition of the entire domain in many small subdomains that, in general, are small polygons (or n-dimensional polytopes in $\mathbb{R}^n$) with few vertices.
The name of the method derives from the fact that the combination between a subdomain and a properly chosen finite set of functions is named *element*, the solution of the problem is discretized by means of a finite number of elements. The functions associated to each subdomain are, in general, low order polynomials that can approximate the real solution only locally; this way numerous elements are required to ensure a global convergence. The process of increasing the number of elements is called *mesh refinement*, while the associated type of convergence is denoted as h-type convergence (here h stands for the average diameter of the elements). The accuracy of the FEM is formally proved to algebraically increase by refining the mesh, the fact that the discrete solution tends to the real one increasing the number of elements is thus ensured by the mathematical foundations on which the method is based [6].
However, the FEM is not the most used method in CFD because it does not automatically respect the mass conservation constraint. Moreover, a stability condition (named inf-sup condition [10] and described in sections 1.3 and 2.4) can not be satisfied when $h \to 0$, not allowing the discrete solution to properly converge to the real one. Therefore, the most used method in CFD is the *finite volume method* [62] that relies on a different formulation that naturally respects the mass balance. Moreover, there exists a third approach associated to the so called *spectral methods* [18]. Several different methods can be denoted as spectral methods (see, for instance, [19]), they are, in general, characterized by an exponential convergence and by the fact that the involved discrete functions have supports that are wider then the one considered in the classical FEM.
In this work we will focus on a method called *spectral element method* (SEM), that is similar to the *hp*-version of the FEM [70]. The notation *hp* means that the method is characterized by a mesh refinement (h) [55] and by the use of polynomials which order can be increased to reduce the error (p) [24]. It is important to highlight that a pure p-type method, that can be seen as a FEM with a single element but with polynomials which order can be arbitrarily raised, ensures an exponential convergence. However, in order to handle the geometrical features of the domain, it is better to consider a mesh with few elements and a *hp*-approach that can exploit

the advantages of both methods at the same time [15]. This way it is possible to obtain very accurate solutions in complex domains with a limited number of degrees of freedom (unknowns in the linear systems associated to the problem). In fact, according to properly chosen error estimators, it is possible to understand if it is convenient to refine the mesh or to increase the polynomial order inside a specific element [20]. This is important in order to reduce the required computational cost associated to the problem that has to be solved.

1.1.1. Galerkin formulation

As the standard FEM, the SEM relies on the Galerkin formulation [42]. The latter will be presented in one dimension in this section in order to understand it better and to present some important results, while it will be generalized to higher dimensional problems only in the next sections. However, since the discussed topics will be extended, the notation associated to the partial derivatives will be exploited also in this section in order to avoid confusion. We highlight that we will discuss the Galerkin formulation as presented in [42].
Let us consider the strong Poisson equation:

$$L(u) = \frac{\partial^2 u}{\partial x^2} + f = 0, \tag{1.1}$$

where $u = u(x)$, the spatial variable x belongs to the open domain Ω defined as $\Omega = (0, 1)$ and homogeneous Dirichlet boundary conditions are prescribed as follows:

$$u(0) = u(1) = 0. \tag{1.2}$$

In order to obtain an integral, or *weak*, formulation of problem (1.1), one can properly select a pair of functional spaces V_{trial} and V_{test}, multiply the equation by a function $v \in V_{test}$, integrate such a product over the entire domain Ω, and seek the solution of the obtained integral problem in V_{test} [62]. Such spaces are defined as:

$$V_{trial} = \{u : u \in H^1(\Omega), u|_{\partial\Omega_D} = g_D\},$$
$$V_{test} = \{v : v \in H^1(\Omega), v|_{\partial\Omega_D} = 0\}.$$

where $\partial\Omega_D$ is the portion of the boundary of Ω associated to the Dirichlet boundary condition and g_D describes such a condition. It can be observed that, in problem (1.1) with the boundary conditions in (1.2), $\partial\Omega_D = \{0, 1\}$ and $g_D(x) = 0 \ \forall x \in \partial\Omega_D$. Therefore, V_{trial} coincides with V_{test} because of the homogeneous Dirichlet boundary conditions.
The integral equation is the following one:

$$\int_0^1 \left(\frac{\partial^2 u}{\partial x^2} + f\right) v dx = 0. \tag{1.3}$$

The latter can be integrated by parts and, using the constraints expressed by the Dirichlet boundary conditions, it can be equivalently written as:

$$\int_0^1 \frac{\partial u}{\partial x}\frac{\partial v}{\partial x} dx = \int_0^1 v f dx. \tag{1.4}$$

Finally, we can denote as the Galerkin approximation of problem (1.1) the function $u^\delta(x) \in V^\delta_{trial} \subset V_{trial}$ that satisfies problem (1.4) when V^δ_{trial} is a generic finite-dimensional vectorial subspace of V_{trial}. We remark that $u^\delta(x)$ is, in general, an approximation of $u(x)$, but it can be obtained much more easily. This way, introducing the forms $a(\cdot,\cdot) : V_{trial} \times V_{test} \to \mathbb{R}$ and $f(\cdot) : V_{test} \to \mathbb{R}$ and denoting the left hand side of problem (1.4) as $a(u,v)$ and its right hand side as $f(v)$, one can define the generalized weak formulation of problem (1.1) as:

Find $u \in V_{trial}$, such that:

$$a(u,v) = f(v), \qquad \forall v \in V_{test}.$$

In an analogous way, the approximate form of the weak formulation can be stated as:

Find $u \in V^\delta_{trial}$, such that:

$$a(u^\delta, v^\delta) = f(v^\delta), \qquad \forall v^\delta \in V^\delta_{test}. \tag{1.5}$$

1.1.1.1. Boundary conditions

An useful property of the Galerkin formulation is that the boundary conditions can be directly inserted in the equation, without having to handle them separately. Let us consider, for example, the previous problem with the following non homogeneous Dirichlet and Neumann boundary conditions:

$$u(0) = g_D, \qquad \frac{\partial u(1)}{\partial x} = g_N. \tag{1.6}$$

To ensure the well posedness of the problem, the test functions have to be zero on the Dirichlet boundary. However, in the pure Galerkin formulation the test space coincides with the trial one, therefore the approximate solution $u^\delta(x)$ is required to be defined as the sum of two different contributions. Let us denote as u^D a known function that satisfies the Dirichlet boundary condition, it is denoted as *lifting function* and it belongs to V^δ_{trial}. On the other hand, we will indicate with $u^H \in V^\delta_{test}$ the unknown part of u^δ [62]. This way it is possible to write u^δ as:

$$u^\delta(x) = u^H(x) + u^D(x). \tag{1.7}$$

Explicitly expanding the discretized version of equation (1.3) integrating by parts and using the assumption (1.7), one can obtain the equation:

$$\begin{aligned} \int_0^1 \frac{\partial u^H}{\partial x}\frac{\partial v^\delta}{\partial x}dx &= \int_0^1 v^\delta f dx + \left[\frac{\partial u^\delta}{\partial x} v^\delta\right]_0^1 - \int_0^1 \frac{\partial u^D}{\partial x}\frac{\partial v^\delta}{\partial x}dx \\ &= \int_0^1 v^\delta f dx + g_N v^\delta(1) - \int_0^1 \frac{\partial u^D}{\partial x}\frac{\partial v^\delta}{\partial x}dx. \end{aligned} \tag{1.8}$$

Here we used the fact that $\dfrac{\partial u^\delta}{\partial x}(0)v^\delta(0) = 0$ because $v^\delta(0) = 0$, since it belongs to V^δ_{test}. It should be noted that the right hand side is known or can be computed, at least numerically,

while the left hand one is unknown because it includes the unknown function u^H. From now on, if the subscripts "test" and "trial" are not required, they will be omitted for brevity. Furthermore, one may need to handle Robin conditions of the form:

$$\alpha \frac{\partial u(\cdot)}{\partial x} + \beta u(\cdot) = g_R, \qquad \alpha \neq 0;$$

it can be observed that such a condition can also be written as:

$$\frac{\partial u(\cdot)}{\partial x} = \frac{1}{\alpha}\left(g_R - \beta u(\cdot)\right), \qquad \alpha \neq 0. \tag{1.9}$$

The relation (1.9) can then be substituted as a Neumann condition in the first line of expression (1.8) to obtain the following equation (here we assumed to have a homogeneous Dirichlet boundary condition in $x = 0$ and the Robin boundary condition in $x = 1$):

$$\int_0^1 \frac{\partial u^\delta}{\partial x}\frac{\partial v^\delta}{\partial x} dx + \frac{\beta}{\alpha} u^\delta(1) v^\delta(1) = \int_0^1 v^\delta f dx + \frac{v g_R}{\alpha} v^\delta(1).$$

Finally, if one wants to express this problem with the general Galerkin formulation, one can state it as follows:

Find $u^\delta = u^D + u^H$, where

$$u^H \in V^\delta_{test}, \qquad u^D \in V^\delta_{trial},$$

such that:

$$a(u^H, v^\delta) = f(v^\delta) - a(u^D, v^\delta), \qquad \forall v^\delta \in V^\delta_{test}.$$

1.1.1.2. Properties of the Galerkin formulation

In order to consider only well posed problems, let us assume that $a(v, u)$ is a bilinear form, therefore the following relation holds:

$$a(c_1 u + c_2 v, w) = c_1 a(u, w) + c_2 a(v, w), \qquad \forall v, w \in V.$$

Here u, v and w are generic functions, while c_1 and c_2 are scalar constants. Moreover, $a(\cdot, \cdot)$ is required to be bounded and coercive:

$$\begin{aligned} |a(v, w)| &\leq \gamma \|v\|_V \|w\|_V, && \forall v, w \in V, \\ a(v, v) &\geq \alpha \|v\|_V^2, && \forall v \in V, \end{aligned}$$

where $\gamma, \alpha \in \mathbb{R}$ are such that $\gamma < +\infty$ and $\alpha > 0$. Under these assumptions, and if $f(\cdot)$ is in the dual space of V, the Lax-Milgram theorem guarantees the existence and the uniqueness of the solution of the abstract Galerkin problem [62]:

Find $u \in V$, such that:

$$a(u, v) = f(v), \qquad \forall v \in V. \tag{1.10}$$

Theorem 1.1.1 (Lax-Milgram Theorem)**.** *Let V be a Hilbert space, $a(\cdot,\cdot) : V \times V \to \mathbb{R}$ a bilinear, continuous and coercive form, and $f(\cdot) : V \to \mathbb{R}$ a continuous and linear form belonging to the dual space of V. Then the solution $u \in V$ of problem (1.10) exists and is unique. Furthermore, denoting as $\alpha > 0$ the coercivity constant of $a(\cdot,\cdot)$, the following inequality holds:*

$$\|u\|_V \leq \frac{1}{\alpha}\|f\|_{V'}.$$

It should be noted that, even if such a topic is out of the scope of the thesis, the Lax-Milgram theorem can be further generalized [26]. Moreover, it is important to observe that the discrete version of the Galerkin problem can be used to construct the linear system that has to be solved to obtain the discrete solution [62]. To deeper explain such a process, let us consider the discrete Galerkin problem:

Find $u^\delta \in V^\delta$, such that:

$$a(u^\delta, v^\delta) = f(v^\delta), \qquad \forall v^\delta \in V^\delta. \tag{1.11}$$

Here V^δ is a finite-dimensional space and, therefore, it can be described, using a set of basis functions, as $V^\delta = \text{span}\{\phi_1(x), ..., \phi_N(x)\}$. Firstly, one can observe that, thanks to the fact that equation (1.11) is required to hold for all v^δ in V^δ, one can consider $v^\delta = \phi_i(x)$ for any $i = 1, ..., N$ and obtain:

$$a(u^\delta, \phi_i(x)) = f(\phi_i(x)), \qquad i = 1, .., N. \tag{1.12}$$

Secondly, since $u^\delta \in V^\delta$, it can be expanded as a linear combination of basis functions as follows:

$$u^\delta(x) = \sum_{j=1}^{N} \hat{u}_j \phi_j(x),$$

and substituted in equation (1.12):

$$a\left(\sum_{j=1}^{N} \hat{u}_j \phi_j(x), \phi_i(x)\right) = f(\phi_i(x)), \qquad i = 1, ..., N. \tag{1.13}$$

Using the bilinearity of $a(\cdot,\cdot)$ one obtains:

$$\sum_{j=1}^{N} \hat{u}_j a\left(\phi_j(x), \phi_i(x)\right) = f(\phi_i(x)), \qquad i = 1, ..., N.$$

Finally, in order to obtain the linear system $A\hat{u} = \text{f}$, one can store the value of each term of the form $a(\phi_j(x), \phi_i(x))$ in the entry (i, j) of the matrix A and the ones of the form $f(\phi_i(x))$ in the i-th position of the vector f:

$$A[i][j] = a(\phi_j(x), \phi_i(x)), \qquad \text{f}[i] = f(\phi_i(x)).$$

This way, each coefficient $\hat{u}_j$ is stored in the j-th position of the vector $\hat{u}$. It is important to observe that, thanks to the bilinearity of $a(\cdot,\cdot)$, the coefficients obtained solving the linear system identify the solution of problem (1.11) even if the only test functions that have been considered were of the form $v^\delta = \phi_i(x)$. In order to prove it, let us consider a generic test function:

$$v^\delta(x) = \sum_{k=1}^{N} \hat{v}_k \phi_k(x),$$

and substitute it into equation (1.11):

$$a\left(u^\delta, \sum_{k=1}^{N} \hat{v}_k \phi_k(x)\right) = f\left(\sum_{k=1}^{N} \hat{v}_k \phi_k(x)\right),$$

that can be expanded as:

$$\begin{aligned}\hat{v}_1 a(u^\delta, \phi_1(x)) + \hat{v}_2 a(u^\delta, \phi_2(x)) + ... + \hat{v}_N a(u^\delta, \phi_N(x)) = &\hat{v}_1 f(\phi_1(x)) + \\ &+ \hat{v}_2 f(\phi_2(x)) + ... + \hat{v}_N f(\phi_N(x)),\end{aligned}$$

and that can be finally rearranged as:

$$\sum_{k=1}^{N} \left[\hat{v}_k \cdot \Big(a(u^\delta, \phi_k(x)) - f(\phi_k(x))\Big)\right] = 0.$$

It can be noted that such a summation is exactly equal zero for any sequence $\{\hat{v}_1, ...\hat{v}_N\}$ due to the fact that $a(u^\delta, \phi_k(x)) - f(\phi_k(x)) = 0 \ \forall k = 1, ..., N$ since u^δ is such that (1.13) holds.
Another key result related to the Galerkin formulation is the Cea's lemma [38]:

Lemma 1.1.2 (Cea's Lemma)**.** *Let V be a Hilbert space and let us consider a bilinear, continuous and coercive form $a(\cdot,\cdot) : V \times V \to \mathbb{R}$ and a continuous linear form $f(\cdot) : V \to \mathbb{R}$. Moreover, let V^δ be a conforming approximation space included in V and α and γ, respectively, the coercivity and the continuity constants associated to $a(\cdot,\cdot)$. Then, the following inequality holds:*

$$\|u - u^\delta\| \leq \frac{\gamma}{\alpha} \inf_{v^\delta \in V^\delta} \|u - v^\delta\|_V.$$

Proof. Let us consider the equation associated to the abstract Galerkin formulation:

$$a(u, v) = f(v), \qquad \forall v \in V, \tag{1.14}$$

and the one associated to its discrete version:

$$a(u^\delta, v^\delta) = f(v^\delta), \qquad \forall v^\delta \in V^\delta. \tag{1.15}$$

One can chooses $v = v^\delta$ (because $V^\delta \subset V$) in equation (1.14) and subtract equation (1.15) from equation (1.14) to obtain:

$$a(u - u^\delta, v^\delta) = 0, \qquad \forall v^\delta \in V^\delta, \tag{1.16}$$

that is known as Galerkin orthogonality. Furthermore, it is important to note that $a(\cdot,\cdot)$ is an inner product on V because such a form is symmetric, coercive and bilinear, and that it is equivalent to the norm $\|\cdot\|_V$. Moreover, the following chain of inequalities holds:

$$\begin{aligned}\alpha\|u-u^\delta\|_V^2 &\leq a(u-u^\delta, u-u^\delta)\\ &= a(u-u^\delta, u-v^\delta+v^\delta-u^\delta)\\ &= a(u-u^\delta, u-v^\delta)+a(u-u^\delta, v^\delta-u^\delta)\\ &= a(u-u^\delta, u-v^\delta)\leq\gamma\|u-u^\delta\|_V\|u-v^\delta\|_V,\end{aligned}$$

where the term $a(u-u^\delta, v^\delta-u^\delta)$ can be deleted because $\left(v^\delta-u^\delta\right)\in V^\delta$ and equation (1.16) holds. The first and the last terms of the chain can be rearranged obtaining:

$$\|u-u^\delta\|\leq\frac{\gamma}{\alpha}\inf_{v^\delta\in V^\delta}\|u-v^\delta\|_V.$$

□

Cea's Lemma is important because, assuming that $\lim_{\delta\to 0}\inf_{v^\delta\in V^\delta}\|v-v^\delta\|_V=0\ \forall v\in V$, one can conclude that

$$\lim_{\delta\to 0}u^\delta=u.$$

Therefore, when the approximation space V^δ approximates V well enough, the approximated solution u^δ can be considered, within a certain tolerance, equal to the continuous one.

1.1.2. Expansion bases

Like in the FEM, also the SEM discrete spaces are obtained by partitioning the entire domain in small subdomains (usually small polytopes with few vertices) where the function can be well approximated by fixed order polynomials. The difference is that the FEM is based on the *h*-type approach: the order of the polynomials is very low, often simply 1 or 2 in each element, and the convergence is obtained refining the mesh (see, for instance, [69]). The opposite approach is referred to as *p*-type method: here the mesh is fixed but the order of the polynomials is increased more and more to achieve the convergence. In particular, if there is only a single element in the entire domain, the *p*-type method can be considered a spectral method [18]. Both the approaches have advantages and disadvantages, therefore, in order to combine them in the best possible way, one can choose the *hp*-type method. The following discussion will be based on the content of the related sections of [42].
Let us consider again the one dimensional setting: in this scenario each element Ω^e is geometrically represented by an interval contained in the entire domain $\Omega=(x_{min},x_{max})$. Let us define N_{el} as the total number of elements, they are required to be non-overlapping and their union has to entirely cover Ω:

$$\Omega=\bigcup_{e=1}^{N_{el}}\Omega^e,\qquad\qquad\Omega^i\cap\Omega^j=\emptyset,\quad\forall i,j:1\leq i,j\leq N_{el}$$

In particular, in one dimension, the domain $\Omega = \{x | x_{min} < x < x_{max}\}$ can be partitioned with a mesh defined by the points:

$$x_{min} = x_0 < x_1 < ... < x_{N_{el}-1} < x_{N_{el}} = x_{max},$$

while the associated elements can be defined as:

$$\Omega^e = \{x | x_{e-1} < x < x_e\}.$$

It is important to observe that the reference interval associated to the standard element Ω^{st} defined as:

$$\Omega^{st} = \{\xi | -1 < \xi < 1\},$$

can be mapped into any arbitrary interval via the mapping defined as:

$$x = \chi^e(\xi) = \frac{(1-\xi)}{2} x_{e-1} + \frac{(1+\xi)}{2} x_e, \qquad \xi \in \Omega^{st}. \tag{1.17}$$

In the reference element the local modes $\phi_i(\xi)$ that characterize the numerical method can be defined in terms of the local coordinate ξ. Then, the local modes can be mapped back to the original element via the mapping (1.17). Furthermore, such a mapping is interesting because it is analytically invertible, the inverse mapping can be expressed as follows:

$$\xi = (\chi^e)^{-1}(x) = 2\frac{(x - x_{e-1})}{(x_e - x_{e-1})} - 1, \qquad x \in \Omega^e.$$

The modes obtained mapping back the local modes of the standard element can be trivially extended with value 0 outside the interval associated to the specific element, this way the global modes $\phi_i^\Omega(x)$ are obtained. Using such global modes it is possible to define the discrete solution as a linear combination of global modes and unknown coefficients as follows:

$$u^\delta(x) = \sum_{i=0}^{N_{dof}-1} \phi_i^\Omega(x)\hat{u}_i = \sum_{e=1}^{N_{el}} \sum_{p=0}^{P} \phi_p^e(\xi)\hat{u}_p^e,$$

where $\phi_p^e(\xi) = \phi_p([\chi^e]^{-1}(x))$, P denotes the polynomial order of the expansion, the subscripts are associated to the element where the mode is different from zero and, finally, N_{dof} is the number of degrees of freedom in global coordinates. The two coefficient vectors $\hat{u}_g = \{\hat{u}_i\}$ and $\hat{u}_l = \{\hat{u}_p^e\}$ are related by the assembly matrix $\mathcal{A}$, that imposes some constraints on the local expansion coefficients because they are more than the global ones. The relation is:

$$\hat{u}_l = \mathcal{A}\hat{u}_g.$$

However, it should be noted that only the degrees of freedom that actively participate to the global expansion modes of different elements have multiple entries in the columns of $\mathcal{A}$, therefore the matrix is very sparse and it should never be explicitly assembled. Instead, it should be substituted by a map that is used for those degrees of freedom with more entries in the columns of the original matrix.

1.1.2.1. Types of expansions

Several different sets of functions can be chosen as local expansion modes [19]. In order to understand why the choice of such a set implies different characteristics of the method, we present three representative types of expansion modes [42]:

$$\begin{aligned}
\phi_p^A(\xi) &= \xi^p && p = 0, ..., P, \\
\phi_p^B(\xi) &= \frac{\prod_{q=0,q\neq p}^{P}(\xi - \xi_q)}{\prod_{q=0,q\neq p}^{P}(\xi_p - \xi_q)} && p = 0, ..., P, \\
\phi_p^C(\xi) &= L_p(\xi) && p = 0, ..., P.
\end{aligned}$$

The expansion set $\{\phi_p^A\}_{p=1}^P$ is called *moment* expansion (because each mode represents a different moment), it is very simple and hierarchical because $\{\phi_p^A\}_{p=1}^P \subset \{\phi_p^A\}_{p=1}^{P+1}$, but leads to very ill-conditioned matrices. It should be noted that the hierarchical sets of expansion modes are useful because they can be easily enriched adding functions to the existing set. On the other hand, a non-hierarchical set is the Lagrange one $\{\phi_p^B\}_{p=1}^P$, it is a *nodal* expansion because the modes are defined via their value in specific points named nodes. The latter is characterized by the important property that $\phi_p^B(\xi_q) = \delta_{pq}$ where δ_{pq} is the Kronecker delta (see [56]). Such a property is very interesting because the values of the discrete solution in correspondence to the nodes are simply the values of the expansion coefficients.
Eventually, the polynomials $L_p(\xi)$ are defined as the Legendre polynomials [42]. This is a *modal* and hierarchical expansion with the interesting property that these polynomials are orthogonal according to the Legendre inner product:

$$(L_p(\xi), L_q(\xi)) = \int_{-1}^{1} L_p(\xi) L_q(\xi) dx = \left(\frac{2}{2p+1}\right) \delta_{pq}.$$

Such a property is important because it can be used to obtain very sparse matrices that are very easy to assemble and the system associated to them are efficiently solved.
It should be observed that, even if a modal basis is chosen, it can always be transformed into a nodal one using the properties of the Lagrange polynomials. In fact, they are very useful to interpolate any smooth function and, if such a function is a polynomial which order is not greater than the number of involved nodes, the interpolation polynomial is exactly the original one.

1.1.2.2. Numerical integration

In order to obtain the discrete linear system from the continuous weak problem one has to be able to numerically compute the involved integrals. Their evaluation, called *quadrature* [23], approximates the integral with a linear combination as follows:

$$\int_{-1}^{1} u(\xi) d\xi \approx \sum_{i=0}^{Q-1} w_i u(\xi_i), \tag{1.18}$$

where Q is the number of quadrature points and w_i some constant weights. The number of weights and their values define the quadrature formula and its order, that is the maximum order of the exactly integrated polynomials. There are several types of numerical integration rules but, in this work, we will focus on the Gaussian quadrature because it is very accurate with high order polynomials and it is widely used in the SEM context. See [57] for a more comprehensive explanation about such type of quadrature rules and a possible extension to arbitrary polygons. Let us assume that the integrand function $u(\xi)$ is smooth enough, using the Lagrange polynomials $h_i(\xi)$ it can be written as:

$$u(\xi) = \sum_{i=0}^{Q-1} u(\xi_i) h_i(\xi) + \varepsilon(u), \tag{1.19}$$

where $\varepsilon(u)$ is the difference between the exact function $u(\xi)$ and its polynomial approximation $\sum_{i=0}^{Q-1} u(\xi_i) h_i(\xi)$.
Substituting expression (1.19) into (1.18) one can obtain expand the integral in the following way:

$$\int_{-1}^{1} u(\xi) d\xi = \sum_{i=0}^{Q-1} w_i u(\xi_i) + R(u), \tag{1.20}$$

where

$$w_i = \int_{-1}^{1} h_i(\xi) d\xi,$$
$$R(u) = \int_{-1}^{1} \varepsilon(u) d\xi.$$

It can be observed that the formula (1.18) is exact only if, when written as in equation (1.20), $R(u) = 0$, this happens whenever $u(\xi)$ is a polynomial of order less than or equal to $Q-1$: $u(\xi) \in \mathcal{P}_{Q-1}([-1,1])$. However, if one properly chooses the locations of the abscissae ξ_i, it is possible to obtain a formula of order $2Q-1$, named Gaussian quadrature rule [72]. Similar formulas can be used to numerically evaluate integrals that include the parametric weight $(1-\xi)^\alpha (1-\xi)^\beta$ as follows:

$$\int_{-1}^{1} (1-\xi)^\alpha (1-\xi)^\beta u(\xi) d\xi.$$

Such integrals are very important in multidimensional problems because the Jacobian of the mapping between an arbitrary element and the reference one can be included in the weight (see section 1.2.2.1). Three types of Gauss quadratures exist, they are called Gauss, Gauss-Radau and Gauss-Lobatto quadrature rules. The first one is associated to abscissae in the inner part of the interval $(-1,1)$, while the abscissae of the second and third one respectively contain only one or both the end points of such an interval. Moreover, it is interesting to observe that the abscissae are associated to the zeros of the P-th order Jacobi polynomial but that their explicit expression is not available: they have to be numerically computed, when needed, at the beginning of a simulation via iterative formulas [56] or they can be read in specific tables. The article [75] describes an algorithm that can be used to efficiently compute the nodes and the

weights involved in Gaussian quadrature formulas.
Finally, it should be noted that the Gauss-Legendre polynomials are also used to obtain the differentiation matrices employing different formulas according to the choice of Gauss quadrature involved.

1.2. Extension to multidimensional problems

In the previous sections we discussed the Galerkin formulation and the most used expansion bases in one dimension. However, the realistic problems are, usually, two-dimensional or three-dimensional, it is therefore compulsory to extend the described topics to use them in multidimensional problems. In this section we will discuss some of the possible generalizations of the one-dimensional expansion bases, then we will describe the main local operations and, finally, the static condensation method. Such a technique is useful to significantly improve the efficiency of the SEM and will be further extended in section 1.3.2. Once more, we remark that the following discussion is based upon [42].

1.2.1. Expansion bases

In this section the main ideas of section 1.1.2 will be extend in multiple dimensions. The standard region can be a quadrilateral or a triangle in two dimensions, while it can be a hexahedron, a prism, a pyramid or a tetrahedron in three. Moreover, the local coordinates will be denoted as ξ_1 and ξ_2 in two dimensions or ξ_1, ξ_2 and ξ_3 in three. Finally, the bases will be named $\phi_{pq}(\xi_1, \xi_2)$ or $\phi_{pqr}(\xi_1, \xi_2, \xi_3)$ respectively in, again, two and three dimensions. Even if several kinds of bases exist, we will focus on the most commonly used in the SEM and that can be described, as follows, in terms of the product of one-dimensional functions:

$$\phi_{pq}(\xi_1, \xi_2) = \psi_p(\xi_1)\psi_q(\xi_2).$$

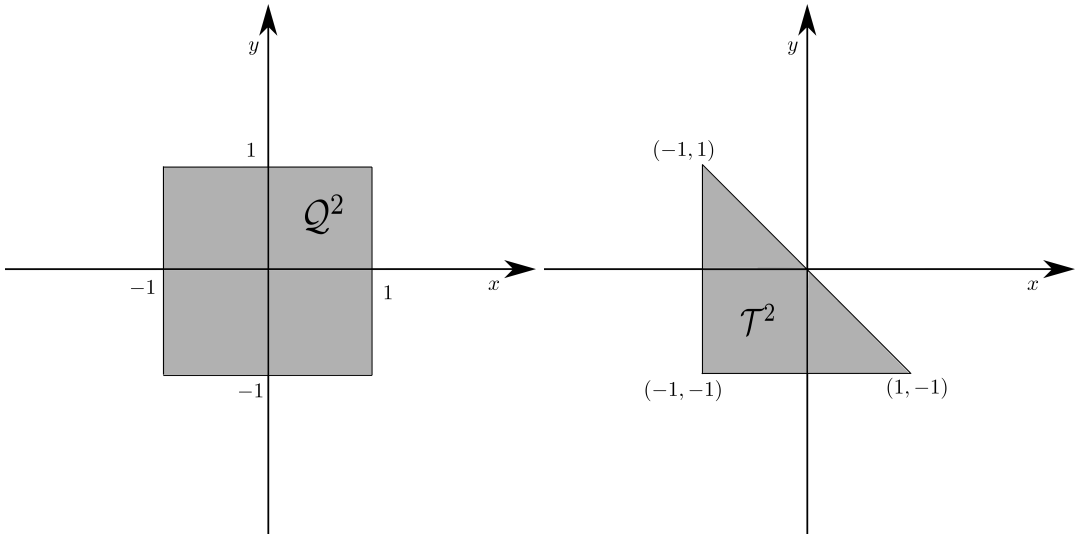
This structure is very useful because, exploiting the sum factorization technique [4], many numerical operations can be performed much more efficiently.
Let us consider a two-dimensional problem and the associated standard region $\Omega^{st} = \mathcal{Q}^2$ defined as follows (see figure 1.1a):

$$\Omega^{st} = \mathcal{Q}^2 = \{-1 \leq \xi_1, \xi_2 \leq 1\}.$$

Since the region is defined as the tensor product of two one-dimensional domains, one can easily generate a basis as the tensor product of two one-dimensional bases, like the ones defined in section 1.1.2.1:

$$\phi_{pq}(\xi_1, \xi_2) = \phi_p(\xi_1)\phi_q(\xi_2), \qquad 0 \leq p, q;\ p \leq P_1;\ q \leq P_2.$$

It can be observed that the order of the polynomials can be different in the two directions, this property is useful when the elements are very long and narrow or the solution develops in a particular direction, such an approach is used to balance the accuracy in both the directions. Moreover, if the involved one-dimensional bases are hierarchical, the same property is guaranteed also for the one defined as their tensor product. Even more flexibility can be achieved if the



(a) Reference square $\mathcal{Q}^2$ (b) Reference triangle $\mathcal{T}^2$

Figure (1.1) Most common reference domains Ω^{st} in two dimensions

boundary/interior decomposition is exploited. It consists in dividing all the expansion basis in two sets: one that includes all the boundary modes and one that includes the internal ones. The former ones are characterized by the fact that they have a unit magnitude at one vertex of the element and are zero at the other ones (vertex modes) or at one node of an edge while they are zero on the other ones (edge modes). In three dimensions the definition can be analogously extended to describe the face modes. The internal modes are all the remaining modes, their value in correspondence to any node on the boundary of the element is always zero. This splitting, that is more intuitive in a basis defined as tensor product, but that can be achieved also in other scenarios, is the key element to ensure the efficiency of the SEM, exploiting the static condensation technique (described in section 1.3.2).
Furthermore, one does not need all the modes associated to the tensor product, but can rely in a smaller set, defined by the union of the modes required to generate an horizontal level of the Pascal's triangle with the edge modes:

$$\mathcal{S}_P = \text{span}\left\{\{\xi_1^i \xi_2^j\}_{(ij)\in\mathcal{P}} \cup \xi_1^P \xi_2 \cup \xi_1 \xi_2^P\right\},$$

$$\mathcal{P} = \{(i,j) | 0 \leq i,j \leq P; i+j \leq P\},$$

where the last two polynomials in the definition of $\mathcal{S}_P$ are needed to enrich a basis that would be suitable for a triangular domain to obtain the space required by a quadrilateral one. Here the subscripts denotes the concerned dimension while the superscripts specify which modes have to be considered within the expansion bases. This space is called *serendipity space* [5] and could be generalized considering both nodal and modal basis together. These results can be simply extended to higher dimensions where the reference domains are cubes or hypercubes.
Unfortunately, when the geometry of the problem is complex, it is better to use unstructured

meshes with triangles or tetrahedrons. Let us focus again on the two-dimensional case, where the reference domain is the following triangle (see figure 1.1b):

$$\mathcal{T}^2 = \{(\xi_1, \xi_2) | -1 \leq \xi_1, \xi_2; \xi_1 + \xi_2 \leq 0\}.$$

In order to exploit again a tensor product structure, it is necessary to consider a different coordinate system through the transformation:

$$\begin{aligned} \eta_1 &= 2\frac{(1+\xi_1)}{(1-\xi_2)} - 1, \\ \eta_2 &= \xi_2, \end{aligned} \tag{1.21}$$

and its inverse:

$$\begin{aligned} \xi_1 &= \frac{(1+\eta_1)(1-\eta_2)}{2} - 1, \\ \xi_2 &= \eta_2. \end{aligned}$$

With this new coordinate system (η_1, η_2) it is possible to redefine $\mathcal{T}^2$ as:

$$\mathcal{T}^2 = \{(\eta_1, \eta_2) | -1 \leq \eta_1, \eta_2 \leq 1\}.$$

It should be noted that such a definition represents the reference square $\mathcal{Q}^2$ in terms of the new coordinate system (η_1, η_2). Therefore, one can observe that the mapping (1.21) transforms the reference triangle into the reference square. However, such a transformation expands a single vertex into an entire edge, while its inverse collapses an entire edge into a single vertex. Because of this, such a coordinate system is also called *collapsed coordinate system*. In three dimensions it is possible to reduce prisms, pyramids and tetrahedrons to hexahedrons in an analogous way but with different mappings, possibly repeatedly applied. Another approach to generate a basis for triangular/tetrahedral elements consists of using the so called *barycentric coordinate system*, that is very useful when the property of rotational symmetry is required [42].

1.2.2. Local operations

1.2.2.1. Integration

As explained in section 1.1.2.2, one has to be able to efficiently and accurately compute the involved integral in order to obtain the linear system that has to be solved. The most trivial extension to the techniques presented in such a section is the numerical integration over the two-dimensional reference domain $\mathcal{Q}^2$:

$$\int_{\mathcal{Q}^2} u(\xi_1, \xi_2) d\xi_1 d\xi_2 = \int_{-1}^{1} \int_{-1}^{1} u(\xi_1, \xi_2) d\xi_1 d\xi_2.$$

Such an integral can be numerically approximated using Q_1 points in the first direction, Q_2 points in the second one and the discussed one-dimensional quadrature rules in both the directions in the following way:

$$\int_{\mathcal{Q}^2} u(\xi_1, \xi_2) d\xi_1 d\xi_2 \approx \sum_{i=0}^{Q_1-1} w_i \left[\sum_{j=0}^{Q_2-1} w_j u(\xi_1^i, \xi_2^j) \right].$$

It should be noted that it is possible to exploit any type of one-dimensional quadrature rule. However, in order to simplify the imposition of the boundary conditions, it is convenient to rely on rules that include both the end points.
In an analogous way, if one wants to integrate over the reference triangle $\mathcal{T}^2$, using the collapsed coordinate system (1.21), one could do it in the following way:

$$\begin{aligned}\int_{\mathcal{T}^2} u(\xi_1,\xi_2)d\xi_1 d\xi_2 &= \int_{-1}^{1}\int_{-1}^{-\xi_2} u(\xi_1,\xi_2)d\xi_1 d\xi_2 \\ &= \int_{-1}^{1}\int_{-1}^{1} u(\eta_1,\eta_2)\left|\frac{\partial(\xi_1,\xi_2)}{\partial(\eta_1,\eta_2)}\right| d\eta_1 d\eta_2 \\ &= \int_{-1}^{1}\int_{-1}^{1} u(\eta_1,\eta_2)\left(\frac{1-\eta_2}{2}\right) d\eta_1 d\eta_2 \\ &\approx \sum_{i=0}^{Q_1-1} w_i \left[\sum_{j=0}^{Q_2-1} w_j u\left(\eta_1^i,\eta_2^j\right)\left(\frac{1-\eta_2^j}{2}\right)\right].\end{aligned} \tag{1.22}$$

However, it is better to use the Gauss-Jacobi quadrature rule:

$$\int_{-1}^{1}(1-\xi)^\alpha(1+\xi)^\beta u(\xi)d\xi = \sum_{i=0}^{Q-1} w^{\alpha,\beta}u(\xi_i^{\alpha,\beta}),$$

because the Jacobian contribution $|\partial(\xi_1,\xi_2)/\partial(\eta_1,\eta_2)|$ can be easily included in the quadrature weights when $\alpha = 1$ and $\beta = 0$. This way, the approximation in the last row of equation (1.22) can be improved with the following quadrature rule:

$$\int_{-1}^{1}\int_{-1}^{1} u(\eta_1,\eta_2)\left(\frac{1-\eta_2}{2}\right) d\eta_1 d\eta_2 = \frac{1}{2}\sum_{i=0}^{Q_1-1} w_i^{0,0}\left[\sum_{j=0}^{Q_2-1} w_j^{1,0} u\left(\eta_1^i,\eta_2^j\right)\right].$$

As in the previous sections, it is possible to generalize these rules to handle reference domains associated with different dimensions and shapes.

1.2.2.2. Differentiation

In this section we will explain how to numerically differentiate a function. Firstly, it should be observed that we will restrict ourselves to differentiation in terms of the local variable ξ, therefore, the functions to be derived has to be approximated via Lagrange polynomials $h_i(\xi)$ but, as we previously remarked, the modal expansions can always be represented in terms of Lagrange polynomials. Since each one-dimensional function can be approximated as:

$$u(\xi) \approx u^\delta(\xi) = \sum_{i=0}^{P} u_i h_i(\xi),$$

the derivative of such an approximation can be expressed as

$$\frac{\partial u^\delta}{\partial \xi}(\xi) = \sum_{i=0}^{P} u_p \frac{\partial h_i}{\partial \xi}(\xi) = \sum_{i=0}^{P} u_i' h_i(\xi),$$

where

$$u_i' = \sum_{j=0}^{P} u_j \frac{\partial h_j}{\partial \xi}(\xi)\bigg|_{\xi_i} .$$

This transformation is very important when non-linear terms are present, for example the advection term in the momentum balance equation:

$$u^\delta(\xi)\frac{\partial u^\delta}{\partial \xi}(\xi), \tag{1.23}$$

can be approximated as:

$$u^\delta(\xi)\frac{\partial u^\delta}{\partial \xi}(\xi) \approx \sum_{i=0}^{P} u_i u_i' h_i(\xi).$$

In fact, if one evaluates (1.23) in $\xi = \xi_k$, one obtains:

$$\begin{aligned} u^\delta(\xi_k)\frac{\partial u^\delta}{\partial \xi_k}(\xi) &= \left(\sum_{i=0}^{P} u_i h_i(\xi_k)\right)\left(\sum_{j=0}^{P} u_j \frac{\partial h_j}{\partial \xi}(\xi_k)\right) \\ &= \left(\sum_{i=0}^{P} u_i h_i(\xi_k)\right)\left(\sum_{j=0}^{P} u_j' h_j(\xi_k)\right) \\ &= \sum_{i=0}^{P} u_i u_i' h_i(\xi), \end{aligned}$$

where, to obtain the last equality, one exploits the fact that $h_i(\xi_k) = \delta_{ik}$ and $h_j(\xi_k) = \delta_{jk}$. It should be observed, however, that if $u^\delta(\xi)$ is a polynomial of order P, the expression (1.23) is represented by a polynomial of order $(2P-1)$ but is approximated by one of order P. Nevertheless, in correspondence to the interpolation nodes, the values of the polynomial of order $(2P-1)$ that interpolates it and the ones of the obtained approximation are the same.
Let us consider the differentiation of a two-dimensional expansion in the reference domain $\mathcal{Q}^2$. As in the one-dimensional case, such an expansion is written in terms of the Lagrange polynomials as follows:

$$u^\delta(\xi_1, \xi_2) = \sum_{i=0}^{P_1}\sum_{j=0}^{P_2} u_{ij} h_i(\xi_1) h_j(\xi_2),$$

with

$$u_{ij} = u^\delta(\xi_1^i, \xi_2^j).$$

Then, it is possible to derive again an expression for its derivative exploiting the fact that the function can be written as the sum of the product of terms that depend by a single coordinate.

The obtained partial derivatives with respect to ξ_1 and ξ_2 are the following ones:

$$\begin{aligned}\frac{\partial u^\delta}{\partial \xi_1}(\xi_1,\xi_2) &= \sum_{i=0}^{P_1}\sum_{j=0}^{P_2} u_{ij}\frac{dh_i(\xi_1)}{d\xi_1}h_j(\xi_2),\\ \frac{\partial u^\delta}{\partial \xi_2}(\xi_1,\xi_2) &= \sum_{i=0}^{P_1}\sum_{j=0}^{P_2} u_{ij}h_i(\xi_1)\frac{dh_j(\xi_2)}{d\xi_2}.\end{aligned} \tag{1.24}$$

Once more, in order to differentiate u^δ on the triangular region $\mathcal{T}^2$, it is convenient to use the Lagrange polynomials with the collapsed coordinates. Let us consider the function:

$$u^\delta(\xi_1,\xi_2) = \sum_{i=0}^{P_1}\sum_{j=0}^{P_2} u_{ij}h_i(\eta_1)h_j(\eta_2),$$

where $u_{ij} = u^\delta(\eta_1^i,\eta_2^j)$ while η_1 and η_2 are defined by the relations (1.21). In order to derive with respect to ξ_1 and ξ_2, one can rely on the chain rule:

$$\begin{pmatrix}\dfrac{\partial}{\partial\xi_1}\\ \dfrac{\partial}{\partial\xi_2}\end{pmatrix} = \begin{pmatrix}\dfrac{2}{(1-\eta_2)}\dfrac{\partial}{\partial\eta_1}\\ 2\dfrac{(1+\eta_1)}{(1-\eta_2)}\dfrac{\partial}{\partial\eta_1}+\dfrac{\partial}{\partial\eta_2}\end{pmatrix}. \tag{1.25}$$

Finally, one can exploit the fact that the definition of $\mathcal{T}^2$ in terms of the collapsed coordinates is identical to the one of $\mathcal{Q}^2$ in terms of the Cartesian ones, this way one can consider equations (1.24) in the variables (η_1,η_2) instead of (ξ_1,ξ_2) and use the same formulas to obtain the derivative of u^δ on a triangle with respect to η_1 and η_2. Finally, one can use the chain rule in equation (1.25) to obtain it in terms of ξ_1 and ξ_2. It is interesting, from the computational point of view, to observe that the denominators in expression (1.25) tend to infinity when $\eta_2 \to 1$. To overcome this problem is useful to consider a Radau-type Gaussian quadrature to avoid consider that specific point.
Similar considerations can obviously be extended to three-dimensional cases in regions with different shapes by exploiting different coordinate systems.

1.2.2.3. Operations on arbitrary elements

In the previous section we discussed the numerical integration and differentiation in the reference domain. However, in general, one has to compute integrals and derivatives in elements with arbitrary shapes, orientations and positions with respect to a globally fixed frame of reference. To handle these scenarios, it is convenient to consider a mapping (different for each element) between the element itself and the reference domain, it will be denoted as:

$$x_1 = \chi_1^e(\xi_1,\xi_2), \qquad x_2 = \chi_2^e(\xi_1,\xi_2). \tag{1.26}$$

For instance, if the element e is geometrically represented by a triangle with straight edges and with vertices, in global coordinates, equal to $\{(x_1^A, x_2^A), (x_1^B, x_2^B), (x_1^C, x_2^C)\}$, and considering C as the collapsed vertex, then the mapping associated to x_1 is the following one:

$$x_1 = \chi_1^e(\eta_1, \eta_2) = x_1^A \frac{(1-\eta_1)(1-\eta_2)}{4} + x_1^B \frac{(1+\eta_1)(1-\eta_2)}{4} + x_1^C \frac{(1+\eta_2)}{2}, \tag{1.27}$$

or, in Cartesian coordinates:

$$x_1 = \chi_1^e(\xi_1, \xi_2) = x_1^A(-\xi_1 - \xi_2) + x_1^B(1+\xi_1) + x_1^C \frac{(1+\xi_2)}{2}. \tag{1.28}$$

Analogously, one can construct a similar mapping for quadrilateral regions with straight edges as follows:

$$\begin{aligned} x_1 = \chi_1(\xi_1, \xi_2) = x_1^A \frac{(1-\xi_1)(1-\xi_2)}{4} + x_1^B \frac{(1+\xi_1)(1-\xi_2)}{4} + \\ + x_1^C \frac{(1+\xi_1)(1+\xi_2)}{4} + x_1^D \frac{(1-\xi_1)(1+\xi_2)}{4}. \end{aligned} \tag{1.29}$$

These mappings are very useful because they allow one to transform complex operations in arbitrary domains into operations that can be locally performed on the reference one. For instance, if one wants to integrate the function $u(x_1, x_2)$ on the arbitrary domain Ω^e, one can exploit the following formula:

$$\int_{\Omega^e} u(x_1, x_2) dx_1 dx_2 = \int_{\Omega^{st}} u(\xi_1, \xi_2) \left| J_{nD} \right| d\xi_1 d\xi_2,$$

where J_{nD} is the n-dimensional Jacobian of the previous mappings. If the domain is two-dimensional and the elements are straight-sided triangles or quadrilaterals, one can evaluate such a quantity computing the derivatives from expressions (1.28) and (1.29):

$$J_{2D} = \begin{vmatrix} \dfrac{\partial x_1}{\partial \xi_1} & \dfrac{\partial x_1}{\partial \xi_2} \\ \dfrac{\partial x_2}{\partial \xi_1} & \dfrac{\partial x_2}{\partial \xi_2} \end{vmatrix} = \frac{\partial x_1}{\partial \xi_1}\frac{\partial x_2}{\partial \xi_2} - \frac{\partial x_1}{\partial \xi_2}\frac{\partial x_2}{\partial \xi_1}. \tag{1.30}$$

Such a Jacobian is very useful also when computing derivatives in arbitrary domains. For instance, let us consider, always in a two-dimensional case for brevity, the chain rule required to compute the gradient:

$$\nabla = \begin{bmatrix} \dfrac{\partial}{\partial x_1} \\ \dfrac{\partial}{\partial x_2} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \xi_1}{\partial x_1}\dfrac{\partial}{\partial \xi_1} & \dfrac{\partial \xi_2}{\partial x_1}\dfrac{\partial}{\partial \xi_2} \\ \dfrac{\partial \xi_1}{\partial x_2}\dfrac{\partial}{\partial \xi_1} & \dfrac{\partial \xi_2}{\partial x_2}\dfrac{\partial}{\partial \xi_2} \end{bmatrix}. \tag{1.31}$$

One still needs an expression for the coefficients in the matrix, to obtain them one can consider the total change of the general function $u(\xi_1, \xi_2)$:

$$du(\xi_1, \xi_2) = \frac{\partial u}{\partial \xi_1} d\xi_1 + \frac{\partial u}{\partial \xi_2} d\xi_2,$$

and, replacing $u(\xi_1, \xi_2)$ by the expressions in equation (1.26), one can obtain the system:

$$\begin{bmatrix} dx_1 \\ \\ dx_2 \end{bmatrix} = \begin{bmatrix} \dfrac{\partial x_1}{\partial \xi_1} & \dfrac{\partial x_1}{\partial \xi_2} \\ \\ \dfrac{\partial x_2}{\partial \xi_1} & \dfrac{\partial x_2}{\partial \xi_2} \end{bmatrix} \begin{bmatrix} d\xi_1 \\ \\ d\xi_2 \end{bmatrix}.$$

Such a system can be inverted to obtain

$$\begin{bmatrix} d\xi_1 \\ \\ d\xi_2 \end{bmatrix} = \frac{1}{J_{2D}} \begin{bmatrix} \dfrac{\partial x_2}{\partial \xi_2} & -\dfrac{\partial x_1}{\partial \xi_2} \\ \\ -\dfrac{\partial x_2}{\partial \xi_1} & \dfrac{\partial x_1}{\partial \xi_1} \end{bmatrix} \begin{bmatrix} dx_1 \\ \\ dx_2 \end{bmatrix}, \tag{1.32}$$

where J_{2D} is the same Jacobian as in expression (1.30). However, since the mapping between the arbitrary element and the standard one is invertible, one can apply the chain rule to ξ_1 and ξ_2 directly, obtaining the following system:

$$\begin{bmatrix} d\xi_1 \\ \\ d\xi_2 \end{bmatrix} = \frac{1}{J_{2D}} \begin{bmatrix} \dfrac{\partial x_1}{\partial \xi_1} & -\dfrac{\partial x_1}{\partial \xi_2} \\ \\ -\dfrac{\partial x_2}{\partial \xi_1} & \dfrac{\partial x_2}{\partial \xi_2} \end{bmatrix} \begin{bmatrix} dx_1 \\ \\ dx_2 \end{bmatrix}. \tag{1.33}$$

Finally, it is possible to equate systems (1.32) and (1.33) to obtain the coefficients needed to compute the gradient in equation (1.31):

$$\frac{\partial \xi_1}{\partial x_1} = \frac{1}{J_{2D}} \frac{\partial x_2}{\partial \xi_2}, \qquad \frac{\partial \xi_1}{\partial x_2} = -\frac{1}{J_{2D}} \frac{\partial x_1}{\partial \xi_2},$$

$$\frac{\partial \xi_2}{\partial x_2} = \frac{1}{J_{2D}} \frac{\partial x_1}{\partial \xi_1}, \qquad \frac{\partial \xi_2}{\partial x_1} = -\frac{1}{J_{2D}} \frac{\partial x_2}{\partial \xi_1}.$$

1.2.3. Global operations

All the described operations can be defined as local because they always involved a single element. However, in order to compute the solution of the Galerkin problem on the entire domain, it is crucial to be able to group together all the local results. Since the elements can be very different from each other, one has to extend the local expansion modes to global functions. Let us consider the arbitrary two-dimensional element Ω^e and a local expansion mode $\phi^e_{pq}(x_1, x_2)$ defined in Ω^e: it can be easily extended to a global expansion mode $\phi^\Omega_{pq}(x_1, x_2)$ defining it as $\phi^\Omega_{pq}(x_1, x_2) = \phi^e_{pq}(x_1, x_2)$ when $(x_1, x_2) \in \Omega^e$, while $\phi^\Omega_{pq}(x_1, x_2) = 0$ otherwise. It can be noted that the extension of a local interior mode is a C^0 continuous function because it is exactly zero on the boundary of the element but the same property does not hold for the local boundary modes. Unfortunately, in order obtain a C^0 continuous solution of the Galerkin problem, it is important to consider only continuous basis functions, therefore one has to properly handle the

boundary modes.
To generate the C^0 continuous basis, one can consider all the interior expansion modes and define again the boundary ones gluing the similar boundary modes together [42]. The definition of similarity of boundary modes can vary according to the type of modes. Let us consider two nodal expansion modes defined in two different elements (the common support can only be a vertex or an edge), they are said to be similar if there is a node, on the common support, where their value is exactly one, while they are zero on the other ones. In an analogous way two modal expansion modes are similar if they have the same polynomial order. However, it should be noted that the modal modes are defined with respect to the specific local coordinate system of their element, it is thus possible that the sign of one of them has to be reversed in order to obtain two matching functions on the common boundary.
Moreover, we remark that this process can become more complex if non-Cartesian coordinate systems are considered or if the problem dimension is higher. In the former case one has to properly manage the specific chosen reference system. For instance, if one uses the collapsed coordinates, it is necessary to correctly orient all the elements in order to obtain consistent coordinate lines in adjacent elements.
In matrix notation, one can define the assembly matrix $\mathcal{A}$ as the matrix the relates the local degrees of freedom to the global ones in the following way:

$$\hat{u}_l = \mathcal{A}\hat{u}_g.$$

Here $\hat{u}_l$ is the vector that contains the coefficients associated to the local degrees of freedom, while $\hat{u}_g$ is the vector of the coefficients associated to the global ones. Denoting as N_{eof} the number of local degrees of freedom and as N_{dof} the number of global ones, the dimension of $\mathcal{A}$ is $N_{eof} \times N_{dof}$. The matrix $\mathcal{A}$ is very sparse, in fact $\mathcal{A}[i][j] \neq 0$ only if the modes associated to the local coefficient $\hat{u}_l[i]$ contributes to the global one $\hat{u}_g[j]$. This way, in each row there is exactly one non-zero entry, while multiple values different than zero can be present in the same column because the global boundary modes are generated by more local modes. Finally, it can be observed that if the expansion modes are of nodal type, then $\mathcal{A}$ can contain only 0 or 1, while, if the modes are modal, the operations of reversing the sign of a mode are represented by means of the value -1.

1.2.3.1. Static condensation method

In section 1.2.2 all the techniques that are needed to construct the local matrices (associated to the problem but restricted to the degrees of freedom within a single element) have been discussed, while in section 1.2.3 the assembly matrix $\mathcal{A}$ has been presented. The latter is useful to transform the system obtained assembling all the local matrices, where the unknowns are expressed in local coordinates, into a new system with global unknowns. Let us assume, in this section, that one wants to solve the following linear systems:

$$Mu = \mathrm{f}, \tag{1.34}$$

where u is the vector of the coefficients associated to the global degrees of freedom. Let us consider the following notation: N_{bnd} is the number of global degrees of freedom associated to

boundary modes, N_{int} the number of global degrees of freedom associated to internal modes, and N_{tot} can thus be defined as $N_{tot} = N_{bnd} + N_{int}$. It is important to highlight that, thanks to the assembly matrix $\mathcal{A}$, one can sort the global degrees of freedom, grouped by elements, in order to have all the unknowns related to boundary modes in the first N_{bnd} positions of the vector u, while considering all the unknowns associated to internal modes in the last N_{int} positions. This way, the different objects in equation (1.34) can be structured as follows:

$$M = \begin{bmatrix} M_b & M_c \\ M_c^T & M_i \end{bmatrix}, \qquad u = \begin{bmatrix} u_b \\ u_i \end{bmatrix}, \qquad \mathrm{f} = \begin{bmatrix} \mathrm{f}_b \\ \mathrm{f}_i \end{bmatrix}. \tag{1.35}$$

Here u and f are simply organized in boundary and interior components, while M is divided in four different blocks. The first one is M_b, it belongs to $\mathbb{R}^{N_{bnd} \times N_{bnd}}$ and each entry of it represents the interaction between two boundary modes. In an analogous way, $M_i \in \mathbb{R}^{N_{int} \times N_{int}}$ contains the information about the interaction between the interior modes. Finally, $M_c \in \mathbb{R}^{N_{bnd} \times N_{int}}$ accounts for the coupling between the boundary and the interior modes, while M_c^T is its transpose. It should be observed that the boundary-boundary matrix M_b is sparse and its bandwidth can be reduced exploiting existent algorithms [27] or heuristics [48], while the boundary-interior matrix M_c is a very sparse block diagonal matrix and it is easily generated by the corresponding terms of the local matrices. Moreover, the interior-interior matrix M_i is block diagonal because the interior modes of an element are orthogonal to the interior modes in other elements. Therefore, it is composed by N_{el} blocks (where N_{el} is the number of elements) that are very small because they contain only the interior degrees of freedom of a single element; such a property is the key factor to efficiently solve system (1.34) using the *static condensation method* [42].
Let us consider again system (1.34) with the structure described in (1.35):

$$\begin{bmatrix} M_b & M_c \\ M_c^T & M_i \end{bmatrix} \begin{bmatrix} u_b \\ u_i \end{bmatrix} = \begin{bmatrix} \mathrm{f}_b \\ \mathrm{f}_i \end{bmatrix}. \tag{1.36}$$

One can pre-multiply such a system by the matrix:

$$\begin{bmatrix} I & -M_c M_i^{-1} \\ 0 & I \end{bmatrix},$$

obtaining the following linear system:

$$\begin{bmatrix} M_b - M_c M_i^{-1} M_c^T & 0 \\ M_c^T & M_i \end{bmatrix} \begin{bmatrix} u_b \\ u_i \end{bmatrix} = \begin{bmatrix} \mathrm{f}_b - M_c M_i^{-1} \mathrm{f}_i \\ \mathrm{f}_i \end{bmatrix}. \tag{1.37}$$

Therefore, it is possible to decouple the boundary unknowns from the interior ones. The equation associated to the boundary unknowns is the following one:

$$\left(M_b - M_c M_i^{-1} M_c^T\right) u_b = \mathrm{f}_b - M_c M_i^{-1} \mathrm{f}_i. \tag{1.38}$$

After having solved equation (1.38), it is possible to exploit the second row of system (1.37) to easily obtain u_i as:

$$u_i = M_i^{-1}\mathrm{f}_i - M_i^{-1}M_c^T u_b.$$

It should be noted that, even if from the mathematical point of view it is always possible to solve a linear system with the described method, from the computational one the method is useful only when it is efficient. Such an efficiency can be ensured if a specific structure of the submatrices can be exploited. However, as noted above, M_i and M_c are two block diagonal matrices with the blocks associated to the different elements. One can take advantage of such a structure to efficiently invert M_i, in fact M_i^{-1} is a block diagonal matrix with each block equal to the inverse of the corresponding block of M_i (each submatrix can be locally inverted), moreover, all the operations that do not involve the matrix M_b can be computed locally. This is very important because the local operations can be performed in parallel, significantly increasing the efficiency of the method. Finally, it can be noted that the only system that has to be solved is the one in equation (1.38), which dimension is only N_{bnd}.
It should be noted that the described method is useful to efficiently solve systems characterized by the structure described in equation (1.36), however, it can be improved pairing it with other methods as in [33], where a multigrid approach has been used. Finally, it is possible to use the static condensation method to enhance the efficiency of the reduced basis method [40].

1.3. Application to incompressible flows

As written in section 1.1.1, the spectral element method is based on the Galerkin formulation. Therefore, in this section we will discuss, following [42] in the first part of it, the weak formulation of the Navier-Stokes equations, the required functional spaces and some techniques to efficiently compute a discrete solution of such a problem.
Let us consider an incompressible and isothermal flow, characterized by a constant viscosity ν and a constant density ρ, coupled with proper boundary conditions on the boundary of the domain Ω. The problem is governed by the incompressible Navier-Stokes equations:

$$\begin{cases} \nabla \cdot \mathbf{u} = 0 \\ \dfrac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} - \nu \Delta \mathbf{u} + \nabla p = 0 \end{cases}, \tag{1.39}$$

where the pressure p has been redefined, as in section A.1, normalizing it over the constant density. In order to obtain a well-posed Galerkin problem, one has to firstly select the proper functional spaces for the velocity $\mathbf{u}$ and for the pressure p. They can be chosen according to the higher spatial derivatives in system (1.39). Therefore, in order to study the property of the Galerkin problem, system (1.39) is equivalent to the steady Stokes problem (1.40). In fact, in both the problems, the highest spatial derivatives are represented by the terms $\Delta\mathbf{u}$ and ∇p and, consequently, the functional spaces required by the weak Navier-Stokes problem coincide with the ones of the Stokes problem.

$$\begin{cases} \nabla \cdot \mathbf{u} = 0 \\ -\nu \Delta \mathbf{u} + \nabla p = \mathbf{f} \end{cases}, \tag{1.40}$$

here f is a forcing term that can include all the remaining terms of equation (1.39) when explicitly treated. Recalling that the Dirichlet boundary conditions can be imposed via a lifting function, the appropriate functional spaces, respectively for the velocity and for the pressure, are the following ones:

$$H_0^1(\Omega) = \{\mathbf{w} \in H^1(\Omega) | \mathbf{w} = 0 \text{ on } \partial\Omega_D\},$$
$$L_0^2(\Omega) = \{q \in L^2(\Omega) | \int_\Omega q dx = 0\},$$

where $\partial\Omega_D$ represents the portion of the boundary of Ω associated to Dirichlet boundary conditions.
In order to obtain the weak formulation of problem (1.40), one multiplies each equation by proper test functions and integrates over the entire domain obtaining:

$$\begin{cases} \int_\Omega (\nabla \cdot \mathbf{u}) q = 0, & \forall q \in L_0^2(\Omega) \\ \int_\Omega (-\nu \Delta \mathbf{u} + \nabla p) \cdot \mathbf{w} = \int_\Omega \mathbf{f} \cdot \mathbf{w}, & \forall \mathbf{w} \in H_0^1(\Omega) \end{cases}. \tag{1.41}$$

If we respectively denote $L_0^2(\Omega)$ and $H_0^1(\Omega)$ as Q and V, we can name as $Q^\delta \subset Q$ and $V^\delta \subset V$ the discrete pressure and velocity spaces. They are defined using a polynomial base inside each element but, unfortunately, their order has to be properly chosen in order to satisfy the so called *inf-sup condition*. In order to study such a condition, let us consider the following notation:

$$\begin{aligned} a(\mathbf{v}, \mathbf{w}) &= \nu \int_\Omega \nabla \mathbf{v} \cdot \nabla \mathbf{w} = \nu \, (\nabla \mathbf{v}, \nabla \mathbf{w}), \\ b(\mathbf{w}, q) &= -\int_\Omega (\nabla \cdot \mathbf{w}) q = -(\nabla \cdot \mathbf{w}, q), \\ f(\mathbf{w}) &= \int_\Omega \mathbf{f} \cdot \mathbf{w} = (\mathbf{f}, \mathbf{w}). \end{aligned} \tag{1.42}$$

This way it is possible to reformulate problem (1.41) in terms of the operators $a(\cdot, \cdot) : V \times V \to \mathbb{R}$ and $b(\cdot, \cdot) : V \times Q \to \mathbb{R}$ for the discrete solution $(\mathbf{u}^\delta, p^\delta) \in V^\delta \times Q^\delta$ as follows:

$$\begin{cases} a(\mathbf{v}, \mathbf{u}^\delta) + b(\mathbf{v}, p^\delta) = f(\mathbf{v}), & \forall \mathbf{v} \in V^\delta \\ b(\mathbf{u}^\delta, q) = 0, & \forall q \in Q^\delta \end{cases}. \tag{1.43}$$

Assuming that all the integrals are exactly computed thanks to proper Gaussian quadrature rules, then the solution of problem (1.43) exists and is unique only if there exists a constant $\beta_\delta > 0$ such that:

$$\sup_{\mathbf{v} \in V^\delta} \frac{b(\mathbf{v}, q)}{\|\mathbf{v}\|_V} \geq \beta_\delta \|q\|_Q, \qquad \forall q \in Q^\delta. \tag{1.44}$$

Such a condition is called *inf-sup condition* and it will be more deeply investigated, together with the Petrov-Galerkin formulation, in section 2.4.1. It is very important because, if it is not satisfied, spurious pressure modes can exist, preventing the solution to be unique, therefore, the

problem can not be numerically solved because it is ill-posed. A spurious pressure mode is a pressure field q^* such that:

$$b(\mathbf{v}, q^*) = 0, \qquad \forall \mathbf{v} \in V^\delta.$$

It should be noted that, if a spurious mode q^* exists and if $(\mathbf{u}^\delta, p^\delta)$ is a solution of problem (1.43), then $(\mathbf{u}^\delta, p^\delta + cq^*)$ is another solution for any $c \in \mathbb{R}$. This way, the solution $(\mathbf{u}^\delta, p^\delta)$ is not unique anymore.
However, when the solution is unique, the following estimates hold:

$$\|\mathbf{u}^\delta\|_V \leq \frac{\gamma}{\alpha_\delta}\|f\|_{V^{\delta\prime}},$$
$$\|p^\delta\|_Q \leq \frac{1}{\beta_\delta}\left(1 + \frac{\gamma^2}{\alpha_\delta}\right)\|f\|_{V^{\delta\prime}},$$

where $V^{\delta\prime}$ is the dual space of V^δ, γ is the continuity constant associated to the operator $a(\cdot,\cdot)$:

$$|a(\mathbf{v}, \mathbf{w})| \leq \gamma\|\mathbf{v}\|_V\|\mathbf{w}\|_V,$$

and α_δ is its coercivity constant:

$$a(\mathbf{v}, \mathbf{v}) \geq \alpha_\delta\|\mathbf{v}\|_V^2.$$

It is important to observe that β_δ may depend on the involved discretization, both on the mesh and on the polynomial order associated to the discrete solutions. Unfortunately, it might happen that, even if for a given type of discretization $\beta_\delta > 0$ exists, it is possible that β_δ tends to zero when the number of elements increases, their average size decreases or the polynomial order increases. One of the possible approaches to overcome this problem is to reduce the dimension of the space Q^δ. An example is given by the $\mathcal{P}_P(\Omega^e)/\mathcal{P}_{P-2}(\Omega^e)$ formulation (see [51] and [52]), where the velocity is approximated with functions that can be represented by polynomials of order P in each element, while the order of the polynomials associated to the pressure is only $P-2$. In this work we will consider only the $\mathcal{P}_P(\Omega^e)/\mathcal{P}_{P-2}(\Omega^e)$ formulation since it leads to inf-sup stable problems.

1.3.1. Iterative methods in CFD

Even if the Stokes equations allow one to easily obtain solutions that are very close to the solutions of the complete Navier-Stokes system when the Reynolds number is very low, such an approximation is worse and worse when Re increases. On the other hand, the Euler equations presented in section A.2 can be used only when Re is very high. Therefore, when Re does not allow one to use these two approximations, one is obliged to solve the complete Navier-Stokes system.
Since in this work we are interested in steady and incompressible solutions, let us consider the steady Navier-Stokes equations with the divergence-free constraint:

$$\begin{cases} \nabla \cdot \mathbf{u} = 0 \\ \mathbf{u} \cdot \nabla\mathbf{u} - \nu\Delta\mathbf{u} + \nabla p = 0 \end{cases}. \tag{1.45}$$

Since only linear problems can be solved, it is necessary to linearize the non-linear term $\mathbf{u} \cdot \nabla \mathbf{u}$ in system (1.45) and solve the non-linear system with an iterative approach. The two most used schemes are the Picard iteration and the Newton one (see [16] for a description of these and other methods).
Let us consider first the Picard iteration method. Here one assumes that, if the iterative solver is converging to a solution $\widetilde{\mathbf{u}}$, the iterations $\mathbf{u}^k$ and $\mathbf{u}^{k+1}$ have to be closer and closer, this way $\mathbf{u}^{k+1}$ can be approximated with $\mathbf{u}^k$. Therefore, the non-linear term can be approximated in the following way:

$$\mathbf{u}^{k+1} \cdot \nabla \mathbf{u}^{k+1} \approx \mathbf{u}^k \cdot \nabla \mathbf{u}^{k+1},$$

where $\mathbf{u}^k$ is the velocity obtained in the last iteration and it is known, while $\mathbf{u}^{k+1}$ is the unknown solution. After the linearization, during each iteration of the Picard method, one has to solve the following Oseen problem:

$$\begin{cases} \nabla \cdot \mathbf{u}^{k+1} = 0 \\ \mathbf{u}^k \cdot \nabla \mathbf{u}^{k+1} - \nu \Delta \mathbf{u}^{k+1} + \nabla p^{k+1} = 0 \end{cases}. \tag{1.46}$$

On the other hand, one has to consider the following expansion to derive the Newton iterations:

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \delta \mathbf{u}^k, \tag{1.47}$$

where $\mathbf{u}^k$ is again the velocity field obtained in the k-th iteration, while $\delta \mathbf{u}^k$ is a correction required to obtain the subsequent velocity field from $\mathbf{u}^k$. Substituting equation (1.47) into the non-linear term, one can write it as:

$$\begin{aligned} \mathbf{u}^{k+1} \cdot \nabla \mathbf{u}^{k+1} &= \left(\mathbf{u}^k + \delta \mathbf{u}^k\right) \cdot \nabla \left(\mathbf{u}^k + \delta \mathbf{u}^k\right) \\ &= \left[\mathbf{u}^k + \left(\mathbf{u}^{k+1} - \mathbf{u}^k\right)\right] \cdot \nabla \left(\mathbf{u}^k + \delta \mathbf{u}^k\right) \\ &= \mathbf{u}^k \cdot \nabla \left(\mathbf{u}^k + \delta \mathbf{u}^k\right) + \left(\mathbf{u}^{k+1} - \mathbf{u}^k\right) \cdot \nabla \left(\mathbf{u}^k + \delta \mathbf{u}^k\right) \\ &= \mathbf{u}^k \cdot \nabla \mathbf{u}^{k+1} + \left(\mathbf{u}^{k+1} - \mathbf{u}^k\right) \cdot \nabla \left(\mathbf{u}^k + \delta \mathbf{u}^k\right) \\ &= \mathbf{u}^k \cdot \nabla \mathbf{u}^{k+1} + \mathbf{u}^{k+1} \cdot \nabla \mathbf{u}^k - \mathbf{u}^k \cdot \nabla \mathbf{u}^k + \mathbf{u}^{k+1} \cdot \nabla \delta \mathbf{u}^k - \mathbf{u}^k \cdot \nabla \delta \mathbf{u}^k \\ &= \mathbf{u}^k \cdot \nabla \mathbf{u}^{k+1} + \mathbf{u}^{k+1} \cdot \nabla \mathbf{u}^k - \mathbf{u}^k \cdot \nabla \mathbf{u}^k + \delta \mathbf{u}^k \cdot \nabla \delta \mathbf{u}^k. \end{aligned}$$

Assuming that the iterative solver is converging, the approximation $\mathbf{u}^{k+1} \approx \mathbf{u}^k$ holds and, therefore, $\delta \mathbf{u}^k \approx 0$. As a result, one can neglect the term $\delta \mathbf{u}^k \cdot \nabla \delta \mathbf{u}^k$ and substitute the obtained approximation of the non-linear term into equation (1.45) in the following way:

$$\begin{cases} \nabla \cdot \mathbf{u}^{k+1} = 0 \\ \mathbf{u}^{k+1} \cdot \nabla \mathbf{u}^k + \mathbf{u}^k \cdot \nabla \mathbf{u}^{k+1} - \nu \Delta \mathbf{u}^{k+1} + \nabla p^{k+1} = \mathbf{u}^k \cdot \nabla \mathbf{u}^k \end{cases}. \tag{1.48}$$

It should be noted that both methods rely on a first velocity field $\mathbf{u}^0$, that can be obtained solving the Stokes problem, that is equivalent to the two methods when $\mathbf{u}^0 = 0$, but they do not need an initial guess for the pressure. Moreover, it is important to highlight that the Picard

iteration method is very stable, in fact it can converge even with very bad initial guesses, but the convergence is only linear. Such a slow converge could be a problem if one wants to compute a solution very close to a bifurcation point because the number of required iterations may increase too much, heavily slowing down the computation. On the other hand, the Newton method is much faster because it is characterized by a quadratic convergence, but it needs a very good initial guess. If a good initial guess is not available, an iterative solver based on the Newton method may diverge. Therefore, sometimes it is better to use some Picard iterations before the Newton ones in order to compute a good enough velocity field for the first Newton step. Such a technique will be used and generalized in section 4.2.1.2 to improve the stability of the deflation method (see section 3.3).

1.3.2. Static condensation method in CFD

In this section the static condensation method discussed in section 1.2.3.1 will be more deeply explained, focusing on the matrices that arise from the Galerkin formulation of the incompressible Navier-Stokes equations. We remark that, since the used methods are implemented in Nektar++, a more comprehensive description of the static condensation method applied to the Navier-Stokes equations and of other techniques can be found in [78], that is the reference followed to develop the following discussion.
Once again, let us consider the incompressible Navier-Stokes system:

$$\begin{cases} \nabla \cdot \mathbf{u} = 0 \\ \mathbf{u} \cdot \nabla \mathbf{u} - \nu \Delta \mathbf{u} + \nabla p = 0 \end{cases}. \tag{1.49}$$

Multiplying by the proper test functions and integrating over the entire domain one can obtain its weak formulation. Moreover, let us consider the notation (1.42) and denote the non-linear operator associated to the term $\mathbf{u} \cdot \nabla \mathbf{u}$ as:

$$c(\mathbf{u}, \mathbf{v}, \mathbf{w}) = \int_\Omega (\mathbf{v} \cdot \nabla \mathbf{w})\, \mathbf{u}.$$

Therefore, the Galerkin formulation can be written as:

Find $(\mathbf{u}, p) \in V \times Q$ such that:

$$\begin{cases} c(\mathbf{v}, \mathbf{u}, \mathbf{u}) + a(\mathbf{v}, \mathbf{u}) + b(\mathbf{v}, p) = f(\mathbf{v}), & \forall \mathbf{v} \in V \\ b(\mathbf{u}, q) = 0, & \forall q \in Q \end{cases}. \tag{1.50}$$

System (1.50) can be linearized according to the Picard iteration:

$$\begin{cases} c(\mathbf{v}, \mathbf{u}^k, \mathbf{u}^{k+1}) + a(\mathbf{v}, \mathbf{u}^{k+1}) + b(\mathbf{v}, p^{k+1}) = f(\mathbf{v}), & \forall \mathbf{v} \in V \\ b(\mathbf{u}^{k+1}, q) = 0, & \forall q \in Q \end{cases}, \tag{1.51}$$

or to the Newton one:

$$\begin{cases} c(\mathbf{v},\mathbf{u}^{k+1},\mathbf{u}^{k})+c(\mathbf{v},\mathbf{u}^{k},\mathbf{u}^{k+1})+a(\mathbf{v},\mathbf{u}^{k+1})+b(\mathbf{v},p^{k+1})=f(\mathbf{v})+c(\mathbf{v},\mathbf{u}^{k},\mathbf{u}^{k}), & \forall \mathbf{v}\in V \\ b(\mathbf{u}^{k+1},q)=0, & \forall q\in Q \end{cases}. \tag{1.52}$$

Finally, systems (1.51) and (1.52) can be discretized and the unknown coefficients can be sorted in order to have the ones related to the velocity boundary modes $\mathbf{u}_{bnd}$ at the beginning of the vector, the ones associated to the velocity interior modes $\mathbf{u}_{int}$ at its end, and the pressure coefficients between $\mathbf{u}_{bnd}$ and $\mathbf{u}_{int}$. The obtained linear system [36] can be written as:

$$\begin{bmatrix} A & -D_{bnd}^{T} & B \\ -D_{bnd} & 0 & -D_{int} \\ \widetilde{B}^{T} & -D_{int}^{T} & C \end{bmatrix} \begin{bmatrix} \mathbf{u}_{bnd} \\ p \\ \mathbf{u}_{int} \end{bmatrix} = \begin{bmatrix} \mathrm{f}_{bnd} \\ 0 \\ \mathrm{f}_{int} \end{bmatrix}, \tag{1.53}$$

where the right hand side has been sorted according to the unknown vector and the following notation has been used:

$$\begin{aligned} A[i][j] &= c(\phi_{bnd}^{j},\phi_{bnd}^{i},\mathbf{u}^{k})+c(\phi_{bnd}^{j},\mathbf{u}^{k},\phi_{bnd}^{i})+a(\phi_{bnd}^{j},\phi_{bnd}^{i}),\\ B[i][j] &= c(\phi_{bnd}^{j},\phi_{int}^{i},\mathbf{u}^{k})+c(\phi_{bnd}^{j},\mathbf{u}^{k},\phi_{int}^{i})+a(\phi_{bnd}^{j},\phi_{int}^{i}),\\ \widetilde{B}^{T}[i][j] &= c(\phi_{int}^{j},\phi_{bnd}^{i},\mathbf{u}^{k})+c(\phi_{int}^{j},\mathbf{u}^{k},\phi_{bnd}^{i})+a(\phi_{int}^{j},\phi_{bnd}^{i}),\\ C[i][j] &= c(\phi_{int}^{j},\phi_{int}^{i},\mathbf{u}^{k})+c(\phi_{int}^{j},\mathbf{u}^{k},\phi_{int}^{i})+a(\phi_{int}^{j},\phi_{int}^{i}),\\ D_{bnd}[i][j] &= b(\phi_{bnd}^{j},\psi^{i}),\\ D_{int}[i][j] &= b(\phi_{int}^{j},\psi^{i}),\\ \mathrm{f}_{bnd}[i] &= f(\phi_{bnd}^{i})+c(\phi_{bnd}^{i},\mathbf{u}^{k},\mathbf{u}^{k}),\\ \mathrm{f}_{int}[i] &= f(\phi_{int}^{i})+c(\phi_{int}^{i},\mathbf{u}^{k},\mathbf{u}^{k}). \end{aligned}$$

Here ϕ_{bnd}^{m} represents the m-th global velocity boundary mode, ϕ_{int}^{m} the m-th global velocity interior mode and ψ^{m} the m-th global pressure mode. Moreover, it should be noted that the terms of the form $c(\phi_{\cdot},\phi_{\cdot},\mathbf{u}^{k})$ or $c(\phi_{\cdot},\mathbf{u}^{k},\mathbf{u}^{k})$ are different from zero only if the Newton linearization has been used.

As in section 1.2.3.1, we highlight that the matrix C is block diagonal because it represents the interaction between the velocity interior modes that are orthogonal to the modes in other elements, it is therefore efficient to explicitly invert it and premultiply system (1.53) as follows:

$$\begin{bmatrix} I & 0 & -BC^{-1} \\ 0 & I & D_{int}C^{-1} \\ 0 & 0 & I \end{bmatrix} \left\{ \begin{bmatrix} A & -D_{bnd}^{T} & B \\ -D_{bnd} & 0 & -D_{int} \\ \widetilde{B}^{T} & -D_{int}^{T} & C \end{bmatrix} \begin{bmatrix} \mathbf{u}_{bnd} \\ p \\ \mathbf{u}_{int} \end{bmatrix} = \begin{bmatrix} \mathrm{f}_{bnd} \\ 0 \\ \mathrm{f}_{int} \end{bmatrix} \right\},$$

where I is the identity matrix, obtaining the following partially decoupled system:

$$\begin{bmatrix} A - BC^{-1}\widetilde{B}^T & -D_{bnd}^T + BC^{-1}D_{int}^T & 0 \\ -D_{bnd} + D_{int}C^{-1}\widetilde{B}^T & -D_{int}C^{-1}D_{int}^T & 0 \\ \widetilde{B}^T & -D_{int}^T & C \end{bmatrix} \begin{bmatrix} \mathbf{u}_{bnd} \\ p \\ \mathbf{u}_{int} \end{bmatrix} = \begin{bmatrix} \mathrm{f}_{bnd} - BC^{-1}\mathrm{f}_{int} \\ D_{int}C^{-1}\mathrm{f}_{int} \\ \mathrm{f}_{int} \end{bmatrix}. \quad (1.54)$$

Moreover, it is possible to perform a second level of static condensation on the 2×2 block that relates the velocity boundary modes and the pressure ones. To do so, one can split the boundary degrees of freedom into the mean pressure p_0 (or a degree of freedom that can represent it) and the remaining degrees of freedom $\hat{p}$. The mean pressure is then grouped with the velocity boundary coefficients in the new variable $\mathbf{b} = [\mathbf{u}_{bnd}, p_0]$ and the first two equations of system (1.54) can therefore be expressed as:

$$\begin{bmatrix} \hat{A} & \hat{B} \\ \hat{C} & \hat{D} \end{bmatrix} \begin{bmatrix} \mathbf{b} \\ \hat{p} \end{bmatrix} = \begin{bmatrix} \hat{\mathrm{f}}_{bnd} \\ \hat{\mathrm{f}}_p \end{bmatrix}, \quad (1.55)$$

where

$$\hat{A} = \begin{bmatrix} A - BC^{-1}\hat{B}^T & \left(-D_{bnd}^T + BC^{-1}D_{int}^T\right)_{\{(i,0)\}} \\ \left(-D_{bnd} + D_{int}C^{-1}\widetilde{B}^T\right)_{\{(0,j)\}} & -\left(D_{int}C^{-1}D_{int}^T\right)_{(0,0)} \end{bmatrix},$$

$$\hat{B} = \begin{bmatrix} \left(-D_{bnd}^T + BC^{-1}D_{int}^T\right)_{\{(i,j):j\neq 0\}} \\ -\left(D_{int}C^{-1}D_{int}^T\right)_{\{(0,j)\}} \end{bmatrix},$$

$$\hat{C} = \begin{bmatrix} -D_{bnd} + D_{int}C^{-1}\widetilde{B}^T & -\left(D_{int}C^{-1}D_{int}^T\right)_{\{(i,0):i\neq 0\}} \end{bmatrix},$$

$$\hat{D} = \begin{bmatrix} -\left(D_{int}C^{-1}D_{int}^T\right)_{\{(i,j):i\neq 0, j\neq 0\}} \end{bmatrix},$$

$$\hat{\mathrm{f}}_{bnd} = \begin{bmatrix} \mathrm{f}_{bnd} - BC^{-1}\mathrm{f}_{int} \\ -\left(D_{int}C^{-1}\mathrm{f}_{int}\right)_0 \end{bmatrix},$$

$$\hat{\mathrm{f}}_p = \begin{bmatrix} \left(D_{int}C^{-1}\mathrm{f}_{int}\right)_{\{i:i\neq 0\}} \end{bmatrix}.$$

Here we assumed that the first element in p was p_0, but the same matrix decomposition can be obtained properly sorting the pressure coefficients. Therefore, the subscripts 0 in the submatrices represent the row or the column associated to the mean pressure.
Moreover, it can be observed that $\hat{D}$ is block diagonal and it is thus possible to efficiently invert it to achieve the second level of the static condensation premultiplying system (1.55) by the proper matrix in the following way:

$$\begin{bmatrix} I & -\hat{B}\hat{D}^{-1} \\ 0 & I \end{bmatrix} \left\{ \begin{bmatrix} \hat{A} & \hat{B} \\ \hat{C} & \hat{D} \end{bmatrix} \begin{bmatrix} \mathbf{b} \\ \hat{p} \end{bmatrix} = \begin{bmatrix} \hat{\mathrm{f}}_{bnd} \\ \hat{\mathrm{f}}_p \end{bmatrix} \right\}. \quad (1.56)$$

This way one can obtain the system:

$$\begin{bmatrix} \hat{A} - \hat{B}\hat{D}^{-1}\hat{C} & 0 \\ \hat{C} & \hat{D} \end{bmatrix} \begin{bmatrix} \mathbf{b} \\ \hat{p} \end{bmatrix} = \begin{bmatrix} \hat{\mathrm{f}}_{bnd} - \hat{B}\hat{D}^{-1}\hat{\mathrm{f}}_p \\ \hat{\mathrm{f}}_p \end{bmatrix}. \tag{1.57}$$

It is important to observe that the system defined by the first row of equation (1.57) is the only one that has to be solved and that it is much smaller than the initial one (equation (1.53)). After having computed $\mathbf{u}_{bnd}$ and p_0 with such a system, it is possible to recover the other pressure coefficients as:

$$\hat{p} = D^{-1}\left(\hat{\mathrm{f}}_p - \hat{C}\mathbf{b}\right).$$

We highlight that the matrix D has already been inverted to compute $\mathbf{b}$, therefore it can be used to directly obtain $\hat{p}$ without solving any linear system. Finally, $\mathbf{u}_{int}$ can be recovered with the third row of equation (1.54):

$$\mathbf{u}_{int} = C^{-1}\left(\mathrm{f}_{int} - \widetilde{B}^T\mathbf{u}_{bnd} + D_{int}^T p\right).$$

Once again, since C^{-1} has already been computed, it is possible to directly recover $\mathbf{u}_{int}$.

2. Reduced Basis Methods

In order to numerically solve a system of PDEs, many different methods have been developed in the last decades. Unfortunately, one of their common disadvantages is that they discretize a continuous system with a combination of numerous unknowns in order to obtain a discrete solution that is as close as possible to the continuous one. This way, even if it is possible to obtain an arbitrary level of accuracy without knowing any information about the solution, assembling and solving the obtained linear systems is very expensive. Moreover, one might be interested in solving several times the same problem with different values of some parameters but, due to the huge computational cost required by such methods, the computation can easily become too expensive. Common situations where the classical numerical methods can not be used because of their cost are, for instance, related to optimal control, uncertainty quantification, design, real time queries and optimization. In this work we propose a class of methods called *reduced basis* (RB) methods [38], they divide the computations in two different phases. The first phase, named offline phase, is very expensive and it is used to set up the model, then the problem is very efficiently solved for a specific value of the parameter in the online phase [50]. Finally, we highlight that the method will be presented as in [38].

2.1. Problem formulation

Let us consider a suitably regular domain $\Omega \subset \mathbb{R}^d$ with $d = 1, 2, 3$ and, in order to properly impose the Dirichlet boundary conditions, let us assume that a measurable portion of the boundary Γ_D is associated to such a condition. It should be noted that, in this work, we always analyze Galerkin problems associated to regular domains, even if it is possible to extend them considering more general geometries [31]. It is then possible to define the Hilbert space V such that:

$$V = V(\Omega) = \{v \in H^1(\Omega) : v_{|\Gamma_D} = 0\},$$

with the inner product $(v, w)_V, \forall v, w \in V$ and the corresponding induced norm $\|v\|_V = \sqrt{(v, v)_V}$, $\forall v \in V$. Finally, let us introduce the parameter space $P \subset \mathbb{R}^n$ that contains the involved parameter vector $\mu = (\mu_1, ..., \mu_n)$. Consequently, it is possible to define the abstract and parametric Galerkin problem as:

Given $\mu \in P$, find $u(\mu) \in V$ such that:

$$a(u(\mu), v; \mu) = f(v; \mu), \qquad \forall v \in V. \tag{2.1}$$

Where $a(\cdot, \cdot; \mu) : V \times V \times P \to \mathbb{R}$ is a bilinear, symmetric, coercive and continuous form for all $\mu \in P$ with respect to the the norm $\|\cdot\|_V$ when needed. Therefore, the following relations hold

for all $u, v, w \in V$:

$$a(c_1 u + c_2 v, w; \mu) = c_1 a(u, w; \mu) + c_2 a(v, w; \mu),$$
$$a(v, w; \mu) = a(w, v; \mu),$$
$$a(v, v; \mu) \geq \alpha(\mu)\|v\|_V^2,$$
$$a(v, w; \mu) \leq \gamma(\mu)\|v\|_V\|w\|_V,$$

where $\alpha(\mu)$ and $\gamma(\mu)$ are respectively defined, for every $\mu \in P$, as:

$$\alpha(\mu) = \inf_{v \in V} \frac{a(v, v, \mu)}{\|v\|_V^2}, \qquad \gamma(\mu) = \sup_{w \in V} \sup_{v \in V} \frac{a(v, w; \mu)}{\|v\|_V\|w\|_V}.$$

Moreover, let us assume that $f(\cdot; \mu) : V \times P \to \mathbb{R}$ is a linear and continuous form with respect to the norm $\|\cdot\|_V$, for any parameter value and any $v, w \in V$:

$$f(c_1 v + c_2 w; \mu) = c_1 f(v; \mu) + c_2 f(w; \mu),$$
$$f(v; \mu) \leq \delta(\mu)\|v\|_V.$$

Such properties are useful to establish the well-posedness of the Galerkin problem, except the symmetry of $a(\cdot, \cdot; \mu)$ that can be used to simplify the formulation and the error analysis [64]. Since, in most cases, obtaining an analytical solution of problem (2.1) is impossible, such a problem is numerically solved: the space V is approximated by a finite-dimensional space $V^\delta \subset V$ and the discrete Galerkin formulation reads as follows:

Given $\mu \in P$, find $u^\delta(\mu) \in V^\delta$ such that:

$$a(u^\delta(\mu), v; \mu) = f(v; \mu), \qquad \forall v \in V^\delta. \tag{2.2}$$

Such problems have been deeply investigated, we thus refer, for instance, to [11], [12] and [13] for some classical results about nonlinear and parametric discrete problems. Once more, we remark that the discrete solution $u^\delta(\mu)$ is an approximation of the solution $u(\mu)$ of the continuous problem (2.1), they can be related using the Galerkin orthogonality and the coercivity and the continuity assumptions as follows:

$$\|u(\mu) - u^\delta(\mu)\|_V \leq \sqrt{\frac{\gamma(\mu)}{\alpha(\mu)}} \inf_{v^\delta \in V^\delta} \|u(\mu) - v^\delta\|_V. \tag{2.3}$$

This is important because it ensures that the solution is approximated with an arbitrary accuracy; because of such a property $u^\delta(\mu)$ will be called *truth solution*. More generally, the manifold $\mathcal{M}$ that contains all the solutions of problem (2.1), i.e.

$$\mathcal{M} = \{u(\mu) : \mu \in P\} \subset V,$$

can be approximated by its discrete counterpart (comprising the discrete solutions of problem (2.2)):

$$\mathcal{M}^\delta = \{u^\delta(\mu) : \mu \in P\} \subset V^\delta.$$

The key assumption, in the reduced basis framework, is that $\mathcal{M}^\delta$ is of low dimension even if the dimension N^δ of V^δ is much higher. The fact that N^δ is huge is the main cause of the high computational cost associated to the numerical solver. Exploiting such an assumption, one would like to seek the discrete solution in a much smaller space that is approximately equivalent to $\mathcal{M}^\delta$. Let V^{rb} be such a space, $N^{rb} \ll N^\delta$ its dimension and $\{\xi_1, ...\xi_{N^{rb}}\} \subset \mathcal{M}^\delta$ the associated reduced basis, therefore V^{rb} can be described as:

$$V^{rb} = \text{span}\{\xi_1, ..., \xi_{N^{rb}}\} \subset V^\delta.$$

Problem (2.2) can be expressed in the reduced space V^{rb} as follows:

Given $\mu \in P$, find $u^{rb}(\mu) \in V^{rb}$ such that:

$$a(u^{rb}(\mu), v; \mu) = f(v; \mu), \qquad \forall v \in V^{rb}. \tag{2.4}$$

After having defined the problem in the reduced space, one would like to prove that the solutions that can be obtained are accurate enough. To do so, let us consider the norm of the difference between the continuous solution $u(\mu)$ and the one in V^{rb} as a measure of the accuracy; such a norm can be divided into two different terms by the triangle inequality:

$$\begin{aligned}\|u(\mu) - u^{rb}(\mu)\|_V &= \|u(\mu) - u^\delta(\mu) + u^\delta(\mu) - u^{rb}(\mu)\|_V \\ &\leq \|u(\mu) - u^\delta(\mu)\|_V + \|u^\delta(\mu) - u^{rb}(\mu)\|_V.\end{aligned}$$

Since any arbitrary accuracy can be achieved by the full order solver, one can neglect the term $\|u(\mu) - u^\delta(\mu)\|_V$ and focus only on $\|u^\delta(\mu) - u^{rb}(\mu)\|_V$. However, it is important to remember that, in order to obtain accurate solutions for any parameter value, one has to control the error associated to any $\mu \in P$ or, equivalently, one could control a global quantity such as:

$$\sup_{u^\delta \in \mathcal{M}^\delta} \inf_{v^{rb} \in V^{rb}} \|u^\delta - v^{rb}\|_V,$$

or, relaxing the supremum over $\mathcal{M}^\delta$, as:

$$\sqrt{\int_{\mu \in P} \inf_{v^{rb} \in V^{rb}} \|u^\delta(\mu) - v^{rb}\|_V^2 d\mu}, \tag{2.5}$$

that introduces the notion of least-squares optimality for the accuracy. Finally, it is important to observe that the accuracy and the efficiency of the method are very problem dependent, in fact, a statement equivalent to relation (2.3) holds for the reduced space too, because it is based on a similar Galerkin formulation:

$$\|u(\mu) - u^{rb}(\mu)\|_V \leq \sqrt{\frac{\gamma(\mu)}{\alpha(\mu)}} \inf_{v^{rb} \in V^{rb}} \|u(\mu) - v^{rb}\|_V.$$

Moreover, in order to properly approximate the manifold $\mathcal{M}^\delta$, one may need another high-dimensional space. Some information on the suitable dimension for V^{rb} are contained in the Kolmogorov N^{rb}-width of $\mathcal{M}^\delta$ in V^{rb} [58], that is defined as:

$$d_{N^{rb}}(\mathcal{M}^\delta) = \inf_{V^{rb}:\ \dim V^{rb} = N^{rb}} \sup_{u^\delta \in \mathcal{M}^\delta} \inf_{v^{rb} \in V^{rb}} \|u^\delta - v^{rb}\|_V.$$

If $d_{N^{rb}}(\mathcal{M}^\delta)$ rapidly decays, it means that $\mathcal{M}^\delta$ can be accurately approximated with low dimensional spaces. On the other hand, if the decay is very slow, the reduced basis method might not be the best approach to handle the problem because it is impossible to summarize a lot of information in few basis functions.

2.2. Proper orthogonal decomposition

One of the most common methods that can be used to generate the reduced space V^{rb} is the *proper orthogonal decomposition* (POD) method [38]. Let us consider a discrete and finite-dimensional subset $P_h \subset P$ of cardinality M. The obtained N^{rb}-dimensional space is the one that minimizes the quantity:

$$\sqrt{\frac{1}{M}\sum_{\mu\in P_h}\inf_{v^{rb}\in V^{rb}}\|u^\delta(\mu)-v^{rb}\|_V^2}, \tag{2.6}$$

over all the N^{rb}-dimensional subspaces of $V^M = \text{span}\{u^\delta(\mu) : \mu \in P_h\}$ [46]. Since V^M is generated by the elements in $\mathcal{M}^\delta(P_h) = \{u^\delta(\mu) : \mu \in P_h\} \subset V^\delta$, the quantity in (2.6) can be considered as a discrete version of the quantity in equation (2.5).
Let us sort the elements of P_h as $\mu_1, ..., \mu_M$, the corresponding discrete solutions can thus be ordered as $u^\delta(\mu_1), ...u^\delta(\mu_M)$. Such discrete solutions will be denoted as $\psi_i = u^\delta(\mu_i)$ in order to lighten the notation. Moreover, it is possible to introduce the following symmetric and linear operator $C : V^M \to V^M$ defined as:

$$C(v^\delta) = \frac{1}{M}\sum_{i=1}^{M}\left(v^\delta, \psi_i\right)_V \psi_i, \qquad\qquad v^\delta \in V^M.$$

Let us consider its eigenvalue-eigenfunction pairs $(\lambda_i, \xi_i) \in \mathbb{R} \times V^M$ normalized with the constraint that $\|\xi_i\| = 1$ and such that $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_M$; they satisfy:

$$\left(C\left(\xi_i\right), \psi_j\right)_V = \lambda_i \left(\xi_i, \psi_j\right)_V, \qquad\qquad 1 \leq j \leq M.$$

The set $\{\xi_1, ...\xi_{N^{rb}}\}$ of the first N^{rb} eigenfunctions generates the N^{rb}-dimensional space V^{POD} that minimizes the quantity in (2.6) and each element ξ_i is an orthogonal POD basis function. With such functions it is possible to define a projection operator $P_{N^{rb}} : V \to V^{POD}$ as follows:

$$P_{N^{rb}}(v) = \sum_{i=1}^{N^{rb}} \left(v, \xi_i\right)_V \xi_i.$$

It can be observed that, thanks to the fact that the POD basis functions are orthonormal to each other, the following relation holds:

$$\left(P_{N^{rb}}(v), \xi_i\right)_V = \left(v, \xi_i\right)_V, \qquad\qquad 1 \leq i \leq N^{rb}.$$

It is important to observe that, when the projection is applied to $M^{\delta}(P_h)$, the following error estimate [74] can be used to properly select the value of N^{rb}:

$$\sqrt{\frac{1}{M}\sum_{i=1}^{M}\|\psi_i - P_{N^{rb}}(\psi_i)\|_V^2} = \sqrt{\sum_{i=N^{rb}+1}^{M}\lambda_i}. \tag{2.7}$$

Therefore, to choose the value of N^{rb}, one can normalize the eigenvalues over their sum obtaining the normalized eigenvalues:

$$\widetilde{\lambda}_i = \frac{\lambda_i}{\sum_{j=1}^{M}\lambda_j},$$

and, fixing a POD tolerance $\tau_{POD} \in [0,1]$, select N^{rb} as:

$$N^{rb} = \underset{N^{rb}=1,2,\dots,M}{\arg\max}\left[\sum_{i=1}^{N^{rb}}\widetilde{\lambda}_i\right] \leq \tau_{POD}. \tag{2.8}$$

This way the quantity on the right hand side of equation (2.7) can be controlled by the quantity $\sqrt{1-\tau_{POD}}$, therefore the error on its left hand side can become arbitrary small increasing such a tolerance.
From the implementation point of view, one can proceed as follows to generate the space V^{POD}:

1. Properly discretize P in order to obtain P_h.
2. Compute the snapshots $\psi_i = u^{\delta}(\mu_i)$ for all μ_i in P_h.
3. Construct the correlation matrix $C \in \mathbb{R}^{M\times M}$ defined as

$$C[i][j] = \frac{1}{M}(\psi_i, \psi_j)_V, \qquad 1 \leq i,j \leq M.$$

4. Compute the eigenvalue-eigenvector pairs (λ_i, v_i) of C (see [39] for an efficient algorithm to perform such a task), with $\|v_i\| = 1$, as:

$$Cv_i = \lambda_i v_i, \qquad 1 \leq i \leq M.$$

5. Sort the eigenvalues in descending order, sort the eigenvectors accordingly.
6. Choose τ_{POD} and compute N^{rb} using formula (2.8).
7. Compute the orthogonal POD basis functions $\{\xi_1, \dots \xi_{N^{rb}}\}$ as:

$$\xi_i(x) = \frac{1}{\sqrt{M}}\sum_{j=1}^{M}(v_i)_j\psi_j(x), \qquad 1 \leq i \leq N^{rb},$$

where the symbol $(v_i)_j$ represents the j-component of the vector v_i.

The described method is optimal in the l^2 norm thanks to the relation (2.7) but, due to the fact that M truth solutions have to be computed, grouped in the correlation matrix and analyzed, the offline phase is very expensive. In fact in general $M \gg N^{rb}$ and, therefore, one has to compute many different snapshots even if most of them have to be discarded because they do not contain important informations. Furthermore, the computation of the eigenvalue-eigenvector pairs of the correlation matrix scales as $\mathcal{O}(M^2 N^{rb})$ and can be considered a very expensive step of the POD method. However, such a method remains very attractive because the obtained reduced space is proved to be optimal in an l^2-sense. An alternative method, that minimizes the error in an l^∞-sense, is the *greedy algorithm* [38].

2.3. Online phase

In the previous sections we described how to properly formulate the Galerkin problem in the reduced space and how to generate such a space. The generation of the reduced space is a very expensive operation, in fact one has to compute many different solutions that have to be then processed again to construct V^{rb}. However, since the aim of the reduced basis method is to be able to obtain a discrete solution of problem (2.1) in a very efficient way, one has to split the computation in two phases: in the offline one the snapshots are computed and V^{rb} is generated, while in the online one the problem is efficiently solved. In order to solve the problem in the online phase, one could construct the full order matrix $A^\delta(\mu)$ and the full order right hand side $\mathrm{f}^\delta(\mu)$ of the linear system associated to problem (2.2) and project them onto V^{rb} as:

$$A^{rb}(\mu) = B^T A^\delta(\mu) B, \qquad \mathrm{f}^{rb}(\mu) = B^T \mathrm{f}^\delta(\mu),$$

where $B \in \mathbb{R}^{N^\delta \times N^{rb}}$ is a matrix which columns contain the different basis functions of V^{rb}. Since $A^\delta(\mu) \in \mathbb{R}^{N^\delta \times N^\delta}$ and $\mathrm{f}^\delta(\mu) \in \mathbb{R}^{N^\delta}$, but $A^{rb}(\mu) \in \mathbb{R}^{N^{rb} \times N^{rb}}$ and $\mathrm{f}^{rb}(\mu) \in \mathbb{R}^{N^{rb}}$, the solution of the full order system

$$A^\delta(\mu) u^\delta = \mathrm{f}^\delta,$$

is much more expensive than the one of the reduced system

$$A^{rb}(\mu) u^{rb} = \mathrm{f}^{rb},$$

because one assumes that $N^\delta \gg N^{rb}$. However, the assembly of the reduced system requires operations that depend on N^δ, significantly reducing the efficiency of the online phase. In order to overcome this problem, one relies on the affine decomposition of each term of the full order system.

2.3.1. Affine decomposition

Let us assume that both $a(\cdot, \cdot; \mu)$ and $f(\cdot; \mu)$ can be expressed as linear combinations of forms that do not depend on μ and coefficients that only depend on it (see [46] for a more detailed explanation):

$$a(v, w; \mu) = \sum_{q=1}^{Q_a} \theta_a^q(\mu) a_q(v, w),$$

$$f(v;\mu) = \sum_{q=1}^{Q_f} \theta_f^q(\mu) f_q(v),$$

where

$$\begin{aligned} a_q : V \times V \to \mathbb{R}, \qquad & f_q : V \to \mathbb{R}, \\ \theta_a^q : P \to \mathbb{R}, \qquad & \theta_f^q : P \to \mathbb{R}. \end{aligned}$$

Since all the forms $a_q(\cdot,\cdot)$ and $f_q(\cdot)$ are independent of μ, their associated matrices and vectors can be constructed during the offline phase as:

$$\begin{aligned} A_q^{rb}[i][j] &= a_q(\xi_j, \xi_i), \qquad 1 \le i, j \le N^{rb}, \\ \mathrm{f}_q^{rb}[i] &= f_q(\xi_i), \qquad 1 \le i \le N^{rb}. \end{aligned}$$

Finally, it is possible to efficiently assemble the reduced solution matrix $A^{rb}(\mu)$ and the associated reduced right hand side as:

$$A^{rb}(\mu) = \sum_{q=1}^{Q_a} \theta_a^q(\mu) A_q^{rb},$$

$$\mathrm{f}^{rb}(\mu) = \sum_{q=1}^{Q_f} \theta_f^q(\mu) \mathrm{f}_q^{rb}.$$

It should be noted that such assemblies respectively scale as $\mathcal{O}\left(Q_a \left(N^{rb}\right)^2\right)$ and $\mathcal{O}\left(Q_f \left(N^{rb}\right)^2\right)$, they are therefore independent from N^δ. Unfortunately it is not always possible to rely on the initial affinity assumption and, to recover an approximation of it, one has to use the *empirical interpolation method* [8].

2.3.1.1. An explanatory example

In order to better understand how to exploit the affine decomposition, let us consider one of the easiest parametric problems: the parametric steady heat conduction problem. Let us denote as Ω a rectangular domain, it is divided into two subdomains Ω_1 and Ω_2 as in figure 2.1. Moreover, let Γ_1^i and Γ_2^i, $i = 1, 2, 3$, respectively be the external boundaries of Ω_1 and Ω_2, they are such that:

$$\left(\bigcup_{i=1}^{3} \Gamma_1^i\right) \cup \left(\bigcup_{j=1}^{3} \Gamma_2^j\right) = \partial\Omega,$$

$$\Gamma_i^j \cap \Gamma_k^l = \emptyset, \qquad \forall i, k \in \{1,2\}, \forall j, l \in \{1,2,3\}.$$

A non-homogeneous Neumann boundary condition is associated to the right side of the domain Γ_2^2, while homogeneous Dirichlet boundary conditions are imposed on the other portions of $\partial\Omega$. The problem is thus governed by the following system:

$$\begin{cases} \nabla \cdot k(\mu) \nabla u(\mu) = 0, & \text{in } \Omega \\ u(\mu) = 0, & \text{in } \partial\Omega \setminus \Gamma_2^2 \\ k(\mu) \nabla u(\mu) \cdot n = g(\mu), & \text{in } \Gamma_2^2 \end{cases}. \tag{2.9}$$

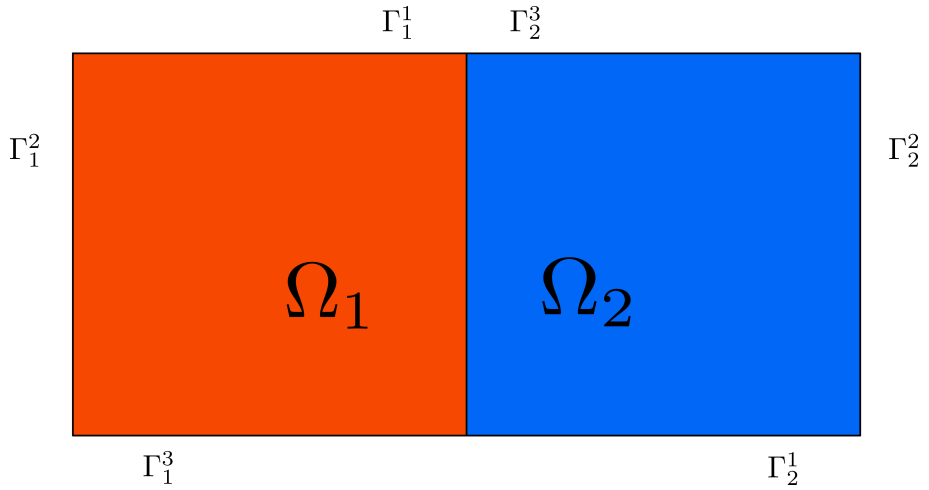


Figure (2.1) Domain of interest associated to the heat diffusion example.

Here $u(\mu)$ is the unknown temperature field, n the outward pointing normal unit vector on Γ_2^2, while $k(\mu)$ and $g(\mu)$ are two known functions that depend on the parameter μ. Physically, $k(\mu)$ is related to the material conductivity while $g(\mu)$ represents the heat flux across Γ_2^2. To consider a more general problem, let us assume that μ is a vector in $\mathbb{R}^2$, the first component of it, named μ_1, is associated to $k(\mu)$, while the second one (namely μ_2), parametrizes $g(\mu)$. For the sake of simplicity we assume that $g(\mu)$ and $k(\mu)$ are, respectively, a constant function and a piecewise constant function. They can be thus defined as:

$$g(\mu) = \mu_2,$$

$$k(\mu) = k_1 + \mu_1 \mathbb{1}_{\Omega_2} = \begin{cases} k_1 & \text{in } \Omega_1 \\ k_1 + \mu_1 & \text{in } \Omega_2 \end{cases},$$

here k_1 is a known constant value associated to the first subdomain, while the difference between the latter value and the corresponding one in Ω_2 is represented by μ_1. In order to obtain the weak formulation, one chooses the proper space V defined as:

$$V = \{v \in H^1(\Omega) : v = 0 \text{ in } \partial\Omega \setminus \Gamma_2^2\},$$

multiplies the Poisson equation in system (2.9) by an arbitrary test function $v \in V$ and integrates over the entire domain Ω obtaining the following equation:

$$\int_\Omega (\nabla \cdot k(\mu)\nabla u(\mu))\, v = 0, \qquad \forall v \in V.$$

It is then possible to integrate it by parts in order to obtain:

$$\int_\Omega k(\mu)\nabla u(\mu)\nabla v = \int_{\partial\Omega} k(\mu)\, (\nabla u(\mu) \cdot n)\, v, \qquad \forall v \in V,$$

where the integral on the right hand side is exactly zero on $(\partial\Omega \setminus \Gamma_2^2)$ because $v \in V$. The obtained variational formulation is, therefore:

Find $u \in V$ such that:

$$a(u(\mu), v; \mu) = f(v; \mu), \qquad \forall v \in V, \tag{2.10}$$

where

$$a(u(\mu), v; \mu) = \int_\Omega k(\mu)\nabla u(\mu)\nabla v,$$

$$f(v; \mu) = \int_{\partial\Omega} k(\mu)\nabla u(\mu)v \cdot n = \int_{\Gamma_2^2} k(\mu)\nabla u(\mu)v \cdot n = \mu_2 \int_{\Gamma_2^2} v.$$

It is important to highlight that, in order to consider a well-posed problem, $k(\mu)$ is required to be greater then 0, therefore the inequalities $k_1 > 0$ and $\mu_1 > -k_1$ have to hold.
In order to derive an affine decomposition, one can split the operator $a(\cdot, \cdot; \mu)$ as follows:

$$\begin{aligned} a(u(\mu), v; \mu) &= \int_\Omega k(\mu)\nabla u(\mu)\nabla v \\ &= \int_{\Omega_1} k(\mu)\nabla u(\mu)\nabla v + \int_{\Omega_2} k(\mu)\nabla u(\mu)\nabla v \\ &= \int_{\Omega_1} k_1\nabla u(\mu)\nabla v + \int_{\Omega_2} (k_1 + \mu_1)\nabla u(\mu)\nabla v \\ &= k_1 \int_{\Omega_1} \nabla u(\mu)\nabla v + (k_1 + \mu_1) \int_{\Omega_2} \nabla u(\mu)\nabla v. \end{aligned}$$

It can be observed that, after having properly discretized the problem, each entry (i, j) of the associated matrix A can be computed as:

$$A[i][j] = a(\xi_j, \xi_i) = k_1 \int_{\Omega_1} \nabla\xi_j\nabla\xi_i + (k_1 + \mu_1) \int_{\Omega_2} \nabla\xi_j\nabla\xi_i, \qquad 1 \le i, j \le N^\delta.$$

Such integrals are no longer dependent on μ and, if the reduced version of problem (2.10) has to be solved in the online phase, they can be precomputed during the offline one in order to increase the efficiency. Let us consider the following notation:

$$\begin{aligned} A_1^{rb} &= \int_{\Omega_1} \nabla\xi_j\nabla\xi_i, & A_2^{rb} &= \int_{\Omega_2} \nabla\xi_j\nabla\xi_i, \\ \theta_a^1(\mu) &= k_1, & \theta_a^2(\mu) &= k_1 + \mu_1. \end{aligned}$$

It is now possible to explicitly write the affine decomposition of the stiffness matrix $A^{rb}(\mu)$ associated to the reduced version of problem (2.10) as:

$$A^{rb}(\mu) = \sum_{q=1}^{2} \theta_a^q(\mu) A_q^{rb}.$$

This way $A^{rb}(\mu)$ can be efficiently assembled and the computation of the required integrals is performed only in the offline phase. The same process can be repeated for the right hand side, even if the summation is over a single index, considering the following notation:

$$\mathrm{f}_1^{rb} = \int_{\Gamma_2^2} v, \qquad\qquad \theta_f^1 = \mu_2,$$

and assembling the right hand side associated to the reduced linear system as $\mathrm{f}^{rb} = \theta_f^1 \mathrm{f}_1^{rb}$.

2.4. RB applied to non-coercive problems

One of the main differences between the formulation that arises from the Navier-Stokes equations and the one considered so far is that the former one can not rely on the coercivity property. In order to explain how a non-coercive problem can be handled, we will consider the following parametrized abstract problem instead of problems (1.43) or (1.50):

Given $\mu \in P$, find $u(\mu) \in V$ such that:

$$a(u(\mu), v; \mu) = f(v; \mu), \qquad\qquad \forall v \in W, \tag{2.11}$$

where now the bilinear form $a(\cdot, \cdot; \mu) : V \times W \times P \to \mathbb{R}$ is not coercive. Moreover, it is important to highlight that the test space W is different from the trial one V, this choice leads to a more general approach called Petrov-Galerkin formulation. We refer to [63] for a deeper explanation of the theory required to solve such problems while, once more, we will briefly outline the main results as discussed in [38].

2.4.1. Theoretical results for non-coercive problems

In this section some important theoretical results about non-coercive problems will be presented. The non-parametric scenario will be considered, nevertheless, the following discussion can be easily extended to parametric problems. We will therefore focus on the following problem:

Find $u \in V$ such that:

$$a(u, v) = f(v), \qquad\qquad \forall v \in W. \tag{2.12}$$

The first theorem is the Banach-Nečas-Babuška theorem [38], it provides the existence, the uniqueness and the stability with respect to the data of the solution $u \in V$ of problem (2.12). However, some of its hypothesis can be relaxed considering V as a Banach space and W as a reflexive Banach space [65].

Theorem 2.4.1. (Banach-Nečas-Babuška Theorem)
Let V and W be two Hilbert spaces, $a(\cdot, \cdot) : V \times W \to \mathbb{R}$ a bilinear and continuous form and $f(\cdot) : W \to \mathbb{R}$ an element of the dual space of W. Then, the solution $u \in V$ of problem (2.12) *exists and is unique if and only if:*

1a. There exists a constant $\beta > 0$ such that:

$$\beta \|v\|_V \leq \sup_{\substack{w \in W \\ w \neq 0}} \frac{a(v,w)}{\|w\|_W}, \qquad \forall v \in V. \tag{2.13}$$

2a. The following implication holds:

$$a(v,w) = 0, \quad \forall v \in V \qquad \Longrightarrow \qquad w = 0.$$

We highlight that the first condition can be written as

$$\beta = \inf_{\substack{v \in V \\ v \neq 0}} \sup_{\substack{w \in W \\ w \neq 0}} \frac{a(v,w)}{\|v\|_V \|w\|_W}, \tag{2.14}$$

and it is therefore referred to as *inf-sup condition*. It should be noted that $\beta > 0$ even if the definition in (2.14) is used instead of the one in (2.13).
Moreover, one can obtain the discrete version of problem (2.12) considering two finite-dimensional spaces $V^\delta \subset V$ and $W^\delta \subset W$ and seeking a discrete solution $u^\delta \in V^\delta$ such that:

$$a(u^\delta, v^\delta) = f(v^\delta), \qquad \forall v^\delta \in W^\delta.$$

Such a problem, if the dimension of V^δ is the same of the one of W^δ, can be transformed into a square linear system with a process analogous to what has been explained in section (1.1.1.2). However, in this context the properties of solvability and stability of the discrete problem can not be inherited by the continuous one, therefore one has to directly apply the Banach-Nečas-Babuška theorem on it. The two conditions can be adapted as follows:

1b. There exists a constant $\beta_\delta > 0$ such that:

$$\beta_\delta \|v^\delta\|_V \leq \sup_{\substack{w^\delta \in W^\delta \\ w^\delta \neq 0}} \frac{a(v^\delta, w^\delta)}{\|w^\delta\|_W}, \qquad \forall v^\delta \in V^\delta. \tag{2.15}$$

2b. The following implication holds:

$$a(v^\delta, w^\delta) = 0, \quad \forall v^\delta \in V^\delta \qquad \Longrightarrow \qquad w^\delta = 0.$$

2.4.2. Non-coercive problem formulation and properties

Let us consider again problem (2.11) and, in order to take into account a well-posed problem, let us assume that there exists a constant $\beta(\mu) \geq \beta > 0$ such that:

$$\beta(\mu) = \inf_{\substack{v \in V \\ v \neq 0}} \sup_{\substack{w \in W \\ w \neq 0}} \frac{a(v,w;\mu)}{\|v\|_V \|w\|_W} \geq \beta, \qquad \forall \mu \in P.$$

Such a continuous parametrized problem can be discretized obtaining:

Given $\mu \in P$, find $u^\delta(\mu) \in V^\delta$ such that:

$$a(u^\mu_\delta(\mu), v^\delta; \mu) = f(v^\delta; \mu), \qquad \forall v^\delta \in W^\delta. \tag{2.16}$$

Since the latter problem is solved with the full order solver, we can reasonably assume that a discretization that satisfies the discrete inf-sup condition has been used.
Finally, it is possible to consider the Petrov-Galerkin formulation of the same problem in the reduced basis framework as follows:

Given $\mu \in P$, find $u^{rb}(\mu) \in V^{rb}$ such that:

$$a(u^{rb}(\mu), v^{rb}; \mu) = f(v^{rb}; \mu), \qquad \forall v^{rb} \in W^{rb}. \tag{2.17}$$

where

$$V^{rb} = \text{span}\{u^\delta(\mu_i) : 1 \leq i \leq N^{rb}\},$$

$$W^{rb} = \text{span}\{A^\mu_\delta u^\delta(\mu_i) : 1 \leq i \leq N^{rb}\}.$$

Here $\{\mu_i\}_{i=1}^N \subset P$ is a set of parameter values and $A^\mu_\delta : V^\delta \to W^\delta$ is an operator such that:

$$(A^\delta v^\delta, w^\delta)_W = a(v^\delta, w^\delta; \mu), \qquad \forall v^\delta \in V^\delta, \forall w^\delta \in W^\delta.$$

Since with such an operator it is possible to realize the supremum associated to the inf-sup condition:

$$\beta_\delta(\mu) = \inf_{\substack{v^\delta \in V^\delta \\ v^\delta \neq 0}} \sup_{\substack{w^\delta \in W^\delta \\ w^\delta \neq 0}} \frac{a(v^\delta, w^\delta; \mu)}{\|v^\delta\|_V \|w^\delta\|_W} = \inf_{\substack{v^\delta \in V^\delta \\ v^\delta \neq 0}} \frac{a(v^\delta, A^\mu_\delta v^\delta; \mu)}{\|v^\delta\|_V \|A^\mu_\delta v^\delta\|_W},$$

the operator A^μ_δ is denoted as *inner supremizer operator* [38]. With the latter property one can prove the stability of problem (2.17) as follows:

$$\begin{aligned}\beta_{rb}(\mu) = \inf_{\substack{v^{rb} \in V^{rb} \\ v^{rb} \neq 0}} \sup_{\substack{w^{rb} \in W^{rb} \\ w^{rb} \neq 0}} \frac{a(v^{rb}, w^{rb}; \mu)}{\|v^{rb}\|_V \|w^{rb}\|_W} &= \inf_{\substack{v^{rb} \in V^{rb} \\ v^{rb} \neq 0}} \frac{a(v^{rb}, A^\mu_\delta v^{rb}; \mu)}{\|v^{rb}\|_V \|A^\mu_\delta v^{rb}\|_W} \\ &\geq \inf_{\substack{v^\delta \in V^\delta \\ v^\delta \neq 0}} \frac{a(v^\delta, A^\mu_\delta v^\delta; \mu)}{\|v^\delta\|_V \|A^\mu_\delta v^\delta\|_W} = \beta_\delta(\mu),\end{aligned}$$

where the inequality is obtained considering the infimum over the bigger space $V^\delta \supset V^{rb}$. Therefore $\beta_{rb}(\mu) \geq \beta_\delta > 0$ for any $\mu \in P$: the required inf-sup condition holds in the reduced order frameworks thanks to the stability of the truth problem.

2.5. Conclusion

In this chapter the reduced basis method has been presented. In sections 2.1 and 2.2 we showed that, even if the required Galerkin formulation is similar to the one involved in the FEM or in the SEM, great care must be taken in order to choose and construct the reduced space where the solution is sought. Moreover, in order to ensure the efficiency that characterizes the method, we explained how to exploit the affine decomposition to inexpensively assemble the linear system. Finally, in order to apply such a technique to the Navier-Stokes system, we presented some theoretical results associated to non-coercive problems. It is important to highlight that the diagrams shown in Chapter 4 could be obtained using only the SEM, the continuation method and the deflation one, however, the computational cost associated to the task would be unaffordable. Therefore, we decided to pair the previous techniques to the reduced basis method in order to prove that it is possible to efficiently compute bifurcation diagrams associated to more than one parameter and with more bifurcation points. A more detailed description of such a pairing can be found in Chapter 4.

3. Numerical Methods for Computing Bifurcation Diagrams

In Chapter 1 we presented the spectral element method, that can be used to compute an approximated solution of an arbitrary system of differential equations. Then, in Chapter 2, the reduced basis method has been introduced, it relies on a discretization method (that, in the context of this thesis, is the SEM) and allows one to compute the solution of the same system much more efficiently when a parameter is varied. However, if one wants to generate a bifurcation diagram, one has to be able to compute a single solution of the problem very efficiently and to obtain more solutions associated to the same system. This is important because the system has to be repeatedly solved for different values of the parameters and, after the bifurcation points, multiple solutions can coexist. In order to handle these problems, we decided to use the continuation method [25] and the deflation method [29], that will be presented, after an initial introduction to the bifurcation theory, in sections 3.2 and 3.3. Finally, they will be coupled in the deflated continuation method in the next chapter in order to generate the bifurcation diagrams.

3.1. Introduction and theoretical background

In everyday life, it is possible to observe several different phenomena where changes occur, suddenly or not, due to an external cause. For example, if one applies a stronger and stronger pressure in the middle of a rigid bar fixed at both ends, at the beginning it will stay still, then it could begin to bend and, finally, it will break. Other examples could be the rise of instabilities, asymmetries, irregularities, chaotic behaviours or motion in, respectively, stable, symmetric, regular, ordered or stationary systems. Since one would like to model in a mathematical way these changes, one is obliged to model also what is acting on the observed subject: to do so, it is useful to consider a parameterization of the external cause with scalar quantities that can express, for example, its intensity. In order to be consistent with the previous sections, we will call this parameter μ, it is important to observe again that μ could be a scalar or a vector, but that this distinction does not influence the following theory. However, it should be noted that very often, in literature, this quantity is referred to as λ.
These phenomena can be modeled and studied with the so called *bifurcation theory* [45]. Even if in this work we are studying bifurcations in PDEs, it is convenient to introduce the reader into this theory using examples of solutions of algebraic equations. In this section we will assume that $\mu \in \mathbb{R}$, furthermore we denote with μ_0 the critical value, if it exists, under which and over which there is a qualitative difference in the solution set. When one is working on algebraic equations, one is interested in the way in which the scalar solutions can be (possibly analytically) represented.

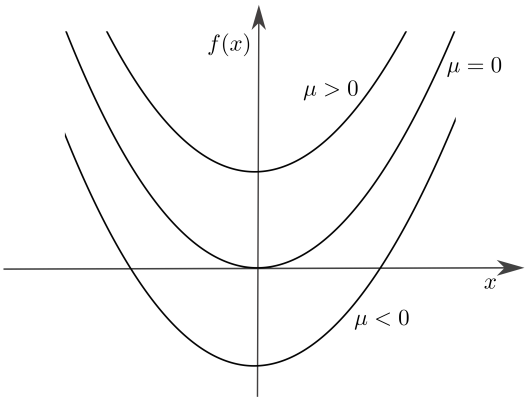


Figure (3.1) Plot of $f(x;\mu) = x^2 + \mu$ for different values of μ

Let us first consider the equation:

$$f(x;\mu) = x^2 + \mu = 0.$$

It represents a centered parabola that can be vertically translated by μ, the solutions of such an equation can be therefore easily described as:

$$\begin{cases} \{\pm\sqrt{-\mu}\}, & \mu < 0 \\ \{0\}, & \mu = 0 \\ \emptyset, & \mu > 0 \end{cases}.$$

In this very simplified setting, it is possible to analytically obtain the critical value $\mu_0 = 0$ and the explicit expression for every existing solutions before and after such a point. A simple visualization of this phenomenon can be obtained with a *bifurcation diagram*: it is a diagram where the solutions are represented on an axis, while the other one accounts for the parameter. In this case one can use a bidimensional representation and plot μ on the x-axis and the solutions x on the y-axis, while, in general, more than one parameter can be involved in the diagram and the solution could be a non scalar quantity. In the latter scenario it is convenient the summarize the information contained in the solutions with a scalar function, while some parameters could be grouped together generating more interesting quantities. It is clear that the plot 3.2 can be divided in three regions: the critical point and the associated single solution, the region before such a point with two solutions for each value of μ, and the region after it, without any solution. We are finally ready to define one of the key elements of the bifurcation theory:

Definition 3.1.1. *Let $\mathcal{F}$ be the set $\mathcal{F} = \{(\mu, x) \in \mathbb{R}^n \times V : \ x$ is a solution of $f(x;\mu) = 0\}$ and $\mathcal{I}$ be the identity operator. A branch $\mathcal{B}$ is defined as a subset of $\mathcal{F}$ such that $\forall g : V \to \mathbb{R}$, $g \in C^1$, the image of $\mathcal{B}$ by the use of $(\mathcal{I}, g) : \mathbb{R}^n \times V \to \mathbb{R}^{n+1}$ is a connected submanifold of class C^1 of $\mathcal{D} \doteq (\mathcal{I}, g)(\mathcal{F})$ that can not be further extended without intersecting other connected submanifolds of class C^1 belonging to $\mathcal{D}$.*

One can observe that in the last example two different branches are present, the first one is $\{(\mu, x) |\ x = \sqrt{-\mu}\}$, while the second one is $\{(\mu, x) |\ x = -\sqrt{-\mu}\}$. It is important to note

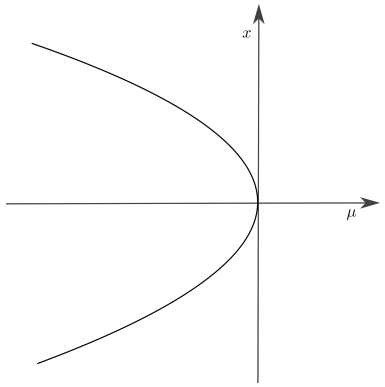


Figure (3.2) Bifurcation diagram relative to the problem $f(x;\mu) = x^2 + \mu = 0$

that they are two distinct branches, in fact, if one chooses the function $g(x) = x^2$, the curve that represents them together is not differentiable anymore. There are several different kinds of bifurcations (see [3] or [44] for a more detailed explanation), in particular, the one explained above is called *turning point*, *fold bifurcation*, *saddle node* or *limit point*. Such a class is characterized by the fact that two solutions annihilate each other or that they are born from a single solution, in both the cases there are two solutions on one side of the critical value (according to the parameter axis) and zero on the other one. In order to describe other types of bifurcations we recall the definition of *bifurcation point*:

Definition 3.1.2. *A solution pair (μ_0, x_0) of the problem $f(x;\mu) = 0$ is a bifurcation point if the number of solutions associated to $\mu < \mu_0$ is different than the same quantity for $\mu > \mu_0$ or if the stability properties of the branches change when μ crosses the critical value μ_0.*

Another important class of bifurcation points is represented by the so called *transcritical* bifurcation points. Let us consider the following equation:

$$\dot{x} = \mu x - x^2. \tag{3.1}$$

The equilibria are always $x = 0$ and $x = \mu$, but their stability changes for $\mu < 0$ or $\mu > 0$. The bifurcation diagram for this problem is shown in figure 3.3, where the heavy lines represent the stable branches, while the dashed lines the unstable ones.
We are finally ready to introduce the *pitchfork* bifurcation points, that are the ones on which this work focuses. The canonical example for this type of bifurcations is the following one:

$$\dot{x} = \mu x - x^3. \tag{3.2}$$

Even if the structure of the problem is very similar to the one of (3.1), their equilibria are very different. One can easily observe that, for $\mu < 0$, the only branch is the trivial one ($x = 0$) and it is stable, while, when $\mu > 0$, there are three branches: the trivial one that becomes unstable and the two stable branches $x = \pm\sqrt{\mu}$. This class of bifurcation points is very common in symmetric problems, it is therefore reasonable that we will face it while solving our test problem, since it is perfectly symmetric. The bifurcation diagram for equation (3.2) is shown in figure 3.4, where the same notation used for the previous example has been assumed for the stability of the branches.

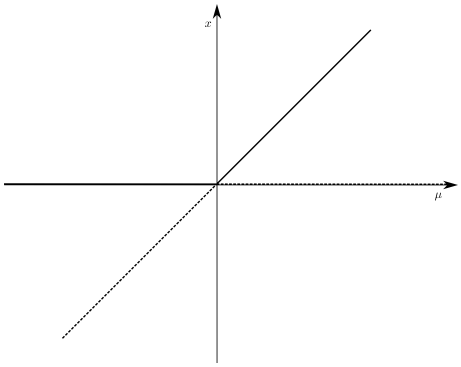


Figure (3.3) Bifurcation diagram relative to the equilibria of the problem $\dot{x} = \mu x - x^2$

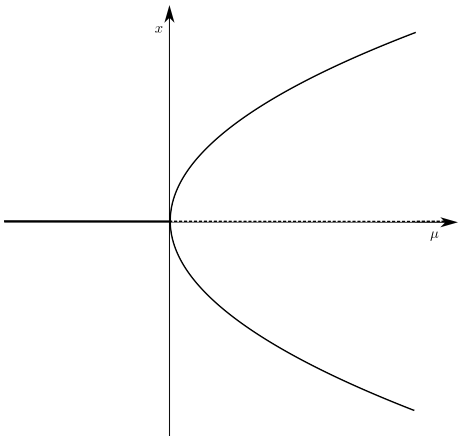


Figure (3.4) Bifurcation diagram relative to the equilibria of the problem $\dot{x} = \mu x - x^3$

3.1.1. Numerical computation of bifurcation diagrams

Since it is often impossible to compute even a single solution of a realistic problem, one is, in most of the cases, unable to analytically compute a bifurcation diagram. Therefore, one is obliged to use some numerical techniques to obtain a discrete approximation of it. It should be noted that, when more than one parameter is involved, it may be complex to properly visualize and interpret the obtained diagram. In this work we will show very simple and interpretable diagram, however, we refer to [71] for a more advanced visualization technique.
The basic idea that is used in this work is to split such a computation into two logic blocks, that have to be alternated to obtain the final result. The first block is called *continuation method*, its aim is to compute, starting from a solution belonging to a given branch, a sequence of solutions that approximates that specific branch. The second block is, instead, named *deflation method*: it is used to compute the first solution that the continuation needs to follow a new branch. Both the methods have advantages and disadvantages that will be analyzed in the next sections. It should be noted that the complexity further increases because, when one numerically solves a PDE for a value of a parameter that is very close to a critical value, the matrix of the associated linear system can become singular, preventing the previous methods to work properly [2].

As remarked in the introduction of this work, the problem of such an approach is that, due to the complexity of its aim, it is very expensive; in fact one has to compute several different solutions for an equal number of different values of the parameters. It is therefore crucial to be able to compute each solution in a very efficient way. On the other hand, the computation of the bifurcation diagram is independent of the discretization method used to solve the equations, and can therefore be applied in a reduced order framework without major modifications. However, we decided to use two slightly different versions of the continuation method and of the deflation one in the offline phase and in the online one in order to obtain the desired result in a more accurate and fast way.

3.2. Continuation method

In section 3.1 the concept of bifurcation diagram has been defined and a first description of the continuation method has been presented. Here the discussion will be further developed, presenting both the advantages and disadvantages of such a method and two different versions of it.
As previously remarked, the aim of the method is to follow a single branch of the diagram, constructing a sequence of solutions that can properly discretize it [25], for this reason it can also be called *branch tracing* or *path following*.
In this section we will assume that a solution of the problem

$$L(\mathbf{u};\mu) = 0, \tag{3.3}$$

exists for any parameter value $\mu \in P$. Equation (3.3) is deliberately abstract so that it is possible to develop a general theory, however, the reader should keep in mind that in this work it will be particularized as the Navier-Stokes system. In that case the velocity-pressure pair $(\mathbf{u}, p)$ will be represented by the abstract variable $\mathbf{u}$, while the parameter μ will play the role of the viscosity ν or will be a vectorial quantity that accounts for the pair (ν, s). Let us remember that s is the inlet velocity scaling: a multiplicative constant that influences the amplitude of the parabolic profile applied as a Dirichlet boundary condition for the inlet velocity.
From now on, in this section, we will assume that $\mu \in \mathbb{R}$. It is possible to do it without loss of generality because the method can be used with $\mu \in \mathbb{R}^N$ letting vary only a single component of the parameter vector, that is exactly what will be done in the last chapter, when a bifurcation diagram with two parameters will be computed. Under such an assumption, it can be noted that equation (3.3) implicitly defines one or more curves $\mathcal{C}_i$ of solution pairs $(\mu, \mathbf{u})$. If L is smooth enough there will be a single branch $\mathcal{C}_0$ that, as long as the Jacobian matrix J_L is full-rank, can be extended [67]. Unfortunately the full-rank condition is respected only by some classes of bifurcation points, for example by the turning points, while other ones, like the pitchfork bifurcations, do not respect it. However, even if from a theoretical point of view this issue prevents the branch to be properly followed, the discrete nature of the continuation method can overcome it.
Let us assume that an iterative numerical solver is available and that we already know a solution pair $(\mu_0, \mathbf{u}_0)$ belonging to the branch $\mathcal{C}_i$. In order to obtain the first solution of a branch, two techniques will be used in this work. The first one consists in using the solver with the null initial

guess (or a best one if it can be obtained solving simpler equations with the *homotopy method*), this method is useful when one wants to compute the first solution of the entire bifurcation diagram and one has not any information about the branches that exist. The second technique is based on the deflation method, that does not allow the solver to converge to known branches. Therefore, if it converges, it will converge to solutions that belong to unknown branches; such method will be further discussed in section 3.3.
The continuation method exploits the fact that, for the iterative solver, each solution $(\mu, \mathbf{u})$ is characterized by a region of attraction such that, if it contains the initial guess used to initialize the iterative solver, it will converge to that specific solution. The aim of the continuation is, in fact, to compute an approximation $(\mu, \widetilde{\mathbf{u}})$ of a solution pair $(\mu, \mathbf{u})$ that should be as close as possible to it. This way, one can start from the solution $(\mu_0, \mathbf{u}_0)$, compute the approximation $(\mu_1, \widetilde{\mathbf{u}}_1)$ and use it as initial guess for the solver to converge to $(\mu_1, \mathbf{u}_1)$. It is then possible to repeat the same procedure computing first $(\mu_2, \widetilde{\mathbf{u}}_2)$ and then $(\mu_2, \mathbf{u}_2)$. In general, the continuation method allows one to easily compute $(\mu_{i+1}, \mathbf{u}_{i+1})$ using the last solution $(\mu_i, \mathbf{u}_i)$. It is therefore possible to iteratively construct a sequence of solution pairs that approximates an entire branch given a first point (a solution pair) belonging to it. The difference between two consecutive solution pairs will be called *step*, while the quantity $\Delta\mu_i = \|\mu_i - \mu_{i-1}\|$ will be considered as a measure of the step size. It is important to remark that in practice, even if μ is a vectorial quantity, only a single component of it is different between μ_i and μ_{i+1}, therefore the step size can be redefined as $\Delta\mu_i = |\mu_{i,j} - \mu_{i-1,j}|$, where the j-th component is the one that varies.
However, if $\Delta\mu_i$ is randomly chosen three problems could arise:

- A turning point occurs, the solver could diverge.
- $\Delta\mu_i$ is too large, therefore the approximation $\widetilde{\mathbf{u}}_i$ is too bad and the solver can not converge, furthermore a pitchfork bifurcation could be overlooked.
- $\Delta\mu_i$ is too small causing a waste of resources, in the worst scenario the computational cost of the method can become prohibitive.

In this work two different continuation methods are used, they will be explained in the following sections, focusing on their implementations and their advantages and disadvantages.

3.2.1. Simple continuation

We will discuss, in this section, the simplest possible version of the continuation method. It is based on the assumption that, due to the continuity of each branch, if the step size $\Delta\mu_i$ is very short, the sought solution $\mathbf{u}_i$ will be very similar to $\mathbf{u}_{i-1}$. Therefore, if $\Delta\mu_i$ is small enough, $\mathbf{u}_{i-1}$ can be used as initial guess to obtain $\mathbf{u}_i$.
One of the main problem of this method is the fact that a way to determine a proper $\Delta\mu_i$ does not exist and very often one is obliged to choose it with heuristics or previous knowledge. For instance, one could decide to fix the same value $\Delta\mu$ for each step size $\Delta\mu_i$, selecting $\Delta\mu$ running several times the simulation and analyzing the converge of the iterative solver. If it can converge in very few iterations the step size is probably too small and many resources would be wasted because of the computation of too many solutions. On the other hand, if $\Delta\mu$ is too large, $\mathbf{u}_{i-1}$

is very different from $\mathbf{u}_i$ and it could cause many different numerical issues. Unfortunately, the optimal number n_{opt} of iterations is not known a priori and it is very problem dependent. In this work we fixed $n_{opt} = 6$ as suggested in [67] but many different and more complex techniques to properly select the best value of $\Delta\mu_i$ at each iteration are described both in [1] and in [67].
Let us analyze the different problems that arise when $\Delta\mu_i$ is too large. Firstly, the solver could be initialized with an initial guess that is so far from the actual solution that it is not able to converge. Secondly, the initial guess could be very different from the sought solution but not enough to make the solver diverge, still it will need many iterations to converge and this phenomenon will slow down the entire computation. Lastly, if $\mathbf{u}_{i-1}$ is between two branches when $\mu = \mu_i$, the solver could converge to a solution belonging to the wrong one.
Furthermore, fixing a constant step size is not a wise choice because there are regions (in the space $P \times V$) where obtaining a good approximation of the next solution is a more complex task than in other parts of the domain. For instance, let us consider the portion $\mathcal{P}_1$ of a branch that is very close to a bifurcation point and the region $\mathcal{P}_2$ that belongs to the same branch but it is far from any singular point. With a fixed step size $\Delta\mu$, the quantity $\|\widetilde{\mathbf{u}}_i - \mathbf{u}_i\|$, that represents the distance between the approximation obtained with the continuation and the sought solution, can vary considerably according to the fact that the pair $(\mu_i, \mathbf{u}_i)$ belongs to $\mathcal{P}_1$ or to $\mathcal{P}_2$. If $\Delta\mu$ is optimal in $\mathcal{P}_1$, it will be too short in $\mathcal{P}_2$ and it will imply a waste of resources; otherwise, if it is optimal in $\mathcal{P}_2$, a very bad approximation will be computed for any point in $\mathcal{P}_1$. Unfortunately, in general, one does not know the proper value of the step size that should be used and, for this reason, it is often chosen in a way that allows the continuation to supply good approximations in the neighbourhood of the bifurcation points, even if the entire computation is slowed down. However, when a prior knowledge is available it is important to exploit it, for instance, if the region that a bifurcation point belongs to is known, it would be clever to use a very small $\Delta\mu$ only in that portion of the branch. Otherwise, if the problem (3.3) is known to became more and more complex to solve when a parameter raises, one could impose that $\Delta\mu_i < \Delta\mu_{i-1}$ if $\mu_i > \mu_{i-1}$ (or viceversa) to improve the stability of the method.
Nevertheless, it should be noted that this simple version of the continuation is very effective when one can not exploit informations on the followed branch apart from the last solution. In fact it allows one to obtain the next solution using only the previous one, that is the only available information about the branch if it is not possible to linearize it. However, the limitations due to the lack of knowledge on $\Delta\mu$, obliged us to implement also a better method, based on a branch linearization, called *pseudo-arclength continuation* [67].

3.2.2. Pseudo-arclength continuation

The key idea behind the *pseudo-arclength continuation* is to use, instead of the parameter itself, the branch arclength to parametrize the curve, this way it is possible to consider the variation of the parameters and the one of the solution together. We remark that we will describe the technique how outlined in [43]. Let us consider a discrete version of problem (3.3):

$$L_\delta(\mathbf{u}_\delta; \mu) = 0, \tag{3.4}$$

where $\mathbf{u}_\delta \in \mathbb{R}^N$, $\mu \in \mathbb{R}$ and $L_\delta : \mathbb{R}^{N+1} \to \mathbb{R}$. Here we assumed again that μ is a scalar quantity, if it is not, it is always possible to consider as parameter only the component of μ the is varying and

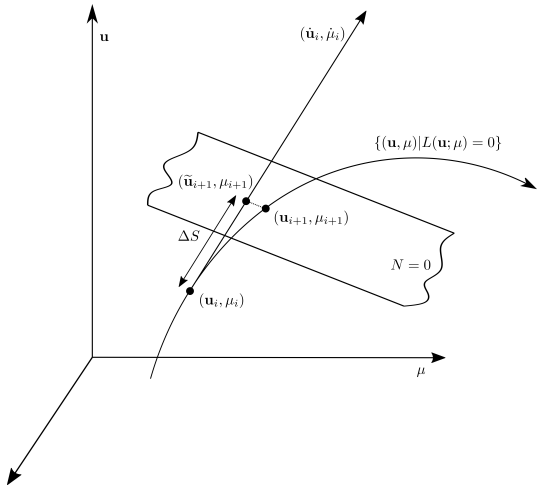


Figure (3.5) Visualization of the pseudo-arclength continuation method

reduce that problem to this one. Furthermore, in order to lighten the notation, the subscripts δ will be omitted both in L_δ and in $\mathbf{u}_\delta$.
Let us denote as $(\mu_i, \mathbf{u}_i)$ a point on the regular path Γ, and with $(\dot{\mu}_i, \dot{\mathbf{u}}_i)$ the unit tangent to Γ in $(\mu_i, \mathbf{u}_i)$. It is then possible to adjoin the following normalization equation to problem (3.4):

$$N(\mathbf{u}, \mu; \Delta S_i) = \dot{\mathbf{u}}_i^T(\mathbf{u} - \mathbf{u}_i) + \dot{\mu}_i(\mu - \mu_i) - \Delta S_i = 0, \tag{3.5}$$

where the dots represent the derivation in the new variable S. It can be noted that equation (3.5) represents a plane that is orthogonal to the tangent vector $(\dot{\mu}_i, \dot{\mathbf{u}}_i)$ and which distance from $(\mu_i, \mathbf{u}_i)$ is exactly ΔS_i. Assuming that ΔS_i and the curvature of Γ are small enough, the intersection between the line tangent to Γ to which $(\mu_i, \mathbf{u}_i)$ belongs and the plane N will be very close to the sought solution pair $(\mu_{i+1}, \mathbf{u}_{i+1})$. Therefore, one wants to solve the system:

$$\begin{cases} L(\mathbf{u}; \mu) = 0 \\ \dot{\mathbf{u}}_i^T(\mathbf{u} - \mathbf{u}_i) + \dot{\mu}_i(\mu - \mu_i) - \Delta S_i = 0 \end{cases}. \tag{3.6}$$

Using the notation $\Delta\mathbf{u}_i = \widetilde{\mathbf{u}}_{i+1} - \mathbf{u}_i$ and $\Delta\mu_i = \widetilde{\mu}_{i+1} - \mu_i$, it is possible to use the Newton method to solve system (3.6), reducing it to the following linear system:

$$\begin{bmatrix} L^i_{\mathbf{u}} & L^i_\mu \\ \dot{\mathbf{u}}_i & \dot{\mu}_i \end{bmatrix} \begin{bmatrix} \Delta\mathbf{u}_i \\ \Delta\mu_i \end{bmatrix} = \begin{bmatrix} L^i \\ \Delta S_i \end{bmatrix}. \tag{3.7}$$

Where in the terms $L^i_\cdot$ the subscripts stand for the derivation operation, while the superscript i represents the fact that the operator is evaluated in the pair $(\mu_i, \mathbf{u}_i)$, therefore $L^i_{\mathbf{u}} = L_{\mathbf{u}}(\mathbf{u}_i; \mu_i)$ and $L^i_\mu = L_\mu(\mathbf{u}_i; \mu_i)$. The branch linearization performed to compute the approximation $(\mu_{i+1}, \widetilde{\mathbf{u}}_{i+1})$ can be visualized in figure 3.5.
One of the problems of the method is that one does not know explicitly $\dot{\mathbf{u}}_i$ and $\dot{\mu}_i$ and has to use an approximation of their values in the previous system. They can be approximated in many

different ways, for example using the relations in (3.8):

$$\begin{cases} \dot{\mathbf{u}} = \dfrac{\mathbf{u}_i - \mathbf{u}_{i-1}}{\Delta S_{i-1}} \\ \dot{\mu} = \dfrac{\mu_i - \mu_{i-1}}{\Delta S_{i-1}} \end{cases} . \tag{3.8}$$

It should be noted, however, that the normalization (3.5) is an assumption and can be changed, for instance generalizing it as

$$N(\mathbf{u}, \mu; \Delta S_i) = \theta \dot{\mathbf{u}}_i^T (\mathbf{u} - \mathbf{u}_i) + (1-\theta)\dot{\mu}_i(\mu - \mu_i) - \Delta S_i = 0, \qquad 0 < \theta < 1,$$

while another possible type of normalization could be:

$$N(\mathbf{u}, \mu; \Delta S_i) = \theta \|\mathbf{u} - \mathbf{u}_i\|^2 + (1-\theta)\|\mu - \mu_i\|^2 - \Delta S_i^2 = 0, \qquad 0 < \theta < 1,$$

that does not involve any unknown derivative.
In this work the used normalization equation is (3.5), and the derivatives are discretized as in (3.8). Since the derivatives approximations require two solutions on the same branch, we used this version of the continuation method only in those situations where we were able to distinguish two consecutive solutions on the same branch, otherwise the simple continuation (subsection 3.2.1) has been used. For example, we were obliged to use the simple continuation after the computation of the first solution and in those situations in which the number of branches in the previous iteration was different than the same number in the iteration before it. In the latter scenario one does not know, in general, which solutions were in the same branches and which branch appeared or disappeared.
Finally, one has to understand how to properly choose the value of ΔS, in order to avoid the issues present in the simple continuation. However, comparing the errors made using a wrong value of $\Delta\mu_i$ and a wrong value of ΔS_i, one observes that much better results can be obtained with the pseudo-arclength continuation. In fact, near a bifurcation point, two consecutive solutions are very different and, since the quantities $\|\widetilde{\mathbf{u}}_{i+1} - \mathbf{u}_i\|$ and $|\mu_{i+1} - \mu_i|$ are bound by ΔS_i, $|\mu_{i+1} - \mu_i|$ is obliged to be very small. On the other hand, when μ_i is far from any critical value and therefore two consecutive solutions are very close, bigger steps are allowed. This way the step size is automatically chosen adaptively, solving the main issue of the simple continuation even with an almost randomly fixed ΔS_i. However, in order to further improve the pseudo-arclength continuation, it is also important to properly choose ΔS_i, many different methods are present in literature, for instance in [1] and in [67]. We decided to implement a very simple method to adaptively set the arclength step size, it is based on the fact that, if the Newton method converges in about 6 iterations, the initial guess was at a proper distance by the sought solution, while if it converges in less or more iterations, the step size was probably too short or too long (such an empirical approach is more deeply discussed in [67]). Therefore, one wants to reduce ΔS_i when the number n of iterations required for the convergence is much higher than 6 and increase it when n is lower. Moreover, in order to manage the continuation of different branches at the same time and the presence of bifurcation points, a minimum value

ΔS_{min} and a maximum value ΔS_{max} has been fixed. The formula used to compute the arclength step size update is the following:

$$\Delta S_{i+1} = \begin{cases} \Delta S_{min}, & \text{if } \frac{6}{n}\Delta S_i < \Delta S_{min} \\ \Delta S_{max}, & \text{if } \frac{6}{n}\Delta S_i > \Delta S_{max} \\ \frac{6}{n}\Delta S_i & \text{otherwise} \end{cases}. \tag{3.9}$$

3.2.2.1. Bordering algorithm

As discussed in section 1.3.2, it is very important to use the static condensation method on the linear system to increase the efficiency of the SEM. Unfortunately, to use such a technique, a very specific structure of the matrix is required, but such a structure is not present in system (3.7). This means that, if one wants to rely on the methods implemented in *Nektar++*, the matrix in the concerned problem has to be properly modified: we decided to implement the bordering algorithm [43] to do it. In order to lighten the notation, let us consider the following system:

$$\begin{bmatrix} A & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} e \\ f \end{bmatrix}. \tag{3.10}$$

Since the matrix in system (3.10) represents the one in (3.7) and the submatrix A stands for $L^i_{\mathbf{u}}$, we can assume that both A and the entire matrix are not singular when the current value of μ is not a critical value. It is thus possible to multiply the first row by A^{-1} to obtain the following equation:

$$A^{-1}Ax + A^{-1}by = x + A^{-1}by = A^{-1}e. \tag{3.11}$$

Moreover, we denote as u and v the following vectors:

$$u = A^{-1}b, \qquad v = A^{-1}e. \tag{3.12}$$

With such a notation, equation (3.11) can be multiplied by c and subtracted by the second row of problem (3.10), obtaining the following equality:

$$cx - cx + dy - cuy = f - cv,$$

that can be rearranged as:

$$y = \frac{f - cv}{d - cu}. \tag{3.13}$$

Therefore, if one wants to solve problem (3.10) relying on an efficient solver of a system associated to the matrix A, one can compute u and v with the relations (3.12), use them to obtain y with the formula (3.13) and then x as:

$$x = v - uy,$$

that can be easily derived from equation (3.11). It should be noted that two linear systems associated to the matrix A have to be solved to compute u and v efficiently, but then it is possible to compute x and y with only three scalar products.

3.3. Deflation method

As previously explained, the continuation method is able to entirely reconstruct a branch of the bifurcation diagram, however, it requires a first solution belonging to it. In order to obtain such a first solution, one can use several methods, in this work we focused on a particular technique called *deflation method* [29]. Since it can be seen as a generalization of a method that can be used to compute the roots of a polynomial, the following simplified scenario will be analyzed first. Moreover, we remark that the method will be described as in [28] and [29].

3.3.1. Deflation method for finding the roots of a polynomial

Let $p(x)$ be a scalar polynomial and let x_0, x_1,..., x_{m-1} be the m roots of it, then $p(x)$ can be written as:

$$p(x) = c_0 \prod_{j=0}^{m-1} (x - x_j). \tag{3.14}$$

If one is able to compute only one root at a time, one can proceed in the following iterative way to obtain all of them. The following notation will be used in the description of the method: the root computed at the k-th iteration will be called x_k. However, it should be noted that, since the terms in (3.14) can be arbitrarily sorted, it is possible to reorder the indices set according to the order in which the root can be found. This way, it is formally correct to use the same notation x_j for both the j-th root in (3.14) and the root found at the j-th iteration of the method. In the first step one can simply compute the root x_0 of $p(x)$, then, in order to find the second one, one can analyze the following polynomial:

$$p_1(x) = c_0 \frac{\prod_{j=0}^{m-1}(x - x_j)}{x - x_0} = c_0 \prod_{j=1}^{m-1} (x - x_j),$$

that can be used to define the problem $p_1(x) = 0$, named *deflated problem*. It can be observed that the roots of $p_1(x)$ are also roots of $p(x)$, but $p_1(x_0) \neq 0$. Since one is able to compute only a single root of a generic polynomial, it is not possible to obtain also x_1 from $p(x)$, however, it is possible to compute it using $p_1(x)$ because x_1 is its first root. The method can be obviously generalized, one wants to discover at the k-th step the root x_k, using the polynomial $p_k(x)$ defined as follows:

$$p_k(x) = c_0 \frac{\prod_{j=0}^{m-1}(x - x_j)}{\prod_{j=0}^{k-1}(x - x_j)} = c_0 \prod_{j=k}^{m-1} (x - x_j).$$

Such polynomials can be iteratively constructed removing the last obtained root from the last analyzed polynomial.

It is important to note that, if the root is sought with a numerical method, the exact solution x_j will not be found, in its place the approximation $\widetilde{x}_j$ will be computed. This implies that x_j is still a root of $p_{j+1}(x)$, even if, in its neighbourhood, the deflated function will be characterized by strong gradients that will prevent the solver to converge again to x_j. Let us consider, for example, the polynomial:

$$q(x) = (x - 1)(x + 1)x,$$

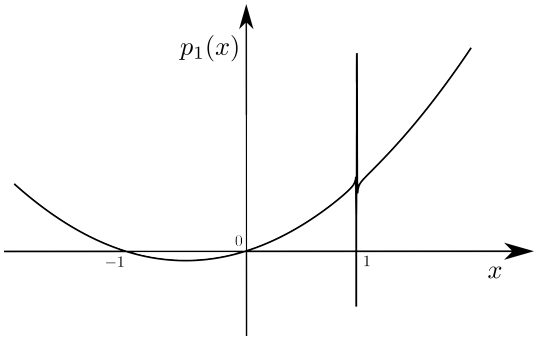


Figure (3.6) Visualization of a numerically deflated function

which roots are $\{-1, 0, 1\}$. Assuming that the root $x = 1$ is the first that can be found, but that the solver is able to compute only the approximation $\widetilde{x} = 0.999$, the first deflated function can be written as:

$$q_1(x) = \frac{x-1}{x-0.999}(x+1)x. \tag{3.15}$$

It can be noted that $x = 1$ is still a root of $q_1(x)$, but that its global behaviour (shown in picture 3.6) is very close to the one of the exactly deflated polynomial $q_1^e(x) = (x+1)x$. This way, if an iterative method starts from an initial guess that is far enough from the exact solution $x = 1$, it will not converge anymore to that root.
The deflation method has been first generalized to face systems of non-linear algebraic equations (see [14] and [47]), and then it has been further modified to find different solutions of PDEs (see [29]).

3.3.2. Deflation method for finding multiple solutions of PDEs

The aim of this section is to generalize the method described in section 3.3.1 to face problems governed by PDEs. Let us consider again the abstract problem:

$$L(\mathbf{u}) = 0. \tag{3.16}$$

As in the previous section, one wants to obtain as many solutions as possible of (3.16), where a solution is a function $\overline{\mathbf{u}}$ such that $L(\overline{\mathbf{u}}) = 0$. In order to describe the deflation method in infinite-dimensional Banach spaces, a generalized definition of the deflation operator is needed (see [29]).

Definition 3.3.1. *Let V, W, Z and U respectively denote three Banach spaces and an open subset of V. Let $L : U \subset V \to W$ be a Fréchet differentiable operator with derivative L'. For each $\mathbf{w} \in U$, $\mathbf{u} \in U\backslash\{\mathbf{w}\}$, let $\mathcal{M}(\mathbf{u};\mathbf{w}) : W \to Z$ be an invertible linear operator. If, for any L such that $L(\mathbf{w}) = 0$ and $L'(\mathbf{w})$ is nonsingular, and for any sequence $\{\mathbf{u}_i\} \subset U\backslash\{\mathbf{w}\}$ converging to $\mathbf{w}$, the following inequality holds:*

$$\liminf_{i\to\infty} \|\mathcal{M}(\mathbf{u}_i;\mathbf{w})L(\mathbf{u}_i)\|_Z > 0, \tag{3.17}$$

then $\mathcal{M}$ is a deflation operator.

From such a definition one deduces two important properties of a deflation operator:

- Spurious solutions can not exist because $\mathcal{M}$ is an invertible operator. A solution $\mathbf{u}_0$ is said to be spurious if $\mathcal{M}(\mathbf{u}_0;\mathbf{w})L(\mathbf{u}_0) = 0$ but $L(\mathbf{u}_0) \neq 0$. If such a solution existed $\mathcal{M}$ would map both $0 \in W$ and $L(\mathbf{u}_0) \in W$ to $0 \in Z$, but this is not possible since $\mathcal{M}$ is linear and invertible.
- Every solution $\widetilde{\mathbf{u}} \neq \mathbf{w}$ of problem (3.16) is preserved. In fact, if $L(\widetilde{\mathbf{u}}) = 0$, then $\mathcal{M}(\widetilde{\mathbf{u}};\mathbf{w})L(\widetilde{\mathbf{u}}) = 0$ by linearity of $\mathcal{M}$.

The previous definition describes a deflation operator, but it is too abstract in practical scenarios. We thus present a more useful condition to understand if an operator is a deflation operator (the next lemma has been first introduced in [14] and then generalized in [29]).

Lemma 3.3.2. *Let $L : U \subset V \to W$ be a Fréchet differentiable operator and $\mathcal{M}(\mathbf{u};\mathbf{w}) : W \to Z$ a linear operator. If, for each $\mathbf{w} \in U$, for any sequence $\{\mathbf{u}_i\} \subset U\backslash\{\mathbf{w}\}$ converging to $\mathbf{w}$ and for any sequence $\{v_i\} \subset W$, the following property holds:*

$$\|\mathbf{u}_i - \mathbf{w}\|\mathcal{M}(\mathbf{u}_i;\mathbf{w})v_i \xrightarrow{Z} 0 \implies v_i \xrightarrow{W} 0, \tag{3.18}$$

then $\mathcal{M}$ is a deflation operator.

Proof. Let us assume that relation (3.18) holds for a given operator $\mathcal{M}(\mathbf{u};\mathbf{w}) : W \to Z$ that is not a deflation operator. Since $\mathcal{M}$ is not a deflation operator, there exist an operator $L : U \subset V \to W$ and an element $\mathbf{w} \in U$ such that L is a Fréchet differentiable operator, $L(\mathbf{w}) = 0$ and $L'(\mathbf{w})$ is nonsingular. Moreover, there exists a sequence $\{\mathbf{u}_i\} \subset U\backslash\{\mathbf{w}\}$ converging to $\mathbf{w}$ and such that

$$\liminf_{i\to\infty} \|\mathcal{M}(\mathbf{u}_i;\mathbf{w})L(\mathbf{u}_i)\|_Z = 0.$$

It is thus possible to define a subsequence $\{\mathbf{v}_i\} \subset U\backslash\{\mathbf{w}\}$such that $\mathcal{M}(\mathbf{v}_i;\mathbf{w})L(\mathbf{v}_i) \xrightarrow{Z} 0$. Let as consider the sequence $\{l_i\} \subset W$ as:

$$l_i = \frac{L(\mathbf{v}_i)}{\|\mathbf{v}_i - \mathbf{w}\|_U}.$$

The limit $\mathcal{M}(\mathbf{v}_i;\mathbf{w})L(\mathbf{v}_i) \xrightarrow{Z} 0$ can then be expressed as follows:

$$\|\mathbf{v}_i - \mathbf{w}\|_U\mathcal{M}(\mathbf{v}_i;\mathbf{w})l_i \xrightarrow{Z} 0.$$

Furthermore, using the fact that $L(\mathbf{w}) = 0$ and that L is a Fréchet differentiable operator, it is possible to expand it in a Taylor series around the function $\mathbf{w}$ in the following way:

$$\begin{aligned} L(\mathbf{v}_i) &= L(\mathbf{w}) + L'(\mathbf{w};\mathbf{v}_i - \mathbf{w}) + o(\|\mathbf{v}_i - \mathbf{w}\|_U^2) \\ &= \quad 0 \quad + L'(\mathbf{w};\mathbf{v}_i - \mathbf{w}) + o(\|\mathbf{v}_i - \mathbf{w}\|_U^2). \end{aligned}$$

Therefore, the following approximation locally holds:

$$\frac{L(\mathbf{v}_i)}{\|\mathbf{v}_i - \mathbf{w}\|_U} = \frac{1}{\|\mathbf{v}_i - \mathbf{w}\|_U}\left[L'(\mathbf{w};\mathbf{v}_i - \mathbf{w}) + o(\|\mathbf{v}_i - \mathbf{w}\|_U^2)\right] \approx L'\left(\mathbf{w};\frac{(\mathbf{v}_i - \mathbf{w})}{\|\mathbf{v}_i - \mathbf{w}\|_U}\right).$$

Finally, since the hypothesis (3.18) holds, one can observe that $l_i \xrightarrow{Z} 0$ or, equivalently:

$$\frac{L(\mathbf{v}_i)}{\|\mathbf{v}_i - \mathbf{w}\|_U} \xrightarrow{Z} 0.$$

The last two statements imply that $L'(\mathbf{w})$ is singular, leading to a contradiction with the first hypothesis. □

The most intuitive deflation operator that can be useful to handle problems governed by PDEs is the direct generalization of what has been used in section 3.3.1, it can be written as

$$\mathcal{M}(\mathbf{u}; \mathbf{w}) = \frac{\mathcal{I}}{\|\mathbf{u} - \mathbf{w}\|_U}. \tag{3.19}$$

Here and from now on, we denote $\mathcal{I}$ as the identity operator on W. Depending on the nature of the solution $\mathbf{w}$ it could be useful to consider the following modified version of (3.19):

$$\mathcal{M}(\mathbf{u}; \mathbf{w}) = \frac{\mathcal{I}}{\|\mathbf{u} - \mathbf{w}\|_U^p}. \tag{3.20}$$

Furthermore, it has been observed that, using these two deflation operators, a numerical solver could converge to unphysical solutions $\overline{\mathbf{u}}$ simply because $\mathcal{M}(\overline{\mathbf{u}}; \mathbf{w}) \to 0$ as $\|\overline{\mathbf{u}} - \mathbf{w}\|_U \to +\infty$. We thus decided to implement the deflation method with the following associated operator, with $p = 1$ and $p = 2$, to face the latter issue:

$$\mathcal{M}(\mathbf{u}; \mathbf{w}) = \mathcal{I} + \frac{\mathcal{I}}{\|\mathbf{u} - \mathbf{w}\|_U^p}. \tag{3.21}$$

Finally, like in section 3.3.1, it is possible to deflate multiple solutions at the same time simply concatenating different deflation operators. For instance, if one has already computed the solutions $\mathbf{u}_1$,...$\mathbf{u}_{n-1}$, it is possible to solve the following problem to seek the n-th one:

$$\mathcal{M}(\mathbf{u}; \mathbf{u}_1) \cdots \mathcal{M}(\mathbf{u}; \mathbf{u}_{n-1}) L(\mathbf{u}) = 0. \tag{3.22}$$

3.3.2.1. Efficiency of the deflation method

One of the main weaknesses of the deflation method is that it does not ensure that the solver will converge to new solutions. In fact, it can not converge to the already known ones because it is pulled away from them by the deflation operator, but if it is too far from other existing solutions (or if they do not exist) the solver will simply diverge. In order to be reasonably sure that new solutions will not be found, it is convenient to fix a high enough maximum number n_{max} of iterations for the iterative solver.
The issue is that, if such a number is too low there is a chance that a new solution could have been found with an higher value of n_{max} but, if it is too high, several computational resources will be wasted in an useless exploration of the solution space. We decided to use $n_{max} = 150$ in the offline phase and $n_{max} = 300$ in the online one because we observed that, with these two values, the solver is able to converge to new branches if they exist and if they are close to known

branches. However, if the deflation method is coupled with an efficient continuation method it can be considered the bottleneck of the method. For instance, in this work a single step of the continuation method requires to solve the linear system twice to obtain the initial guess and, on average, 6 times to converge to the next solution with the Newton method. Instead, if the solver can not find a new solution of the deflated system, 150 (or 300 in the online phase) iteration will be wasted. It is thus crucial to use a very efficient deflation method to reduce as much as possible the time invested in solving the deflated systems.
If one wants to solve the deflated system $G(\mathbf{u};\mathbf{w}) = \mathcal{M}(\mathbf{u};\mathbf{w})L(\mathbf{u}) = 0$ with the Newton method, it shall iteratively solve the following linear system (note that, from now on, the dependency from $\mathbf{u}$ or $\mathbf{w}$ will be omitted for brevity):

$$J_G \Delta\mathbf{u} = -G. \tag{3.23}$$

Considering for simplicity $\mathcal{M} : U \to \mathbb{R}$, one can expand the Jacobian of the deflated system as:

$$J_G = \mathcal{M} J_L + L\mathcal{M}'^T. \tag{3.24}$$

It can be noted that the computational cost of the assembly of J_G grows quadratically with the degrees of freedom and that, even if J_L is sparse, J_G is not because of the term $L\mathcal{M}'^T$. Therefore, the computation would be further slowed down solving the problem using equation (3.23) directly. Here we propose a method, suggested in [28], to efficiently obtain the solution of equation (3.23) without solving the system itself but modifying the Newton residual of the undeflated system.
Let us consider the undeflated system $L(\mathbf{u}) = 0$ with the associated discretization:

$$J_L \Delta\mathbf{u}_L = -L, \tag{3.25}$$

and the deflated one $G(\mathbf{u}) = 0$ with the analogous Newton discretization:

$$J_G \Delta\mathbf{u}_G = -G. \tag{3.26}$$

Finally, we remember that the Sherman-Morrison formula can be used to obtain an analytical expression of the inverse of a matrix with a rank-one perturbation. If $A \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$ and $c \in \mathbb{R}^n$, then the Sherman-Morrison formula states that:

$$(A + bc^T)^{-1} = A^{-1} - \frac{A^{-1}bc^TA^{-1}}{1 + c^TA^{-1}b}.$$

Using such a formula it is possible to efficiently solve (3.26) in the following way:

$$\begin{aligned}
\Delta\mathbf{u}_G &= -J_G^{-1}G \\
&= -\left(\mathcal{M}J_L + L\mathcal{M}'^T\right)^{-1}(\mathcal{M}L) \\
&= -\left(\mathcal{M}^{-1}J_L^{-1} - \frac{\mathcal{M}^{-1}J_L^{-1}L\mathcal{M}'^T\mathcal{M}^{-1}J_L^{-1}}{1+\mathcal{M}'^T\mathcal{M}^{-1}J_L^{-1}L}\right)(\mathcal{M}L) \\
&= -J_L^{-1}L + \frac{J_L^{-1}L\mathcal{M}'^T\mathcal{M}^{-1}J_L^{-1}L}{1+\mathcal{M}'^T\mathcal{M}^{-1}J_L^{-1}L} \\
&= \left(1 - \frac{\mathcal{M}^{-1}\mathcal{M}'^TJ_L^{-1}L}{1+\mathcal{M}^{-1}\mathcal{M}'^TJ_L^{-1}L}\right)(-J_L^{-1}L) \\
&= \frac{1}{1+\mathcal{M}^{-1}\mathcal{M}'^TJ_L^{-1}L}\Delta\mathbf{u}_L \\
&= \tau\Delta\mathbf{u}_L.
\end{aligned}$$

We highlight that τ can be very efficiently computed, in fact only the computation of a scalar product is needed, and it scales linearly with the total number of degrees of freedom. Moreover, we used such a formula to couple a heuristic to the deflation method in order to increase its efficiency and effectiveness (see section 4.2.1.2). It is important to observe that the computation of $\Delta\mathbf{u}_G$ does not require to assemble and solve system (3.26), instead, one has to solve system (3.25) (that has already been assembled to solve the undeflated system) and then one can obtain $\Delta\mathbf{u}_G$ as:

$$\Delta\mathbf{u}_G = \tau\Delta\mathbf{u}_L.$$

3.4. Conclusion

In this chapter, after a brief introduction, the two methods through which it is possible to compute the bifurcation diagrams shown in Chapter 4 have been introduced. The first one is the continuation method, it is used to obtain an initial guess for the iterative solver in order to follow a specific branch of the diagram. Its accuracy is important because a bad initial guess may imply serious stability issues or the loss of a branch of the diagram. In order to be able to always obtain accurate guesses, we decided to implement two versions of it: the simple continuation (section 3.2.1) and the pseudo-arclength continuation (section 3.2.2); they will be used in different phases of the deflated continuation method (see section 4.2.1.1). The second method that has been introduced in this chapter is the deflation method, it is used to ensure that the iterative solver does not converge to already computed solutions. This way, it is possible to obtain different solutions associated to the same values of the parameters. Such a method is exploited to compute the first solutions on the unknown branches of the diagram, however, since it is very expensive, we focused on its efficiency in section 3.3.2.1.

4. Numerical results

4.1. Overview of the problem

We are finally ready to present the results that can be obtained using the previously described methods. As explained in the introductory chapter and in section A.1, the aim of this work is to apply different numerical techniques in order to compute bifurcation diagrams efficiently. Since we want to study the motion of a fluid, the problem is governed by the Navier-Stokes (NS) equations, that can be simplified using the assumptions in section A.1, obtaining the following system:

$$\begin{cases} \nabla \cdot \mathbf{u} = 0 \\ \mathbf{u} \cdot \nabla \mathbf{u} - \nu \Delta \mathbf{u} + \nabla p = 0 \end{cases}. \tag{4.1}$$

The test case of this work is, therefore, the study of the incompressible, steady and viscous flow in a two-dimensional and straight channel with two backward facing steps and an inlet narrower than the channel height. Before proceeding, it is important to highlight that, since Re is the only important involved parameter and it is non-dimensional, the specific units of measurement can be arbitrarily defined. For this reason they will not be specified, however, if one is interested in the specific physical values, one can consider all the values as associated to the SI system.
The domain is represented in figure 4.1; it is not to scale because, this way, it is possible to focus on the qualitative geometrical features and on the portions of the boundary associated to specific boundary conditions. The domain is the composition of two rectangles, the height of the bigger one is 7.5 length units and its length 90 length units; on the other hand the height and the length of the smaller one are, respectively, only 2.5 and 10 length units. Finally, the smaller rectangle is vertically centered with respect to the bigger one. The vertical wall on the left (in figure in blue) is the inlet, from this side the fluid can enter into the domain and, consequently, a Dirichlet boundary condition is imposed on the velocity. Since the extreme points of such an edge are $(2.5, 0)$ and $(5, 0)$, the inlet condition is the following one:

$$\begin{cases} u = 20(5 - y)(y - 2.5) \\ v = 0 \end{cases}, \tag{4.2}$$

where the multiplicative factor in front of the term $(5 - y)(y - 2.5)$ is used to properly modify the characteristic velocity U of the fluid to obtain the correct values of the Reynolds number. It should be noted that expression (4.2) represents a parabolic profile, it is the most physical one because the velocity has to be equal 0 on the walls due to the no-slip condition [73]. However, it should be noted that, in order to obtain the asymmetrical solutions without using specific numerical techniques, it is possible to modify the inlet velocity profile as in [76]. Condition (4.2)

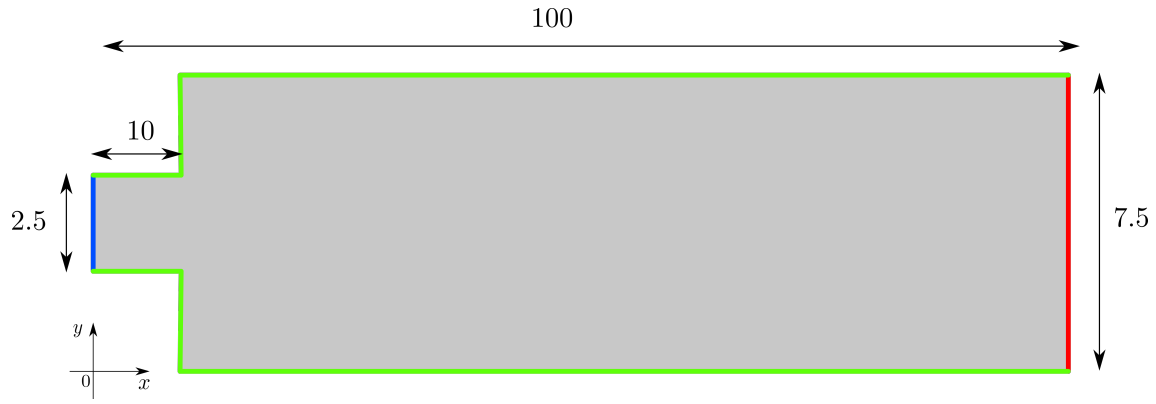
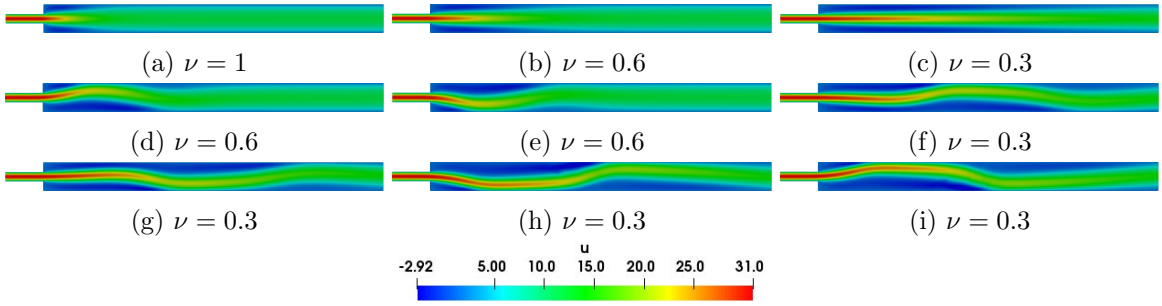


Figure (4.1) Test case domain. The colours highlight the specific type of boundary: the inlet is the blue wall, the outlet the red one, while the remaining ones are impermeable rigid walls.

will be modified as follows in section 4.3:

$$\begin{cases} u = 20s(5-y)(y-2.5) \\ v = 0 \end{cases},$$

where the scaling factor s is the second considered parameter.
The vertical wall on the right, instead, is the outlet: the side from which the fluid exits from the domain. Here homogeneous Neumann boundary condition are used to impose a stress free (or natural) boundary condition on the velocity. Finally, the remaining walls are characterized by no-slip conditions [73]: they are homogeneous Dirichlet boundary conditions that represent the fact that, due to the viscosity of the fluid, the fluid particles adjacent to an impermeable wall are required to move with the same velocity of the wall itself.
Even if it looks very simple, many different solutions can coexist: they are shown in figure 4.2. Only the streamwise component of the velocity is represented, that is the one associated to the horizontal direction, from left to right. Each picture can be interpreted in the following way: where the colour is blue the fluid is still (or going from right to left very slowly in dark blue regions), while in the rest of the domain it is moving from left to right with an horizontal velocity magnitude explained by the legend. Coherently with the Bernoulli law, it is possible to observe that the highest velocities are present in the narrowest region, while the backward facing steps allow the bifurcations to occur. The fluid can go through the channel closer to one of the two walls or in a symmetric way: in physics this phenomenon is named Coanda effect (see [73] for a complete explanation of the Coanda effect and of the Bernoulli law). Moreover, it can be noted that, due to the symmetry of the domain and of the boundary conditions, if a field $\mathbf{u}$ is a solution of problem (4.1), another solution can be obtained mirroring $\mathbf{u}$ with respect to the horizontal symmetry axis $y = 3.75$, such a phenomenon can be observed, for instance, in solutions 4.2d and 4.2e.



(a) $\nu = 1$ (b) $\nu = 0.6$ (c) $\nu = 0.3$

(d) $\nu = 0.6$ (e) $\nu = 0.6$ (f) $\nu = 0.3$

(g) $\nu = 0.3$ (h) $\nu = 0.3$ (i) $\nu = 0.3$

Figure (4.2) Examples of solutions that can be obtained varying the viscosity: only the streamwise component of the velocity is represented

4.2. Results with a single parameter

In this section the results that can be obtained with a single varying parameter will be shown and, since our test case is very similar to the model described in [37], we expect very similar results.
As said in section A.2, the Reynolds number is the only non-dimensional number that is present in system (4.1) after the normalization, and for this reason it is responsible for the bifurcations. Let us remember that it is defined as $Re = \frac{UL}{\overline{\nu}}$, where U, L and $\overline{\nu}$ respectively represent the characteristic velocity, length and viscosity. In this section the varying parameter will be the kinematic viscosity. Note that, if U and L are constant, the variation of Re is simply the inverse of the variation of ν and that it is thus possible to observe different solutions using such a parameter.
The aim of this work is to develop an efficient method to compute an entire bifurcation diagram, however, in this section, we will only focus on the methods that allow one to compute such a diagram both in the offline and in the online phase. In fact, even if the computation of the diagram in the online phase is very efficient, it is not possible to obtain it without computing almost the same diagram in the offline one. Let us suppose that one wants to exploit the described methods only in the online phase, then the reduced space would be generated by snapshots belonging to a single branch. In the test case described in the previous section the only branch that can be found without the deflation method is the symmetrical one and, therefore, the reduced solutions are linear combinations of symmetrical fields. It is thus clear that the reduced space is not rich enough and the remaining branches can not be obtained with the deflation method.
To avoid such an issue, one has to obtain, in the offline phase, solutions belonging to every branch that will be reconstructed in the online one. Therefore, when a single parameter is involved in the computation of the diagram, the entire bifurcation diagram has to be computed in both the phases. However, even if it is not efficient, we performed such a task in order to empirically prove that the continuation method and the deflation one can be used to obtain the entire diagram in the reduced framework. On the other hand, in section 4.3 the problem of the efficiency will be handled exploiting in a more clever way the offline-online splitting, even if all the needed techniques will already have been covered in this section and in the previous

chapters. In such an approach one computes few bifurcation diagrams with a single parameter in the offline phase, while the entire diagram is computed only in the online one. However, this difference will be more deeply explained in section 4.3.1.

4.2.1. Offline phase

4.2.1.1. Deflated continuation method

In order to efficiently compute a bifurcation diagram in the online phase, a reduced space that already contains enough information to reconstruct every branch is needed. To construct it, all the different branches have to be followed also in the offline phase, this way solutions belonging to different branches can be then grouped together to generate a rich enough reduced space. In this section a method that can be used to compute solutions belonging to different branches will be discussed. It can be noted that, in order to compute such a set of solutions in an efficient way, it is better to directly compute a bifurcation diagram with the described techniques rather then computing a set of isolated solutions. In fact, if one wanted to compute several different solutions without computing a diagram, one should be able to obtain many solutions on each branch (otherwise the reduced space would not contain enough information) and should use many different initial guesses that would not be easily obtained.
In sections 3.2 and 3.3 the continuation and deflation methods have been presented. They are very useful when one wants to obtain local informations about known or unknown branches of a bifurcation diagram but, in order to entirely obtain such a diagram, they have to be used together in a coordinate way. The proposed technique to bind the two methods is called *deflated continuation* and it is discussed in [28]. The idea behind the method is to initially compute a first solution, that will belong to the first branch, and then use it as initial guess for the continuation method to obtain that branch entirely. Moreover it is important to use the deflation method after each step of the continuation one in order to discover new branches that will be followed, with the continuation, in parallel.
Unfortunately, we observed that the deflation method works fine only if an unknown branch is close to a known one. Let us assume that every already computed solution belongs to the branch $\mathcal{B}_0$ and no informations about other branches are available. If one wants to use the deflation method to find the branch $\mathcal{B}_1$, one should use the iterative solver on the deflated system using as initial guess a constant field or a perturbation of an existent solution. Actually, these two approaches are equivalent, in fact, in general, if the first solution on $\mathcal{B}_0$ has been obtained starting from a constant solution, the iterative solver would always converge to pressure-velocity pairs in $\mathcal{B}_0$ with a similar initial guess. This way, after n steps of the iterative solver, the actual iteration would be very close to $\mathcal{B}_0$ and can be thus considered as a perturbation of a solution on that branch. The problem is that, as remarked in sections 1.3.1 and 3.3.2.1, the iterative solver requires an initial guess close enough to the sought solution in order to converge to it and, if any solution on $\mathcal{B}_1$ is too far from $\mathcal{B}_0$, it would not be able to converge simply because the initial guess is very bad.
In order to solve this issue it is important to use the deflation at each step of the continuation method, in fact, under the assumption that only pitchfork or transcritical bifurcations are present (see section 3.1), right after the critical point the different branches are very close and it is

possible to converge to solutions belonging to a branch different from the one used to obtain the initial guess.
The pseudo-code of the deflated continuation, even if in a very simplified version, is represented below:

```
//First solution
ν = ν_0;
Computation of u_0: a solution of L(u; ν) = 0;
S = {u_0}  // Set of solutions to be continued

for(int i = 1; i < N_max; i++)
{
        // Continuation of known branches
        S_tmp = ∅  // Set of solutions to be deflated and continued in the next
                   ↪ step
        G(·; ν) = L(·; ν)  // The deflated system is the undeflated one
        for(int j = 0; j < S.size(); j++)
        {
                Computation of ũ_i using S[j]; // Simple continuation
                ν = ν + Δν;  // Parameter update
                Computation of u_i: a solution of G(u; ν) = 0;
                if the solver converged to u_{i,j}
                {
                        S_tmp = S_tmp ∪ {u_{i,j}};  // S_tmp update
                        G(·; ν) = (1 + 1/(‖· − u_{i,j}‖)) G(·; ν);  // G update
                }
        }
        // Deflation method
        for(int j = 0; j < S_tmp.size(); j++)
        {
                Computation of u_i: a solution of G(u; ν) = 0;
                if the solver converged to u_{i,j}
                {
                        S_tmp = S_tmp ∪ {u_{i,j}};  // S_tmp update
                        G(·; ν) = (1 + 1/(‖· − u_{i,j}‖)) G(·; ν);  // G update
                }
        }
        // Setup for the next continuation step
        S = S_tmp;
}
```

The presented version is a simplified one for many different reasons, however, we will analyze only the ones related to the numerical methods. Firstly, the aim of the pseudo-code was only to represent the logical structure of the method, but great care must be taken to manage the different data structures, especially if a better continuation method is used. Secondly, as remarked in section 3.3.2.1, it is important to modify only the residuals of each Newton iteration instead of the entire linear problem as written in the pseudo-code. Thirdly, in order to exploit as much information as possible from the data, it would be advisable to use a better continuation method, like the pseudo-arclength continuation method, even if it would slow down the computation of each single solution. In fact, as explained in section 3.2.2, with the pseudo-arclength continuation the computation of the next initial guess is more expensive, but one can afford bigger steps, reducing also the overall needed amount of time and memory. The latter is an important issue for the algorithm, because all the solutions have to be stored in order to construct the reduced space, and a lower number of solutions implies lighter data structures. Furthermore, the time needed to generate all the matrices and the vectors that will be used in the online phase increases if a bigger number of snapshots is available. However, the reduced space dimension is almost the same in both the cases, because it depends by the information contained in the snapshots and not by their number, therefore a longer computation does not imply a better result.
Another downside of using smaller steps is that the deflation can be considered the bottleneck of the deflated continuation method and, if one wants to use it at each steps, it is better to use bigger steps in order to avoid wasting too much time in useless iterations in the portion of the method related to the deflation. However, it should be noted that, with the pseudo-arclength continuation, two different solutions are required to obtain each initial guess, therefore the data structure has to be properly modified or adapted in order to be able to recognize which are the last two solutions on each branch (we remember that the number of branches is not known a priori). On the other hand, the pseudo-arclength continuation method can not be used at each step because two solutions on the same branch could not be available. In these situations, that mostly occur at the first two steps where 0 or 1 solutions are available, or after a bifurcation point where the number of known branches changes in two consecutive steps, the simple continuation method is used.
Moreover, it can be observed that the deflation is used also in the continuation phase, this is useful when two branches are very close to each other, for instance after a bifurcation point. In fact, if two branches are very close, it is possible that the iterative solver converges to a solution belonging to a branch even if the initial guess is a numerical approximation of a solution on the other one. On the other hand, if the branches are very far, the same solution can be obtained solving the deflated or the undeflated system, thanks to the property of the deflation function.
It should be noted that, even if we said that the deflation can converge only if it is used in a little neighbourhood of a bifurcation point, we also suggested to use a continuation method that allows very long steps. Unfortunately, they could be so long that, at the j-th step, the critical point could be between $(\nu_j, \mathbf{u}_j)$ and $(\nu_{j+1}, \mathbf{u}_{j+1})$ but far from both. In order to face this new problem we decided to limit the quantity $\Delta\nu_j$ with an upper bound $\Delta\nu_{max}$. However, since we did not want to lose the accuracy obtained with the pseudo-arclength continuation method, we modified it in the following way. Note that the same notation as in section 3.2.2 will be used,

even if the abstract parameter μ will be substituted by the viscosity ν. Therefore, the system that one wants to solve is the following one:

$$\begin{bmatrix} L^i_{\mathbf{u}} & L^i_\nu \\ \dot{\mathbf{u}}_i & \dot{\nu}_i \end{bmatrix} \begin{bmatrix} \Delta\mathbf{u}_i \\ \Delta\nu_i \end{bmatrix} = \begin{bmatrix} L^i \\ \Delta S_i \end{bmatrix}. \tag{4.3}$$

When the resulting $\Delta\nu_i$ is less than $\Delta\nu_{max}$ no correction is needed, otherwise one can fix $\Delta\nu_i = \Delta\nu_{max}$ removing it from the unknowns, and then reconstruct the residual of the velocity with the formula:

$$L^i_{\mathbf{u}}\Delta\mathbf{u}_i = L^i - L^i_\nu\Delta\nu_{max}. \tag{4.4}$$

It should be noted that, in order to know if $\Delta\nu_i > \Delta\nu_{max}$, one has to solve system (4.3) and, if it is solved with the bordering algorithm (section 3.2.2.1), two linear systems associated to the same matrix $L^i_{\mathbf{u}}$ have to be solved. Then, solving equation (4.4), the same system is solved for the third time; it is thus convenient to use a proper matrix factorization in order to increase the efficiency of this phase. We remark that the same approach can be used to impose a lower bound $\Delta\nu_{min}$ to the step size, it can be useful when, due to the presence of a bifurcation, the chosen steps are so small that the method get stuck in its neighbourhood.
Finally, if some information about the position of the bifurcation points is available, it is clever to exploit it. Let us assume that the first critical value is associated to $\nu = \nu_0$ and that this information is at least approximately known, since the deflation is the bottleneck of the deflated continuation method, it is uselessly expensive to use the deflation before such a value. In fact it is impossible to converge to new branches because they do not exist, while, on the other hand, using it around the critical value (or right after it if one is sure about the the value of ν_0) will increase the probability of discovering new solutions. Similarly, if the nature of the bifurcation is known, and therefore one knows that only n branches exist, the deflation phase can be avoided after having found exactly n different solutions.

4.2.1.2. A simple heuristic to improve the deflation method

In section 3.3.2.1 we proved that it is possible to use the deflation simply acting on the residual of the Newton method but, unfortunately, we observed that in our test case the iterative solver on the deflated system was too slow and this ruined its convergence. We thus decided to implement a heuristic, based on the results in section 3.3.2.1, to improve the efficiency and the effectiveness of such a method. In order to describe it, we briefly summarize the used notation:

$$L(\mathbf{u};\nu) = 0, \tag{4.5}$$

is the initial problem that has to be solved. Let us assume that it has been solved once, obtaining the solution $\mathbf{u}_0$; we can thus define the deflated problem multiplying the previous one by a deflation function $\mathcal{M}(\mathbf{u};\mathbf{u}_0)$:

$$G(\mathbf{u};\nu,\mathbf{u}_0) = \frac{1}{\|\mathbf{u}-\mathbf{u}_0\|^p_U} L(\mathbf{u};\nu) = \mathcal{M}(\mathbf{u};\mathbf{u}_0)L(\mathbf{u};\nu) = 0, \tag{4.6}$$

where U is a subspace of the space where the solution is sought but, in general, can be a generic normed space. The solution set of problem (4.6) is the same of the undeflated one but without

$\mathbf{u}_0$ therefore, computing a solution of it, is equivalent to obtain a second solution $\mathbf{u}_1$ of (4.5). The deflated problem can thus be generalized in the following way assuming that n solutions $\mathbf{u}_0, ...\mathbf{u}_{n-1}$ of (4.5) are obtained using each time a different deflated system:

$$\begin{aligned} G(\mathbf{u}; \nu, \mathbf{u}_0, ..., \mathbf{u}_{n-1}) &= \frac{1}{\|\mathbf{u} - \mathbf{u}_{n-1}\|_U^p} \cdot ... \cdot \frac{1}{\|\mathbf{u} - \mathbf{u}_0\|_U^p} L(\mathbf{u}; \nu) \\ &= \mathcal{M}(\mathbf{u}; \mathbf{u}_0, ..., \mathbf{u}_{n-1}) L(\mathbf{u}; \nu) = 0. \end{aligned} \tag{4.7}$$

Using, for example, the Newton method, problems (4.5) and (4.7) can be respectively discretized as follows:

$$J_L \Delta \mathbf{u}_L = -L, \tag{4.8}$$

$$J_G \Delta \mathbf{u}_G = -G. \tag{4.9}$$

Finally, it has been proved (see section 3.3.2.1) that constructing J_G and G at each iteration is not necessary, since solving (4.9) is equivalent to solve (4.8), compute the scalar quantity τ and directly obtain $\Delta \mathbf{u}_G$ as:

$$\Delta \mathbf{u}_G = \tau \Delta \mathbf{u}_L. \tag{4.10}$$

Finally, we remember that τ can be efficiently computed with the following formula, where all the dependencies are omitted for brevity:

$$\tau = \frac{1}{1 + \mathcal{M}^{-1} \mathcal{M}'^T J_L^{-1} L}. \tag{4.11}$$

The problem is that, in our test case, $|\tau|$ was, sometimes, very small, about 10^{-7} or less, implying that $\|\Delta \mathbf{u}_G\| \approx 0$. This way the iterative solver got almost blocked for several iterations on fields that were always very close to each other but that were not solutions of the system. We also noted that, even changing the value of p, this behaviour was always present. In order to improve the method we observed that, since τ is a scalar quantity, it can not modify the direction of the vector $\Delta \mathbf{u}_L$ but only its magnitude or its orientation (when $\tau < 0$). Assuming that the iterative solver is converging to an already found solution $\mathbf{u}_i$ of problem (4.5), $\tau = 1$ means that the deflation is not doing anything, $\tau > 1$ (or $0 < \tau < 1$) means that the steps are bigger (or smaller) than usual towards $\mathbf{u}_i$, and finally $\tau < 0$ means that the deflation is pushing the iterative solver away from $\mathbf{u}_i$.
It is thus essential to modify τ while maintaining its sign. The first modification to avoid very small values of τ is to fix a threshold τ_t such that, if $|\tau| < \tau_t$, the value of τ is automatically changed in $\pm\tau_t$, with the sign chosen according to the original one. Actually, we observed that it was better to split such a condition in two different conditions with the positive and negative thresholds $\tau_t^+ = 0.6$ and $\tau_t^- = -0.4$ in order to use, then, the following expression to initially set τ:

$$\tau = \begin{cases} \tau_t^-, & \text{if } \tau > \tau_t^- \\ \tau_t^+, & \text{if } \tau < \tau_t^+ \\ \tau & \text{otherwise} \end{cases}.$$

Moreover, we wanted, as much as possible, to avoid setting τ with these constant thresholds, because they are not adaptive or problem dependent. Instead, we preferred to use a strength

factor to modify τ maintaining at least a proportionality between the used value and the one obtained with the formula (4.11). Analyzing the sign of τ, we understood that sometimes it could be negative, then become positive and, after some iterations, return negative again. We interpreted this behaviour as follows: at the beginning, when $\tau < 0$, the current iteration is too close to a known solution $\mathbf{u}_i$, in these iterations the deflation is pushing the solver away from $\mathbf{u}_i$. However, when it becomes positive, it means that the deflation function does not consider $\mathbf{u}_i$ as too close. Unfortunately, the solver could still converge to $\mathbf{u}_i$ if the current iteration is not far enough from it, this way the subsequent iterations would be closer and closer to $\mathbf{u}_i$, until τ would eventually become negative again. This problem occurs because the steps between two subsequent iterations are too small, and it is due to the fact that, if τ is too small, the threshold is used to modify τ, but also the threshold is very small. Unfortunately, considering a bigger threshold would destroy the stability of the solver. We thus decided to multiply τ with a strength factor $c \geq 1$ obtaining the following expression:

$$\tau = \begin{cases} \tau_t^-, & \text{if } c\tau > \tau_t^- \\ \tau_t^+, & \text{if } c\tau < \tau_t^+ \\ c\tau & \text{otherwise} \end{cases}.$$

In order to properly set c we used an iterative approach: initially, it is set to 1 but then, each time τ becomes positive, c is multiplied by 1.75. This way, it is more and more likely to obtain a value of τ such that $c\tau > \tau_t^+$ or $c\tau < \tau_t^-$ so that the actually used value in equation (4.10) is not related to the thresholds and the steps are bigger and bigger. This approach solves the previous problem but it could happen that the steps between two iterations are so big that the solver could diverge.
If the current iterations are very far, in norm, from any solution, but τ is negative and the previous iteration was not that far from any other solution, it means that the last $|c\tau|$ factor was too big. Therefore, the last update is ignored and the next one is computed iteratively dividing the strength factor by 2 in order to obtain a better field. Otherwise, if τ is positive or the previous iteration was already very far from other solutions, the linearization method is changed from the Newton method to the Oseen one. It can be useful because the Newton method can converge only if the initial guess is close enough to the solution, while the Oseen one can slowly converge from further initial guesses. However, the Oseen method is very slow, therefore we decided to switch back to the Newton method when, with the Oseen iterations, the solver is converging and the distance between two subsequent iterations is small enough.
In any case, if the iterative solver has not converged in 150 iterations, we decide that it will not converge because the other solutions are too far or they do not exist. We thus move on to the next initial guess in the deflation phase or the next value of ν to follow the known branches.

4.2.1.3. Bifurcation diagram analysis

In this section the nature of the snapshots and the effectiveness of the described techniques will be analyzed. To do it, we will show the associated bifurcation diagram, this way it will be possible to discuss the obtained result and understand the different features of solutions belonging to different branches.

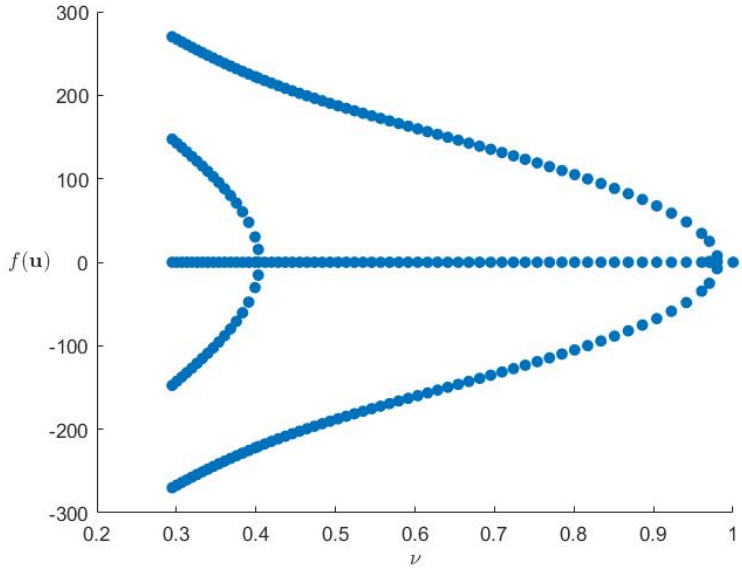


Figure (4.3) Bifurcation diagram obtained in the offline phase. Each point represent a single SEM solution.

In order to compare our results with the ones present in [29], we decided to use the same function to obtain the bifurcation diagram, that is defined as:

$$f(\mathbf{u}, p) = f(\mathbf{u}) = \pm \int_{\Omega} \|\mathbf{u} - \mathcal{R}\mathbf{u}\|^2, \tag{4.12}$$

where the sign is chosen according to the fact that the jet is closer to the upper wall or to the lower one. It should be noted that here $\mathbf{u} = (u, v)$, the variable $\mathbf{u}$ does not take into account the pressure as in the previous sections and, for this reason, only the velocity is involved in the computation of the bifurcation diagram. Furthermore, the velocity field $\mathcal{R}\mathbf{u}$ is obtained reflecting $\mathbf{u}$ through the horizontal symmetry axis. This way, the obtained output can be interpreted as a measure of the asymmetry of the solutions, if $\mathbf{u}$ is perfectly symmetric $f(\mathbf{u}) = 0$, while, on the other hand, when the velocity fields $\mathbf{u}$ is significantly asymmetric, $|f(\mathbf{u})|$ is very high.
The obtained bifurcation diagram is shown in figure 4.3, it contains two pitchfork bifurcations and five different branches. The diagram is obtained with an initial solution associated to $\nu = 1$ computed simply using a constant initial guess, then the initial branch is followed with the continuation method decreasing ν and the deflation is used to obtain the initial point on new branches.
It can be observed that the points are closer to each other for low values of ν, while they are farther and farther as the parameter increases. This is due to the fact that the Reynolds number increases when ν decreases, therefore the computation of each solution is more complex for low viscosities and using a better initial guess (with smaller step sizes) improves the stability of the method. Moreover, it should be noted that the symmetrical solutions are always very similar, while the asymmetric ones can significantly vary even for small variations of the parameter. This implies that the step sizes chosen by the continuation method for the symmetric branch

are much bigger than the ones for the other branches. If one is only interested in obtaining the branches this is not an issue, but it can ruin the deflation method effectiveness, in fact it is not possible to deflate an entire branch but only single solutions. In order to better understand the problem, let us assume that the branches $\mathcal{B}_0$ and $\mathcal{B}_1$ have already been found, let us denote as $(\nu_0, \mathbf{u}_0)$ the last obtained solution pair on $\mathcal{B}_0$ and with $(\nu_1, \mathbf{u}_1)$ and $(\nu_2, \mathbf{u}_2)$ the last two solution pairs on $\mathcal{B}_1$. Finally let us assume that $\nu_2 > \nu_0 > \nu_1$ and that the Newton method is used with a perturbation of $\mathbf{u}_0$ as initial guess and with $\nu = \nu_0$. Since a solution $\mathbf{u}_3$ on $\mathcal{B}_1$ associated to $\nu = \nu_0$ is not know, it can be obtained by the Newton method even if the deflation operator is built using both $\mathbf{u}_1$ and $\mathbf{u}_2$ because it is a solution of the deflated system. To avoid such a problem, while maintaining the accuracy of the pseudo-arclength continuation and meaningful step sizes, we decided to use on every branch the minimum step size chosen by the continuation of all the known branches. To do it, one can use a technique similar to the one presented in section 4.2.1.1 to impose a maximum step size.
Moreover, it should be noted that, in the described test case, there exist a limited number of solutions for any value of the parameter but such a property does not always hold true. For instance, if one considers the three-dimensional and axisymmetric version of the domain described in section 4.1, where the two rectangles are substituted by two cylinders, infinite different solutions exist. In fact, given a solution $\widetilde{\mathbf{u}}$ of the problem, it is always possible to generate infinite different solutions rotating $\widetilde{\mathbf{u}}$ around the symmetry axis. Therefore, one has to modify the deflation method to deflate an entire group of solutions at the same time; the easiest way to do it is to modify the deflation function as described in [9]. However, imposing the same step sizes on all the branches is still required to effectively deflate the known solutions.
Since the first obtained solution is computed using $\nu = 1$ and then the viscosity is slowly decreased, we will refer to the bifurcation point associated to $\nu \approx 0.970$ as the first bifurcation point, while the one for $\nu \approx 0.395$ as the second one. Then, to better interpret the diagram in figure 4.3, it is convenient to relate each branch to the solutions shown in figure 4.2. Let us consider the central branch: for each solution $\mathbf{u}$ belonging to it $f(\mathbf{u}) = 0$, therefore it can be said that those solutions are symmetrical. In fact the solution 4.2a is associated to $\nu = 1$, 4.2c to $\nu = 0.3$ and 4.2b is a representative solution of the central branch for a viscosity value between the two critical values. These solutions are symmetrical and it should be noted that with lower viscosities (or better higher Reynolds numbers) the jet becomes narrower but longer; this phenomenon is reasonable thinking about the Reynolds number effects as follows. Since $Re = \frac{UL}{\nu}$, the same qualitative behaviour could be obtained fixing a proper value of ν and increasing the velocity maintaining Re in the same range of values as before. In that case the values of the velocity in each point would be different, but it would be easier to visualize the fact that the jet elongates and restricts increasing the velocity (and therefore the Reynolds number).
Then, let us consider the upper branch that arises from the first bifurcation point, the associated solutions are asymmetric and their asymmetry grows more and more because $|f(\mathbf{u})|$ increases when ν is decreased; two representative solutions on it are in figures 4.2d and 4.2i. It can be observed that at the beginning only the first part of the jet is asymmetric, while for higher Re the asymmetry propagates throughout the entire channel. It can be noted that solutions 4.2e and 4.2h are the reflections of the previous ones and, in fact, they belong to the lower branch that detaches from the first bifurcation point. Finally, solutions 4.2f and 4.2g can be seen as

representatives solutions for the remaining two branches (respectively the upper and the lower one). Let us observe that the asymmetry arises closer to the outlet because the Reynolds number is higher.

4.2.1.4. Choice of the technique used to construct the reduced space

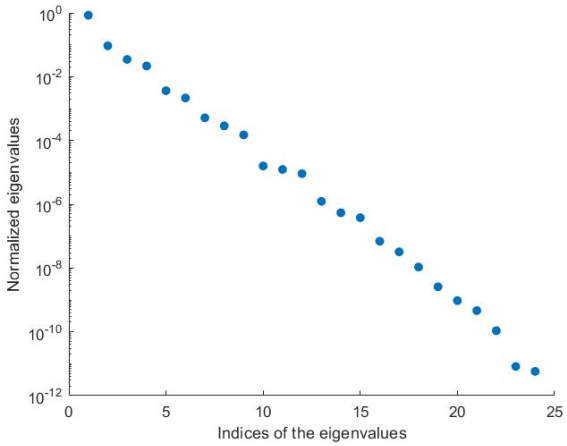
In order to perform the offline-online splitting it is important to compute, during the offline phase, all the required vectors and matrices that allow one to efficiently solve the same problem in a reduced space. Such a space can be generated with different methods, like the POD method (see section 2.2) or the greedy algorithm (see [38]). In the latter approach, one iteratively updates the snapshots set adding, at each step, a new snapshot that is selected looking for the worst approximated solution with the current reduced space. If the method is optimized and an error estimate is available in order to efficiently obtain the reduced space, it is faster than the POD method. However, the two techniques converge to the same space only when the reduced spaces sizes tend to infinity. Unfortunately, the greedy algorithm can not be easily used when the aim is to compute an entire bifurcation diagram, in fact one should be able to face at least two different problems. The first one is related to the fact that more than one solution can be associated to a single value of the parameter, therefore the error estimate should be modified to be able to catch the error on each branch. Secondly, in order to obtain the bifurcations, the parameters have to be above a critical threshold but, if the parameters are in a range where strong instabilities occur, the solver could diverge because of the absence of a good initial guess (in this work obtained with the continuation method). Therefore, the greedy algorithm should be properly modified in order to face the first problem, a continuation method could be useful to reach the needed values of the parameter (but it would involve the computation of many full order solutions) while some other method should be used to obtain solutions on different branches in regions far from the bifurcation points.
On the other hand, the deflated continuation method seems to be perfectly adapted to the POD method, in fact it allows one to have snapshots belonging to all the different branches and it can ensure good initial guesses for the Newton method to enable it to converge even for higher Reynolds numbers. All the snapshots can then be grouped together to generate, using the POD method, a global reduced base. It is called global because one can choose to use a local reduced basis approach, this topic will be analyzed in Chapter 5.
Finally, since the POD method is based on the analysis of the eigenvalues of the snapshots correlation matrix, it is interesting to observe their decay. In order to obtain a general result we considered four different scenarios: in the first one, that will be called the reference setting, we computed 24 snapshots belonging to the first three branches and using the constraint that, in the continuation method, $\Delta\nu_i < \Delta\nu_{max} = 0.02 \cdot \nu$. With the reference setting we obtained snapshots associated to values of ν in the interval $[0.85, 1]$, the corresponding decay of the eigenvalues of the correlation matrix is shown in figure 4.4a. This way, it can be noted that all the snapshots are in a small neighbourhood of a bifurcation point but, even if the reduced order model performs worse in such a region, the eigenvalues decay is exponential as expected [32]. The eigenvalues in the plots are normalized over the sum of all the eigenvalues and, this way, it can be seen that the first eigenvalue is always very close to 1 even if the sum of the normalized eigenvalues is exactly 1. This property explains the fact that the first POD mode contains most of the information

contained in the snapshots. Furthermore, an exponential decay can be clearly observed, it is a very useful property because this way, as explained in section 2.2, one can fix a threshold to select the best dimension for the reduced space and generate a very small subspace that contains almost all the information of the solution manifold.
In figure 4.4b the step sizes in the continuation method are smaller, this way all the snapshots are closer to the bifurcation, in fact they are associated to viscosities in the interval $[0.91, 1]$. However, it can be noted that the decay is qualitatively the same as in figure 4.4a, therefore the step sizes in the continuation method do not deeply influence the result. This can be considered an expected property since the eigenvalues decay depends by the manifold that is approximated by the snapshots and the continuation step sizes do not change it. On the other hand, if such a manifold changes, for example when only one of the three branches (the symmetric one in the plot) is considered, the decay is very different. Such a phenomenon can be seen in figure 4.4c: the decay is much faster even if with 24 snapshots the viscosity reached the value 0.64 and therefore its range is wider. A faster decay means that the reduced space dimension will be much lower, it is reasonable since it can summarize the needed information in less snapshots since they are all similar. Finally, in figure 4.4d, the reference settings are considered, but with 100 snapshots instead of 24. The range of the viscosity is $[0.51, 1]$, therefore the manifold is different from the one discretized with the reference setting, but the decay is much slower, in fact the 24-th normalized eigenvalue is around 10^{-6} instead of 10^{-11} as in figures 4.4a or 4.4b.

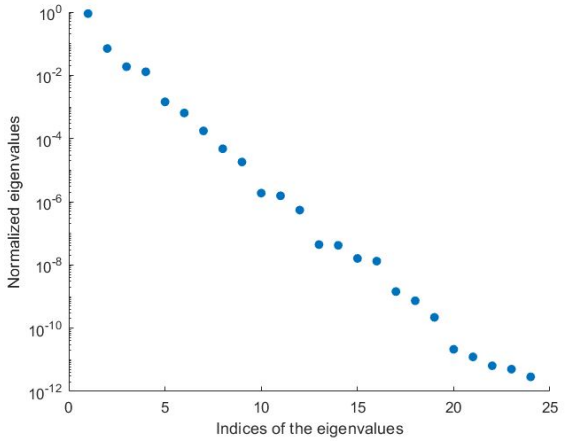
4.2.2. Online phase

4.2.2.1. Reconstructed bifurcation diagram

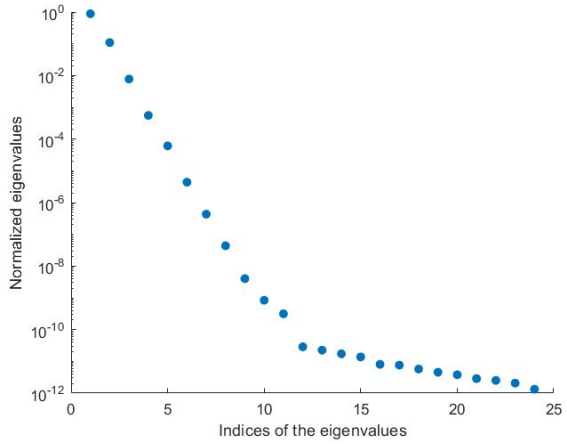
In this section our aim is to use the reduced space built in the offline phase to quickly and accurately reconstruct the entire diagram described by the snapshots. The main idea used to obtain the diagram is very similar to the one used in the offline phase: the continuation method will be used to follow the first branch, then the first solutions on new branches will be computed with the deflation, and finally these new branches will be entirely tracked with the continuation method.
The differences between the offline and online deflated continuations are mainly three. Firstly, the first solution in the online phase is obtained starting from the projection of the corresponding full order solution on the reduced space as initial guess. This is very important because one of the main weakness of the global basis with snapshots coming from different branches is the stability, this way a very good initial guess is used for the first solution, while the initial guesses for the subsequent solutions are obtained with the continuation method. Therefore, every solution is very close to the initial guess used to compute it and the iterative solver can converge to it even if the global basis is noisy. Secondly, since in the online phase the computation of a solution is a very cheap operation, we decided to use the simple continuation (see section 3.2.1) with small step sizes. To have reasonable values for each $\Delta\nu_i$ we refined the offline parameter grid adding, between each pair of subsequent parameter values used in the offline phase, a fixed number of points (usually between 2 and 20). This way the step sizes are smaller near the bifurcation points because in those regions the steps used in the offline phase were very small, while they are bigger otherwise. We remark that with this approach one of the main weakness



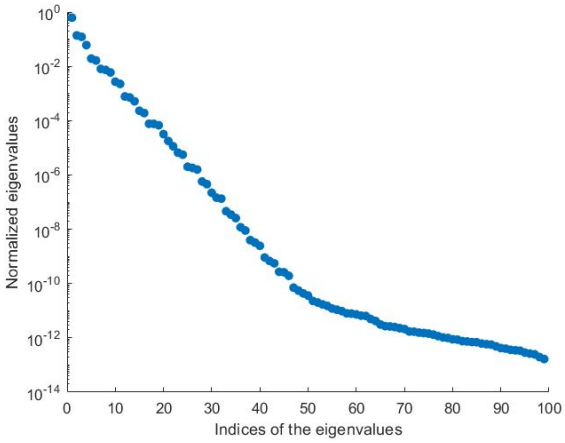
(a) Reference settings



(b) Reference settings but with $\Delta\nu_{max} = 0.01 \cdot \nu$



(c) Reference settings but with all the snapshots belonging to the same branch



(d) Reference settings but with 100 snapshots

Figure (4.4) Eigenvalues decay obtained with different settings of the model. The reference settings are: 24 snapshots belonging to three branches with the constraint $\Delta\nu_i < \Delta\nu_{max} = 0.02 \cdot \nu$ for the continuation step sizes.

of the simple continuation can be solved because the steps become proportional to the steps chosen by the pseudo-arclength continuation used offline. Thirdly, we observed that the deflation method works fine also in the reduced framework, we thus modified it only slightly to improve the performances. The first change is due to the fact that with some settings the solver could converge in only one or two iterations, and the deflation did not have enough time to avoid the convergence, therefore we used as initial guess in the deflation phase the zero field to slow down the convergence. Moreover, we observed that the deflation was more stable than in the offline phase but that it still needed to be associated to a heuristic, even if it is simpler than the one discussed in section 4.2.1.2. We maintained the strength factor c and the thresholds even if the values used to update c and as thresholds have been empirically modified to prevent the solver to immediately diverge. On the other hand, since the continuation steps are very small and the computation of a solution very cheap, we decided to always use the Newton method and avoid the process of iteratively dividing τ by 2 when the solver diverged.
The obtained bifurcation diagram is shown in figure 4.5, it can easily be noted that the reduced model is able to capture all the different branches and entirely reconstruct the diagram even if it includes some bifurcation points. This specific diagram has been obtained with a POD tolerance equal to 0.999, the dimension of the reduced space was 37, while the snapshots are computed using polynomials of order 10 in each element. The accuracy of the solutions obtained in this phase is proved by an empirical error analysis, that has been performed comparing each online solution with the full order field obtained using that specific solution as initial guess; the error is shown in figure 4.6. It can be observed that, even if the reduced order model is able to capture the different natures of the branches, the error increases in the neighbourhood of the bifurcation points. This phenomenon is due to the fact that in the inner part of each branch a small variation of the parameter implies a small variation of the solutions, while this does not happen immediately after a bifurcation point. It can also be noted looking at the bifurcation diagram, the almost vertical part of the asymmetric branches represents the fact that there are few solutions that are only slightly asymmetric, but these fields can not be approximated as well as the other ones because very few snapshots are similar to them. In the error plot each line represent a different branch, it can be noted that the mirrored branches have the same error, while the symmetric one is often represented more accurately. Moreover, the solutions in the online phase have been computed with the tolerance of the iterative solver equal to 10^{-6}, it can be observed that the average error is very close to it, even if, after the bifurcation points, the accuracy decreases of a couple of order of magnitude.
Moreover, the oscillations are due to the fact that the reduced base is able to more faithfully reproduce fields that are very close to the most important snapshots. Such a behaviour can be more easily explained using the greedy algorithm because in that case the base is generated by solutions, but it can be observed also using the POD method, since the informations contained in the reduced spaces obtained with the two approaches are similar.
Lastly, it is interesting to observe that the error decay over the reduced space dimension is exponential as expected, this phenomenon can be observed in figure 4.7. Such a plot has been obtained computing several solutions in a neighbourhood of the first bifurcation point ($\nu \in (0.85, 1)$), this is important because, as observed in figure 4.6, such a region is the one where the error is higher. In order to obtain a reliable result, we set both the offline and online iterative solver tolerances

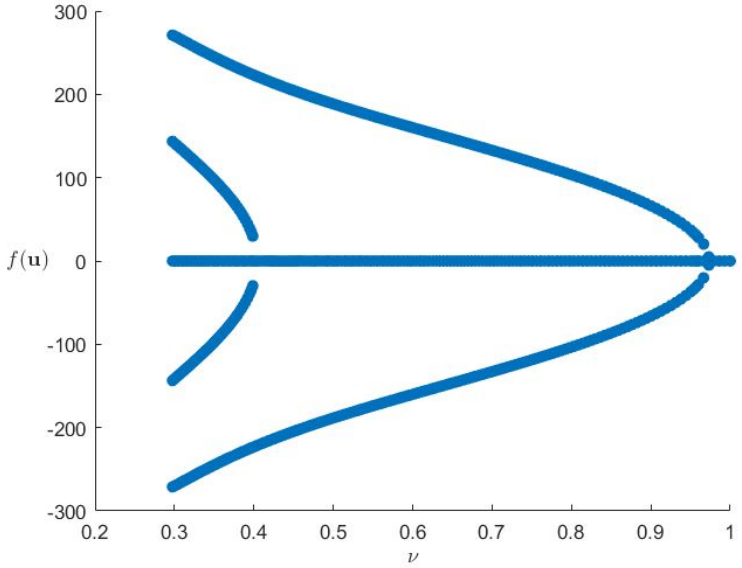
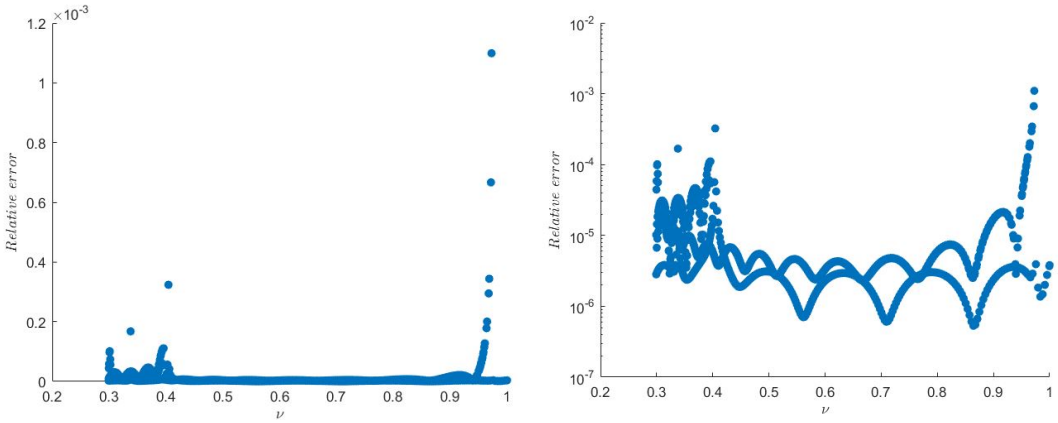


Figure (4.5) Bifurcation diagram efficiently obtained in the online phase.

equal to 10^{-10}, this way one can recognize that the exponential decay is present only when the error is much greater than such a tolerance, while it remains constant when its magnitude is comparable with such a value. Moreover, the maximum and the average relative errors, that have been computed over 138 different solutions, are very close to each other and exhibit the same behaviour. This can be interpreted saying that the error decay has to be exponential also in smaller neighborhoods of the bifurcation point, in fact, if the closest solution to the critical point was associated to a bigger error that would not exponentially decrease, the maximum error would be much bigger of the average one.

4.2.2.2. Unphysical branches

Even if we proved that it is possible to use the reduced order methods to reconstruct an entire bifurcation diagram, many different parameters are required to properly set the model up. The most important one is the tolerance used by the POD method to choose the dimension of the reduced space, unfortunately a general technique to set it in the best way is not known. The problem is that if the tolerance is too low the reduced space is too small and the different branches can not be reconstructed properly. In such a case, in the deflation steps, the solver could always diverge or it could converge only far from the bifurcation point, but, on the other hand, a too strict tolerance can imply several issues.
Firstly, if the reduced space is generated by too many snapshots it is very noisy, leading to serious stability issues, in the worst scenario the solver is not able to converge even starting from very good initial guesses. Secondly, the online matrices are full, therefore the resolution of the linear system can become very expensive. Finally, if the tolerance is not high enough to destroy the convergence but the reduced base is still very noisy, unphysical solutions can be found by the solver. When these solutions are associated to random points in the bifurcation



(a) Uniform scale (b) Logarithmic scale

Figure (4.6) Error with respect to full order solutions relative to the solutions in figure 4.5

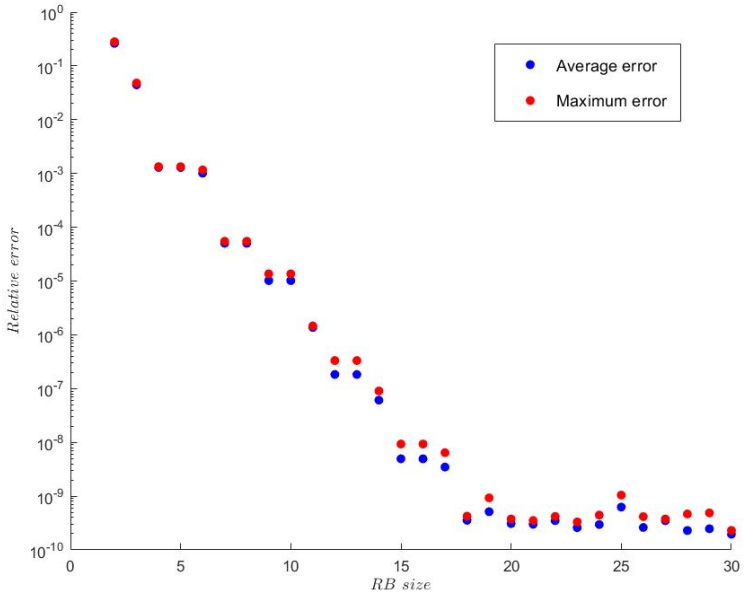
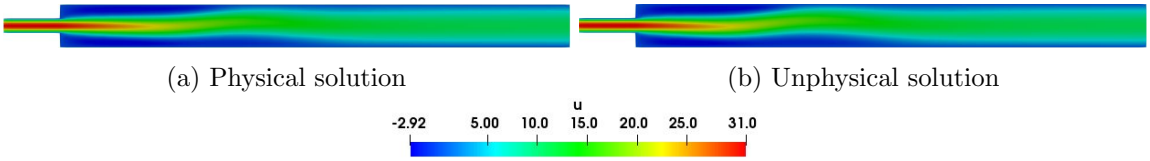


Figure (4.7) Maximum and average relative errors over the reduced space dimension. A clear exponential decay can be observed.



(a) Physical solution (b) Unphysical solution

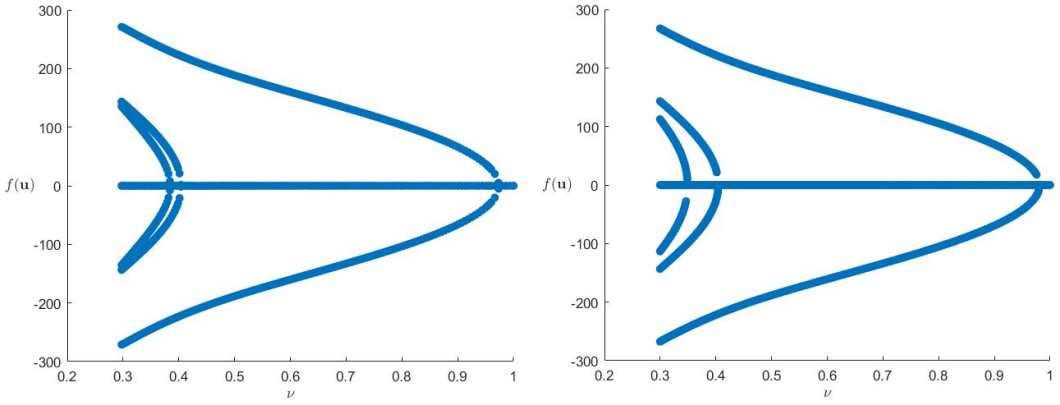
Figure (4.8) Examples of solutions belonging to the upper branches starting from the second (physical solutions) and third (unphysical solutions) bifurcations points in diagram 4.9a.

diagram or they look unphysical, it is easy to understand that they are due to numerical issues and they have to be discarded. However, sometimes their associated points in the bifurcation diagram are grouped like in a standard branch and they look coherent with the equation and the boundary conditions. Therefore, it is not always possible to easily understand if a solution is physical or not simply looking at it. The visualization of two different but similar solutions that have been obtained in the online phase are represented in figure 4.8, one of them is physical while the other one is not. It can be observed that the two solutions are associated to the same viscosity and that they are almost indistinguishable, the only visible difference is that the unphysical jet (figure 4.8b) is a bit narrower than the physical one (figure 4.8a), but it is impossible to say which one is feasible simply looking at them.
Such a phenomenon can also be observed in the bifurcation diagrams in figures 4.9a and 4.9b, that have been computed with, respectively, 64 and 76 basis. In order to figure out if the branches that have not been obtained in the offline phase are physical, we performed an error analysis. To do it, we used the reprojection of each online solution $\mathbf{u}_{RB}$ in the full order space as initial guess for the offline solver, obtaining the solution $\mathbf{u}_{SEM}$; then the error is computed as $\|\mathbf{u}_{SEM} - \mathbf{u}_{RB}\|_{L^2}$. It should be noted that, with such an approach, the offline solver always converged in a single iteration for the physical branches in regions far from the bifurcations points, while it needed more iterations for the unphysical ones to converge to the closest admissible fields. In order to prove that those branches are not physical it is possible to look at the error plots in figure 4.10. The error associated to physical solutions, at least far from the bifurcation points, is always below a certain threshold (for example a couple of order above the tolerance used by the iterative solver that, for these plots, was 10^{-8}), while the error of the unphysical ones is always very high. It is also interesting to observe that, since the physical and unphysical branches are closer and closer (decreasing the viscosity), the error of the latter is smaller and smaller, but it is still much higher than the one relative to feasible solutions.
Finally, we anticipate that such a phenomenon will be important in section 4.3.2.2 and that it will be used to better interpret the diagram of the error of a diagram with two parameters.

4.3. Results with two parameters

4.3.1. Overview of the problem and motivation

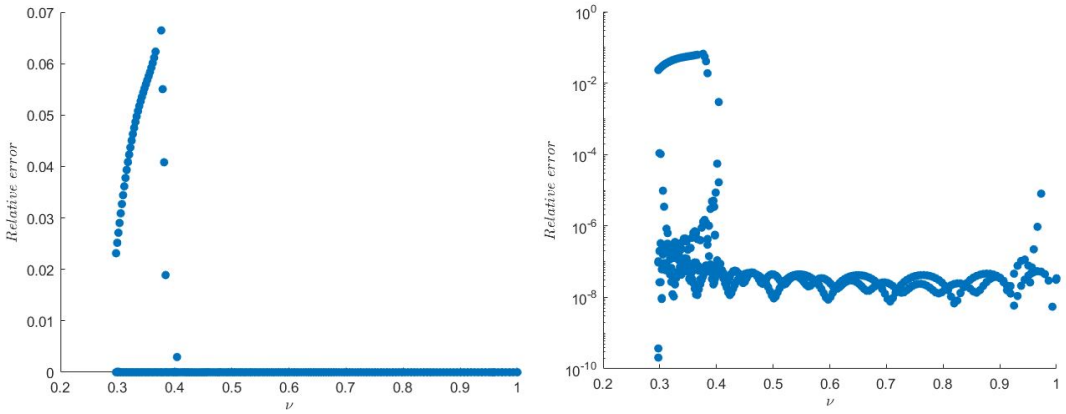
In section 4.2 we proved that it is possible to reconstruct an entire bifurcation diagram in a very efficient and accurate way. The result can be achieved using, in the offline phase, the



(a) RB size = 64

(b) RB size = 76

Figure (4.9) Bifurcation diagrams obtained in the online phase obtained using snapshots coming from only 5 branches.



(a) Uniform scale

(b) Logarithmic scale

Figure (4.10) Error with respect to full order solutions relative to the solutions in figure 4.9a

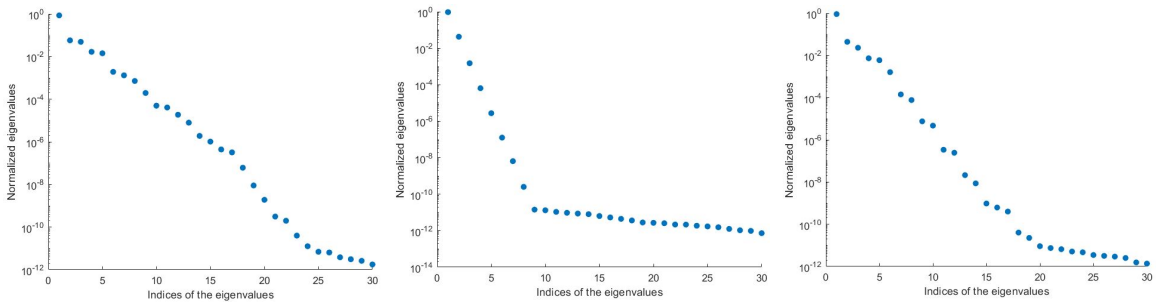
deflated continuation to have snapshots coming from all the different branches, generating, with the POD method, a global reduced basis and, finally, using again the deflated continuation to reconstruct the diagram in the online phase. Even if such a result can be interesting from the theoretical point of view, the diagram has been, in practice, already computed during the offline phase; therefore the online reconstruction loses its relevance. One could reply that in the online phase one can afford very small continuation steps and therefore the branches are discretized in a better way. However, it is not possible to obtain it without using short steps also in the computation of the snapshots, otherwise the asymmetric branches would not be found using the deflation. Furthermore, because of this constraint in the offline phase, several full order different solutions have to be computed. For instance, in order to obtain a diagram with $\nu \in (0.3, 1)$ with $\Delta\nu_i < \Delta\nu_{max} = 0.02 \cdot \nu$, one needs 315 snapshots. Once more, let us note that the dependency of $\Delta\nu_{max}$ by ν is useful to use smaller steps for higher Reynolds numbers. We remark that, if one wants to compute a bifurcation diagram letting vary two or more parameters with the previously described method, the number of snapshots could easily grow too much. Moreover, one can not use a tensorial product of more one-dimensional grids because the bifurcation points depend by an unknown combination of all the parameters, otherwise the property of the continuation method to use adaptive step sizes would be lost. A more general approach, letting vary n parameters (named μ_1,...,μ_n), can be summarized by the following pseudo-code:

```
// The parameters are organized in a vector, the current value of μ_j is μ[j].

for(double μ[n] = μ^n_min; μ[n] < μ^n_max; μ[n] = μ[n] + (Δμ)_n)
{
        for(double μ[n − 1] = μ^{n−1}_min; μ[n − 1] < μ^{n−1}_max; μ[n − 1] = μ[n − 1] + (Δμ)_{n−1})
        {
                .
                .
                .
                        for(double μ[2] = μ^2_min; μ[2] < μ^2_max; μ[2] = μ[2] + (Δμ)_2)
                        {
                                Computation of a one-dimensional bifurcation
                                        ↪ diagram letting vary only μ_1
                        }
                .
                .
                .
        }
}
```

Where, in order to improve the method, each increment $(\Delta\mu)_i$ can be a function of the subsequent parameters $\mu_{i+1}, ..., \mu_n$. Let us assume that the snapshots required to compute the one-dimensional diagram in the inner loop are always approximately N_1 and that, for every index j such that $1 < j \leq n$, $N_j = (\mu^j_{max} - \mu^j_{min})/(\Delta\mu)_j$. Then the total number of solutions

that have to be computed is, approximately, $\prod_{j=1}^{n} N_j$. Let us remember that the initial value of a parameter could be the maximum of its range and then it can be decreased till its minimum. Since the result is the same, the best approach should take into account the specific problem, solving the easiest problem at the beginning, and then moving on to more and more complex problems. For instance, in CFD, if μ_i represents the Reynolds number, it is better to begin with μ^i_{min} and then use the continuation to obtain good guesses for higher Reynolds numbers. Viceversa, if a parameter represents the viscosity, it is better to slowly decrease it because an higher viscosity can stabilize the numerical method.
It can be observed that this is a good approach only if every bifurcation point can be obtained letting vary only μ_1 (for example, if all the bifurcation points are pitchfork bifurcations, this hypothesis is respected). Moreover, if the first solution of the inner loop is obtained using as initial guess the closest solution to it, every initial guess is obtained with the pseudo-arclength continuation method except for a very small number of them (the ones related to the first snapshots of each one-dimensional diagram) that are computed with the simple continuation. For this reason, even if this approach is very expensive because a huge number of snapshots have to be computed, it is very accurate and could lead to a very accurate reconstruction of the diagram in the online phase.
However, such a method would have the same weakness of the one discussed in section 4.2: the online phase would be almost useless because the diagram would have been already computed, even if in a very expensive way, during the offline phase. In order to solve such a problem and to develop an efficient way to compute such a diagram, we chose a different approach: a one-dimensional diagram is always computed letting vary only the first parameter, but only the two extrema of each interval are used as values in the other dimensions. It can be noted that it is equivalent to use the previous schema with $\Delta\mu_j = \mu^j_{max} - \mu^j_{min}$ for any j greater than 1. This fact implies that the simple continuation in the directions different from the first one can not work because the step is too long but, on the other hand, only $2^{n-1}N_1$ snapshots are required. Therefore, since the simple continuation can not work, we decided to use the zero field as initial guess for the first solution of the inner loop, this way the computation of that solution is much slower but the gain in terms of computational time required for the computation of the snapshots is huge. However, the computation of the snapshots is still very demanding, but this is the most efficient way to get informations from every dimensions of the parameter space. We remark that in the offline phase, using the second approach, one does not obtain a multi-dimensional bifurcation diagram but only more one-dimensional diagrams. The snapshots coming from them will then grouped together to generate the reduced basis and the entire bifurcation diagram will be constructed only in the online phase using the first approach (in the online phase one can afford it because of the efficiency that characterizes such a phase).
In this work we decided to fix $n = 2$ to prove that the method works fine, therefore only two one-dimensional diagrams are computed in the offline phase. In order to exploit and generalize all the results in section 4.2, the first parameter is, once more, the viscosity ν, while the second one is a multiplicative scaling factor of the inlet boundary condition, that will be called s. One could observe that we previously wrote that the bifurcations are due to the Reynolds number and that these two parameters are strongly related to it. In fact the viscosity is part of it and the maximum or the mean velocity of the Dirichlet boundary condition at the inlet is often used



(a) 2 values of s, 15 values of ν (b) 30 values of s, 1 value of ν (c) 5 values of s, 6 values of ν

Figure (4.11) Eigenvalues decay obtained with different settings of the model obtained using the scaling as a parameter with different sampling for the snapshots.

as characteristic velocity of the fluid. In order to be sure to use two parameters that contain different informations, we looked again at the eigenvalues decay of the correlation matrix.
One would expect to find a behaviour similar to the one present in figure 4.4 if the two parameters lead to different features in the solutions. On the other hand, if the scaling was not interesting, a too strong decay would have occurred, with the first eigenvalue very high and the other ones very low. An intermediate phenomenon where the two parameters are too strongly related (and therefore they contain very similar informations) could be discovered if the eigenvalues decay, in a simulation with snapshots obtained letting vary both the parameters, would not be exponential.
In figure 4.11a the setting is similar to the one used to obtain the diagrams: only 2 values of scaling are considered, while the two required one-dimensional bifurcation diagrams (that included a bifurcation point) have been obtained letting varying only the viscosity. It can be observed that there are two different slopes due to the fact that the snapshots can be clustered in two groups according to their associated scaling, but that the decay is again exponential. In figure 4.11b all the snapshots have been computed with $\nu = 1$ but with 30 different values of scaling, the plot is very similar to the one in figure 4.4c because in both the scenarios all the snapshots belong to the symmetric branch. Finally, in figure 4.11c, the snapshots have been obtained with a tensor product of 5 values of scaling and 6 values of viscosity. It is interesting to remark that similar solutions can be obtained properly varying both the parameters (such a phenomenon will be better explained in section 4.3.2.2), therefore the solution manifold is smaller and it is possible to summarize it with fewer POD modes. However, it is important to highlight that the eigenvalues decay is always exponential and very similar to the one shown in the subplots of figure 4.4. Therefore, one can conclude that the scaling can be considered as an interesting parameter and additional informations are added to the space when it is taken into account.

4.3.2. Results

In this section the results that can be obtained using the described techniques will be presented. They are the generalization of the ones presented in section 4.2.2, however, since we have already analyzed the eigenvalues decay in section 4.3.1, we will focus on the obtained bifur-

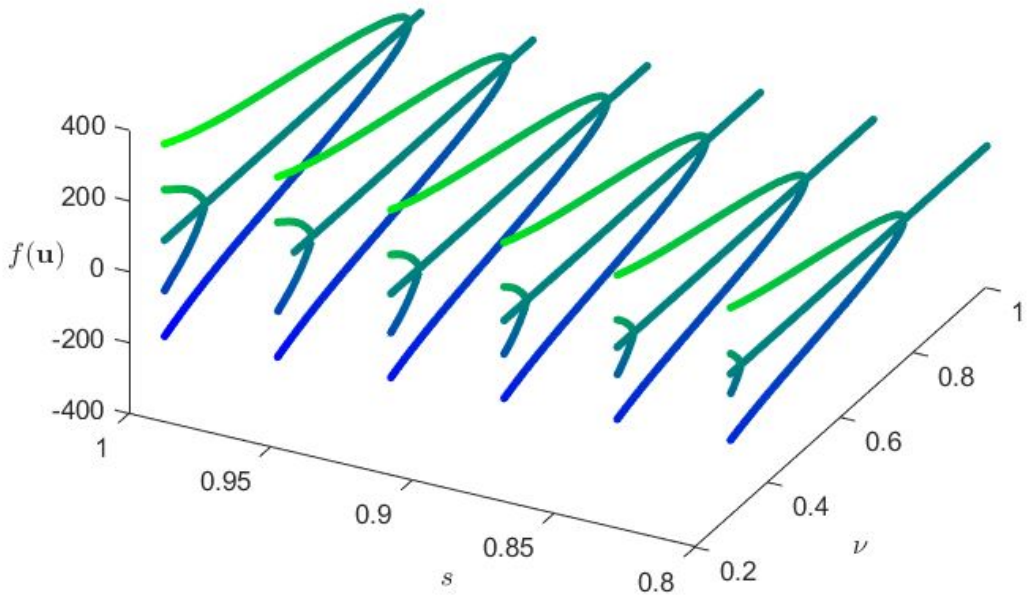


Figure (4.12) Bifurcation diagram efficiently obtained in the online phase with two parameters and two bifurcations. The colour gradient remarks the value $f(\mathbf{u})$ in each point in order to allow the reader to more easily interpret the diagram.

cation diagram and on its accuracy. In order to prove that the method works, we immediately show a complete bifurcation diagram with two parameters over the tensorial parameter domain $(\nu, s) \in [0.3, 1] \times [0.8, 1]$ (figure 4.12). In the next subsections it will be analyzed, with a particular focus on its accuracy and on the issues that have to be faced to obtain it.

4.3.2.1. Efficiency quantification

We want to remark the fact that the diagram in figure 4.12 has been discretized with 16970 different solutions, therefore it would have been very expensive to compute it with the full order solver. Such solutions are associated to parameters obtained with a finer grid (finer with respect to the one in the offline phase) in both the directions; this can partially explain the huge number of solutions, even if further details will be presented soon. Obviously, if one is interested in a diagram that looks continuous in both the parameters, one can simply refine the grid in the second direction, the computational cost will proportionally increase with the number of solutions, but the result can be obtained using the same method. However, even if many solutions have to be computed, the computation of such a diagram is very efficient because obtaining a single solution is a very inexpensive operation in the online phase.
It is not possible to compare the time needed to perform the entire offline phase with the one spent in the online one because in the online phase much more solutions are computed. Moreover, it would also be unfair to compare the time needed for solving the problem once

because it depends by the goodness of the initial guess: smaller steps of the continuation method or different types of such a method would imply very different initial guesses. As explained above we chose the pseudo-arclength continuation method in the offline phase to afford bigger steps but, due to the fact that the online solver is very fast, we preferred to use the simple continuation method with the step sizes equal to a fixed fraction of their offline counterpart. Finally, the iterative solver needs much more iterations when the parameters are very close to critical values or when, in the online phase, the reduced space is very noisy.
Therefore, the only meaningful times to be compared are the ones required, on average, by the iterative solvers to perform a single step. We observed that, even if they depend by the numbers of degrees of freedom, the orders of magnitude of these quantities in the offline and in the online one are very different. The offline solver is based on the methods of *Nektar++* and, even if very big linear systems have to be assembled and solved, it is very high-performance due to the static condensation method used to solve them (see section 1.3.2). On the other hand, in the online phase the assembly of the system is very efficient due to the affine decomposition (see section 2.3.1), while solving it is very fast because of its small dimension.
In this work we always used few thousands degrees of freedom in the offline phase (note that this number is very low, thanks to the SEM that allows a much faster convergence, when compared with classical FEM solutions), being able to perform a single step of the iterative solver in, on average, approximately 0.03 seconds for a system with about 1500 degrees of freedom and approximately 0.67 seconds for about 7000 degrees of freedom. Instead, in the online phase, we always used reduced spaces of dimensions lower than about 100, discovering that a single iteration of the solver needed between 10^{-6} and 10^{-4} seconds depending on the system dimension. We remember that the involved software is *ITHACA-SEM* and that it is possible to reduce the computational times that much thanks to efficient operations in the online phase and to the fact that the Navier-Stokes equations are first linearized and then projected on the reduced space. Therefore, on average, one can say that the online phase is about 10000 times faster.

4.3.2.2. Stability and accuracy issues

The main problem that has to be faced when one wants to reconstruct an entire bifurcation diagram with more parameters and bifurcation points is related to the stability of the online solver. As discussed in section 4.2.2.2, the model is very sensitive to the dimension of the reduced space but, with a single varying parameter, many different choices on the tolerance of the POD method lead to very good results. Instead, when one lets vary two parameters at the same time, the reduced space is very noisy and it spoils the stability of the solver.
Moreover, the two chosen parameters are strongly related to the Reynolds number, but we think that the situation can only worsen if the parameters are unrelated, for example considering two geometrical parameters that influence the domain only locally in two distinct region (see [35], where the SEM has been used with the reduced basis method and a geometrical parameter). It can be noted that the Reynolds number is the governing parameter also looking at the diagram 4.12 from another point of view, like in figure 4.13. Even if understanding the structure of the diagram has become harder, this perspective highlights the fact that the bifurcation points (grouped according to their natures) generate two straight lines. Such a phenomenon can be

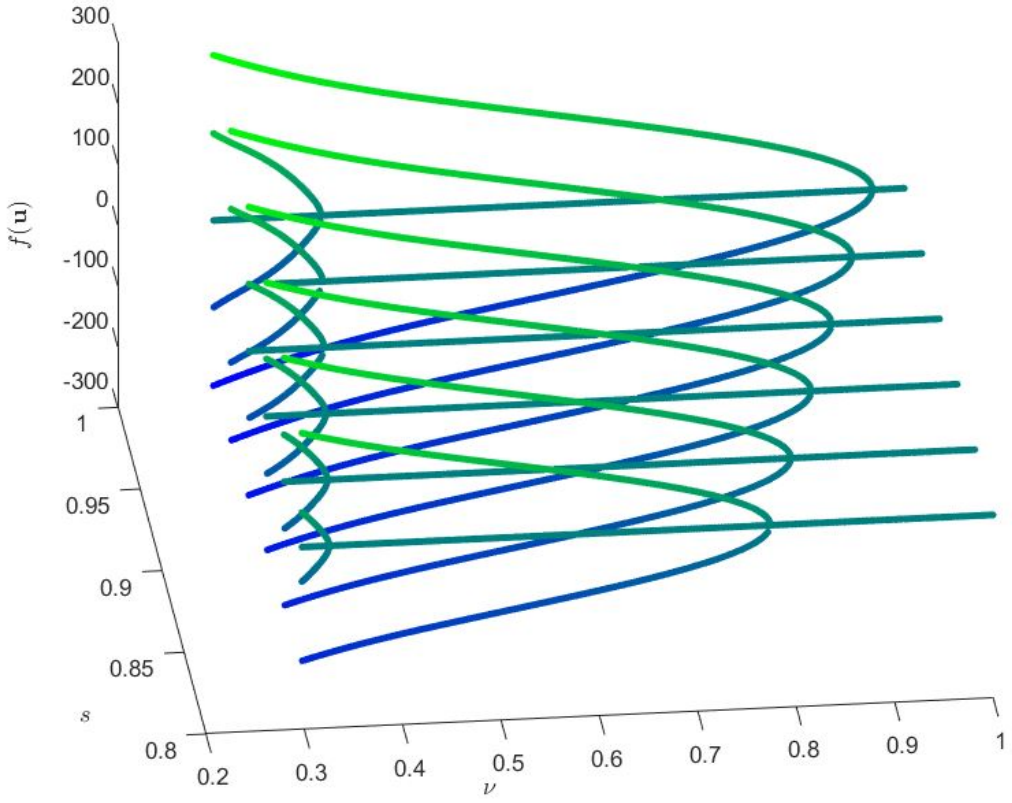


Figure (4.13) Bifurcation diagram obtained in the offline phase. Such a diagram is the same as in figure 4.12 but from a different perspective to observe the bifurcation points in a better way.

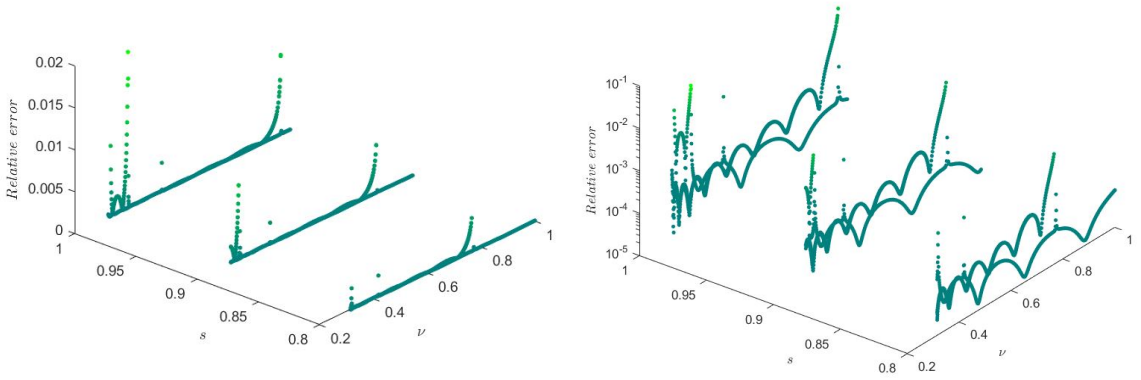
explained considering the fact that if one multiplies both the characteristic velocity and the viscosity for the same number, Re does not change. Indeed, different choices for the characteristic velocity are admissible and, probably, the most physical choice is strongly related, but not exactly proportional, to the inlet velocity; therefore the scaling does not influence the Reynolds number directly but only indirectly.
Such a property is reflected on the bifurcation diagram, because the lines that connect the bifurcation points are straight lines but, if two critical points of the same nature are associated to the values (ν_1, s_1) and (ν_2, s_2), the two ratios ν_i/s_i are slightly different. If they were equal it would mean that the inlet velocity (its peak or its average) is the physical characteristic velocity. Instead, the small difference means that the local features of the domain and of the velocity and pressure fields are important in the exact definition of the Reynolds number. Finally, we think that with other parameters the bifurcation points would be grouped in different curves and that the reduced space would became even noisier, leading to even worse stability issues.
When one wants to reconstruct a bifurcation diagram with only a bifurcation point, even if with two parameters, the iterative solver is able to converge with many different values of the POD tolerance but, if more bifurcation points are involved, great care must be taken to allow the method to be stable. In order to improve the stability we tried different approaches. Initially, we obviously tried to select the best POD tolerance. The problem is that, to obtain two bifurcations using two parameters, the range of the viscosity should be wide enough, five different kind of solutions are present and there are solutions that are very similar even for different values of the parameters. About the third factor, one would expect, for instance, that the solution associated to the pair (ν_1, s_1) is qualitatively very close to the one associated to $(\nu_2, \frac{\nu_2}{\nu_1} s_1)$ (if they are properly normalized over the inlet condition) because these pairs would generate the same global Re. The first two factors lead to a very big reduced space, while the third one can generate much noise in the basis. However, it is not possible to avoid these issues because they are strictly related to the problem itself, we thus decided to keep the POD tolerance very low (for example the diagram 4.12 is obtained with such a tolerance equal to 0.997). A low tolerance implies that the noisy and less useful data are discarded but, due to the first two factors, the basis is still big enough to represent the different phenomena. The choice of the POD tolerance equal to 0.997 in the previous diagrams led to a reduced basis of dimension 29.
In order to overcame the issue related to the fact that the basis is very noisy due to similar solutions for different values of the parameters, we tried three different approaches related to the offline grid of the scaling. At the beginning we simply used an uniform grid in the second direction over the interval (s_{min}, s_{max}) to compute the snapshots, that is the first idea exposed in section 4.3.1. We observed that the method was moderately stable for values of scaling far from the extrema of the interval but that it was unstable near them. This phenomenon led to the second approach, here we tried to use a non-uniform grid, based on the Chebyshev-Gauss points distribution, to capture more informations near the end of the interval, while maintaining the stability in its center. The expression of the n values according to such a distribution in the reference interval $[-1, 1]$ can be described as follows:

$$\overline{s}_i = \cos\theta_i, \qquad \theta_i = \frac{i\pi}{n-1}, \qquad i = 0, ..., n-1, \qquad \overline{s}_i \in [-1, 1].$$

The previous formula can be used to obtain a similar distribution over the interval $[s_{min}, s_{max}]$ in the following way:

$$s_i = \frac{s_{max} - s_{min}}{2}\bar{s}_i + \frac{s_{max} + s_{min}}{2}, \qquad i = 0, ..., n-1 \qquad s_i. \in [s_{min}, s_{max}]$$

Using such a distribution, we observed that the stability slowly increased with the number n of values on the scaling grid in the offline phase. The problem was that even with $n = 40$ the online solver often diverged or converged to unphysical solutions. Further increasing n would have implied a very expensive offline phase (the one-dimensional bifurcation diagram over the viscosity has to be computed n times) and a loss of the online phase relevance because the diagram would have been already computed in the offline phase. However, such an approach is, probably, the best one if more unrelated parameters are considered, like geometrical ones, because this way it is possible to capture enough information while maintaining the stability of the solver.
Finally, we tried a third approach (already briefly presented in section 4.3.1) to reduce as much as possible the noise in the reduced space. In the offline phase all the snapshots are computed for values of scaling equal to s_{min} or to s_{max}, therefore only two one-dimensional bifurcation diagrams are computed in such a phase. This way the basis contains the information about the second parameter but the number of viscosity-scaling pairs that generate the same Reynolds number are limited. In this work the best results have been obtained with the third approach, but it is possible that it could badly perform in more complex scenarios.
A wiser choice in the offline sampling is very useful to improve the accuracy, but could not be enough to always allow the converge of the online solver. In order to further improve the model we decided to use more snapshots but, since we observed that too many values of the second parameter spoiled the reduced basis, we preferred to refine the offline grid associated to the viscosity. This is very useful mostly when there are branches that are much shorter than the others, in fact they would contain much less snapshots and, therefore, the POD method could discard their information. In this work this issue is present, since the two branches that arise from the last bifurcation point are very close to the end of the domain. It could be noted that, in order to obtain an equivalent result without the scaling, the constraint $\Delta\nu_i < \Delta\nu_{max} = 0.02 \cdot \nu$ led to a clean reduced basis and a stable online solver, while , in order to obtain the diagram 4.12 the stronger constraint $\Delta\nu_i < \Delta\nu_{max} = 0.0075 \cdot \nu$ has been used. Another way to obtain an equivalent result would be to enlarge the viscosity domain, this way the last branches would be longer, would contain more snapshots, and therefore their information would be more important for the POD method. Moreover, to avoid adding noise to the reduced basis because of bad solutions, we raised the order of the polynomials in each element from 10 to 12. This way the computation of the snapshots and of the matrices/vectors needed in the online phase is slower, but we observed that in the deflation phase (both offline and online) the new branches are found much earlier.
Finally, we show the error associated to the two-dimensional bifurcation diagram in figure 4.14, both in uniform and in logarithmic scale. Looking at subplot 4.14b one can observe that the lowest relative errors are much higher than the ones in 4.6b. This is due to the fact that, even with all the techniques described above, the reduced basis was still noisy and, in order to ensure the convergence even near the bifurcation points, we have been obliged to raise the tolerance



(a) Uniform scale

(b) Logarithmic scale

Figure (4.14) Error with respect to full order solutions

used by the iterative solver to 10^{-5}. Other two features that can be noted are related to the peaks: firstly, as in figure 4.14, the error significantly increases near the bifurcation points and is characterized by an oscillatory behaviour. This is again, as for the one-dimensional problem, due to the fact that the reduced order models do not perform perfectly when the solution manifold is not smooth enough, and the bifurcation points destroy the smoothness of such a manifold. Secondly, it can be observed that the error raises also for values of the parameters such that $s/\nu \approx 2$. This phenomenon is probably due to the presence of some fictitious critical point that arises only in the online phase like the unphysical branches in section 4.2.2.2. To obtain the diagram the reduced space dimension was only 29 but, probably thanks to the higher number of snapshots, it already contained the information to recognize the fake branches presence, even if the solver did not converge to unphysical solutions.

4.4. Conclusion

In this chapter the results that can be obtained with the described methods have been shown. In section 4.2.1.1 we described how to couple the continuation method with the deflation one to efficiently generate a bifurcation diagram, then, in section 4.2.1.2 we discussed a possible heuristic to improve the effectiveness of the deflation method and, finally, we analyzed the first bifurcation diagram in section 4.2.1.3. The diagram in figure 4.3 can be considered as the first proof of the effectiveness of the described approach. However, it has been obtained with full order solutions and, therefore, its computation was very expensive. Subsequently, we proved that it is possible to efficiently reconstruct such a diagram exploiting the reduced basis methods, even if great care must be taken to handle the stability issues of the online solver. Moreover, we extended such a result to a scenario characterized by the presence of two different parameters, demonstrating that it is possible to obtain results that would be unaffordable without a reduced order method.
On the other hand, we were also interested in the accuracy of the method and, therefore, we analyzed the norm of the difference between the full order solutions and the reduced ones,

observing that such a quantity decays exponentially with the dimension of the reduced space. However, since the aim of the reduced basis method is to efficiently reconstruct smooth solution manifolds, the error near the bifurcation points was higher than elsewhere, even if we observed that its decay is exponential even in those critical regions.

5. Conclusion

The purpose of this work was to develop a technique to efficiently compute bifurcation diagrams using different existing methods together.
A bifurcation diagram is a diagram that graphically summarizes specific informations about the solutions of a non-linear equation (that can be differential or not). Obtaining it analytically is usually impossible in the context of differential equations because one is, often, not able to compute the different solutions. On the other hand, one could numerically compute a finite number of approximated solutions to obtain a discrete bifurcation diagram, but different techniques are required in order to find all the existing solutions of the problem. However, the latter is a costly computation and, if several parameters are involved or if their range is wide, the computational cost may become prohibitive. Therefore, we decided to rely on a reduced order method to split the task in two different steps: the first one, namely the offline phase, is used as a preparation step, while the second one, namely the online phase, is characterized by the actual computation of the bifurcation diagram. As explained in Chapter 2, the reduced basis method can be used to construct the bifurcation diagram because, after the generation of the reduced space involved in the Galerkin formulation of the problem in the offline phase, the computation of a single solution is an inexpensive process. However, the offline phase is very expensive because solutions belonging to all the branches of the diagram have to be computed. Moreover, such solutions are characterized by numerous degrees of freedom and, to obtain them all, different numerical methods have to be used.
In order to reduce the computational cost associated to the offline phase, we decided to rely on the spectral element method because such a method is characterized by an exponential convergence when the total number of degrees of freedom is incremented (see Chapter 1 for the description of the method). This way, it is possible to obtain very accurate approximations using a limited number of unknowns. For instance, the mesh used to obtain the results discussed in Chapter 4 is presented in figure 5.1; such a mesh is associated to the domain Ω described in section 4.1. It can be observed that it is very coarse but that the areas of the elements closer to the inlet are half of the ones of the remaining elements. This is useful because we decided to fix the same polynomial order inside all the elements and, this way, we could maintain an acceptable level of accuracy in the most important region of the domain. In fact, the different solutions shown in figure 4.2 can be distinguished simply analyzing the first part of the domain. On the other hand, it was useful to consider an extended channel because, otherwise, the outlet boundary conditions would have strongly influenced the solutions and the solver would have converged to unphysical pressure-velocity fields. It can be observed that different results can be obtained with the same mesh but with polynomials of different order. For instance, using the mesh in figure 5.1, one can obtain good approximations of symmetric solutions using polynomials of order 4, while one needs, at least, polynomials of order 6 to compute the solutions belonging to the first three branches of the diagram in figure 4.3 and of order 8 to obtain the

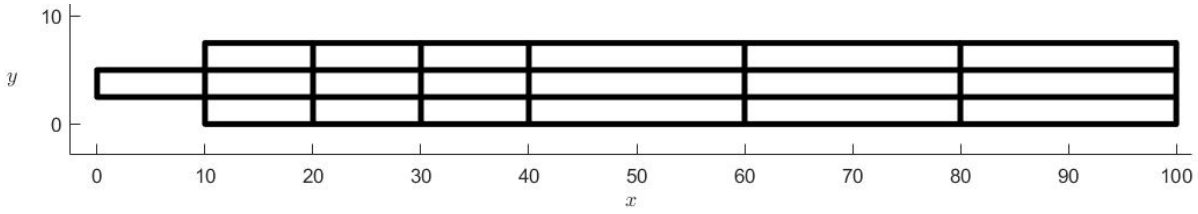


Figure (5.1) Mesh used to compute the snapshots. The spectral element method is able to capture the different features of the solution even with very coarse meshes thanks to the high order of the involved polynomials.

last two branches. However, to increase the stability of the online solver, we observed that it was better to further increase the order of such polynomials.
Finally, we used two different methods to generate the bifurcation diagram: the continuation method and the deflation method. Assuming one has already obtained a solution on a branch, it is possible to entirely track such a branch using the continuation method. The key idea behind this method is to exploit the last n computed solutions $\mathbf{u}_{i-1}, ..., \mathbf{u}_{i-n}$ to derive a good initial guess for the next one $\mathbf{u}_i$ to be sure that the solver, starting from the obtained initial guess, will converge to $\mathbf{u}_i$. In this work we discussed and used two different approaches, associated to $n = 1$ and $n = 2$, and respectively named simple continuation (see section 3.2.1) and pseudo-arclength continuation (see section 3.2.2). Such methods have advantages and disadvantages that have been more deeply discussed in their specific sections and in Chapter 4. Finally, the deflation method has been used to discover solutions on unknown branches (see section 3.3) preventing the solver to converge to known pressure-velocity fields.
In Chapter 4 we presented several results that can be obtained with such an approach. We proved that it is possible to compute bifurcation diagrams including more bifurcation points and characterized by more parameters exploiting the continuation and the deflation methods, that the eigenvalues decay is weakly influenced by the presence of different branches, and that one can exploit the offline-online splitting to efficiently reconstruct the diagram.
However, we also highlighted that many stability problems may occur in complex scenarios, preventing the solver to converge to any solution. As discussed in sections 4.2.2 and 4.3.2.2, in order to handle them we decided to use a global reduced basis and to properly select the tolerance associated to the POD method and the range of variation of the involved parameters. This way, it is possible to efficiently and accurately reconstruct the diagram, however, some of the tasks performed in the offline phase have to be repeated in order to choose such values.
An other possible approach, that could improve the stability of the online solver, is related to the local reduced basis approach [34]. In such an approach the snapshots have to be clustered and a reduced basis associated to each cluster has to be generated (here any of the several existing clustering techniques [41] can be used). The stability of the online solver is supposed to improve because each basis is much less noisy than a global one, in fact the snapshots in each cluster are very similar to each other and, therefore, the different clusters represent different branches or portions of them. Moreover, the associated bases can be considered related to the speficic part of the branches. However, it is important to remark that the deflation in the online phase should be properly modified to use multiple basis. In fact the solver could only converge to solutions

that can be represented by the considered basis, it is therefore compulsory to select at each step the correct one. However, since we observed that the deflation works fine with a basis associated to few branches, one could use, at each step, two bases together to be able to represent both the known branch that contains the initial guess and the sought unknown solution.
On the other hand, if one is interested in improving the accuracy of the method without modifying it, it is possible to enrich the reduced space with special velocity fields named *supremizers* [46]. As discussed in Chapters 1 and 2, the *inf-sup* condition is required both in the full order problem and in the reduced one to ensure the stability. However, the stability of the full order problem does not imply the one of the reduced problem [7]. In order to obtain a stable problem, one can consider, for each pressure basis function p_j, an other function denoted as *inner supremizer velocity function* $T^\mu p_j$ and defined as:

$$T^\mu p_j := \arg\sup_{\mathbf{v}\in V^\delta} \frac{b(\mathbf{v}, p_j; \mu)}{\|\mathbf{v}\|_V},$$

where V^δ is the full order space and $b(\cdot, \cdot; \mu)$ has been defined in section 1.3. It should be observed that $T^\mu p_j$ is the solution of the following discrete elliptic problem:

$$(T^\mu p_j, \mathbf{v})_V = b(\mathbf{v}, p_j; \mu), \qquad \forall \mathbf{v} \in V^\delta. \tag{5.1}$$

After having solved problem (5.1), it is possible to enrich the reduced space V^{rb} as follows:

$$\widetilde{V}^{rb} := V^{rb} \oplus \text{span}\{T^\mu p_j : j = 1, ..., N^{rb}\}.$$

This way, if one seeks the solution of the reduced problem in $\widetilde{V}^{rb}$, the stability is ensured, even if it is more expensive because the dimension of $\widetilde{V}^{rb}$ is $2N^{rb}$ instead of N^{rb} (that is the dimension of V^{rb}).
Furthermore, we want to describe a possible application of such a work, even if such a topic is deeper discussed in [60]. The application is the mitral regurgitation: it is a disease in which the blood flows from the left ventricle of the heart to the left atrium through the mitral valve. The echocardiography is the most used tool to detect and quantify the dangerousness of such a disease, however, when the blood flux is very close to the wall, it is complex to correctly interpret the obtained images and the disease may be not recognized. It is therefore useful to exploit the direct simulation of the flow to quantify the blood flow rate to understand if a patient requires a proper treatment. Anyway, as previously written, simulating the exact flux of the blood is a very expensive process, it is therefore convenient to rely on reduced order methods to be able to effectively use such a technology. From the mathematical point of view and considering a parametrized domain to describe the heart and parametrized coefficients to describe the blood, it is possible to describe the different possible blood fluxes as different solutions of the Navier-Stokes equations. It is therefore useful to rely on the bifurcation theory to analyze them, for instance with the techniques described in this thesis.
The domain used in Chapter 5 can be considered as a very simplified version of the mitral valve (the small inlet) and of the left atrium (the channel), while the involved parameters are associated to the blood viscosity, that can be related to its temperature, and to its velocity in the mitral valve [36]. We observed that the asymmetrical solutions exist only above certain

thresholds of the parameters but, as observed in [36], the bifurcation points are strongly related to the shape and to the dimension of the domain. It is therefore advisable to consider more parameters and a more faithful approximation of the domain before applying the described method to a real scenario. However, if one is only interested in the values of the parameters associated to the bifurcation points, it is possible to detect them analyzing the sign of the real part of the eigenvalues of the linearized problem, as described in [59] or in [61]. Once more, even if one is only interested in such a result, it is important to consider reduced order methods to obtain it in an efficient way, otherwise the method would be too expensive to be applied to biomedical problems.
It is also important to note that the results shown in Chapter 4 could be obtained using the FEM instead of the SEM. However, the discussed approach, based on the SEM, is characterized by two main advantages. Firstly, in order to obtain the same level of accuracy, a FEM solution would require much more degrees of freedom, therefore, the entire offline phase would be significantly more expensive. In fact assembling and solving the linear systems scale with the number of degrees of freedom and the construction of the reduced space is greatly influenced by such a quantity. Secondly, the support of each SEM basis is much wider than the one of a FEM basis. Such a property implies that the reduced bases, characterized by a global support, are more similar to the SEM ones [30], therefore the obtained discretizations are more strongly related and the projections on the reduced space are more similar to the original snapshots (to parity of reduced space dimension). Moreover, the discussed method can be generalized as in [53] and [49]: here the *reduced basis element method* (RBEM) is presented. Such a method is similar to the RB one, but exploits the fact that the SEM basis functions are characterized by a wider support in order to construct a reduced space associated to each element (or to each region of a partitioned domain). The RBEM can be used to ulteriorly improve the efficiency and the accuracy of the RB method.
Finally, we want to highlight some of the future perspectives of this work. Firstly, one can modify the described methods to improve the stability of the online solver in order to consider more parameters or a wider parameter domain. Such a result can be achieved, as written above, exploiting the *supremizers* [46] or a local reduced basis approach [34]. Secondly, in order to apply the described method to biomedical problems as previously discussed, it is important to consider three-dimensional [60] and curved [36] domains. Lastly, since the described method is very expensive and its computational cost will further increase when paired with more realistic geometries, a wider parameter domain or more than two parameters, it is important to improve its efficiency. To do it, one can use the RBEM [53] or adapt one of the described methods. For instance, we observed that the deflation method can be considered the bottleneck of the deflated continuation method. To lighten such a technique, one could pair it with another method to automatically decide if the deflation is required or not, e.g. one could solve an eigenvalue problem as in [61] to detect the bifurcation points and then activate the deflation in a small neighbourhood of such a critical point.

A. Appendix: Fundamentals of Computational Fluid Dynamics

A.1. The Navier-Stokes equations

The starting point of a thesis in *computational fluid dynamics* (CFD) can only be an overview of the *Navier Stokes* (NS) equation system [22], that can be written in the following way:

$$\begin{cases} \dfrac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \\ \dfrac{\partial (\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = \nabla \cdot \sigma + \mathbf{f} \\ \dfrac{\partial E}{\partial t} + \nabla \cdot (E\mathbf{u}) = \nabla \cdot (\sigma \mathbf{u} + \mathbf{q}) + \mathbf{f} \cdot \mathbf{u} \end{cases} . \tag{A.1}$$

Here $\mathbf{u} = \mathbf{u}(\mathbf{x}, t)$ is the velocity field, ρ the density, E the total specific energy, σ the stress tensor, $\mathbf{q}$ the heat flux vector and, finally, $\mathbf{f}$ is the external force [42]. Even if many different materials can be associated to as many expressions of σ, in this work we want to simplify as much as possible the model, therefore we will consider only Newtonian fluids where the following relation holds:

$$\sigma = -\widetilde{p}\,\mathbf{I} + \mu \left[\nabla \mathbf{u} + (\nabla \mathbf{u})^T \right] - \lambda (\nabla \cdot \mathbf{u})\, \mathbf{I}. \tag{A.2}$$

Here $\widetilde{p}$ is the pressure, $\mathbf{I}$ is the unit tensor and μ and λ are the the viscosity coefficients. It should be noted that, if one is not interested in discontinuous solutions (useful for example to model the shocks), can assume that the *continuum hypothesis* [73] holds and can express the thermodynamic equilibrium by means of the Stokes' hypothesis, that implies that

$$2\lambda + 3\mu = 0.$$

We refer to [17] for a more detailed description of the hypothesis and for an alternative formulation.
Another term that can be modeled in different ways is the heat flux $\mathbf{q}$ in the energy equation, even if the most common one is obtained using the Fourier law:

$$\mathbf{q} = -k(T)\nabla T,$$

where T is the temperature.
The Navier-Stokes equations are the most accurate continuous model to describe the motion of a fluid but, due to their complexity, it is impossible to solve them analytically except in few simplified cases. Therefore, one is obliged to solve them, or at least one of their approximations,

numerically. They are called, respectively, the mass balance equation, the momentum balance equation and the energy balance equation because they describe the variation of those quantities with the constraint that they have to be conserved. In order to simplify the system (A.1), the first constraint that one can add is the incompressibility constraint [16]:

$$\frac{D\rho}{Dt} = \frac{d}{dt}\rho\left(\mathbf{x}, t\right) = \frac{\partial \rho}{\partial t} + \mathbf{u} \cdot \nabla \rho = 0.$$

Such a constraint represents the fact that the density of the fluid can not vary on the streamlines. With this assumption the first equation can be simplified into $\nabla \cdot \mathbf{u} = 0$; it can be proved that such an approximation is acceptable if the characteristic velocity of the fluid is much lower than the speed of sound. The second approximation that we will use is $\rho = const$, this is very useful because it allows to decouple the energy equation from the other two. In fact, in this case, $\mathbf{u}$ and $\widetilde{p}$ are the only remaining unknowns and, with the mass and momentum balance equations, there is the same number of equations and unknowns. It is interesting to observe that, even if we will decouple and ignore the energy equation, one could solve first the mass and momentum equations obtaining the pressure and velocity fields, and then use them to compute the energy using the remaining equation. Finally, since we are interested in steady flows without external forces, the following additional constraints are required:

$$\frac{\partial \mathbf{u}}{\partial t} = 0, \qquad \mathbf{f} = \mathbf{0}.$$

Using all these assumptions together, redefining the pressure as $p = \widetilde{p}/\rho$ and employing the kinematic viscosity $\nu = \mu/\rho$ instead of the dynamic one μ, it is possible to obtain the following system:

$$\begin{cases} \nabla \cdot \mathbf{u} = 0 \\ \mathbf{u} \cdot \nabla \mathbf{u} - \nu \Delta \mathbf{u} + \nabla p = 0 \end{cases}. \tag{A.3}$$

Such a system describes the steady motion of an incompressible fluid. It should be noted that it is an approximation, however, it is very accurate when the described assumptions hold. Finally, we highlight that this is the system that will be solved to obtain the results shown in Chapter 4.

A.2. The Reynolds number

Before proceeding, it is important to understand the role played by the different terms in the momentum equation. To do it, one can fix three quantities that represent the system and normalize the equations according to these values. In particular, if U is a characteristic velocity, L is a characteristic length and $\overline{\nu}$ is the characteristic kinematic viscosity, it is possible to define the *Reynolds number* (Re) as $Re = \frac{UL}{\overline{\nu}}$. Such a non-dimensional number is useful because the momentum equation (in equation (A.3)) can be written in the following way:

$$\mathbf{u} \cdot \nabla \mathbf{u} - \frac{1}{Re}\Delta \mathbf{u} + \nabla p = 0. \tag{A.4}$$

However, it should be noted that the velocity and the pressure in equation (A.3) are associated with proper units of measure, while in equation (A.4) they are not. On the other hand, the structure of the mass balance equation does not change after the normalization while the energy balance equation can be written in terms of another non-dimensional parameter: the Prandtl number [73]. Thanks to the normalized momentum balance equation, it is possible to understand that Re accounts for the ratio between the magnitude of the inertial terms and the diffusive ones in the equation itself. In fact $\mathbf{u} \cdot \nabla \mathbf{u}$ represents the advection because it models the fact that the velocity transports its own gradient acting as an inertial phenomenon, while, on the other hand, $\Delta \mathbf{u}$ is associated to the diffusion. Therefore, if Re is much lower than 1, the motion is governed by the diffusion and by the pressure gradient, the equation that represents the limit phenomenon for which $Re \to 0$ is called *Stokes equation*. The Stokes equation is a good approximation for flows that are very slow, or very viscous or in a domain with very small length-scales. The equation can be solved with many different methods because it does not contain the non-linear term, that is the actual source of complexity. The other intuitive approximation is the limit for very high Re, where the governing terms are the advection and the pressure gradient. The resulting equation is called *Euler equation* and it is much more used because, in common scenarios, Re is at least of order 10^5. However, it is much more complex than the Stokes equation because it maintains the non-linear advection term. Anyway, it should be noted that several phenomena in fluid dynamics are due to the presence of the boundary layer, a thin region around an impermeable wall that can be modeled including all the different terms in the Navier-Stokes equations [66].
Since we are interested in bifurcations but they are not present when the Reynolds number is very low, we decided to consider values of such a number belonging to the interval $(100, 1000)$; in such a range of Re both the Stokes equation and the Euler equation are bad approximation of (A.3), therefore one is obliged to directly solve (A.3). It is important to understand that Re is the only physical parameter in the equation (if one does not consider parameters associated to the domain or to the boundary conditions), therefore, even if one chooses different parameters, every change in the solution can always be related to it. This observation will be relevant to properly analyze the results shown in the last chapters, where two different parameters will vary but they may be considered as part of the Reynolds number.

A.3. Conclusion

It is important to highlight that, when the Reynolds number is high enough, the flow can become turbulent, significantly increasing the complexity of the simulations because a turbulent flow is characterized by the presence of numerous multi-scale vortexes called *eddies* [54]. To model it, one can mainly choose between three different approaches [77].
The first one, called *direct numerical simulation* (DNS) does not require a turbulence model but the discretization technique has to be able to catch the smallest scales of the flow. The second one, named *Reynolds-averaged Navier-Stokes equations* (RANS), relies on one or more coefficients to mimic the effect of the turbulence on the main flow. Lastly, the *large eddy simulation* (LES) is an intermediate technique that mimics the smallest scales but directly simulates the bigger ones. However, since in this work we are not interested in turbulent flows,

we decided to avoid any turbulence model. Instead, we preferred to use the SEM to directly simulate the flow, to do it one has to transform the strong Navier-Stokes equations (A.3) into its weak counterpart as explained in sections 1.1.1 and 1.3. Furthermore, we highlight that a similar formulation is also required by the reduced basis method.
Finally, even if we wrote that the Reynolds number is the key parameter to describe the motion of an incompressible fluid, we will consider the viscosity and a multiplicative factor in the inlet boundary condition as varying parameters in Chapter 4. However, it will be possible to relate the features of the obtained solutions to the Reynolds number thanks to its definition.

Bibliography

[1] *Introduction to numerical continuation methods.*
E.L. Allgower, K. Georg. Volume 45. SIAM (2003)

[2] *Bifurcations in nonlinear systems. Computational issues*
F.L. Alvarado. IEEE International Symposium on Circuits and Systems, pages 922-925. IEEE (1990)

[3] *A primer of nonlinear analysis.*
A. Ambrosetti, G. Prodi. Cambridge Studies in Advanced Mathematics, number 34. Cambridge University Press (1995)

[4] *Efficient matrix computation for tensor-product isogeometric analysis: The use of sum factorization.*
P. Antolin, A. Buffa, F. Calabro, M. Martinelli, G. Sangalli. Computer Methods in Applied Mechanics and Engineering, volume 285, pages 817-828. Elsevier (2015)

[5] *The serendipity family of finite elements.*
D.N. Arnold, G. Awanou. Foundations of Computational Mathematics, volume 11, number 3, pages 337-344. Springer (2011)

[6] *Error-bounds for finite element method.*
I. Babuška. Numerische Mathematik, volume 16, number 4, pages 322-333. Springer (1971)

[7] *Supremizer stabilization of POD-Galerkin approximation of parametrized Navier-Stokes equations.*
F. Ballarin, A. Manzoni, A. Quarteroni, G. Rozza. MATHICSE Technical Report, École Polytechnique Fédérale de Lausanne (2014)

[8] *An empirical interpolation method: application to efficient reduced-basis discretization of partial differential equations.*
M. Barrault, Y. Maday, N.C. Nguyen, A.T. Patera. Comptes Rendus Mathematique, volume 339, number 9, pages 667-672. Elsevier (2004)

[9] *Bifurcation analysis with symmetry groups.*
N. Boullé, P. Farrel. ENS Rennes and University of Oxford. Summer internship 2017

[10] *A proof of the inf-sup condition for the Stokes equations on Lipschitz domains.*
J.H. Bramble. Mathematical Models and Methods in Applied Sciences, volume 13, number 3, pages 361-371. Worlds Scientific (2003)

[11] *Finite dimensional approximation of nonlinear problems. I. Branches of nonsingular solutions.*
F. Brezzi, J. Rappaz, P.A. Raviart. Numerische Mathematik, volume 36, number 1, pages 1-25. Springer (1980)

[12] *Finite dimensional approximation of nonlinear problems. II. Limit points.*
F. Brezzi, J. Rappaz, P.A. Raviart. Numerische Mathematik, volume 37, number 1, pages 1-28. Springer (1981)

[13] *Finite dimensional approximation of nonlinear problems. III. Simple bifurcation points.*
F. Brezzi, J. Rappaz, P.A. Raviart. Numerische Mathematik, volume 38, number 1, pages 1-30. Springer (1982)

[14] *Deflation techniques for the calculation of further solutions of a nonlinear system.*
K.M. Brown, W.B. Gearhart. SIAM, Journal on Scientific Computing, volume 37, number 4, pages A2026-A2045. SIAM (2015)

[15] *Convergence of an adaptive hp finite element strategy in higher space-dimensions.*
M. Bürg, W. Dörfler. Applied Numerical Mathematics, volume 61, number 11, pages 1132-1146. Elsevier (2011)

[16] *Numerical methods for incompressible flow.*
M. Burger. Lecture notes based on lecture notes by René Pinnau, TU Darmstadt (2004)

[17] *A note on Stokes' hypothesis.*
G. Buresti. Acta Mechanica, volume 226, number 10, pages 3555-3559. Springer (2015)

[18] *Spectral methods: evolution to complex geometries and applications to fluid dynamics.*
C. Canuto, M.Y. Hussaini, A. Quarteroni, T.A. Zang. Springer (2007)

[19] *Spectral methods: fundamentals in single domains.*
C. Canuto, M.Y. Hussaini, A. Quarteroni, T.A. Zang. Scientific Computation, Springer (2006)

[20] *On the numerical analysis of adaptive spectral/hp methods for elliptic problems.*
C. Canuto, M. Verani. Analysis and numerics of partial differential equations, pages 165-192. Springer (2013)

[21] *Computational fluid dynamics.*
T. Chung. Cambridge University Press (2010)

[22] *Navier-Stokes equations.*
P. Constantin, C. Foias. University of Chicago Press (1988)

[23] *Methods of numerical integration.*
P.J. Davis, P. Rabinowitz. Courier Corporation (2007)

[24] *High-order methods for incompressible fluid flow.*
M.O. Deville, P.F. Fischer, E. Mund et al. Volume 9. Cambridge University Press (2002)

[25] *Numerical bifurcation methods and their application to fluid dynamics: analysis beyond simulation.*
H.A. Dijkstra et al. Communications in Computational Physics, volume 15, number 1, pages 1-45. Cambridge University Press (2014)

[26] *Generalizations of the Lax-Milgram theorem.*
D. Drivaliaris, N. Yannakakis. Boundary Value Problems, volume 2007, number 1. Springer (2007)

[27] *A new matrix bandwidth reduction algorithm.*
A. Esposito, M.F. Catalano, F. Malucelli, L. Tarricone. Operations Research Letters, volume 23, number 3-5 pages 99-107. Elsevier (1998)

[28] *The computation of disconnected bifurcation diagrams.*
P.E. Farrel, C.H.L. Beentjes, A. Birkisson. ArXiv preprint arXiv:1603.00809 (2016)

[29] *Deflation techniques for finding distinct solutions of nonlinear partial differential equations.*
P.E. Farrel, A. Birkisson, S.W. Funke. SIAM Journal on Scientific Computing, volume 37, number 4, pages A2026-A2045. SIAM (2015)

[30] *A stabilized POD model for turbulent flows over a range of Reynolds numbers: Optimal parameter sampling and constrained projection.*
L. Fick, Y. Maday, A.T. Patera, T. Taddei. Journal of Computational Physics, volume 371, pages 214-243. Elsevier (2018)

[31] *Elliptic problems in nonsmooth domains.*
P. Grisvard. SIAM (2011)

[32] *Convergence rates of the POD-greedy method.*
B. Haasdonk. ESAIM: Mathematical Modelling and Numerical Analysis, volume 47, number 3, pages 859-873. EDP Sciences (2013)

[33] *A fast spectral element solver combining static condensation and multigrid techniques.*
L. Haupt, J. Stiller, W.E. Nagel. Journal of Computational Physics 255, pages 384-395. Elsevier (2013)

[34] *A localized reduced-order modeling approach for PDEs with bifurcating solutions.*
M.W. Hess, A. Alla, A. Quaini, G. Rozza, M. Gunzburger. Computer Methods in Applied Mechanics and Engineering, volume 351, pages 379-403. Elsevier (2019)

[35] *A spectral element reduced basis method for Navier-Stokes equations with geometric variations.*
M.W. Hess, A. Quaini, G. Rozza. ArXiv preprint arXiv:1812.11051 (2018)

[36] *Reduced basis model order reduction for Navier-Stokes equations in domains with walls of varying curvature.*
M.W. Hess, A. Quaini, G. Rozza. ArXiv preprint arXiv:1901.03708 (2019)

[37] *A spectral element reduced basis method in parametric CFD.*
M.W. Hess, G. Rozza. European Conference on Numerical Mathematics and Advanced Applications, pages 693-701. Springer (2017)

[38] *Certified reduced basis methods for parametrized partial differential equations.*
J.S. Hesthaven, G. Rozza, B. Stamn. Springer (2016)

[39] *Fast SVD for large-scale matrices.*
M. Holmes, A. Gray, C. Isbell. Workshop on Efficient Machine Learning at NIPS, volume 58, pages 249-252 (2007)

[40] *A static condensation reduced basis element method: Complex problems.*
D.B.P. Huynh, D.J. Knezevic, A.T. Patera. Computer Methods in Applied Mechanics and Engineering, volume 259, pages 197-216. Elsevier (2013)

[41] *An introduction to statistical learning.*
G. James, D. Witten, T. Hastie, R. Tibshirani. Volume 112. Springer (2013)

[42] *Spectral/hp element methods for CFD.*
G.E Karniadakis, S.J. Sherwin. Numerical Mathematics and Scientific Computation. Oxford University Press, USA (1999)

[43] *Numerical methods in bifurcation problems.*
H.B. Keller. Tata Institute of Fundamental Research, Springer (1986)

[44] *Bifurcation theory: an introduction with applications to PDEs.*
H. Kielhöfer. Volume 156, Springer Science and Business Media (2006)

[45] *Elements of applied bifurcation theory.*
Y.A. Kuznetsov. Volume 112. Springer Science and Business Media (2013)

[46] *Model order reduction in fluid dynamics: challenges and perspectives.*
T. Lassila, A. Manzoni, A. Quarteroni, G. Rozza. Reduced order methods for modeling and computational reduction, pages 235-273. Springer (2014)

[47] *Newton's method with deflation for isolated singularities of polynomial systems.*
A. Leykin, J. Verschelde, A. Zhao. Theoretical Computer Science, volume 359, number 1-3, pages 111-122. Elsevier (2006)

[48] *Heuristics for matrix bandwidth reduction.*
A. Lim, B. Rodrigues, F. Xiao. European Journal of Operational Research, volume 174, number 1, pages 69-91. Elsevier (2006)

[49] *A reduced basis element method for the steady Stokes problem.*
A.E. Løvgren, Y. Maday, E.M. Rønquist. ESAIM: Mathematical Modelling and Numerical Analysis, volume 40, number 3, pages 529-552. EDP Sciences (2006)

[50] *The reduced basis element method: Offline-online decomposition in the nonconforming, nonaffine case.*
A.E. Løvgren, Y. Maday, E.M. Rønquist. Spectral and High Order Methods for Partial Differential Equations, pages 247-254. Springer (2011)

[51] *A well-posed optimal spectral element approximation for the Stokes problem.*
Y. Maday, A.T. Patera, E.M. Rønquist. Technical Report 87-148, ICASE, Hampton, VA (1987)

[52] *Spectral element methods for the Navier-Stokes equations.*
Y. Maday, A.T. Patera. IN: State-of-the-art surveys on computational mechanics (A90-47176 21-64). New York, American Society of Mechanical Engineers, 1989, p. 71-143. Research supported by DARPA (1989)

[53] *A reduced-basis element method.*
Y. Maday, E.M. Rønquist. Journal of Scientific Computing, volume 17, number 1-4, pages 447-459. Springer (2002)

[54] *The physics of fluid turbulence.*
W.D. McComb. Chemical physics (1990)

[55] *Mesh smoothing for the spectral element method.*
K. Mittal, P. Fischer. Journal of Scientific Computing, volume 78, number 2, pages 1152-1173. Springer (2019)

[56] *Metodi e algoritmi per il calcolo numerico.*
G. Monegato. Clut (2008)

[57] *Generalized Gaussian quadrature rules on arbitrary polygons.*
S. Mousavi, H. Xiao, N. Sukumar. International Journal for Numerical Methods in Engineering, volume 82, pages 99-113, number 1. Wiley Online Library (2010)

[58] *Reduced basis approximation and a posteriori error estimation for parametrized partial differential equations.*
A.T. Patera, G. Rozza et al. MIT Cambridge, MA, USA (2007)

[59] *Reduced basis approaches for parametrized bifurcation problems held by non-linear Von Kármán equations.*
F. Pichi, G. Rozza. Journal of Scientific Computing (2019)

[60] *Computational reduction strategies for the detection of steady bifurcations in incompressible fluid-dynamics: applications to Coanda effect in cardiology.*
G. Pitton, A. Quaini, G. Rozza. Journal of Computational Physics, volume 344, pages 534-557. Elsevier (2017)

[61] *On the application of reduced basis methods to bifurcation problems in incompressible fluid dynamics.*

G. Pitton, G. Rozza. Journal of Scientific Computing, volume 73, number 1, pages 157-177. Springer (2017)

[62] *Modellistica numerica per problemi differenziali.*
A. Quarteroni. Volume 97. Springer (2016)

[63] *Reduced basis approximation and a posteriori error estimation for Stokes flows in parametrized geometries: Roles of the inf-sup stability constants.*
G. Rozza, D.B.P. Huynh, A. Manzoni. Volume 125, pages 115-152, number 1. Numerische Mathematik (2013)

[64] *Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations.*
G. Rozza, D.B.P. Huynh, A.T. Patera. Archives of Computational Methods in Engineering, volume 15, number 3. Springer (2007)

[65] *Notes on the Banach-Necas-Babuška theorem and Kato's minimum modulus of operators.*
N. Saito. ArXiv preprint arXiv:1711.01533 (2017)

[66] *Boundary-layer theory.*
H. Schlichting, K. Gersten. Springer (2016)

[67] *Practical bifurcation and stability analysis.*
R. Seydel. Volume 5. Springer Science and Business Media (2009)

[68] *Nektar++: An open-source spectral/hp element framework.*
S.J. Sherwin et al. Computer Physics communications, volume 192, pages 205-219. Elsevier (2015)

[69] *Delaunay refinement algorithms for triangular mesh generation.*
J.R. Shewchuk. Computational geometry, volume 22, number 1-3, pages 1-74. Elsevier (2002)

[70] *Higher-order finite element methods.*
P. Solin, K. Segeth, I. Dolezel. Chapman and Hall/CRC (2003)

[71] *Computation and visualization of bifurcation surfaces.*
D. Stiefs, T. Gross, R. Steuer, U. Feudel. International Journal of Bifurcation and Chaos, volume 18, number 8, pages 2191-2206. World Scientific (2008)

[72] *Introduction to numerical analysis.*
J. Stoer, R. Bulirsch. Springer (2002)

[73] *Physical fluid dynamics.*
D.J. Tritton. Springer Science and Business Media (2012)

[74] *POD-Galerkin reduced-order modeling with adaptive finite element snapshots.*
S. Ullmann, M. Rotkvic, J. Lang. Journal of Computational Physics, volume 325, pages 244-258. Elsevier (2016)

[75] *On computing rational Gauss-Chebyshev quadrature formulas.*
J. Van Deun, A. Bultheel, P. González. Mathematics of Computation, volume 75, number 253, pages 307-326 (2006)

[76] *Iterative solvers and inflow boundary conditions for plane sudden expansion flow.*
E. Wahba. Applied Mathematical Modelling, volume 31, number 11, pages 2553-2563. Elsevier (2007)

[77] *Turbulence modeling for CFD.*
D.C Wilcox et al. Volume 2. DCW industries La Canada, CA (1998)

[78] *Nektar++: Online documentation.*
Accessed 01/09/2019, <http://doc.nektar.info/doxygen/3.3/annotated.html>