NTNU
Norwegian University of
Science and Technology

A Reduced Basis Method for steady Stokes flow

Lars Snekkerhaugen

Master of Science in Physics and Mathematics
Submission date: July 2018
Supervisor: Trond Kvamsdal, IMF
Co-supervisor: Kjetil André Johannessen, SINTEF

Norwegian University of Science and Technology
Department of Mathematical Sciences

Abstract

In light of the increasing demand for many-query and real-time PDE solutions, reduced basis methods stand as a promising technique for developing solvers with the desired performance. In this thesis, we present the theory to implement both a finite element method and a proper orthogonal decomposition (POD) Galerkin reduced basis method. Numerical analysis is done for the parametrized steady Stokes equations. For flow around a NACA airfoil, we find that only a small number of reduced basis functions N_u and N_p for velocity and pressure, respectively, are needed to obtain a reduced solution of sufficient accuracy. Reduced basis methods based on inf-sup stable finite element solutions do not generally inherit the inf-sup stability of the underlying finite element method. For the reduced basis method, we find a region of different values N_u and N_p, where the reduced solutions are stable. The reduced basis method gives a reduction in degrees of freedom of $1500 : 1$ and $360 : 1$ for the reduced velocity and the reduced pressure, respectively. This reduction gives an attractive speedup of order $O(10^5)$ compared to the finite element method. In order to expand the stable region of different values N_u and N_p, a reduced basis method with supremizer stabilization is implemented. Enriching the reduced velocity space by supremizers do in fact expand the stable region, but it introduces additional degrees of freedom to the performance-critical online stage, slowing it down.

Sammendrag

I lyset av økende etterspørsel etter flerforespørselsløsninger og sanntidsløsninger av PDEer står reduserte basis-metoder som en lovende teknikk for å utvikle løsere med ønsket ytelse. I denne avhandlingen presenterer vi teorien som ligger til grunn for å implementere både en endelig element-metode og en "ekte" ortogonal dekomposisjon (POD) Galerkin redusert basis-metode. Numeriske analyser er gjort for de parametriserte stasjonære Stokes likningene. For flyt rundt en NACA vingeprofil finner vi ut at kun et lite antall reduserte basisfunskjoner N_u og N_p for hastighet og trykk respektivt, er nødvendig for å oppnå en redusert løsning med god nok nøyaktighet. Reduserte basisfunksjoner som er basert på inf-sup stabile endelig element-løsninger arver generelt ikke inf-sup stabiliteten til den underliggende endelige element-metoden. For den reduserte basis-metoden finner vi en region av forskjellige verdier N_u og N_p der de reduserte basisløsningene er stabile. Den reduserte basis-metoden gir en reduksjon i antall frihetsgrader på $1500 : 1$ og $360 : 1$ for den reduserte hastigheten og det reduserte trykket respektivt. Denne reduksjonen gjør at den reduserte basis-metoden løses i størrelsesorden 10^5 ganger raskere enn den endelige element-metoden. For å utvide den stabile regionen av ulike verdier N_u og N_p implementeres en redusert basis-metode med maksimator-stabilisering. Ved å berike det reduserte hastighetsrommet med maksimatorer utvides faktisk den stabile regionen, men det fører også med seg flere frihetsgrader som gjør det ytelseskritiske online-steget tregere.

Preface

This thesis is the result of my work with the course *TMA4910 Numerical Mathematics, Master Thesis* at the Norwegian University of Science and Technology (NTNU), Department of Mathematical Sciences. The work has been carried out in collaboration with SINTEF during the spring of 2018 and completes my time as a student at NTNU.

I would like to thank my supervisor Trond Kvamsdal, who has contributed as an advisor with useful advice and suggestions throughout this thesis. Without him, I would never have been introduced to SINTEF and reduced order modeling in the first place. Also, I am in debt to Kjetil André Johannessen, my co-supervisor from SINTEF during this thesis. Without our weekly meetings, his constant feedback, advise and inspiration, this journey would not have been as fun as it has. Also, I owe a big thanks to Eivind Fonn my "reduced basis guru". Thank you for letting me ask all the stupid questions needed to be asked during a process like this, and of course, thank you for always being available on email and skype every day in the week. Without you Eivind, Kjetil and Trond, I would never have gotten this far with this thesis. Also a large thanks to Adil Rasheed for letting me work with SINTEF in the first place, for this I am truly grateful. Last, but not least, I would like to thank Solveig for her understanding and support during late working hours and still being strong enough to pull me out of bed the next morning.

Lars Snekkerhaugen
Trondheim
July 2018

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Over recent decades the importance of numerical simulations of partial differential equations (PDEs) has increased and the numerical models are getting more complex. This is due to the increased computational power of computers, and also because of the development of numerical algorithms caused by iterative solution techniques for linear systems, such as the conjugate gradient method, preconditioned gradient methods, and multigrid methods [57].

Numerical simulations of complex models tend to yield a large number of degrees of freedom which in turn gives time-consuming and computationally costly methods [10]. Well-established techniques for simulating PDEs exist, such as finite elements and finite volumes [30]. Let us denote these classical models by high - fidelity models. With a computation time of several hours, or even days, high - fidelity methods are no longer sufficient for real-time problems or many - query problems [31], unless an expensive high - performance computing device is provided. Examples of real-time problems or many - query problems are optimal control problems [42], parameter estimation [32] and shape optimization [40].

Common for these type of problems is that they depend on one or more input parameter(s) associated with the output(s) of interest. Typical inputs specify the geometry, physical properties, boundary conditions and loads/sources [32, 54]. This parameter dependency is the key to deal with the increasing demand for real-time many - query models.

Reduced order modeling (ROM), also often referred to as model order reduction (MOR) in the literature, is a fast-evolving field. The idea behind ROM is to transform a given high - fidelity problem to an equivalent low dimensional reduced problem independent of the dimension of the high - fidelity problem and hence computationally inexpensive. One of the most interesting models in this framework is the *reduced basis method*. The reduced basis method is not a new method. It was first introduced by Almroth et al. in the late 1970s [4] and further developed by Noor [43, 44] and Noor and Peters [45].

The reduced basis method transforms a high - fidelity problem to an equivalent low dimensional reduced problem by a projection onto a subspace spanned by some solutions of the governing PDE for a selected set of parameters [53]. Different projection methods can be used, e.g. Galerkin reduced basis method or Least - squares reduced basis method [49]. Two major techniques for building the reduced space, that the original problem is projected onto, are *proper orthogonal decomposition* (POD) [49, 20] and *greedy algorithms* [34]. The beauty of the reduced basis method becomes clear when the parameter - dependency of the problem makes it possible to separate the computations into an expensive parameter - independent offline stage and an inexpensive online stage, depending on the desired input parameter.

A field of growing demand for time - effective flow simulations is the field of renewable energy and particularly the wind sector. Simulating flow around objects such as wind turbines is a time - consuming process. Even though many methods already exist they often require high - performance computers [31, 46]. The governing equations in the field of flow simulations are the Navier - Stokes equations. Recent work that is done on the reduced basis method for the Navier - Stokes equations can be found in e.g. [8, 31, 49, 24, 7]. Due to the limited time horizon for this thesis, the scope is set to the linear case of the Navier - Stokes equations namely the steady Stokes equations. This is done as it is desired to implement own solvers for the high - fidelity approximation and the reduced basis approximation to carry out the numerical analysis. The outputs of interest for the steady Stokes problem are the velocity field and pressure field. The reduced basis method has also been developed for the steady Stokes method and some of the more recent contributions can be found in e.g. [49, 53, 52, 37].

In this thesis, we first investigate the Galerkin finite element method based on isogeometric analysis [22, 48]. We focus on how to build the finite element spaces in order to obtain stable and optimal convergent solutions [48, 17]. By stable solutions we mean in the sense of the LBB - condition (Ladyzhenskaya - Babuška - Brezzi condition) [14]. An important part of the thesis is to get to know parametrized saddle - point problems in order to build a proper finite element solver for the steady Stokes problem. The solver is necessary to conduct the numerical analysis given, but it is also needed as a strong foundation for the implementation of an RB solver. We build a solver for the steady Stokes equations where the input parameters are both physical and geometrical. Therefore we study in detail a parametric map from a parameter - independent domain to a parameter - dependent domain and we study how to deal with this map when deriving the equations for the reduced basis solver.

Further, we investigate the offline-online decomposition given by the parametric - dependence [49, 54]. In detail, we evaluate how to sample input parameters in order to create a set of high - fidelity (HF) solutions that gives a good representation of the entire solution space. Both regular methods [49, 62] and statistical methods [41, 56] are investigated. The set of HF solutions, known as *snapshots*, is created during the offline - stage. A POD of the snapshots is performed in order to extract low dimensional reduced spaces for the velocity and the pressure method. During the online - stage a reduced order approximation of both the velocity and the pressure field is obtained.

We conduct numerical error analysis in order to evaluate the stability and the accuracy of the reduced solutions. This is done by comparing the reduced solutions to the high - fidelity solutions. In particular, we seek to investigate for what choice of number of reduced basis functions are the reduced solutions unstable and more important when are they not. As reduced basis methods are one of the first choices of methods to investigate in the need of real-time solutions we report on the computation time of the reduced solver compared to the high - fidelity solver.

In order to deal with pressure instability in the reduced basis context, we also investigate the method of enriching the reduced velocity space with supremizers. We seek to fulfill a reduced version of the LBB - condition by implementing the approximate supremizer stabilization technique within a POD context [8]. Approximate supremizer stabilization is used in order to preserve the offline-online decomposition of the computations. Numerical error analysis is carried out in the same fashion as for the classical reduced basis method. The computation time is reported in order to compare the performance of the two methods.

In Chapter 2 we study the saddle - point problem and its stability. We also present basic theory of the finite element method, isogeometric analysis and thorough discussion on the theory of reduced basis method with POD. This provides us with the necessary theory used to build the isogeometric based finite element solver and the POD based reduced basis solver.

In Chapter 3 we give two examples of non-parametrized Stokes equations. The first example provides us with a verification of the finite element solver, whereas the second example introduces Stokes flow around a NACA 4620 airfoil. Chapter 4 is devoted entirely to a parametrized version of the second example, comparing a finite element solution and a reduced basis solution for this example. Chapter 5 concludes this thesis with a discussion of the results from the examples given and presents ideas for future work.

Chapter 2

Theory

This chapter provides the reader with some basic knowledge on the theory this master thesis is built upon. As the steady Stokes equations can be recognized as a saddle-point problem, this chapter gives some theory for parametrized saddle-point problems. An effort is made in order to develop a finite element solver, solving a non-parametrized steady Stokes problem before moving on to a parametrized steady Stokes problem. It is worth noting that conditions given for existence and uniqueness of solutions to parametrized saddle-point problems, has a very similar counterpart in the non-parametrized case. In Chapter 3 we present some further theory and examples on a type of saddle-point problem, namely the steady Stokes equations.

At the center of attention for reduced basis methods are parameterized partial differential equations. These are PDEs depending on some set of parameters. This could be physical parameters of the system considered such as the Reynolds number in nonlinear viscous fluids modelled by the Navier-Stokes equations, the thermal conductivity of the heat equation, boundary conditions and/or source- and sink terms [34, 49]. Let us denote the physical parameters by a vector $\boldsymbol{\mu_{ph}}$. Also the computational domain itself could be represented by parameters such as scaling parameters for stretching and/or compressing. We will denote such geometric parameters by a vector $\boldsymbol{\mu_g}$. Usually we will not make the distinction between physical and geometric parameters, unless it is necessary, and hence simply use the vector $\boldsymbol{\mu}$ for the parameters upon which the PDEs depend on. Note that this will also be done for spaces: given the space $\mathbb{Y}(\boldsymbol{\mu_g})$ depending on some geometric parameters $\boldsymbol{\mu_g}$ we will for simplicity denote it $\mathbb{Y}(\boldsymbol{\mu})$. We denote the parameter space $\mathbb{P}$ such that $\boldsymbol{\mu} \in \mathbb{P} \subset \mathbb{R}$.

2.1 Mixed variational Problems

Mixed variational problems, or also called saddle point problems, is a class of problems that arise naturally in many different fields e.g. constrained optimization, optimal control problems and fluid dynamics such as the Stokes equations which this thesis mainly focuses on [59, 11, 49].

2.1.1 Abstract weak formulation

A saddle point problem can be cast in an abstract weak formulation such as the following. Let $\mathbb{X}(\boldsymbol{\mu})$ and $\mathbb{Q}(\boldsymbol{\mu})$ be two Hilbert spaces Note that they are parameter-dependent as the domain is assumed to be parameter-dependent. Further let $\mathbb{X}'(\boldsymbol{\mu})$ and $\mathbb{Q}'(\boldsymbol{\mu})$ be their respective dual spaces, that is the spaces of linear and bounded functionals defined on $\mathbb{X}(\boldsymbol{\mu})$ and $\mathbb{Q}(\boldsymbol{\mu})$ respectively. Given two bilinear forms $d(\cdot,\cdot;\boldsymbol{\mu}) : \mathbb{X}(\boldsymbol{\mu}) \times \mathbb{X}(\boldsymbol{\mu}) \to \mathbb{R}$ and $b(\cdot,\cdot;\boldsymbol{\mu}) : \mathbb{X}(\boldsymbol{\mu}) \times \mathbb{Q}(\boldsymbol{\mu}) \to \mathbb{R}$ and two bounded linear functionals $f_1(\cdot;\boldsymbol{\mu}) \in \mathbb{X}'(\boldsymbol{\mu})$ and $f_2(\cdot;\boldsymbol{\mu}) \in \mathbb{Q}'(\boldsymbol{\mu})$ we consider the problem:
given $\boldsymbol{\mu} \in \mathbb{P}$ find $(x, p) \in \mathbb{X} \times \mathbb{Q}$ such that

$$\begin{aligned} d(x(\boldsymbol{\mu}), w(\boldsymbol{\mu}); \boldsymbol{\mu}) + b(w, p(\boldsymbol{\mu}); \boldsymbol{\mu}) &= f_1(w; \boldsymbol{\mu}) \qquad \forall w \in \mathbb{X}(\boldsymbol{\mu}), \\ b(x(\boldsymbol{\mu}), q; \boldsymbol{\mu}) &= f_2(q; \boldsymbol{\mu}) \qquad \forall q \in \mathbb{Q}(\boldsymbol{\mu}), \end{aligned} \tag{2.1}$$

Throughout the thesis $(x(\boldsymbol{\mu}), p(\boldsymbol{\mu}))$ will be referred to as *the exact solution* of the physical problem modelled by the equations (2.1), and $w \in \mathbb{X}(\boldsymbol{\mu}), q \in \mathbb{Q}(\boldsymbol{\mu})$ are called test functions belonging to the same Hilbert spaces as the solutions $(x(\boldsymbol{\mu}), p(\boldsymbol{\mu}))$ respectively. The saddle-point problem models a given problem in a given domain $\Omega \in \mathbb{R}^d$, with $d = 2, 3$. We deonote by $\mathbf{x}$ the spatial coordinates of the domain, that is $\mathbf{x} \in \Omega$. In this thesis we look at the 2D case. A typical problem which can be cast in the framework (2.1) is the Stokes equations, which is analyzed by e.g. Quarteroni in [48] or Braess in [14] and for the parametric case we refer to the work done by Manzoni in [39]. From here on we will assume problem (2.1) to be set on a parameter-independent domain, such that $\mathbb{X}(\boldsymbol{\mu}) \times \mathbb{Q}(\boldsymbol{\mu}) = \mathbb{X} \times \mathbb{Q}$. In Section 2.2.5 we will show how to solve a problem set on a parameter-dependent domain.

Varying $\boldsymbol{\mu}$ over the parameter space $\mathbb{P}$ we create a set of all solutions to problem (2.1). This set is referred to as the solution manifold in [34] and is the set

$$\mathbb{M} = \{(x(\boldsymbol{\mu}), p(\boldsymbol{\mu})) | \boldsymbol{\mu} \in \mathbb{P}\} \subset \mathbb{X} \times \mathbb{Q}. \tag{2.2}$$

Braess gives a brief description as to why problem (2.1) is in fact a saddle point problem [14]. For a fixed $\boldsymbol{\mu} = \boldsymbol{\mu_f}$ the couple $(x, p) = (x(\boldsymbol{\mu_f}), p(\boldsymbol{\mu_f}))$ is a solution of problem (2.1) if and only if it is a saddle point of the Lagrangian function

$$\mathcal{L}(w, q) = \frac{1}{2} d(w, w; \boldsymbol{\mu_f}) + b(w, q; \boldsymbol{\mu_f}) - f_1(w; \boldsymbol{\mu_f}) - f_2(q; \boldsymbol{\mu_f}),$$

i.e. every solution (x, p) of problem (2.1) satisfies the following saddle point property

$$\mathcal{L}(x, q) \leq \mathcal{L}(x, p) \leq \mathcal{L}(w, p).$$

For this to hold the bilinear forms, $d(\cdot,\cdot;) = d(\cdot,\cdot;\boldsymbol{\mu_f})$ and $b(\cdot,\cdot) = b(\cdot,\cdot;\boldsymbol{\mu_f})$ are assumed continuous and the former is assumed non-negative $d(w, w; \boldsymbol{\mu_f}) \geq 0$, as stated in [14]. For further details on the saddle-point problem as a Lagrangian function, see e.g. [14, 48, 49].

2.1.2 Existence, Uniqueness and Stability

Let us now define the continuity of the bilinear forms $d(\cdot,\cdot;\boldsymbol{\mu})$ and $b(\cdot,\cdot;\boldsymbol{\mu})$ more formally. We endow the two Hilbert spaces $\mathbb{X}$ and $\mathbb{Q}$ with the norms $||\cdot||_{\mathbb{X}}$ and $||\cdot||_{\mathbb{Q}}$ respectively. Then the assumption that the bilinear forms $d(\cdot,\cdot;\boldsymbol{\mu})$ and $b(\cdot,\cdot;\boldsymbol{\mu})$ are continuous is the same as saying there exists two constants $\gamma_d(\boldsymbol{\mu}), \gamma_b(\boldsymbol{\mu}) > 0$ such that for any $\boldsymbol{\mu} \in \mathbb{P}$

$$|d(x,w;\boldsymbol{\mu})| \leq \gamma_d(\boldsymbol{\mu})\|x\|_{\mathbb{X}}\|w\|_{\mathbb{X}}, \quad |b(w,q;\boldsymbol{\mu})| \leq \gamma_b(\boldsymbol{\mu})\|w\|_{\mathbb{X}}\|q\|_{\mathbb{Q}} \qquad \forall x, w \in \mathbb{X},\ \forall q \in \mathbb{Q}. \tag{2.3}$$

Following the work done by Quarteroni et al. in [49] we introduce the space $\mathbb{X}^0$

$$\mathbb{X}^0 = \{w \in \mathbb{X} : b(w,q;\boldsymbol{\mu}) = 0\ \forall q \in \mathbb{Q},\ \forall \boldsymbol{\mu} \in \mathbb{P}\},$$

as indeed must be a subspace of $\mathbb{X}$, i.e. $\mathbb{X}^0 \subset \mathbb{X}$. This provides us with a less strict coercivity condition necessary for existence of a unique solution than could have been. For a given $\boldsymbol{\mu} \in \mathbb{P}$ there exist a unique solution $(x(\boldsymbol{\mu}), p(\boldsymbol{\mu})) \in \mathbb{X}\times\mathbb{Q}$ to the saddle point problem (2.1) if the continuity assumptions together with the following assumptions hold.

First we assume the bilinear form $d(\cdot,\cdot;\boldsymbol{\mu})$ to be coercive on the space $\mathbb{X}^0$. That is there exist a constant $\alpha(\boldsymbol{\mu}) > 0$ such that

$$d(w,w;\boldsymbol{\mu}) > \alpha(\boldsymbol{\mu})\|w\|^2_{\mathbb{X}} \quad \forall w \in \mathbb{X}^0,\ \forall \boldsymbol{\mu} \in \mathbb{P}. \tag{2.4}$$

Second we assume there exist a constant $\beta(\boldsymbol{\mu}) > 0$ such that the bilinear form $b(\cdot,\cdot;\boldsymbol{\mu})$ satisfies the condition

$$\inf_{\substack{q\in\mathbb{Q}\\ \|q\|_{\mathbb{Q}}\neq 0}} \ \sup_{\substack{w\in\mathbb{X}\\ \|w\|_{\mathbb{X}}\neq 0}} \frac{b(w,q;\boldsymbol{\mu})}{\|w\|_{\mathbb{X}}\|q\|_{\mathbb{Q}}} \geq \beta(\boldsymbol{\mu}) > 0 \quad \forall \boldsymbol{\mu} \in \mathbb{P}. \tag{2.5}$$

The above condition is often referred to as the inf-sup condtion or the LBB-condition (Ladyzhenskaya-Babuška-Brezzi condition). For a deeper analysis we refer to e.g. [39, 48]. For a proof of the existence of a unique solution $(x(\boldsymbol{\mu}), p(\boldsymbol{\mu}))$ see e.g. [48] or [14].

Remark 2.1.1. *We note that as $\mathbb{X}^0$ is a subspace of $\mathbb{X}$, then if the bilinear form $d(\cdot,\cdot;\boldsymbol{\mu})$ is coercive over the space $\mathbb{X}$, then condition* (2.4) *automatically holds. This is in fact the case for the Stokes equations as we will see later on.*

For a fixed parameter $\boldsymbol{\mu} = \boldsymbol{\mu_f}$ such that $(x,p) = (x(\boldsymbol{\mu_f}), p(\boldsymbol{\mu_f}))$, $\alpha = \alpha(\boldsymbol{\mu_f})$, $\gamma_d = \gamma_d(\boldsymbol{\mu_f})$ and $\beta = \beta(\boldsymbol{\mu_f})$ not only does the continuity conditions (2.3), the coercivity condition (2.4) and the inf-sup condition (2.5) provide existence of a unique solution, moreover it provides the stability estimates

$$\begin{aligned} \|x\|_{\mathbb{X}} \leq & \frac{1}{\alpha}\left(\|f_1(\cdot;\boldsymbol{\mu_f})\|_{\mathbb{X}'} + \frac{\alpha+\gamma_d}{\beta}\|f_2(\cdot;\boldsymbol{\mu_f})\|_{\mathbb{Q}'}\right) \\ \|p\|_{\mathbb{X}} \leq & \frac{1}{\beta}\left(\left(1+\frac{\gamma}{\alpha}\right)\|f_1(\cdot;\boldsymbol{\mu_f})\|_{\mathbb{X}'} + \frac{\gamma_d(\alpha+\gamma_d)}{\alpha\beta}\|f_2(\cdot;\boldsymbol{\mu_f})\|_{\mathbb{Q}'}\right) \end{aligned} \tag{2.6}$$

The proof of the estimates in equations (2.6) is found in [48].

2.2 The Galerkin Finite Element Method

The examples given in Chapter 3 are solved using a finite element solver based on isogeometric analysis using *B-spline basis functions* and not a classical finite element solver. In this section we will present the details on deriving a linear system from the *Galerkin problem* as this sets the foundation for the *Galerkin reduced basis* problem.

2.2.1 Galerkin approximation

In order to numerically solve an infinite problem which can be written in the framework of the weak formulation (2.1) of a saddle point problem it must be discretized. This is done by introducing the finite dimensional subspaces $\mathbb{X}_h$ and $\mathbb{Q}_h$ of the Hilbert spaces $\mathbb{X}$ and $\mathbb{Q}$ respectively, already introduced in Section 2.1. In our case these new approximation spaces will be finite element piecewise polynomial spaces, discretized with grid size h. We seek *the (finite) Galerkin approximations* $(x_h(\boldsymbol{\mu}), p_h(\boldsymbol{\mu})) \in \mathbb{X}_h \times \mathbb{Q}_h$ of the truth solutions $(x(\boldsymbol{\mu}), p(\boldsymbol{\mu}))$. The Galerkin approximations are also often called *high-fidelity solutions*. Then for a problem in a domain Ω that can be cast in the abstract form (2.1), we introduce the Galerkin approximation:
for a given $\boldsymbol{\mu} \in \mathbb{P}$ find $(x_h(\boldsymbol{\mu}), p_h(\boldsymbol{\mu})) \in \mathbb{X}_h \times \mathbb{Q}_h$ such that

$$\begin{aligned} d(x_h(\boldsymbol{\mu}), w_h; \boldsymbol{\mu}) + b(w_h, p_h(\boldsymbol{\mu}); \boldsymbol{\mu}) &= f_1(w_h; \boldsymbol{\mu}) \qquad \forall w_h \in \mathbb{X}_h, \\ b(x_h(\boldsymbol{\mu}), q_h; \boldsymbol{\mu}) &= f_2(q_h; \boldsymbol{\mu}) \qquad \forall q_h \in \mathbb{Q}_h, \end{aligned} \tag{2.7}$$

where the subscript h is related to the grid size of the high-fidelity problem. Now we can define the discrete version of the continuous solution manifold (2.2) as following

$$\mathbb{M}_h = \{(x_h(\boldsymbol{\mu}), p(\boldsymbol{\mu}))|\ \boldsymbol{\mu} \in \mathbb{P}\} \subset \mathbb{X}_h \times \mathbb{Q}_h. \tag{2.8}$$

Obviously the discrete solution manifold is a subset of the exact solution mainfold, i.e. $\mathbb{M}_h \subset \mathbb{M}$, and we assume that choosing the discretization fine enough, or equivalently choosing h small enough, we can approximate $\mathbb{M}$ by $\mathbb{M}_h$ within an acceptable approximation error.

Let N_h and M_h be the dimension of the approximation spaces such that $N_h = \dim(\mathbb{X}_h)$ and $M_h = \dim(\mathbb{Q}_h)$. Then let $\{\phi_i\}_{i=1}^{N_h}$ and $\{\psi_i\}_{i=1}^{M_h}$ denote two bases for $\mathbb{X}_h$ and $\mathbb{Q}_h$ respectively. Denote the spatial coordinates $\mathbf{x} \in \Omega$. Now we can write our high-fidelity solutions as

$$x_h(\boldsymbol{\mu}) = x_h(\mathbf{x}; \boldsymbol{\mu}) = \sum_{i=1}^{N_h} x_h(\boldsymbol{\mu})^i \phi_i(\mathbf{x}) \qquad p_h(\boldsymbol{\mu}) = p_h(\mathbf{x}; \boldsymbol{\mu}) = \sum_{i=1}^{M_h} p_h(\boldsymbol{\mu})^i \psi_i(\mathbf{x}) \tag{2.9}$$

where

$$\mathbf{x_h}(\boldsymbol{\mu}) = [x_h(\boldsymbol{\mu})^1, \ldots, x_h(\boldsymbol{\mu})^{N_h}]^\top, \qquad \mathbf{p_h}(\boldsymbol{\mu}) = [p_h(\boldsymbol{\mu})^1, \ldots, p_h(\boldsymbol{\mu})^{M_h}]^\top$$

holds the coefficients associated with the degrees of freedom of $x_h(\boldsymbol{\mu})$ and $p_h(\boldsymbol{\mu})$ respectively. Further we can write the test functions as $w_h = \phi_i(\mathbf{x})$ and $q_h = \psi_l(\mathbf{x})$ since equation (2.7) holds for all $w_h \in \mathbb{X}_h$ and all $q_h \in \mathbb{Q}_h$. Inserted in the Galerkin problem (2.7)

$$\begin{gathered} d\left(\sum_{j=1}^{N_h} x_h(\boldsymbol{\mu})^j \phi_j(\mathbf{x}), \phi_i(\mathbf{x}); \boldsymbol{\mu}\right) + b\left(\phi_i(\mathbf{x}), \sum_{k=1}^{M_h} p_h(\boldsymbol{\mu})^k \psi_k(\mathbf{x}); \boldsymbol{\mu}\right) = f_1\left(\phi_i(\mathbf{x}); \boldsymbol{\mu}\right) \quad i = 1, \ldots, N_h \\ \sum_{j=1}^{N_h} d\left(\phi_j(\mathbf{x}), \phi_i(\mathbf{x}); \boldsymbol{\mu}\right) x_h(\boldsymbol{\mu})^j + \sum_{k=1}^{M_h} b(\phi_i(\mathbf{x}), \psi_k(\mathbf{x}); \boldsymbol{\mu}) p_h(\boldsymbol{\mu})^k = f_1(\phi_i(\mathbf{x}); \boldsymbol{\mu}) \quad i = 1, \ldots, N_h \\ D_h(\boldsymbol{\mu})\mathbf{x_h}(\boldsymbol{\mu}) + B_h^\top(\boldsymbol{\mu})\mathbf{p_h}(\boldsymbol{\mu}) = \mathbf{f_{1h}}(\boldsymbol{\mu}) \end{gathered} \tag{2.10}$$

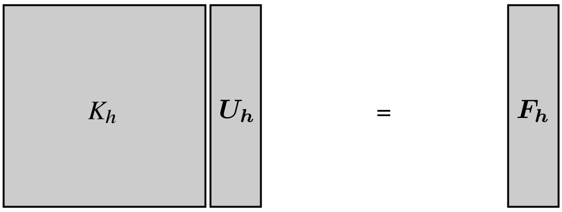


Figure 2.1: The linear system in equation (2.12) with $K_h = K_h(\boldsymbol{\mu}) \in \mathbb{R}^{N_h+M_h \times N_h+M_h}$, $\mathbf{U_h} = \mathbf{U_h}(\boldsymbol{\mu}) \in \mathbb{R}^{N_h+M_h}$ and $\mathbf{F_h} = \mathbf{F_h}(\boldsymbol{\mu}) \in \mathbb{R}^{N_h+M_h}$.

and

$$\begin{gathered}
b\left(\sum_{j=1}^{N_h} x_h(\boldsymbol{\mu})^j \phi_j(\mathbf{x}), \psi_l(\mathbf{x}); \boldsymbol{\mu}\right) = f_2\left(\psi_l(\mathbf{x}); \boldsymbol{\mu}\right) \quad l = 1, \ldots, M_h \\
\sum_{j=1}^{N_h} b\left(\phi_j(\mathbf{x}), \psi_l(\mathbf{x}); \boldsymbol{\mu}\right) x_h(\boldsymbol{\mu})^j = f_2(\psi_l(\mathbf{x}); \boldsymbol{\mu}) \quad l = 1, \ldots, M_h. \\
B_h(\boldsymbol{\mu})\mathbf{x_h}(\boldsymbol{\mu}) = \mathbf{f_{2h}}(\boldsymbol{\mu})
\end{gathered} \tag{2.11}$$

That is the Galerkin high-fidelity approximation is equivalent to solving the linear system

$$\underbrace{\begin{bmatrix} D_h(\boldsymbol{\mu}) & B_h^\top(\boldsymbol{\mu}) \\ B_h(\boldsymbol{\mu}) & 0 \end{bmatrix}}_{K_h(\boldsymbol{\mu})} \underbrace{\begin{bmatrix} \mathbf{x_h}(\boldsymbol{\mu}) \\ \mathbf{p_h}(\boldsymbol{\mu}) \end{bmatrix}}_{\mathbf{U_h}(\boldsymbol{\mu})} = \underbrace{\begin{bmatrix} \mathbf{f_{1h}}(\boldsymbol{\mu}) \\ \mathbf{f_{2h}}(\boldsymbol{\mu}) \end{bmatrix}}_{\mathbf{F_h}(\boldsymbol{\mu})}, \tag{2.12}$$

where the matrices $D_h(\boldsymbol{\mu}) \in \mathbb{R}^{N_h \times N_h}$ and $B_h(\boldsymbol{\mu}) \in \mathbb{R}^{M_h \times N_h}$ holds the elements

$$(D_h(\boldsymbol{\mu}))_{ij} = d(\phi_j, \phi_i; \boldsymbol{\mu}), \quad 1 \leq i, j \leq N_h \qquad (B_h(\boldsymbol{\mu}))_{lj} = b(\phi_j, \psi_l; \boldsymbol{\mu}), \quad 1 \leq j \leq N_h,\ 1 \leq l \leq M_h. \tag{2.13}$$

$\mathbf{f}_{1h}(\boldsymbol{\mu}) \in \mathbb{R}^{N_h}$ and $\mathbf{f}_{2h}(\boldsymbol{\mu}) \in \mathbb{R}^{M_h}$ denote the vectors with elements

$$(\mathbf{f}_{1h}(\boldsymbol{\mu}))_i = f_1(\phi_i; \boldsymbol{\mu}), \quad 1 \leq i \leq N_h \qquad (\mathbf{f}_{2h}(\boldsymbol{\mu}))_l = f_2(\psi_l; \boldsymbol{\mu}), \qquad 1 \leq l \leq M_h. \tag{2.14}$$

The linear system (2.7) of dimension $N_h + M_h$ is shown in Figure 2.1.

2.2.2 Uniqueness and stability of the Galerkin high-fidelity approximation

Following what we did in Section 2.1.1 we define the discrete counterpart to the space $\mathbb{X}^0$, namely

$$\mathbb{X}_h^0 = \{w_h \in \mathbb{X}_h : b(w_h, q_h; \boldsymbol{\mu}) = 0, \quad \forall q_h \in Q_h,\ \forall \boldsymbol{\mu} \in \mathbb{P}\}.$$

We still assume $f_1(\cdot;\boldsymbol{\mu})$ and $f_2(\cdot;\boldsymbol{\mu})$ to be bounded linear functionals. If the assumption that conditions (2.3) holds, then the bilinear forms $d(\cdot,\cdot;\boldsymbol{\mu})$ and $b(\cdot,\cdot;\boldsymbol{\mu})$ are also continuous on $\mathbb{X}_h \times \mathbb{X}_h$ and $\mathbb{X}_h \times \mathbb{Q}_h$ respectively as $\mathbb{X}_h \subset \mathbb{X}$ and $\mathbb{Q}_h \subset \mathbb{Q}$.

In addition to assuming that the continuity condition holds we assume the following two hold. First assume the bilinear form $d(\cdot,\cdot;\boldsymbol{\mu})$ is coercive on the space $\mathbb{X}_h^0$, that is there exists a constant $\alpha_h(\boldsymbol{\mu}) > 0$ such that

$$d(w_h, w_h; \boldsymbol{\mu}) > \alpha_h(\boldsymbol{\mu}) \|w_h\|_{\mathbb{X}}^2 \quad \forall w_h \in \mathbb{X}_h^0, \quad \forall \boldsymbol{\mu} \in \mathbb{P}. \tag{2.15}$$

Note that this condition does not necessarily hold even if the coercivity condition (2.4) holds in the continuous case and that is because $\mathbb{X}_h^0 \not\subset \mathbb{X}^0$.

Second assume there exist a constant $\beta_h(\boldsymbol{\mu}) > 0$ such that the bilinear form $b(\cdot,\cdot;\boldsymbol{\mu})$ satisfies the discrete inf-sup-condition

$$\beta_h^s(\boldsymbol{\mu}) = \inf_{\substack{q_h \in \mathbb{Q}_h \\ \|q_h\|_{\mathbb{Q}_h} \neq 0}} \sup_{\substack{w_h \in \mathbb{X}_h \\ \|w_h\|_{\mathbb{X}_h} \neq 0}} \frac{b(w_h, q_h; \boldsymbol{\mu})}{\|w_h\|_{\mathbb{X}_h} \|q_h\|_{\mathbb{Q}_h}} \geq \beta_h(\boldsymbol{\mu}) > 0 \quad \forall \boldsymbol{\mu} \in \mathbb{P}. \tag{2.16}$$

Then problem (2.1) has a unique solution $(x_h(\boldsymbol{\mu}), p_h(\boldsymbol{\mu}))$. The coercivity condition and the inf-sup condition are not automatically fullfilled for weakly coercive problems even if they hold for the continuous case. This is why they are explicitly given on the discrete spaces $\mathbb{X}_h$ and $\mathbb{Q}_h$. Further details are given by Manzoni in [39], whereas a proof for the parameter-independent case is found in [48].

For a fixed parameter $\boldsymbol{\mu} = \boldsymbol{\mu_f}$ such that $(x_h, p_h) = (x_h(\boldsymbol{\mu_f}), p_h(\boldsymbol{\mu_f}))$, $\alpha_h = \alpha_h(\boldsymbol{\mu_f})$, $\gamma_d = \gamma_d(\boldsymbol{\mu_f})$ and $\beta_h = \beta_h(\boldsymbol{\mu_f})$, if the constants α_h and β_h are independent of h the following stability estimates are provided

$$\begin{aligned} \|x_h\|_{\mathbb{X}_h} \leq & \frac{1}{\alpha_h}\left(\|f_1(\cdot;\boldsymbol{\mu_f})\|_{\mathbb{X}'} + \frac{\alpha_h + \gamma_d}{\beta_h} \|f_2(\cdot;\boldsymbol{\mu_f})\|_{\mathbb{Q}'} \right), \\ \|p_h\|_{\mathbb{X}} \leq & \frac{1}{\beta_h}\left(\left(1 + \frac{\gamma}{\alpha_h}\right) \|f_1(\cdot;\boldsymbol{\mu_f})\|_{\mathbb{X}'} + \frac{\gamma_d(\alpha_h + \gamma_d)}{\alpha_h \beta_h} \|f_2(\cdot;\boldsymbol{\mu_f})\|_{\mathbb{Q}'} \right). \end{aligned} \tag{2.17}$$

The proof of the estimates in equations (2.17) is the same as for the stability estimates (2.6) and is given by Quarteroni in [48].

Remark 2.2.1. *Provided that the constants α_h and β_h can be bounded from below by some constants c_1 and c_2 independent from h, then it is possible to provide error estimates $\|x - x_h\|_{\mathbb{X}}$ and $\|p - p_h\|_{\mathbb{Q}}$ with optimal convergence rate, see [48] for further details.*

Remark 2.2.2. *The inf-sup condition* (2.16) *is necessary to guarantee uniqueness of $p_h(\boldsymbol{\mu})$ in the solution (x_h, p_h). In cases where* (2.16) *does not hold it is possible to find functions $g_h(\boldsymbol{\mu}) \in \mathbb{Q}_h$ such that*

$$b(w_h, g_h(\boldsymbol{\mu})) = 0, \ \forall w_h \in \mathbb{X}_h,$$

then we see from the weak formulation (2.7) *that*

$$\begin{aligned} d(x_h(\boldsymbol{\mu}), w_h; \boldsymbol{\mu}) + b(w_h, p_h(\boldsymbol{\mu}); \boldsymbol{\mu}) + 0 &= f_1(w_h; \boldsymbol{\mu}) \qquad \forall w_h \in \mathbb{X}_h, \\ d(x_h(\boldsymbol{\mu}), w_h; \boldsymbol{\mu}) + b(w_h, p_h(\boldsymbol{\mu}); \boldsymbol{\mu}) + b(w_h, g_h(\boldsymbol{\mu}); \boldsymbol{\mu}) &= f_1(w_h; \boldsymbol{\mu}) \qquad \forall w_h \in \mathbb{X}_h, \\ d(x_h(\boldsymbol{\mu}), w_h, \boldsymbol{\mu})) + b(w_h, p_h(\boldsymbol{\mu}) + g_h(\boldsymbol{\mu}); \boldsymbol{\mu}) &= f_1(w_h; \boldsymbol{\mu}) \qquad \forall w_h \in \mathbb{X}_h, \end{aligned}$$

and indeed $(x_h(\boldsymbol{\mu}), p_h(\boldsymbol{\mu}) + g_h(\boldsymbol{\mu}))$ *would be a solution of problem* (2.7) *too. The functions* $g_h(\boldsymbol{\mu})$ *are called spurious modes and causes numerical instabilities, [48]. A possible way to deal with spurious modes is to choose the finite element spaces* $\mathbb{X}_h$ *and* $\mathbb{Q}_h$ *such that the inf-sup condition* (2.16) *holds. See Section 2.2.3 for further details.*

Manzoni also gives an alternative way to express the inf-sup condition (2.16) which comes in handy when we work with reduced order modelling. In [39] Manzoni introduces a parameter-dependent supremizer operator $T_p^{\boldsymbol{\mu}} : \mathbb{Q}_h \to \mathbb{X}_h$ defined by

$$(T_p^{\boldsymbol{\mu}} q_h, w_h)_{\mathbb{X}} = b(w_h, q_h; \boldsymbol{\mu}) \quad \forall w_h \in \mathbb{X}_h. \tag{2.18}$$

Then the inf-sup condition (2.16) is expressed as

$$T_p^{\boldsymbol{\mu}} q_h = \arg \sup_{w_h \in \mathbb{X}_h} \frac{b(q_h, w_h; \boldsymbol{\mu})}{\|w_h\|_{\mathbb{X}}} \quad \text{and} \quad \beta_h^s(\boldsymbol{\mu}) = \inf_{q_h \in \mathbb{Q}_h} \frac{\|T_p^{\boldsymbol{\mu}} q_h\|_{\mathbb{X}}}{\|q_h\|_{\mathbb{Q}}}$$

Let us further denote by $X_{h,x}$, $X_{h,p}$ the matrices associated with the scalar products in the spaces $\mathbb{X}$ and $\mathbb{Q}$ respectively. Then equation (2.18) can be cast in the linear system

$$X_{h,x}\mathbf{t}_{\mathbf{h}}^{\boldsymbol{\mu}}(\mathbf{q}_{\mathbf{h}}) = B_h^{\top}(\boldsymbol{\mu})\mathbf{q}_{\mathbf{h}} \quad \forall \mathbf{q}_{\mathbf{h}} \in \mathbb{R}^{M_h}. \tag{2.19}$$

Then the solution $\mathbf{t}_{\mathbf{h}}^{\boldsymbol{\mu}}(\mathbf{q}_{\mathbf{h}})$ of equation (2.19) given a $\mathbf{q}_{\mathbf{h}} \in \mathbb{R}^{M_h}$ is the supremizer realizing the inf-sup condition (2.16).

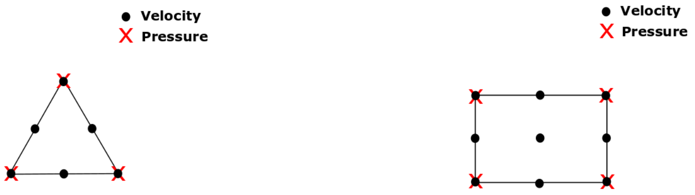
2.2.3 The Taylor-Hood element

It is well-known that finite element approximations of saddle-point problems can be meaningless because of the indefinite nature of the system. As mentioned in Remark 2.2.2 a possible way to handle this issue is by the proper choice of finite element spaces. This class of spaces are often called inf-sup stable elements and satisfy the discrete version of the LBB condition (2.16) making the finite element solutions stable [8]. Chapter 3 studies the steady Stokes equations, where the desired solutions are the velocity field and the pressure, hence it is necessary to find proper elements for the velocity space and the pressure space. The finite-element solver built for this uses a well-known inf-sup stable family of elements; the Taylor-Hood elements, providing continuous velocity and continuous pressure [48]. This section is no more than a brief introduction to the Taylor-Hood elements, for a more in-depth reading on this particular element or other elements see e.g. [14, 48, 16].

The Taylor-Hood element is a triangular or rectangular element where the velocity polynomial has one higher degree than the pressure polynomial. The pressure must at least be continuous so the least possible polynomial degree for the velocity is two. The Taylor-Hood element is often denoted $\mathbb{P}_k - \mathbb{P}_{k-1}$, $k \geq 2$ for triangular elements and $\mathbb{Q}_k - \mathbb{Q}_{k-1}$, $k \geq 2$ for rectangular elements. Here we have introduced the polynomial notation

$$\begin{aligned} \mathbb{P}_k &: \text{polynomials of degree } \leq k \\ \mathbb{Q}_k &: \text{polynomials of degree } \leq k \text{ in each variable.} \end{aligned}$$

The Taylor-Hood elements are inf-sup stable for $k \geq 2$, that means the inf-sup condition is satisfied and the finite element solutions are stable. For a proof of the inf-sup stability of



(a) $\mathbb{P}_2 - \mathbb{P}_1$.

(b) $\mathbb{Q}_2 - \mathbb{Q}_1$.

Figure 2.2: Example of Taylor-Hood elements where the black dots indicates the degree of freedom for the velocity and the red x indicates the degree of freedom for the pressure.

the $\mathbb{P}_k - \mathbb{P}_{k-1}$ elements see e.g. [16] and [15] for an outline of proof for the inf-sup stability of the $\mathbb{Q}_k - \mathbb{Q}_{k-1}$.

See Figure 2.2 for an example of Taylor-Hood elements $\mathbb{P}_2 - \mathbb{P}_1$ and $\mathbb{Q}_2 - \mathbb{Q}_1$ along with its nodes for the velocity and the pressure.

An important property of the Taylor-Hood elements is that they are optimal convergent [21, 50]. This gives us the following remark.

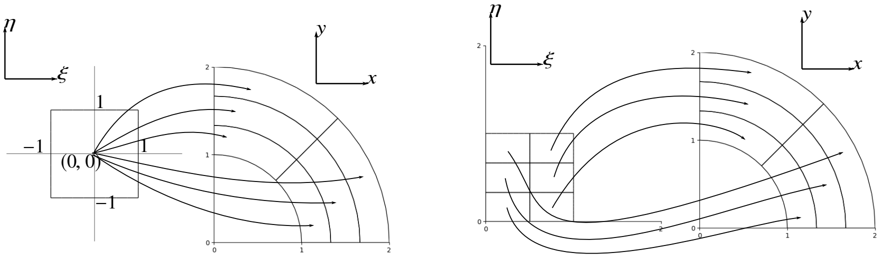
Remark 2.2.3. *For the steady Stokes equations in Chapter 2, we seek the approximate solutions* $(\mathbf{u_h}, p_h) \in \mathbb{V}_h \times \mathbb{Q}_h$ *of the exact solutions* $(\boldsymbol{u}, p) \in \mathbb{V} \times \mathbb{Q}$*. If the spaces are the Sobolev space* $\mathbb{V} = H^1$ *and the space of all integrable functions* $\mathbb{Q} = L^2$ *respectively then if the exact solutions* $(\mathbf{u}, p)$ *are smooth enough and the finite element solutions* $(\mathbf{u_h}, p_h)$ *are polynomials of degree* k *and* $k - 1$ *respectively, then*

$$|\mathbf{u} - \mathbf{u_h}|_{H^1} + \|p - p_h\|_{L^2} \propto O(h^k).$$

2.2.4 Parent element, index space and physical space

This section will just give an illustrative picture of how isogeometric analysis is used to solve a physical problem. This is done to better understand how to work with a parametrized physical problem where the geometry is parameter-dependent. We still assume to be in the parameter-independent domain as Section 2.2.5 deals with the parameter-dependent case. *NURBS* (non-uniform rational B-splines) are the most widely used computational geometry technologies in isogeometric analysis. NURBS may consist of several patches, but as this section is not meant as an introduction on NURBS, we will only consider a single patch as this is sufficient for the work done in this thesis. For an in-depth reading on Isogemetric analysis and NURBS we refer the reader to [22].

NURBS are, as the name might suggest, built from B-splines, which we introduce in Section 2.3. In classical finite element analysis we partition the physical space into elements creating a partitioning, let us say $\mathcal{T}_h$. Finite element methods always consist of an integration step and in classical FEM the numerical integration is performed on a reference element. The reference element is mapped into a single element in the partitioning $\mathcal{T}_h$, meaning that each element has its own mapping, see Figure 2.3a. In isogeometric analysis *the B-spline parameter space* is partitioned into elements such that each element in the



(a) Classical finite element analysis. The reference element is local to each individual element in the physical space. Each element has its own mapping from the reference element.

(b) Isogeometric analysis. The parameter space is local to each patch (the entire physical space in our case as we consider just one patch). A single map maps all elements in the parameter space to the corresponding element in the physical space.

Figure 2.3: Mapping from parameter space/reference element to the physical space for classical finite element analysis and Isogeometric analysis.

physical space are images of the corresponding element in the parameter space, see Figure 2.3b. As in Figure 2.3b we name the coordinates in the parameter space ξ and η and let $\tilde{\Omega}^e$ be an element in the parameter space $\tilde{\Omega}$. Let us further denote by x and y the coordinates of the physical space Ω and let Ω^e be the element. Then we define the mapping from the parameter space to the physical space as $\Phi : \tilde{\Omega} \rightarrow \Omega$. The numerical integration for the isogeometric based finite element solver is done by a Gaussian quadrature rule on a *parent element*, hence we denote by $\hat{\xi}$ and $\hat{\eta}$ the coordinates of the parent element $\hat{\Omega}^e$.

Further we introduce an affine mapping $\Theta : \hat{\Omega}^e \rightarrow \tilde{\Omega}^e$ from the parent element to the parameter element. To perform the integration on the parent element $\hat{\Omega}^e$, we first need to pullback the problem from the physical element Ω^e to the parameter element $\tilde{\Omega}^e$ through the inverse mapping Φ^{-1} and further pullback the problem onto the parent element $\hat{\Omega}^e$ through the inverse mapping Θ^{-1} as seen in Figure 2.4. We will however not study these two mappings further in detail, only a third mapping introduced in Section 2.2.5. See Section 2.2.5 and 2.2.6 for more details on mapping and change of variable formulas.

2.2.5 Mapping to the parameter independent domain

In Section 2.2.4 we saw some idea on how to solve a problem set on a parameter-independent domain Ω. In the following we will focus on 2D problems as introduced in e.g. [37, 49]. Let $\tilde{\Omega}(\boldsymbol{\mu}) \in \mathbb{R}^2$ be the parameter-dependent physical domain obtained as a mapping $\mathcal{F}(\boldsymbol{\mu})$ from the parameter-independent domain Ω. We notify the reader that the parameter-dependent space $\tilde{\Omega}(\boldsymbol{\mu})$ introduced here, is not the same as the parameter space $\tilde{\Omega}$ introduced in the previous section. From here on the tilde notation is used for the parameter-dependent physical space. For a given parameter-vector $\boldsymbol{\mu} \in \mathbb{P}$ we define the parametric map as

$$\mathcal{F}(\boldsymbol{\mu}) : \Omega \times \mathbb{P} \rightarrow \mathbb{R}^2,$$

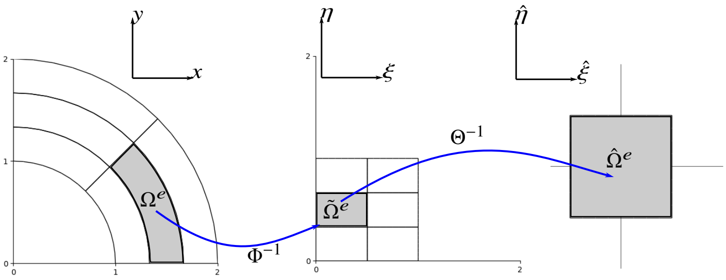


Figure 2.4: One element at the time is pulled back from the physical space to the parent element through the parameter space. The numerical integration is done on the parent element.

and hence we obtain the parameter-dependent domain as

$$\tilde{\Omega}(\boldsymbol{\mu}) = \mathcal{F}(\Omega; \boldsymbol{\mu}) \quad \forall \boldsymbol{\mu} \in \mathbb{P}. \tag{2.20}$$

Note that we here write $\mathcal{F}(\boldsymbol{\mu})$, but it should be clear for the reader that this geometric map in fact only depends on the geometric parameters $\boldsymbol{\mu_g}$. Figure 2.5 shows an example of a parametric map $\mathcal{F}(\boldsymbol{\mu})$ stretching and rotating the domain Ω.

Let us denote the original problem set on the original domain $\tilde{\Omega}(\boldsymbol{\mu})$ in the form

$$\tilde{P} = \tilde{P}(\tilde{\Omega}(\boldsymbol{\mu}); \boldsymbol{\mu}).$$

Here we use the tilde notation to denote the parameter-dependent problem yielding the coordinates $\tilde{\mathbf{x}}$ for the parameter-dependent space, whereas we denote the coordinates of the parameter-independent space by $\mathbf{x}$. In Section 2.2.4 we saw how to solve the problem once it is set on a parameter-independent domain. Hence our goal is to pull the problem $\tilde{P}$, set on $\tilde{\Omega}$, back to the parameter-independent domain Ω through the inverse mapping $\mathcal{F}^{-1}(\boldsymbol{\mu})$ yielding

$$P = P(\Omega; \boldsymbol{\mu}),$$

but to do so, we need the proper tools.

2.2.6 Formulas for change of variable

By means of the inverse mapping $\boldsymbol{\mathcal{F}}^{-1}(\cdot; \boldsymbol{\mu})$ we want to pull the problem set on the parameter-dependent domain $\tilde{\Omega}(\boldsymbol{\mu})$ back to the parameter-independent domain Ω. In order for us to do so, here we give the definition of the Jacobian matrix $\mathbb{J}_{\boldsymbol{\mathcal{F}}}(\mathbf{x}; \boldsymbol{\mu}) \in \mathbb{R}^{2\times 2}$ of the map $\boldsymbol{\mathcal{F}}(\cdot; \boldsymbol{\mu})$

$$(J_{\boldsymbol{\mathcal{F}}}(\mathbf{x}; \boldsymbol{\mu}))_{ij} = \frac{\partial(\tilde{\mathbf{x}})_i}{\partial(\mathbf{x})_j}(\mathbf{x}) = \frac{\partial(\boldsymbol{\mathcal{F}}(\mathbf{x}; \boldsymbol{\mu}))_i}{\partial(\mathbf{x})_j}(\mathbf{x}).$$

With $|J_{\boldsymbol{\mathcal{F}}}(\mathbf{x}; \boldsymbol{\mu})|$ we denote the determinant of the Jacobian matrix, and we assume it to be different from 0 for any $\boldsymbol{\mu} \in \mathbb{P}$ and $\mathbf{x} \in \Omega$ so that the map $\mathcal{F}(\cdot; \boldsymbol{\mu})$ is well defined. Likewise

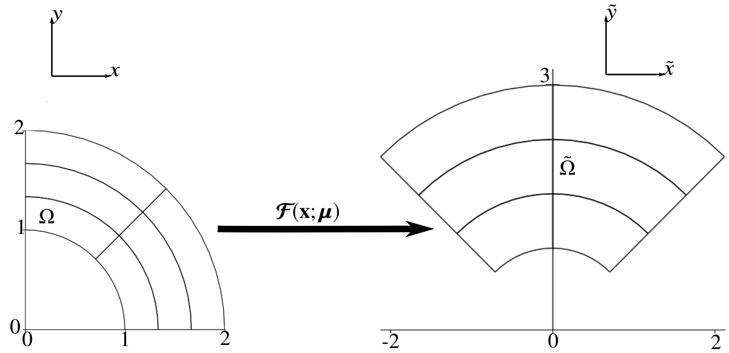


Figure 2.5: The parameter-dependent domain $\tilde{\Omega} = \tilde{\Omega}(\boldsymbol{\mu})$ is obtained as a parametric map of a parameter-independent domain Ω.

the Jacobian matrix of the inverse mapping $\boldsymbol{\mathcal{F}}^{-1}(\cdot;\boldsymbol{\mu})$ is

$$(J_{\mathcal{F}^{-1}}(\tilde{\mathbf{x}};\boldsymbol{\mu}))_{ij} = \frac{\partial(\mathbf{x})_i}{\partial(\tilde{\mathbf{x}})_j}(\tilde{\mathbf{x}}) = \frac{\partial(\boldsymbol{\mathcal{F}}^{-1}(\tilde{\mathbf{x}};\boldsymbol{\mu}))_i}{\partial(\tilde{\mathbf{x}})_j}(\tilde{\mathbf{x}}).$$

With the Jacobian matrix defined for the map, it is now possible to give the change of variable formula for any integrable function $\tilde{f} : \tilde{\Omega}(\boldsymbol{\mu}) \to \mathbb{R}$

$$\int_{\tilde{\Omega}(\boldsymbol{\mu})} \tilde{f}(\tilde{\mathbf{x}})\, \mathrm{d}\tilde{\Omega} = \int_{\Omega} f(\mathbf{x}) |\mathbb{J}_{\mathcal{F}}(\mathbf{x};\boldsymbol{\mu})|\, \mathrm{d}\Omega,$$

where $f = \tilde{f} \circ \boldsymbol{\mathcal{F}}$. The only thing we still need is a tool to evaluate integrals involving derivatives. In 2D the chain rule gives us

$$\frac{\partial \tilde{f}(\tilde{\mathbf{x}})}{\partial \tilde{x}_i} = \frac{\partial f(\mathbf{x})}{\partial x_1}\frac{\partial x_1}{\partial \tilde{x}_i} + \frac{\partial f(\mathbf{x})}{\partial x_2}\frac{\partial x_2}{\partial \tilde{x}_i}$$

and then $\tilde{\nabla}\tilde{f}(\tilde{\mathbf{x}})$ can be written in the parameter-independent variables as

$$\begin{gathered}\tilde{\nabla}\tilde{f}(\tilde{\mathbf{x}}) = \begin{bmatrix} \frac{\partial \tilde{f}(\tilde{\mathbf{x}})}{\partial \tilde{x}_1} \\ \frac{\partial \tilde{f}(\tilde{\mathbf{x}})}{\partial \tilde{x}_2} \end{bmatrix} = \begin{bmatrix} \frac{\partial \tilde{f}(\tilde{\mathbf{x}})}{\partial x_1}\frac{\partial x_1}{\partial \tilde{x}_1} + \frac{\partial \tilde{f}(\tilde{\mathbf{x}})}{\partial x_2}\frac{\partial x_2}{\partial \tilde{x}_1} \\ \frac{\partial \tilde{f}(\tilde{\mathbf{x}})}{\partial x_1}\frac{\partial x_1}{\partial \tilde{x}_2} + \frac{\partial \tilde{f}(\tilde{\mathbf{x}})}{\partial x_2}\frac{\partial x_2}{\partial \tilde{x}_2} \end{bmatrix} = \\ \begin{bmatrix} \frac{\partial x_1}{\partial \tilde{x}_1} & \frac{\partial x_2}{\partial \tilde{x}_1} \\ \frac{\partial x_1}{\partial \tilde{x}_2} & \frac{\partial x_2}{\partial \tilde{x}_2} \end{bmatrix} \begin{bmatrix} \frac{\partial \tilde{f}(\tilde{\mathbf{x}})}{\partial x_1} \\ \frac{\partial \tilde{f}(\tilde{\mathbf{x}})}{\partial x_2} \end{bmatrix} = \left(J_{\mathcal{F}^{-1}}(\mathbf{x};\boldsymbol{\mu})\right)^{\top} \nabla f(\mathbf{x}),\end{gathered} \tag{2.21}$$

where $\tilde{\nabla}$ is the gradient with respect to the parameter-dependent coordinates $\tilde{\mathbf{x}}$, and ∇ is the gradient w.r.t the parameter-independent coordinates $\mathbf{x}$. By the inverse function theorem [55] Quarteroni et al. states in [49] that

$$J_{\mathcal{F}^{-1}}(\tilde{\mathbf{x}};\boldsymbol{\mu}) = (J_{\mathcal{F}}(\mathbf{x};\boldsymbol{\mu}))^{-1},$$

holds if the Jacobian matrix of the map $\mathcal{F}(\cdot;\boldsymbol{\mu})$ is continuous and nonsingular at a point $\mathbf{x}$. Here $\tilde{\mathbf{x}} = \boldsymbol{\mathcal{F}}(\mathbf{x};\boldsymbol{\mu})$. Then the change of variable for derivatives (2.21) is given by

$$\tilde{\nabla}\tilde{f}(\tilde{\mathbf{x}}) = (J_{\mathcal{F}}(\mathbf{x};\boldsymbol{\mu}))^{-\top}\,\nabla f(\mathbf{x}).$$

The extension to the vector case of the formulas given in this section is pretty much straight forward and will therefore not be given. See e.g. [49] on the vector case. The tools given in this section is applied to an example given in Chapter 4.

2.3 B-splines

The Finite Element solver used in this thesis is based on isogeometric analysis, and hence we will briefly discuss the topic of B-splines. We will study the B-spline basis functions in one dimension for simplicity, and we denote the i-th basis function by $B_{i,p}(\xi)$, where p is the polynomial degree of the basis function and $\xi \in \mathbb{R}$ is the input parameter. These basis functions are used to construct *B-spline curves*.

2.3.1 B-spline curves

A B-spline curve in $\mathbb{R}^d$ where $d = 1, 2, 3, \ldots$ is constructed in the same manner as in classical finite element analysis (FEA) by taking a linear combination of the basis functions. The difference is that in classical FEA, the vector-valued nodal coefficients of the basis functions interpolates with the curve, as with B-spline curves the vector-valued coefficients, more commonly known as *control points*, are in general not interpolating the B-spline curve, only at its end-points as seen in Figure 2.6. Given n control points $\mathbf{C_i} \in \mathbb{R}^d, i = 1, 2, \ldots, n$ and the corresponding basis functions of degree p, $B_{i,p}(\xi), i = 1, 2, \ldots, n$ a B-spline, piecewise polynomial curve is given by

$$\mathbf{S}(\xi) = \sum_{i=1}^{n} B_{i,p}(\xi)\mathbf{C_i}.$$

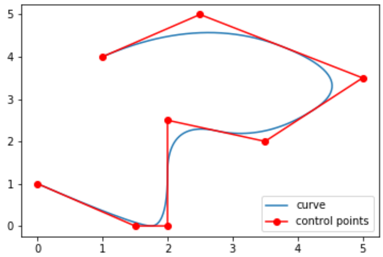


Figure 2.6: A B-spline piecewise quadratic curve in $\mathbb{R}^2$ displayed with its control points only interpolating at the end points.

2.3.2 Knot vector

The B-spline basis functions take input values from the *parameter space*. A *knot vector* in one dimension is a vector of non-decreasing coordinates in the parameter space. Let us denote the knot vector $\boldsymbol{\xi} = [\xi_1, \xi_2, \ldots, \xi_{n+p+1}]$, where $\xi_i \in \mathbb{R}$ is the i-th *knot*, p is the polynomial degree of the B-spline basis functions and n is the number of B-spline basis functions.

Knot vectors are said to be uniform if the knots are equally spaced in the parameter space and non-uniform if the knots are not equally spaced. Some knots may be repeated, i.e. they may have the same value and we denote the multiplicity of knot i by m_i. As we will see, the multiplicity of the knots implies important properties of the B-spline basis functions. If the multiplicity of the first and the last knot is $p+1$, the knot vector is said to be open. Also, the distinct knots partition the parameter space into elements.

Let us explain this explicit. Let an open knot vector $\boldsymbol{\Xi}$ be given on the interval $[a, b]$. Then we have $\xi_1 = a$ and $\xi_{n+p+1} = b$. Also let the knot vector have the same multiplicity for all internal knots, that means $m = m_i,\ \forall i \backslash \{1, n+p+1\}$. As the first an last knot is repeated $p+1$ times and the internal knots are repeated m times, we have that the number of partitions of $[a, b]$ or equivalently the number of knot spans must be the same as number of unique internal knots plus one, which is $n_{ks} = \frac{(n+p+1)-2(p+1)}{m} + 1 = \frac{(n-p-1)}{m} + 1$.

2.3.3 Basis functions and B-spline functions in one dimension

For a given knot vector $\boldsymbol{\xi} = [\xi_1, \ldots, \xi_{n+p+1}]$ the n B-spline basis functions are given by the *cox-de Boor recursion formula* given in e.g. [22]. Starting with piecewise constants for $p = 0$

$$B_{i,0}(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1}, \\ 0 & \text{otherwise.} \end{cases}$$

For $p = 1, 2, 3, \ldots$ the B-spline basis functions are given by

$$B_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} B_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} B_{i+1,p-1}(\xi).$$

$\boldsymbol{\xi} = [\xi_1, \ldots, \xi_{n+p+1}]$ is the knot vector of a total of n B-spline basis functions of degree p. The *spline space* $\mathbb{S}^p_{\boldsymbol{\xi}}$ is the linear space of all linear combinations of these B-spline functions, i.e.

$$\mathbb{S}^p_{\boldsymbol{\xi}} = \left\{ \sum_{i=1}^{n} c_i B_{i,p}(\xi) | c_j \in \mathbb{R} \text{ for } 1 \leq i \leq n \right\}.$$

In [38] it is shown that the B-spline basis functions are linearly independent and hence they form a basis for the space $\mathbb{S}^p_{\boldsymbol{\xi}}$, i.e.

$$\mathbb{S}^p_{\boldsymbol{\xi}} = \text{span}\{\mathrm{B_{1,p}}(\xi), \ldots, \mathrm{B_{n,p}}(\xi)\},$$

and we see that $\dim(\mathbb{S}^{\mathrm{p}}_{\boldsymbol{\xi}}) = \mathrm{n}$. For the purpose of the examples analyzed in this thesis, it is sufficient with uniform open knot vectors. Hence we use the following notation of the spline space

$$\mathbb{S}^p_c = \left\{ B_{i,p}(\xi)|_{[\xi_i, \xi_{i+1}]} \in \mathbb{P}_p \wedge B_{i,p}(\xi)|_{\xi_i} \in C^c, \forall i : \xi_{i+1} > \xi_i \right\}.$$

This space provides us with basis functions that are polynomials of degree p on each element $[\xi_i, \xi_{i+1}]$, $\forall i : \xi_{i+1} > \xi_i$ and they are c times continuously differentiable across the element boundaries, ξ_i, $\forall i : \xi_{i+1} > \xi_i$. The multiplicity of knot i is then given by the relation

$$c = p - m_i.$$

Note that m_i can at most be $p + 1$ as the lowest order of continuity allowed is discontinuity C^{-1}.

2.3.4 Basis functions and B-spline spaces in two dimension

This section shortly addresses the extension of the above Section 2.3.3 to two dimensions. Let $\Xi = [\xi_1, \ldots, \xi_{n_b+p+1}]$ and $H = [\eta_1, \ldots, \eta_{m_b+q+1}]$ be two knot vectors in the ξ-direction and the η-direction of the parameter space Ω associated with the B-spline basis functions $\{B^1_{i,p}(\xi)\}_{i=1}^{n_b}$ and $\{B^2_{i,q}(\eta)\}_{i=1}^{m_b}$ respectively. p, q are the polynomial degree of the basis functions and n_b, m_b are the number of basis functions in the ξ-direction and η-direction respectively. Then the knot vectors partition the parameter space Ω into a rectangular mesh, let us call it $\mathcal{R}_h$ and let h be the associated mesh size. Then we define the tensor product B-spline basis functions associated with this mesh as

$$B_{ij,pq}(\xi, \eta) = B^1_{i,p}(\xi) \bigotimes B^2_{i,q}(\eta) \qquad i = 1, \ldots, n_b \; j = 1, \ldots, m_b,$$

and the spline space spanned by these basis functions as

$$\mathbb{S}^{p,q}_{c_1,c_2} = \mathrm{span}\{B_{\mathrm{ij,pq}}(\xi, \eta)\}^{\mathrm{n_b,m_b}}_{\mathrm{i=1,j=1}},$$

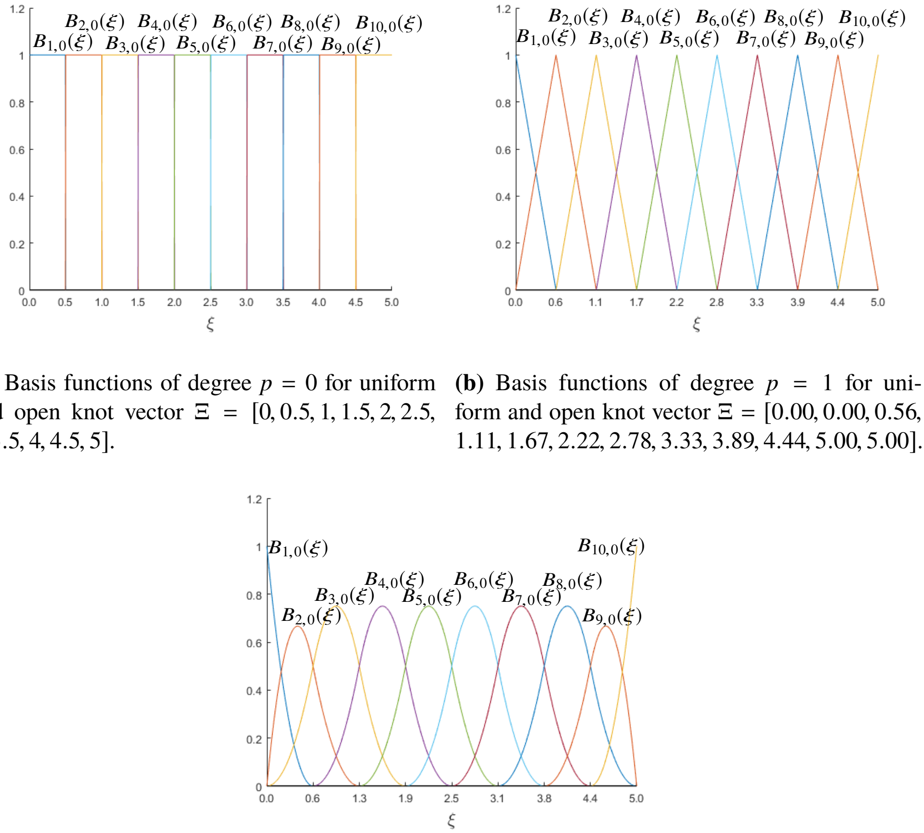
where c_1 gives the continuity across the element boundaries in the ξ-direction of the $B^1_{i,p}$ basis functions and c_2 gives the continuity across the element boundaries in the η-direction of the $B^2_{i,q}$ basis functions. For further details see [17].

Remark 2.3.1. *As mentioned earlier, the finite element solver built for this thesis uses Taylor-Hood elements $\mathbb{Q}_{k+1} - \mathbb{Q}_k$ for $k \geq 1$. This corresponds to the choice $\mathbb{V}_h = \{\boldsymbol{v} : \boldsymbol{v} \circ \Phi \in \mathbb{S}^{k+1,k+1}_{0,0} \times \mathbb{S}^{k+1,k+1}_{0,0}\}$ for the velocity space and $\mathbb{Q}_h = \{q : q \circ \Phi \in \mathbb{S}^{k,k}_{0,0}\}$ for the pressure space in the Stokes equation, [17]. Note that this is the spaces of functions which are spline functions on the parameter space, where the mapping Φ from the parameter to the physical space is the mapping from Figure 2.4. See Chapter 3 for further details on the velocity and pressure space.*

2.3.5 Properties B-spline basis functions

As seen in Figure 2.7a and 2.7b B-spline basis functions for $p = 0$ and $p = 1$ are the same as ordinary FEM basis functions. For quadratic B-spline basis functions seen in Figure 2.7c and higher degree B-spline basis functions they are all identical, but shifted relative to each other. The B-spline basis functions always share support with (including itself) $2p + 1$ basis functions and each function has support on $p + 1$ knot spans.

For uniform and open knot vectors, B-spline basis functions are not interpolant at internal knots, only at end knots in 1D and at corner knots in higher dimensions. In



(a) Basis functions of degree $p = 0$ for uniform and open knot vector $\Xi = [0, 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5]$.

(b) Basis functions of degree $p = 1$ for uniform and open knot vector $\Xi = [0.00, 0.00, 0.56, 1.11, 1.67, 2.22, 2.78, 3.33, 3.89, 4.44, 5.00, 5.00]$.

(c) Basis functions of degree $p = 2$ for uniform and open knot vector $\Xi = [0.00, 0.00, 0.00, 0.63, 1.25, 1.88, 2.50, 3.13, 3.75, 4.38, 5.00, 5.00, 5.00]$.

Figure 2.7: B-spline basis functions of degree $p = 1, 2, 3$ on the interval $[0, 5]$.

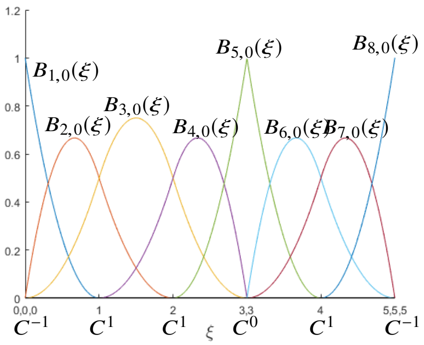


Figure 2.8: Basis functions of degree 2 for non-uniform and open knot vector $\Xi = [0, 0, 0, 1, 2, 3, 3, 4, 5, 5, 5]$ displayed with order continuity across element boundaries, i.e. the knots.

general, because the B-spline basis functions of degree p have support on $p + 1$ knot spans, repeated knots will decrease the continuous derivatives of basis functions over the knots. In fact basis functions of degree p have $p - m_i$ continuous derivatives across knot ξ_i, where m_i is the multiplicity of ξ_i. When the multiplicity of the i-th knot is exactly p, i.e. $m_i = p$, the basis is interpolant at that point as seen in Figure 2.8. This differs internal knots from internal nodes in ordinary FEM, where internal nodes are interpolation points of the basis functions. Hence for a uniform and open knot vector the basis functions are interpolating at the end points $[\xi_1, \xi_{n+p+1}]$ and they have $p - 1$ continuous derivatives across the knots as seen for knot 1, 2 and 4 in Figure 2.8.

2.4 Sobolev space

As the finite element method is written in the language of functional analysis, we need to provide the right tools to solve the desired problem. Therefore we familiarize the reader with the notion of the *Sobolev spaces*. In Section 2.1.1 we introduced the notion of the weak form. To solve problems cast in the framework of the weak formulation 2.1, we need the solution (x, p) and the test functions w and q to lie in certain spaces for the weak formulation to hold.

Let Ω be an open subset of $\mathbb{R}^n$. Denote the boundary of Ω by $\partial\Omega$ and let it be piecewise smooth. The Sobolev spaces are built on the function space $L^2(\Omega)$, that is the space of all square-integrable functions on Ω, [14, 48]. The $L^2(\Omega)$ space is defined as

$$L^2(\Omega) = \{f : \Omega \to \mathbb{R} \ : \ \int_\Omega (f(\mathbf{x}))^2 \, \mathrm{d}\Omega < \infty\}.$$

We need the Sobolev spaces $H^1(\Omega)$ and $H^1_0(\Omega)$ which can be found in e.g. [1, 48] and is defined as

$$H^1(\Omega) = \{f \in L^2(\Omega) : Df \in L^2(\Omega)\},$$

and

$$H^1_0(\Omega) = \{f \in H^1(\Omega) : f|_{\partial\Omega} = 0\}.$$

The former is the spaces of all functions f such that f and its first weak derivative Df are square integrable on Ω and the latter is similar. We also require f to be zero at the boundary. $H_0^1(\Omega)$ is a space used when the whole boundary of Ω, that is $\partial\Omega$ is a homogeneous Dirichlet boundary. If just a part of the boundary is Dirichlet, with a function value g on the boundary, we will use the space

$$H^1_{\Gamma_D}(\Omega) = \{f \in H^1(\Omega) : f|_{\Gamma_D} = g\},$$

where Γ_D is the part of the boundary $\partial\Omega$ being Dirichlet.

The Sobolev spaces are Hilbert spaces with respect to the following inner product

$$(f, g)_{H^1(\Omega)} = \int_\Omega fg \mathrm{d}\Omega + \int_\Omega \nabla f \nabla g \ \mathrm{d}\Omega,$$

which induces the H^1-norm

$$\|f\|_{H^1(\Omega)} = \sqrt{(f, f)_{H^1(\Omega)}}.$$

We also define the H^1-seminorm

$$|f|_{H^1(\Omega)} = \|\nabla f\|_{L^2(\Omega)}$$

2.5 Reduced Basis Methods

In short, reduced basis (RB) methods are used to approximate any member of the solution manifold $\mathbb{M}$ with a low number of basis functions, known as *reduced basis functions*. The RB method has two stages, first a costly offline stage constructing RB functions based on the discrete solution manifold $\mathbb{M}_h$ in some way. The second stage is an online stage for (hopefully) efficient computations of the *RB solution* $(x_N(\boldsymbol{\mu}), p_N(\boldsymbol{\mu}))$.

We assume that for a given $\boldsymbol{\mu}$ there exist a high-fidelity solution $(x_h(\boldsymbol{\mu}), p_h(\boldsymbol{\mu}))$ such that we can approximate the exact solution $(x(\boldsymbol{\mu}), p(\boldsymbol{\mu}))$ of problem (2.1.1) with a desired accuracy for some choice of h, i.e. $\|x(\boldsymbol{\mu}) - x_h(\boldsymbol{\mu})\|_{\mathbb{X}} + \|p(\boldsymbol{\mu}) - p_h(\boldsymbol{\mu})\|_{\mathbb{Q}} < \varepsilon_{\text{tol}}$ and for a chosen h we have that ε_{tol} can become arbitrarily small. Then by the triangle inequality we have that

$$\begin{aligned} \|x(\boldsymbol{\mu}) - x_N(\boldsymbol{\mu})\|_{\mathbb{X}} + \|p(\boldsymbol{\mu}) - p_N(\boldsymbol{\mu})\|_{\mathbb{Q}} &\leq \|x(\boldsymbol{\mu}) - x_h(\boldsymbol{\mu})\|_{\mathbb{X}} \\ +\|p(\boldsymbol{\mu}) - p_h(\boldsymbol{\mu})\|_{\mathbb{Q}} &+ \|x_h(\boldsymbol{\mu}) - x_N(\boldsymbol{\mu})\|_{\mathbb{X}} + \|p_h(\boldsymbol{\mu}) - p_N(\boldsymbol{\mu})\|_{\mathbb{Q}} \\ &\leq \varepsilon_{\text{tol}} + \|x_h(\boldsymbol{\mu}) - x_N(\boldsymbol{\mu})\|_{\mathbb{X}} + \|p_h(\boldsymbol{\mu}) - p_N(\boldsymbol{\mu})\|_{\mathbb{Q}}, \end{aligned} \tag{2.22}$$

and hence the accuracy of which the RB solution approximates the exact solution can be controlled by how good the RB solution approximates the high-fidelity solution.

Given the discrete solution manifold $\mathbb{M}_h$ in (2.8), we now hope that any solution $(x_h(\boldsymbol{\mu}), p_h(\boldsymbol{\mu}))$ can be well approximated by linearly combining only a few elements of $\mathbb{M}_h$, i.e. we want to represent the discrete solution manifold with a few number of chosen basis functions. As pointed out by Hesthaven, choosing these basis functions, hereby called the

RB functions, appropriately we can represent the solution manifold with a small error, see [34].

The main idea of creating this reduced basis is to choose a set of n_s selected parameters

$$\{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{n_s}\} \subset \mathbb{Q}$$

creating a set of high-fidelity solutions that we call *snapshots*

$$\{(x_h(\boldsymbol{\mu}_1), p_h(\boldsymbol{\mu}_1)), \ldots, (x_h(\boldsymbol{\mu}_{n_s}), p_h(\boldsymbol{\mu}_{n_s}))\}$$

From these we want to generate a set of N_u RB functions for the $x_h(\boldsymbol{\mu})$ component of the solution and N_p RB functions for the $p_h(\boldsymbol{\mu})$ component of the solution yielding

$$\{\phi_1^{rb}, \ldots, \phi_{N_u}^{rb}\}, \quad \{\psi_1^{rb}, \ldots, \psi_{N_p}^{rb}\},$$

where N_u and N_p not necessarily are equal. These RB functions spans the *reduced basis spaces* $\mathbb{X}_N$ and $\mathbb{Q}_N$, that is the reduced basis approximation spaces to the high-fidelity spaces $\mathbb{X}_h$ and $\mathbb{Q}_h$ such that

$$\mathbb{X}_N = \text{span}\{\phi_1^{rb}, \ldots, \phi_{N_u}^{rb}\} \subset \mathbb{X}_h, \qquad \mathbb{Q}_N = \text{span}\{\psi_1^{rb}, \ldots, \psi_{N_p}^{rb}\} \subset \mathbb{Q}_h.$$

It is worth noting that these RB functions are in general not solutions of the high-fidelity problem (2.1.1), but hopefully they will approximate the set of snapshots i.e.

$$\mathbb{X}_N \approx \text{span}\{x_h(\boldsymbol{\mu}_1), \ldots, x_h(\boldsymbol{\mu}_{n_s})\} \qquad \mathbb{Q}_N \approx \text{span}\{p_h(\boldsymbol{\mu}_1), \ldots, p_h(\boldsymbol{\mu}_{n_s})\},$$

Where we have assumed that the set of snapshots are representative for the solution manifold $\mathbb{M}_h$. Further we assume low dimensionality of $\mathbb{M}_h$, which implies that $N_u \ll N_h$ and $N_p \ll M_h$. In Section 2.7 we will discuss how to construct N basis functions out of a set of n_s snapshots using the proper orthogonal decomposition technique.

2.5.1 Galerkin reduced basis approximation

For $\boldsymbol{\mu} \in \mathbb{P}$ we formulate the Galerkin RB approximation of the weak formulation (2.1) as: find $(x_N(\boldsymbol{\mu}), p_N(\boldsymbol{\mu})) \in \mathbb{X}_N \times \mathbb{Q}_N$ s.t.

$$\begin{aligned} d(x_N(\boldsymbol{\mu}), w_N(\boldsymbol{\mu}); \boldsymbol{\mu}) + b(w_N, p_N(\boldsymbol{\mu}); \boldsymbol{\mu}) &= f_1(w_N; \boldsymbol{\mu}) \qquad \forall w_N \in \mathbb{X}_N(\boldsymbol{\mu}), \\ b(x_N(\boldsymbol{\mu}), q_N; \boldsymbol{\mu}) &= f_2(q_N; \boldsymbol{\mu}) \qquad \forall q_N \in \mathbb{Q}_N(\boldsymbol{\mu}), \end{aligned} \tag{2.23}$$

As the *reduced basis solutions* $(x_N(\boldsymbol{\mu}), p_N(\boldsymbol{\mu}))$ are functions of $\mathbb{X}_N \times \mathbb{Q}_N$, it can be written as a linear combination of the RB functions spanning $\mathbb{X}_N \times \mathbb{Q}_N$,

$$x_N(\boldsymbol{\mu}) = \sum_{i=1}^{N_u} \phi_i^{rb} \, (\mathbf{x_N}(\boldsymbol{\mu}))_i, \qquad p_N(\boldsymbol{\mu}) = \sum_{i=j}^{N_p} \psi_j^{rb} \, (\mathbf{p_N}(\boldsymbol{\mu}))_j \tag{2.24}$$

where $\mathbf{x_N}(\boldsymbol{\mu}) = \left[(\mathbf{x_N}(\boldsymbol{\mu}))_1, \ldots, (\mathbf{x_N}(\boldsymbol{\mu}))_{N_u}\right]^\top \in \mathbb{R}^{N_u}$ and $\mathbf{p_N}(\boldsymbol{\mu}) = \left[(\mathbf{p_N}(\boldsymbol{\mu}))_1, \ldots, (\mathbf{p_N}(\boldsymbol{\mu}))_{N_p}\right]^\top \in \mathbb{R}^{N_p}$ is the RB coefficients of $x_N(\boldsymbol{\mu})$ and $p_N(\boldsymbol{\mu})$ respectively. As the weak formulation

(2.23) holds for all $w_N \in \mathbb{X}_N(\boldsymbol{\mu})$ and $q_N \in \mathbb{Q}_N(\boldsymbol{\mu})$ we follow the same line of thought as in (2.10) and (2.11). We then write the test functions $w_N \in \mathbb{X}_N$ as $w_N = \phi_j^{rb}$ and $q_N \in \mathbb{Q}_N$ as $q_N = \psi_l^{rb}$ and inserting this together with equation (2.24) into equation (2.23) we see that as in equation (2.12) we get a linear system

$$\underbrace{\begin{bmatrix} D_N(\boldsymbol{\mu}) & B_N^\top(\boldsymbol{\mu}) \\ B_N(\boldsymbol{\mu}) & 0 \end{bmatrix}}_{K_N(\boldsymbol{\mu})} \underbrace{\begin{bmatrix} \mathbf{x_N}(\boldsymbol{\mu}) \\ \mathbf{p_N}(\boldsymbol{\mu}) \end{bmatrix}}_{\mathbf{U_N}(\boldsymbol{\mu})} = \underbrace{\begin{bmatrix} \mathbf{f_{1N}}(\boldsymbol{\mu}) \\ \mathbf{f_{2N}}(\boldsymbol{\mu}) \end{bmatrix}}_{\mathbf{F_N}(\boldsymbol{\mu})}, \tag{2.25}$$

As the RB functions $(\phi_j^{rb}, \psi_l^{rb})$ belong to $\mathbb{X}_N \times \mathbb{Q}_N \subset \mathbb{X}_h \times \mathbb{Q}_h$ for $j = 1, \dots, N_u$ and $l = 1, \dots, N_p$ we can write them as linear combinations of the original basis functions

$$\phi_j^{rb} = \sum_{i=1}^{N_h} (V_u)_{ij}\phi_i, \qquad \psi_l^{rb} = \sum_{i=1}^{M_h} (V_p)_{il}\psi_i \quad j = 1, \dots, N_u \; l = 1, \dots, N_p. \tag{2.26}$$

Then from equation (2.26) we define the two *transformation matrices* $V_u \in \mathbb{R}^{N_h \times N_u}$ and $V_p \in \mathbb{R}^{M_h \times N_p}$ such that column vectors $(\mathbf{V_u})_j$ and $(\mathbf{V_p})_l$ of the transformation matrices $V_u = [(\mathbf{V_u})_1 | \dots |(\mathbf{V_u})_{N_u}]$ and $V_p = [(\mathbf{V_p})_1 | \dots |(\mathbf{V_p})_{N_p}]$ respectively holds the coefficients for representing the $j-th$ RB function ϕ_j^{rb} in terms of the $\mathbb{X}_h$-basis $\{\phi_i\}_{i=1}^{N_h}$ and the $l-th$ RB function ψ_l^{rb} in terms of the $\mathbb{Q}_h$-basis $\{\psi_i\}_{i=1}^{M_h}$. From this we can compute the RB matrices and vectors from the original ones. We compute the RB matrix $D_N(\boldsymbol{\mu}) \in \mathbb{R}^{N_u \times N_u}$ as

$$\begin{gathered} (D_N(\boldsymbol{\mu}))_{ij} = d(\phi_j^{rb}, \phi_i^{rb}; \boldsymbol{\mu}) = \sum_{k=1}^{N_h} \sum_{l=1}^{N_h} (V_u)_{kj} d(\phi_k, \phi_l; \boldsymbol{\mu})(V_u)_{li} \quad \text{for } i, j = 1 \dots, N_u \\ D_N(\boldsymbol{\mu}) = V_u^\top D_h(\boldsymbol{\mu}) V_u \end{gathered} \tag{2.27}$$

and $B_N(\boldsymbol{\mu}) \in \mathbb{R}^{N_p \times N_u}$ as

$$\begin{gathered} (B_N(\boldsymbol{\mu}))_{lj} = b(\phi_j^{rb}, \psi_l^{rb}; \boldsymbol{\mu}) = \sum_{k=1}^{N_h} \sum_{i=1}^{M_h} (V_u)_{kj} b(\phi_k, \psi_i; \boldsymbol{\mu})(V_p)_{il} \quad \text{for } l = 1, \dots, N_p \; j = 1, \dots, N_u \\ B_N(\boldsymbol{\mu}) = V_p^\top B_h(\boldsymbol{\mu}) V_u \end{gathered} \tag{2.28}$$

In the same manner we compute the RB right hand side vector $\mathbf{f_{1N}}(\boldsymbol{\mu}) \in \mathbb{R}^{N_u}$

$$\begin{gathered} (\mathbf{f_{1N}}(\boldsymbol{\mu}))_i = f_1(\phi_i^{rb}; \boldsymbol{\mu}) = \sum_{k=1}^{N_h} (V_u)_{ki} f_1(\phi_i; \boldsymbol{\mu}) \quad \text{for } i = 1, \dots, N_u \\ \mathbf{f_{1N}}(\boldsymbol{\mu}) = V_u^\top \mathbf{f_{1h}}(\boldsymbol{\mu}), \end{gathered} \tag{2.29}$$

and the RB right hand side vector $\mathbf{f_{2N}}(\boldsymbol{\mu}) \in \mathbb{R}^{N_p}$

$$\begin{gathered} (\mathbf{f_{2N}}(\boldsymbol{\mu}))_i = f_2(\psi_i^{rb}; \boldsymbol{\mu}) = \sum_{k=1}^{M_h} (V_p)_{ki} f_2(\phi_i; \boldsymbol{\mu}) \quad \text{for } i = 1, \dots, N_p \\ \mathbf{f_{2N}}(\boldsymbol{\mu}) = V_p^\top \mathbf{f_{2h}}(\boldsymbol{\mu}) \end{gathered} \tag{2.30}$$

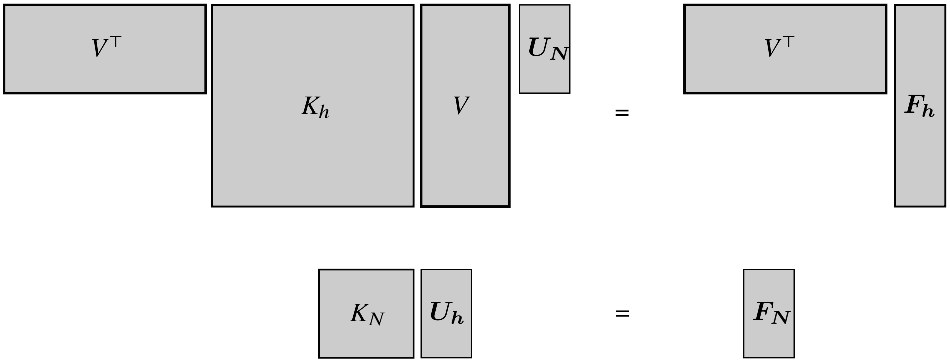


Figure 2.9: The linear system in equation (2.25) with $K_h = K_h(\boldsymbol{\mu}) \in \mathbb{R}^{(N_h+M_h)\times(N_h+M_h)}$, $\mathbf{U_N} = \mathbf{U_N}(\boldsymbol{\mu}) \in \mathbb{R}^{(N_u+N_p)}$, $\mathbf{F_h} = \mathbf{F_h}(\boldsymbol{\mu}) \in \mathbb{R}^{(N_h+M_h)}$, $V \in \mathbb{R}^{(N_h+M_h)\times(N_u+N_p)}$, $K_N = K_N(\boldsymbol{\mu}) \in \mathbb{R}^{(N_u+N_p)\times(N_u+N_p)}$, and $\mathbf{F_N} = \mathbf{F_N}(\boldsymbol{\mu}) \in \mathbb{R}^{(N_u+N_p)}$.

If we make the block matrix $V \in \mathbb{R}^{(N_h+M_h)\times(N_u+N_p)}$ with V_u and V_p on the diagonal, i.e.

$$V = \begin{bmatrix} V_u & 0 \\ 0 & V_p \end{bmatrix}$$

it is possible to compute the matrix $K_N(\boldsymbol{\mu})$ and the vector $\mathbf{F_N}(\boldsymbol{\mu})$ from equation (2.25) using the corresponding matrix and vector from the linear system (2.12) as

$$\begin{aligned} V^\top K_h(\boldsymbol{\mu})V &= \begin{bmatrix} V_u^\top & 0 \\ 0 & V_p^\top \end{bmatrix} \begin{bmatrix} D_h(\boldsymbol{\mu}) & B_h^\top(\boldsymbol{\mu}) \\ B_h(\boldsymbol{\mu}) & 0 \end{bmatrix} \begin{bmatrix} V_u & 0 \\ 0 & V_p \end{bmatrix} = \begin{bmatrix} V_u^\top D_h(\boldsymbol{\mu})V_u & V_u^\top B_h^\top(\boldsymbol{\mu})V_p \\ V_p^\top B_h(\boldsymbol{\mu})V_u & 0 \end{bmatrix} \\ &= \begin{bmatrix} D_N(\boldsymbol{\mu}) & B_N^\top(\boldsymbol{\mu}) \\ B_N(\boldsymbol{\mu}) & 0 \end{bmatrix} = K_N(\boldsymbol{\mu}) \end{aligned}$$

and

$$V^\top \mathbf{F_h}(\boldsymbol{\mu}) = \begin{bmatrix} V_u^\top & 0 \\ 0 & V_p^\top \end{bmatrix} \begin{bmatrix} \mathbf{f_{1h}}(\boldsymbol{\mu}) \\ \mathbf{f_{2h}}(\boldsymbol{\mu}) \end{bmatrix} = \begin{bmatrix} V_u^\top \mathbf{f_{1h}}(\boldsymbol{\mu}) \\ V_p^\top \mathbf{f_{2h}}(\boldsymbol{\mu}) \end{bmatrix} = \begin{bmatrix} \mathbf{f_{1N}}(\boldsymbol{\mu}) \\ \mathbf{f_{2N}}(\boldsymbol{\mu}) \end{bmatrix} = \mathbf{F_N}(\boldsymbol{\mu})$$

The linear system of size $N_u + N_p$ in equation (2.25) is shown in Figure 2.9 to illustrate the dimension reduction compared to the linear system in Figure 2.1.

2.5.2 Full order representation of the reduced basis approximation vector

After solving the RB system in equation (2.25), we obtain the N_u-dimensional RB vector $\mathbf{x_N}(\mu)$ and the N_p-dimensional RB vector $\mathbf{p_N}(\mu)$. Hence we can compute the RB solution $(x_N(\boldsymbol{\mu}), p_N(\boldsymbol{\mu}))$ given in equation (2.24). This is the approximate solution to the high-fidelity function in equation (2.9). For error computations it is preferable if we know the N_h-dimensional vector holding the degrees of freedom (DoF) of $x_N(\boldsymbol{\mu})$ corresponding to $\mathbf{x_h}(\boldsymbol{\mu}) \in \mathbb{R}^{N_h}$ and likewise the M_h-dimensional vector holding the DoF of $p_h(\boldsymbol{\mu})$. To obtain these vectors we insert equation (2.26) into equation (2.24) which gives us

$$x_N(\boldsymbol{\mu}) = \sum_{i=1}^{N_u} \phi_i^{rb} \left(\mathbf{x_N}(\boldsymbol{\mu})\right)_i = \sum_{i=1}^{N_u} \sum_{j=1}^{N_h} (V_u)_{ji} \phi_j \left(\mathbf{x_N}(\boldsymbol{\mu})\right)_i = \sum_{j=1}^{N_h} \phi_j \left(\sum_{i=1}^{N_u} (V_u)_{ji} \left(\mathbf{x_N}(\boldsymbol{\mu})\right)_i \right). \tag{2.31}$$

and

$$p_N(\boldsymbol{\mu}) = \sum_{i=1}^{N_p} \psi_i^{rb} \left(\mathbf{p_N}(\boldsymbol{\mu})\right)_i = \sum_{i=1}^{N_p} \sum_{j=1}^{M_h} (V_p)_{ji} \psi_j \left(\mathbf{p_N}(\boldsymbol{\mu})\right)_i = \sum_{j=1}^{M_h} \psi_j \left(\sum_{i=1}^{N_p} (V_p)_{ji} \left(\mathbf{p_N}(\boldsymbol{\mu})\right)_i \right). \tag{2.32}$$

From equation (2.31) we conclude that $\left(\sum_{i=1}^{N_u} (V_u)_{ji} \left(\mathbf{x_N}(\boldsymbol{\mu})\right)_i\right)$ corresponds to the coefficients $(\mathbf{x_h}(\boldsymbol{\mu}))_j$ in the first expression in equation (2.9) and hence

$$V_u \mathbf{x_N}(\boldsymbol{\mu}) \in \mathbb{R}^{N_h},$$

corresponds to the coefficient vector $\mathbf{x_h}(\boldsymbol{\mu})$. From equation (2.32) the conclusion is similar, $\left(\sum_{i=1}^{N_p} (V_p)_{ji} \left(\mathbf{p_N}(\boldsymbol{\mu})\right)_i\right)$ corresponds to the coefficients $(\mathbf{p_h}(\boldsymbol{\mu}))_j$ in the second expression in equation in (2.9) and hence

$$V_p \mathbf{p_N}(\boldsymbol{\mu}) \in \mathbb{R}^{M_h},$$

corresponds to the coefficient vector $\mathbf{p_h}(\boldsymbol{\mu})$.

To simplify notation in the following we use the compact representation of the solution of the Galerkin problem (2.7) namely $\mathbf{U_h}(\boldsymbol{\mu}) := [\mathbf{x_h}(\boldsymbol{\mu}), \mathbf{p_h}(\boldsymbol{\mu})]^\top$ and the reduced solution is $\mathbf{U_N}(\boldsymbol{\mu}) := [\mathbf{x_N}(\boldsymbol{\mu}), \mathbf{p_N}(\boldsymbol{\mu})]^\top$.

The transformation matrix V is constructed by a POD method introduced in 2.7 making the columns of V orthogonal. As in [49] we name $\tilde{\mathbf{U}}_\mathbf{h} = \tilde{\mathbf{U}}_\mathbf{h}(\boldsymbol{\mu}) = V\mathbf{U_N}(\boldsymbol{\mu}) \in \mathbb{R}^{N_h+M_h}$ *the full order representation* of $\mathbf{U_N}(\boldsymbol{\mu})$. If we now denote by $\hat{\mathbb{V}}_N$ the subspace of $\mathbb{R}^{N_h+M_h}$ that is spanned by the column vectors of V, i.e. $\hat{\mathbb{V}}_N = \mathrm{span}\{(\mathbf{V})_1, \ldots, (\mathbf{V})_\mathrm{N}\} \subset \mathbb{R}^{\mathrm{N_h+M_h}}$, we know that $V\mathbf{U_N}(\boldsymbol{\mu}) \in \hat{\mathbb{V}}_N$ as in fact multiplying with the matrix V provides an orthogonal projection on the reduced subspace $\hat{\mathbb{V}}_N$. As $\mathbf{U_h}(\boldsymbol{\mu}) \in \mathbb{R}^{N_h+M_h}$ does not belong entirely in $\hat{\mathbb{V}}_N$ we have that $\tilde{\mathbf{U}}_\mathbf{h}$ solves a high-fidelity problem, different from problem (2.12), that we derive from problem (2.25)

$$\begin{aligned} K_N(\boldsymbol{\mu})\mathbf{U_N}(\boldsymbol{\mu}) &= \mathbf{F_N}(\boldsymbol{\mu}), \\ V^\top K_h(\boldsymbol{\mu}) V \mathbf{U_N}(\boldsymbol{\mu}) &= V^\top \mathbf{F_h}(\boldsymbol{\mu}), \\ V^\top K_h(\boldsymbol{\mu}) V V^\top V \mathbf{U_N}(\boldsymbol{\mu}) &= V^\top \mathbf{F_h}(\boldsymbol{\mu}), \\ V V^\top K_h(\boldsymbol{\mu}) V V^\top \tilde{\mathbf{U}}_\mathbf{h} &= V V^\top \mathbf{F_h}(\boldsymbol{\mu}). \end{aligned} \tag{2.33}$$

In the third line we have assumed that $V^\top V = I_N$, i.e. the columns of V are orthonormal. In the fourth line we left multiply with V. We will come back to the assumption that $V^\top V = I_N$ when constructing the transformation matrix V in Section 2.7.

2.5.3 Error computations

In the introduction of Section 2.5 we discussed that the accuracy of the RB solution is controlled by the second term on the right hand side of equation (2.22) and hence we write the error we want to study as

$$e_h^u(\boldsymbol{\mu}) = x_h(\boldsymbol{\mu}) - x_N(\boldsymbol{\mu}), \qquad e_h^p(\boldsymbol{\mu}) = p_h(\boldsymbol{\mu}) - p_N(\boldsymbol{\mu}), \tag{2.34}$$

and from Section 2.5.2 we get that the vector representation of the error is the difference between the high-fidelity approximation vector $\mathbf{U_h}(\boldsymbol{\mu})$ and the full order representation $\tilde{\mathbf{U}}_\mathbf{h}(\boldsymbol{\mu})$, split in the two components of the solution we have

$$\mathbf{e_h^u}(\boldsymbol{\mu}) = \mathbf{x_h}(\boldsymbol{\mu}) - V_u\mathbf{x_N}(\boldsymbol{\mu}), \qquad \mathbf{e_h^p}(\boldsymbol{\mu}) = \mathbf{p_h}(\boldsymbol{\mu}) - V_p\mathbf{p_N}(\boldsymbol{\mu}). \tag{2.35}$$

In [49] a geometric interpretation of the error in equation (2.35) is introduced. We exploit the fact that the transformation matrix introduced in equation (2.26) allows us to define an orthogonal projection on the reduced subspace $\hat{\mathbb{V}}_N$. We will come back to this projection in Section 2.7, and for now we denote it by $\mathbb{P}$. Then we can write the error in equation (2.35) as

$$\mathbf{e_h}(\boldsymbol{\mu}) := [\mathbf{e_h^u}(\boldsymbol{\mu}), \mathbf{e_h^p}(\boldsymbol{\mu})]^\top = \mathbf{U_h}(\boldsymbol{\mu}) - V\mathbf{U_N}(\boldsymbol{\mu}) = (\mathbf{U_h}(\boldsymbol{\mu}) - \mathbb{P}\mathbf{U_h}(\boldsymbol{\mu})) + (\mathbb{P}\mathbf{U_h}(\boldsymbol{\mu}) - V\mathbf{U_N}(\boldsymbol{\mu})), \tag{2.36}$$

where the first parentheses on the right hand side is due to the fact that the high-fidelity approximation vector $\mathbf{U_h}(\boldsymbol{\mu})$ does not strictly belong to the reduced subspace $\hat{\mathbb{V}}_N$. In Section 2.7 we will see that when the projection $\mathbb{P}$ is defined correctly, it minimize this error. The second parentheses is due to the fact that $\mathbf{U_h}(\boldsymbol{\mu})$ solves the high-fidelity problem in equation (2.12) and $V\mathbf{U_N}(\boldsymbol{\mu})$ solves the problem in equation (2.33). As $\mathbb{P}$ is an orthogonal projection the expressions in the two parentheses are orthogonal as $(\mathbf{U_h}(\boldsymbol{\mu}) - \mathbb{P}\mathbf{U_h}(\boldsymbol{\mu})) \in \hat{\mathbb{V}}_N^\perp$ and $(\mathbb{P}\mathbf{U_h}(\boldsymbol{\mu}) - V\mathbf{U_N}(\boldsymbol{\mu})) \in \hat{\mathbb{V}}_N$.

It is worth mentioning that the error in the second parentheses, namely $(\mathbb{P}\mathbf{U_h}(\boldsymbol{\mu}) - V\mathbf{U_N}(\boldsymbol{\mu}))$ is already minimized with respect to the POD basis V. As already mentioned, this error is due to the fact that $\mathbf{U_h}(\boldsymbol{\mu})$ solves the high-fidelity problem in equation (2.12) and $V\mathbf{U_N}(\boldsymbol{\mu})$ solves the problem in equation (2.33). The latter problem is in fact derived from the former through a Galerkin projection as described in [8]. This is obvious by the fact that the residual we get when inserting $\mathbf{U_h} \approx V\mathbf{U_N}$ into the linear system (2.12), is orthogonal to the columns of V

$$V^\top(K_h(\boldsymbol{\mu})V\mathbf{U_N}(\boldsymbol{\mu}) - \mathbf{F_h}(\boldsymbol{\mu})) = \mathbf{0}, \tag{2.37}$$

which can be rewritten as the high-fidelity problem (2.33).

From a computational stand point it could be preferable that we express the error between the high-fidelity approximation and the RB approximation given in equation (2.34) using vectors

$$\begin{aligned} e_h^u(\boldsymbol{\mu}) = x_h(\boldsymbol{\mu}) - x_N(\boldsymbol{\mu}) &= \sum_{j=1}^{N_h} x_h^j(\boldsymbol{\mu})\phi_j(\mathbf{x}) - \sum_{j=1}^{N_h} \phi_j \left(\sum_{i=1}^{N_u} (V_u)_{ji} \, (\mathbf{x_N}(\boldsymbol{\mu}))_i \right) \\ &= \mathbf{\Phi}^\top \mathbf{x_h}(\boldsymbol{\mu}) - \mathbf{\Phi}^\top V_u \mathbf{x_N}(\boldsymbol{\mu}) = \mathbf{\Phi}^\top \mathbf{e_h^u}(\boldsymbol{\mu}), \end{aligned}$$

where $\mathbf{\Phi} = [\phi_1, \dots, \phi_{N_h}]^\top \in \mathbb{R}^{N_h}$ is a vector with the i-th element being the i-th basis function of $\mathbb{X}_h$, and we used equations (2.9) and (2.31) to represent $x_h(\boldsymbol{\mu})$ and $x_N(\boldsymbol{\mu})$ respectively. In the same manner we get

$$\begin{aligned} e_h^p(\boldsymbol{\mu}) &= p_h(\boldsymbol{\mu}) - p_N(\boldsymbol{\mu}) = \sum_{j=1}^{M_h} p_h^j(\boldsymbol{\mu})\psi_j(\mathbf{x}) - \sum_{j=1}^{M_h} \psi_j \left(\sum_{i=1}^{N_p} (V_p)_{ji} \left(\mathbf{p_N}(\boldsymbol{\mu})\right)_i \right) \\ &= \mathbf{\Psi}^\top \mathbf{p_h}(\boldsymbol{\mu}) - \mathbf{\Psi}^\top V_p \mathbf{p_N}(\boldsymbol{\mu}) = \mathbf{\Psi}^\top \mathbf{e_h^p}(\boldsymbol{\mu}), \end{aligned}$$

where $\mathbf{\Psi} = [\psi_1, \dots, \psi_{N_h}]^\top \in \mathbb{R}^{M_h}$ is a vector with the i-th element being the i-th basis function of $\mathbb{Q}_h$, and we used equations (2.9) and (2.32) to represent $p_h(\boldsymbol{\mu})$ and $p_N(\boldsymbol{\mu})$ respectively.

If we now want to measure the errors $e_h^u = e_h^u(\boldsymbol{\mu})$ and $e_h^p = e_h^p(\boldsymbol{\mu})$ expressed as the error vectors $\mathbf{e_h^u} = \mathbf{e_h^u}(\boldsymbol{\mu})$ and $\mathbf{e_h^p} = \mathbf{e_h^p}(\boldsymbol{\mu})$ measured in some X-norm induced by some X-inner product, i.e. $||\cdot||_X^2 = (\cdot, \cdot)_X$ and some Q-norm induced by some Q-inner product, i.e. $||\cdot||_Q^2 = (\cdot, \cdot)_Q$ respectively, we get

$$\begin{aligned} ||e_h^u||_X^2 &= (e_h^u, e_h^u)_X = \left(\mathbf{\Phi}^\top \mathbf{e_h^u}(\boldsymbol{\mu}), \mathbf{\Phi}^\top \mathbf{e_h^u}(\boldsymbol{\mu})\right)_X \\ &= \mathbf{e_h^u}(\boldsymbol{\mu})^\top \left(\mathbf{\Phi}, \mathbf{\Phi}^\top\right)_X \mathbf{e_h^u}(\boldsymbol{\mu}) = \mathbf{e_h^u}(\boldsymbol{\mu})^\top X_{h,x} \mathbf{e_h^u}(\boldsymbol{\mu}), \end{aligned}$$

where $X_{h,x}$ is the matrix with elements

$$(X_{h,x})_{ij} = \left(\phi_i, \phi_j\right)_X,$$

for the X-inner product. In the same way we get

$$||e_h^p||_Q^2 = \mathbf{e_h^p}(\boldsymbol{\mu})^\top X_{h,p} \mathbf{e_h^p}(\boldsymbol{\mu}),$$

where $X_{h,p}$ is the matrix with elements

$$(X_{h,p})_{ij} = \left(\psi_i, \psi_j\right)_Q,$$

for the Q-inner product. For the H^1 seminorm, $|\cdot|_{H^1}$ we have

$$|e_h^u|_{H^1}^2 = \mathbf{e_h^u}(\boldsymbol{\mu})^\top H_x \mathbf{e_h^u}(\boldsymbol{\mu}), \qquad |e_h^p|_{H^1}^2 = \mathbf{e_h^p}(\boldsymbol{\mu})^\top H_p \mathbf{e_h^p}(\boldsymbol{\mu}),$$

where the matrix H_x and H_p have the elements

$$(H_x)_{ij} = \left(\nabla\phi_i, \nabla\phi_j\right)_{L^2}, \qquad (H_p)_{ij} = \left(\nabla\psi_i, \nabla\psi_j\right)_{L^2}.$$

2.5.4 Offline/online computations

As stated in e.g. [34, 49] affine parameter dependence of the bilinear and linear forms in (2.7) is of great importance for the efficiency of the RB methods to hold as this allows an offline/online decomposition. In the case of non-affine problems we need to make an approximate affine expansion to the problem using the Empirical Interpolation Method

(EIM) [9]. Assuming affine parameter dependence means that the parametric and spatial dependence of both the bilinear form and linear form in (2.12) can be written as

$$\begin{aligned} d(x_h, w_h; \boldsymbol{\mu}) &= \sum_{q=1}^{Q_a} \theta_a^q(\boldsymbol{\mu}) d_q(x_h, w_h) \quad \forall x_h, w_h \in \mathbb{X}_h,\ \boldsymbol{\mu} \in \mathbb{P} \\ b(x_h, q_h; \boldsymbol{\mu}) &= \sum_{q=1}^{Q_b} \theta_b^q(\boldsymbol{\mu}) b_q(x_h, q_h) \quad \forall x_h \in \mathbb{X}_h,\ q_h \in \mathbb{Q}_h,\ \boldsymbol{\mu} \in \mathbb{P} \\ f_1(w_h; \boldsymbol{\mu}) &= \sum_{q=1}^{Q_{f_1}} \theta_{f_1}^q(\boldsymbol{\mu}) f_{1q}(w_h) \quad \forall w_h \in \mathbb{X}_h,\ \boldsymbol{\mu} \in \mathbb{P} \\ f_2(q_h; \boldsymbol{\mu}) &= \sum_{q=1}^{Q_{f_2}} \theta_{f_2}^q(\boldsymbol{\mu}) f_{2q}(q_h) \quad \forall q_h \in \mathbb{Q}_h,\ \boldsymbol{\mu} \in \mathbb{P} \end{aligned} \tag{2.38}$$

Here $\theta_a^q : \mathbb{P} \to \mathbb{R}, q = 1, \dots, Q_a, \theta_b^q : \mathbb{P} \to \mathbb{R}, q = 1, \dots, Q_b$, $\theta_{f_1}^q : \mathbb{P} \to \mathbb{R}, q = 1, \dots, Q_{f_1}$ and $\theta_{f_2}^q : \mathbb{P} \to \mathbb{R}, q = 1, \dots, Q_{f_2}$ are $\boldsymbol{\mu}$-dependent functions, while $d_q : \mathbb{X}_h \times \mathbb{X}_h \to \mathbb{R}, b_q : \mathbb{X}_h \times \mathbb{Q}_h \to \mathbb{R}, f_{1q} : \mathbb{X}_h \to \mathbb{R}, f_{2q} : \mathbb{Q}_h \to \mathbb{R}$ are $\boldsymbol{\mu}$-independent. In fact this would mean that also the linear system (2.12) is affinely parametrized

$$\begin{aligned} D_h(\boldsymbol{\mu}) &= \sum_{q=1}^{Q_a} \theta_a^q(\boldsymbol{\mu}) D_h^q, \\ B_h(\boldsymbol{\mu}) &= \sum_{q=1}^{Q_b} \theta_b^q(\boldsymbol{\mu}) B_h^q, \\ \mathbf{f_{1h}}(\boldsymbol{\mu}) &= \sum_{q=1}^{Q_{f_1}} \theta_{f_1}^q(\boldsymbol{\mu}) \mathbf{f_{1h}}^q, \\ \mathbf{f_{2h}}(\boldsymbol{\mu}) &= \sum_{q=1}^{Q_{f_2}} \theta_{f_2}^q(\boldsymbol{\mu}) \mathbf{f_{2h}}^q, \end{aligned} \tag{2.39}$$

where D_h^q for $q = 1, \dots, Q_a$, B_h^q for $q = 1, \dots, Q_b$, $\mathbf{f_{1h}^q}$ for $q = 1, \dots, Q_{f_1}$ and $\mathbf{f_{2h}^q}$ for $q = 1, \dots, Q_{f_2}$ are $\boldsymbol{\mu}$-independent and have the elements

$$(D_h^q)_{ij} = d_q(\phi_j, \phi_i) \qquad 1 \le i, j \le N_h,\ 1 \le q \le Q_a, \tag{2.40}$$

$$(B_h^q)_{ij} = B_q(\phi_j, \psi_i) \qquad 1 \le j \le N_h,\ 1 \le i \le M_h\ 1 \le q \le Q_b, \tag{2.41}$$

$$(\mathbf{f_{1h}}^q)_i = f_{1q}(\phi_i) \qquad 1 \le i \le N_h,\ 1 \le q \le Q_{f_1}. \tag{2.42}$$

$$(\mathbf{f_{2h}}^q)_i = f_{2q}(\psi_i) \qquad 1 \le i \le M_h,\ 1 \le q \le Q_{f_2}. \tag{2.43}$$

This affine parametric dependence is of great importance for the linear system (2.25) when considered from a computational standpoint. Since the system in (2.12) is affinely parametrized so is (2.25)

$$D_N(\boldsymbol{\mu}) = \sum_{q=1}^{Q_a} \theta_a^q(\mu) D_N^q, \tag{2.44}$$

$$B_N(\boldsymbol{\mu}) = \sum_{q=1}^{Q_b} \theta_b^q(\mu) B_N^q, \tag{2.45}$$

$$\mathbf{f_{1N}}(\boldsymbol{\mu}) = \sum_{q=1}^{Q_{f_1}} \theta_{f_1}^q(\mu) \mathbf{f_{1N}}^q. \tag{2.46}$$

$$\mathbf{f_{2N}}(\boldsymbol{\mu}) = \sum_{q=1}^{Q_{f_2}} \theta_{f_2}^q(\mu) \mathbf{f_{2N}}^q. \tag{2.47}$$

This allows us to efficiently assemble the RB solution matrices $D_N(\boldsymbol{\mu})$ and $B_N(\boldsymbol{\mu})$ and the RB right hand side vectors $\mathbf{f_{1N}}(\boldsymbol{\mu})$ and $\mathbf{f_{2N}}(\boldsymbol{\mu})$ during the online stage of our computations. This is done by precomputing all the following matrices and vectors during the offline stage

$$D_N^q = V_u^\top D_h^q V_u, \qquad 1 \le q \le Q_a, \tag{2.48}$$

$$B_N^q = V_p^\top B_h^q V_u, \qquad 1 \le q \le Q_b, \tag{2.49}$$

$$\mathbf{f_{1N}}^q = V_u^\top \mathbf{f_{1h}}^q, \qquad 1 \le q \le Q_{f_1}, \tag{2.50}$$

$$\mathbf{f_{2N}}^q = V_u^\top \mathbf{f_{2h}}^q, \qquad 1 \le q \le Q_{f_2}, \tag{2.51}$$

— which in fact is possible as they are parameter independent — and computing the sums in equations (2.44) - (2.47).

The **offline** stage of the computations includes equations (2.40) - (2.43). As well we must assemble the matrices and vectors in equation (2.39) and solve the high-fidelity system in equation (2.12), which are all $(N_h + M_h)$ - dependent. We solve the high-fidelity system n_s times creating n_s snapshots in order to construct the transformation matrices V_u and V_p by equation (2.63) which is also $(N_h + M_h)$ - dependent. In addition we precompute the parameter independent matrices and vectors in equations (2.48) - (2.51), which are also $(N_h + M_h)$ - dependent.

The **online** stage of the computations consists of assembling the RB matrices and RB right hand side vectors given by equations (2.44) - (2.47). It also requires solving the RB linear system in equation (2.25) which are $(N_u + N_p)$-dependent, but more importantly $(N_h + M_h)$-independent which is the reason why RB methods are so efficient compared to FEM methods for $N_u \ll N_h$ and $N_p \ll M_h$.

2.6 Singular Value Decomposition

We briefely introduce the singular value decomposition, SVD. For a further discussion see [58]. For a real matrix $A \in \mathbb{R}^{m \times n}$ with rank $r \le \min(m, n)$ we diagonalize A into

$$A = U \Sigma Z^\top, \tag{2.52}$$

where the orthogonal matrices

$$U = [\boldsymbol{\xi_1}| \dots |\boldsymbol{\xi_m}] \in \mathbb{R}^{m\times m} \qquad Z = [\boldsymbol{\psi_1}| \dots |\boldsymbol{\psi_n}] \in \mathbb{R}^{n\times n}, \tag{2.53}$$

holds the left and right singular vectors of A respectively. The matrix

$$\Sigma = \operatorname{diag}(\sigma_1, \dots, \sigma_r, 0, \dots, 0) \in \mathbb{R}^{m\times n}, \qquad \sigma_1 \geq \dots \geq \sigma_r > 0 \tag{2.54}$$

contains the non-zero singular values of matrix A, with $r \leq \min(m, n)$. The column vectors of U and Z are the eigenvectors of $AA^\top$ and $A^\top A$ respectively. This is due to the fact that $AA^\top$ and $A^\top A$ are symmetric. To make this somewhat clearer we can by the SVD (2.52) compute the following

$$AA^\top = U\Sigma Z^\top (U\Sigma Z^\top)^\top = U\Sigma\Sigma^\top U^\top, \tag{2.55}$$

where

$$\Sigma\Sigma^\top = \Sigma^2 = \operatorname{diag}(\sigma_1^2, \dots, \sigma_r^2, 0, \dots, 0), \tag{2.56}$$

Since $Z^\top Z = I$. As $AA^\top$ is symmetric, the left singular vectors of A are the eigenvectors of $AA^\top$. The same holds for $A^\top A$ regarding the right singular vectors of A. $\sigma_1^2, \dots, \sigma_r^2$ are the eigenvalues of $AA^\top$ and $A^\top A$.

Remark 2.6.1. *Worth noting is that for a symmetric matrix $K \in \mathbb{R}^{m\times m}$ there is a close relationship between eigenvalues and singular values as $\sigma_i(K) = |\lambda_i(K)|$, with $\lambda_1(K) \geq \lambda_2(K) \geq \dots \geq \lambda_m(K)$ the eigenvalues of K and $\sigma_i(K)$ is the i-th singular value of K. Also, for a symmetric positive semi-definite matrix $\sigma_i(K) = \lambda_i(K)$ [58].*

Remark 2.6.2. *Because of Remark 2.6.1 and the fact that for a matrix A with r positive singular values $rank(A) = rank(\Sigma) = r$ we know that $rank(AA^\top) = rank(A^\top A) = rank(A) = r$.*

The singular values of a matrix $A \in \mathbb{R}^{m\times n}$ are related to the norm of the matrix,

$$||A||_\mathrm{F} = \sqrt{\sum_{i=1}^{r} \sigma_i^2}, \qquad ||A||_2 = \sigma_{\max},$$

where $||\cdot||_\mathrm{F}$ is the Frobenius norm. This relation between the singular values and the *norm* means that the singular values capture the energy of its matrix. This could be exploited by representing the matrix A by the best n rank approximation matrix $A_n \in \mathbb{R}^{m\times n}$

$$A_n = \sum_{i=1}^{n} \sigma_i \boldsymbol{\xi_i} \boldsymbol{\psi_i}^\top. \tag{2.57}$$

The best n rank matrix has its application in e.g. image compression [58, 49]. To get a good understanding of how much energy of the matrix each singular values contains and how many singular values to include, it is good practice to plot the singluar values of the matrix.

2.7 Proper Orthogonal Decomposition

Proper orthogonal decomposition (POD) is *Principal Component Analysis* (PCA) used in mechanical engineering. We refer the interested reader to [33] for a closer discussion of the PCA. POD finds application in reducing dimensionality of large datasets by transforming the original variables to a new set of uncorrelated variables called POD modes [20]. The aim is to save most of the energy from the original system in the first few POD modes. In this section we will construct a POD basis corresponding to a set of snapshots and storing the POD basis in a transformation matrix V, equivalent to those used to obtain RB functions in Section 2.5.

We start by choosing an appropriate set of parameters

$$\Xi_{train} = \{\boldsymbol{\mu_1}, \ldots, \boldsymbol{\mu_{n_s}}\}. \tag{2.58}$$

This set is called a training set as it will be used to create a corresponding set of high-fidelity solutions, used to train a reduced basis model to approximate the high-fidelity model. Let us denote the corresponding set of high-fidelity solutions by

$$\{u_h(\boldsymbol{\mu_1}), \ldots, u_h(\boldsymbol{\mu_{n_s}})\}$$

called snapshots, where $u_h(\boldsymbol{\mu_i}) \in V_h$ for $1 \leq i \leq n_s$. The high-fidelity solutions could for example be solutions to the saddle-point problem in (2.7). Given a basis $\{\phi_j\}_{j=1}^{N_h}$ of a finite dimensional space $\mathbb{V}_h$, where $\dim(\mathbb{V}_h) = N_h$, we can write the snapshot functions $u_h(\boldsymbol{\mu_i})$ as

$$u_h(\boldsymbol{\mu_i}) = \sum_{j=1}^{N_h} (\mathbf{u_h}(\boldsymbol{\mu_i}))_j \, \phi_j, \quad 1 \leq i \leq n_s$$

where

$$\mathbf{u_h}^{(i)} = \mathbf{u_h}(\boldsymbol{\mu_i}) = [(\mathbf{u_h}(\boldsymbol{\mu_i}))_1, \ldots, (\mathbf{u_h}(\boldsymbol{\mu_i}))_{N_h}]^\top, \quad 1 \leq i \leq n_s$$

are the snapshot vectors that represents the DoF of the snapshot functions. We save these DoF vectors, hereby called snapshots, in a matrix

$$S = [\mathbf{u_h^{(1)}} | \ldots | \mathbf{u_h^{(n_s)}}] \in \mathbb{R}^{N_h \times n_s},$$

named *the snapshot matrix*.

Now let the symmetric positive semi-definite matrix X_h be associated with the $\mathbb{V}$-inner product, i.e.

$$(X_h)_{ij} = (\phi_i, \phi_j)_{\mathbb{V}}$$

Let $W = [\mathbf{w_1} | \ldots | \mathbf{w_N}] \in \mathbb{R}^{N_h \times N}$ be an N-dimensional and orthonormal basis with respect to the $\mathbb{V}$-inner product. As the snapshots functions $u_h(\boldsymbol{\mu_i})$ belong to $\mathbb{V}_h$, we seek a POD basis which minimizes the squares of error messured in the X_h-norm between each snapshot $\mathbf{u_i}^{(i)}$ and its X_h-orthogonal projection onto the subspace spanned by W, see [49] for details. Then we denote the X_h-orthogonal projection of a snapshot $\mathbf{u_h^{(i)}} \in \mathbb{R}^{N_h}$ onto the subspace spanned by W by

$$\mathbb{P}_W^{X_h} \mathbf{u_h^{(i)}} = \sum_{j=1}^{N} (\mathbf{u_h^{(i)}}, \mathbf{w_j})_{X_h} \mathbf{w_j} = W W^\top X_h \mathbf{u_h^{(i)}} \tag{2.59}$$

As introduced in e.g. [36, 34, 49], the POD basis V is the only one N-dimensional and $\mathbb{V}$-orthonormal basis (orthonormal w.r.t. the $\mathbb{V}$-inner product) of all possible N-dimensional and $\mathbb{V}$-orthonormal bases W that minimizes the sum of squares of error between all snapshots $\mathbf{u_h^{(i)}}$, $1 \leq i \leq n_s$ and their projection $\mathbb{P}_W^{X_h}\mathbf{u_h^{(i)}}$ onto span(W), i.e.

$$\sum_{i=1}^{n_s} \|\mathbf{u_h^{(i)}} - VV^\top X_h \mathbf{u_h^{(i)}}\|_{X_h}^2 = \min_{W \in \Upsilon_N^{X_h}} \sum_{i=1}^{n_s} \|\mathbf{u_h^{(i)}} - WW^\top X_h \mathbf{u_h^{(i)}}\|_{X_h}^2, \tag{2.60}$$

where $\Upsilon_N^{X_h} = \{W \in \mathbb{R}^{N_h \times N} : W^\top X_h W = I_N\}$ and the error is measured with respect to the X_h-norm. The sum of squares of errors between each snapshot $\mathbf{u_h^{(i)}}$ and its X_h-orthogonal projection measured in equation (2.60) is by [49] the same as the sum of the squares of the eigenvalues of the neglected POD modes, that is

$$\sum_{i=1}^{n_s} \|\mathbf{u_h^{(i)}} - VV^\top X_h \mathbf{u_h^{(i)}}\|_{X_h}^2 = \sum_{i=N+1}^{r} \lambda_i. \tag{2.61}$$

As in e.g., [60, 34], we introduce *the correlation matrix C* with elements being the $\mathbb{V}$−inner-product between the different snapshot functions $u_h(\boldsymbol{\mu_i})$, i.e.

$$(C)_{ij} = (u_h(\boldsymbol{\mu_i}), u_h(\boldsymbol{\mu_j}))_{\mathbb{V}}, \quad 1 \leq i, j \leq n_s. \tag{2.62}$$

Remark 2.7.1. *As stated in [36], the correlation matrix C is symmetric positive semi-definite and has rank r given by the dimension of the span of the snapshot functions, i.e. $r = dim\left(span\left\{u_h(\boldsymbol{\mu_1}), \ldots, u_h(\boldsymbol{\mu_{n_s}})\right\}\right)$.*

C is the key to construct the POD basis. Let $\lambda_1 \geq, \ldots, \geq \lambda_r > 0$ denote the positive eigenvalues of C. Now we can construct a POD basis $V = [\boldsymbol{\xi_1}| \ldots |\boldsymbol{\xi_N}] \in \mathbb{R}^{N_h \times N}$ where $N \leq r$. By choosing the N largest eigenvalues of C and its corresponding eigenvectors $\boldsymbol{\psi_1}, \ldots, \boldsymbol{\psi_N}$ we have that

$$\boldsymbol{\xi_i} = \frac{1}{\sqrt{\lambda_i}} \sum_{j=1}^{N_h} (\boldsymbol{\psi_i})_j \mathbf{u_j} = \frac{1}{\sqrt{\lambda_i}} S\boldsymbol{\psi_i} \quad 1 \leq i \leq N. \tag{2.63}$$

The POD basis is orthonormal with respect to the $\mathbb{V}$-inner product by construction.

2.7.1 Orthonormality of the reduced basis

We want to show that the POD basis V is orthonormal with respect to the $\mathbb{V}$-inner product and hence so is also the RB functions ϕ_i^{rb} for $i = 1, \ldots, N$ given in equation (2.26).

Proof. We start by the $\mathbb{V}$-inner product of our RB functions ϕ_i^{rb}

$$\begin{aligned}\left(\phi_i^{rb}, \phi_j^{rb}\right)_{\mathbb{V}} &= \left(\sum_{k=1}^{N_h} (\boldsymbol{\xi_i})_k \phi_k, \sum_{l=1}^{N_h} (\boldsymbol{\xi_j})_l \phi_l\right)_{\mathbb{V}} = \sum_{k=1}^{N_h} \sum_{l=1}^{N_h} (\boldsymbol{\xi_i})_k \, (\phi_k, \phi_l)_{\mathbb{V}} \, (\boldsymbol{\xi_j})_l \\ &= \boldsymbol{\xi_i}^\top X_h \boldsymbol{\xi_j},\end{aligned} \tag{2.64}$$

where $\{\phi_i\}_{i=1}^{N_h}$ are the basis functions of $\mathbb{V}_h$ and $\boldsymbol{\xi_i}$ for $i = 1, \dots, N$ are the column vectors of the transformation matrix V. Studying the correlation matrix C we see that

$$\begin{aligned}
(C)_{ij} = \left(u_h(\boldsymbol{\mu_i}), u_h(\boldsymbol{\mu_j})\right)_{\mathbb{V}} &= \left(\sum_{k=1}^{N_h} \left(\mathbf{u_h^{(i)}}\right)_k \phi_k, \sum_{l=1}^{N_h} \left(\mathbf{u_h^{(j)}}\right)_l \phi_l\right)_{\mathbb{V}} \\
&= \sum_{k=1}^{N_h} \sum_{l=1}^{N_h} \left(\mathbf{u_h^{(i)}}\right)_k (\phi_k, \phi_l)_{\mathbb{V}} \left(\mathbf{u_h^{(j)}}\right)_l, \quad 1 \le i, j \le n_s \\
C =& S^\top X_h S.
\end{aligned} \tag{2.65}$$

Now, we can rewrite equation (2.64) by using the POD basis in equation (2.63) and the correlation matrix in equation (2.62)

$$\boldsymbol{\xi_i}^\top X_h \boldsymbol{\xi_j} = \frac{1}{\sqrt{\lambda_i}} (S\boldsymbol{\psi}_i)^\top X_h \frac{1}{\sqrt{\lambda_j}} \left(S\boldsymbol{\psi}_j\right) = \frac{1}{\sqrt{\lambda_i \lambda_j}} \boldsymbol{\psi}_i^\top S^\top X_h S \boldsymbol{\psi}_j = \frac{1}{\sqrt{\lambda_i \lambda_j}} \boldsymbol{\psi}_i^\top C \boldsymbol{\psi}_j. \tag{2.66}$$

As the correlation matrix C is symmetric positive semi-definite as stated in Remark 2.7.1, it has a eigenvalue decomposition

$$C = Q \Lambda Q^\top, \qquad Q = [\boldsymbol{\psi}_1 | \dots | \boldsymbol{\psi}_{n_s}] \qquad \Lambda = \mathrm{diag}(\lambda_1, \dots, \lambda_r, 0, \dots, 0),$$

with the same positive eigenvalues and the same eigenvectors as given in equation (2.63) and $r = \mathrm{rank(C)}$. Since C is symmetric positive semi-definite, its eigenvectors are orthonormal, i.e. $\boldsymbol{\psi_i}^\top \boldsymbol{\psi_j} = \delta_{ij}$, where δ_{ij} is the Kronecker Delta function. Taking advantage of this, equation (2.66) now gives us

$$\begin{aligned}
\boldsymbol{\xi_i}^\top X_h \boldsymbol{\xi_j} &= \frac{1}{\sqrt{\lambda_i \lambda_j}} \boldsymbol{\psi}_i^\top C \boldsymbol{\psi}_j = \frac{1}{\sqrt{\lambda_i \lambda_j}} \boldsymbol{\psi}_i^\top Q \Lambda Q^\top \boldsymbol{\psi}_j \\
&= \frac{1}{\sqrt{\lambda_i \lambda_j}} \mathbf{e}_i^\top \Lambda \mathbf{e}_j = \frac{\lambda_i}{\sqrt{\lambda_i \lambda_j}} \delta_{ij} = \delta_{ij},
\end{aligned}$$

where $\mathbf{e_i}$ is the i-th unit vector. Hence it follows that the RB functions are orthonormal with respect to the $\mathbb{V}$-inner product and so is the POD basis V

$$V^\top X_h V = I_N,$$

$I_N \in \mathbb{R}^{N \times N}$ is the identity matrix. □

2.7.2 POD basis minimizing the $||\cdot||_2$ norm

In [34] Hesthaven compares the relation between the SVD and POD. This is done by changing the $\mathbb{V}$-inner product in equation (2.62) with the Eucledian inner-product between the vectors $\mathbf{u_i}$ and $\mathbf{u_j}$ for $i, j = 1, \dots, N_h$, that is the vectors of the degrees of freedom for the corresponding snapshot functions $u_h(\boldsymbol{\mu_i})$ and $u_h(\boldsymbol{\mu_j})$. Then, as stated in [49], the POD basis will minimize the $||\cdot||_2$-norm. Hence the elements of the correlation matrix C given in equation (2.65) would be

$$C = S^\top S \tag{2.67}$$

From Remark 2.7.1 we know that C is symmetric positive semi-definite, and we know from Remark 2.6.1 that its eigenvalues are the same as its singular values, i.e. $\lambda_i(C) = \sigma_i(C)$ for $i = 1, \ldots, r$. Hence choosing the N largest eigenvalues of C when constructing the POD basis in equation (2.63) is equivalent to choosing the N largest singular values of C. As discussed in Section 2.6, these values capture as much of the energy of the correlation matrix C as possible.

Looking at the SVD of the snapshot matrix $S = U\Sigma Z$ where U and Z are given in equation (2.53) and Σ is given in equation (2.54), we must have that

$$S\boldsymbol{\psi_i} = \sigma_i\boldsymbol{\xi_i}$$

and

$$C = S^\top S = (U\Sigma Z)^\top (U\Sigma Z) = Z^\top\Sigma^\top\Sigma Z = Z^\top\Sigma^2 Z. \tag{2.68}$$

As discussed in Section 2.6, we now see that the right singular vectors $\boldsymbol{\psi_i}$, $i = 1, \ldots, n_s$ of S also are the eigenvectors of C as

$$C\boldsymbol{\psi_i} = S^\top S\boldsymbol{\psi_i} = \sigma_i^2\boldsymbol{\psi_i},$$

with eigenvalues $\lambda_i = \sigma_i^2$. Hence the POD basis given by (2.63) is in fact just the left singular vectors of S

$$\frac{1}{\sqrt{\lambda_i}}S\boldsymbol{\psi_i} = \frac{1}{\sigma_i}\sigma_i\boldsymbol{\xi_i} = \boldsymbol{\xi_i} \quad 1 \leq i \leq N.$$

Since we are changing the $\mathbb{V}$-inner product in equation (2.62) with the Eucledian inner-product between the vectors $\mathbf{u_i}$ and $\mathbf{u_j}$ for $i, j = 1, \ldots, N_h$, we have that the POD basis V is an N dimensional orthonormal basis, i.e. $V^\top V = I_N$.

Proof. As $C = S^\top S$ gives $V = [\boldsymbol{\xi_1}|\ldots|\boldsymbol{\xi_N}]$ we begin by the Eucledian inner product between the POD basis functions

$$\boldsymbol{\xi_i^\top}\boldsymbol{\xi_j} = \frac{1}{\sigma_i}(S\boldsymbol{\psi_i})^\top\frac{1}{\sigma_j}S\boldsymbol{\psi_j} = \frac{1}{\sigma_i\sigma_j}\boldsymbol{\psi_i}^\top S^\top S\boldsymbol{\psi_j} = \frac{1}{\sigma_i\sigma_j}\boldsymbol{\psi_i}^\top C\boldsymbol{\psi_j} \tag{2.69}$$

If we insert the SVD of C from equation (2.68) into equation (2.69) where Z holds the orthonormal eigenvectors $\boldsymbol{\psi_i}$ for $i = 1, \ldots, n_s$ of the correlation matrix, we get

$$\boldsymbol{\xi_i^\top}\boldsymbol{\xi_j} = \frac{1}{\sigma_i\sigma_j}\boldsymbol{\psi_i}^\top C\boldsymbol{\psi_j} = \frac{1}{\sigma_i\sigma_j}\boldsymbol{\psi_i}^\top Z\Sigma^2 Z^\top\boldsymbol{\psi_j} = \frac{1}{\sigma_i\sigma_j}\mathbf{e_i}^\top\Sigma^2\mathbf{e_j} = \frac{\sigma_i^2}{\sigma_i\sigma_j}\delta_{ij} = \delta_{ij}$$
$$V^\top V = I_N$$

□

The POD basis V obtained by the correlation matrix in (2.67) minimizes the $||\cdot||_2$−norm of the sum of squares of error between the snapshots $\mathbf{u_h^{(i)}}$ and their projection onto span(V) in equation (2.60), i.e.

$$\sum_{i=1}^{n_s}||\mathbf{u_h^{(i)}} - VV^\top\mathbf{u_h^{(i)}}||_2^2 = \min_{W\in\Upsilon_N}\sum_{i=1}^{n_s}||\mathbf{u_h^{(i)}} - WW^\top\mathbf{u_h^{(i)}}||_2^2, \tag{2.70}$$

where $W \in \Upsilon_N = \{W \in \mathbb{R}^{N_h \times N} : W^\top W = I_N\}$ is the set of all $N-$dimensional orthonormal bases. Recalling the best n approximation matrix of a matrix (2.57), the best rank N matrix approximation S_N of S is given by

$$S_N = \sum_{i=1}^{N} \sigma_i \boldsymbol{\xi_i} \boldsymbol{\psi_i}^\top = \sum_{i=1}^{N} \sigma_i \boldsymbol{\xi_i} \frac{1}{\sigma_i} \left(S^\top \boldsymbol{\xi_i}\right)^\top = \sum_{i=1}^{N} \boldsymbol{\xi_i} \left(\boldsymbol{\xi_i}^\top S\right) = VV^\top S$$

where we have used a property from the SVD of S, $S^\top \boldsymbol{\xi_i} = \sigma_i \boldsymbol{\psi_i}$. This again coincides with the fact that the POD basis tries to represent most of the energy from the snapshots, and it does so by minimizing the squares of errors of the difference between each snapshot $\mathbf{u_h^{(i)}}$ and the best n rank matrix approximation S_N of the set of all snapshots. Also this could be used to prove the proposition 6.1 from [49] which states that the error in equation (2.70) is the same as the sum of squares of the singluar values of the neglected POD modes

$$\sum_{i=1}^{n_s} \|\mathbf{u_h^{(i)}} - VV^\top \mathbf{u_h^{(i)}}\|_2^2 = \min_{W \in \Upsilon_N} \sum_{i=1}^{n_s} \|\mathbf{u_h^{(i)}} - WW^\top \mathbf{u_h^{(i)}}\|_2^2 = \sum_{i=N+1}^{r} \sigma_i^2, \tag{2.71}$$

where N still is the number of chosen eigenvalues in equation (2.63) and $r = \text{rank}(S)$.

Remark 2.7.2. *By choosing $\tilde{S} = X_h^{1/2} S$ and inserting instead of S in equation* (2.67) *we get*

$$C = \tilde{S}^\top \tilde{S} = S^\top X_h S$$

such that the SVD yields the same results as the POD method introduced in the previous chapter. We still need to use S and not $\tilde{S}$ to construct the POD basis if we want to use this method.

2.7.3 Choosing the right dimension

The goal is to choose as few POD bases, also called POD modes, as possible while still keeping a good enough approximation of the original data set. The approximation is often measured by the *relative information content*, used in e.g. [25, 49] and defined as

$$I(N) = \frac{\sum_{i=1}^{N} \lambda_i}{\sum_{i=1}^{r} \lambda_i}, \tag{2.72}$$

where λ_i is the i'th eigenvalue of the correlation matrix C. As the eigenvalues of the correlation matrix C describes the energy captured by the snapshots, $I(N)$ provides us with a method of how to choose N eigenvalues from the correlation matrix by choosing N as the smallest number such that the energy captured by the last $r - N$ POD modes is smaller than or equal to some tolerance of our choice $\varepsilon_{\text{tol}}^2$,

$$I(N) \geq 1 - \varepsilon_{\text{tol}}^2. \tag{2.73}$$

As suggested by equations (2.61) and (2.71) the sum of the squares of the singular values corresponding to the neglected POD modes is equal to the error in the POD basis. For instance in the case for the $\|\cdot\|_2$-norm the relative error between the snapshots stored in the matrix S and the approximation of these $VV^\top S$ provided by the chosen POD basis V is given as

$$\frac{||S - VV^\top S||_F}{||S||_F} = \sqrt{\frac{\sum_{i=N+1}^r \sigma_i^2}{\sum_{i=1}^r \sigma_i^2}} = \varepsilon. \tag{2.74}$$

The relative error in the above equation is related to the relative information content as

$$I(N) = 1 - \varepsilon^2 = \frac{\sum_{i=1}^r \sigma_i^2 - \sum_{i=N+1}^r \sigma_i^2}{\sum_{i=1}^r \sigma_i^2} = \frac{\sum_{i=1}^N \sigma_i^2}{\sum_{i=1}^r \sigma_i^2} = \frac{\sum_{i=1}^N \lambda_i^2}{\sum_{i=1}^r \lambda_i}.$$

Then the requirement in equation (2.73) is the same as requiring the relative error of the POD approximation in equation (2.74) to be smaller than or equal to ε_{tol}

$$\frac{||S - VV^\top S||_F}{||S||_F} \leq \varepsilon_{tol}.$$

2.8 Ensuring stability of the reduced saddle-point problem

In Section 2.5 we have introduced the Galerkin reduced basis formulation for a saddle-point problem, equation (2.23), and showed how to obtain a reduced basis linear system, equation (2.25). This is realized through the transformation matrices V_u and V_p which we have learned how to construct by a POD method in Section 2.7. Still we have not considered whether our reduced basis solutions will be stable or not. In order to build RB spaces providing stable reduced basis solutions for the Galerkin RB saddle-point problem a reduced inf-sup condition equivalent to the discrete inf-sup condition (2.16) must hold.

2.8.1 Reduced inf-sup condition

To obtain numerical stability, our goal is to avoid spurious modes and hence we must require that there exists $\beta_N > 0$ such that

$$\beta_N(\boldsymbol{\mu}) \inf_{\mathbf{q_N} \neq \mathbf{0}} \sup_{\mathbf{w_N} \neq \mathbf{0}} \frac{\mathbf{q_N}^\top B_N(\boldsymbol{\mu}) \mathbf{w_N}}{||\mathbf{w_N}||_{X_N} ||\mathbf{q_N}||_{Q_N}} \geq \beta_N \quad \forall \boldsymbol{\mu} \in \mathbb{P}. \tag{2.75}$$

Even though the RB linear system in equation (2.25) is built on a POD basis from stable high-fidelity solutions, this does not guarantee that the reduced inf-sup condition (2.75) holds. Hence instead of solving the RB linear system (2.25), we should enrich the space $\mathbb{X}_N$ such that the condition (2.75) holds. Then instead of $\mathbb{X}_N$ we define the space $\tilde{\mathbb{X}}_N$ with dimension $N_u + N_s$ such that

$$\tilde{\mathbb{X}}_N = \text{span}\{\phi_1^{rb}, \ldots, \phi_{N_p}^{rb}, \chi_1^{rb}, \ldots, \chi_{N_s}^{rb}\}.$$

A function $w_N(\boldsymbol{\mu})$ belonging to $\tilde{\mathbb{X}}_N$ is written in terms of its basis as

$$w_N(\boldsymbol{\mu}) = \sum_{i=1}^{N_u} w_N(\boldsymbol{\mu})^{(i)} \phi_i^{rb} + \sum_{j=1}^{N_s} s_N(\boldsymbol{\mu})^{(j)} \chi_j^{rb} \in \tilde{\mathbb{X}}_N,$$

where the vector $\mathbf{w_N} = [w_N(\boldsymbol{\mu})^{(1)}, \dots, w_N(\boldsymbol{\mu})^{(N_u)}, s_N(\boldsymbol{\mu})^{(1)}, \dots, s_N(\boldsymbol{\mu})^{(N_s)}]^\top \in \mathbb{R}^{N_u+N_s}$ holds the reduced basis coefficients of the function $w_N(\boldsymbol{\mu})$.

2.8.2 Enrichment procedures

There are two possible choices for enrichment of the space $\tilde{\mathbb{X}}_N$. The first possibility is to solve equation (2.19) choosing $\mathbf{q_h} = \boldsymbol{\xi_i}$ for $i = 1, \dots, N_p$, that is all the POD bases stored in $V_p = [\boldsymbol{\xi_1} | \dots | \boldsymbol{\xi_{N_p}}]^\top$ and storing all solutions $\mathbf{t_h}^{\boldsymbol{\mu}}(\boldsymbol{\xi_i})$ in a supremizer transformation matrix $V_s = V_s(\boldsymbol{\mu})$ that is

$$V_s = X_{h,x}^{-1} B_h^\top(\boldsymbol{\mu}) V_p.$$

In [8] the authors suggests that the set in V_s should be orthonormalized with a Gram-Schimdt procedure while the authors in [49] does not.

This solution is, as we see, parameter-dependent and hence not optimal as it must be done during the online stage and the matrix multiplications are N_h and M_h dependent. On the other hand it is still interesting as it provides a Galerkin reduced basis solution that is inf-sup stable, meaning that condition (2.75) holds. See e.g. [8, 49] for the proof of this. We will move on to the next possibility as this is the interesting case for the numerical computations.

Recalling the notion of the training set in equation (2.58), the second possibility is to make an approximate supremizer enrichment depending only on the parameter training set such that the entire enrichment can be done during the offline step. This is done by creating a supremizer snapshot matrix for the parameter training set $\Xi_{train} = \{\boldsymbol{\mu_1}, \dots, \boldsymbol{\mu_{n_s}}\}$. For all training parameters we solve the linear system (2.19) and we define the supremizer snapshot matrix

$$S_s = [\mathbf{t_h}^{\boldsymbol{\mu_1}}(\mathbf{p_h}(\boldsymbol{\mu_1})) | \dots | \mathbf{t_h}^{\boldsymbol{\mu_{ns}}}(\mathbf{p_h}(\boldsymbol{\mu_{n_s}}))].$$

Then we compute a POD basis from the correlation matrix

$$C_s = S_s^\top X_{h,x} S_s,$$

and store $N_s < n_s$ POD modes in the supremizer transformation matrix $V_s \in \mathbb{R}^{N_h \times N_s}$. Finally we define the new transformation matrix $\tilde{V}_u = [V_u \;\; V_s] \in \mathbb{R}^{N_h \times (N_u+N_s)}$ and the reduced matrices and vectors in equation (2.25) becomes

$$D_N(\boldsymbol{\mu}) = \tilde{V}_u^\top D_h(\boldsymbol{\mu}) \tilde{V}_u = \begin{bmatrix} V_u^\top D_h(\boldsymbol{\mu}) V_u & V_u^\top D_h(\boldsymbol{\mu}) V_s \\ V_s^\top D_h(\boldsymbol{\mu}) V_u & V_s^\top D_h(\boldsymbol{\mu}) V_s \end{bmatrix} \in \mathbb{R}^{(N_u+N_s)\times(N_u+N_s)}, \tag{2.76}$$

$$B_N(\boldsymbol{\mu}) = V_p^\top B_h(\boldsymbol{\mu}) \tilde{V}_u = \begin{bmatrix} V_p^\top B_h(\boldsymbol{\mu}) V_u & V_p^\top B_h(\boldsymbol{\mu}) V_s \end{bmatrix} \in \mathbb{R}^{N_p \times (N_u+N_s)}, \tag{2.77}$$

$$\mathbf{f_{1N}}(\boldsymbol{\mu}) = \tilde{V}_u^\top \mathbf{f_{1h}}(\boldsymbol{\mu}) = \begin{bmatrix} V_u^\top \mathbf{f_{1h}}(\boldsymbol{\mu}) \\ V_s^\top \mathbf{f_{1h}}(\boldsymbol{\mu}) \end{bmatrix} \in \mathbb{R}^{N_u+N_s}, \tag{2.78}$$

$$\mathbf{f_{2N}}(\boldsymbol{\mu}) = V_p^\top \mathbf{f_{2h}}(\boldsymbol{\mu}) \in \mathbb{R}^{N_p}. \tag{2.79}$$

Chapter 3

FEM solutions of the Stokes Equations

The foundation of a reduced basis solver is a set of high-fidelity solutions, called snapshots, produced from a numerical solver e.g. a finite element solver or a finite volume solver. In this chapter we study the solutions of a finite element solver for the steady Stokes equations [14]. The steady Stokes equations describe an incompressible viscous fluid where viscous forces are dominant and convective forces are neglected. This is a simplification of the Navier-Stokes equations that is acceptable for low Reynolds numbers, that is Re $\ll 1$ [48]. This happens when dealing with a very viscous liquid or flows of very low velocity, also known as creeping flows.

The Reynolds number (Re) is a dimensionless value measuring the ratio between inertial and viscous forces describing whether the flow is laminar or turbulent. Low Re indicates a viscous creeping flow, moderate Re indicates laminar flow and high Re indicates turbulent flow. The Reynolds number is defined as

$$\mathrm{Re} = \frac{\rho U L}{\mu} = \frac{U L}{\nu}, \tag{3.1}$$

where U and L are a characteristic velocity and length measures for the flow. The kinematic viscosity $\nu = \mu/\rho$ is the ratio between the viscosity μ and the mass density ρ. The material property μ describes the viscosity of a fluid, high values of μ means a thick fluid. Whereas thick fluids also have high kinematic viscosity ν so does in fact gasses because of low mass density ρ. See [63] for further details on the viscosity and kinematic viscosity.

3.1 The Stokes equation

It is possible to describe the motion of an incompressible, steady viscous fluid by the steady Stokes equation [14]. The steady Stokes equations are a simplification of the Navier-Stokes equations where the convective term is neglected due to low Reynolds numbers. This is a simplification that is possible for Re$\ll$ 1, see [48]. We study a 2D domain $\Omega \subset \mathbb{R}^2$ with a boundary $\partial\Omega = \Gamma_D \cup \Gamma_N$. The Stokes equation is given by

$$\begin{aligned} -\nu\nabla^2\mathbf{u} + \nabla p &= \mathbf{f} && \text{in } \Omega, \\ \operatorname{div} \mathbf{u} &= 0 && \text{in } \Omega, \\ \mathbf{u} &= \mathbf{g}(\mathbf{x}) && \text{on } \Gamma_D, \\ -p\hat{\mathbf{n}} + \nu(\nabla\mathbf{u})\hat{\mathbf{n}} &= \mathbf{h}(\mathbf{x}) && \text{on } \Gamma_N, \end{aligned} \tag{3.2}$$

where **u** and p represent the velocity field and the pressure field respectively and ν is the constant kinematic viscosity. **f** is some forcing term and **g(x)** is the prescribed velocity on the Dirichlet boundary Γ_D. **h**(**x**) describes the forces acting on the Neumann boundary Γ_N. The first equation in (3.2) describes the conservation of the linear momentum of the fluid whereas the second equation describes the mass conservation of the fluid [49].

The first two examples we look at will be parameter-independent hence for simplicity we have omitted the parameter-dependent notation for the solutions instead of the notation $(\mathbf{u}(\boldsymbol{\mu}), p(\boldsymbol{\mu}))$ introduced in 2 we start of by the notation $(\mathbf{u}, p)$. We will go back to the parametric dependency notation in Section 4.1.

3.1.1 Weak formulation

As pointed out in Chapter 2, the Stokes equations can be cast in the weak form of the saddle-point problem as in equation (2.1). To do so, we start by defining the space $\mathbb{X}$ from equation (2.1) and denote it instead by $\mathbb{V}$. $\mathbb{V}$ is named the velocity space and it is a Hilbert space such that $\mathbf{u} \in \mathbb{V}$. We also define the Hilbert space $\mathbb{Q}$ such that $\mathbf{p} \in \mathbb{Q}$ naming it the pressure space. To derive the weak form, we multiply both equations in (3.2) with each their proper test function and integrate over the domain Ω.

First, multiply the first equation in (3.2) by a test function $\mathbf{v} \in \mathbb{V}$ and integrate over the domain Ω

$$\int_\Omega -\nu(\nabla^2\mathbf{u})\mathbf{v}\, \mathrm{d}\Omega + \int_\Omega \nabla \mathrm{p} \cdot \mathbf{v}\, \mathrm{d}\Omega = \int_\Omega \mathbf{f} \cdot \mathbf{v}\, \mathrm{d}\Omega. \tag{3.3}$$

First we look at the integrals on the left hand side in equation (3.3). By the chain rule we get

$$\begin{aligned} \int_\Omega -\nu(\nabla^2\mathbf{u})\mathbf{v}\, \mathrm{d}\Omega + \int_\Omega \nabla \mathrm{p} \cdot \mathbf{v}\, \mathrm{d}\Omega = \int_\Omega \nu\nabla\mathbf{u} : \nabla\mathbf{v}\, \mathrm{d}\Omega - \int_\Omega \nu\nabla \cdot (\nabla\mathbf{u}\mathbf{v})\, \mathrm{d}\Omega \\ + \int_\Omega \nabla \cdot (p\mathbf{v})\, \mathrm{d}\Omega - \int_\Omega \mathrm{p}\nabla \cdot \mathbf{v}\, \mathrm{d}\Omega \end{aligned} \tag{3.4}$$

If we take the second and third integral on the right hand side in equation (3.4), then by applying Greens formula [2] we get

$$\begin{aligned}\int_\Omega \nabla\cdot(p\mathbf{v})\,\mathrm{d}\Omega - \int_\Omega \nu\nabla\cdot(\nabla\mathbf{u}\mathbf{v})\,\mathrm{d}\Omega &= \int_{\partial\Omega}(p\mathbf{v})\cdot\hat{\mathbf{n}}\,\mathrm{d}\Gamma - \int_{\partial\Omega}\nu\mathbf{v}\cdot(\nabla\mathbf{u})\hat{\mathbf{n}}\,\mathrm{d}\Gamma\\ &= \int_{\partial\Omega}\mathbf{v}\cdot\left(p\hat{\mathbf{n}} - \nu(\nabla\mathbf{u})\hat{\mathbf{n}}\right)\,\mathrm{d}\Gamma\\ &= \int_{\Gamma_D}\mathbf{v}\cdot\left(p\hat{\mathbf{n}} - \nu(\nabla\mathbf{u})\hat{\mathbf{n}}\right)\,\mathrm{d}\Gamma + \int_{\Gamma_\mathrm{N}}\mathbf{v}\cdot\left(\mathrm{p}\hat{\mathbf{n}} - \nu(\nabla\mathbf{u})\hat{\mathbf{n}}\right)\,\mathrm{d}\Gamma\\ &= 0 - \int_{\Gamma_N}\mathbf{h}\cdot\mathbf{v}\,\mathrm{d}\Gamma.\end{aligned} \tag{3.5}$$

In the last line we inserted the boundary conditions from (3.2) and used that $\mathbf{v}|_{\Gamma_D} = \mathbf{0}$. Using equations (3.4) and (3.5) we can write equation (3.3) as

$$\int_\Omega \nu\nabla\mathbf{u} : \nabla\mathbf{v}\,\mathrm{d}\Omega - \int_\Omega \mathrm{p}\nabla\cdot\mathbf{v}\,\mathrm{d}\Omega = \int_\Omega \mathbf{f}\cdot\mathbf{v}\,\mathrm{d}\Omega + \int_{\Gamma_\mathrm{N}}\mathbf{h}\cdot\mathbf{v}\,\mathrm{d}\Gamma =: \mathrm{f}(\mathbf{v}). \tag{3.6}$$

Upon defining $\mathbf{u} = \mathring{\mathbf{u}} + \mathbf{r_g}$, where $\mathbf{r_g}$ is a lifting function such that $\mathbf{r_g}|_{\Gamma_D} = g(\mathbf{x})$, then equation (3.6) can also be written in the abstract form

$$a(\mathring{\mathbf{u}}, \mathbf{v}) + b(\mathbf{v}, p) = f(\mathbf{v}) - a(\mathbf{r_g}, \mathbf{v}). \tag{3.7}$$

Here the bilinear and symmetric form $a : \mathbb{V}\times\mathbb{V} \longrightarrow \mathbb{R}$ is

$$a(\mathring{\mathbf{u}}, \mathbf{v}) = \int_\Omega \nu\nabla\mathring{\mathbf{u}} : \nabla\mathbf{v}\,\mathrm{d}\Omega. \tag{3.8}$$

The bilinear form $b : \mathbb{V}\times\mathbb{Q} \longrightarrow \mathbb{R}$ is

$$b(\mathbf{v}, p) = -\int_\Omega p\nabla\cdot\mathbf{v}\,\mathrm{d}\Omega. \tag{3.9}$$

The linear functional $f \in \mathbb{V}'$ is

$$f(\mathbf{v}) = \int_\Omega \mathbf{f}\cdot\mathbf{v}\,\mathrm{d}\Omega + \int_{\Gamma_\mathrm{N}}\mathbf{h}\cdot\mathbf{v}\,\mathrm{d}\Gamma. \tag{3.10}$$

We define the right hand side of equation (3.7) as a linear functional $f \in \mathbb{V}'$ such that

$$f_1(\mathbf{v}) := f(\mathbf{v}) - a(\mathbf{r_g}, \mathbf{v}) = \int_\Omega \mathbf{f}\cdot\mathbf{v}\,\mathrm{d}\Omega + \int_{\Gamma_\mathrm{N}}\mathbf{h}\cdot\mathbf{v}\,\mathrm{d}\Gamma - \int_\Omega \nu\nabla\mathbf{r_g} : \nabla\mathbf{v}\,\mathrm{d}\Omega. \tag{3.11}$$

Notice that $a(\mathring{\mathbf{u}}, \mathbf{v}) + a(\mathbf{r_g}, \mathbf{v}) = a(\mathring{\mathbf{u}} + \mathbf{r_g}, \mathbf{v}) = a(\mathbf{u}, \mathbf{v}) = \int_\Omega \nu\nabla\mathbf{u} : \nabla\mathbf{v}\,\mathrm{d}\Omega$.

Second, multiply the second equation in (3.2) by a test function $q \in \mathbb{Q}$ and integrate over the domain Ω

$$\int_\Omega q\nabla\cdot\mathbf{u}\,\mathrm{d}\Omega = 0 \tag{3.12}$$

As for the first equation, we write the solution as a sum of the homogeneous solution and a lifting function $\mathbf{u} = \mathring{\mathbf{u}} + \mathbf{r_g}$, where $\mathbf{r_g}$ is such that $\mathbf{r_g}|_{\Gamma_D} = g(\mathbf{x})$. Then we write equation (3.12) in the abstract form

$$b(\mathring{\mathbf{u}}, q) = -b(\mathbf{r_g}, q), \tag{3.13}$$

where upon we define the linear functional $f_2 \in \mathbb{Q}'$ such that

$$f_2(q) = -b(\mathbf{r_g}, q). \tag{3.14}$$

From now on we write $\mathbf{u}$ instead of $\mathring{\mathbf{u}}$ and we define the velocity space to be the 2D Hilbert space $\mathbb{V} = [H^1_{\Gamma_D}(\Omega)]^2$ and the pressure space to be the space of all square integrable functions $\mathbb{Q} = L^2(\Omega)$. The weak formulation of the problem (3.2) is given by: find $(\mathbf{u}, p) \in \mathbb{V} \times \mathbb{Q}$ such that

$$\begin{aligned} a(\mathbf{u}, \mathbf{v}) + b(\mathbf{v}, p) &= f_1(\mathbf{v}), \quad \forall \mathbf{v} \in \mathbb{V} \\ b(\mathbf{u}, q) &= f_2(q) \quad \forall q \in \mathbb{Q}, \end{aligned} \tag{3.15}$$

and the solution of problem (3.2) is obtained as the sum $\mathbf{u} + \mathbf{r_g}$. By changing the bilinear form $d(\cdot, \cdot)$ with $a(\cdot, \cdot)$ in equation (2.1) we see that it is the same weak formulation as in equation (3.15) (without the $\boldsymbol{\mu}$-dependency and the change of spaces of course).

The existence of a unique solution of problem (3.15) is proved by showing that the following conditions holds [48]
First we require the bilinear forms $a(\cdot, \cdot)$ and $b(\cdot, \cdot)$ to be continuous, equivalent to what we did in equation (2.3). there exists two constants $\gamma_a, \gamma_b > 0$ such that

$$|a(\mathbf{v}, \mathbf{w})| \leq \gamma_a \|\mathbf{v}\|_{\mathbb{V}} \|\mathbf{w}\|_{\mathbb{V}}, \quad |b(\mathbf{w}, q)| \leq \gamma_b \|\mathbf{w}\|_{\mathbb{V}} \|q\|_{\mathbb{Q}} \qquad \forall \mathbf{v}, \mathbf{w} \in \mathbb{V},\ \forall q \in \mathbb{Q}. \tag{3.16}$$

Second we require the bilinear form $a(\cdot, \cdot)$ to be coercive on the space $\mathbb{V}^0$, that is there exist a constant $\alpha > 0$ such that

$$a(\mathbf{v}, \mathbf{v}) > \alpha \|\mathbf{v}\|^2_{\mathbb{V}} \quad \forall \mathbf{v} \in \mathbb{V}^0, \tag{3.17}$$

where $\mathbb{V}^0 = \{\mathbf{v} \in \mathbb{V} : b(\mathbf{v}, q) = 0\ \forall q \in \mathbb{Q}\}$.

Third we require that the bilinear form $b(\cdot, \cdot)$ satisfies the LBB-condition (2.5).

3.1.2 Galerkin approximation

To obtain a numerical approximation of the weak formulation in equation (3.15), we need to project the original infinite problem onto a finite-dimensional subspace. We approximate the infinite space $\mathbb{V}$ by a finite dimensional subspace denoted $\mathbb{V}_h = [\tilde{\mathbb{V}}_h]^2$, and similar for the infinite space $\mathbb{Q}$ we approximate it by a finite dimensional subspace denoted $\mathbb{Q}_h$. Let the dimensions of the two approximation spaces be $\mathrm{dim}(\mathbb{V}_\mathrm{h}) = \mathrm{N_h} = 2\,\mathrm{dim}(\tilde{\mathbb{V}}_\mathrm{h}) = 2\tilde{\mathrm{N}}_\mathrm{h}$ and $\mathrm{dim}(\mathbb{Q}_\mathrm{h}) = \mathrm{M_h}$. Then the Galerkin approximation of the weak formulation (3.15) reads: find $(\mathbf{u}_h, p_h) \in \mathbb{V}_h \times \mathbb{Q}_h$ such that

$$\begin{aligned} a(\mathbf{u_h}, \mathbf{v_h}) + b(\mathbf{v_h}, p_h) &= f_1(\mathbf{v_h}), \quad \forall \mathbf{v_h} \in \mathbb{V}_h \\ b(\mathbf{u_h}, q_h) &= f_2(q_h) \quad \forall q_h \in \mathbb{Q}_h, \end{aligned} \tag{3.18}$$

where the bilinear forms $a(\cdot,\cdot)$, $b(\cdot,\cdot)$ are the same as given in equations (3.8) and (3.9). The linear forms $f_1(\cdot)$, $f_2(\cdot)$ are the same as in equations (3.11) and (3.14).

Let $\{\boldsymbol{\phi_i}\}_{i=1}^{N_h}$ denote a basis for $\mathbb{V}_h$ such that $\{\boldsymbol{\phi_1}, \cdots, \boldsymbol{\phi_{\tilde{N}_h}}, \boldsymbol{\phi_{\tilde{N}_h+1}}, \cdots, \boldsymbol{\phi_{N_h}}\} = \{\left[\begin{smallmatrix}\phi_1\\0\end{smallmatrix}\right], \cdots, \left[\begin{smallmatrix}\phi_{\tilde{N}_h}\\0\end{smallmatrix}\right], \left[\begin{smallmatrix}0\\\phi_1\end{smallmatrix}\right], \cdots, \left[\begin{smallmatrix}0\\\phi_{\tilde{N}_h}\end{smallmatrix}\right]\}$ where $\{\phi_i\}_{i=1}^{\tilde{N}_h}$ denotes a basis for $\tilde{\mathbb{V}}_h$. Finally, let $\{\psi_i\}_{i=1}^{M_h}$ denote a basis for $\mathbb{Q}_h$.

We set the Galerkin solutions of problem (3.18) as

$$\mathbf{u_h} = \sum_{j=1}^{N_h} w_h^{(j)} \boldsymbol{\phi_j} \qquad p_h = \sum_{k=1}^{M_h} p_h^{(k)} \psi_k \tag{3.19}$$

where the vectors $\mathbf{w_h}$ and $\mathbf{p_h}$ hold the unknown coefficients $w_h^{(j)}$ and $p_h^{(i)}$. As equation (3.18) holds for all functions $\mathbf{v_h}$ and q_h, we can choose the test functions $\mathbf{v_h} = \boldsymbol{\phi_i}$ and $q_h = \psi_i$. Inserting these together with the formulations (3.19) into equation (3.18) we obtain

$$\begin{gathered} a\left(\sum_{j=1}^{N_h} w_h^{(j)} \boldsymbol{\phi_j}, \boldsymbol{\phi_i}\right) + b\left(\boldsymbol{\phi_i}, \sum_{i=k}^{M_h} p_h^{(k)} \psi_k\right) = f_1(\boldsymbol{\phi_i}), \quad \text{for } i = 1, \ldots, N_h, \\ \sum_{j=1}^{N_h} a\left(\boldsymbol{\phi_j}, \boldsymbol{\phi_i}\right) w_h^{(j)} + \sum_{k=1}^{M_h} b\left(\boldsymbol{\phi_i}, \psi_k\right) p_h^{(k)} = f_1(\boldsymbol{\phi_i}), \quad \text{for } i = 1, \ldots, N_h, \\ A_h \mathbf{w_h} + B_h^\top \mathbf{p_h} = \mathbf{f_1}, \end{gathered} \tag{3.20}$$

and

$$\begin{gathered} b\left(\sum_{j=1}^{N_h} w_h^{(j)} \boldsymbol{\phi_j}, \psi_i\right) = f_2(\psi_i) \quad \text{for } i = 1, \ldots, M_h, \\ \sum_{j=1}^{N_h} b(\boldsymbol{\phi_j}, \psi_i) w_h^{(j)} = f_2(\psi_i) \quad \text{for } i = 1, \ldots, M_h, \\ B_h \mathbf{w_h} = \mathbf{f_2}. \end{gathered} \tag{3.21}$$

The elements of the matrix A are given as

$$(A_h)_{kl} = \begin{cases} a\left(\left[\begin{smallmatrix}\phi_j\\0\end{smallmatrix}\right], \left[\begin{smallmatrix}\phi_i\\0\end{smallmatrix}\right]\right) = \nu \sum_{l=0}^{1} \int_\Omega \phi_{i,x_l} \phi_{j,x_l} \, \mathrm{d}\Omega & 1 \le i,j \le \tilde{N}_h,\ k = i,\ l = j \\ a\left(\left[\begin{smallmatrix}\phi_j\\0\end{smallmatrix}\right], \left[\begin{smallmatrix}0\\\phi_i\end{smallmatrix}\right]\right) = 0 & 1 \le i,j \le \tilde{N}_h,\ k = i + \tilde{N}_h,\ l = j \\ a\left(\left[\begin{smallmatrix}0\\\phi_j\end{smallmatrix}\right], \left[\begin{smallmatrix}\phi_i\\0\end{smallmatrix}\right]\right) = 0 & 1 \le i,j \le \tilde{N}_h,\ k = i,\ l = j + \tilde{N}_h \\ a\left(\left[\begin{smallmatrix}0\\\phi_j\end{smallmatrix}\right], \left[\begin{smallmatrix}0\\\phi_i\end{smallmatrix}\right]\right) = a\left(\left[\begin{smallmatrix}\phi_j\\0\end{smallmatrix}\right], \left[\begin{smallmatrix}\phi_i\\0\end{smallmatrix}\right]\right) & 1 \le i,j \le \tilde{N}_h,\ k = i + \tilde{N}_h,\ l = j + \tilde{N}_h \end{cases}, \tag{3.22}$$

that is A is a block matrix of the form

$$A_h = \begin{bmatrix} A_{h1} & 0 \\ 0 & A_{h1}, \end{bmatrix}. \tag{3.23}$$

where

$$(A_{h1})_{ij} = \nu \int_{\Omega} \phi_{i,x_0}\phi_{j,x_0} + \phi_{i,x_1}\phi_{j,x_1} \,\mathrm{d}\Omega \qquad 1 \leq \mathrm{i,j} \leq \tilde{\mathrm{N}}_{\mathrm{h}}. \tag{3.24}$$

The elements of the matrix B_h are given as

$$(B_h)_{il} = \begin{cases} b\left(\left[\begin{smallmatrix}\phi_j\\0\end{smallmatrix}\right], \psi_i\right) = B_{h1} = -\int_{\Omega} \phi_{j,x_0}\psi_i \,\mathrm{d}\Omega & 1 \leq i \leq M_h,\ 1 \leq j \leq \tilde{N}_h,\ l = j \\ b\left(\left[\begin{smallmatrix}0\\\phi_j\end{smallmatrix}\right], \psi_i\right) = B_{h2} = -\int_{\Omega} \phi_{j,x_1}\psi_i \,\mathrm{d}\Omega & 1 \leq i \leq M_h,\ 1 \leq j \leq \tilde{N}_h,\ l = j + \tilde{N}_h \end{cases}, \tag{3.25}$$

that is B_h is a matrix of the form

$$B_h = \begin{bmatrix} B_{h1} & B_{h2} \end{bmatrix}. \tag{3.26}$$

To write the elements of $\mathbf{f_{h1}}$ in equation (3.20) and the elements of $\mathbf{f_{h2}}$ in equation (3.21) explicitly, we first denote the discretization of the lifting function $\mathbf{r_g}$ by $\mathbf{u_g} = \sum_{i=1}^{N_h} w_g^{(i)} \boldsymbol{\phi_i}$ where $\mathbf{w_g} \in \mathbb{R}^{N_h}$ holds the known coefficients of the discrete lifting function. $\mathbf{w_g}$ can also be written as a concatenation of the two vectors $\mathbf{w_{g_1}} \in \mathbb{R}^{\tilde{N}_h}$ and $\mathbf{w_{g_2}} \in \mathbb{R}^{\tilde{N}_h}$ such that $\mathbf{w_g} = \begin{bmatrix} \mathbf{w_{g_1}} \\ \mathbf{w_{g_2}} \end{bmatrix}$. Here $\mathbf{w_{g_1}}$ is the coefficients associated with the basis functions $\boldsymbol{\phi_i}$ for $1 \leq i \leq \tilde{N}_h$ and $\mathbf{w_{g_2}}$ is the coefficients associated with the basis functions $\boldsymbol{\phi_i}$ for $\tilde{N}_h + 1 \leq i \leq 2\tilde{N}_h$ holding the $x-$ and $y-$coordinates of the discrete lifting function respectively. The elements of the vector $\mathbf{f_{h1}}$ are given as

$$(\mathbf{f_{h1}})_j = \begin{cases} f_1\left(\left[\begin{smallmatrix}\phi_i\\0\end{smallmatrix}\right]\right) = \int_{\Omega}(\mathbf{f})_1\phi_i \,\mathrm{d}\Omega + \int_{\Gamma_\mathrm{N}}(\mathbf{h})_1\phi_\mathrm{i} \,\mathrm{d}\Gamma - (\mathrm{A_{h1}}\mathbf{w_{g_1}})_\mathrm{i} & 1 \leq i \leq \tilde{N}_h,\ j = i, \\ f_1\left(\left[\begin{smallmatrix}0\\\phi_i\end{smallmatrix}\right]\right) = \int_{\Omega}(\mathbf{f})_2\phi_i \,\mathrm{d}\Omega + \int_{\Gamma_\mathrm{N}}(\mathbf{h})_2\phi_\mathrm{i} \,\mathrm{d}\Gamma - (\mathrm{A_{h1}}\mathbf{w_{g_2}})_\mathrm{i} & 1 \leq i \leq \tilde{N}_h,\ j = i + \tilde{N}_h, \end{cases} \tag{3.27}$$

where $(A_{h1}\mathbf{w_{g_r}})_i$ is the i-th element of the vector $(A_{h1}\mathbf{w_{g_r}})$ for $r = 1, 2$. The elements of the vector $\mathbf{f_{h2}}$ are given as

$$(\mathbf{f_{h2}})_i = f_2(\psi_i) = -(B_h\mathbf{w_g})_i \qquad 1 \leq i \leq M_h. \tag{3.28}$$

By equation (3.20) and (3.21) we can write the Galerkin approximation (3.18) as the linear system

$$\underbrace{\begin{bmatrix} A_h & B_h^\top \\ B_h & 0 \end{bmatrix}}_{K_h} \underbrace{\begin{bmatrix} \mathbf{w_h} \\ \mathbf{p_h} \end{bmatrix}}_{\mathbf{U_h}} = \underbrace{\begin{bmatrix} \mathbf{f_{h1}} \\ \mathbf{f_{h2}} \end{bmatrix}}_{\mathbf{F_h}} \tag{3.29}$$

where the elements are as given before. This linear system is very similar to the one for the saddle-point problem in equation (2.12). For the Galerkin problem (3.18) to have a unique solution, three conditions must hold as for the Galerkin approximation of the saddle-point problem (2.7).

First the bilinear forms $a(\cdot,\cdot)$ and $b(\cdot,\cdot)$ must be continuous, that is to satisfy the continuity condition (3.16).

Second the former bilinear form must satisfy a coercivity condtion equivalent to(2.15), that is there exists a constant $\alpha_h > 0$ such that

$$a(\mathbf{v_h}, \mathbf{v_h}) > \alpha_h \|\mathbf{v_h}\|^2_{\mathbb{V}} \quad \forall \mathbf{v_h} \in \mathbb{V}^0_h,$$

where $\mathbb{V}^0_h = \{\mathbf{v_h} \in \mathbb{V}_h : b(\mathbf{v_h}, q_h) = 0, \quad \forall q_h \in \mathbb{Q}_h$.

Third, the bilinear form $b(\cdot, \cdot)$ must satisfy the discrete inf-sup condition (2.16).

The first two conditions are fairly easy to prove holding, but the last must be paid some more attention. In the case where the inf-sup condition does not hold, we get spurious modes, or *spurious pressure modes* as it is named for the Stokes problem, see Remark 2.2.2 for further details. By choosing inf-sup stable elements for the finite element spaces $\mathbb{V}_h$ and $\mathbb{Q}_h$, the inf-sup condition (2.16) holds and hence the linear system (3.29) is non-singular. This we can see by the following.

By the coercivity of $a(\cdot, \cdot)$ the A_h matrix is nonsingular and from equation (3.20) we have that

$$\mathbf{w_h} = A_h^{-1}(\mathbf{f_{h1}} - B_h^\top \mathbf{p_h}). \tag{3.30}$$

Inserting equation (3.30) into equation (3.21) we get that

$$B_h A_h^{-1} B_h^\top = B_h A_h^{-1} \mathbf{f_{h1}} - \mathbf{f_{h2}}. \tag{3.31}$$

Then we have a unique solution $(\mathbf{w_h}, \mathbf{p_h})$ in the case where equation (3.31) admits a unique solution. Since A_h is symmetric positive definite, we get a unique solution of equation (3.31) when $\ker B_h^\top = \{0\}$. As shown by Quarteroni, this is equivalent to the discrete inf-sup condition for the Galerkin approximation (3.18), [48].

3.2 Example 1: Verification on the unit square

To verify the FEM solver this thesis is built upon, we test it for some reference solution on the unit square $[0, 1] \times [0, 1]$. When choosing a reference solution, certain things must be kept in mind. First of all we want a divergence free reference solution. White explains how this is possible when deriving the velocity vector $\mathbf{u} = [\begin{smallmatrix} u \\ v \end{smallmatrix}]$ from a clever device such as a stream function $\phi(x, y)$, see [63]. The velocity vector is then

$$u = \frac{\partial \phi}{\partial y} \quad y = -\frac{\partial \phi}{\partial x} \implies \nabla \cdot \mathbf{u} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = \frac{\partial^2 \phi}{\partial x \partial y} - \frac{\partial^2 \phi}{\partial y \partial x} = 0.$$

Also it is necessary to ensure the reference solution and the boundary conditions coincides. Last but not least, it is important that we do not choose a pure polynomial as a reference solution in which case the FEM solver would solve the problem exactly. If so, we would not be able to check the convergence of the method.

To verify the logic of the FEM solver used for the Stokes equations we choose the function

$$\phi = x^2(1 - x)^2 y^2 (1 - y)^2 e^{xy}, \tag{3.32}$$

and we get the reference solution

$$\mathbf{u} = \begin{bmatrix} \frac{\partial \phi}{\partial y} \\ -\frac{\partial \phi}{\partial x} \end{bmatrix}. \tag{3.33}$$

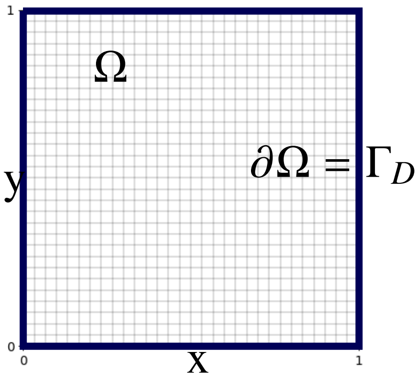


Figure 3.1: Mesh of physical space with $N \times N = 30 \times 30$ elements and the whole boundary $\partial\Omega$ being a homogeneous Dirichlet boundary.

Also, we choose the pressure to be given as $p = e^{xy}$. On the unit square $\Omega = [0, 1] \times [0, 1]$, the reference solution solves the steady Stokes problem (3.2) with homogeneous Dirichlet boundary conditions, that is

$$\begin{aligned} -\nu\nabla^2\mathbf{u} + \nabla p = \mathbf{f} &\qquad \text{in } \Omega, \\ \operatorname{div} \mathbf{u} = 0 &\qquad \text{in } \Omega, \\ \mathbf{u} = \mathbf{0} &\qquad \text{on } \Gamma_D, \end{aligned} \tag{3.34}$$

where Γ_D is the whole boundary of the unit square, see Figure 3.1.

The weak formulation of problem (3.34) is:
Find $(\mathbf{u}, p) \in \mathbb{V} \times \mathbb{Q}$ such that

$$\begin{aligned} a(\mathbf{u}, \mathbf{v}) + b(\mathbf{v}, p) = f_1(\mathbf{v}), &\quad \forall \mathbf{v} \in \mathbb{V} \\ b(\mathbf{u}, q) = 0 \quad \forall q \in \mathbb{Q}. \end{aligned} \tag{3.35}$$

where $\mathbb{V} = [H_0^1(\Omega)]^2$ and $\mathbb{Q} = L_0^2(\Omega)$. From *theorem* 15.2 in [48] we know that $(\mathbf{u}, p)$ solves equation (3.35), if and only if, it is a saddle-point of the Lagrangian functional

$$\mathcal{L}(\mathbf{v}, q) = \frac{1}{2}a(\mathbf{v}, \mathbf{v}) + b(\mathbf{v}, q) - f_1(\mathbf{v})$$

where the pressure q plays the role of the Lagrange multiplier associated with the divergence-free constraint in equation (3.35).

Compared to the problem (3.2) we see that problem (3.34) has no boundary conditions for the pressure p, and hence the pressure is not uniquely defined for this case. To deal with this, we require that the integral of the pressure p over the unit square Ω to be some value C. For this particular example C will be the value

$$C = \int_0^1 \int_0^1 e^{xy} \mathrm{d}x\mathrm{d}y \approx 1.3179.$$

3.2.1 Galerkin high-fidelity approximation

We denote by $\mathbf{u_h}$ and p_h the Galerkin approximation of the velocity function $\mathbf{u}$ and the pressure function p respectively. Let us approximate the velocity space $\mathbb{V}$ and pressure space $\mathbb{Q}$ with the discrete spaces $\mathbb{V}_h = \{\mathbf{v} : \mathbf{v} \circ \Phi \in \mathbb{S}^{2,2}_{0,0} \times \mathbb{S}^{2,2}_{0,0}\}$ and $\mathbb{Q}_h = \{q : q \circ \Phi \in \mathbb{S}^{1,1}_{0,0}\}$ respectively. These spaces are inf-sup stable Taylor-Hood element spaces from Section 2.2.3 and the mapping Φ is the mapping from the parameter space to the original space defined in Section 2.2.4. We derive the Galerkin formulation of problem (3.34) by multiplying each equation with each test function $\mathbf{v_h} \in \mathbb{V}_h$ and $q_h \in \mathbb{Q}_h$ respectively and integrating over the domain Ω, while at the same time requiring the pressure integral to be constant. The Galerkin problem reads: find $(\mathbf{u}_h, p_h) \in \mathbb{V}_h \times \mathbb{Q}_h$ such that

$$\begin{aligned} a(\mathbf{u_h}, \mathbf{v_h}) + b(\mathbf{v_h}, p_h) &= f_1(\mathbf{v_h}), \quad \forall \mathbf{v_h} \in \mathbb{V}_h \\ b(\mathbf{u_h}, q_h) &= 0 \quad \forall q_h \in \mathbb{Q}_h. \\ (p_h, 1)_{L^2(\Omega)} &= C, \end{aligned} \tag{3.36}$$

where the bilinear forms $a(\cdot,\cdot)$, $b(\cdot,\cdot)$ are the same as given in equations (3.8) and (3.9). The last equation is the extra condition on the pressure, $(p_h, 1)_{L^2(\Omega)} = \int_\Omega p_h \, \mathrm{d}\Omega = C$. The linear form $f_1(\cdot)$ is

$$f_1(\mathbf{v_h}) = \int_\Omega \mathbf{f} \cdot \mathbf{v_h} \, \mathrm{d}\Omega. \tag{3.37}$$

If we proceed in the same manner as we did when we derived the linear systems (3.20) and (3.21), we derive from equations (3.36) the linear system

$$\begin{bmatrix} A_h & B_h^\top & 0 \\ B_h & 0 & \mathbf{L} \\ 0 & \mathbf{L}^\top & 0 \end{bmatrix} \begin{bmatrix} \mathbf{w_h} \\ \mathbf{p_h} \\ l \end{bmatrix} = \begin{bmatrix} \mathbf{f_{h1}} \\ \mathbf{0} \\ C \end{bmatrix}, \tag{3.38}$$

where the elements of matrix A_h and B_h are given in equations (3.22) and (3.25) respectively. The elements of vector $\mathbf{L}$ are

$$(\mathbf{L})_i = (\psi_i, 1)_{L^2(\Omega)} = \int_\Omega \psi_i \, \mathrm{d}\Omega. \tag{3.39}$$

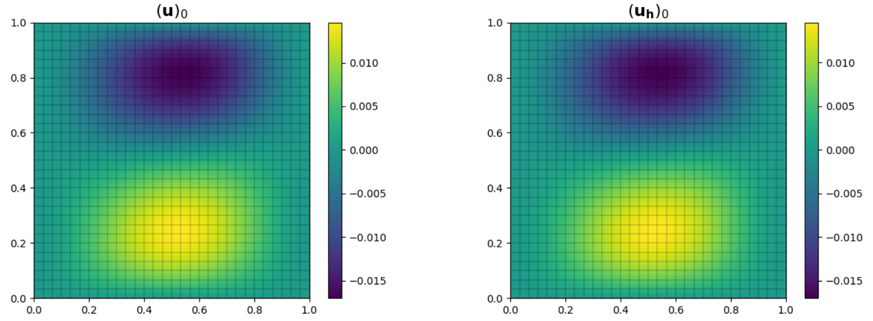
The elements of the vector $\mathbf{f_{h1}}$ are

$$(\mathbf{f_{h1}})_j = \begin{cases} f_1\left(\left[\begin{smallmatrix} \phi_i \\ 0 \end{smallmatrix}\right]\right) = \int_\Omega (\mathbf{f})_1 \phi_i \, \mathrm{d}\Omega & 1 \le i \le \tilde{N}_h, \ j = i \\ f_1\left(\left[\begin{smallmatrix} 0 \\ \phi_i \end{smallmatrix}\right]\right) = \int_\Omega (\mathbf{f})_2 \phi_i \, \mathrm{d}\Omega & 1 \le i \le \tilde{N}_h, \ j = i + \tilde{N}_h \end{cases}. \tag{3.40}$$

In equation (3.38) the expression l appears and it is called the Lagrange multiplier. For a further discussion on this Lagrange multiplier, we refer the interested reader to e.g. [11, 47].

3.2.2 Solution and convergence

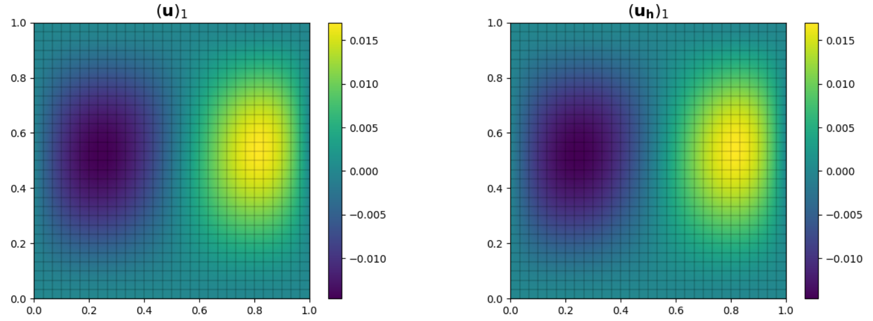
For the given reference solution (3.33) we can compute $\mathbf{f}$ from equation (3.34) and we obtain an approximate solution $(\mathbf{u_h}, p_h)$ solving the linear system (3.38). As the spline



(a) The exact velocity u.

(b) The Galerkin approximated velocity u_h

Figure 3.2: Velocity in the x-direction for problem (3.34) both exact and Galerkin approximation.



(a) The exact velocity v.

(b) The Galerkin approximated velocity v_h

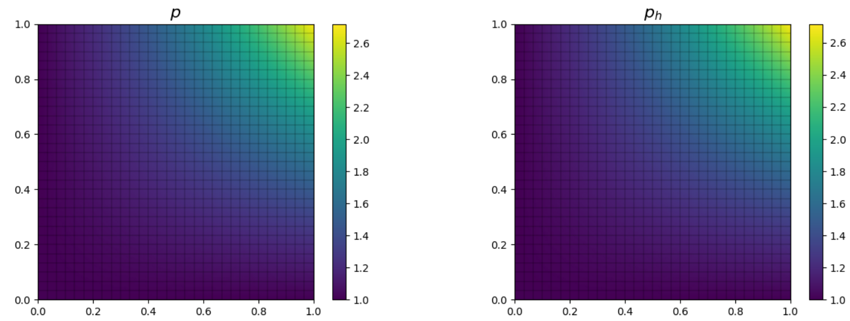
Figure 3.3: Velocity in the y-direction for problem (3.34) both exact and Galerkin approximation.

spaces indicate, we solve the problem using splines of polynomial degree $P_u = 2$ for the velocity basis and polynomial degree $P_p = 1$ for the pressure basis. We solve the problem on a 30×30 elements grid resulting in $N_h = 2 \cdot \tilde{N}_h = 2 \cdot 3962$ $M_h = 961$ degrees of freedom (DoF). The exact and approximate solution of the velocity in the x-direction, the y-direction and the pressure is shown in Figure 3.2, 3.3 and 3.4 respectively.

When considering the convergence of the provided steady Stokes FEM solver we should have some sort of notion of what is optimal convergence. Given a domain Ω and a smooth function $f : \Omega \longrightarrow \mathbb{R}^2$ we approximate f with the finite element solution f_h belonging to a finite element space $\mathbb{V}_h$. then we obtain optimal convergence in $L^2(\Omega)$ or equivalently $\mathbb{V}_h$ is optimally convergent in $L^2(\Omega)$ if

$$\inf_{f_h \in V_h} \|f_h - f\|_{L^2(\Omega)} = O(h^{s+1}), \tag{3.41}$$

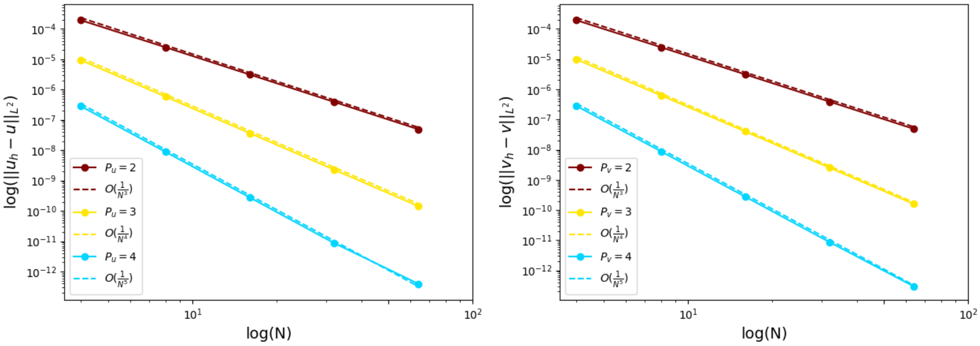
where s is the polynomial degree of the finite element space $\mathbb{V}_h$ and h is the length of the element. For further details see [12]. In Section 2.2.3 we state that the choice of inf-sup



(a) The exact pressure p.

(b) The Galerkin approximated pressure p_h

Figure 3.4: Pressure for problem (3.34) both exact and Galerkin approximation.



(a) Error of velocity in x-direction measured in the L^2-norm.

(b) Error of velocity in y-direction measured in the L^2-norm.

Figure 3.5: Convergence plots for error between exact solution $(u(x, y), v(x, y))$ and high-fidelity approximation $(u_h(x, y), v_h(x, y))$. N is the element size in each spatial direction.

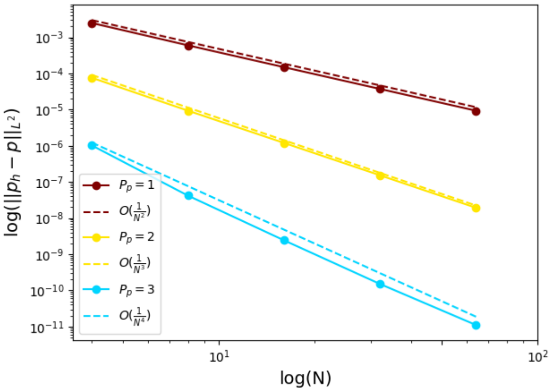


Figure 3.6: Convergence plot for error between exact solution $p(x, y)$ and high-fidelity approximation $p_h(x, y)$ measured in the L^2-norm. N is the element size in each spatial direction.

Table 3.1: $N_h = 2 \cdot \tilde{N}_h$ degrees of freedom for the velocity field for different polynomial degrees and grid partitioning used in the convergence plot 3.5.

(P_u,P_p) $N{\times}N$	(2,1)	(3,2)	(4,3)
4×4	124	324	620
8×8	532	1348	2548
16×16	2212	5508	10340
32×32	9028	22276	41668
64×64	36484	89604	167300

stable Taylor-Hood elements gives optimal convergence. In figures 3.5 and 3.6 we verify optimal convergence numerically though only in the sense of the L^2-norm for both velocity and pressure. Because of the awful expressions of the velocity reference solution (3.33) when measured in the H^1-norm, this is done for mere simplicity.

The polynomial degrees P_u and P_p for the spline basis functions used to compute the convergence in figures 3.5 and 3.6 together with the grid size and the resulting degrees of freedom for both velocity and pressure is shown in Tables 3.1 and 3.2.

Table 3.2: M_h degrees of freedom for the pressure field for different polynomial degrees and grid partitioning used in the convergence plot 3.6.

(P_u,P_p) $N{\times}N$	(2,1)	(3,2)	(4,3)
4×4	25	81	169
8×8	81	289	625
16×16	289	1089	2401
32×32	1089	4225	9409
64×64	4225	16641	37249

3.3 Example 2: Stokes Flow around a NACA airfoil

We look at a more relevant example, namely Stokes flow around a NACA airfoil. Flow simulations around objects of technical interest are under rapid evolution. Stokes flow around a NACA airfoil is perhaps of little interest as when simulating flow around airfoils the transition from laminar to turbulent flow is an important part of quantifying the performance of the airfoil measured by e.g. the lift and drag [26]. As laminar and turbulent flows means high Reynolds numbers, flow simulations must be done using the Navier-Stokes equations, not the steady Stokes equations. The Navier-Stokes equations are beyond the scope of this thesis, simulations around a NACA airfoil using a Navier-Stokes solver can be found in e.g. [46].

Nevertheless, we will study a steady Stokes flow around a NACA airfoil. The airfoil in this example is of length 1 unit, where the length is measured as the length of the straight line from the leading edge to the trailing edge of the airfoil. In this example the camber is 4% of the chord, and the camber position is 60% of the chord. The thickness of the airfoil is 20% of the chord. Hence the name of this airfoil is NACA 4620, see [3] for more information on the NACA airfoil and the equations used to construct it.

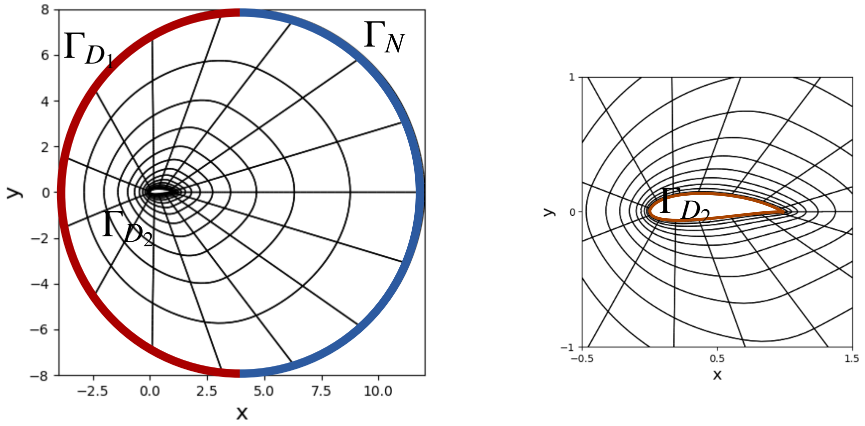
We let the inflow struck the airfoil parallel with the chord choosing the Dirichlet boundary Γ_{D_1} as in Figure 3.7. On the Dirichlet boundary Γ_{D_2} a no-slip boundary condition is introduced. Let there be no forces on the rest of the boundary introducing a homogeneous Neumann boundary Γ_N. This problem is modelled by the steady Stokes equations (3.2). Let $\mathbf{g}$ be the inflow velocity on the Dirichlet boundary Γ_{D_1}. As there is no sinks or sources $\mathbf{f} = \mathbf{0}$ and the homogeneous Neumann condtion is $\mathbf{h} = \mathbf{0}$. Summarized the steady Stokes problem for this particular example reads

$$\begin{aligned} -\nu\nabla^2\mathbf{u} + \nabla p &= 0 && \text{in } \Omega, \\ \operatorname{div} \mathbf{u} &= 0 && \text{in } \Omega, \\ \mathbf{u} &= \mathbf{g} && \text{on } \Gamma_{D_1}, \\ \mathbf{u} &= \mathbf{0} && \text{on } \Gamma_{D_2}, \\ -p\hat{\mathbf{n}} + \nu(\nabla\mathbf{u})\hat{\mathbf{n}} &= \mathbf{0} && \text{on } \Gamma_N, \end{aligned} \tag{3.42}$$

In the introduction to this chapter the Reynolds number was given as $Re = \frac{Ul}{\nu}$, where U is a characteristic velocity of the problem, l is a characteristic length and ν is the kinematic viscosity. For the NACA airfoil the characteristic velocity is the inflow velocity $\mathbf{g}$ at the Dirichlet boundary Γ_{D_1} The characteristic length is the chord parallel to the stream, which is the length of the straight line from the leading edge to the trailing edge of the NACA airfoil. Without loss of generality, we choose $\nu = 1$ and $\mathbf{g} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ which gives a characteristic velocity of $U = 1$. As the characteristic length of the airfoil is the same as the chord length, which is 1 unit, the Reynolds number is Re = 1.

Upon inserting $\mathbf{f} = \mathbf{0}$ and $\mathbf{h} = \mathbf{0}$ into equation (3.11) and $\nu = 1$ into (3.8) we get the weak formulation of problem (3.42) from equation (3.15). Choose the velocity space $\mathbb{V} = [H_0^1(\Omega)]^2$ and the pressure space $\mathbb{Q} = L_0^2(\Omega)$. The weak formulation reads:
Find $(\mathbf{u}, p) \in \mathbb{V} \times \mathbb{Q}$ such that

$$\begin{aligned} a(\mathbf{u}, \mathbf{v}) + b(\mathbf{v}, p) &= -a(\mathbf{r_g}, \mathbf{v}), \quad \forall \mathbf{v} \in \mathbb{V} \\ b(\mathbf{u}, q) &= -b(\mathbf{r_g}, q) \quad \forall q \in \mathbb{Q}, \end{aligned} \tag{3.43}$$



(b) Zoomed in on NACA airfoil.

(a) Full view of Ω.

Figure 3.7: NACA airfoil on a 15 × 15 grid displayed with its boundary $\partial\Omega = \Gamma_{D_1} \bigcup \Gamma_{D_2} \bigcup \Gamma_N$.

where $\mathbf{r_g} \in H^1(\Omega)$ is a lifting function such that $\mathbf{r_g}|_{\Gamma_{D_1}=1} \wedge \mathbf{r_g}|_{\Gamma_{D_2}} = 0$. Then we obtain the solution of the problem (3.42) as $(\mathbf{u} + \mathbf{r_g}, p) \in H^1(\Omega) \times L^2(\Omega)$.

3.3.1 Galerkin high-fidelity approximation

To obtain numerical stable solutions, we choose Taylor-Hood finite element spaces as the approximation spaces for both the velocity field and the pressure field. Therefore we choose the discrete spaces $\mathbb{V}_h = \{\mathbf{v} : \mathbf{v} \circ \Phi \in \mathbb{S}^{2,2}_{0,0} \times \mathbb{S}^{2,2}_{0,0}\}$ and $\mathbb{Q}_h = \{q : q \circ \Phi \in \mathbb{S}^{1,1}_{0,0}\}$ to approximate the velocity space and pressure space respectively. The map Φ is the map from the parameter space to the original space defined in Section 2.2.4. We denote by $\mathbf{u_h}$ and p_h the Galerkin approximation of the velocity function $\mathbf{u}$ and the pressure function p respectively. Introducing the discrete approximation of the lifting function as $\mathbf{u_g}$ and inserting this together with the high-fidelity approximations $(\mathbf{u_h}, p_h)$ into the weak formulation (3.43), we derive the Galerkin formulation of problem (3.42) as find $(\mathbf{u}_h, p_h) \in \mathbb{V}_h \times \mathbb{Q}_h$ such that

$$\begin{aligned} a(\mathbf{u_h}, \mathbf{v_h}) + b(\mathbf{v_h}, p_h) &= -a(\mathbf{u_g}, \mathbf{v_h}), \quad \forall \mathbf{v_h} \in \mathbb{V}_h \\ b(\mathbf{u_h}, q_h) &= -b(\mathbf{u_g}, q_h) \quad \forall q_h \in \mathbb{Q}_h. \end{aligned} \tag{3.44}$$

Let $\{\boldsymbol{\phi_i}\}_{i=1}^{N_h}$ denote a basis for $\mathbb{V}_h$ and let $\{\psi_i\}_{i=1}^{M_h}$ denote a basis for $\mathbb{Q}_h$. Then we can express the high-fidelity solutions $(\mathbf{u_h}, p_h)$ as $\mathbf{u_h} = \sum_{j=1}^{N_h} w_h^{(j)} \boldsymbol{\phi_j}$, and $p_h = \sum_{k=1}^{M_h} p_h^{(k)} \psi_k$, where the vectors $\mathbf{w_h}$ and $\mathbf{p_h}$ holds the unknown coefficients $w_h^{(j)}$ and $p_h^{(i)}$ associated with the degrees of freedom of the high-fidelity solutions. The same holds for the lifting function which can be expressed as $\mathbf{u_g} = \sum_{j=1}^{N_h} w_g^{(j)} \boldsymbol{\phi_j}$ with the coefficent vector $\mathbf{w_g}$ holding the coefficients $w_g^{(j)}$.

If we proceed in the same manner as we did deriving the linear systems (3.20) and (3.21) we derive from equations (3.44) the linear system

$$\begin{bmatrix} A_h & B_h^\top \\ B_h & 0 \end{bmatrix} \begin{bmatrix} \mathbf{w_h} \\ \mathbf{p_h} \end{bmatrix} = \begin{bmatrix} \mathbf{f_{h1}} \\ \mathbf{f_{h2}} \end{bmatrix}, \tag{3.45}$$

where the elements of matrices A_h and B_h are given as

$$(A_h)_{ij} = a(\boldsymbol{\phi_j}, \boldsymbol{\phi_i}) = \int_\Omega \nabla\boldsymbol{\phi_j} : \nabla\boldsymbol{\phi_i} \,\mathrm{d}\Omega, \quad \mathrm{1 \leq i, j \leq N_h}.$$

$$(B_h)_{il} = b(\boldsymbol{\phi_l}, \psi_i) = -\int_\Omega \psi_i \nabla \cdot \boldsymbol{\phi_l} \,\mathrm{d}\Omega, \quad \mathrm{1 \leq l \leq N_h,\ 1 \leq i \leq M_h}.$$

The elements of the vectors $\mathbf{f_{h1}}$ and $\mathbf{f_{h2}}$ are

$$(\mathbf{f_{h1}})_i = -(A_h \mathbf{w_g})_i, \quad 1 \leq i \leq N_h.$$

$$(\mathbf{f_{h2}})_j = -(B_h \mathbf{w_g})_j, \quad 1 \leq j \leq M_h.$$

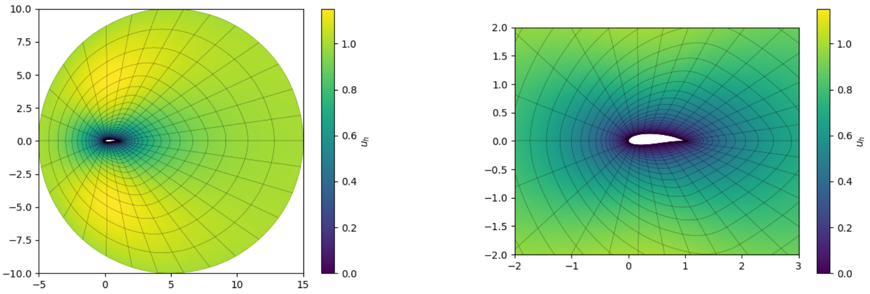
Then the high-fidelity of the steady Stokes problem is obtained as

$$(\sum_{i=1}^{N_h} (w_h^{(i)} + w_g^{(i)}) \boldsymbol{\phi_i}, \sum_{j=1}^{M_h} p_h^{(j)} \psi_j)$$

.

3.3.2 Solutions and relative error

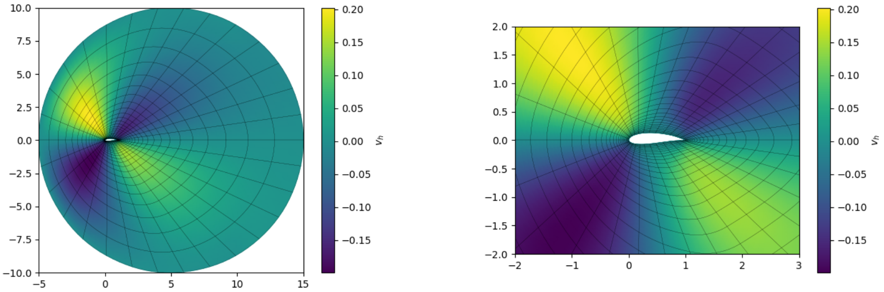
The linear system (3.45) is solved on a domain of 30 × 30 elements and with the given polynomial spaces for the velocity basis and the pressure basis, this gives $N_h = 2 \cdot 3540$ degrees of freedom for the velocity basis and $M_h = 930$ degrees of freedom for the pressure basis. A high-fidelity velocity solution of problem (3.42) is found in figures 3.8 and 3.9. In Figure 3.10 the velocity in both directions are illustrated as streamlines instead, and the pressure is represented as the color map. We observe that the pressure is high in front of the airfoil and low at the back, which seems reasonable.



(a) NACA airfoil from far out.

(b) NACA airfoil close up.

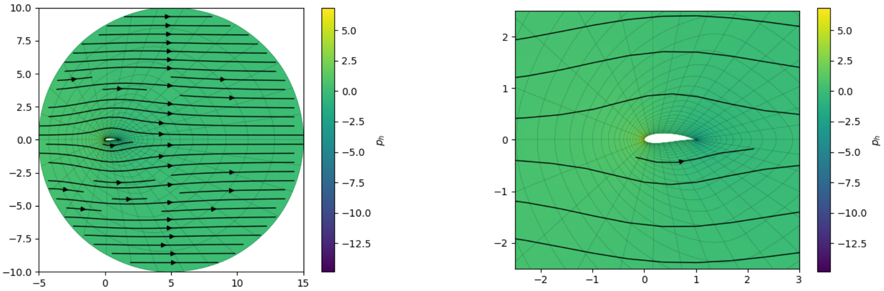
Figure 3.8: Galerkin approximation of velocity in x-direction , u_h, around a NACA airfoil. Solved on a 30 × 30 grid.



(a) NACA airfoil from far out.

(b) NACA airfoil close up.

Figure 3.9: Galerkin approximation of velocity in y-direction , v_h, around a NACA airfoil. Solved on a 30×30 grid.



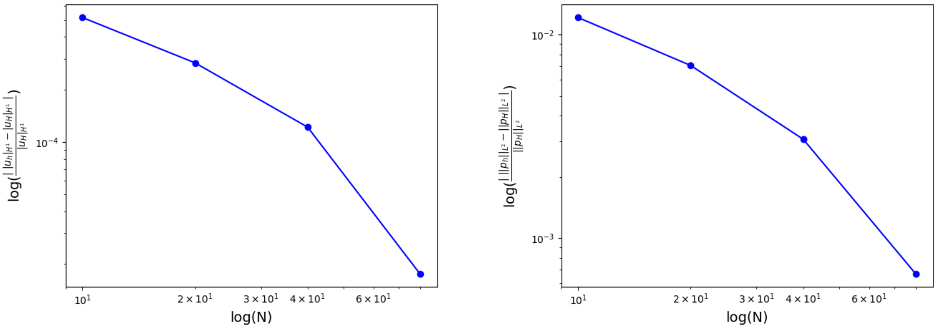
(a) NACA airfoil from far out.

(b) NACA airfoil close up.

Figure 3.10: Galerkin approximation of velocity field $\mathbf{u_h}$ and pressure p_h around NACA airfoil. Velocity field displayed as streamlines and pressure as colormap. Solved on a 30×30 grid.

In the previous example 3.2 we could verify the logic of the finite element solver comparing the numerical solutions to the analytic reference solution. As we stand without a reference solution for this example, we should still make some kind of verification of the solver. We will use a method based on relative error. The idea is that the high-fidelity solutions approximate the exact solutions with higher accuracy for finer grid. Then computing a high-fidelity solution for a very fine grid we assume this solution to be relatively close to the exact solution. If we then try to approximate this finest level solution with solutions on finer and finer grids, but still coarser grids than the finest level, the solutions for the coarser grids should converge towards the finest level solution.

Let us denote the solutions on the finest level grid by $(\mathbf{u_H}, p_H)$. We measure the relative error between the norms of the finest level solutions and the approximation on a coarser grid relative to the norm of the finest level solutions. The velocity is measured in the H^1-seminorm and the pressure is measured in the L^2-norm. Then the relative error between a



(a) Relative error velocity field, polynomial degree $P_u = 2$.

(b) Relative error pressure field, polynomial degree $P_u = 2$.

Figure 3.11: Error between norms of the high-fidelity solutions $(\mathbf{u_h}, p_h)$ and the norms of the finest level high-fidelity solutions $(\mathbf{u_H}, p_H)$ relative to the norm of the finest level high-fidelity solutions. Measured for finer and finer grids $N \times N$. The finest level high-fidelity solutions are computed on the finest grid 320×320.

high-fidelity solution $(\mathbf{u_h}, p_h)$ and the finest level solution $(\mathbf{u_H}, p_H)$ is

$$\frac{|\ |\mathbf{u_H}|_{H^1} - |\mathbf{u_h}|_{H^1}\ |}{|\mathbf{u_H}|_{H^1}} \qquad \frac{|\ \|p_H\|_{L^2} - \|p_h\|_{L^2}\ |}{\|p_H\|_{L^2}}. \tag{3.46}$$

As the finest level grid we choose a grid of 320×320 elements, which gives $N_h = 2 \cdot 409203$ degrees of freedom for the velocity field and $M_h = 102720$ degrees of freedom for the pressure field. The result of computing the relative error for increasing grid sizes $N \times N$ is shown in Figure 3.11.

Chapter 4

RB solutions of the Stokes Equations

The reduced basis solver needs to be built upon a finite element solver approximating the exact solution with a high enough accuracy. The key to a reduced basis solver is to formulate the problem in such a way that all components can be written as affine combinations. As discussed in [24] and also in Section 2.5.4, the affine combinations is what makes the offline-online decomposition possible and reduced basis solvers attractive. The offline-online decomposition can also be done for non-affine parametric dependence, see e.g. [51]

As discussed in Section 2.5, it is required that the reduced basis solutions approximate the exact solution within a desired accuracy. For the steady Stokes solutions $(\mathbf{u}(\boldsymbol{\mu}), p_h(\boldsymbol{\mu})) \in \mathbb{V} \times \mathbb{Q}$ where the velocity space is the Hilbert space $\mathbb{V} = [H^1(\Omega)]^2$ and the pressure space is the space $L^2(\Omega)$, this accuracy is measured as

$$\begin{aligned} |\mathbf{u}(\boldsymbol{\mu}) - \mathbf{u_N}(\boldsymbol{\mu})|_{H^1(\Omega)} + \|p(\boldsymbol{\mu}) - p_N(\boldsymbol{\mu})\|_{L^2(\Omega)} \leq |\mathbf{u}(\boldsymbol{\mu}) - \mathbf{u_h}(\boldsymbol{\mu})|_{H^1(\Omega)} \\ + \|p(\boldsymbol{\mu}) - p_h(\boldsymbol{\mu})\|_{L^2(\Omega)} + |\mathbf{u_h}(\boldsymbol{\mu}) - \mathbf{u_N}(\boldsymbol{\mu})|_{H^1(\Omega)} + \|p_h(\boldsymbol{\mu}) - p_N(\boldsymbol{\mu})\|_{L^2(\Omega)} \leq \\ |\mathbf{u_h}(\boldsymbol{\mu}) - \mathbf{u_N}(\boldsymbol{\mu})|_{H^1(\Omega)} + \|p_h(\boldsymbol{\mu}) - p_N(\boldsymbol{\mu})\|_{L^2(\Omega)} + \varepsilon. \end{aligned} \tag{4.1}$$

In equation (4.1) we require the high-fidelity solutions $(\mathbf{u_h}(\boldsymbol{\mu}), p_h(\boldsymbol{\mu}))$ to approximate the exact solutions $(\mathbf{u}(\boldsymbol{\mu}), p(\boldsymbol{\mu}))$ within a desired accuracy ε. That is

$$|\mathbf{u}(\boldsymbol{\mu}) - \mathbf{u_h}(\boldsymbol{\mu})|_{H^1(\Omega)} + \|p(\boldsymbol{\mu}) - p_h(\boldsymbol{\mu})\|_{L^2(\Omega)} \leq \varepsilon \tag{4.2}$$

4.1 A parametrized Galerkin formulation

Let us now consider the problem (3.2) upon a parameter-dependent domain $\tilde{\Omega}(\boldsymbol{\mu}) \in \mathbb{R}^2$. Let us put the parameters in a vector, and for now we will denote the parameter vector by $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_p)^\top \in \mathbb{P}$, where $\mathbb{P}$ denotes the parameter set. Let the parameter-dependent domain be such that it is obtained through a mapping $\boldsymbol{\mathcal{F}}$ from a parameter-independent domain Ω, i.e. $\tilde{\mathbf{x}} = \boldsymbol{\mathcal{F}}(\mathbf{x};\boldsymbol{\mu_g})$ which yields

$$\tilde{\Omega}(\boldsymbol{\mu}) = \boldsymbol{\mathcal{F}}(\Omega;\boldsymbol{\mu_g}) \qquad \forall \boldsymbol{\mu_g} \in \mathbb{P}.$$

Here $\boldsymbol{\mu_g}$ denotes the parameters that the domain depends on, called geometric parameters. The solution now depends on geometric parameters $\boldsymbol{\mu_g}$, and we will also vary certain physical parameters making the solution depend on some physical parameters $\boldsymbol{\mu_{ph}}$. We will from now on denote the parameters by $\boldsymbol{\mu}$, and it should be clear from the context what parameter dependency is needed, physical, geometric or both.

The solution of the parameter-dependent problem will be given as $(\tilde{\mathbf{u}}(\boldsymbol{\mu}), \tilde{p}(\boldsymbol{\mu})) = (\tilde{\mathbf{u}}(\tilde{\mathbf{x}};\boldsymbol{\mu}), \tilde{p}(\tilde{\mathbf{x}};\boldsymbol{\mu}))$. Then the weak formulation of the parameter-dependent version of problem (3.2) reads: find $(\tilde{\mathbf{u}}(\boldsymbol{\mu}), \tilde{p}(\boldsymbol{\mu})) \in \tilde{\mathbb{V}}(\boldsymbol{\mu}) \times \tilde{\mathbb{Q}}(\boldsymbol{\mu})$ such that

$$\begin{aligned} \tilde{a}(\tilde{\mathbf{u}}(\boldsymbol{\mu}), \tilde{\mathbf{v}};\boldsymbol{\mu}) + \tilde{b}(\tilde{\mathbf{v}}, \tilde{p}(\boldsymbol{\mu});\boldsymbol{\mu}) &= \tilde{f}_1(\tilde{\mathbf{v}};\boldsymbol{\mu}), \quad \forall \tilde{\mathbf{v}} \in \tilde{\mathbb{V}}(\boldsymbol{\mu}) \\ \tilde{b}(\tilde{\mathbf{u}}(\boldsymbol{\mu}), \tilde{q};\boldsymbol{\mu}) &= \tilde{f}_2(\tilde{q};\boldsymbol{\mu}) \quad \forall \tilde{q} \in \tilde{\mathbb{Q}}(\boldsymbol{\mu}). \end{aligned} \tag{4.3}$$

The bilinear forms $\tilde{a}(\cdot,\cdot;\boldsymbol{\mu})$ and $\tilde{b}(\cdot,\cdot;\boldsymbol{\mu})$ are the same as in equation (3.8) and (3.9) and the linear forms $\tilde{f}_1(\cdot;\boldsymbol{\mu})$ and $\tilde{f}_2(\cdot;\boldsymbol{\mu})$ are the same as in equation (3.11) and (3.14).

By means of the inverse mapping $\boldsymbol{\mathcal{F}}^{-1}(\cdot;\boldsymbol{\mu})$, we want to pull the problem set on the parameter-dependent domain $\tilde{\Omega}(\boldsymbol{\mu})$ back to the parameter-independent domain Ω. In order for us to do so, we recall from Section 2.2.5 the Jacobian matrix $\mathbb{J}_{\boldsymbol{\mathcal{F}}}(\mathbf{x};\boldsymbol{\mu}) \in \mathbb{R}^{2\times 2}$ of the map $\boldsymbol{\mathcal{F}}(\cdot;\boldsymbol{\mu})$

$$(\mathbb{J}_{\boldsymbol{\mathcal{F}}}(\mathbf{x};\boldsymbol{\mu}))_{ij} = \frac{\partial(\tilde{\mathbf{x}})_i}{\partial(\mathbf{x})_j}(\mathbf{x}) = \frac{\partial(\boldsymbol{\mathcal{F}}(\mathbf{x};\boldsymbol{\mu}))_i}{\partial(\mathbf{x})_j}(\mathbf{x}).$$

Then we recall from Section 2.2.6 the change of variable formula for any integrable function $\tilde{f} : \tilde{\Omega}(\boldsymbol{\mu}) \to \mathbb{R}$

$$\int_{\tilde{\Omega}(\boldsymbol{\mu})} \tilde{f}(\tilde{\mathbf{x}}) \, \mathrm{d}\tilde{\Omega} = \int_{\Omega} f(\mathbf{x}) |\mathbb{J}_{\boldsymbol{\mathcal{F}}}(\mathbf{x};\boldsymbol{\mu})| \, \mathrm{d}\Omega,$$

where $|\mathbb{J}_{\boldsymbol{\mathcal{F}}}(\mathbf{x};\boldsymbol{\mu})|$ is the determinant of the Jacobian matrix and $f = \tilde{f} \circ \boldsymbol{\mathcal{F}}$. From the same section we also recall the change of variable for derivatives

$$\tilde{\nabla} \tilde{f}(\tilde{\mathbf{x}}) = (\mathbb{J}_{\boldsymbol{\mathcal{F}}}(\mathbf{x};\boldsymbol{\mu}))^{-\top} \nabla f(\mathbf{x}).$$

By this we are able to express the Galerkin formulation of the weak problem (4.3) on the parameter-independent domain Ω as:
find $(\mathbf{u_h}(\boldsymbol{\mu}), p_h(\boldsymbol{\mu})) = (\mathbf{u_h}(\mathbf{x};\boldsymbol{\mu}), p_h(\mathbf{x};\boldsymbol{\mu})) \in \mathbb{V}_h \times \mathbb{Q}_h$ such that

$$\begin{aligned} a(\mathbf{u_h}(\boldsymbol{\mu}), \mathbf{v_h};\boldsymbol{\mu}) + b(\mathbf{v_h}, p_h(\boldsymbol{\mu});\boldsymbol{\mu}) &= f_1(\mathbf{v_h};\boldsymbol{\mu}), \quad \forall \mathbf{v_h} \in \mathbb{V}_h \\ b(\mathbf{u_h}(\boldsymbol{\mu}), q_h;\boldsymbol{\mu}) &= f_2(q_h;\boldsymbol{\mu}) \quad \forall q_h \in \mathbb{Q}_h. \end{aligned} \tag{4.4}$$

All forms are derived from the weak formulation (4.3). The derivation is shown in the following.

$$\begin{gathered}\tilde{a}(\tilde{\mathbf{u}}_{\mathbf{h}}(\boldsymbol{\mu}), \tilde{\mathbf{v}}_{\mathbf{h}}; \boldsymbol{\mu}) = \int_{\tilde{\Omega}(\boldsymbol{\mu})} \nu \tilde{\nabla} \tilde{\mathbf{u}}_{\mathbf{h}}(\boldsymbol{\mu}) : \tilde{\nabla} \tilde{\mathbf{v}}_{\mathbf{h}} \, \mathrm{d}\Omega = \\ \int_{\Omega} \nu \left((\mathbb{J}_{\mathcal{F}}(\mathbf{x}; \boldsymbol{\mu}))^{-\top} \nabla \mathbf{u}_{\mathbf{h}}(\boldsymbol{\mu}) \right) : ((\mathbb{J}_{\mathcal{F}}(\mathbf{x}; \boldsymbol{\mu}))^{-\top} \nabla \mathbf{v}_{\mathbf{h}}) |\mathbb{J}_{\mathcal{F}}(\mathbf{x}; \boldsymbol{\mu})| \, \mathrm{d}\Omega \\ =: a(\mathbf{u}_{\mathbf{h}}(\boldsymbol{\mu}), \mathbf{v}_{\mathbf{h}}; \boldsymbol{\mu}),\end{gathered} \tag{4.5}$$

here $\mathbf{u}_{\mathbf{h}}(\boldsymbol{\mu}) = \mathbf{u}_{\mathbf{h}}(\mathbf{x}; \boldsymbol{\mu}) = \tilde{\mathbf{u}}_{\mathbf{h}}(\tilde{\mathbf{x}}; (\boldsymbol{\mu})) = \tilde{\mathbf{u}}_{\mathbf{h}}(\mathcal{F}(\mathbf{x}; \boldsymbol{\mu}); (\boldsymbol{\mu}))$, which for simplicity we will write as $\mathbf{u}_{\mathbf{h}}(\mathbf{x}) = \tilde{\mathbf{u}}_{\mathbf{h}} \circ \mathcal{F}(\mathbf{x})$ as the mere point is to show that $\mathbf{u}_{\mathbf{h}}(\boldsymbol{\mu})$ and $\tilde{\mathbf{u}}_{\mathbf{h}}(\boldsymbol{\mu})$ are the same function expressed in parameter-independent coordinates $\mathbf{x}$ and parameter-dependent coordinates $\tilde{\mathbf{x}}$ respectively. Similarily we have that $\mathbf{v}_{\mathbf{h}}(\mathbf{x}) = \tilde{\mathbf{v}}_{\mathbf{h}} \circ \mathcal{F}(\mathbf{x})$. Note that ∇ and $\tilde{\nabla}$ are the derivatives with respect to parameter-independent coordinates $\mathbf{x}$ and parameter-dependent coordinates $\tilde{\mathbf{x}}$ respectively.

$$\begin{gathered}\tilde{b}(\tilde{\mathbf{v}}_{\mathbf{h}}, \tilde{p}_h(\boldsymbol{\mu}); \boldsymbol{\mu}) = -\int_{\tilde{\Omega}(\boldsymbol{\mu})} \tilde{p}_h(\boldsymbol{\mu}) \tilde{\nabla} \cdot \tilde{\mathbf{v}}_{\mathbf{h}} \, \mathrm{d}\Omega = \\ -\int_{\Omega} p_h(\boldsymbol{\mu}) \left((\mathbb{J}_{\mathcal{F}}(\mathbf{x}; \boldsymbol{\mu}))^{-\top} \nabla \right) \cdot \mathbf{v}_{\mathbf{h}} |\mathbb{J}_{\mathcal{F}}(\mathbf{x}; \boldsymbol{\mu})| \, \mathrm{d}\Omega \\ =: b(\mathbf{v}_{\mathbf{h}}, p_h(\boldsymbol{\mu}); \boldsymbol{\mu}),\end{gathered} \tag{4.6}$$

in the same fashion as for equation (4.5) we have $p_h(\mathbf{x}) = \tilde{p}_h \circ \mathcal{F}(\mathbf{x})$.

$$\begin{gathered}\tilde{\mathrm{f}}_1(\tilde{\mathbf{v}}_{\mathbf{h}}; \boldsymbol{\mu}) = \int_{\tilde{\Omega}(\boldsymbol{\mu})} \tilde{\mathbf{f}}(\boldsymbol{\mu}) \cdot \tilde{\mathbf{v}}_{\mathbf{h}} \, \mathrm{d}\tilde{\Omega} + \int_{\tilde{\Gamma}_{\mathrm{N}}(\boldsymbol{\mu})} \tilde{\mathbf{h}}(\boldsymbol{\mu}) \cdot \tilde{\mathbf{v}}_{\mathbf{h}} \, \mathrm{d}\tilde{\Gamma} - \int_{\tilde{\Omega}(\boldsymbol{\mu})} \nu \tilde{\nabla} \tilde{\mathbf{u}}_{\mathbf{g}}(\boldsymbol{\mu}) : \tilde{\nabla} \tilde{\mathbf{v}}_{\mathbf{h}} \, \mathrm{d}\tilde{\Omega} = \\ \int_{\Omega} \mathbf{f}(\boldsymbol{\mu}) \cdot \mathbf{v}_{\mathbf{h}} |\mathbb{J}_{\mathcal{F}}(\mathbf{x}; \boldsymbol{\mu})| \, \mathrm{d}\Omega + \int_{\Gamma_{\mathrm{N}}} \mathbf{h}(\boldsymbol{\mu}) \cdot \mathbf{v}_{\mathbf{h}} |\mathbb{J}_{\mathcal{F}}(\mathbf{x}; \boldsymbol{\mu}) \hat{\mathbf{t}}| \, \mathrm{d}\Gamma - \mathrm{a}(\mathbf{u}_{\mathbf{g}}(\boldsymbol{\mu}), \mathbf{v}_{\mathbf{h}}; \boldsymbol{\mu}) =: \mathrm{f}_1(\mathbf{v}_{\mathbf{h}}; \boldsymbol{\mu})\end{gathered} \tag{4.7}$$

$$\begin{gathered}\tilde{f}_2(\tilde{q}_h; \boldsymbol{\mu}) = -\tilde{b}(\tilde{\mathbf{u}}_{\mathbf{g}}(\boldsymbol{\mu}), \tilde{q}_h; \boldsymbol{\mu}) = \\ -b(\mathbf{u}_{\mathbf{g}}(\boldsymbol{\mu}), q_h; \boldsymbol{\mu}) =: f_2(q_h; \boldsymbol{\mu}).\end{gathered} \tag{4.8}$$

In the same manner as we did for the linear system (3.29), we can derive a parameter-dependent linear system by inserting

$$\mathbf{u}_{\mathbf{h}}(\boldsymbol{\mu}) = \sum_{j=1}^{N_h} w_h(\boldsymbol{\mu})^{(j)} \boldsymbol{\phi}_{\boldsymbol{j}} \quad \text{and} \quad p_h(\boldsymbol{\mu}) = \sum_{k=1}^{M_h} p_h(\boldsymbol{\mu})^{(k)} \psi_k$$

into the Galerkin weak formulation (4.4)

$$\underbrace{\begin{bmatrix} A_h(\boldsymbol{\mu}) & B_h^{\top}(\boldsymbol{\mu}) \\ B_h(\boldsymbol{\mu}) & 0 \end{bmatrix}}_{K_h(\boldsymbol{\mu})} \underbrace{\begin{bmatrix} \mathbf{w}_{\mathbf{h}}(\boldsymbol{\mu}) \\ \mathbf{p}_{\mathbf{h}}(\boldsymbol{\mu}) \end{bmatrix}}_{\mathbf{U}_{\mathbf{h}}(\boldsymbol{\mu})} = \underbrace{\begin{bmatrix} \mathbf{f}_{\mathbf{h1}}(\boldsymbol{\mu}) \\ \mathbf{f}_{\mathbf{h2}}(\boldsymbol{\mu}) \end{bmatrix}}_{\mathbf{F}_{\mathbf{h}}(\boldsymbol{\mu})}. \tag{4.9}$$

Note that by expressing the lifting function $\mathbf{u_g}(\boldsymbol{\mu})$ in the same manner as done earlier we get $\mathbf{u_g}(\boldsymbol{\mu}) = \sum_{i=1}^{N_h} w_g(\boldsymbol{\mu})^{(i)}\boldsymbol{\phi_i}$. The elements of system (4.9) are then given in the following:

$$(A_h(\boldsymbol{\mu}))_{ij} = a(\boldsymbol{\phi_j}, \boldsymbol{\phi_i}; \boldsymbol{\mu}) \quad 1 \leq i, j \leq N_h, \tag{4.10}$$

where $a(\cdot, \cdot; \boldsymbol{\mu})$ is the same bilinear form as in equation (4.5).

$$(B_h(\boldsymbol{\mu}))_{ij} = b(\boldsymbol{\phi_j}, \psi_i; \boldsymbol{\mu}) \quad 1 \leq i \leq M_h,\ 1 \leq j \leq N_h, \tag{4.11}$$

where $b(\cdot, \cdot; \boldsymbol{\mu})$ is the same bilinear form as in equation (4.6).

$$(\mathbf{f_{h1}}(\boldsymbol{\mu}))_i = f_1(\boldsymbol{\phi_i}; \boldsymbol{\mu}) \quad 1 \leq i \leq N_h, \tag{4.12}$$

where $f_1(\cdot; \boldsymbol{\mu})$ is the same linear form as in equation (4.7).

$$(\mathbf{f_{h2}}(\boldsymbol{\mu}))_i = f_2(\psi_i; \boldsymbol{\mu}) \quad 1 \leq i \leq M_h, \tag{4.13}$$

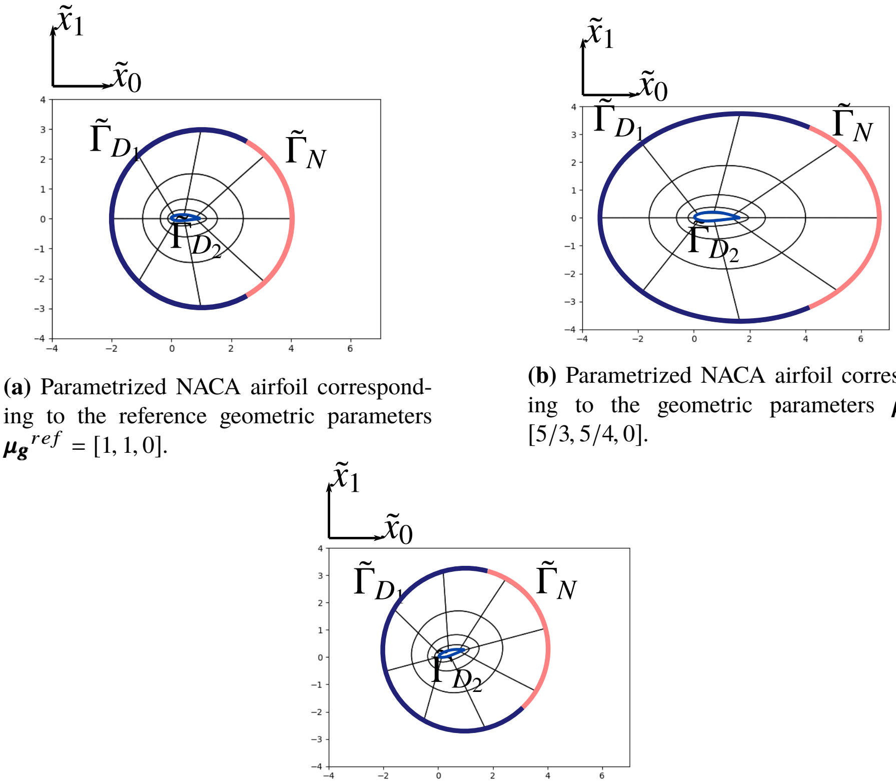
where $f_2(\cdot; \boldsymbol{\mu})$ is the same linear form as in equation (4.8).

4.2 Example 3: NACA airfoil revisited

Now we revisit example 3.3 where we studied a Stokes flow around a NACA airfoil. It could be interesting to study the flow around this NACA airfoil if we changed the shape of it. More precisely we want to stretch and compress the airfoil in both x- and y-direction. We name the scaling factors for x- and y-direction by μ_1 and μ_2 respectively. Also we rotate the airfoil by an angle μ_3. Figure 4.1 shows different geometry configurations. More details on how we do this scaling and rotation is presented in the next Section 4.2.1. Having introduced a parameter-dependent domain $\tilde{\Omega}(\boldsymbol{\mu_g})$ depending on the geometric parameters $\boldsymbol{\mu_g} = [\mu_1, \mu_2, \mu_3]^\top$ we can express the steady Stokes problem from the previous example 3.3 on this parameter-dependent domain. Note that we will use the notation $\tilde{\Omega}(\boldsymbol{\mu})$ instead of $\tilde{\Omega}(\boldsymbol{\mu_g})$ as discussed in the introduction to Chapter 2. The tilde notation is used to illustrate that the problem is set on a parameter-dependent domain and the problem reads:

$$\begin{aligned}
-\mu_4 \tilde{\nabla}^2 \tilde{\mathbf{u}}(\boldsymbol{\mu}) + \tilde{\nabla}\tilde{p}(\boldsymbol{\mu}) &= 0 && \text{in } \tilde{\Omega}(\boldsymbol{\mu}),\\
\operatorname{div} \tilde{\mathbf{u}}(\boldsymbol{\mu}) &= 0 && \text{in } \tilde{\Omega}(\boldsymbol{\mu}),\\
\tilde{\mathbf{u}}(\boldsymbol{\mu}) &= \mu_5 \tilde{\mathbf{g}} && \text{on } \tilde{\Gamma}_{D_1}(\boldsymbol{\mu}),\\
\tilde{\mathbf{u}}(\boldsymbol{\mu}) &= \mathbf{0} && \text{on } \tilde{\Gamma}_{D_2}(\boldsymbol{\mu}),\\
-\tilde{p}(\boldsymbol{\mu})\hat{\mathbf{n}} + \nu(\tilde{\nabla}\tilde{\mathbf{u}})(\boldsymbol{\mu})\hat{\mathbf{n}} &= \mathbf{0} && \text{on } \tilde{\Gamma}_N(\boldsymbol{\mu}),
\end{aligned} \tag{4.14}$$

where $\tilde{\mathbf{g}}$ is some function such that $\tilde{\mathbf{g}}|_{\tilde{\Gamma}_{D_1}(\boldsymbol{\mu})} = 1$. In addition to parametrizing the computational domain we have introduced the parameter μ_4 representing the kinematic viscosity ν we used in the previous example. Also we have parametrized the velocity on the inflow boundary $\tilde{\Gamma}_{D_1}(\boldsymbol{\mu})$ by choosing the value μ_5 for the inflow velocity.



(a) Parametrized NACA airfoil corresponding to the reference geometric parameters $\boldsymbol{\mu_g}^{ref} = [1, 1, 0]$.

(b) Parametrized NACA airfoil corresponding to the geometric parameters $\boldsymbol{\mu_g} = [5/3, 5/4, 0]$.

(c) Parametrized NACA airfoil corresponding to the geometric parameters $\boldsymbol{\mu_g} = [1, 1, \pi/12]$.

Figure 4.1: The NACA airfoil computational domain $\tilde{\Omega}$ corresponding to different geometric parameters.

4.2.1 A parametric map

We will vary the shape of the NACA airfoil by stretching or compressing it in either the $\tilde{x}$-direction or in the $\tilde{y}$-direction. Also we want the possibility to rotate it. Let $\mathbf{x}$ be the coordinates of the parameter-independent domain we studied in example 3.3. Then we can represent the rotation by μ_3 radians and the $(\tilde{x}, \tilde{y})$-scaling with scaling factors (μ_1, μ_2) by the map

$$\begin{bmatrix} \tilde{x}_0 \\ \tilde{x}_1 \end{bmatrix} = \begin{bmatrix} \cos(\mu_3) & -\sin(\mu_3) \\ \sin(\mu_3) & \cos(\mu_3) \end{bmatrix} \begin{bmatrix} \mu_1 & 0 \\ 0 & \mu_2 \end{bmatrix} = \begin{bmatrix} \mu_1 \cos(\mu_3) & -\mu_2 \sin(\mu_3) \\ \mu_1 \sin(\mu_3) & \mu_2 \cos(\mu_3) \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \end{bmatrix}, \tag{4.15}$$

where μ_1 and μ_2 are the stretching parameters in $x-$ and y-direction respectively and μ_3 is the counterclockwise rotation angle measured in radians. Note that equation (4.15) is nothing but the map $\tilde{\mathbf{x}} = \boldsymbol{\mathcal{F}}(\mathbf{x}; \boldsymbol{\mu})$ from equation (2.20).

Figure 4.1 shows different geometry configurations for different geometry parameter vectors $\boldsymbol{\mu_g}$. The parameter configuration that gives a geometry equivalent to the parameter-independent geometry we call the *reference parameters* and denote the vector holding the reference parameters by $\boldsymbol{\mu_g}^{ref}$. For the NACA airfoil in this example the reference parameter configuration is $\boldsymbol{\mu_g}^{ref} = [1, 1, 0]$, that is scale by one in x- and y-direction and zero rotation, seen in Figure 4.1a. Figure 4.1b shows scaling in x, y-direction and Figure 4.1c shows rotation.

Note that the Dirichlet boundary is not half of the circle as in the previous example. The Dirichlet boundary is fixed in such a way that when rotating the airfoil to its limits either counterclockwise or clockwise at least the left half of the circle will always be Dirichlet. For the example given in this thesis the rotation limits are $[-\pi/6, \pi/6]$. Therefore the Dirichlet boundary should be approximately $\pi + \pi/6 + \pi/6 = 4\pi/3$, that is $2/3$ of the circle encircling the NACA airfoil. When we say approximately it is due to the fact that the circle is not rotated around its center, we rotate in fact the airfoil around its leading edge. The important part is that we have fixed the Dirichlet boundary.

The outer boundary $\Gamma_{D_1]} \bigcup \Gamma_N$ is chosen to be far out from the NACA airfoil such that the simulations close to the NACA airfoil is not affected with the outer boundary of the domain. We are only interested in the behavior of the simulations close to the airfoil and not further out. In Figure 4.1a the radius of the outer circle is $r = 3$. This is due to illustration purposes, the radius for the parameter-independent domain is $r = 10$ for the actual simulations. The map $\mathcal{F}$ stretch the whole domain and not just the NACA airfoil. As the outer boundary is so far away from the airfoil, ideally the stretching of the boundary will not influence the simulations near the airfoil.

As seen from Figure 4.1a, and even more clear in Figure 3.7b, the grid-lines going in circles around the airfoil lie closer and closer near the airfoil than they do further out. The mesh is constructed in this way, refining the grid to have more grid lines nearer the airfoil. The reason for this is that on the trailing edge of the airfoil it has some sharp edges because of the way it is constructed. Then to obtain good enough numerical results in this part of the domain it is important to have a fine enough grid in this region. A full discussion around the mesh quality and design parameters for the meshing is beyond the scope of this thesis. To create the NACA 4-digit airfoil the Splipy package for python is used, see [35] for further details.

The map (4.15) is not optimal. As seen in Figure 4.1b the stretching in both $\tilde{x}-$ and $\tilde{y}-$direction stretches the whole computational domain and not just the airfoil. Figure 4.2 shows the map from the parameter domain to the parameter-independent domain. We see that the η-direction is mapped to the axis going radial out from the airfoil. Hence the ideal map would be

$$\begin{bmatrix} \tilde{x}_0 \\ \tilde{x}_1 \end{bmatrix} = (1-\eta) \begin{bmatrix} \mu_1 \cos(\mu_3) & -\mu_2 \sin(\mu_3) \\ \mu_1 \sin(\mu_3) & \mu_2 \cos(\mu_3) \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \end{bmatrix}. \tag{4.16}$$

With this map the scaling in the $\tilde{x}-$ and $\tilde{y}$-direction have less effect the further you move out from the airfoil and in fact the effect of the scaling would be zero at the circle periphery encircling the computational domain. This mapping yields non-affine terms though and that is why we work with the simplified mapping (4.15). This comes at a cost, we see that the size of the elements increase because of the scaling, just compare the elements of the NACA domain in Figure 4.1a to Figure 4.1b. This must be kept in mind when considering the relative error computed in the previous example 3.3. See Section 4.2.3 for a further discussion.

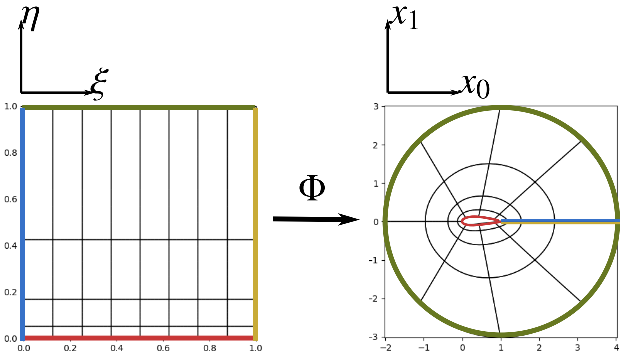


Figure 4.2: Map Φ of parameter space to a parameter idependent NACA airfoil. The domain consists of 4×8 elements. The ξ-direction of the parameter space is mapped to the angular direction in the physical space and the η-direction of the parameter space is mapped to the radial direction in the physical space.

4.2.2 Galerkin high-fidelity approximation

With help from the equations derived in the previous Section 4.1 we can derive a parametrized Galerkin formulation and a parametrized linear system. The Jacobian matrix of the map $\boldsymbol{\mathcal{F}}(\mathbf{x};\boldsymbol{\mu})$ from equation (4.15) is

$$\mathbb{J}_{\boldsymbol{\mathcal{F}}}(\boldsymbol{\mu}) = \mathbb{J}_{\boldsymbol{\mathcal{F}}}(\mathbf{x};\boldsymbol{\mu}) = \begin{bmatrix} \mu_1 \cos(\mu_3) & -\mu_2 \sin(\mu_3) \\ \mu_1 \sin(\mu_3) & \mu_2 \cos(\mu_3) \end{bmatrix}. \qquad (4.17)$$

We can also compute the determinant of the Jacobian matrix

$$|J_{\boldsymbol{\mathcal{F}}}(\boldsymbol{\mu})| = \mu_1\mu_2 \cos^2(\mu_3) + \mu_1\mu_2 \sin^2(\mu_3) = \mu_1\mu_2,$$

and the inverse transpose

$$\begin{aligned} (J_{\boldsymbol{\mathcal{F}}}(\boldsymbol{\mu}))^{-1} &= \frac{1}{\mu_1\mu_2} \begin{bmatrix} \mu_2 \cos(\mu_3) & \mu_2 \sin(\mu_3) \\ -\mu_1 \sin(\mu_3) & \mu_1 \cos(\mu_3) \end{bmatrix}, \\ (J_{\boldsymbol{\mathcal{F}}}(\boldsymbol{\mu}))^{-\top} &= \frac{1}{\mu_1\mu_2} \begin{bmatrix} \mu_2 \cos(\mu_3) & -\mu_1 \sin(\mu_3) \\ \mu_2 \sin(\mu_3) & \mu_1 \cos(\mu_3) \end{bmatrix}. \end{aligned}$$

Following the steps of Section 4.1 we pull back the original problem (4.14) from the parameter-dependent domain $\tilde{\Omega}(\boldsymbol{\mu})$ onto the parameter-independent domain Ω and provide the high-fidelity solutions $(\mathbf{u_h}(\boldsymbol{\mu}), p_h(\boldsymbol{\mu})) = (\mathbf{u_h}(\mathbf{x};\boldsymbol{\mu}), p_h(\mathbf{x};\boldsymbol{\mu}))$.

To do this we start by expressing the high-fidelity solutions by their respective basis functions

$$\tilde{\mathbf{u}}_{\mathbf{h}}(\boldsymbol{\mu}) = \sum_{j=1}^{N_h} w_h^{(j)}(\boldsymbol{\mu})\tilde{\boldsymbol{\phi}}_{\boldsymbol{j}}, \qquad \tilde{p}_h = \sum_{k=1}^{M_h} p_h^{(k)}(\boldsymbol{\mu})\tilde{\psi}_k.$$

As the weak formulation (4.3) holds for all $\tilde{\mathbf{v}}_{\mathbf{h}} \in \tilde{\mathbb{V}}_h$ and for all $\tilde{q}_h \in \tilde{\mathbb{Q}}_h$ we choose the test functions $\tilde{\mathbf{v}}_{\mathbf{h}} = \tilde{\phi}_{\mathbf{j}}$ and $\tilde{q}_h = \tilde{\psi}_l$. This we insert into the weak formulation (4.3). Then

Introducing the notation $\boldsymbol{\phi_i} = \begin{bmatrix} (\boldsymbol{\phi_i})_0 \\ (\boldsymbol{\phi_i})_1 \end{bmatrix}$ and $\nabla\boldsymbol{\phi_i} = \begin{bmatrix} (\boldsymbol{\phi_i})_{0,x_0} & (\boldsymbol{\phi_i})_{1,x_0} \\ (\boldsymbol{\phi_i})_{0,x_1} & (\boldsymbol{\phi_i})_{1,x_1} \end{bmatrix}$ where $(\boldsymbol{\phi_i})_{p,x_k}$ is the derivative of the p-th element of the vector $\boldsymbol{\phi_i}$ with respect to x_k, we compute

$$(J_{\mathcal{F}}(\boldsymbol{\mu}))^{-\top}\,\nabla\boldsymbol{\phi_i} = \left(\frac{1}{\mu_1\mu_2}\begin{bmatrix} \mu_2\cos(\mu_3) & -\mu_1\sin(\mu_3) \\ \mu_2\sin(\mu_3) & \mu_1\cos(\mu_3) \end{bmatrix}\begin{bmatrix} (\boldsymbol{\phi_j})_{0,x_0} & (\boldsymbol{\phi_j})_{1,x_0} \\ (\boldsymbol{\phi_j})_{0,x_1} & (\boldsymbol{\phi_j})_{1,x_1} \end{bmatrix}\right) =$$

$$\frac{1}{\mu_1\mu_2}\begin{bmatrix} \mu_2\cos(\mu_3)(\boldsymbol{\phi_i})_{0,x_0} - \mu_1\sin(\mu_3)(\boldsymbol{\phi_i})_{0,x_1} & \mu_2\cos(\mu_3)(\boldsymbol{\phi_i})_{1,x_0} - \mu_1\sin(\mu_3)(\boldsymbol{\phi_i})_{1,x_1} \\ \mu_2\sin(\mu_3)(\boldsymbol{\phi_i})_{0,x_0} + \mu_1\cos(\mu_3)(\boldsymbol{\phi_i})_{0,x_1} & \mu_2\sin(\mu_3)(\boldsymbol{\phi_i})_{1,x_0} + \mu_1\cos(\mu_3)(\boldsymbol{\phi_i})_{1,x_1} \end{bmatrix}.$$

Then we have got all we need to compute the bilinear and linear forms in equations (4.5) - (4.8). The computations are omitted here, see the appendix 5.1 for the full derivation. We get the linear system

$$\begin{bmatrix} A_h(\boldsymbol{\mu}) & B_h^\top(\boldsymbol{\mu}) \\ B_h(\boldsymbol{\mu}) & 0 \end{bmatrix}\begin{bmatrix} \mathbf{w_h}(\boldsymbol{\mu}) \\ \mathbf{p_h}(\boldsymbol{\mu}) \end{bmatrix} = \begin{bmatrix} \mathbf{f_{1h}}(\boldsymbol{\mu}) \\ \mathbf{f_{2h}}(\boldsymbol{\mu}) \end{bmatrix}. \tag{4.18}$$

where the matrices and vectors are the affine combinations

$$\begin{aligned} A_h(\boldsymbol{\mu}) &= \theta_a^1(\boldsymbol{\mu})A_1 + \theta_a^2(\boldsymbol{\mu})A_2 \\ B_h(\boldsymbol{\mu}) &= \theta_b^1(\boldsymbol{\mu})B_1 - \theta_b^2(\boldsymbol{\mu})B_2 + \theta_b^3(\boldsymbol{\mu})B_3 + \theta_b^4(\boldsymbol{\mu})B_4 \\ \mathbf{f_{1h}}(\boldsymbol{\mu}) &= \theta_{f_1}^1(\boldsymbol{\mu})A_1\mathbf{w_g} + \theta_{f_1}^2(\boldsymbol{\mu})A_2\mathbf{w_g} \\ \mathbf{f_{2h}}(\boldsymbol{\mu}) &= \theta_{f_2}^1(\boldsymbol{\mu})B_1\mathbf{w_g} + \theta_{f_2}^2(\boldsymbol{\mu})B_2\mathbf{w_g} + \theta_{f_2}^3(\boldsymbol{\mu})B_3\mathbf{w_g} + \theta_{f_2}^4(\boldsymbol{\mu})B_4\mathbf{w_g}. \end{aligned} \tag{4.19}$$

Here $\mathbf{w_g}$ is the coefficient vector of the discrete lifting function. We approximate the lifting function with the discrete lifting function $\mathbf{u_g} = \sum_{i=1}^{N_h} w_g^{(i)}\boldsymbol{\phi_i}$, where this approximation takes the value 1 at the boundary of the domain. Then $\mu_5\mathbf{u_g}$ will obtain the right value at the inflow Dirichlet boundary, that is $\mu_5\mathbf{u_g}|_{\Gamma_{D_1}} = \mu_5$.

From equation (4.19) we recognize the theta-functions first introduced in equation (2.38). The theta-functions and the elements of the matrices and vectors in (4.19) are

$$\begin{aligned} \theta_a^1(\boldsymbol{\mu}) &= \frac{\mu_4\mu_2}{\mu_1} & (A_1)_{ij} &= a_1(\boldsymbol{\phi_j},\boldsymbol{\phi_i}) = \int_\Omega \left((\boldsymbol{\phi_j})_{0,x_0}(\boldsymbol{\phi_i})_{0,x_0} + (\boldsymbol{\phi_j})_{1,x_0}(\boldsymbol{\phi_i})_{1,x_0}\right)\,\mathrm{d}\Omega \\ \theta_a^2(\boldsymbol{\mu}) &= \frac{\mu_4\mu_1}{\mu_2} & (A_2)_{ij} &= a_2(\boldsymbol{\phi_j},\boldsymbol{\phi_i}) = \int_\Omega \left((\boldsymbol{\phi_j})_{0,x_1}(\boldsymbol{\phi_i})_{0,x_1} + (\boldsymbol{\phi_j})_{1,x_1}(\boldsymbol{\phi_i})_{1,x_1}\right)\,\mathrm{d}\Omega \\ \theta_b^1(\boldsymbol{\mu}) &= \mu_2\cos(\mu_3) & (B_1)_{ij} &= b_1(\boldsymbol{\phi_j},\psi_i) = -\int_\Omega \psi_i(\boldsymbol{\phi_j})_{0,x_0}\,\mathrm{d}\Omega \\ \theta_b^2(\boldsymbol{\mu}) &= -\mu_1\sin(\mu_3) & (B_2)_{ij} &= b_2(\boldsymbol{\phi_j},\psi_i) = -\int_\Omega \psi_i(\boldsymbol{\phi_j})_{0,x_1}\,\mathrm{d}\Omega \\ \theta_b^3(\boldsymbol{\mu}) &= \mu_2\sin(\mu_3) & (B_3)_{ij} &= b_3(\boldsymbol{\phi_j},\psi_i) = -\int_\Omega \psi_i(\boldsymbol{\phi_j})_{1,x_0}\,\mathrm{d}\Omega \\ \theta_b^4(\boldsymbol{\mu}) &= \mu_1\cos(\mu_3) & (B_4)_{ij} &= b_4(\boldsymbol{\phi_j},\psi_i) = -\int_\Omega \psi_i(\boldsymbol{\phi_j})_{1,x_1}\,\mathrm{d}\Omega \end{aligned}$$

$$\begin{aligned} \theta_{f_1}^1(\boldsymbol{\mu}) &= -\frac{\mu_4\mu_5\mu_2}{\mu_1} & \theta_{f_1}^2(\boldsymbol{\mu}) &= -\frac{\mu_4\mu_5\mu_1}{\mu_2} \\ \theta_{f_2}^1(\boldsymbol{\mu}) &= -\mu_2\mu_5\cos(\mu_3) & \theta_{f_2}^2(\boldsymbol{\mu}) &= \mu_1\mu_5\sin(\mu_3) \\ \theta_{f_2}^3(\boldsymbol{\mu}) &= -\mu_2\mu_5\sin(\mu_3) & \theta_{f_2}^4(\boldsymbol{\mu}) &= -\mu_1\mu_5\cos(\mu_3), \end{aligned} \tag{4.20}$$

The high-fidelity solutions are then obtained as

$$\mathbf{u_h}(\boldsymbol{\mu}) = \sum_{j=1}^{N_h}(w_h(\boldsymbol{\mu})^{(j)} + \mu_4 w_g^{(j)})\boldsymbol{\phi_j} \quad \text{and} \quad p_h(\boldsymbol{\mu}) = \sum_{k=1}^{M_h} p_h(\boldsymbol{\mu})^{(k)}\psi_k$$

where the elements of the solution vector from the linear system (4.9) are $\mathbf{w_h}(\boldsymbol{\mu}) = [w_h(\boldsymbol{\mu})^{(1)}, w_h(\boldsymbol{\mu})^{(2)}, \ldots, w_h(\boldsymbol{\mu})^{(N_h)}]$ and $\mathbf{p_h}(\boldsymbol{\mu}) = [p_h(\boldsymbol{\mu})^{(1)}, p_h(\boldsymbol{\mu})^{(2)}, \ldots, p_h(\boldsymbol{\mu})^{(M_h)}]$. $\mathbf{w_h}(\boldsymbol{\mu})$ and $\mathbf{p_h}(\boldsymbol{\mu})$ are the vectors associated with the degrees of freedom for the RB velocity and the RB pressure respectively.

4.2.3 Snapshots and relative error

As remarked in Chapter 3 the Stokes equations are a simplification of the Navier-Stokes equations acceptable for low Reynolds numbers. As the Stokes equations are already simplified linear equations, without loss of generality we choose the parameters $\boldsymbol{\mu}$ such that $0.05 \leq \text{Re} \leq 30$, [13].

Then we can decide the limits of the geometric and physical parameters. Let the parameter space for the geometric parameters be $[0.5, 3]^2 \times [-\pi/6, \pi/6]$. That is scaling in x, y-direction by scales in the interval $[0.5, 3]$ and rotate by an angle in the interval $[-\pi/6, \pi/6]$. Then we use this information to decide the physical parameters.

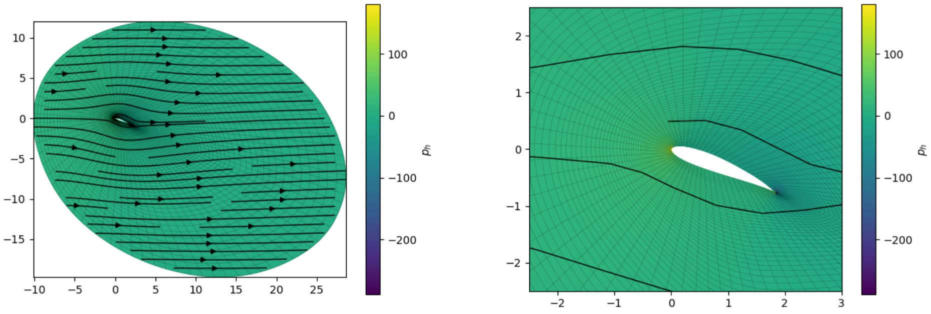
The Reynolds number defined in equation (3.1) is $\text{Re} = \frac{VL}{\nu}$, where L, the characteristic length, is the chord length as defined in the previous example 3.3. The chord will be at min 0.5 and max 3 due to the geometric parameter space. Let the interval for the kinematic viscosity $\nu = \mu_4$ be $[1, 10]$ then the characteristic velocity, that is the inflow μ_5 will be in the interval $[1, 10]$ due to the Reynolds number interval. Summarized the parameter space is $\mathbb{P} = [0.5, 3]^2 \times [-\pi/6, \pi/6] \times [1, 10]^2$, where $[0.5, 3]^2 \times [-\pi/6, \pi/6]$ is the geometric parameters and $[1, 10]^2$ is the physical parameters.

To create a reduced basis solver we compute a set of high-fidelity solutions of the linear system (4.18). This is done for a parameter training set $\Xi_{train} = \{\boldsymbol{\mu_1}, \ldots, \boldsymbol{\mu_{n_s}}\}$ creating the corresponding high-fidelity velocity snapshots $\{\mathbf{u_h}(\boldsymbol{\mu_1}), \ldots, \mathbf{u_h}(\boldsymbol{\mu_{n_s}})\}$ and the corresponding high-fidelity pressure snapshots $\{\mathbf{p_h}(\boldsymbol{\mu_1}), \ldots, \mathbf{p_h}(\boldsymbol{\mu_{n_s}})\}$. The snapshots are saved in their respective snapshot matrices

$$S_u = [\mathbf{u_h}(\boldsymbol{\mu_1})| \ldots |\mathbf{u_h}(\boldsymbol{\mu_{n_s}})] \qquad S_p = [\mathbf{p_h}(\boldsymbol{\mu_1})| \ldots |\mathbf{p_h}(\boldsymbol{\mu_{n_s}})]. \tag{4.21}$$

Figures 4.3 and 4.4 shows a couple of snapshots for a couple of different parameter configurations.

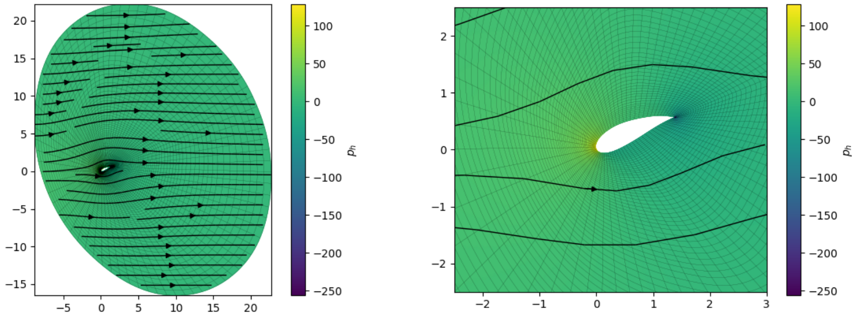
As seen in Section 4.2.1 the element size increase because of the mapping. At the most the elements are scaled by a factor of 3 in both directions. This must be kept in mind when studying the relative error introduced in the previous example in Section 3.3.2. Hence we must make sure the grid is refined such that we have enough grid-lines close to the airfoil to obtain good enough accuracy (our choice) also for the scaled airfoil (equivalent to scaled elements). We require in the following that the high-fidelity solutions approximate the exact solutions good enough, that is for a ε of our choice in equation (4.2). This we do by requiring the relative error (3.46) between the finest level high-fidelity solutions and the high-fidelity solutions to be smaller than or equal to 10^{-3}. Then we are satisfied



(a) NACA airfoil far out.

(b) NACA airfoil close up.

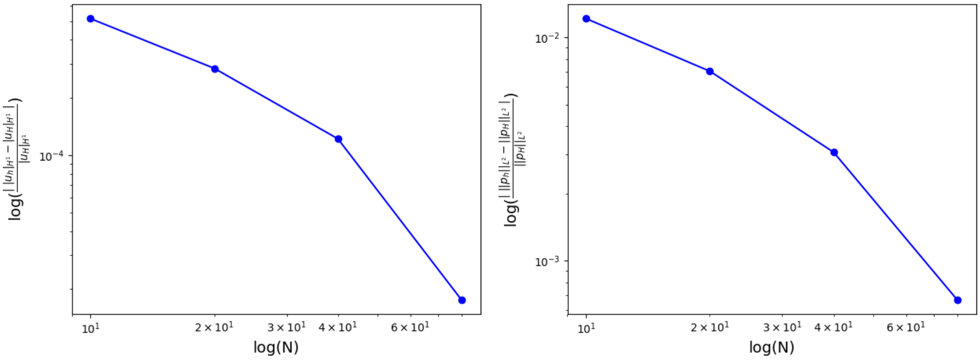
Figure 4.3: Galerkin approximation of velocity field $\mathbf{u_h}$ and pressure p_h around stretched and rotated NACA airfoil. Velocity field displayed as streamlines and pressure as colormap. Solved on a 80×80 grid. Snapshot corresponding to the parameters $\boldsymbol{\mu} = [2, 1.5, -\pi/8, 5, 8]$.



(a) NACA airfoil far out.

(b) NACA airfoil close up.

Figure 4.4: Galerkin approximation of velocity field $\mathbf{u_h}$ and pressure p_h around stretched and rotated NACA airfoil. Velocity field displayed as streamlines and pressure as colormap. Solved on a 80×80 grid. Snapshot corresponding to the parameters $\boldsymbol{\mu} = [1.5, 2, \pi/8, 5, 8]$.



(a) Relative error velocity field.

(b) Relative error pressure field.

Figure 4.5: Error between norms of the high-fidelity solutions ($\mathbf{u_h}, p_h$) and the norms of the finest level high-fidelity solutions ($\mathbf{u_H}, p_H$) relative to the norm of the finest level high-fidelity solutions. Measured for finer and finer grids $N \times N$, with full scaling $\mu_1 = \mu_2 = 3$. The finest level high-fidelity solutions are computed on the finest grid 320×320.

with the accuracy of which the high-fidelity solutions in equation (4.2) approximate the exact solutions. Then the accuracy of which the reduced solutions approximate the exact solutions is decided by how good the reduced basis solutions approximate the high-fidelity solutions.

As the map (4.15) scales the elements we compute the relative error as in Figure 3.11 for a max scaled grid and we obtain the relative error seen in Figure 4.5. Then to obtain the desired relative error of 10^{-3} we will in the following do all computations on a 80×80 grid which gives $N_h = 2 \cdot 25499$ degrees of freedom for the high-fidelity velocity approximation and $M_h = 6480$ degrees of freedom for the high-fidelity pressure approximation.

4.2.4 Eigenvalues of the correlation matrices

When considering to make a reduced order model of some given problem, certain preliminary work should be done to consider the reducibility of the problem at hand. Before even doing computations, aspects such as the parametric complexity of the problem could be considered. This is reflected by the number of affine terms given by the numbers Q_a, Q_b, Q_{f_1} and Q_{f_2} in the affine expansions (2.39). A high model complexity is equivalent with high number of affine terms and can make the problem less attractive to reduced order modeling. More details on this and other possible methods such as Kolmogorov n-width can be found in e.g. [49, 18]. This section discuss observations made after high-fidelity solutions are computed.

To understand if problem (4.14) is reducible or not we should compute a set of snapshots for both the velocity and the pressure and store them in the snapshot matrices S_u and S_p in equation (4.21). Further we compute the corresponding correlation matrices C_u and C_p defined by equation (2.65), where we have used the inner product associated with the

Table 4.1: Number of parameter values in each parameter interval for different regular grids.

Parameter	μ_1	μ_2	μ_3	μ_4	μ_5
Interval	$[0.5, 3]$	$[0.5, 3]$	$[-\pi/6, \pi/6]$	$[1, 10]$	$[1, 10]$
uniform grid 1	6	2	2	2	2
uniform grid 2	2	6	2	2	2
uniform grid 3	2	2	6	2	2
uniform grid 4	2	2	2	6	2
uniform grid 5	2	2	2	2	6
uniform grid 6	4	4	1	1	1
uniform grid 7	6	6	1	1	1
uniform grid 8	8	8	1	1	1
uniform grid 9	6	6	3	1	1
uniform grid 10	6	6	1	3	1
uniform grid 11	6	6	1	1	3
uniform grid 12	6	6	2	2	1
uniform grid 13	6	6	2	1	2
uniform grid 14	6	6	1	2	2
uniform grid 15	7	7	3	1	1
uniform grid 16	6	6	4	1	1
uniform grid 17	7	7	3	2	2
uniform grid 18	8	8	3	1	1
uniform grid 19	11	11	5	1	1
gauss legendre	7	7	3	1	1

H^1-seminorm for the velocity space and the L^2-inner product for the pressure space, that is

$$C_u = S_u^\top X_{h,\mathbf{u_h}} S_u \qquad C_p = S_p^\top X_{h,p_h} S_p. \tag{4.22}$$

Here we have used the notation of the inner product matrices in Section 2.5.3, that is $(X_{h,\mathbf{u_h}})_{ij} = \left(\nabla\phi_{\mathbf{i}}, \nabla\phi_{\mathbf{j}}\right)_{L^2(\Omega)}$ and $(X_{h,p_h})_{ij} = \left(\psi_i, \psi_j\right)_{L^2(\Omega)}$.

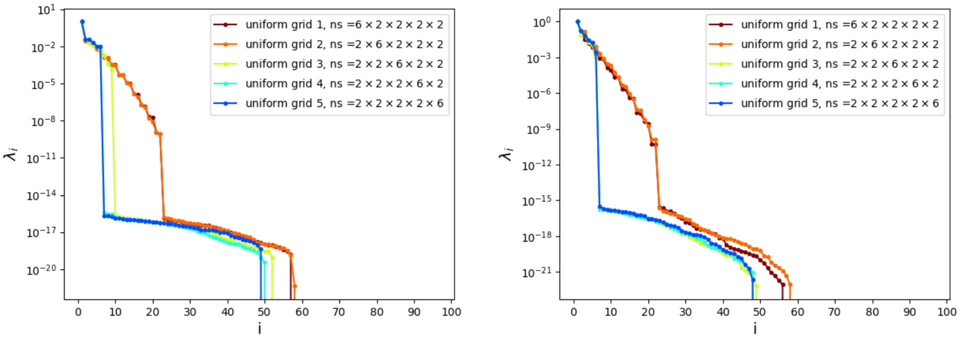
Furthermore we should check the decay of the eigenvalues of the correlation matrix. The eigenvalues are associated with the energy of the snapshots so for a sufficient rapid decay the energy of the solution set could be represented by a limited number of POD modes. Then if the solution set reflects the solution manifold $\mathbb{M}_h$ it means that the solution manifold could be approximated by a limited number of POD modes and a reduced order model is preferable.

Then the challenge is to choose a parameter training set Ξ_{train} such that the snapshots constructed from these parameters in fact is a good enough representation of the solution manifold. In order to capture the variability of the snapshots as the parameters vary over the parameter space $\mathbb{P}$ we will investigate different sampling strategies to compute the snapshots $(\mathbf{u_h}(\boldsymbol{\mu_i}), \mathbf{p_h}(\boldsymbol{\mu_i}))$. We think of the parameter space $\mathbb{P}$ as a five dimensional space and choose different number of points in each parameter direction to vary the parameter vectors $\boldsymbol{\mu_i}$.

Two ideas on how to sample the parameter sets comes to mind, by a form of regular sampling method and by a form of random sampling method. Here uniform grids, that is grids with equidistant spacing, are used to better understand which parameters are important to capture the variability of the snapshots. See Figure 4.10a for an example of a uniform grid of a $[-1, 1]^2$ parameter space. Afterwards we compare the uniform grids with an other regular grid made by *Gauss-Legendre* sampling, a random sampling method and a *Latin Hypercube* sampling method. All regular grids used are summarized in Table 4.1 and all random sampling methods used are summarized in Table 4.2.

Table 4.2: Number of parameter vectors for different random based sampling methods

Parameter	$\boldsymbol{\mu_i} = [\mu_1, \mu_2, \mu_3, \mu_4, \mu_5]^\top$
	i
LHS 1	50
LHS 2	150
LHS 3	200
LHS 4	300
LHS 5	400
LHS 6	605
random	150



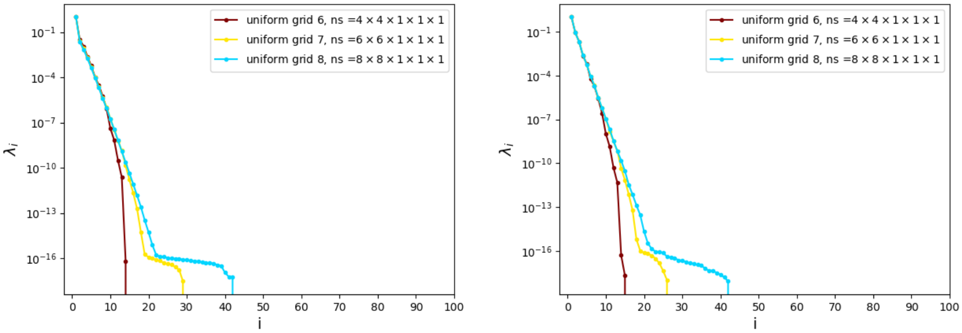
(a) Eigenvalues corresponding to the correlation matrix C_u.

(b) Eigenvalues corresponding to the correlation matrix C_p.

Figure 4.6: 100 first eigenvalues of correlation matrix corresponding to the sampling strategies uniform grid 1 – 5. All eigenvalues are normalized with respect to λ_1.

One could wonder what we look for when plotting the eigenvalues. Since the eigenvalues of the correlation matrix represents the energy of the solution manifold in some way we seek the parameter-set best representing the energy of the solution set. We do not know how many non-zero eigenvalues, but given two sets of eigenvalues corresponding to each their snapshot matrix the one with the most nonzero eigenvalues — and hence most probably the largest total sum of the eigenvalues — would seem to be the one capturing the most of the total energy. After we have found a training set that gives a good representation of the solution manifold, then we evaluate the rapid decay of the eigenvalues.

First we start of by five uniform grids named uniform grid 1 – 5. As seen in Table 4.1 we choose two uniform values in the parameter interval for four parameter directions at a time while the fifth parameter direction has 6 uniform values. When plotting these grids in Figure 4.6 we observe that for the pressure correlation matrix C_p it is important to have a larger number of points in the parameter direction μ_1 and μ_2 to capture the variability of the solution set, shown in Figure 4.6b. Figure 4.6a also shows that uniform grid 3, that is a larger number of points in the parameter direction μ_3, is not as important μ_1 and μ_2 to capture the variability of the solution set, but it is still more important than parameter directions μ_4 and μ_5, which in fact seems equally unimportant. These results could lead



(a) Eigenvalues corresponding to the correlation matrix C_u.

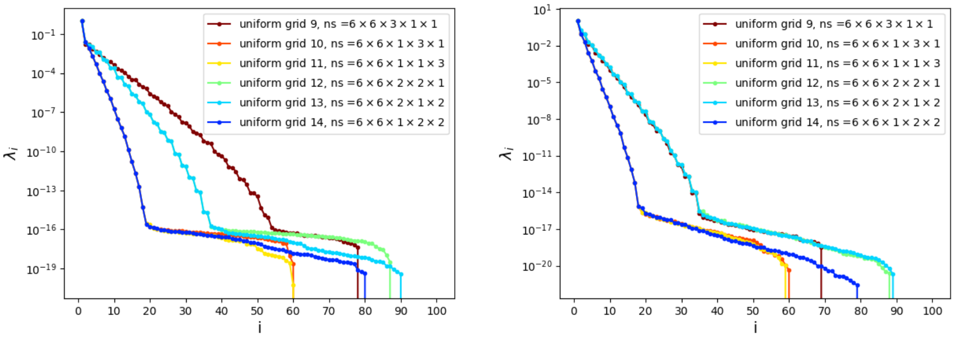
(b) Eigenvalues corresponding to the correlation matrix C_p.

Figure 4.7: 100 first eigenvalues of correlation matrix corresponding to the sampling strategies uniform grid $6-8$. All eigenvalues are normalized with respect to λ_1.

the reader to wonder if it is possible to better capture the variability of the solution set by combining a large number of points in some of the parameter directions. As the parameter directions μ_1 and μ_2 seems equally important we choose the same number of points in these directions. In Figure 4.7 we seek to find how many equally spaced points for both μ_1 and μ_2 seems to be sufficient, while fixing only one point in the other parameter directions. We conclude that six points in each direction seems to suffice for now.

Next we see what happens if a larger number of points in some of the other parameter directions is combined with a larger number of points in the directions μ_1 and μ_2. Figure 4.8 shows that uniform grids 9, 12 and 13 seems to be better than uniform grids 10, 11 and 14. We note that uniform grids 9, 12 and 13 have in common a larger number of points in the directions μ_1, μ_2 and μ_3. From Figure 4.8a the best of the methods is uniform grid 9 which has the largest number of points in parameter directions μ_1, μ_2 and μ_3.

As uniform grid 9 seemed as a good way of choosing parameter values we study this grid further by comparing it to larger grids. Figure 4.9 shows that further increasing the number of points in parameter direction μ_3, such as uniform grid 16, gives no improvement. Increasing the number of points in direction μ_1 and μ_2 by one or two points gives uniform grid 15 and uniform grid 18 respectively, which are slightly better than the other. As the uniform grids 15 and 18 are very close to each other we would say that uniform grid 15 is maybe better than 18 as the difference in parameter configurations is $7\times7\times3\times1\times1 = 147$ to $8\times8\times3\times1\times1 = 192$. That is an increase of approximately 30% in the amount of parameter configurations from uniform grid 15 to 18 and as the two grids are almost equally good at explaining the variability of the solution set, uniform grid 15 is preferred due to limited computational resources. Uniform grid 17 confirms that parameter directions μ_4 and μ_5 seems unimportant to better capture the variability of the solution set as uniform grid 15 and 17 seems to be equally good. As we are satisfied with how the uniform grid 15 performs we try another regular grid with the same parameter configuration, namely Gauss-Legendre sampling with parameter configuration $7 \times 7 \times 3 \times 1 \times 1$, see Table 4.1. Gauss-Legendre



(a) Eigenvalues corresponding to the correlation matrix C_u.

(b) Eigenvalues corresponding to the correlation matrix C_p.

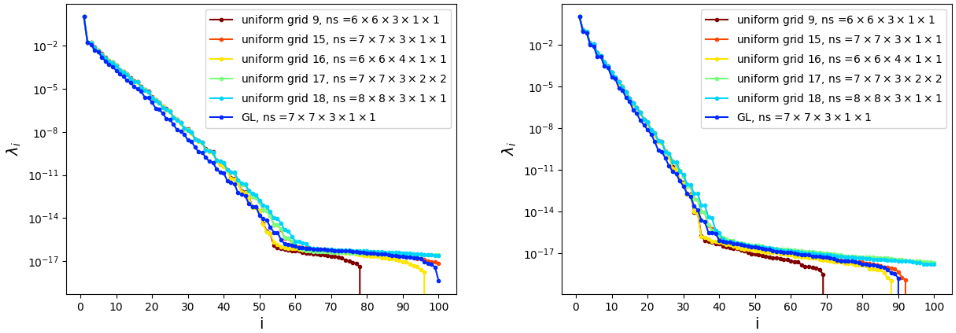
Figure 4.8: 100 first eigenvalues of correlation matrix corresponding to the sampling strategies uniform grid 9 – 14. All eigenvalues are normalized with respect to λ_1.

points gives a regular grid with points slightly pushed closer to the boundary than a uniform grid as shown in Figure 4.10b. In [62] we see an example of Gauss-Legendre points used for a Weighted POD method. See [49] for examples of other regular grids. Figure 4.9 shows that the Gauss-Legendre grid performs slightly worse than the uniform grid 15.

Beside regular grids, an other intuitive way of choosing our training set Ξ_{train} is choosing it somehow random. In [41] the use of *simple random sampling* and *Latin Hypercube sampling* (LHS) is recognized as good methods for selecting values of input parameters when analyzing output from a computer code. Lhs is used e.g. in [19] as a method to provide coverage of the parameter space in Population of models. Lhs is also a random sampling such as the simple random sampling method, but LHS is a method designed by partitioning the parameter space into equally large cells and if a cell is filled with a parameter value, then that cell will not be chosen again. This makes LHS an improved version of the simple random sampling as it is designed such that the chosen points are marginally spread over the parameter interval for each parameter. See [56] for further details. An example of 50 sampled parameter values by the methods simple randdom sampling and LHS sampling for a 2D parameter space $[-1, 1]^2$ can be seen in figures 4.11a and 4.11b respectively.

As we where satisfied with uniform grid 15 yielding 147 parameter vectors we use approximately the same amount of points for the simple random sampling and the Latin Hypercube sampling when comparing them to each other in Figure 4.12. As observed they are hard to differ, and the reader can trust the author that they also are hard to differ for larger number of points so we will only study LHS in the following.

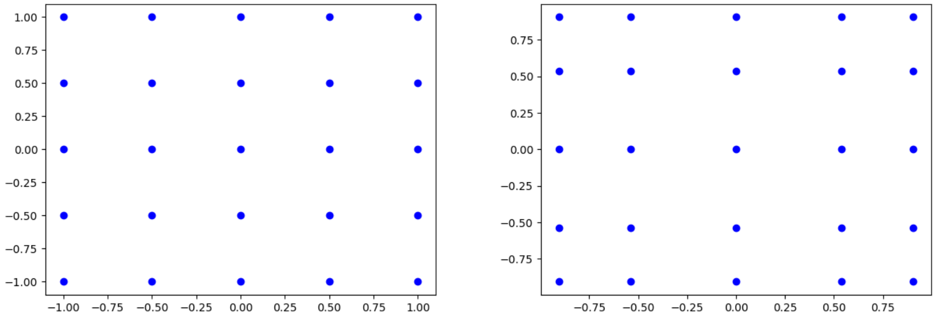
Now we study the influence of the number of snapshots n_s on the eigenvalues of the correlation matrices C_u and C_p. In figures 4.13a and 4.13b we see up to the first 100 eigenvalues of the correlation matrices C_u and C_p obtained using increasing number of Latin Hybercube samples. We observe that $n_s = 300$ LHS, that is sampling method LHS 4, is more than sufficient for both the velocity correlation matrix and the pressure correlation



(a) Eigenvalues corresponding to the correlation matrix C_u.

(b) Eigenvalues corresponding to the correlation matrix C_p.

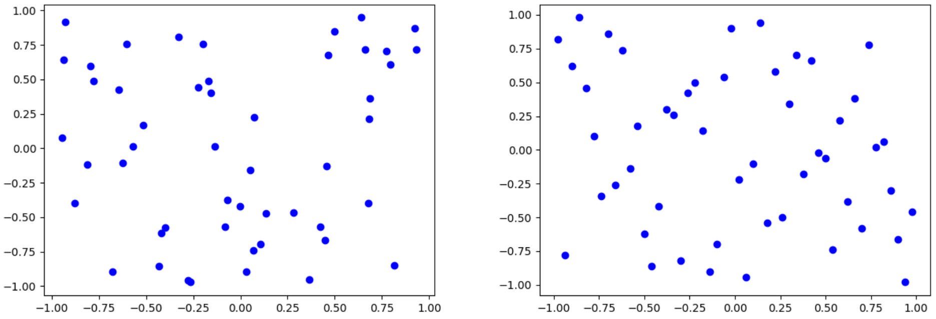
Figure 4.9: 100 first eigenvalues of correlation matrix corresponding to the sampling strategies uniform grid 9,uniform grid $15 - 18$ and Gauss - Legendre sampling. All eigenvalues are normalized with respect to λ_1.



(a) 5 × 5 uniform grid

(b) 5 × 5 Gauss-Legendre grid.

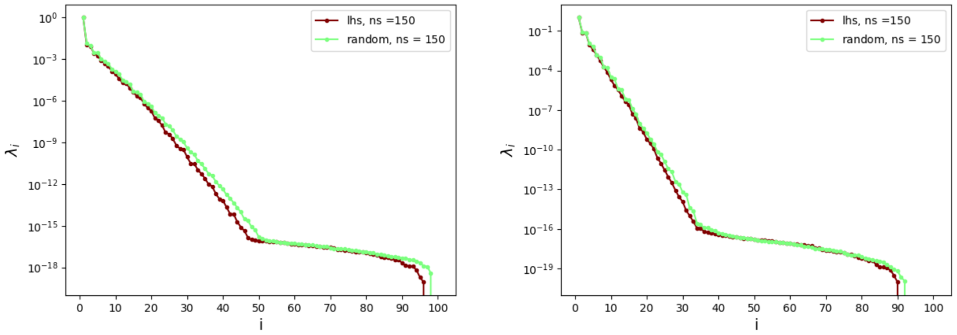
Figure 4.10: Regular grids to sample a 2D parameter domain.



(a) 50 points sampled using a simple random sampling strategy.

(b) 50 points sampled using a Latin Hypercube sampling strategy.

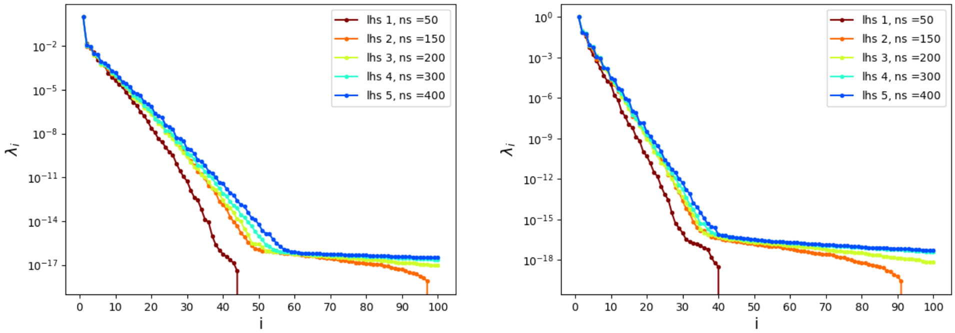
Figure 4.11: Random sampling points to sample a 2D parameter domain.



(a) Eigenvalues corresponding to the correlation matrix C_u.

(b) Eigenvalues corresponding to the correlation matrix C_p.

Figure 4.12: 100 first eigenvalues of correlation matrix corresponding to the sampling strategies Latin Hypercube sampling and simple random sampling. All eigenvalues are normalized with respect to λ_1.



(a) Eigenvalues corresponding to the correlation matrix C_u.

(b) Eigenvalues corresponding to the correlation matrix C_p.

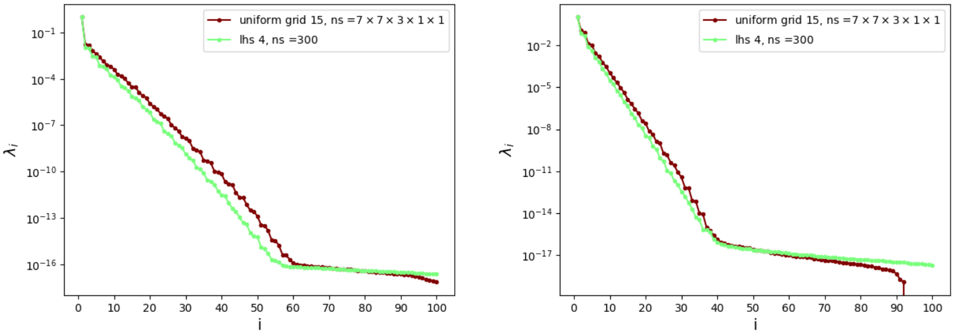
Figure 4.13: 100 first eigenvalues of correlation matrix corresponding to the sampling strategies Latin Hypercube sampling $1-4$. All eigenvalues are normalized with respect to λ_1.

matrix. Let us also compare LHS 4 to the uniform grid 15. Observed from Figure 4.14 both methods seems to have about the same amount of non-zero eigenvalues, but the LHS 4 has a more linear decay of eigenvalues than the uniform grid as the uniform grid gives a more curved decay of the eigenvalues. An interesting characteristic of the different methods which can also be observed in similar plots, e.g. see [49].

Last, but not least we check if the methods uniform grid 15 and LHS 4 actually are good enough. We do this by comparing them to a great deal larger grid and sampling method, that is the uniform grid 19 from Table 4.1 with $11 \times 11 \times 5 \times 1 \times 1$ sampling points and the LHS 6 consisting of the same number of points, that is 605 points. From Figure 4.14 we observe that LHS 4 is pretty much similar to LHS 6, only for small eigenvalues do they differ, and the same goes for uniform grid 15 compared to uniform grid 19. Also note that the increased number of points in uniform grid 19 has made the decay of eigenvalues more linear than uniform grid 15 has, making it look more like the decay in the eigenvalues of the methods LHS 4 and 6. My guess is that this is the more correct decay of the eigenvalues, especially since the LHS method gives this decay and LHS is a method developed to give a good representation of the parameter space (and equivalently a good representation of the solution set). Still uniform grid 15 and LHS 4 seems to be more than sufficient methods and much less demanding than uniform grid 19 and LHS 6 with respect to computational resources.

In the end we make some last remarks. As less than the 100 first eigenvalues of the correlation matrices C_u and C_p captures the energy of the solution manifold represented by n_s snapshots computed from a system of $N_h = 2 \cdot 25499$ and $M_h = 6480$ degrees of freedom for the velocity field and the pressure field respectively, the decay of these eigenvalues is rapid enough to make a reduced order model.

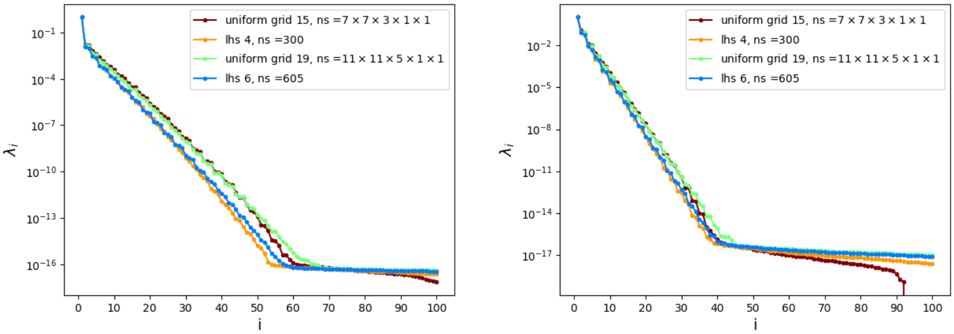
The last figures 4.14 and 4.15 shows that the uniform grid methods gives slightly better representation of the eigenvalues than the LHS methods, or at least much less points are needed for the training set Ξ_{train} using uniform grids than LHS methods to obtain



(a) Eigenvalues corresponding to the correlation matrix C_u.

(b) Eigenvalues corresponding to the correlation matrix C_p.

Figure 4.14: 100 first eigenvalues of correlation matrix corresponding to the sampling strategies uniform grid 15 and Latin Hypercube sampling 4. All eigenvalues are normalized with respect to λ_1.



(a) Eigenvalues corresponding to the correlation matrix C_u.

(b) Eigenvalues corresponding to the correlation matrix C_p.

Figure 4.15: 100 first eigenvalues of correlation matrix corresponding to the sampling strategies uniform grid 15, uniform grid 19, Latin Hypercube sampling 3 and Latin Hypercube sampling 5. All eigenvalues are normalized with respect to λ_1.

the same accurate approximation of the spectrum of the solution set. When it comes to computational efficiency this makes the uniform grids more attractive than the LHS methods for this problem. Note that this is in fact problem dependent, see [49] for an example where it is the other way around. A hypothesis to why it is like this for the Stokes problem on the NACA airfoil is that since only the three first parameters $\mu_1 - \mu_3$ is important to vary to get a good approximation of the solution set the best uniform grids can be designed to vary only these values, whereas the Latin Hypercube sampling is designed to represent the whole parameter space. By constructing a LHS method for only the first three parameters LHS would probably perform better.

The last remark of this section is given to the three significant parameters $\mu_1 - \mu_3$ and as why they seem more important than μ_4 and μ_5. From a physical perspective $\mu_1 - \mu_3$ changes the characteristics of the problem drastic as they stretch/compresses the airfoil and rotate it giving fundamental different solutions whereas μ_4 and μ_5 decrease/increase the kinematic viscosity slightly or increase/decrease the inflow slightly.

This is also expressed algebraic. In equation (4.19) the three parameters $\mu_1 - \mu_3$ are included in more theta-functions than μ_4 and μ_5 as we see in equation (4.20). Also μ_4 and μ_5 are mainly in the theta functions for the right hand side functions and does not affect the rank of the solution set much. On the other hand $\mu_1 - \mu_3$ are in almost every theta-function. As discussed in the introduction to Section 4.2.4 it is the number of theta-functions for the A and B matrix, namely Q_a and Q_b that decides the complexity of our model. Hence varying parameters $\mu_1 - \mu_3$ is needed for the computed snapshots to capture the complexity of our model.

4.2.5 Accuracy of the Galerkin reduced basis approximation

The reduced basis functions are linear combinations of the high-fidelity basis functions with coefficients given from a POD method, as in equation (2.26). These coefficients stored as columns in a transformation matrix V are called a POD basis or POD mode. We need to compute a POD basis for both the velocity snapshots S_u and the pressure snapshots S_p. This is done as in equation (2.63) and we obtain

$$\begin{aligned} \boldsymbol{\xi_i^u} &= \frac{1}{\sqrt{\lambda_i^u}} S_u \boldsymbol{\psi_i^u} \quad 1 \leq i \leq N_u \\ \boldsymbol{\xi_i^p} &= \frac{1}{\sqrt{\lambda_i^p}} S_p \boldsymbol{\psi_i^p} \quad 1 \leq i \leq N_p, \end{aligned} \tag{4.23}$$

where λ_i^u, λ_i^p are the eigenvalues associated with the velocity and pressure correlation matrices (4.22) respectively and $\boldsymbol{\psi_i^u}, \boldsymbol{\psi_i^p}$ are the corresponding eigenvectors. We choose N_u, N_p largest eigenvalues of the correlation matrices when constructing the transformation matrices as the eigenvalues are associated with the energy of the system.

The POD modes $\boldsymbol{\xi_i^u}, \boldsymbol{\xi_j^p}$ $1 \leq \ldots i \leq N_u$ $1 \leq \ldots j \leq N_p$ are stored in each their transformation matrix as $V_u = [\boldsymbol{\xi_1^u} | \ldots | \boldsymbol{\xi_{N_u}^u}]$ and $V_p = [\boldsymbol{\xi_1^p} | \ldots | \boldsymbol{\xi_{N_p}^p}]$. Further we denote by $(\mathbf{u_N}(\boldsymbol{\mu}), \mathbf{p_N}(\boldsymbol{\mu})) \in \mathbb{R}^{N_u} \times \mathbb{R}^{N_p}$ the reduced basis solutions such that $\mathbf{u_h}(\boldsymbol{\mu}) \approx V_u \mathbf{u_N}(\boldsymbol{\mu})$ and $\mathbf{p_h}(\boldsymbol{\mu}) \approx V_p \mathbf{p_N}(\boldsymbol{\mu})$. $V_u \mathbf{u_N}(\boldsymbol{\mu})$ and $V_p \mathbf{p_N}(\boldsymbol{\mu})$ are the full order representation of the RB velocity vector and the RB pressure vector respectively equivalent to what is introduced in Section 2.5.2. Then by storing the transformation matrices as

$$V = \begin{bmatrix} V_u & 0 \\ 0 & V_p \end{bmatrix}$$

the Galerkin reduced basis linear system can be obtained as a Galerkin projection. We insert the full order representation of the RB velocity vector and the RB pressure vector into the linear system (4.18) and require the residual to be orthogonal to the columns of V:

$$\begin{bmatrix} V_u^\top & 0 \\ 0 & V_p^\top \end{bmatrix} \begin{bmatrix} A_h(\boldsymbol{\mu}) V_u \mathbf{u_N}(\boldsymbol{\mu}) + B_h^\top(\boldsymbol{\mu}) V_p \mathbf{p_N}(\boldsymbol{\mu}) - \mathbf{f_{1h}}(\boldsymbol{\mu}) \\ B_h(\boldsymbol{\mu}) V_u \mathbf{u_N}(\boldsymbol{\mu}) - \mathbf{f_{2h}}(\boldsymbol{\mu}) \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}$$

This is equivalent to the linear system

$$\begin{bmatrix} A_N(\boldsymbol{\mu}) & B_N^\top(\boldsymbol{\mu}) \\ B_N(\boldsymbol{\mu}) & 0 \end{bmatrix} \begin{bmatrix} \mathbf{w_N}(\boldsymbol{\mu}) \\ \mathbf{p_N}(\boldsymbol{\mu}) \end{bmatrix} = \begin{bmatrix} \mathbf{f_{1N}}(\boldsymbol{\mu}) \\ \mathbf{f_{2N}}(\boldsymbol{\mu}) \end{bmatrix}. \tag{4.24}$$

where we have defined the matrices and vectors

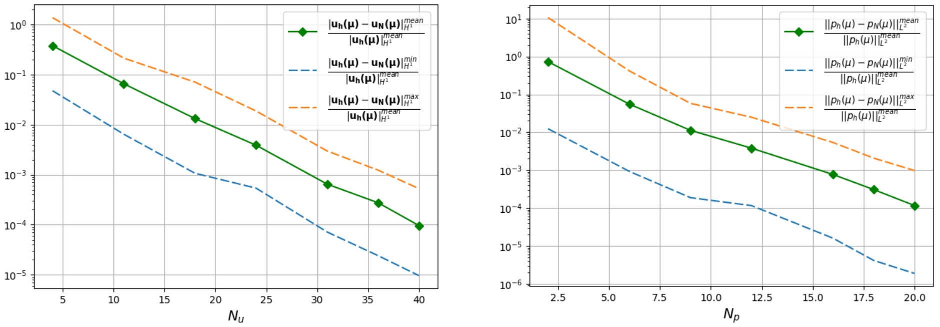
$$\begin{aligned} A_N(\boldsymbol{\mu}) &= V_u^\top A_h(\boldsymbol{\mu}) V_u & B_N(\boldsymbol{\mu}) &= V_p^\top B_h(\boldsymbol{\mu}) V_u \\ \mathbf{f_{1N}}(\boldsymbol{\mu}) &= V_u^\top \mathbf{f_{1h}}(\boldsymbol{\mu}) & \mathbf{f_{2N}}(\boldsymbol{\mu}) &= V_p^\top \mathbf{f_{2h}}(\boldsymbol{\mu}). \end{aligned} \tag{4.25}$$

As the high-fidelity matrices and vectors in equation (4.25) can be written as affine combinations so can the reduced matrices and vectors in equation (4.25)

$$\begin{aligned} A_N(\boldsymbol{\mu}) &= V_u^\top A_h(\boldsymbol{\mu}) V_u = \theta_a^1(\boldsymbol{\mu}) V_u^\top A_1 V_u + \theta_a^2(\boldsymbol{\mu}) V_u^\top A_2 V_u \\ B_N(\boldsymbol{\mu}) &= V_p^\top B_h(\boldsymbol{\mu}) V_u = \theta_b^1(\boldsymbol{\mu}) V_p^\top B_1 V_u - \theta_b^2(\boldsymbol{\mu}) V_p^\top B_2 V_u + \theta_b^3(\boldsymbol{\mu}) V_p^\top B_3 V_u + \theta_b^4(\boldsymbol{\mu}) V_p^\top B_4 V_u \\ \mathbf{f_{1N}}(\boldsymbol{\mu}) &= V_u^\top \mathbf{f_{1h}}(\boldsymbol{\mu}) = \theta_{f_1}^1(\boldsymbol{\mu}) V_u^\top A_1 \mathbf{w_g} + \theta_{f_1}^2(\boldsymbol{\mu}) V_u^\top A_2 \mathbf{w_g} \\ \mathbf{f_{2N}}(\boldsymbol{\mu}) &= V_p^\top \mathbf{f_{2h}}(\boldsymbol{\mu}) = \theta_{f_2}^1(\boldsymbol{\mu}) V_p^\top B_1 \mathbf{w_g} + \theta_{f_2}^2(\boldsymbol{\mu}) V_p^\top B_2 \mathbf{w_g} + \theta_{f_2}^3(\boldsymbol{\mu}) V_p^\top B_3 \mathbf{w_g} + \theta_{f_2}^4(\boldsymbol{\mu}) V_p^\top B_4 \mathbf{w_g}. \end{aligned} \tag{4.26}$$

where the theta-functions are as defined in equation (4.20). During the offline stage we pre-compute all parameter-independent matrices and vectors in equation (4.26) and during the online stage we compute the sums for a given input parameter vector $\boldsymbol{\mu}$, equivalent to what is introduced in Section 2.5.4.

When deciding how to sample snapshots we need to consider how we can expect the input parameter vector $\boldsymbol{\mu}$ to vary. In the following we will see that if the input parameters are expected to be random values then the RB model should be trained on a training set of random parameters and if the input parameters are expected to be regular the RB model should be trained on a training set of regular parameters. As in the previous Section 4.2.4 let us make our training set Ξ_{train} using uniform grid 17 and LHS with 300 sampled parameter vectors and see how they affect the accuracy of the RB solution when the test set Ξ_{test} is constructed by simple random sampling or from an equally spaced grid respectively. Note that the test sets contains a larger number of values than the training sets. The random test



(a) The error between the high-fidelity solution $\mathbf{u_h}(\boldsymbol{\mu})$ and the reduced basis solution $\mathbf{u_N}(\boldsymbol{\mu})$ measured in the H^1-seminorm.

(b) The error between the high-fidelity solution $p_h(\boldsymbol{\mu})$ and the reduced basis solution $p_N(\boldsymbol{\mu})$ measured in the L^2-norm.

Figure 4.16: The average error between high-fidelity solutions and RB solutions measured in their natural norm. Computed on a test set of 350 random sampled parameters using a uniformly spaced training set of $7 \times 7 \times 3 \times 2 \times 2$ parameters, uniform grid 17.

set consists of 350 samples and the uniform grid test set is $8 \times 8 \times 6 \times 1 \times 1$ parameter vectors different from the training set, all values contained inside the parameter interval from Table 4.1. Also note that the reason why we use uniform grid 17, even though we concluded in Section 4.2.4 that uniform grid 15 was sufficient, is that we want the whole parameter interval for parameters μ_4 and μ_5 to be represented.

We study the accuracy of RB functions $(\mathbf{u_N}(\boldsymbol{\mu}), p_N(\boldsymbol{\mu}))$, obtained through solving equation (4.24), constructed from the training set Ξ_{train} for both the methods uniform grid 17 and LHS 4. The accuracy of the RB solutions is measured in their respective natural norm, that is the H^1-seminorm and L^2-norm respectively, yielding the errors $|\mathbf{u_h}(\boldsymbol{\mu}) - \mathbf{u_N}(\boldsymbol{\mu})|_{H^1} = \|\nabla(\mathbf{u_h}(\boldsymbol{\mu}) - \mathbf{u_N}(\boldsymbol{\mu}))\|_{L^2}$ and $\|p_h(\boldsymbol{\mu}) - p_N(\boldsymbol{\mu})\|_{L^2}$ respectively. As this is done for all parameter vectors $\boldsymbol{\mu_i}$ in the test set $\Xi_{test} = [\boldsymbol{\mu_1}, \ldots, \boldsymbol{\mu_n}]^\top$ we compute the mean of the error over all n test parameters. We then plot the errors relative to the mean of the respective high-fidelity solutions measured in their natural norm. That is we plot

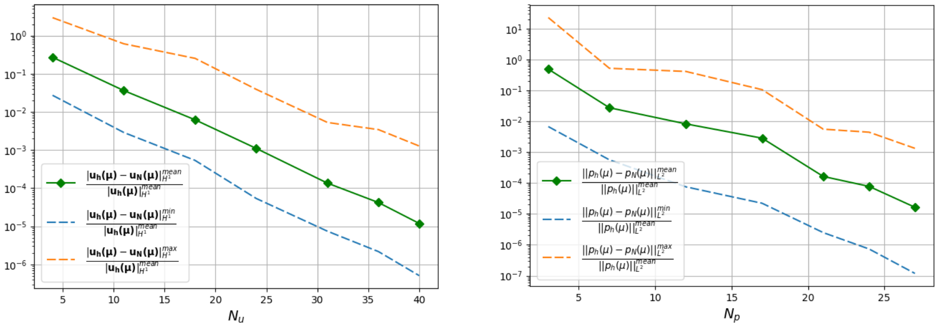
$$\frac{|\mathbf{u_h}(\boldsymbol{\mu}) - \mathbf{u_N}(\boldsymbol{\mu})|_{H^1}^{mean}}{|\mathbf{u_h}(\boldsymbol{\mu})|_{H^1}^{mean}} \qquad \frac{\|p_h(\boldsymbol{\mu}) - p_N(\boldsymbol{\mu})\|_{L^2}^{mean}}{\|p_h(\boldsymbol{\mu}))\|_{L^2}^{mean}}.$$

We also plot the largest and the smallest values of the error denoted by *max* and *min* divided with the mean of the respective high-fidelity solution measured in their natural norm, that is

$$\frac{|\mathbf{u_h}(\boldsymbol{\mu}) - \mathbf{u_N}(\boldsymbol{\mu})|_{H^1}^{min}}{|\mathbf{u_h}(\boldsymbol{\mu})|_{H^1}^{mean}} \qquad \frac{|\mathbf{u_h}(\boldsymbol{\mu}) - \mathbf{u_N}(\boldsymbol{\mu})|_{H^1}^{max}}{|\mathbf{u_h}(\boldsymbol{\mu})|_{H^1}^{mean}}$$

$$\frac{\|p_h(\boldsymbol{\mu}) - p_N(\boldsymbol{\mu})\|_{L^2}^{min}}{\|p_h(\boldsymbol{\mu}))\|_{L^2}^{mean}} \qquad \frac{\|p_h(\boldsymbol{\mu}) - p_N(\boldsymbol{\mu})\|_{L^2}^{max}}{\|p_h(\boldsymbol{\mu}))\|_{L^2}^{mean}}$$

These error computations are done for increasing number of reduced basis functions N_u and N_p. We choose the numbers N_u of the reduced velocity basis functions and the number



(a) The error between the high-fidelity solution $\mathbf{u_h}(\boldsymbol{\mu})$ and the reduced basis solution $\mathbf{u_N}(\boldsymbol{\mu})$ measured in the H^1-seminorm.

(b) The error between the high-fidelity solution $p_h(\boldsymbol{\mu})$ and the reduced basis solution $p_N(\boldsymbol{\mu})$ measured in the L^2-norm.

Figure 4.17: The average error between high-fidelity solutions and RB solutions measured in their natural norm. Computed on a test set of 350 random sampled parameters.The training set is constructed using a Latin Hypercube sampled training set of 300 parameters, LHS 4.

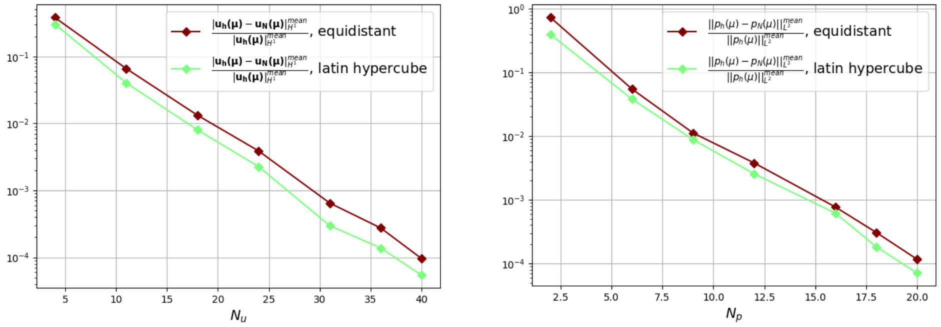
N_p of the reduced pressure basis functions such that $N_u = 2 \cdot N_p$, always making sure that both N_u is less than the rank of C_u and N_p less than the rank of C_p.

First we choose a test set of 350 parameter vectors sampled by a simple random sampling method. The error between the high-fidelity solutions and the RB solutions constructed by a training set based on the uniform grid 17 is given in Figure 4.16. In Figure 4.17 the error between the high-fidelity solutions and the RB solutions constructed by a training set based on LHS 4 is given. The mean of the error is plotted together with the maximum and the minimum of the measured errors to show the variability in the error computations over the test set.

In Figure 4.18 we compare the average errors from figures 4.16 and 4.17 and we see that the RB solutions based on a training set from a LHS method have a better accuracy than the RB solutions based on the uniform grid 17 when the test set is based on the single random sampling method.

Second we choose a test set of $8 \times 8 \times 6 \times 1 \times 1$ parameter vectors from a uniform grid method. The mean of the error between the high-fidelity solutions and the RB solutions constructed by a training set based on the uniform grid 17 is given in Figure 4.19. In Figure 4.20 the mean of the error between the high-fidelity solutions and the RB solutions constructed by a training set based on LHS 4 is given. The maximum and the minimum of the measured errors are also plotted. We observe that the min-max interval in Figure 4.20 is somewhat wider than the min-max interval in Figure 4.19 and since it is the same test set, this means that the training set from the LHS method does not cover the solution set as good as the training set from the uniform grid 17 does (for this type of test set), and hence naturally we get a wider min-max interval as the test set is a large uniform grid.

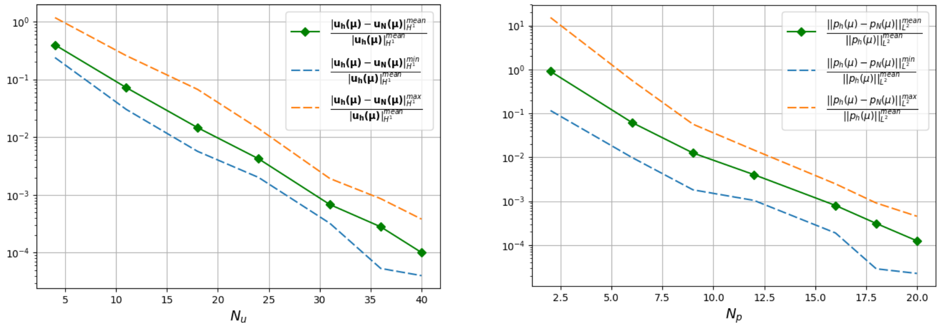
In Figure 4.21 we compare the average errors from figures 4.19 and 4.20 and we see that the RB solutions based on a training set from the uniform grid 17 most often have a better accuracy than the RB solutions based on the LHS method when the test set is based



(a) The error between the high-fidelity solution $\mathbf{u_h}(\boldsymbol{\mu})$ and the reduced basis solution $\mathbf{u_N}(\boldsymbol{\mu})$ measured in the H^1-seminorm.

(b) The error between the high-fidelity solution $p_h(\boldsymbol{\mu})$ and the reduced basis solution $p_N(\boldsymbol{\mu})$ measured in the L^2-norm.

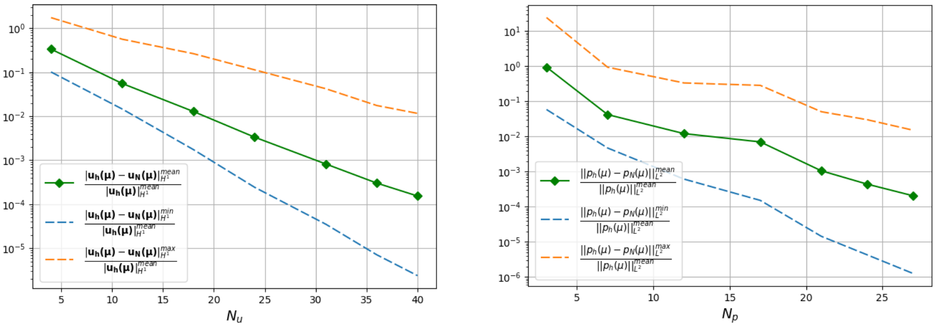
Figure 4.18: The average error between high-fidelity solutions and RB solutions measured in their natural norm. Computed on a test set of 350 random sampled parameters. The training sets are constructed using $7 \times 7 \times 3 \times 2 \times 2$ uniformly spaced parameters and 300 Latin Hypercube sampled parameters respectively.



(a) The error between the high-fidelity solution $\mathbf{u_h}(\boldsymbol{\mu})$ and the reduced basis solution $\mathbf{u_N}(\boldsymbol{\mu})$ measured in the H^1-seminorm.

(b) The error between the high-fidelity solution $p_h(\boldsymbol{\mu})$ and the reduced basis solution $p_N(\boldsymbol{\mu})$ measured in the L^2-norm.

Figure 4.19: The average error between high-fidelity solutions and RB solutions measured in their natural norm. Computed on a test set of $8 \times 8 \times 6 \times 1 \times 1$ uniformly spaced parameters using a uniformly spaced training set of $7 \times 7 \times 3 \times 2 \times 2$ parameters.



(a) The error between the high-fidelity solution $\mathbf{u_h}(\boldsymbol{\mu})$ and the reduced basis solution $\mathbf{u_N}(\boldsymbol{\mu})$ measured in the H^1-seminorm.

(b) The error between the high-fidelity solution $p_h(\boldsymbol{\mu})$ and the reduced basis solution $p_N(\boldsymbol{\mu})$ measured in the L^2-norm.

Figure 4.20: The average error between high-fidelity solutions and RB solutions measured in their natural norm. Computed on a test set of $8 \times 8 \times 6 \times 1 \times 1$ uniformly spaced parameters using a Latin Hypercube sampled training set of 300 parameters.

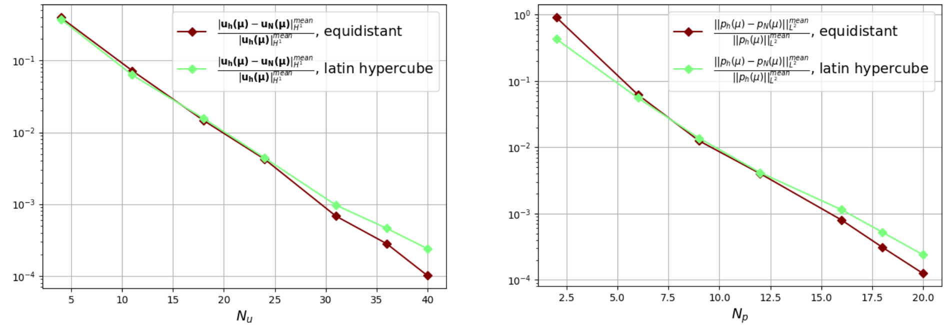
on a uniform grid.

Hence we state that it is of great importance to know whether the input parameter vector will be random or regular to compute RB solutions of best possible accuracy compared to the high-fidelity solution of the input parameters. From the previous Section 2.54 we saw that a larger number of parameter vectors are needed for the LHS method than for a uniform grid to approximate the spectrum of the solution set accurate enough. Still, the results of this section shows that the RB solutions bases on the LHS method still yields a better accuracy than RB solutions based on a uniform grid when the input parameter vector for the problem is chosen random. We have also seen that vice versa is true.

Remark 4.2.1. *Here it is in place to make some remarks on the accuracy observed. For the Figures 4.16-4.21 the best observed relative error is approximately $O(10^{-4})$. If we for some reason would like to obtain an error of, let us say $O(10^{-8})$ then we cannot use the eigenvalues and eigenvectors of a correlation matrix defined by $C = S^\top X_h S$. As we observe from the previous section, the smallest eigenvalues of C are of approximately size $O(10^{15} - 10^{16})$, that is they hit zero in machine precision. The eigenvalues of C are the same as the square of the singular values of the matrix $\tilde{S} = X_h^{1/2} S$, see Remark 2.7.2. As seen from the previous Section 4.2.4 the decay of the eigenvalues are approximately linear, and therefore the singular values corresponding to the smallest eigenvalues are approximately $O(10^{-8})$. This means that because of machine precision we loose some information stored in $\tilde{S}$ when we compute $C = \tilde{S}^\top S$ (as all eigenvalues corresponding to the singular values smaller than $O(10^{-8}$ is lost). To handle this we can solve the SVD of the matrix $\tilde{S}$ and equivalently obtain the POD basis as*

$$V = [X_h^{1/2} \tilde{\xi}_1 | \dots | X_h^{1/2} \tilde{\xi}_N],$$

where $\tilde{\xi}_i$ is the i-th left singular vector of the matrix $\tilde{S}$.



(a) The error between the high-fidelity solution $\mathbf{u_h}(\boldsymbol{\mu})$ and the reduced basis solution $\mathbf{u_N}(\boldsymbol{\mu})$ measured in the H^1-seminorm.

(b) The error between the high-fidelity solution $p_h(\boldsymbol{\mu})$ and the reduced basis solution $p_N(\boldsymbol{\mu})$ measured in the L^2-norm.

Figure 4.21: The average error between high-fidelity solutions and RB solutions measured in their natural norm. Computed on a test set of $8 \times 8 \times 6 \times 1 \times 1$ uniformly spaced parameters. The training sets are constructed using $7 \times 7 \times 3 \times 2 \times 2$ uniformly spaced parameters and 300 Latin Hypercube sampled parameters respectively.

In the following we work with a reduced basis solver based on a training set with parameters sampled from uniform grid 17.

4.2.6 Choosing the reduced basis dimension

A common way of choosing the number of reduced basis functions is the relative information content $I(N)$ equation (2.72). For a choice N the energy retained by the $r - N$ POD modes not included in the POD basis stored in V is smaller than or equal to ε_{tol}^2. Here r is the rank of the correlation matrix. Figure 4.22 shows a plot of $I(N)$ for both the velocity correlation matrix and the pressure correlation matrix. We see that the total energy of the system is mainly stored in the first few eigenvalues.

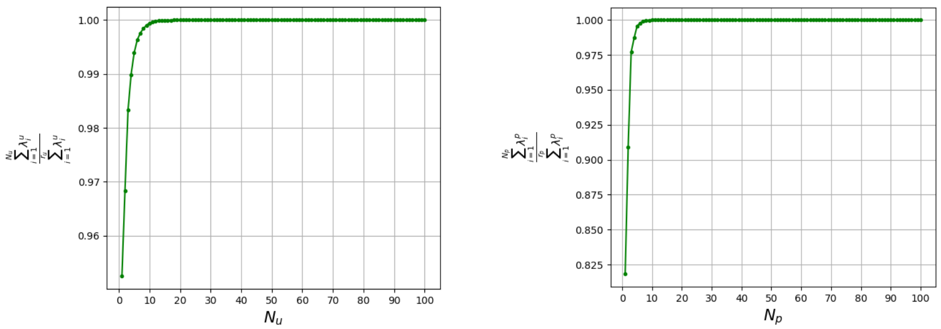
In the following we will explain how to choose the reduced basis dimension in general, before we turn back to the case for the velocity and the pressure. By choosing N POD modes to store in V, the projection error done is given by equation (2.61). Here $\mathbf{u_h}(\boldsymbol{\mu}_i) = \mathbf{u_h}^{(i)}$ for $1 \leq i \leq n_s$ is the n_s provided snapshots and $\|\cdot\|_{X_h}$ is the abstract norm associated with the natural norm of the snapshots. Assuming that these snapshots are a sufficient representation of the solution set, then, from the same equation (2.61), we get that the mean projection error for an arbitrary solution coefficient vector $\mathbf{u_h}(\boldsymbol{\mu})$, should be approximately

$$\|\mathbf{u_h}(\boldsymbol{\mu}) - VV^\top X_h \mathbf{u_h}(\boldsymbol{\mu})\|_{X_h} \approx \sqrt{\frac{\sum_{i=N+1}^{r} \lambda_i}{n_s}}, \tag{4.27}$$

where λ_i is the i-th eigenvalue of the correlation matrix.

From equation (2.36) we have that the error of the reduced basis solution can be written as

$$\|\mathbf{e_h}(\boldsymbol{\mu})\|_{X_h} \leq \|\big(\mathbf{u_h}(\boldsymbol{\mu}) - VV^\top X_h \mathbf{u_h}(\boldsymbol{\mu})\big)\|_{X_h} + \|\big(VV^\top X_h \mathbf{u_h}(\boldsymbol{\mu}) - V\mathbf{u_N}(\boldsymbol{\mu})\big)\|_{X_h}, \tag{4.28}$$



(a) The $I(N)$ function based on the correlation matrix C_u of the velocity snapshots.

(b) The $I(N)$ function based on the correlation matrix C_p of the pressure snapshots.

Figure 4.22: The function $I(N)$ from equation (4.30). Here plotted for the 100 first eigenvalues.

where we have inserted the V-orthogonal projection from definition (2.59). The second term in the right hand side of the error is minimized as discussed in Section 2.5.3. The first term of the error is minimized by the definition of the POD basis from equation (2.60) and it is (approximately) given by equation (4.27). If we assume optimal convergence for both terms of the error in equation (4.28), then the error on the left hand side of the same equation and the error in (4.27) converge optimal in the same norm and only a constant should separate them. In fact we see that this holds for both the velocity and the pressure in our case, see Figure 4.23.

We want the error in equation (4.28) to be smaller than some ε of our choice. Then as only a constant k separates this error from the error in equation (4.27) we choose the smallest N as our reduced basis dimension such that the following holds

$$\|\mathbf{e_h}(\boldsymbol{\mu})\|_{X_h} = k\sqrt{\frac{\sum_{i=N+1}^{r}\lambda_i}{n_s}} \leq \varepsilon. \tag{4.29}$$

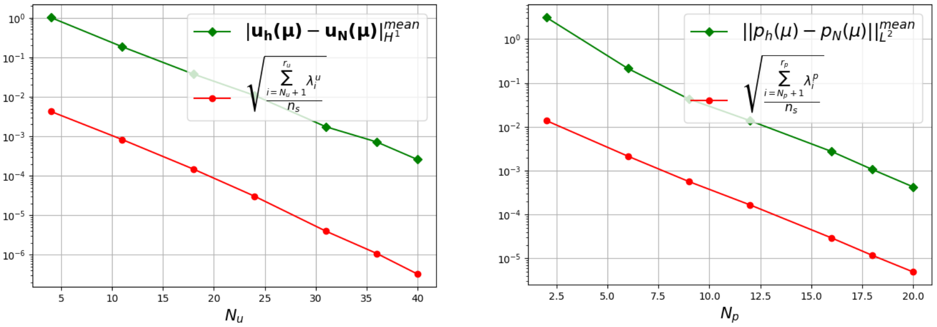
Note that since the relative information content in equation (2.72) is given as

$$I(N) = \frac{\sum_{i=1}^{N}\lambda_i}{\sum_{i=1}^{r}\lambda_i} = 1 - \frac{\sum_{i=N+1}^{r}\lambda_i}{\sum_{i=1}^{r}\lambda_i}, \tag{4.30}$$

then the condition for choosing the reduced basis dimension in equation (4.29) relates to the relative information content as:
choose the smallest N such that the following holds

$$I(N) \geq 1 - \frac{n_s}{\sum_{i=1}^{r}\lambda_i}\frac{\varepsilon^2}{k^2}. \tag{4.31}$$

From Figure 4.23 we have that for the two constants $k_1 \approx 5 \cdot 10^2$ and $k_2 \approx 10^2$ we get



(a) The error of the reduced velocity.

(b) The error of the reduced pressure.

Figure 4.23: The error from equation (4.28) plotted against the error in equation (4.27) for different choises of N reduced basis functions.

the error of the velocity and the pressure respectively

$$||\mathbf{e}_\mathbf{h}^\mathbf{u}(\boldsymbol{\mu})||_{X_{h,u}} = k_1\sqrt{\frac{\sum_{i=N_u+1}^{r_u} \lambda_i^u}{n_s}} \leq \varepsilon_u, \qquad ||\mathbf{e}_\mathbf{h}^\mathbf{p}(\boldsymbol{\mu})||_{X_{h,p}} = k_2\sqrt{\frac{\sum_{i=N_p+1}^{r_p} \lambda_i^p}{n_s}} \leq \varepsilon_p, \tag{4.32}$$

where λ_i^u, λ_i^p are the i-th eigenvalue of the velocity correlation matrix and the pressure correlation matrix respectively and r_u, r_p are likewise their rank. We choose $\varepsilon = \varepsilon_u = 10^{-3}$ for the accuracy of the reduced velocity solution, insert it in equation (4.31), and choose the smallest N_u such that it holds. Then we need $N_u = 34$ which matches what we observe in Figure 4.23a. In the same manner we choose $\varepsilon = \varepsilon_p = 10^{-3}$ for the accuracy of the reduced pressure solution, insert it in equation (4.31), and choose the smallest N_p such that it holds. Then we need $N_p = 18$ which matches what we observe in Figure 4.23b.

4.2.7 Computational performance

In the end, the whole point of RB methods is to reduce computational time (CT) of the problem at hand. As described in Section 2.5.4 the computation of a RB solution consists of a expensive offline stage as it depends on the high-fidelity dimension $N_h + M_h$ and a fast online stage independent of $N_h + M_h$. To measure computation time of the high-fidelity solver, the offline stage and the online stage, the calculations are done on a HPC cluster. This is done to obtain as accurate measures of computation time as possible. This cluster uses a job scheduling system named Slurm and hence no other processes like background programs and other disturbances can interupt the computations and hence they get more accurate. See [6] for more information.

The steady Stokes problem is solved on a geometry partitioning of 80×80 Taylor-Hood elements with polynomial degree $P_u = 2$ for the velocity basis and $P_p = 1$ for the pressure basis. This gives $N_h = 2 \cdot 25499$ degrees of freedom for the velocity field and $M_h = 6480$ degrees of freedom for the pressure field. The computational time of the high-fidelity solution using the finite element method (assembly and solution) takes on average 1min

Table 4.3: Mean computation time and standard deviation of the FE model and the RB model.

Model	Number of runs	Mean CT	standard deviation of CT
FE (assembly+solution)	200	1 min 51 s	2.144412 s
RB offline	10	45 min 51 s	28.924272 s
RB online	200	0.000343 s	0.000049 s

and 51s with a standard deviation of 2.1s. We create the RB solutions using the uniform grid 15 as the training set. As we want a error of $O(10^{-3})$ for the RB solutions, we know from the previous Section 4.2.6 that we should choose $N_u = 34$ reduced basis functions for the velocity field and $N_p = 18$ reduced basis functions for the pressure field. This gives us a reduction of 1500 : 1 in the degrees of freedom for the velocity field and a reduction of 360 : 1 for the pressure field. The offline computation time of RB method was on average 45min 51s with a standard deviation of 28.9s. The online computation time was on average $3.43 \cdot 10^{-4}$s with a standard deviation of $4.9 \cdot 10^{-5}$. This provides a speedup of order $O(10^5)$. The numerical results are summarized in Table 4.3 together with the number of runs used to calculate each computation time and the standard deviation of the calculated times.

The numerical results were obtained on a computing server of the type Dell PE630 with 2× Intel Xeon E5-2630 v4 10 cores 2.20GHz, 128GB RAM, 300GB local disk. The solver is built upon the Python packages NumPy and Ntils which both contain routines for parallel programming. The packages are in a way self optimizing as using these routines the packages will run the code in parallel in case this is faster, see [5] and [61] for more details.

4.2.8 Stability of solutions

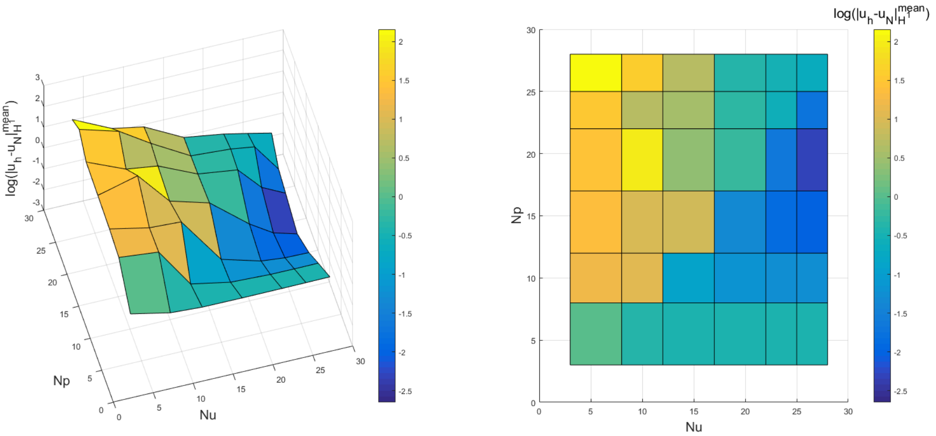
As introduced in Section 2.8.2 we must enrich the velocity space in order to be certain the reduced solutions are stable. Still we have up to this point not enriched the velocity space and still our reduced solutions seem to be stable. This makes us wonder whether there is possible to compute stable reduced basis solutions, that we know are stable, without the enrichment of the velocity space discussed in Section 2.8.

Several authors seem to be doing RB methods for the steady Stokes and the Navier-Stokes equations with the same number of reduced basis functions for both the velocity field and the pressure field, that is $N_u = N_p$, see e.g. [49, 8, 23]. This thesis on the other hand has so far only studied the case where $N_u = 2 \cdot N_p$. In the following we conduct the error analysis over a grid of different values for N_u and N_p, exploring other relations than just $N_u = 2 \cdot N_p$.

In particular we start of by the number of RB velocity functions

$$N_u = [3, 8, 12, 17, 22, 25, 28], \qquad N_p = [3, 8, 12, 17, 22, 25, 28],$$

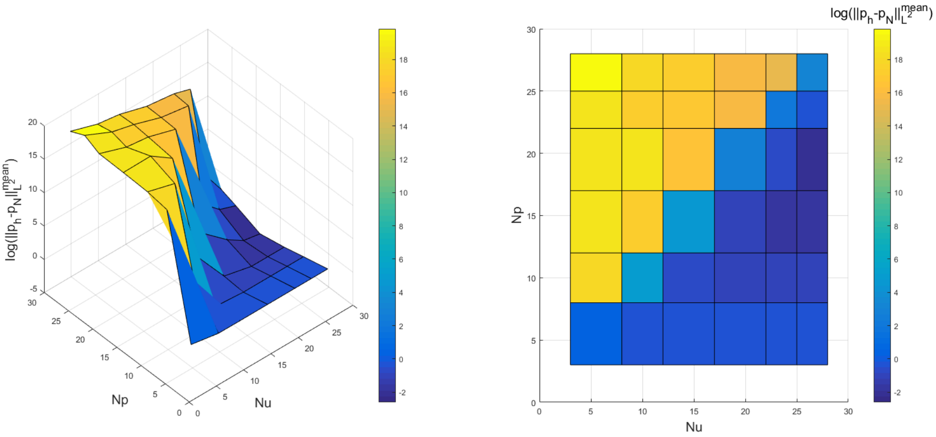
and we conduct error analysis on the grid $N_u \times N_p$. Note that this grid has the diagonal $N_u = N_p$ and this is the choice used for the RB solver in e.g. [8], where enrichment of the velocity space by supremizers is used. The training set used to compute the RB solutions is the uniform grid 17 from Table 4.1 and the test set used to compute the mean of the



(a) 3D view of the mean of the error.

(b) 2D view of the mean of the error.

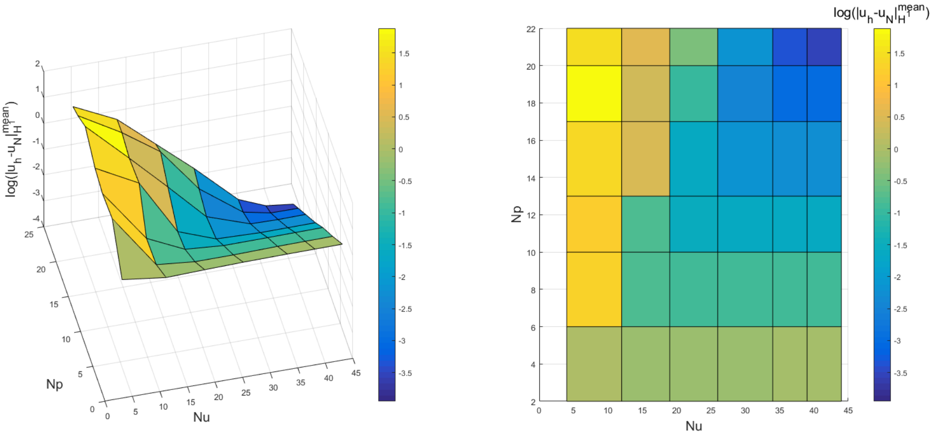
Figure 4.24: Error analysis of the reduced basis velocity $\mathbf{u_N}(\boldsymbol{\mu})$. Mean error measured in the H^1-seminorm for different number of velocity reduced basis functions N_u and pressure reduced basis functions N_p. Equal spacing in N_u and N_p such that the diagonal of the 2D surface is $N_u = N_p$.



(a) 3D view of the mean of the error.

(b) 2D view of the mean of the error.

Figure 4.25: Error analysis of the reduced basis pressure $p_N(\boldsymbol{\mu})$. Mean error measured in the H^1-seminorm for different number of velocity reduced basis functions N_u and pressure reduced basis functions N_p. Equal spacing in N_u and N_p such that the diagonal of the 2D surface is $N_u = N_p$.



(a) 3D view of the mean of the error.

(b) 2D view of the mean of the error.

Figure 4.26: Error analysis of the reduced basis velocity $\mathbf{u_N}(\boldsymbol{\mu})$. Mean error measured in the H^1-seminorm for different number of velocity reduced basis functions N_u and pressure reduced basis functions N_p. Different spacing in N_u and N_p such that the diagonal of the 2D surface is $N_u = 2 \cdot N_p$.

error is a $(8 \times 8 \times 4 \times 1 \times 1)$ uniform grid. The error between the reduced solutions and the high-fidelity solutions are measured in their natural norm, that is the H^1-seminorm for the reduced velocity and the L^2-norm for the reduced pressure.

Figure 4.24 shows that the diagonal $N_u = N_p$ does not yield a decreasing error for increasing number of RB functions for the reduced velocity, and the same holds for the corresponding reduced pressure in Figure 4.25. We observe that we need $N_u > N_p$ in order to obtain a decreasing error for increasing number of RB functions. Do also note that we need $N_p > 8$. Observe from Figure 4.25b that the region on and above the diagonal, that is $N_p \geq N_u$, we have infinite error for the reduced pressure, that is the reduced LBB-condition (2.75) is not satisfied and the linear system (4.24) is singular.

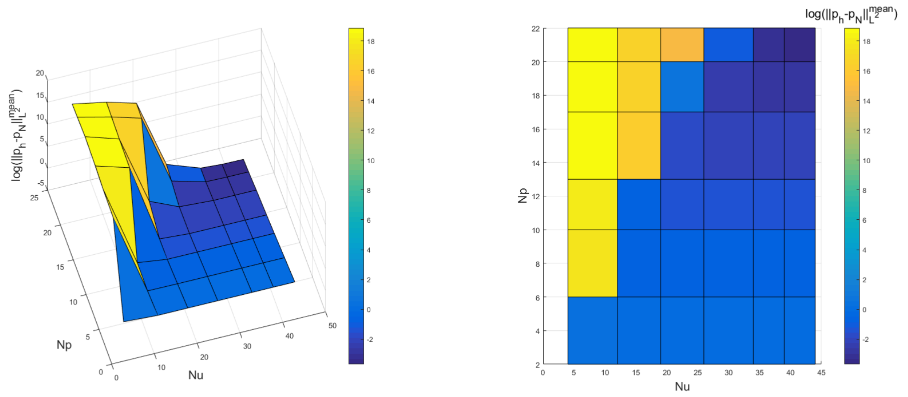
Next we look at the number of RB velocity functions N_u and N_p such that

$$N_u = [4, 12, 19, 26, 34, 39, 44] \qquad N_p = [2, 6, 10, 13, 17, 20, 22].$$

We conduct the error analysis on the grid $N_u \times N_p$. Note that this grid has $N_u = 2 \cdot N_p$ on its diagonal. In this case the diagonal in figures 4.26 and 4.27 gives a decreasing error for both the reduced velocity and the reduced pressure confirming the results from the previous Section 4.2.5.

Still it is a thin line to walk, we need $N_u > N_p$ in order to have a decreasing error for increasing number of basis functions. Figures 4.26 and 4.27 shows clearly that just of diagonal, that is right above the diagonal $N_u = 2 \cdot N_p$ increasing N_p slightly, the pressure is unstable and the velocity does not converge.

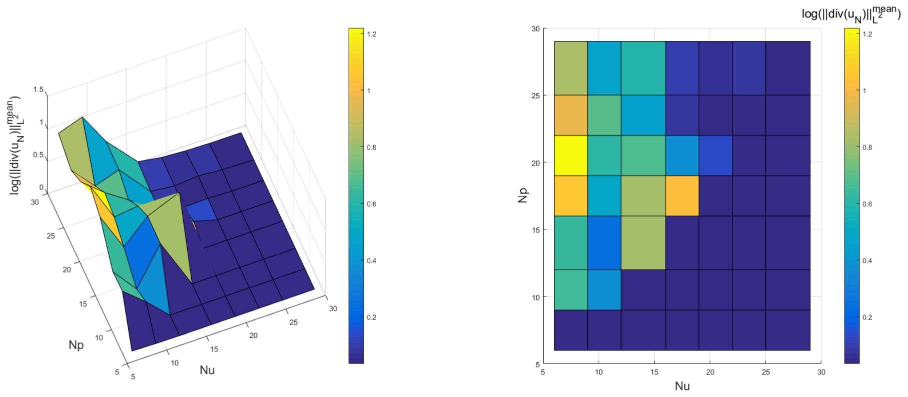
To complement the error analysis we look at the divergence property of the reduced velocity solution. From the problem description (4.14) we have that the divergence of the velocity should be zero. This is only true over the whole domain and not point-wise



(a) 3D view of the mean of the error.

(b) 2D view of the mean of the error.

Figure 4.27: Error analysis of the reduced basis pressure $p_N(\boldsymbol{\mu})$. Mean error measured in the H^1-seminorm for different number of velocity reduced basis functions N_u and pressure reduced basis functions N_p. Different spacing in N_u and N_p such that the diagonal of the 2D surface is $N_u = 2 \cdot N_p$.



(a) 3D view of the divergence.

(b) 2D view of the divergence.

Figure 4.28: Mean divergence of the reduced basis velocity $\mathbf{u_N}(\boldsymbol{\mu})$, measured in the H^1-seminorm for different number of velocity reduced basis functions N_u and pressure reduced basis functions N_p. Equal spacing in N_u and N_p such that the diagonal of the 2D surface is $N_u = N_p$.

because the weak formulation (4.3) only requires weak divergence. The map (4.15) does in fact not preserve divergence of vector fields either. Hence the divergence of the reduced velocity should not be exactly zero. In order to preserve divergence a different divergence-preserving transformation must be used as done by Evans and Hughes in [27, 29, 28]. Such a transformation is called Piola transform and is beyond the scope of this thesis.

Still we expect the divergence of the reduced velocity to be close to zero and we plot the divergence for the same grid as in Figure 4.24, seen in Figure 4.28. We observe the divergence of the velocity to be closer to zero for the part of the domain where $N_u > N_p$ which is consistent with the observations already done. The divergence increase when N_p is large compared to N_u.

4.2.9 Supremizer

In this section we introduce a RB method with supremizer stabilization. This is done so that we can compare it to the reduced basis method used in this thesis with respect to both error analysis and computational performance.

As discussed in Section 2.8 there is two possible ways to enrich the reduced velocity space, by exact supremizer and by approximate supremizer. The exact method can be proven to satisfy the reduced inf-sup condition (2.75), but this method gives a parameter-dependent transformation matrix V_s for the supremizers. Thus it is not possible to conduct the offline-online decomposition. Due to computation time, this is not interesting. The approximate supremizer method makes it possible to conduct the offline-online decomposition, but it is not possible to show rigorously that the reduced inf-sup condition (2.75) holds. Ballarin et al. give heuristic criteria for the approximate supremizer enrichment such that the reduced inf-sup condition holds, see [8].

The supremizers are computed from the linear system

$$X_{h,\mathbf{u}}\mathbf{t}_{\mathbf{h}}^{\boldsymbol{\mu}}(\mathbf{p_h}(\boldsymbol{\mu_i}) = B_h^{\top}(\boldsymbol{\mu})\mathbf{p_h}(\boldsymbol{\mu_i}) \quad 1 \leq i \leq n_s, \tag{4.33}$$

where n_s is the number of snapshots, $\mathbf{p_h}(\boldsymbol{\mu_i})$ is the pressure snapshot corresponding to parameter $(\boldsymbol{\mu_i})$ and $X_{h,\mathbf{u}}$ is the matrix associated with the scalar product in the velocity space $\mathbb{V} = [H^1(\Omega)]^2$, that is $(X_{h,\mathbf{u}})_{ij} = (\nabla\boldsymbol{\phi_i}, \nabla\boldsymbol{\phi_j})_{L^2(\Omega)}$. This is equivalent to the linear system (2.19). We store all the solutions of equation (4.33) in the supremizer snapshot matrix

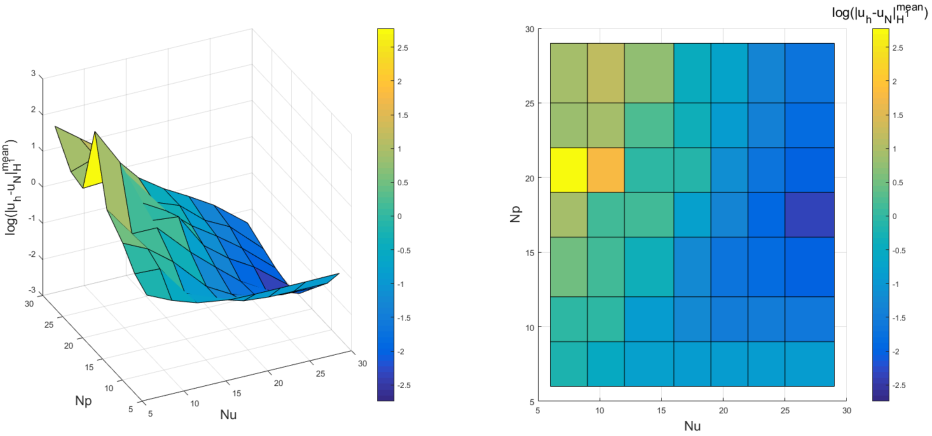
$$S_s = [\mathbf{t_h}^{\boldsymbol{\mu}_1}(\mathbf{p_h}(\boldsymbol{\mu}_1))| \ldots |\mathbf{t_h}^{\boldsymbol{\mu}_{ns}}(\mathbf{p_h}(\boldsymbol{\mu}_{\boldsymbol{n_s}}))],$$

and compute a POD basis from the correlation matrix $C_s = S_s^{\top} X_{h,\mathbf{u}} S_s$. The POD modes of the transformation matrix $V_s = [\boldsymbol{\xi_1^s}| \ldots |\boldsymbol{\xi_{N_s}^s}] \in \mathbb{R}^{N_h \times N_s}$ for the supremizers are computed as

$$\boldsymbol{\xi_i^s} = \frac{1}{\sqrt{\lambda_i^s}} S_s \boldsymbol{\psi_i^s} \quad 1 \leq i \leq N_s$$

Now we define the new transformation matrix $\tilde{V}_u = [V_u \; V_s] \in \mathbb{R}^{N_h \times (N_u + N_s)}$ and the reduced matrices and vectors in equation (4.24) becomes

$$A_N(\boldsymbol{\mu}) = \tilde{V}_u^{\top} A_h(\boldsymbol{\mu}) \tilde{V}_u = \begin{bmatrix} V_u^{\top} A_h(\boldsymbol{\mu}) V_u & V_u^{\top} A_h(\boldsymbol{\mu}) V_s \\ V_s^{\top} A_h(\boldsymbol{\mu}) V_u & V_s^{\top} A_h(\boldsymbol{\mu}) V_s \end{bmatrix} = \begin{bmatrix} A_{N,uu}(\boldsymbol{\mu}) & A_{N,us}(\boldsymbol{\mu}) \\ A_{N,su}(\boldsymbol{\mu}) & A_{N,ss}(\boldsymbol{\mu}) \end{bmatrix} \in \mathbb{R}^{(N_u+N_s)\times(N_u+N_s)}, \tag{4.34}$$



(a) 3D view of the mean of the error.

(b) 2D view of the mean of the error.

Figure 4.29: Error analysis of the reduced basis velocity $\mathbf{u_N}(\boldsymbol{\mu})$. Mean error measured in the H^1-seminorm for different number of velocity reduced basis functions N_u and pressure reduced basis functions N_p. Equal spacing in N_u and N_p such that the diagonal of the 2D surface is $N_u = N_p$ and $N_s = \left\lceil \frac{N_p+1}{2} \right\rceil$.

$$B_N(\boldsymbol{\mu}) = V_p^\top B_h(\boldsymbol{\mu})\tilde{V}_u = \begin{bmatrix} V_p^\top B_h(\boldsymbol{\mu})V_u & V_p^\top B_h(\boldsymbol{\mu})V_s \end{bmatrix} = \begin{bmatrix} B_{N,pu}(\boldsymbol{\mu}) & B_{N,ps}(\boldsymbol{\mu}) \end{bmatrix} \in \mathbb{R}^{N_p \times (N_u+N_s)}, \tag{4.35}$$

$$\mathbf{f_{1N}}(\boldsymbol{\mu}) = \tilde{V}_u^\top \mathbf{f_{1h}}(\boldsymbol{\mu}) = \begin{bmatrix} V_u^\top \mathbf{f_{1h}}(\boldsymbol{\mu}) \\ V_s^\top \mathbf{f_{1h}}(\boldsymbol{\mu}) \end{bmatrix} = \begin{bmatrix} \mathbf{f_{1N,u}}(\boldsymbol{\mu}) \\ \mathbf{f_{1N,s}}(\boldsymbol{\mu}) \end{bmatrix} \in \mathbb{R}^{N_u+N_s}, \tag{4.36}$$

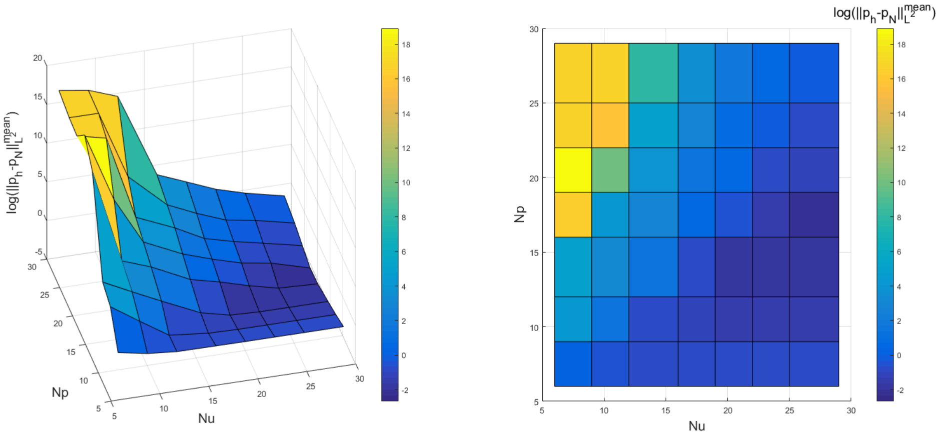
$$\mathbf{f_{2N}}(\boldsymbol{\mu}) = V_p^\top \mathbf{f_{2h}}(\boldsymbol{\mu}) \in \mathbb{R}^{N_p}. \tag{4.37}$$

The affine combinations of the reduced matrices and vectors in equations (4.34)-(4.37) follows directly from inserting the affine combinations of the high-fidelity matrices and vectors in equation (4.19). Then it is possible to do an offline-online decomposition of the reduced system.

We solve the reduced linear system (4.24) for the new reduced matrices and vectors in equations (4.34)-(4.37). As done in section (4.2.8), we will in the following conduct a error analysis for different choices of N_u, N_s and N_p. For a physical and geometric parameter-dependence Ballarin et al. states by heuristics that $N_s > N_p/2$ gives a reliable method for the Navier-Stokes equations, [8]. Motivated by this we choose $N_s = \left\lceil \frac{N_p+1}{2} \right\rceil$ and we conduct the error analysis on the grid $N_u \times N_p$ such that

$$N_u = [6, 9, 12, 16, 19, 22, 25, 29] \qquad N_p = [6, 9, 12, 16, 19, 22, 25, 29].$$

The training set used to compute the RB solutions is the uniform grid 17 from Table 4.1 and the test set used to compute the mean of the error is a $(8 \times 8 \times 4 \times 1 \times 1)$ uniform grid.



(a) 3D view of the mean of the error.

(b) 2D view of the mean of the error.

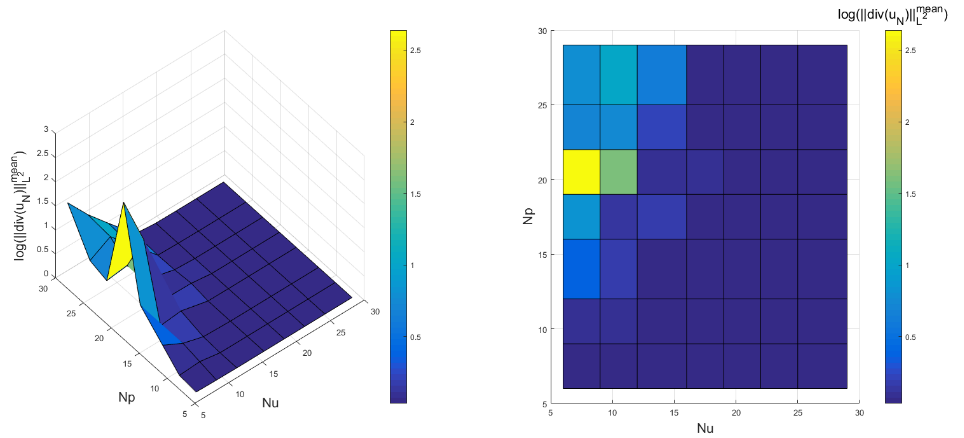
Figure 4.30: Error analysis of the reduced basis pressure $p_N(\boldsymbol{\mu})$. Mean error measured in the H^1-seminorm for different number of velocity reduced basis functions N_u and pressure reduced basis functions N_p. Equal spacing in N_u and N_p such that the diagonal of the 2D surface is $N_u = N_p$ and $N_s = \left\lceil \frac{N_p+1}{2} \right\rceil$.

The error between the reduced solutions and the high-fidelity solutions are measured in their natural norm, that is the H^1-seminorm for the reduced velocity and the L^2-norm for the reduced pressure.

From Figure 4.29 we observe that the error of the reduced velocity decrease for increasing number of reduced basis functions N_u and N_p. Observe that as for the case in Figure 4.24 for low N_u, increasing N_p also increases the error slightly. From Figure 4.30b observe that in the top left corner of the domain, we observe that the pressure still goes towards infinity and the reduced inf-sup condition (2.75) is not satisfied. Comparing Figure 4.30 to 4.25 the instability of the pressure seems to creep back since we introduced the supremizer stabilization. Seen in Figure 4.31 the jump of the divergence in the top left corner confirm the resent thoughts. In the region with unstable pressure we get a stable velocity solution, but at the expense of the divergence.

As the reduced pressure was not stable for the whole grid in Figure 4.30, we increase N_s such that $N_s = N_p$. Besides this we study exactly the same case as above. As observed in Figure 4.32 the error of the reduced velocity behaves pretty much as in Figure 4.29. What is interesting, is what we observe in Figure 4.33 where the instability of the pressure has almost disappeared. This is also reflected in Figure 4.34 where the divergence is much lower compared to to what we observe in Figure 4.31 for the case where $N_s = \left\lceil \frac{N_p+1}{2} \right\rceil$ and also in Figure 4.28 where $N_s = 0$.

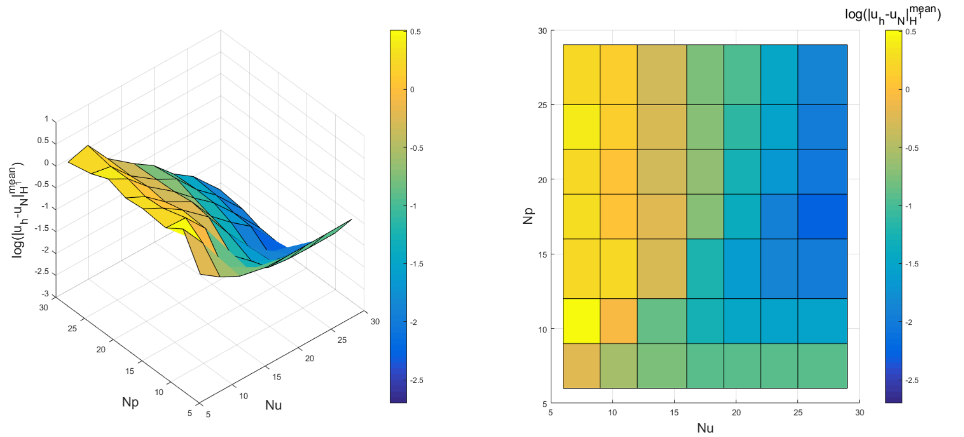
Some last remarks on this section. As long as we choose N_u and N_p carefully, we observe that it is possible to obtain the same convergence in the error of the reduced velocity approximation with and without stabilization. This is also true for the reduced pressure. But as observed, the reduced pressure is much more sensitive to the choice of



(a) 3D view of the divergence.

(b) 2D view of the divergence.

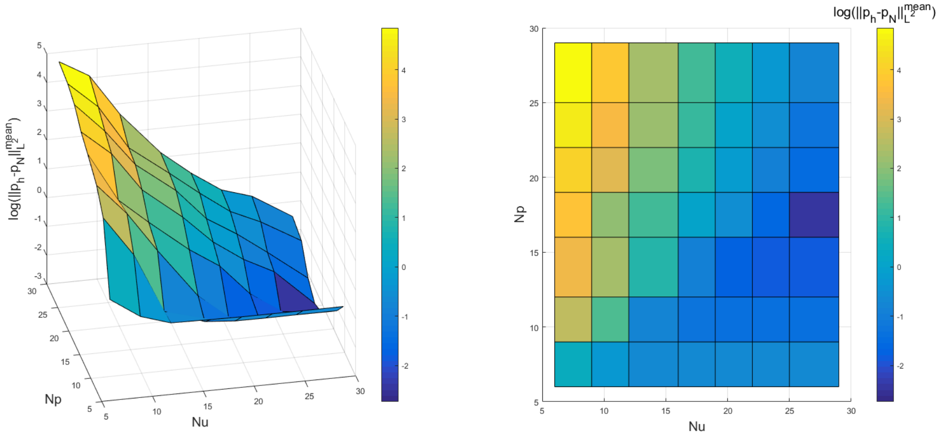
Figure 4.31: Mean divergence of the reduced basis velocity $\mathbf{u_N}(\boldsymbol{\mu})$, measured in the H^1-seminorm for different number of velocity reduced basis functions N_u and pressure reduced basis functions N_p. Equal spacing in N_u and N_p such that the diagonal of the 2D surface is $N_u = N_p$ and $N_s = \left\lceil \frac{N_p+1}{2} \right\rceil$.



(a) 3D view of the mean of the error.

(b) 2D view of the mean of the error.

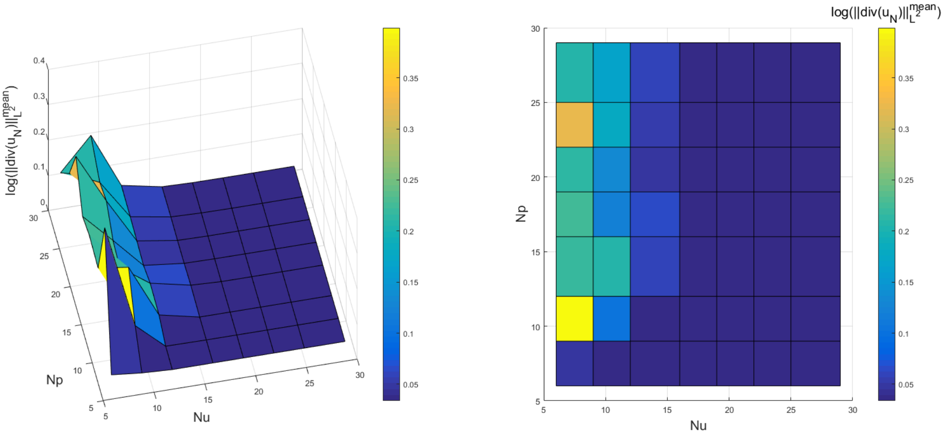
Figure 4.32: Error analysis of the reduced basis velocity $\mathbf{u_N}(\boldsymbol{\mu})$. Mean error measured in the H^1-seminorm for different number of velocity reduced basis functions N_u and pressure reduced basis functions N_p. Equal spacing in N_u and N_p such that the diagonal of the 2D surface is $N_u = N_p$ and $N_s = N_p$.



(a) 3D view of the mean of the error.

(b) 2D view of the mean of the error.

Figure 4.33: Error analysis of the reduced basis pressure $p_N(\boldsymbol{\mu})$. Mean error measured in the H^1-seminorm for different number of velocity reduced basis functions N_u and pressure reduced basis functions N_p. Equal spacing in N_u and N_p such that the diagonal of the 2D surface is $N_u = N_p$ and $N_s = N_p$.



(a) 3D view of the divergence.

(b) 2D view of the divergence.

Figure 4.34: Mean divergence of the reduced basis velocity $\mathbf{u_N}(\boldsymbol{\mu})$, measured in the H^1-seminorm for different number of velocity reduced basis functions N_u and pressure reduced basis functions N_p. Equal spacing in N_u and N_p such that the diagonal of the 2D surface is $N_u = N_p$ and $N_s = N_p$.

number of reduced basis functions for the case without supremizer stabilization.

4.2.10 Computation time with supremizer

From equations (4.34)-(4.37) we understand that the reduced solver becomes slightly more complicated when introducing the supremizer stabiization. In particluar we now need to solve one additional system in the offline stage, namely the linear system (4.33). Also the linear system (4.24) changes size from $(N_u+N_p)\times(N_u+N_p)$ to $(N_u+N_p+N_s)\times(N_u+N_p+N_s)$ which affects the online stage. As the whole point of RB methods is to reduce computational time (CT) of the problem at hand we should measure the computation time for the reduced basis solver with supremizer stabilization as we did for the reduced basis solver without supremizer stabilization.

The calculations of the computation times is carried out in the same way as in Section 4.2.7. The steady Stokes problem is solved on a geometry partitioning of 80×80 Taylor-Hood elements with polynomial degree $P_u = 2$ for the velocity basis and $P_p = 1$ for the pressure basis. This gives $N_h = 2 \cdot 25499$ degrees of freedom for the velocity field and $M_h = 6480$ degrees of freedom for the pressure field. The computational time of the high-fidelity solution using the finite element method (assembly and solution) takes on average 1min and 54s with a standard deviation of 1.9s. Note that this is approximately the same as in Table 4.3 as it should be.

We create the RB solutions using the uniform grid 15 as the training set. As we want an error of $O(10^{-3})$ for the RB solutions, we know from Section 4.2.6 that we should choose $N_u = 34$ reduced basis functions for the velocity field and $N_p = 18$ reduced basis functions for the pressure field. As observed in Section 4.2.9, the supremizer stabilization does not influence the accuracy of the velocity, at least not in the region already stable. The results from Section 4.2.6 holds. We also choose $N_s = 18$.

As the N_s supremizers enriches the reduced velocity basis, we get a reduction of $981 : 1$ in the degrees of freedom for the velocity field and a reduction of $360 : 1$ for the pressure field. The offline computation time of RB method was on average 49min 27s with a standard deviation of 13.2s. The online computation time was on average $7.28 \cdot 10^{-4}$s with a standard deviation of $8.8 \cdot 10^{-5}$. This provides a speedup of order $O(10^5)$. The numerical results are summarized in Table 4.4 together with the number of runs used to calculate each computation time and the standard deviation of the calculated times. The difference between the offline computation time between the two methods are $4-5$min and the online computation time seems to have doubled.

Table 4.4: Mean computation time and standard deviation of the FE model and the RB model using a supremizer stabilization.

Model	Number of runs	Mean CT	standard deviation of CT
FE (assembly+solution)	200	1 min 54 s	1.904436 s
RB offline	15	49 min 27 s	13.216114 s
RB online	200	0.000728 s	0.000088 s

Chapter 5

Conclusion

Throughout this thesis, we have built a finite element solver and a reduced basis solver for a steady Stokes flow around a NACA airfoil. We compared the performance of the reduced basis solver to the finite element solver in terms of both numerical error analysis and computation time.

The first step of this master thesis, as presented in the introduction, was to build a finite element solver for the Stokes equations to provide this thesis with the desired numerical results. A solver is implemented for both a unit square verification example and for a parametrized Stokes flow around a NACA airfoil. As seen by the convergence results in Figure 3.5 and Figure 3.6 and the relative error in Figure 4.5 the solver must be said to be of good enough accuracy. All numerical results in this thesis are provided directly or indirectly from this solver.

As we solve the parametrized steady Stokes equations where the geometry is parameter-dependent, the second step of this thesis was to express the reduced system matrices and vectors as affine combinations. This was a must in order to ensure a time-effective offline-online decomposition of the reduced basis (RB) solver. As seen in Section 4.2.2 the parametric map $\mathcal{F}$ made it possible to obtain all matrices and vectors of the high-fidelity system as affine combinations, see equation (4.19). Therefore the equations of the reduced basis system are also affine combinations as seen in equation (4.26) and we have ensured a time-effective offline-online decomposition.

We also wanted to investigate how to sample the parameters in order to span the solution set of the high-fidelity problem in a best possible way. For this particular problem we learned that the following holds: in order to create snapshots that span the whole solution set, it is more important to vary the geometric parameters over the parameter space than the physical parameters. This is due to the fact that changing the geometry changes the characteristics of the flow more drastic than the physical parameters do. Therefore, choosing the input parameters from a uniform grid with more points in the parameter directions of the geometric parameters seems to be one of the best strategies for parameter sampling. This method of uniform sampling captures the variability of the solution set better than sampling from Gauss-Legendre, Latin Hypercube, and simple random sampling. This is interesting since other problems such as the steady heat conduction-convection problem in [49] clearly shows that a Latin Hypercube sampling captures the variability of the solution set better than any other regular grid. This means that without any further analysis of parameter sampling for different problems, one should always consider different sampling methods when considering to make a reduced model.

The reduced basis solver was trained on two different training sets. One training set of a sufficient number of uniform parameter configurations and the other training set of a sufficient number of LHS parameter configurations. This showed that when the test set of input parameters were equidistant, the reduced basis solver showed the best accuracy when based on the uniform training set, see Figure 4.21. Also, when the test set of input parameters were chosen by simple random sampling, the reduced basis solver showed the best accuracy when based on the training set of LHS points, see Figure 4.18. Indeed, we conclude that the training set must be chosen sufficiently large to capture the variability of the solution set, but not too large because of computational efficiency. In addition, the sampling method for the training set should be based on the expected input parameters. For random input parameters, the training set should be sampled from a random sampling procedure. When expecting regular input parameters the training set should be sampled from a uniform grid for this particular problem.

These results were conducted using a reduced basis solver without supremizer stabilization. For this analysis I suggested to use N_u reduced velocity basis functions and N_p reduced pressure basis functions such that $N_u = 2 \cdot N_p$. To confirm that this choice is stable for the problem at hand, the mean error was plotted for different number of reduced basis functions. As observed from figures 4.26 and 4.27 both errors seem to converge along the diagonal, that is where $N_u = 2 \cdot N_p$. Do note that this is a very thin line to walk as increasing the number of reduced pressure basis functions slightly gives unstable solutions, as illustrated in figures 4.24 and 4.25, where the diagonal $N_u = N_p$ gives unstable solutions.

For the desired accuracy of the high-fidelity solver, the computations were done on a grid of 80×80 elements with $N_h = 2 \cdot 25499$ degrees of freedom for the high-fidelity velocity and $M_h = 6480$ degrees of freedom for the high-fidelity pressure. In comparison, $N_u = 34$ reduced velocity basis functions and $N_p = 18$ reduced pressure functions were necessary for the desired accuracy of the reduced basis model. This is a reduction of $1500 : 1$ in degrees of freedom for the velocity and $360 : 1$ in degrees of freedom for the pressure. The computation time for the high-fidelity solver was on average 1min 51s. The computation time for the online stage of the reduced basis solver was on average $3.43 \cdot 10^{-4}$s. This provides an attractive speedup of $O(10^5)$ for the reduced basis solver. The computation time for the offline stage was on average 45min 51s.

As the reduced basis solver was not stable for all possible number of reduced basis functions, an additional solver was built by enriching the reduced velocity space with supremizers to make the solver more stable [8]. By choosing $N_s = N_p/2$ supremizers, we learned that a larger portion of the grid from Figure 4.25 (made by different number of reduced velocity basis functions and different number of reduced pressure functions) becomes stable. This is seen by comparing Figure 4.30 to Figure 4.25. Further increasing the number of supremizers to $N_s = N_p$ makes approximately the whole grid stable, comparing Figure 4.33 to Figure 4.30.

A reduced basis solver with supremizer stabilization comes at a cost. First of all, an additional linear system was solved for each snapshot created, see equation (4.33). Still, as the size of this linear system was $N_h \times N_h$ it involved less operations than the high-fidelity system in equation (3.45) of size $(N_h + M_h) \times (N_h + M_h)$ also solved for each snapshot, so the increased computation time was not significant. Second of all the reduced linear system in equation (4.24), solved in the online stage, was larger for a reduced basis solver with supremizer stabilization than without. In fact, this linear system increase from the size $(N_u + N_p) \times (N_u + N_p)$ without supremizer to $(N_u + N_s + N_p) \times (N_u + N_s + N_p)$ with supremizers.

Of this particular reason, we have measured the time complexity also for the reduced solver with supremizer stabilization. We have used $N_u = 34$ reduced velocity basis functions, $N_p = 18$ reduced pressure basis functions and $N_s = 18$ supremizers where the latter are used to enrich the velocity space. For the same high-fidelity system as earlier, this gave a reduction in degrees of freedom of $981 : 1$ for the velocity and $360 : 1$ for the pressure. The computation time for the online stage of the reduced basis solver was on average $7.28 \cdot 10^{-4}$s. This is approximately the double of the computation time for the solver without supremizers, but still, it provided a speedup of $O(10^5)$ for the reduced basis solver. The computation time for the offline stage was on average 49min 27s, this is an increase of approximately 7% of the time spent in the offline stage.

This means that in this case, building the reduced basis solver with supremizer stabilization adds unnecessary complexity to the solver as equally good results are obtained without the supremizers. Building the solver without supremizers requires that a further stability analysis is done in order to not use an unstable combination of N_u reduced velocity basis functions and N_p reduced pressure basis functions. But this is in fact also required by the solver with supremizers, as the wrong choice of number of supremizers could give unstable solutions. Still, the solver with supremizers gives you a much better choice of how to choose your reduced basis functions.

Future work

To expand the stability analysis provided here, an analysis of the stability factor $\beta_N(\boldsymbol{\mu})$ should be provided by checking its value for different numbers of reduced basis functions N_u and N_p. This is a good way of providing numerical evidence of stability, and could perhaps be used to provide bounds on the number of reduced basis functions N_u and N_p for which choices that gives stable solutions or not. This analysis could also be helpful in the case of the reduced solver with supremizer stabilization. Analyzing the stability factor can be used to provide bounds for which numbers N_s of supremizers the solutions are stable, similar to what is done for the Navier-Stokes equations in [8].

The natural expansion of the steady Stokes equations are the Navier-Stokes equations. It would be interesting to do similar analysis for the Navier-Stokes equations to see if there is a natural choice of the reduced basis functions that gives stable solutions without introducing stabilization techniques. Then it could also be interesting to build the high-fidelity solver, that the reduced basis solver is based on, from other stable finite elements than the Taylor-Hood elements. This in order to see if the high-fidelity snapshots affect the stability of the reduced basis solutions. If this is the case, then it is interesting to see if it is possible to choose the number of reduced basis functions in such a way that the solutions are stable without introducing stabilization techniques.

Bibliography

[1] R. Adams and J. Fournier. *Sobolev Spaces*. Elsevier Science, 2. edition, 2003.

[2] R. A. Adams and C. Essex. *Calculus: A Complete Course*. Pearson, 8. edition, 2013.

[3] airfoiltools.com. *NACA 4 digit airfoil generator*, 2018. `http://airfoiltools.com/airfoil/naca4digit`,(accessed June 24, 2018).

[4] B. O. Almroth, P. Stern, and F. A. Brogan. Automatic choice of global shape functions in structural analysis. *AIAA Journal*, 16(5):525 – 528, 1978.

[5] A. M. Archibald, M. Spacek, P. Virtanen, et al. *NumPy*, Oct. 2015. `http://scipy-cookbook.readthedocs.io/items/ParallelProgramming.html`,(accessed July 4, 2018).

[6] D. Auble, D. Bartkiewicz, B. Christiansen, M. Garey, I. Hartung, M. Jette, F. Moll, A. A. Sanchez, and T. Wickberg. *Slurm workload manager*, 2013. `https://slurm.schedmd.com/overview.html`,(accessed June 29, 2018).

[7] J. Baiges, R. Codina, and S. Idelsohn. Explicit reduced-order models for the stabilized finite element approximation of the incompressible Navier-Stokes equations. *International Journal for Numerical Methods in Fluids*, 72(12):1219–1243, Feb. 2013.

[8] F. Ballarin, A. Manzoni, A. Quarteroni, and G. Rozza. Supremizer stabilization of POD-Galerkin approximation of parametrized steady incompressible Navier-Stokes equations. *International Journal for Numerical Methods in Engineering*, 102(5):1136–1161, 2014.

[9] M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera. An 'empirical interpolation' method: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematique*, 339(9):667 – 672, 2004.

[10] M. A. Bazaz, M. un Nabi, and S. Janardhanan. A review of parametric model order reduction techniques. In *2012 IEEE International Conference on Signal Processing, Computing and Control*, pages 1–6, March 2012.

[11] M. Benzi, G. H. Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numerica*, 14:1–137, 2005.

[12] D. Boffi, F. Brezzi, and M. Fortin. *Mixed Finite Element Methods and Applications*. Springer, 2013.

[13] F. Boyer and P. Fabrie. *Steady Stokes equations*, pages 229–343. Springer New York, 2013.

[14] D. Braess. *Finite elements - Theory, fast solvers, and applications in solid mechanics*. Cambridge, 2007.

[15] F. Brezzi and R. S. Falk. Stability of higher-order Hood-Taylor methods. *SIAM Journal on Numerical Analysis*, 28:581–590, June 1991.

[16] F. Brezzi and M. Fortin. *Mixed and Hybrid Finite Element Methods*. Springer, 1991.

[17] A. Buffa and C. de Falco. IsoGeometric Analysis: Stable elements for the 2D Stokes equation. *Journal of Applied Analysis*, pages 1407–1422, March 2011.

[18] A. Buffa, Y. Maday, A. T. Patera, C. Prud'homme, and G. Turinici. A priori convergence of the Greedy algorithm for the parametrized reduced basis method. *ESAIM: Mathematical Modelling and Numerical Analysis*, pages 595 – 603, Jan. 2012.

[19] K. Burrage, P. Burrage, D. Donovan, and B. Thompson. Populations of models, Experimental Designs and coverage of parameter space by Latin Hypercube and Orthogonal Sampling. *Procedia Computer Science*, 51:1762–1771, 2015.

[20] A. Chatterjee. An introduction to the proper orthogonal decomposition. *Current science*, 78(7):808–817, 2000.

[21] L. Chen. Finite element methods for Stokes equations, March 2018. `https://www.math.uci.edu/~chenlong/226/FEMStokes.pdf`.

[22] J. Cottrell, T. Hughes, and Y. Bazilevs. *Isogeometric Analysis Toward Integration of CAD and FEA*. Wiley, 2009.

[23] S. Deparis and A. E. Løvgren. Stabilized Reduced Basis Approximation of Incompressible Three-Dimensional Navier-Stokes Equations in Parametrized Deformed Domains. *Journal of Scientific Computing*, 50(1):198–212, Jan 2012.

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

[25] J. Du, F. Fang, C. Pain, I. Navon, J. Zhu, and D. Ham. POD reduced-order unstructured mesh modeling applied to 2D and 3D fluid flow. *Computers Mathematics with Applications*, 65:362 – 379, 2013.

[26] D. C. Eleni, T. I. Athanasios, and M. P. Dionissios. Evaluation of the turbulence models for the simulation of the flow over a National Advisory Committee for Aeronautics (NACA) 0012 airfoil. *Journal of Mechanical Engineering Research*, 4(3):100–111, March 2012.

[27] J. A. Evans and T. J. Hughes. Isogeometric divergence-conforming B-splines for the unsteady Navier-Stokes equations. *Journal of Computational Physics*, 241:141 – 167, 2013.

[28] J. A. Evans and T. J. R. Hughes. Isogeometric divergence-conforming B-splines for the Darcy–Stokes-Brinkman equations. *Mathematical Models and Methods in Applied Sciences*, 23(04):671–741, 2013.

[29] J. A. Evans and T. J. R. Hughes. Isogeometric divergence-conforming B-splines for the steady Navier-Stokes equations. *Mathematical Models and Methods in Applied Sciences*, 23(08):1421–1478, 2013.

[30] N. Fallah, C. Bailey, M. Cross, and G. Taylor. Comparison of finite element and finite volume methods application in geometrically nonlinear stress analysis. *Applied Mathematical Modelling*, 24(7):439 – 455, 2000.

[31] E. Fonn, M. Tabib, M. S. Siddiqui, A. Rasheed, and T. Kvamsdal. A step towards reduced order modelling of flow characterized by wakes using Proper Orthogonal Decomposition. *Energy Procedia*, 137:452 – 459, 2017.

[32] M. A. Grepl, N. C. Nguyen, K. Veroy, A. T. Patera, and G. R. Liu. *10. Certified Rapid Solution of Partial Differential Equations for Real-Time Parameter Estimation and Optimization*, pages 199–216. Siam, 2007.

[33] W. Härdle and L. Simar. *Applied Multivariate Statistical Analysis*. Springer, 4. edition, 2015.

[34] J. Hesthaven, G. Rozza, and B. Stamm. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. Springer, 2016.

[35] K. A. Johannessen, E. Fonn, A. M. Kvarving, et al. SpliPy: Open source scientific tools for Python, 2013. `https://github.com/sintefmath/Splipy`,(accessed June 22, 2018).

[36] K. Kunisch and S. Volkwein. Galerkin proper orthogonal decomposition methods for parabolic problems. *Numerische Mathematik*, 90(1):117–148, 2001.

[37] A. E. Løvgren, Y. Maday, and E. M. Rønquist. A reduced basis element method for the steady Stokes problem. *ESAIM: Mathematical Modelling and Numerical Analysis*, 40(3):529 – 552, 2006.

[38] T. Lyche and K. Mørken. Spline methods. University Lecture Notes, 2011.

[39] A. Manzoni. An efficient computational framework for reduced basis approximation and a posteriori error estimation of parametrized Navier-Stokes flows. *ESAIM: Mathematical Modelling Numerical Analysis*, 48(4):1199 – 1226, 2014.

[40] A. Manzoni, A. Quarteroni, and G. Rozza. Shape optimization for viscous flows by reduced basis methods and free-form deformation. *International Journal for Numerical Methods in Fluids*, 70(5):646–670, 2011.

[41] M. Mckay, R. Beckman, and W. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1):55–61, 2000.

[42] F. Negri, G. Rozza, A. Manzoni, and A. Quarteroni. Reduced basis method for parametrized elliptic optimal control problems. *SIAM Journal on Scientific Computing*, 35(5):2316–2340, 2013.

[43] A. K. Noor. Recent advances in reduction methods for nonlinear problems. *Computers Structures*, 13(1):31 – 44, 1981.

[44] A. K. Noor. On making large nonlinear problems small. *Computer Methods in Applied Mechanics and Engineering*, 34(1):955 – 985, 1982.

[45] A. K. Noor and J. M. Peters. Reduced basis technique for nonlinear analysis of structures. *AIAA Journal*, 18(4):455 – 462, 1980.

[46] K. Nordanger, R. Holdahl, A. M. Kvarving, A. Rasheed, and T. Kvamsdal. Implementation and comparison of three isogeometric Navier-Stokes solvers applied to simulation of flow past a fixed 2D NACA0012 airfoil at high Reynolds number. *Computer Methods in Applied Mechanics and Engineering*, 284:664 – 688, Feb. 2015.

[47] W. Oźański. The Lagrange multiplier and the stationary Stokes equations. *International Journal for Numerical Methods in Fluids*, March 2017.

[48] A. Quarteroni. *Numerical Models for Differential Problems*. Springer, 4. edition, 2009.

[49] A. Quarteroni, A. Manzoni, and F. Negri. *Reduced Basis Methods for Partial Differential Equations An Introduction*. Springer, 2016.

[50] A. Quarteroni and A. Valli. *Numerical Approximation of Partial Differential Equations*. Springer, 1994.

[51] G. Rozza. *shape design by optimal flow control and reduced basis techniques : applications to bypass configurations in haemodynamics*. PhD thesis, EPFL, 2005.

[52] G. Rozza. Reduced basis methods for Stokes equations in domains with non-affine parameter dependence. *Computing and Visualization in Science*, 12(1):23–35, Jan 2009.

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

[54] G. Rozza, D. B. P. Huynh, and A. T. Patera. Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations. *Archives of Computational Methods in Engineering*, 15(3):1–47, Sep. 2007.

[55] W. Rudin. *Principles of mathematical analysis*. McGraw-Hill, 3. edition, 1976.

[56] T. J. Santner, B. J. Williams, and W. I. Notz. *The Design and Analysis of Computer Experiments*. Springer, New York, NY, 2003.

[57] W. Schilders, H. van der Vorst, and J. Rommes. *Model Order Reduction: Theory, Research Aspects and Applications*. Springer, 2008.

[58] G. Strang. *Introduction to Linear Algebra*. Wellesley - Cambridge Press, 4. edition, 2009.

[59] F. Tröltzsch. *Optimal Control of Partial Differential Equations: Theory, Methods and Applications*. The American Mathematical Society, 2010.

[60] S. Ullmann, M. Rotkvic, and J. Lang. POD-Galerkin reduced-order modeling with adaptive finite element snapshots. *Journal of Computational Physics*, 325:244–258, 2016.

[61] G. van Zwieten et al. *Nutils*, 2014. `http://docs.nutils.org/en/latest/`,(accessed July 4, 2018).

[62] L. Venturi, F. Ballarin, and G. Rozza. A Weighted POD Method for Elliptic PDEs with Random Inputs. *`https://arxiv.org/abs/1802.08724`*, Feb. 2018.

[63] F. M. White. *Fluid Mechanics*. McGraw-Hill, 6. edition, 2008.

Appendix

5.1 Full derivation of the Steady Stokes linear system on a parameter-dependent domain

Following the steps of Section 4.1 we pull back the original problem (4.14) from the parameter-dependent domain $\tilde{\Omega}(\boldsymbol{\mu})$ onto the parameter-independent domain Ω and provide an approximate solution $(\mathbf{u_h}(\boldsymbol{\mu}), p_h(\boldsymbol{\mu}))$ through the linear system (4.9). The following components are needed.

The Jacobian matrix of the map $\boldsymbol{\mathcal{F}}(\mathbf{x};\boldsymbol{\mu})$ from equation (4.15) is

$$\mathbb{J}_{\mathcal{F}}(\boldsymbol{\mu}) = \mathbb{J}_{\mathcal{F}}(\mathbf{x};\boldsymbol{\mu}) = \begin{bmatrix} \mu_1 \cos(\mu_3) & -\mu_2 \sin(\mu_3) \\ \mu_1 \sin(\mu_3) & \mu_2 \cos(\mu_3) \end{bmatrix}.$$

We can also compute the determinant of the Jacobian matrix

$$|J_{\mathcal{F}}(\boldsymbol{\mu})| = \mu_1\mu_2 \cos^2(\mu_3) + \mu_1\mu_2 \sin^2(\mu_3) = \mu_1\mu_2,$$

and the inverse transpose

$$\begin{aligned} (J_{\mathcal{F}}(\boldsymbol{\mu}))^{-1} &= \frac{1}{\mu_1\mu_2} \begin{bmatrix} \mu_2 \cos(\mu_3) & \mu_2 \sin(\mu_3) \\ -\mu_1 \sin(\mu_3) & \mu_1 \cos(\mu_3) \end{bmatrix}, \\ (J_{\mathcal{F}}(\boldsymbol{\mu}))^{-\top} &= \frac{1}{\mu_1\mu_2} \begin{bmatrix} \mu_2 \cos(\mu_3) & -\mu_1 \sin(\mu_3) \\ \mu_2 \sin(\mu_3) & \mu_1 \cos(\mu_3) \end{bmatrix}. \end{aligned}$$

we introduce the notation $\boldsymbol{\phi_i} = \begin{bmatrix} (\boldsymbol{\phi_i})_0 \\ (\boldsymbol{\phi_i})_1 \end{bmatrix}$ and $\nabla\boldsymbol{\phi_i} = \begin{bmatrix} (\boldsymbol{\phi_i})_{0,x_0} & (\boldsymbol{\phi_i})_{1,x_0} \\ (\boldsymbol{\phi_i})_{0,x_1} & (\boldsymbol{\phi_i})_{1,x_1} \end{bmatrix}$ where $(\boldsymbol{\phi_i})_{p,x_k}$ is the derivative of the p-th element of the vector $\boldsymbol{\phi_i}$ with respect to x_k.

First we compute

$$(J_{\mathcal{F}}(\boldsymbol{\mu}))^{-\top}\, \nabla\boldsymbol{\phi_i} = \left(\frac{1}{\mu_1\mu_2} \begin{bmatrix} \mu_2 \cos(\mu_3) & -\mu_1 \sin(\mu_3) \\ \mu_2 \sin(\mu_3) & \mu_1 \cos(\mu_3) \end{bmatrix} \begin{bmatrix} (\boldsymbol{\phi_j})_{0,x_0} & (\boldsymbol{\phi_j})_{1,x_0} \\ (\boldsymbol{\phi_j})_{0,x_1} & (\boldsymbol{\phi_j})_{1,x_1} \end{bmatrix} \right) =$$

$$\frac{1}{\mu_1\mu_2} \begin{bmatrix} \mu_2 \cos(\mu_3)(\boldsymbol{\phi_i})_{0,x_0} - \mu_1 \sin(\mu_3)(\boldsymbol{\phi_i})_{0,x_1} & \mu_2 \cos(\mu_3)(\boldsymbol{\phi_i})_{1,x_0} - \mu_1 \sin(\mu_3)(\boldsymbol{\phi_i})_{1,x_1} \\ \mu_2 \sin(\mu_3)(\boldsymbol{\phi_i})_{0,x_0} + \mu_1 \cos(\mu_3)(\boldsymbol{\phi_i})_{0,x_1} & \mu_2 \sin(\mu_3)(\boldsymbol{\phi_i})_{1,x_0} + \mu_1 \cos(\mu_3)(\boldsymbol{\phi_i})_{1,x_1} \end{bmatrix}$$

Now it is possible to compute all the bilinear and linear forms in equations (4.5) - (4.8).

$$
\begin{aligned}
&(A(\boldsymbol{\mu}))_{ij} = a(\boldsymbol{\phi_j}, \boldsymbol{\phi_i}; \boldsymbol{\mu}) = \\
&\frac{\mu_4}{\mu_1\mu_2}[\int_\Omega \left(\mu_2\cos(\mu_3)(\boldsymbol{\phi_j})_{0,x_0} - \mu_1\sin(\mu_3)(\boldsymbol{\phi_j})_{0,x_1}\right)\left(\mu_2\cos(\mu_3)(\boldsymbol{\phi_i})_{0,x_0} - \mu_1\sin(\mu_3)(\boldsymbol{\phi_i})_{0,x_1}\right)\,\mathrm{d}\Omega \\
&+\int_\Omega \left(\mu_2\cos(\mu_3)(\boldsymbol{\phi_j})_{1,x_0} - \mu_1\sin(\mu_3)(\boldsymbol{\phi_j})_{1,x_1}\right)\left(\mu_2\cos(\mu_3)(\boldsymbol{\phi_i})_{1,x_0} - \mu_1\sin(\mu_3)(\boldsymbol{\phi_i})_{1,x_1}\right)\,\mathrm{d}\Omega \\
&+\int_\Omega \left(\mu_2\sin(\mu_3)(\boldsymbol{\phi_j})_{0,x_0} + \mu_1\cos(\mu_3)(\boldsymbol{\phi_j})_{0,x_1}\right)\left(\mu_2\sin(\mu_3)(\boldsymbol{\phi_i})_{0,x_0} + \mu_1\cos(\mu_3)(\boldsymbol{\phi_i})_{0,x_1}\right)\,\mathrm{d}\Omega \\
&+\int_\Omega \left(\mu_2\sin(\mu_3)(\boldsymbol{\phi_j})_{1,x_0} + \mu_1\cos(\mu_3)(\boldsymbol{\phi_j})_{1,x_1}\right)\left(\mu_2\sin(\mu_3)(\boldsymbol{\phi_i})_{1,x_0} + \mu_1\cos(\mu_3)(\boldsymbol{\phi_i})_{1,x_1}\right)\,\mathrm{d}\Omega] \\[2ex]
&=\frac{\mu_4}{\mu_1\mu_2}[\int_\Omega \mu_2^2\cos^2(\mu_3)\left((\boldsymbol{\phi_j})_{0,x_0}(\boldsymbol{\phi_i})_{0,x_0} + (\boldsymbol{\phi_j})_{1,x_0}(\boldsymbol{\phi_i})_{1,x_0}\right) \\
&\qquad\qquad\qquad + \mu_2^2\sin^2(\mu_3)\left((\boldsymbol{\phi_j})_{0,x_0}(\boldsymbol{\phi_i})_{0,x_0} + (\boldsymbol{\phi_j})_{1,x_0}(\boldsymbol{\phi_i})_{1,x_0}\right)\,\mathrm{d}\Omega \\
&\quad+\int_\Omega \mu_1^2\cos^2(\mu_3)\left((\boldsymbol{\phi_j})_{0,x_1}(\boldsymbol{\phi_i})_{0,x_1} + (\boldsymbol{\phi_j})_{1,x_1}(\boldsymbol{\phi_i})_{1,x_1}\right) \\
&\qquad\qquad\qquad + \mu_1^2\sin^2(\mu_3)\left((\boldsymbol{\phi_j})_{0,x_1}(\boldsymbol{\phi_i})_{0,x_1} + (\boldsymbol{\phi_j})_{1,x_1}(\boldsymbol{\phi_i})_{1,x_1}\right)\,\mathrm{d}\Omega \\
&\quad+\int_\Omega \mu_1\mu_2\cos(\mu_3)\sin(\mu_3)\left((\boldsymbol{\phi_j})_{0,x_0}(\boldsymbol{\phi_i})_{0,x_1} + (\boldsymbol{\phi_j})_{0,x_1}(\boldsymbol{\phi_i})_{0,x_0} + (\boldsymbol{\phi_j})_{1,x_0}(\boldsymbol{\phi_i})_{1,x_1} + (\boldsymbol{\phi_j})_{1,x_1}(\boldsymbol{\phi_i})_{1,x_0}\right)\,\mathrm{d}\Omega \\
&\quad-\int_\Omega \mu_1\mu_2\cos(\mu_3)\sin(\mu_3)\left((\boldsymbol{\phi_j})_{0,x_0}(\boldsymbol{\phi_i})_{0,x_1} + (\boldsymbol{\phi_j})_{0,x_1}(\boldsymbol{\phi_i})_{0,x_0} + (\boldsymbol{\phi_j})_{1,x_0}(\boldsymbol{\phi_i})_{1,x_1} + (\boldsymbol{\phi_j})_{1,x_1}(\boldsymbol{\phi_i})_{1,x_0}\right)\,\mathrm{d}\Omega] \\[2ex]
&=\frac{\mu_4}{\mu_1\mu_2}[\int_\Omega \mu_2^2\left((\boldsymbol{\phi_j})_{0,x_0}(\boldsymbol{\phi_i})_{0,x_0} + (\boldsymbol{\phi_j})_{1,x_0}(\boldsymbol{\phi_i})_{1,x_0}\right)\,\mathrm{d}\Omega \\
&\qquad\qquad\qquad + \int_\Omega \mu_1^2\left((\boldsymbol{\phi_j})_{0,x_1}(\boldsymbol{\phi_i})_{0,x_1} + (\boldsymbol{\phi_j})_{1,x_1}(\boldsymbol{\phi_i})_{1,x_1}\right)\,\mathrm{d}\Omega] = \\
&\frac{\mu_4\mu_2}{\mu_1}a_1(\boldsymbol{\phi_j}, \boldsymbol{\phi_i}) + \frac{\mu_4\mu_1}{\mu_2}a_2(\boldsymbol{\phi_j}, \boldsymbol{\phi_i}) = \frac{\mu_4\mu_2}{\mu_1}(A_1)_{ij} + \frac{\mu_4\mu_1}{\mu_2}(A_2)_{ij} \qquad 1 \le i, j \le N_h.
\end{aligned} \tag{5.1}
$$

$$
\begin{aligned}
&(B(\boldsymbol{\mu}))_{ij} = b(\boldsymbol{\phi_j}, \psi_i; \boldsymbol{\mu}) = -\int_\Omega \psi_i \left(\frac{1}{\mu_1\mu_2}\begin{bmatrix}\mu_2\cos(\mu_3) & -\mu_1\sin(\mu_3) \\ \mu_2\sin(\mu_3) & \mu_1\cos(\mu_3)\end{bmatrix}\nabla\right)\cdot\boldsymbol{\phi_j}\ \mu_1\mu_2\,\mathrm{d}\Omega = \\
&-\int_\Omega \psi_i\left[\mu_2\cos(\mu_3)\tfrac{\partial}{\partial x_0} - \mu_1\sin(\mu_3)\tfrac{\partial}{\partial x_1} \quad \mu_2\sin(\mu_3)\tfrac{\partial}{\partial x_0} + \mu_1\cos(\mu_3)\tfrac{\partial}{\partial x_1}\right]\boldsymbol{\phi_j}\,\mathrm{d}\Omega \\
&= -\mu_2\cos(\mu_3)\int_\Omega \psi_i(\boldsymbol{\phi_j})_{0,x_0}\,\mathrm{d}\Omega + \mu_1\sin(\mu_3)\int_\Omega \psi_{\mathrm{i}}(\boldsymbol{\phi_j})_{0,\mathrm{x}_1}\,\mathrm{d}\Omega \\
&\qquad\qquad\qquad - \mu_2\sin(\mu_3)\int_\Omega \psi_i(\boldsymbol{\phi_j})_{1,x_0}\,\mathrm{d}\Omega - \mu_1\cos(\mu_3)\int_\Omega \psi_{\mathrm{i}}(\boldsymbol{\phi_j})_{1,\mathrm{x}_1}\,\mathrm{d}\Omega \\
&= \mu_2\cos(\mu_3)b_1(\boldsymbol{\phi_j}, \psi_i) - \mu_1\sin(\mu_3)b_2(\boldsymbol{\phi_j}, \psi_i) + \mu_2\sin(\mu_3)b_3(\boldsymbol{\phi_j}, \psi_i) + \mu_1\cos(\mu_3)b_4(\boldsymbol{\phi_j}, \psi_i) \\
&= \mu_2\cos(\mu_3)(B_1)_{ij} - \mu_1\sin(\mu_3)(B_2)_{ij} \\
&\qquad\qquad\qquad + \mu_2\sin(\mu_3)(B_3)_{ij} + \mu_1\cos(\mu_3)(B_4)_{ij} \quad 1 \le i \le M_h,\ 1 \le j \le N_h.
\end{aligned} \tag{5.2}
$$

For the elements of the vector $\mathbf{f_1}(\boldsymbol{\mu})$ and $\mathbf{f_2}(\boldsymbol{\mu})$ we express the approximation of the lifting function as $\mathbf{u_g} = \sum_{i=1}^{N_h} w_g^{(i)}\boldsymbol{\phi_i}$, where this approximation takes the value 1 at the boundary of the domain. Then $\mu_5\mathbf{u_g}$ will obtain the right value at the inflow Dirichlet boundary, that is $\mu_5\mathbf{u_g}|_{\Gamma_{D_1}} = \mu_5$. This gives us the following

$$
\begin{aligned}
(\mathbf{f_1}(\boldsymbol{\mu}))_i &= f_1(\boldsymbol{\phi_i};\boldsymbol{\mu}) = -\sum_{j=1}^{N_h} a(\boldsymbol{\phi_j},\boldsymbol{\phi_i};\boldsymbol{\mu})\mu_5 w_g^{(j)} = \\
-\frac{\mu_4\mu_5\mu_2}{\mu_1}\sum_{j=1}^{N_h}&\left(a_1(\boldsymbol{\phi_j},\boldsymbol{\phi_i})w_g^{(j)}\right) - \frac{\mu_4\mu_5\mu_1}{\mu_2}\sum_{j=1}^{N_h}\left(a_2(\boldsymbol{\phi_j},\boldsymbol{\phi_i})w_g^{(j)}\right) = \\
-\frac{\mu_4\mu_5\mu_2}{\mu_1}\sum_{j=1}^{N_h}&\left((A_1)_{ij}w_g^{(j)}\right) - \frac{\mu_4\mu_5\mu_1}{\mu_2}\sum_{j=1}^{N_h}\left((A_2)_{ij}w_g^{(j)}\right) \qquad 1 \le i \le N_h,
\end{aligned}
\tag{5.3}
$$

where the lifting function coefficients are stored in the vector $\mathbf{w_g} = [w_g^{(1)}, w_g^{(2)}, \dots, w_g^{(N_h)}]$.

$$
\begin{aligned}
(\mathbf{f_2}(\boldsymbol{\mu}))_i &= f_2(\psi_i;\boldsymbol{\mu}) = -\sum_{j}^{N_h}\left(b(\boldsymbol{\phi_j},\psi_i;\boldsymbol{\mu})\mu_5 w_g^{(j)}\right) = \\
&-\mu_2\mu_5\cos(\mu_3)\sum_{j}^{N_h}\left(b_1(\boldsymbol{\phi_j},\psi_i)w_g^{(j)}\right) + \mu_1\mu_5\sin(\mu_3)\sum_{j}^{N_h}\left(b_2(\boldsymbol{\phi_j},\psi_i)w_g^{(j)}\right) \\
&- \mu_2\mu_5\sin(\mu_3)\sum_{j}^{N_h}\left(b_3(\boldsymbol{\phi_j},\psi_i)w_g^{(j)}\right) - \mu_1\mu_5\cos(\mu_3)\sum_{j}^{N_h}\left(b_4(\boldsymbol{\phi_j},\psi_i)w_g^{(j)}\right) = \\
&-\mu_2\mu_5\cos(\mu_3)\sum_{j}^{N_h}\left((B_1)_{ij}w_g^{(j)}\right) + \mu_1\mu_5\sin(\mu_3)\sum_{j}^{N_h}\left((B_2)_{ij}w_g^{(j)}\right) \\
&- \mu_2\mu_5\sin(\mu_3)\sum_{j}^{N_h}\left((B_3)_{ij}w_g^{(j)}\right) - \mu_1\mu_5\cos(\mu_3)\sum_{j}^{N_h}\left((B_4)_{ij}w_g^{(j)}\right) \qquad 1 \le i \le M_h,
\end{aligned}
\tag{5.4}
$$

Equations (5.1) - (5.4) can be written as the linear system

$$
\begin{bmatrix} A_h(\boldsymbol{\mu}) & B_h^{\top}(\boldsymbol{\mu}) \\ B_h(\boldsymbol{\mu}) & 0 \end{bmatrix}
\begin{bmatrix} \mathbf{w_h}(\boldsymbol{\mu}) \\ \mathbf{p_h}(\boldsymbol{\mu}) \end{bmatrix}
=
\begin{bmatrix} \mathbf{f_{1h}}(\boldsymbol{\mu}) \\ \mathbf{f_{2h}}(\boldsymbol{\mu}) \end{bmatrix}.
\tag{5.5}
$$

where the matrices and vectors are the affine combinations

$$
\begin{gathered}
A_h(\boldsymbol{\mu}) = \frac{\mu_4\mu_2}{\mu_1}A_1 + \frac{\mu_4\mu_1}{\mu_2}A_2 \\
B_h(\boldsymbol{\mu}) = \mu_2\cos(\mu_3)B_1 - \mu_1\sin(\mu_3)B_2 + \mu_2\sin(\mu_3)B_3 + \mu_1\cos(\mu_3)B_4 \\
\mathbf{f_{1h}}(\boldsymbol{\mu}) = -\frac{\mu_4\mu_5\mu_2}{\mu_1}A_1\mathbf{w_g} - \frac{\mu_4\mu_5\mu_1}{\mu_2}A_2\mathbf{w_g} \\
\mathbf{f_{2h}}(\boldsymbol{\mu}) = -\mu_2\mu_5\cos(\mu_3)B_1\mathbf{w_g} + \mu_1\mu_5\sin(\mu_3)B_2\mathbf{w_g} - \mu_2\mu_5\sin(\mu_3)B_3\mathbf{w_g} - \mu_1\mu_5\cos(\mu_3)B_4\mathbf{w_g},
\end{gathered}
\tag{5.6}
$$

where the elements of the matrices A_1, A_2, B_1 and B_2 are given in equations (5.1) and (5.2). The linear system (5.5) along with its components are the same as the linear system in equation (4.18)