Institut für
Angewandte Mathematik

Leibniz
Universität
Hannover

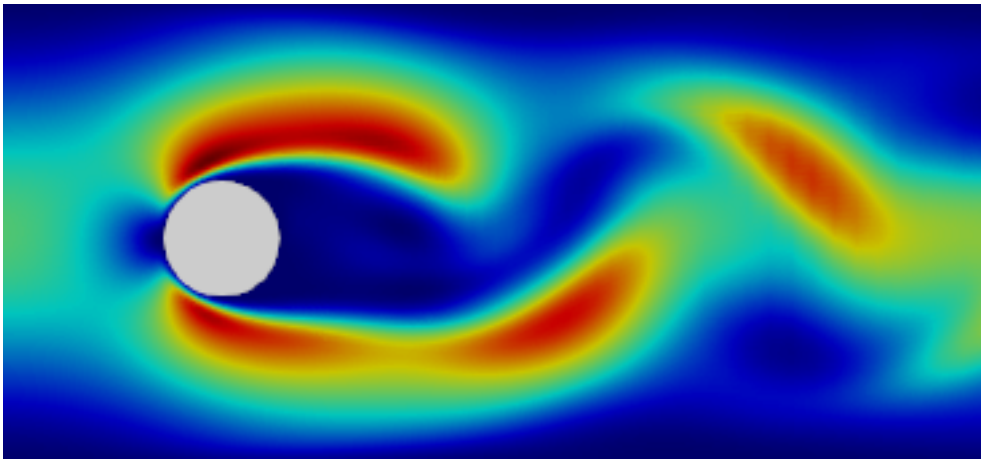
August 19, 2021

Leibniz Universität Hannover

Institut für Angewandte Mathematik

Master thesis

Proper Orthogonal Decomposition for Fluid Mechanics Problems



Author:
Julian Roth (10012482)

Advisor:
Prof. Dr. Thomas Wick
Coadvisor:
Prof. Dr. Sven Beuchler

Abstract

The goal of this thesis is the reduced order modelling (ROM) of fluid mechanics problems by using the proper orthogonal decomposition (POD). The main motivation of this work is the need for real-time simulations for many engineering problems, where the underlying partial differential equation might even be solved multiple times with different parameters.

In this thesis, the fundamentals of the proper orthogonal decomposition are explored. Thereafter, a reduced order model based on the proper orthogonal decomposition (POD-ROM) is presented for parabolic problems and demonstrated experimentally for the heat equation. After the finite element simulation of the Navier-Stokes equations, we focus on the main goal, which is the reduced order modelling of these equations. The velocity and pressure are computed by separate reduced order models. First, we solve the momentum equation, which yields the reduced velocity. Next, three different methods for the pressure reconstruction are discussed and we recover the pressure with a neural network based approach.

Zusammenfassung

Das Ziel dieser Arbeit ist die Modellordnungsreduktion von Problemen aus der Strömungsmechanik mithilfe der Proper Orthogonal Decomposition (POD). Die Hauptmotivation dieser Abschlussarbeit ist der Bedarf nach Echtzeit-Simulationen in vielen Praxisproblemen, in denen die zugrundeliegende partielle Differentialgleichung manchmal wiederholt mit unterschiedlichen Parametern gelöst weden muss.

Diese Arbeit erforscht die Grundlagen der Proper Orthogonal Decomposition. Daraufhin wird ein Proper Orthogonal Decompostion basiertes Modellordnungsreduktionsverfahren für parabolische Probleme vorgestellt und am Beispiel der Wärmeleitungsgleichung in numerischen Experimenten demonstriert. Nach der Finite Elemente Simulation der Navier-Stokes-Gleichungen wird der Fokus auf das Hauptziel dieser Arbeit gelegt, das in der ordnungsreduzierenden Modellbildung dieser Gleichungen besteht. Die Geschwindigkeit und der Druck werden durch gesonderte ordnungsreduzierende Modelle berechnet. Zuerst lösen wir die Impulserhaltungsgleichung und erhalten daraus die Geschwindigkeit. Dann werden drei verschiedene Methoden zur Druckrekonstruktion erörtert und der Druck wird mit einem auf neuronalen Netzwerken basierenden Verfahren ermittelt.

Table of Contents

Chapter 1

Introduction

Fluid mechanics is the study of fluid flow. This subject of research has applications to many areas of nature and technology that surround us. In nature, fluid mechanics models the flow of water in rivers, lakes and oceans. Furthermore, it enables us to create simulations of the earth's atmosphere and make forecasts of the weather on the next day. The study of fluids is also crucial for the industry, especially when combining the Navier-Stokes equations, which describe fluid flow, with differential equations that describe other physical phenomena, like the heat equation or Maxwell's equations. These coupled problems are called multiphysics problems and for example describe the combustion in cars' engines by combining fluid flow with chemical reactions. Fluid mechanics is of exceptional significance for the automobile and aerospace industries, where with the help of the Navier-Stokes equations quantities of interest like the drag of an airplane or car can be modeled. Traditionally, these quantities of interest have been determined by real world experiments, e.g. by testing different airplane wing shapes in a wind tunnel. However, these experiments are slow and costly. Hence, with the advent of the digital age and the rise of computational power, more and more experiments have been replaced by computer simulations based on digital twins. Scientific computing methods can lead to a large reduction in cost and a faster research and development cycle. One of the most popular methods for the simulation of fluid flow is the Finite Element Method (FEM). To increase the accuracy of finite element solutions, one needs to increase the number of unknowns in the linear equation systems. However, this is very computationally demanding and may require many days of compute on high performance computing clusters. Moreover, in optimization and control the underlying partial differential equation needs to be solved multiple times, which poses a challenge to many classical numerical methods.

Therefore, reduced order models (ROM) based on the proper orthogonal decomposition (POD) are being used in many real world applications. These reduced order models solve the problem of the high computational cost of high fidelity simulations by reducing the number of unknowns from usually a few million to the order of 10 unknowns. Drastically reducing the number of unknowns leads to a significant decrease in compute time and storing capacity, which in turn enables real-time simulations [45]. This data-driven approach computes a new basis, which contains important features of the fluid flow (or more generally of the solution of the partial differential equation), with the help of the proper orthogonal decomposition. Therein, the proper orthogonal decomposition calculates the new basis from a set of snapshots from the computationally expensive high fidelity direct numerical simulation [6, 11, 12, 15, 37, 69, 65, 42], e.g. from finite element simulations, or from experimental data [5, 33]. Reduced order models have many areas of applications, including optimization of chemical processes, mechanical engineering, acoustics and vibrations, microelectronics and electromagnetism, as well as computational aerodynamics to only name a few [9]. Moreover, the proper orthogonal decomposition has been used to compute reduced-order controllers [4] and to solve inverse problems [25].

In this work, we will analyze a proper orthogonal decomposition based reduced order model for the time-dependent Navier-Stokes equations. In Chapter 2, we will recapitulate the mathematical tools needed for the analysis of partial differential equations and the theoretical background of the proper orthogonal decomposition. Next, we introduce the main idea behind the proper orthogonal decomposition and how it can be used used in $\mathbb{R}^m$ in Chapter 3. In Chapter 4, we investigate the usage of the proper orthogonal decomposition for reduced order modelling at the example of the heat equation, which is a prototypical example of a parabolic problem. The finite element simulation of the Navier-Stokes equations are being discussed in Chapter 5 and we present first numerical experiments of a two dimensional laminar flow around a cylinder. In Chapter 6, we explain how the proper orthogonal decomposition can be used for reduced order modelling of the Navier-Stokes equations. Herein, the velocity and pressure are being modeled in a partitioned manner and different methods for the pressure reconstruction are being presented, including a neural network based approach which is novel to the best of the author's knowledge. Thereafter, the results of the reduced order model are being compared to the finite element solution. Finally, we will draw the conclusions of this thesis in Chapter 7 and give several ideas for future research. In Appendix A, we prove Theorem 3.3.1, which states that the POD basis consists of the left singular vectors from the singular value decomposition.

Chapter 2

Notation and mathematical tools

2.1 Domains

Let us consider open domains $\Omega \subset \mathbb{R}^d$ with Lipschitz boundary $\partial\Omega$, where $d \in \{2, 3\}$ is the dimension. By $\boldsymbol{n}$ we denote the outer unit normal vector with respect to $\partial\Omega$.

2.2 Derivatives

For partial derivatives, we use the notation

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{x}_k}(\cdot) &= \partial_k(\cdot) \quad \text{with } 1 \leq k \leq d, \\ \frac{\partial}{\partial t}(\cdot) &= \partial_t(\cdot), \end{aligned}$$

and

$$\frac{\partial^2}{\partial \boldsymbol{x}_k \partial \boldsymbol{x}_l}(\cdot) = \partial_{kl}(\cdot) \quad \text{with } 1 \leq k, l \leq d.$$

The **gradient** of a scalar-valued function $v : \mathbb{R}^d \to \mathbb{R}$ is

$$\nabla v = \begin{pmatrix} \partial_1 v \\ \vdots \\ \partial_d v \end{pmatrix},$$

and for a vector-valued function $\boldsymbol{v} : \mathbb{R}^d \to \mathbb{R}^d$ it reads

$$\nabla \boldsymbol{v} = \begin{pmatrix} \partial_1 v_1 & \cdots & \partial_d v_1 \\ \vdots & \ddots & \vdots \\ \partial_1 v_d & \cdots & \partial_d v_d \end{pmatrix}.$$

The **divergence** of a vector-valued function $\boldsymbol{v} : \mathbb{R}^d \to \mathbb{R}^d$ is defined as

$$\nabla \cdot \boldsymbol{v} = \sum_{k=1}^{d} \partial_k v_k$$

and for a tensor-valued function $\boldsymbol{\sigma} : \mathbb{R}^d \to \mathbb{R}^{d \times d}$ it is given by

$$\nabla \cdot \boldsymbol{\sigma} = \begin{pmatrix} \sum_{k=1}^{d} \partial_k \sigma_{1k} \\ \vdots \\ \sum_{k=1}^{d} \partial_k \sigma_{dk} \end{pmatrix}.$$

The **Laplace operator** of a scalar valued function $u : \mathbb{R}^d \to \mathbb{R}$ reads

$$\Delta u = \nabla \cdot \nabla u = \sum_{k=1}^{d} \partial_{kk} u.$$

The convection term from the Navier-Stokes equations is defined as

$$(\boldsymbol{v} \cdot \nabla)\boldsymbol{v} = \begin{pmatrix} \boldsymbol{v} \cdot \nabla v_1 \\ \vdots \\ \boldsymbol{v} \cdot \nabla v_d \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{d} (\partial_k v_1) v_k \\ \vdots \\ \sum_{k=1}^{d} (\partial_k v_d) v_k \end{pmatrix}$$

for a vector-valued velocity $\boldsymbol{v} : \mathbb{R}^d \to \mathbb{R}^d$.

2.3 Bounded operators and spectral theory

Let V and W denote two real normed spaces. Then a **bounded linear operator** $T : V \to W$ satisfies the properties

- $T(\alpha v_1 + \beta v_2) = \alpha T(v_1) + \beta T(v_2) \qquad \forall \alpha, \beta \in \mathbb{R},\ v_1, v_2 \in V,$ (linear)
- $\exists c > 0 :\ \|Tv\|_W \leq c\|v\|_V \qquad \forall v \in V.$ (bounded)

The space of all bounded linear operators from V to W is written as $L(V, W)$ and the **dual space** of V is defined as $V^* = L(V, \mathbb{R})$. Further, we employ the shorthand notation $L(V) = L(V, V)$. The **adjoint operator** $T^* : W^* \to V^*$ of a linear operator $T \in L(V, W)$ is defined by the relation

$$\langle T^* f, v \rangle_{V^* \times V} = \langle f, Tv \rangle_{W^* \times W} \qquad \forall\ (f, v) \in W^* \times V.$$

Here, $\langle \cdot, \cdot \rangle_{V^* \times V}$ represents the duality pairing of V^* and V, which is given by

$$\langle g, v \rangle_{V^* \times V} = g(v) \qquad \forall\ (g, v) \in V^* \times V.$$

Let V be a real Hilbert space and $T \in L(V)$ a linear operator. T is a **self-adjoint** operator, if $T = T^*$. T is a **non-negative** operator, if

$$(Tv, v)_V \geq 0 \qquad \forall\ v \in V.$$

Let V and W denote two real Banach spaces. Then, $T \in L(V, W)$ is a **compact** operator, if for every sequence $\{v_n\}_{n \in \mathbb{N}} \subset V$ the sequence $\{Tv_n\}_{n \in \mathbb{N}} \subset W$ has a convergent subsequence.

Let V denote a Banach space. The **spectrum** of $T \in L(V)$ is given by

$$\sigma(T) = \{\lambda \in \mathbb{C} \mid (\lambda I - T)^{-1} \notin L(V)\},$$

where $I : V \to V, v \mapsto v$ is the identity operator. Let $v \in V \setminus \{0\}$ and $\lambda \in \mathbb{C}$ such that $Tv = \lambda v$. Then, v is an **eigenvector** of T with corresponding **eigenvalue** λ. The set of all eigenvalues is the point spectrum of T.

Theorem 2.3.1 (Hilbert-Schmidt)**.** *Let V be a Hilbert space and $T \in L(V)$ a compact, self-adjoint operator. Then, there exists a complete orthonormal basis $\{\psi_i\}_{i \in \mathbb{N}} \subset V$ with eigenvalues $\{\lambda_i\}_{i \in \mathbb{N}}$ such that*

$$T\psi_i = \lambda_i \psi_i \qquad \forall\ i \in \mathbb{N}$$

and

$$\lambda_i \xrightarrow[i \to \infty]{} 0.$$

2.4 Function spaces

To introduce the Lebesgue spaces $L^p(\Omega)$ with $1 \leq p \leq \infty$, we first define their norms for a function $u : \Omega \to \mathbb{R}$, which are given by

$$\|u\|_{L^p(\Omega)} = \left(\int_\Omega |u(\boldsymbol{x})|^p \, \mathrm{d}\boldsymbol{x} \right)^{\frac{1}{p}}, \qquad 1 \leq p < \infty,$$
$$\|u\|_{L^\infty(\Omega)} = \operatorname*{ess\,sup}_{\boldsymbol{x} \in \Omega} |u(\boldsymbol{x})|.$$

For $1 \leq p \leq \infty$, the **Lebesgue space** is defined as

$$L^p(\Omega) = \{f : \Omega \to \mathbb{R} \mid f \text{ measurable }, \|f\|_{L^p(\Omega)} < \infty\}.$$

These spaces are Banach spaces and for $p = 2$ we get the Hilbert space $L^2(\Omega)$ with inner product

$$(u, v)_{L^2(\Omega)} = \int_\Omega u(\boldsymbol{x}) v(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}.$$

For vector-valued functions $\boldsymbol{u}, \boldsymbol{v} : \mathbb{R}^d \to \mathbb{R}^d$ the L^2-inner product is defined as

$$(\boldsymbol{u}, \boldsymbol{v})_{L^2(\Omega)} = \int_\Omega \boldsymbol{u}(\boldsymbol{x}) \cdot \boldsymbol{v}(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x},$$

and for tensor-valued functions $A, B : \mathbb{R}^d \to \mathbb{R}^{d \times d}$ it reads

$$(A, B)_{L^2(\Omega)} = \int_\Omega A(\boldsymbol{x}) : B(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}.$$

Here, : represents the Frobenius inner product, which is given by $A(\boldsymbol{x}) : B(\boldsymbol{x}) = \sum_{i=1}^d \sum_{j=1}^d A_{ij}(\boldsymbol{x}) B_{ij}(\boldsymbol{x})$. For convenience, we will use the shorthand notation

$$(\cdot, \cdot) := (\cdot, \cdot)_{L^2(\Omega)}$$

for the L^2-inner product.

Let $C_c^\infty(\Omega)$ denote the space of infinitely differentiable functions with compact support. We call $\boldsymbol{w} = \nabla u \in [L^2(\Omega)]^d$ the **variational gradient** of $u \in L^2(\Omega)$, if

$$\int_\Omega \boldsymbol{w} \cdot \boldsymbol{\phi} \, \mathrm{d}\boldsymbol{x} = -\int_\Omega u(\nabla \cdot \boldsymbol{\phi}) \, \mathrm{d}\boldsymbol{x} \quad \forall \boldsymbol{\phi} \in [C_c^\infty(\Omega)]^d .$$

The **Sobolev spaces** play an important role in the analysis of partial differential equations. In this thesis, we only require Sobolev spaces, which contain functions that are one time weakly differentiable, i.e.

$$H^1(\Omega) = \{f \in L^2(\Omega) \mid \nabla f \in [L^2(\Omega)]^d\}.$$

When working with homogeneous Dirichlet boundary conditions, we will consider the space

$$H_0^1(\Omega) = \{f \in H^1(\Omega) \mid f = 0 \text{ on } \partial\Omega\}.$$

Moreover, $H^1(\Omega)$ is a Hilbert space with the inner product

$$(u, v)_{H^1(\Omega)} = (u, v)_{L^2(\Omega)} + (\nabla u, \nabla v)_{L^2(\Omega)}.$$

To introduce the Bochner spaces $L^p((0, T), \Omega)$ with $1 \leq p \leq \infty$, which are needed for the analysis of time dependent partial differential equations, we first define their norms for a function $u : (0, T) \times \Omega \to \mathbb{R}$, which are given by

$$\|u\|_{L^p((0,T),\Omega)} = \left(\int_0^T \|u(t)\|_{L^p(\Omega)}^p \, \mathrm{d}t \right)^{\frac{1}{p}}, \qquad 1 \leq p < \infty,$$
$$\|u\|_{L^\infty((0,T),\Omega)} = \operatorname*{ess\,sup}_{t \in (0,T)} \|u(t)\|_{L^\infty(\Omega)}.$$

For $1 \leq p \leq \infty$, the **Bochner space** is defined as

$$L^p((0, T), \Omega) = \{f : (0, T) \times \Omega \to \mathbb{R} \mid f \text{ Bochner measurable }, \|f\|_{L^p((0,T),\Omega)} < \infty\}.$$

More information on Bochner spaces can be found in [23].

2.5 Differentiation in Banach spaces

Let X and Y denote two Banach spaces and $f : X \to Y$ be a mapping. Then, the notion of a derivative from elementary calculus textbooks can also be extended to Banach spaces. Herein, the **Gâteux detivative** $f'(x)\delta x$ plays the role of the directional derivative of f at the point $x \in X$ in the direction of $\delta x \in X$, i.e.

$$f'(x)\delta x = \lim_{\epsilon \to 0} \frac{f(x + \epsilon\delta x) - f(x)}{\epsilon} = \frac{\mathrm{d}}{\mathrm{d}\epsilon} f(x + \epsilon\delta x) \mid_{\epsilon=0} .$$

The **Fréchet derivative** at point $x \in X$ exists, if there is a linear operator $A \in L(X, Y)$ and a mapping $r(x, \cdot) : X \to Y$ such that

$$f(x + h) = f(x) + Ah + r(u, h) \qquad \forall h \in X$$

and

$$\frac{\|r(u, h)\|_Y}{\|h\|_X} \xrightarrow[\|h\|_X \to 0]{} 0.$$

Then A is the Fréchet derivative at point x and we write $f'(x) = A$.

2.6 Optimization

Let us consider the equality constrained nonlinear minimization problem

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} f(\boldsymbol{x}) \qquad \text{s.t.} \qquad \boldsymbol{c}(\boldsymbol{x}) = \boldsymbol{0},$$

with the objective function $f : \mathbb{R}^n \to \mathbb{R}$ and the equality constraints $\boldsymbol{c} : \mathbb{R}^n \to \mathbb{R}^m$ with $m \leq n$. We call a point $x \in \mathbb{R}^n$ **admissible**, if the constraints are satisfied, i.e. $\boldsymbol{c}(\boldsymbol{x}) = \boldsymbol{0}$. Let us define the **Lagrange function** $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ as

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^T \boldsymbol{c}(\boldsymbol{x}).$$

Let f and $\boldsymbol{c}$ be continuously differentiable, $\bar{\boldsymbol{x}}$ be a local solution to the minimization problem and $\{\nabla c_i(\bar{\boldsymbol{x}})\}_{i=1}^m$ be linearly independent. Then, there exists a unique **Lagrange multiplier** $\bar{\boldsymbol{\lambda}} = (\bar{\lambda}_1, \ldots, \bar{\lambda}_m) \in \mathbb{R}^m$ such that

$$\begin{aligned} \nabla_{\boldsymbol{x}} \mathcal{L}(\bar{\boldsymbol{x}}, \bar{\boldsymbol{\lambda}}) &= \nabla f(\bar{\boldsymbol{x}}) + \nabla \boldsymbol{c}(\bar{\boldsymbol{x}})^T \bar{\boldsymbol{\lambda}} = \boldsymbol{0}, \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}(\bar{\boldsymbol{x}}, \bar{\boldsymbol{\lambda}}) &= \boldsymbol{c}(\bar{\boldsymbol{x}}) = \boldsymbol{0}. \end{aligned}$$

These are the **Karush-Kuhn-Tucker conditions** (KKT conditions), which are first order necessary optimality conditions. Additionally, let f and $\boldsymbol{c}$ be twice continuously differentiable. Then, the Hessian matrix of the Lagrange function is positive semidefinite on $\ker \nabla \boldsymbol{c}(\bar{\boldsymbol{x}})$, i.e.

$$\boldsymbol{v}^T \nabla^2_{\boldsymbol{xx}} \mathcal{L}(\bar{\boldsymbol{x}}, \bar{\boldsymbol{\lambda}}) \boldsymbol{v} \geq 0 \qquad \forall\ \boldsymbol{v} \in \ker \nabla \boldsymbol{c}(\bar{\boldsymbol{x}}).$$

These are the second order necessary optimality conditions for a local minimum. More information on optimization can be found in [53].

2.7 Acronyms

Acronym	Meaning
CG	Conjuagte Gradient
DoF	Degree of Freedom
FEM	Finite Element Method
KKT	Karush-Kuhn-Tucker
LBB	Ladyzhenskaya-Babuska-Brezzi
NN	Neural Network
PDE	Partial Differential Equation
POD	Proper Orthogonal Decomposition
POD-ROM	Proper Orthogonal Decomposition based Reduced Order Modelling
RMSE	Root Mean Squared Error
ROM	Reduced Order Modelling
SSOR	Symmetric Successive Overrelaxation
SVD	Singular Value Decomposition

Figure 2.1: Acronyms used in this thesis

Chapter 3

The POD in $\mathbb{R}^m$

In this chapter, we introduce the main idea behind the proper orthogonal decomposition (POD) and how it can be used used in $\mathbb{R}^m$. The main objective of this method is finding low dimensional approximations to the data, which preserve the essential information of some high dimensional data set, e.g. solutions from a direct numerical simulation. The proper orthogonal decomposition is also being used in many other disciplines, where it is being referred to as the Karhunen-Loève decomposition (stochastics) [40, 47], Principal Component Analysis (data analysis) [39], Hotelling transform (statistics) [35] or emprirical orthogonal functions (meteorology and geophysics) [57]. For more in depth introductions to the proper orthogonal decomposition, see also [67, 18, 68, 30]. The following chapter largely follows [68].

3.1 Introduction

Let $Y = [\boldsymbol{u}_1, \dots, \boldsymbol{u}_n] \in \mathbb{R}^{m\times n}$ be a real-valued matrix of rank $d \leq \min(m, n)$, whose columns $\boldsymbol{u}_j \in \mathbb{R}^m, 1 \leq j \leq n$ are called snapshots. The Singular Value Decomposition (SVD) [52, 28] yields the factorization

$$Y = \Psi\Sigma\Phi^T \tag{3.1}$$

of the snapshot matrix Y, where $\Psi = [\boldsymbol{\psi}_1, \dots, \boldsymbol{\psi}_m] \in \mathbb{R}^{m\times m}$ and $\Phi = [\boldsymbol{\phi}_1, \dots, \boldsymbol{\phi}_n] \in \mathbb{R}^{n\times n}$ are orthogonal matrices,

$$\Sigma = \begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix} \in \mathbb{R}^{m\times n},$$

with $D = \operatorname{diag}(\sigma_1, \dots, \sigma_d) \in \mathbb{R}^{d\times d}$ and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_d > 0$. Then $\{\boldsymbol{\psi}_i\}_{i=1}^d$ and $\{\boldsymbol{\phi}_i\}_{i=1}^d$ are the eigenvectors of YY^T and Y^TY with eigenvalues $\lambda_i = \sigma_i^2 > 0$ for $1 \leq i \leq d$, due to the relations

$$Y\boldsymbol{\phi}_i = \sigma_i\boldsymbol{\psi}_i \text{ and } Y^T\boldsymbol{\psi}_i = \sigma_i\boldsymbol{\phi}_i \quad \text{for } 1 \leq i \leq d.$$

This will later play a crucial role when we introduce the method of snapshots. Using the fact that Σ has rank d, we simplify (3.1) to

$$Y = \Psi^{(d)} D \left(\Phi^{(d)}\right)^T = \sum_{i=1}^{d} \sigma_i \boldsymbol{\psi}_i \boldsymbol{\phi}_i^T, \tag{3.2}$$

where $\Psi^{(d)} = [\boldsymbol{\psi}_1, \dots, \boldsymbol{\psi}_d] \in \mathbb{R}^{m\times d}$ and $\Phi^{(d)} = [\boldsymbol{\phi}_1, \dots, \boldsymbol{\phi}_d] \in \mathbb{R}^{n\times d}$. From (3.2), we deduce that

$$\begin{aligned} \boldsymbol{u}_j &= \sum_{i=1}^{d} \left[D \left(\Phi^{(d)}\right)^T \right]_{ij} \boldsymbol{\psi}_i = \sum_{i=1}^{d} \Big[\underbrace{\left(\Psi^{(d)}\right)^T \Psi^{(d)}}_{=I\in\mathbb{R}^{d\times d}} D \left(\Phi^{(d)}\right)^T \Big]_{ij} \boldsymbol{\psi}_i \\ &\overset{(3.2)}{=} \sum_{i=1}^{d} \left[\left(\Psi^{(d)}\right)^T Y \right]_{ij} \boldsymbol{\psi}_i = \sum_{i=1}^{d} (\boldsymbol{\psi}_i, \boldsymbol{u}_j)_{\mathbb{R}^m} \boldsymbol{\psi}_i. \end{aligned}$$

We have thus derived the Fourier series representation of $\boldsymbol{u}_j$, which has the form

$$\boldsymbol{u}_j = \sum_{i=1}^{d} (\boldsymbol{u}_j, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \boldsymbol{\psi}_i.$$

3.2 Image compression

In a short interlude, we will now demonstrate how images can be compressed with the help of the singular value decomposition. In the case of image compression [18][Section 2.4], we are given a grayscale image $Y \in \mathbb{R}^{m \times n}$ and are interested in finding a matrix $\tilde{Y} \in \mathbb{R}^{m \times n}$ of rank r with $r \ll d$ such that $\| Y - \tilde{Y} \|$ is minimal. Then we have by the Eckart-Young Theorem [32] that

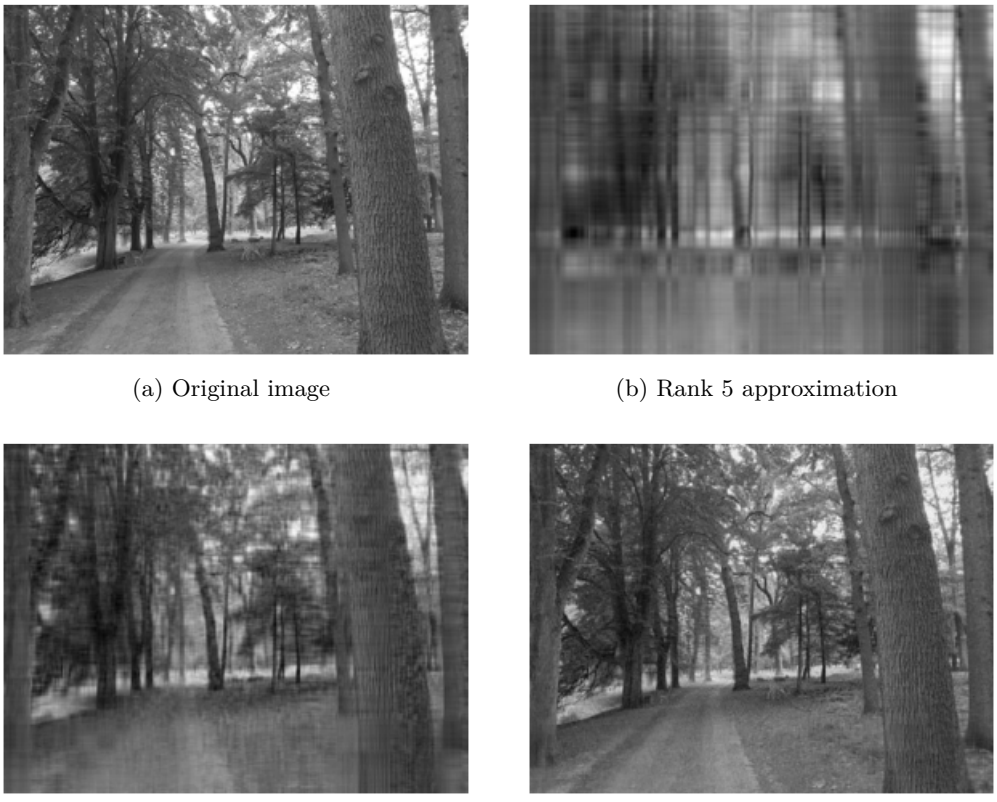
$$\min_{\operatorname{rank}(\tilde{Y})=r} \left\| Y - \tilde{Y} \right\| = \left\| Y - Y^{(r)} \right\| = \begin{cases} \sigma_{r+1} & \text{for the 2-norm,} \\ \sqrt{\sum_{i=r+1}^{d} \sigma_i^2} & \text{for the Frobenius norm,} \end{cases} \tag{3.3}$$

which means that the truncated SVD matrix

$$Y^{(r)} := \Psi \begin{pmatrix} \Sigma^{(r)} & 0 \\ 0 & 0 \end{pmatrix} \Phi^T = \sum_{i=1}^{r} \sigma_i \boldsymbol{\psi}_i \boldsymbol{\phi}_i^T \tag{3.4}$$

with $\Sigma^{(r)} := \operatorname{diag}(\Sigma_{11}, \ldots, \Sigma_{rr})$ solves the minimization problem. Therefore in the application of the POD for image compression, we simply compute a truncated SVD which has a sufficiently high information content.

We use a grayscale image of a forest, which is 400 pixels wide and 300 pixels high. Each pixel has an integer value between 0 (black) and 255 (white). In Figure 3.1, we compare the original image of a forest with its SVD approximations of rank 5, 25 and 100.



(a) Original image

(b) Rank 5 approximation

(c) Rank 25 approximation

(d) Rank 100 approximation

Figure 3.1: Approximation of the original image by images of lower rank

The matrix representation of the image has full rank, i.e. it has rank 300. Its singular values, the diagonal entries of the matrix Σ in the SVD, are being shown in the following Figure.

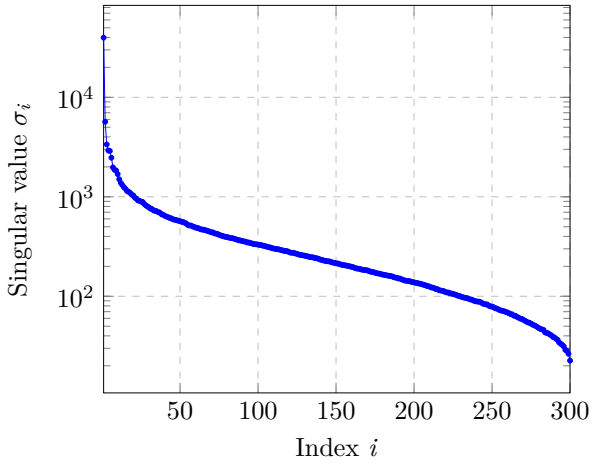


Figure 3.2: Singular values of the image of the forest

In Figure 3.2, we can see that the singular values decay rapidly. We thus expect the rank r approximation $Y^{(r)}$ from equation (3.4) to be a good approximation to the data matrix Y for even small values of r.

The only question that remains unanswered is how we should choose r. In practice, r is often being determined heuristically, by choosing r sufficiently large such that the ratio of the energy of $Y^{(r)}$ to the energy of Y is bigger than some threshold, i.e. finding the smallest $r \in \mathbb{N}^+$ such that

$$\mathcal{E}(r) := \frac{\sum_{i=1}^{r} \sigma_i}{\sum_{i=1}^{d} \sigma_i} \geq 99\ \%.$$

However, when we later introduce the proper orthogonal decomposition for partial differential equations, in the energy ratio we don't sum over the singular values σ_i but over the eigenvalues $\lambda_i = \sigma_i^2$.

In the next Figure, it is being shown how the energy ratio $\mathcal{E}(r)$ depends on the size r of the truncated SVD and that a big portion of the energy ratio can be attributed to the first few singular values, which motivates a truncated singular value decomposition.

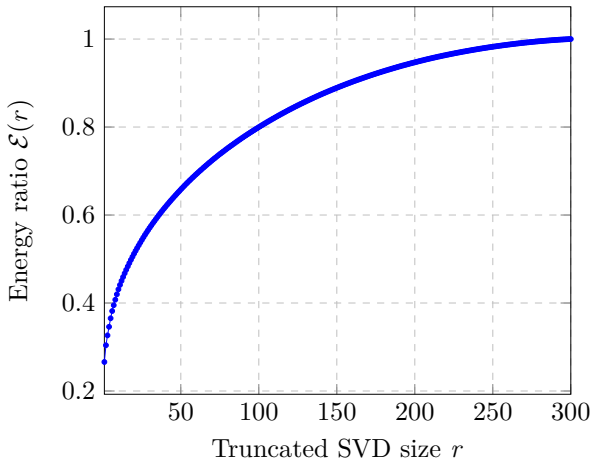


Figure 3.3: Energy ratios in dependence on the size of the truncated SVD

In Figure 3.3, we can observe that the rank 5, rank 25 and rank 100 approximations from Figure 3.1 have energy ratios of 39 %, 55 % and 80 % respectively.

3.3 The connection between the POD and the SVD

Let us now return to the proper orthogonal decomposition, derive it through a series of optimization problems and elucidate how it is connected to the singular value decomposition. We will now iteratively compute an orthonormal basis, which approximates the data set $\{\boldsymbol{u}_i\}_{i=1}^n$ as good as possible. These vectors $\{\boldsymbol{\psi}_i\}_{i=1}^r$ from our computations will be called the POD basis of rank r and we will refer to $\boldsymbol{\psi}_i$ as the POD basis vectors or POD modes. The first POD basis vector solves the maximization problem

$$\max_{\tilde{\boldsymbol{\psi}}_1 \in \mathbb{R}^m} \sum_{j=1}^{n} |(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_1)_{\mathbb{R}^m}|^2 \quad \text{s.t.} \quad \|\tilde{\boldsymbol{\psi}}_1\|_{\mathbb{R}^m}^2 = 1, \tag{$\mathbf{P}^1$}$$

and it can be shown through the Lagrange formalism that the first left singular vector $\boldsymbol{\psi}_1$ solves ($\mathbf{P}^1$). The second basis vector is the solution of

$$\max_{\tilde{\boldsymbol{\psi}}_2 \in \mathbb{R}^m} \sum_{j=1}^{n} |(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_2)_{\mathbb{R}^m}|^2 \quad \text{s.t.} \quad \|\tilde{\boldsymbol{\psi}}_2\|_{\mathbb{R}^m}^2 = 1,\ (\boldsymbol{\psi}_1, \tilde{\boldsymbol{\psi}}_2)_{\mathbb{R}^m} = 0, \tag{$\mathbf{P}^2$}$$

and is given by the second left singular vector $\boldsymbol{\psi}_2$. Continuing this procedure inductively, we arrive at the following theorem.

Theorem 3.3.1 (POD basis)**.** *Let $Y = [\boldsymbol{u}_1, \dots, \boldsymbol{u}_n] \in \mathbb{R}^{m \times n}$ with rank $d \leq \min(m, n)$. Moreover, let $Y = \Psi \Sigma \Phi^T$ be the singular value decomposition as described in (3.1). Then for $1 \leq r \leq d$ the optimization problem*

$$\max_{\tilde{\boldsymbol{\psi}}_1, \dots, \tilde{\boldsymbol{\psi}}_r \in \mathbb{R}^m} \sum_{i=1}^{r} \sum_{j=1}^{n} |(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_i)_{\mathbb{R}^m}|^2 \quad \textit{s.t.} \quad (\tilde{\boldsymbol{\psi}}_i, \tilde{\boldsymbol{\psi}}_j)_{\mathbb{R}^m} = \delta_{ij}\ \forall 1 \leq i, j \leq r \tag{$\mathbf{P}^r$}$$

is being solved by the left singular vectors $\{\boldsymbol{\psi}_i\}_{i=1}^r$ and it holds that

$$\arg\max(\boldsymbol{P}^r) = \sum_{i=1}^{r} \sigma_i^2 = \sum_{i=1}^{r} \lambda_i.$$

Proof. Provided in Appendix A.1. □

Having made the connection between the POD basis and the SVD through Theorem 3.3.1, we again get the optimality of the POD basis from the Eckart-Young theorem from (3.3), which means that

$$\left\| Y - Y^{(r)} \right\|_F^2 \leq \left\| Y - \tilde{Y} \right\|_F^2 \tag{3.5}$$

for all matrices $\tilde{Y} \in \mathbb{R}^{m \times n}$ of rank r. The optimal rank r approximation $Y^{(r)}$ defined in (3.4) can be rewritten as

$$Y^{(r)} = \Psi \begin{pmatrix} \Sigma^{(r)} & 0 \\ 0 & 0 \end{pmatrix} \Phi^T = \Psi^{(r)} \underbrace{\Sigma^{(r)} \left(\Phi^{(r)} \right)^T}_{=: B^{(r)}},$$

where $\Psi^{(r)} = [\boldsymbol{\psi}_1, \dots, \boldsymbol{\psi}_r] \in \mathbb{R}^{m \times r}, \Phi^{(r)} = [\boldsymbol{\phi}_1, \dots, \boldsymbol{\phi}_r] \in \mathbb{R}^{n \times r}$ and $\Sigma^{(r)} := \operatorname{diag}(\Sigma_{11}, \dots, \Sigma_{rr})$. Plugging this into (3.5) and using the definition of the Frobenius norm yields

$$\begin{aligned} \left\| Y - Y^{(r)} \right\|_F^2 = \left\| Y - \Psi^{(r)} B^{(r)} \right\|_F^2 &= \sum_{i=1}^{m} \sum_{j=1}^{n} \left| Y_{ij} - \sum_{k=1}^{r} \Psi_{ik}^{(r)} B_{kj}^{(r)} \right|^2 \\ &= \sum_{i=1}^{m} \sum_{j=1}^{n} \left| Y_{ij} - \sum_{k=1}^{r} (\boldsymbol{u}_j, \boldsymbol{\psi}_k)_{\mathbb{R}^m} \Psi_{ik}^{(r)} \right|^2 \\ &= \sum_{j=1}^{n} \left\| \boldsymbol{u}_j - \sum_{k=1}^{r} (\boldsymbol{u}_j, \boldsymbol{\psi}_k)_{\mathbb{R}^m} \boldsymbol{\psi}_k \right\|_{\mathbb{R}^m}^2. \end{aligned} \tag{3.6}$$

Let us now consider another decomposition of $Y = \tilde{\Psi}^{(d)} C^{(d)}$, where the columns $\{\tilde{\boldsymbol{\psi}}_i\}_{i=1}^d$ of $\tilde{\Psi}^{(d)} \in \mathbb{R}^{m\times d}$ are pairwise orthonormal and

$$C_{ij}^{(d)} = \left(\tilde{\boldsymbol{\psi}}_i, \boldsymbol{u}_j\right)_{\mathbb{R}^m} \quad \text{for } 1 \le i \le d, 1 \le j \le n.$$

For $\tilde{Y} := \tilde{\Psi}^{(r)} C^{(r)}$, where $\tilde{\Psi}^{(r)} \in \mathbb{R}^{m\times r}$ denotes the matrix with the first r columns of $\tilde{\Psi}^{(d)}$ and $C^{(r)} \in \mathbb{R}^{r\times n}$ denotes the matrix with the first r rows of $C^{(d)}$, we get analogously to (3.6) that

$$\left\| Y - \tilde{Y} \right\|_F^2 = \sum_{j=1}^n \left\| \boldsymbol{u}_j - \sum_{k=1}^r \left(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_k\right)_{\mathbb{R}^m} \tilde{\boldsymbol{\psi}}_k \right\|_{\mathbb{R}^m}^2,$$

and hence by the Eckart-Young Theorem for the Frobenius norm (3.5), we get

$$\sum_{j=1}^n \left\| \boldsymbol{u}_j - \sum_{k=1}^r \left(\boldsymbol{u}_j, \boldsymbol{\psi}_k\right)_{\mathbb{R}^m} \boldsymbol{\psi}_k \right\|_{\mathbb{R}^m}^2 \le \sum_{j=1}^n \left\| \boldsymbol{u}_j - \sum_{k=1}^r \left(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_k\right)_{\mathbb{R}^m} \tilde{\boldsymbol{\psi}}_k \right\|_{\mathbb{R}^m}^2.$$

We have thus shown that

$$\min_{\tilde{\boldsymbol{\psi}}_1,\dots,\tilde{\boldsymbol{\psi}}_r \in \mathbb{R}^m} \sum_{j=1}^n \left\| \boldsymbol{u}_j - \sum_{i=1}^r \left(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_i\right)_{\mathbb{R}^m} \tilde{\boldsymbol{\psi}}_i \right\|_{\mathbb{R}^m}^2 \quad \text{s.t.} \quad (\tilde{\boldsymbol{\psi}}_i, \tilde{\boldsymbol{\psi}}_j)_{\mathbb{R}^m} = \delta_{ij} \ \forall 1 \le i,j \le r \qquad (\hat{\mathbf{P}}^r)$$

is an optimization problem that is equivalent to $(\mathbf{P}^r)$. When determining a POD basis of rank r, computing the POD modes with the highest energy (see $(\mathbf{P}^r)$) and computing the POD modes with minimal error between $\boldsymbol{u}_j$ and $\boldsymbol{u}_j^{(r)} := \sum_{i=1}^r (\boldsymbol{u}_j, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \boldsymbol{\psi}_i$ (see $(\hat{\mathbf{P}}^r)$) yields the same basis vectors, namely the first r left singular vectors of Y.

In practice, computing the singular value decomposition of Y is computationally expensive or even infeasible, since $Y \in \mathbb{R}^{m\times n}$ is a dense matrix and m or n are very large. However, we can transform the search for the POD basis into an eigenvalue problem of size $m \times m$ or $n \times n$, which comes in handy if $m \ll n$ or $n \ll m$. For the sake of completeness, we mention the classical method introduced by Lumley [48] in 1967, where one needs to solve the $m \times m$ eigenvalue problem

$$YY^T \boldsymbol{\psi}_i = \lambda \boldsymbol{\psi}_i \quad \text{for } 1 \le i \le r,$$

whose eigenvectors are the left singular vectors $\{\boldsymbol{\psi}_i\}_{i=1}^r$ and the eigenvalues are $\lambda_i = \sigma_i^2$ for $1 \le i \le r$. This method is only practical for $m \ll n$ and we will see later on in Chapter 4 that m will be the number of degrees of freedom in the spatial discretization of the PDE, which is much larger than the number of time steps n. Hence, the classical method is not being used for reduced order modelling of PDEs. Therefore, in most applications the method of snapshots, which has been introduced by Sirovich [62, 63, 64] in 1987, is being used. Herein, we need to solve the $n \times n$ eigenvalue problem

$$Y^T Y \boldsymbol{\phi}_i = \lambda_i \boldsymbol{\phi}_i \quad \text{for } 1 \le i \le r, \tag{3.7}$$

whose eigenvectors are the right singular vectors $\{\boldsymbol{\phi}_i\}_{i=1}^r$ and the eigenvalues are $\lambda_i = \sigma_i^2$ for $1 \le i \le r$. From this the POD basis $\{\boldsymbol{\psi}_i\}_{i=1}^r$ can be calculated through the relation

$$\boldsymbol{\psi}_i = \frac{1}{\sqrt{\lambda_i}} Y \boldsymbol{\phi}_i \quad \text{for } 1 \le i \le r.$$

Finally, we mention that for the choice of r in real world problems the formula

$$\mathcal{E}(r) := \frac{\sum_{i=1}^r \lambda_i}{\sum_{i=1}^d \lambda_i} \ge \delta_{\mathcal{E}}$$

is being employed, where $\mathcal{E}(r)$ is called the energy ratio and the threshold $\delta_{\mathcal{E}} \in [0,1]$ is fixed. Since we want most of the energy to be preserved by the POD modes, we choose $\delta_{\mathcal{E}}$ close to 1, e.g. $\delta_{\mathcal{E}} = 0.99$.

Note that unlike in the case of image compression (Section 3.2), where we sum over the singular values σ_i in the energy ratio, we now sum over the eigenvalues $\lambda_i = \sigma_i^2$.

Our approach for the POD method in $\mathbb{R}^m$ can be summarized in the following algorithm.

Algorithm 1 POD basis in $\mathbb{R}^m$

Input: Snapshots $\{\boldsymbol{u}_j\}_{j=1}^n \subset \mathbb{R}^m$ and threshold $\delta_{\mathcal{E}} \in [0,1]$.
Output: POD basis $\{\boldsymbol{\psi}_i\}_{i=1}^r \subset \mathbb{R}^m$ and eigenvalues $\{\lambda_i\}_{i=1}^r$.

Set $Y = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n] \in \mathbb{R}^{m\times n}$.
if $m \approx n$ **then**
  Compute singular value decomposition $[\Psi, \Sigma, \Phi] = \mathrm{SVD}(Y)$.
  Compute $r = \min\left\{ r \in \mathbb{N} \;\middle|\; \mathcal{E}(r) = \sum_{i=1}^r \Sigma_{ii}^2 / \sum_{i=1}^d \Sigma_{ii}^2 \geq \delta_{\mathcal{E}}, \;\; 1 \leq r \leq d \right\}$.
  Set $\lambda_i = \Sigma_{ii}^2$ and $\boldsymbol{\psi}_i = \Psi_{\cdot,i} \in \mathbb{R}^m$ for $1 \leq i \leq r$.
else if $m \ll n$ **then**
  Compute eigenvalue decomposition $[\Psi, \Lambda] = \mathrm{Eig}(YY^T)$, where $YY^T \in \mathbb{R}^{m\times m}$.
  Compute $r = \min\left\{ r \in \mathbb{N} \;\middle|\; \mathcal{E}(r) = \sum_{i=1}^r \Lambda_{ii} / \sum_{i=1}^d \Lambda_{ii} \geq \delta_{\mathcal{E}}, \;\; 1 \leq r \leq d \right\}$.
  Set $\lambda_i = \Lambda_{ii}$ and $\boldsymbol{\psi}_i = \Psi_{\cdot,i} \in \mathbb{R}^m$ for $1 \leq i \leq r$.
else if $n \ll m$ **then**
  Compute eigenvalue decomposition $[\Phi, \Lambda] = \mathrm{Eig}(Y^TY)$, where $Y^TY \in \mathbb{R}^{n\times n}$.
  Compute $r = \min\left\{ r \in \mathbb{N} \;\middle|\; \mathcal{E}(r) = \sum_{i=1}^r \Lambda_{ii} / \sum_{i=1}^d \Lambda_{ii} \geq \delta_{\mathcal{E}}, \;\; 1 \leq r \leq d \right\}$.
  Set $\lambda_i = \Lambda_{ii}$ and $\boldsymbol{\psi}_i = Y\Phi_{\cdot,i}/\sqrt{\lambda_i} \in \mathbb{R}^m$ for $1 \leq i \leq r$.

3.4 POD with weighted inner product

In the next chapter, we will not be working with the Euclidean inner product anymore, but with the L^2-inner product. For two functions

$$\psi := \sum_{j=1}^m \psi_j^h \varphi_j^h, \;\; \phi := \sum_{j=1}^m \phi_j^h \varphi_j^h \;\; \in L^2(\Omega),$$

we will calculate their L^2-inner product as

$$(\psi, \phi)_{L^2(\Omega)} = \left(\sum_{j=1}^m \psi_j^h \varphi_j^h, \sum_{i=1}^m \phi_i^h \varphi_i^h \right)_{L^2(\Omega)} = \sum_{i,j=1}^m \psi_j^h (\varphi_j^h, \varphi_i^h)_{L^2(\Omega)} \phi_i^h = (\boldsymbol{\psi}^h)^T M_h \boldsymbol{\phi}^h, \tag{3.8}$$

where $M_h \in \mathbb{R}^{m\times m}$ is the mass matrix with entries $(M_h)_{ij} = (\varphi_j^h, \varphi_i^h)_{L^2(\Omega)}$ for $1 \leq i,j \leq m$. We observe that the L^2-inner product of two functions ψ and ϕ corresponds to a weighted inner product of their coefficient vectors $\boldsymbol{\psi}^h = (\psi_1^h, \ldots, \psi_m^h) \in \mathbb{R}^m$ and $\boldsymbol{\phi}^h = (\phi_1^h, \ldots, \phi_m^h) \in \mathbb{R}^m$. Therefore, let us now extend the POD from Section 3.3 to weighted inner products.

We consider a weighted inner product

$$(\boldsymbol{\psi}, \boldsymbol{\phi})_W := \boldsymbol{\psi}^T W \boldsymbol{\phi} = (\boldsymbol{\psi}, W\boldsymbol{\phi})_{\mathbb{R}^m} = (W\boldsymbol{\psi}, \boldsymbol{\phi})_{\mathbb{R}^m} \quad \text{for } \boldsymbol{\psi}, \boldsymbol{\phi} \in \mathbb{R}^m,$$

where $W \in \mathbb{R}^{m\times m}$ is a symmetric, positive definite matrix. The weighted inner product induces the norm $\|\boldsymbol{\psi}\|_W = \sqrt{(\boldsymbol{\psi}, \boldsymbol{\psi})_W}$ for $\boldsymbol{\psi} \in \mathbb{R}^m$. If we choose W to be the identity matrix in $\mathbb{R}^{m\times m}$, then the weighted inner product simplifies to the Euclidean inner product.

To determine the POD vectors, we first replace $(\mathbf{P}^1)$ by

$$\max_{\tilde{\boldsymbol{\psi}}_1 \in \mathbb{R}^m} \sum_{j=1}^n |(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_1)_W|^2 \quad \text{s.t.} \quad \|\tilde{\boldsymbol{\psi}}_1\|_W^2 = 1, \tag{$\mathbf{P}^1_W$}$$

and then we replace ($\mathbf{P}^2$) by

$$\max_{\tilde{\boldsymbol{\psi}}_2 \in \mathbb{R}^m} \sum_{j=1}^{n} |(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_2)_W|^2 \quad \text{s.t.} \quad \|\tilde{\boldsymbol{\psi}}_2\|_W^2 = 1,\ (\boldsymbol{\psi}_1, \tilde{\boldsymbol{\psi}}_2)_W = 0. \tag{$\mathbf{P}^2_W$}$$

By induction, we arrive at the following theorem.

Theorem 3.4.1 (POD basis with weighted inner product)**.** *Let $Y = [\boldsymbol{u}_1, \dots, \boldsymbol{u}_n] \in \mathbb{R}^{m\times n}$ with rank $d \leq \min(m, n)$, $W \in \mathbb{R}^{m\times m}$ be a symmetric, positive definite matrix and $\bar{Y} = W^{\frac{1}{2}} Y$. Moreover, let $\bar{Y} = \bar{\Psi}\Sigma\bar{\Phi}^T$ be the singular value decomposition of $\bar{Y}$, where $\bar{\Psi} = [\bar{\boldsymbol{\psi}}_1, \dots, \bar{\boldsymbol{\psi}}_m] \in \mathbb{R}^{m\times m}$ and $\bar{\Phi} = [\bar{\boldsymbol{\phi}}_1, \dots, \bar{\boldsymbol{\phi}}_n] \in \mathbb{R}^{n\times n}$ are orthogonal matrices,*

$$\Sigma = \begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix} \in \mathbb{R}^{m\times n},$$

with $D = \operatorname{diag}(\sigma_1, \dots, \sigma_d) \in \mathbb{R}^{d\times d}$ and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_d > 0$. Then for $1 \leq r \leq d$ the optimization problem

$$\max_{\tilde{\boldsymbol{\psi}}_1, \dots, \tilde{\boldsymbol{\psi}}_r \in \mathbb{R}^m} \sum_{i=1}^{r} \sum_{j=1}^{n} |(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_i)_W|^2 \quad \textit{s.t.} \quad (\tilde{\boldsymbol{\psi}}_i, \tilde{\boldsymbol{\psi}}_j)_W = \delta_{ij}\ \forall 1 \leq i, j \leq r \tag{$\mathbf{P}^r_W$}$$

is being solved by the vectors $\boldsymbol{\psi}_i = W^{-\frac{1}{2}} \bar{\boldsymbol{\psi}}_i$ for $1 \leq i \leq r$ and it holds that

$$\arg\max(\boldsymbol{P}^r_W) = \sum_{i=1}^{r} \sigma_i^2 = \sum_{i=1}^{r} \lambda_i.$$

Proof. The proof follows the same steps as Theorem 3.3.1. □

Remark 3.4.2. *One can again show that*

$$\min_{\tilde{\boldsymbol{\psi}}_1, \dots, \tilde{\boldsymbol{\psi}}_r \in \mathbb{R}^m} \sum_{j=1}^{n} \left\| \boldsymbol{u}_j - \sum_{i=1}^{r} \left(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_i \right)_W \tilde{\boldsymbol{\psi}}_i \right\|_W^2 \quad \textit{s.t.} \quad (\tilde{\boldsymbol{\psi}}_i, \tilde{\boldsymbol{\psi}}_j)_W = \delta_{ij}\ \forall 1 \leq i, j \leq r \tag{$\hat{\mathbf{P}}^r_W$}$$

is an optimization problem that is equivalent to ($\boldsymbol{P}^r_W$).

Remark 3.4.3. *It holds that $\bar{Y}^T \bar{Y} = Y^T W Y$ and thus due to the singular value decomposition, the method of snapshots reads: Solve the $n \times n$ eigenvalue problem*

$$Y^T W Y \bar{\boldsymbol{\phi}}_i = \lambda_i \bar{\boldsymbol{\phi}}_i \quad \textit{for } 1 \leq i \leq i \leq r,$$

and set

$$\boldsymbol{\psi}_i = W^{-\frac{1}{2}} \bar{\boldsymbol{\psi}}_i = \frac{1}{\sqrt{\lambda_i}} W^{-\frac{1}{2}} \left(\bar{Y} \bar{\boldsymbol{\phi}}_i \right) = \frac{1}{\sqrt{\lambda_i}} W^{-\frac{1}{2}} W^{\frac{1}{2}} Y \bar{\boldsymbol{\phi}}_i = \frac{1}{\sqrt{\lambda_i}} Y \bar{\boldsymbol{\phi}}_i,$$

for $1 \leq i \leq r$. Notice that

$$(\boldsymbol{\psi}_i, \boldsymbol{\psi}_j)_W = \boldsymbol{\psi}_i^T W \boldsymbol{\psi}_j = \frac{\delta_{ij}\lambda_j}{\sqrt{\lambda_i \lambda_j}} = \delta_{ij} \quad \forall 1 \leq i, j \leq r,$$

and that the matrix $W^{\frac{1}{2}}$ is not needed for the method of snapshots.

We can thus extend Algorithm 1 for the proper orthogonal decomposition in $\mathbb{R}^m$ to weighted inner products now.

Algorithm 2 POD basis in $\mathbb{R}^m$ with weighted inner product

Input: Snapshots $\{\boldsymbol{u}_j\}_{j=1}^n \subset \mathbb{R}^m$, threshold $\delta_{\mathcal{E}} \in [0,1]$ and symmetric, positive definite matrix $W \in \mathbb{R}^{m\times m}$.

Output: POD basis $\{\boldsymbol{\psi}_i\}_{i=1}^r \subset \mathbb{R}^m$ and eigenvalues $\{\lambda_i\}_{i=1}^r$.

Set $Y = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n] \in \mathbb{R}^{m\times n}$.
if $m \approx n$ **then**
  Determine $\bar{Y} = W^{\frac{1}{2}} Y \in \mathbb{R}^{m\times n}$.
  Compute singular value decomposition $[\bar{\Psi}, \Sigma, \bar{\Phi}] = \mathrm{SVD}(\bar{Y})$.
  Compute $r = \min\left\{ r \in \mathbb{N} \;\middle|\; \mathcal{E}(r) = \sum_{i=1}^r \Sigma_{ii}^2 / \sum_{i=1}^d \Sigma_{ii}^2 \geq \delta_{\mathcal{E}}, \;\; 1 \leq r \leq d \right\}$.
  Set $\lambda_i = \Sigma_{ii}^2$ and $\boldsymbol{\psi}_i = W^{-\frac{1}{2}} \bar{\Psi}_{\cdot,i} \in \mathbb{R}^m$ for $1 \leq i \leq r$.
else if $m \ll n$ **then**
  Determine $\bar{Y} = W^{\frac{1}{2}} Y \in \mathbb{R}^{m\times n}$.
  Compute eigenvalue decomposition $[\bar{\Psi}, \Lambda] = \mathrm{Eig}(\bar{Y}\,\bar{Y}^T)$, where $\bar{Y}\,\bar{Y}^T \in \mathbb{R}^{m\times m}$.
  Compute $r = \min\left\{ r \in \mathbb{N} \;\middle|\; \mathcal{E}(r) = \sum_{i=1}^r \Lambda_{ii} / \sum_{i=1}^d \Lambda_{ii} \geq \delta_{\mathcal{E}}, \;\; 1 \leq r \leq d \right\}$.
  Set $\lambda_i = \Lambda_{ii}$ and $\boldsymbol{\psi}_i = W^{-\frac{1}{2}} \bar{\Psi}_{\cdot,i} \in \mathbb{R}^m$ for $1 \leq i \leq r$.
else if $n \ll m$ **then**
  Compute eigenvalue decomposition $[\bar{\Phi}, \Lambda] = \mathrm{Eig}(Y^T W Y)$, where $Y^T W Y \in \mathbb{R}^{n\times n}$.
  Compute $r = \min\left\{ r \in \mathbb{N} \;\middle|\; \mathcal{E}(r) = \sum_{i=1}^r \Lambda_{ii} / \sum_{i=1}^d \Lambda_{ii} \geq \delta_{\mathcal{E}}, \;\; 1 \leq r \leq d \right\}$.
  Set $\lambda_i = \Lambda_{ii}$ and $\boldsymbol{\psi}_i = Y \bar{\Phi}_{\cdot,i} / \sqrt{\lambda_i} \in \mathbb{R}^m$ for $1 \leq i \leq r$.

Chapter 4

The POD for parabolic problems

In this chapter, we will investigate the usage of the proper orthogonal decomposition for reduced order modelling at the example of a prototypical parabolic partial differential equation, the heat equation. To build a reduced order model, we need to follow three steps. In the first step, the heat equation needs to be solved by direct numerical simulation. In this thesis, the Finite Element Method (FEM) with m degrees of freedom (DoF) yields n solutions $\boldsymbol{\eta}^{h,1}, \dots, \boldsymbol{\eta}^{h,n}$ at the time steps $t_1, ..., t_n$. These solutions $\boldsymbol{\eta}^{h,i} := \boldsymbol{\eta}^h(t_i)$ are called snapshots and make up the snapshot matrix $Y = [\boldsymbol{\eta}^{h,1}, ..., \boldsymbol{\eta}^{h,n}] \in \mathbb{R}^{m \times n}$. In the second step, the POD basis in $\mathbb{R}^m$ with weighted inner product from Section 3.4 is being computed from the matrix Y with the method of snapshots, since the number of time steps n is usually much lower than the number of degrees of freedom m in the spatial FEM discretization. In the third step, the POD basis is being used to solve a reduced order model with r degrees of freedom, where $r \ll m$ is the size of the POD basis. We will refer to this three step process as proper orthogonal decomposition based reduced order modelling (POD-ROM). The interested reader can find more information on POD-ROM for parabolic PDEs in [43, 68, 49], [70][Chapter 4] and [26][Chapter 3].

4.1 Model problem and direct numerical simulation

Let $\Omega \subset \mathbb{R}^d$ with $d \in \{1, 2, 3\}$ be a bounded, open domain with Lipschitz boundary $\partial\Omega$. Let $0 < T < \infty$ be the end time. Let $H = L^2(\Omega)$ and $V = H_0^1(\Omega)$. Let $u_0 \in H$ be the initial condition and $f \in L^2((0, T), H)$ the right hand side. Then the strong formulation of the heat equation with homogeneous Dirichlet boundary conditions reads:

Formulation 4.1.1 (Heat equation)**.** *Find* $u : (0, T) \times \Omega \to \mathbb{R}$ *such that*

$$\begin{aligned} \partial_t u - \Delta u &= f \quad \text{in } (0, T) \times \Omega, \\ u &= 0 \quad \text{on } (0, T) \times \partial\Omega, \\ u(0, \cdot) &= u_0 \quad \text{in } \{0\} \times \Omega. \end{aligned}$$

Multiplying from the right hand side with a test function $\varphi \in V$, integrating over the domain Ω and integrating by parts yields the weak formulation:

Find $u \in W(0, T) := \{\varphi \in L^2((0, T), V) \mid \partial_t \varphi \in L^2((0, T), V^*)\}$ such that

$$(\partial_t u, \varphi) + (\nabla u, \nabla \varphi) = (f, \varphi) \quad \forall \varphi \in V.$$

This problem can now be solved with the finite element method by introducing a finite dimensional subspace $V_h = \text{span}\{\varphi_1^h, \dots, \varphi_m^h\} \subset V$. Plugging the finite element ansatz

$$u(t, \boldsymbol{x}) \approx u^h(t, \boldsymbol{x}) = \sum_{i=1}^{m} \eta_i^h(t) \varphi_i^h(\boldsymbol{x})$$

into the weak formulation gives us the vector valued ordinary differential equation (ODE)

$$\begin{aligned}\left(\partial_t u^h, \varphi_i^h\right) + \left(\nabla u^h, \nabla \varphi_i^h\right) &= \left(f, \varphi_i^h\right) \quad \forall 1 \le i \le m,\\ \Leftrightarrow \sum_{j=1}^{m} \left(\varphi_j^h, \varphi_i^h\right) \dot{\eta}_j^h(t) + \sum_{j=1}^{m} \left(\nabla \varphi_j^h, \nabla \varphi_i^h\right) \eta_j^h(t) &= \left(f, \varphi_i^h\right) \quad \forall 1 \le i \le m.\end{aligned}$$

In matrix notation this problem is then equivalent to:

Formulation 4.1.2 (FEM spatial discretization of heat equation)**.**

$$M_h \dot{\boldsymbol{\eta}}^h(t) + S_h \boldsymbol{\eta}^h(t) = \boldsymbol{f}^h(t) \quad \forall t \in (0, T), \tag{4.1a}$$

$$M_h \boldsymbol{\eta}^h(0) = \boldsymbol{\eta}_0^h, \tag{4.1b}$$

where

$$\begin{aligned}(M_h)_{ij} &:= \left(\varphi_j^h, \varphi_i^h\right) && \text{for } 1 \le i, j \le m,\\ (S_h)_{ij} &:= \left(\nabla \varphi_j^h, \nabla \varphi_i^h\right) && \text{for } 1 \le i, j \le m,\\ \left(f^h(t)\right)_i &:= \left(f(t), \varphi_i^h\right) && \text{for } 1 \le i \le m,\\ \left(\eta_0^h\right)_i &:= \left(u_0, \varphi_i^h\right) && \text{for } 1 \le i \le m.\end{aligned}$$

In general, for accurate finite element simulations the number of degrees of freedom m is very large and multiphysics calculations may take weeks or even a few months on modern supercomputers. Therefore, the proper orthogonal decomposition, which has been introduced in Chapter 3, now trades off accuracy for speed by reducing the dimensionality of the problem. It computes a low dimensional basis $\{\varphi_i^r\}_{i=1}^r$ with $r \ll m$ from snapshots of numerical simulations. This simplifies the ordinary differential equation (4.1) and leads to an enormous decrease in computation time.

In the following, we will use r as a sub- or superscript for vectors and matrices that arise from reduced order modelling. We will use h as a sub- or superscript for vectors and matrices that arise from finite element modelling. When it comes to additional indices, we will use superscripts to indicate the time step and subscripts for the spatial components.

4.2 Continuous version of POD

Before we take a closer look at how the method of snapshots works in the case of parabolic PDEs, let us first take a step back and derive the POD minimization problem for time-dependent PDEs from first principles. Let $u \in W(0, T)$ denote the analytical solution to the heat equation (Formulation 4.1.1) and let $X = H$ or $X = V$. We use the short hand notation $u(t) := u(t, \cdot)$ and define

$$\mathcal{V} := \operatorname{span}\{u(t) \mid t \in [0, T]\} \subseteq V \subseteq X,$$

which may have infinite dimension. Then the aim of the POD is finding a set of functions $\{\psi_i\}_{i=1}^r \subset X$ which solve

$$\min_{\tilde{\psi}_1, \dots, \tilde{\psi}_r \in X} \int_0^T \left\| u(t) - \sum_{i=1}^{r} \left(u(t), \tilde{\psi}_i\right)_X \tilde{\psi}_i \right\|_X^2 \mathrm{d}t \quad \text{s.t.} \quad (\tilde{\psi}_i, \tilde{\psi}_j)_X = \delta_{ij} \ \forall 1 \le i, j \le r. \tag{$\hat{\mathbf{P}}_C^r$}$$

To solve this optimization problem, we first introduce the bounded, compact, non-negative and self-adjoint operator $\mathcal{R} : X \to \mathcal{V} \subseteq X$ with

$$\mathcal{R}\psi = \int_0^T (\psi, u(t))_X u(t) \, \mathrm{d}t \quad \text{for } \psi \in X.$$

With the help of the Hilbert-Schmidt theory of compact operators, one can then show that the minimization problem $(\hat{\mathbf{P}}_C^r)$ is being solved by the first r eigenfunctions $\{\psi_i\}_{i=1}^r$ of $\mathcal{R}$ with corresponding eigenvalues $\{\lambda_i\}_{i=1}^r$ [68][Theorem 2.1.5] and we get the error identity

$$\int_0^T \left\| u(t) - \sum_{i=1}^{r} (u(t), \psi_i)_X \, \psi_i \right\|_X^2 \mathrm{d}t = \sum_{i=r+1}^{\infty} \lambda_i.$$

In practice, we are never dealing with the continuous version of the POD but rather with a discrete setting. Therefore, in the following section we will modify the continuous version of the POD such that $[0, T]$ is being replaced by a finite set of time steps $0 = t_1 < t_2 < \cdots < t_n = T$ and the infinite dimensional function space V is being replaced by the finite dimensional subspace V_h.

4.3 Discrete version of POD

In the first step of the derivation of the discrete version of the proper orthogonal decomposition, let us replace the infinite dimensional function space V by the finite dimensional subspace V_h, which is the span of the finite element basis functions $\{\varphi_i^h\}_{i=1}^m$. Then, we can express functions $\psi, \phi \in V_h$ as a linear combination of the finite element basis, i.e.

$$\psi = \sum_{j=1}^{m} \psi_j^h \varphi_j^h, \quad \phi = \sum_{j=1}^{m} \phi_j^h \varphi_j^h.$$

Using this representation of the two functions, $(\psi, \phi)_X$ simplifies to become a weighted inner product of their coefficient vectors $\boldsymbol{\psi}^h = (\psi_1^h, \ldots, \psi_m^h) \in \mathbb{R}^m$ and $\boldsymbol{\phi}^h = (\phi_1^h, \ldots, \phi_m^h) \in \mathbb{R}^m$. In (3.8), we have shown that in the case $X = H$ it holds that

$$(\psi, \phi) = (\boldsymbol{\psi}^h)^T M_h \boldsymbol{\phi}^h,$$

where $M_h \in \mathbb{R}^{m \times m}$ is the mass matrix with entries $(M_h)_{ij} = (\varphi_j^h, \varphi_i^h)$ for $1 \leq i, j \leq m$. Similarly, in the case $X = V$ it holds that

$$(\nabla\psi, \nabla\phi) = (\boldsymbol{\psi}^h)^T S_h \boldsymbol{\phi}^h,$$

where $S_h \in \mathbb{R}^{m \times m}$ is the stiffness matrix with entries $(S_h)_{ij} = (\nabla\varphi_j^h, \nabla\varphi_i^h)$ for $1 \leq i, j \leq m$. In this thesis we will only consider POD-ROMs that are based on the L^2-inner product ($X = H$), but there are some results which demonstrate that the H^1-inner product ($X = V$) is a viable alternative for fluid dynamics problems and yields better results for turbulent flows [38].

In the next step, let us now approximate the integral over the time interval $[0, T]$. In practice, we don't know the solution $u(t)$ for all times $t \in [0, T]$, but only have access to the snapshots $u^i = u(t_i)$ of the solution for the time steps $0 = t_1 < t_2 < \cdots < t_n = T$. Therefore, we approximate the integral from 0 to T by a quadrature formula with quadrature weights $\{\alpha_i\}_{i=1}^n$ and quadrature points $\{t_i\}_{i=1}^n$.

Let $\{\boldsymbol{\eta}^{h,i}\}_{i=1}^n \subset \mathbb{R}^m$ denote the snapshots from the finite element simulations. We again set $\mathcal{V} = \operatorname{span}\{\boldsymbol{\eta}^{h,i} \mid 1 \leq i \leq n\}$ with $d = \dim \mathcal{V} \leq \min(m, n)$. Due to the previous derivations, the discrete version of the optimization problem $(\hat{\mathbf{P}}_C^r)$ reads

$$\min_{\tilde{\boldsymbol{\psi}}_1, \ldots, \tilde{\boldsymbol{\psi}}_r \in \mathbb{R}^m} \sum_{j=1}^{n} \alpha_j \left\| \boldsymbol{\eta}^{h,j} - \sum_{i=1}^{r} \left(\boldsymbol{\eta}^{h,j}, \tilde{\boldsymbol{\psi}}_i \right)_W \tilde{\boldsymbol{\psi}}_i \right\|_W^2 \quad \text{s.t.} \quad (\tilde{\boldsymbol{\psi}}_i, \tilde{\boldsymbol{\psi}}_j)_W = \delta_{ij} \ \forall 1 \leq i, j \leq r, \qquad (\hat{\mathbf{P}}_D^r)$$

where as mentioned before the weight matrix W depends on the choice of the function space X, namely

$$W = \begin{cases} M_h & \text{for } X = H, \\ S_h & \text{for } X = V. \end{cases}$$

As in the continuous setting, we define the linear, bounded, non-negative and self-adjoint operator $\mathcal{R}^{h,n} : \mathbb{R}^m \to \mathbb{R}^m$ with

$$\mathcal{R}^{h,n}\psi = \sum_{j=1}^{n} \alpha_j \left(\boldsymbol{\eta}^{h,j}, \boldsymbol{\psi}\right)_W \boldsymbol{\eta}^{h,j} \quad \text{for } \boldsymbol{\psi} \in \mathbb{R}^m.$$

The minimization problem $(\hat{\mathbf{P}}_D^r)$ is being solved by the first r eigenvectors $\{\boldsymbol{\psi}_i\}_{i=1}^r \subset \mathbb{R}^m$ of $\mathcal{R}^{h,n}$ with corresponding eigenvalues $\{\lambda_i\}_{i=1}^r$ and the POD approximation error satisfies the formula

$$\sum_{j=1}^{n} \alpha_j \left\| \boldsymbol{\eta}^{h,j} - \sum_{i=1}^{r} \left(\boldsymbol{\eta}^{h,j}, \boldsymbol{\psi}_i\right)_W \boldsymbol{\psi}_i \right\|_W^2 = \sum_{i=r+1}^{n} \lambda_i. \tag{4.2}$$

In the special case where $\alpha_j = 1$ for all the weights α_j, the discrete version of the POD simplifies to the problem statement from Section 3.4. Otherwise we need to slightly modify Algorithm 2. For this purpose, we define the diagonal matrix $D = \operatorname{diag}(\alpha_1, \ldots, \alpha_n) \in \mathbb{R}^{n\times n}$.

Algorithm 3 POD basis for parabolic PDE

Input: Snapshots $\{\boldsymbol{\eta}^{h,i}\}_{i=1}^n \subset \mathbb{R}^m$, threshold $\delta_{\mathcal{E}} \in [0,1]$, symmetric, positive definite matrix $W \in \mathbb{R}^{m\times m}$ and diagonal matrix $D \in \mathbb{R}^{n\times n}$ containing the quadrature weights.

Output: POD basis $\{\boldsymbol{\psi}_i\}_{i=1}^r \subset \mathbb{R}^m$ and eigenvalues $\{\lambda_i\}_{i=1}^r$.

1: Set $Y = [\boldsymbol{\eta}^{h,1}, \ldots, \boldsymbol{\eta}^{h,n}] \in \mathbb{R}^{m\times n}$.
2: **if** $m \approx n$ **then**
3: Determine $\bar{Y} = W^{\frac{1}{2}} Y D^{\frac{1}{2}} \in \mathbb{R}^{m\times n}$.
4: Compute singular value decomposition $[\bar{\Psi}, \Sigma, \bar{\Phi}] = \mathrm{SVD}(\bar{Y})$.
5: Compute $r = \min\left\{ r \in \mathbb{N} \;\middle|\; \mathcal{E}(r) = \sum_{i=1}^r \Sigma_{ii}^2 / \sum_{i=1}^d \Sigma_{ii}^2 \geq \delta_{\mathcal{E}}, \;\; 1 \leq r \leq d \right\}$.
6: Set $\lambda_i = \Sigma_{ii}^2$ and $\boldsymbol{\psi}_i = W^{-\frac{1}{2}} \bar{\Psi}_{\cdot,i} \in \mathbb{R}^m$ for $1 \leq i \leq r$.
7: **else if** $m \ll n$ **then**
8: Determine $\bar{Y} = W^{\frac{1}{2}} Y D^{\frac{1}{2}} \in \mathbb{R}^{m\times n}$.
9: Compute eigenvalue decomposition $[\bar{\Psi}, \Lambda] = \mathrm{Eig}(\bar{Y}\,\bar{Y}^T)$, where $\bar{Y}\,\bar{Y}^T \in \mathbb{R}^{m\times m}$.
10: Compute $r = \min\left\{ r \in \mathbb{N} \;\middle|\; \mathcal{E}(r) = \sum_{i=1}^r \Lambda_{ii} / \sum_{i=1}^d \Lambda_{ii} \geq \delta_{\mathcal{E}}, \;\; 1 \leq r \leq d \right\}$.
11: Set $\lambda_i = \Lambda_{ii}$ and $\boldsymbol{\psi}_i = W^{-\frac{1}{2}} \bar{\Psi}_{\cdot,i} \in \mathbb{R}^m$ for $1 \leq i \leq r$.
12: **else if** $n \ll m$ **then**
13: Compute eigenvalue decomposition $[\bar{\Phi}, \Lambda] = \mathrm{Eig}(D^{\frac{1}{2}} Y^T W Y D^{\frac{1}{2}})$, where $D^{\frac{1}{2}} Y^T W Y D^{\frac{1}{2}} \in \mathbb{R}^{n\times n}$.
14: Compute $r = \min\left\{ r \in \mathbb{N} \;\middle|\; \mathcal{E}(r) = \sum_{i=1}^r \Lambda_{ii} / \sum_{i=1}^d \Lambda_{ii} \geq \delta_{\mathcal{E}}, \;\; 1 \leq r \leq d \right\}$.
15: Set $\lambda_i = \Lambda_{ii}$ and $\boldsymbol{\psi}_i = Y D^{\frac{1}{2}} \bar{\Phi}_{\cdot,i} / \sqrt{\lambda_i} \in \mathbb{R}^m$ for $1 \leq i \leq r$.

4.4 Practical considerations for POD computation

Now that we have built a theoretical framework around the proper orthogonal decomposition, let us discuss how Algorithm 3 is implemented in practice. Often all snapshots cannot be loaded at once into memory to perform a singular value decomposition of a $\mathbb{R}^{m\times n}$ matrix. Therefore, when using the proper orthogonal method in computational fluid mechanics, we instead compute the eigenvalues of the correlation matrix, a $\mathbb{R}^{n\times n}$ matrix. The number of time steps n is usually a few orders smaller than the number of degrees freedom m, i.e. we have $n \ll m$, which makes the method of snapshots (3.7) very efficient. Hence, in reduced order modelling software, we almost always only consider the case $n \ll m$ from Algorithm 3. The only modification is that we compute the correlation matrix

$$C = D^{\frac{1}{2}} Y^T W Y D^{\frac{1}{2}}$$

elementwise and often don't assemble the snapshot matrix Y explicitly. Instead the entries of the correlation matrix are being determined by

$$C_{ij} = C_{ji} = \sqrt{\alpha_i}\sqrt{\alpha_j} (\boldsymbol{\eta}^{h,j})^T M_h \boldsymbol{\eta}^{h,i} = \sqrt{\alpha_i}\sqrt{\alpha_j} \left(u^{h,j}, u^{h,i}\right)_{L^2(\Omega)} \quad \text{for } 1 \leq i, j \leq n.$$

The eigenvectors $\bar{\boldsymbol{\phi}}_1, \dots, \bar{\boldsymbol{\phi}}_n$ and eigenvalues $\lambda_1, \dots, \lambda_n$ of the correlation matrix, as well as the size r of the POD basis are being computed as described in Algorithm 3. Finally, the POD modes can be calculated by

$$\boldsymbol{\psi}_i = \frac{1}{\sqrt{\lambda_i}} \sum_{j=1}^{n} \sqrt{\alpha_j} \left(\bar{\phi}_i\right)_j \boldsymbol{\eta}^{h,j} \quad \text{for } 1 \leq i \leq r. \tag{4.3}$$

In case that the number of snapshots n gets too large, even the computation of all eigenvalues and eigenvectors might become infeasible. Then it is also sufficient to compute the first r eigenvectors and eigenvalues of the correlation matrix with an iterative method. In Algorithm 3, we only needed the eigenvectors $\lambda_{r+1}, \dots \lambda_n$ to evaluate $\mathcal{E}(r)$ and with this find the optimal POD basis size r. However, we can instead use the POD error representation (4.2), which for the special case $r = 0$ reads

$$\sum_{j=1}^{n} \alpha_j \|\boldsymbol{\eta}^{h,j}\|_W^2 = \sum_{i=1}^{n} \lambda_i.$$

Using this formula, the energy ratio can be calculated by

$$\mathcal{E}(r) = \frac{\sum_{i=1}^{r} \lambda_i}{\sum_{i=1}^{n} \lambda_i} = \frac{\sum_{i=1}^{r} \lambda_i}{\sum_{j=1}^{n} \alpha_j \|\boldsymbol{\eta}^{h,j}\|_W^2}, \tag{4.4}$$

which doesn't require the explicit knowledge of the remaining eigenvalues $\lambda_{r+1}, \dots \lambda_n$ anymore.

In equation (4.3), it can be observed that the POD vectors are just linear combinations of the snapshots. Therefore, the POD vectors will by construction satisfy homogeneous boundary conditions, if they are being satisfied by the snapshots. Additionally, they will automatically fulfill any incompressibility condition that the snapshots possess. This is not true anymore when dealing with inhomogeneous boundary conditions. If the inhomogeneous Dirichlet boundary conditions don't depend on the time, then the inhomogeneous Dirichlet boundary conditions can be subtracted from the snapshots before computing the POD basis. Then by construction the POD solution will satisfy the correct boundary conditions. In fluid mechanics, this is often done by a centering approach, where one first computes the mean $\bar{\boldsymbol{\eta}}^h = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{\eta}^{h,i}$ and the mean flow $\bar{\boldsymbol{u}}(\boldsymbol{x}) = \sum_{i=1}^{m} \bar{\eta}_i^h \boldsymbol{\varphi}_i^h(\boldsymbol{x})$, such that one can use the ansatz

$$\boldsymbol{u}^r(t, \boldsymbol{x}) = \bar{\boldsymbol{u}}(\boldsymbol{x}) + \sum_{i=1}^{r} \eta_i^r(t) \boldsymbol{\psi}_i(\boldsymbol{x})$$

for reduced order modelling. However, in the case of the heat equation we have homogeneous Dirichlet boundary conditions. Therefore, we instead use the non-centered ansatz

$$u^r(t, \boldsymbol{x}) = \sum_{i=1}^{r} \eta_i^r(t) \psi_i(\boldsymbol{x}). \tag{4.5}$$

4.5 Reduced order modelling

In the first step, we computed the FEM solution of a time discretized version of the heat equation (4.1). In the second step, we calculated the POD modes $\{\psi_i\}_{i=1}^r$ from the snapshots of the finite element simulation. Now we want to use the POD basis and ansatz (4.5) to create a reduced order model that is computationally much easier to solve than its FEM counterpart. Following the same steps as in Section 4.1, the POD-ROM ansatz (4.5) yields the following spatially discretized version of the heat equation.

Formulation 4.5.1 (POD-ROM spatial discretization of heat equation)**.**

$$M_r \dot{\boldsymbol{\eta}}^r(t) + S_r \boldsymbol{\eta}^r(t) = \boldsymbol{f}^r(t) \quad \forall t \in (0, T) \tag{4.6a}$$

$$M_r \boldsymbol{\eta}^r(0) = \boldsymbol{\eta}_0^r \tag{4.6b}$$

where

$$\begin{aligned}
(M_r)_{ij} &:= (\psi_j, \psi_i) && \text{for } 1 \le i,j \le r,\\
(S_r)_{ij} &:= (\nabla\psi_j, \nabla\psi_i) && \text{for } 1 \le i,j \le r,\\
(f^r(t))_i &:= (f(t), \psi_i) && \text{for } 1 \le i \le r,\\
(\eta_0^r)_i &:= (u_0, \psi_i) && \text{for } 1 \le i \le r.
\end{aligned}$$

To solve Formulation 4.5.1, we now only need to discretize the time derivative, which can be can done with e.g. the One-Step-θ scheme. We then have a fully discretized POD-ROM problem.

Formulation 4.5.2 (POD-ROM discretization of heat equation)**.** *Find* $\{\boldsymbol{\eta}^{r,i}\}_{i=1}^n \subset \mathbb{R}^r$ *such that*

$$(M_r + k_i\theta S_r)\,\boldsymbol{\eta}^{r,i} = (M_r - k_i(1-\theta)S_r)\,\boldsymbol{\eta}^{r,i-1} + k_i\theta\boldsymbol{f}^{r,i} + k_i(1-\theta)\boldsymbol{f}^{r,i-1} \quad \text{for } 2 \le i \le n, \tag{4.7}$$

where $k_i := t_i - t_{i-1}$ *denotes the time step size and we have the same initial conditions as in Formulation 4.5.1.*

We have thus reduced the dimension of the ordinary differential equation that we need to solve from an n dimensional problem (4.1) to an r dimensional problem (4.6) with the help of the proper orthogonal decomposition. In the process, we needed to compute the POD basis and one might ask whether this additional computational effort can be justified by a faster computation time of (4.7) in comparison to traditional finite element simulations. The answer to this question depends on whether the dynamics of the differential equation can be reduced to a lower dimensional manifold, i.e. is $r \ll m$? This question needs to be answered for each problem separately, but it is straightforward to check whether POD-ROM is reasonable for a given PDE. One simply has to compute a few snapshots from a direct numerical simulation of the full order system. With the proper orthogonal decomposition, one then needs to compute the singular values of the snapshot matrix and verify that they decay rapidly. If this is the case, then the dynamics of the underlying PDE can be approximated accurately by only a few modes. But the fast decay of the singular values might be violated, e.g. in the case of the one-dimensional linear transport equation [45], and in that case reduced order modelling is not useful. In practice, if the singular values decay rapidly, we already get good POD-ROM results for $10 \le r \le 100$ [45].

Finally, we should also mention how the reduced matrices M_r and S_r and the reduced vectors $\{\boldsymbol{f}^{r,i}\}_{i=1}^n$ can be computed efficiently. Since the main goal of reduced order modelling is computational efficiency, we use an offline/online splitting. In the offline stage, we compute the POD basis and all time-independent components of the linear equation system (4.7). In the online stage, all components that depend on the time are being computed and we solve the linear equation system. Note that in the online stage one should avoid computations which are of full order, i.e. of order m, since in reduced order modelling we try to transform our computations into the the lower dimensional space of dimension $r \ll m$. In the offline stage, we first store all POD basis vectors in the matrix $\Psi = [\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_r] \in \mathbb{R}^{m\times r}$ and then compute the reduced matrices by

$$\begin{aligned}
M_r &= \Psi^T M_h \Psi,\\
S_r &= \Psi^T S_h \Psi.
\end{aligned}$$

The only thing that is not straightforward is how we should deal with the reduced vectors $\{\boldsymbol{f}^{r,i}\}_{i=1}^n$. Since the function f depends on time, we most likely need to determine these reduced vectors in every step of the online stage, when we solve (4.7). If $f(t,\boldsymbol{x})$ is seperable, i.e.

$$f(t,\boldsymbol{x}) = \gamma(t)g(\boldsymbol{x}) \quad \text{or} \quad f(t,\boldsymbol{x}) = \sum_{j=1}^{l} \gamma_j(t)g_j(\boldsymbol{x}),$$

we can compute the L^2-scalar products (g, φ_i^h) for $1 \le i \le m$ or (g_j, φ_i^h) for $1 \le i \le m$, $1 \le j \le l$ in the offline stage and in the online stage simply restrict them to the POD space with the help of Ψ

and multiply with $\gamma(t_i)$ [26]. In the general case where $f(t,x)$ is not separable, the reduced vectors $\{\boldsymbol{f}^{r,i}\}_{i=1}^n$ need to be assembled in the online stage and are given by

$$\boldsymbol{f}^{r,i} = \Psi^T \boldsymbol{f}^{h,i},$$

where

$$\boldsymbol{f}^{h,i} := \begin{pmatrix} (f(t_i), \varphi_1^h) \\ \vdots \\ (f(t_i), \varphi_m^h) \end{pmatrix} \in \mathbb{R}^m.$$

4.6 Numerical results

For the numerical experiments of reduced order modelling of the heat equation, we will consider the testcase from step 26 of the deal.II tutorials [20]. We solve the heat equation (Formulation 4.1.1) on the L-shaped domain $\Omega = (-1,1)^2 \setminus (0,1)^2$ with homogeneous Dirichlet boundary conditions and zero initial conditions. We define the right hand side to be

$$f(t,\boldsymbol{x}) := \begin{cases} \chi_{(0.5,1)\times(-0.5,0)}(\boldsymbol{x}) & \text{for } \frac{t \bmod 0.2}{0.2} \in [0, 0.2], \\ \chi_{(-0.5,0)\times(0.5,1)}(\boldsymbol{x}) & \text{for } \frac{t \bmod 0.2}{0.2} \in [0.5, 0.7], \\ 0 & \text{else,} \end{cases}$$

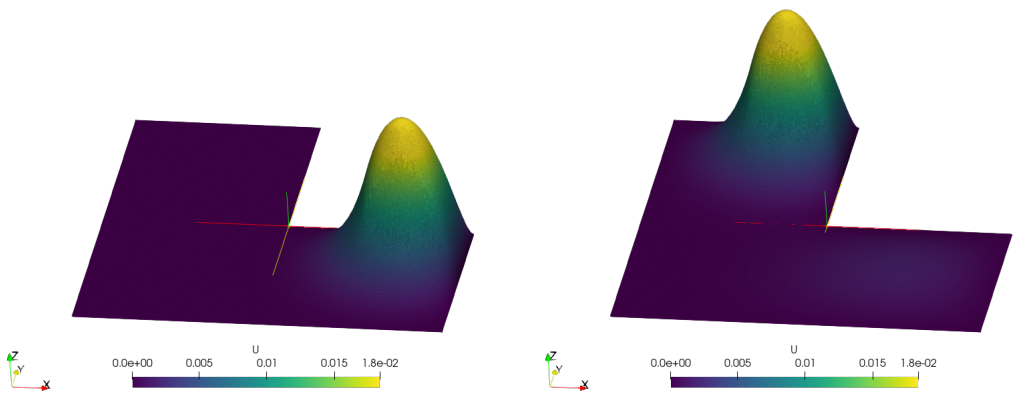
where χ_A denotes the indicator function of the set $A \subset \Omega$, i.e.

$$\chi_A(\boldsymbol{x}) = \begin{cases} 1 & \text{for } \boldsymbol{x} \in A, \\ 0 & \text{for } \boldsymbol{x} \in \Omega \setminus A. \end{cases}$$

From a physical point of view, this problem statement corresponds to square shaped heat sources, which are located in two corners of the L-shape. One then alternates to turn on each heat source. After heating, the temperature then diffuses again. We have zero temperature (0 K) at the beginning of the simulation and at the boundary of the L-shape.

4.6.1 FEM results

For the finite element simulations, the L-shape has been globally refined 7 times, yielding 49,665 degrees of freedom when using linear finite elements. The final time was 0.5 and the time step size was 0.002, giving us a total of 251 snapshots. We chose $\theta = \frac{1}{2}$, which gave us the Crank-Nicholson scheme for the discretization of the time derivative. Each time step has been solved with the conjugate gradient (CG) method with symmetric successive overrelaxation (SSOR) preconditioner. The following computations have been done on an AMD Ryzen 7 2700X with 16 GB RAM.



(a) Snapshot with first peak (t = 0.038) (b) Snapshot with second peak (t = 0.138)

Figure 4.1: Snapshots of the FEM solution of the heat equation

4.6.2 POD resuts

Next, the proper orthogonal decomposition has been applied to the correlation matrix that has been built from the snapshots of the FEM simulation. The total energy of the problem, i.e. the sum of the eigenvalues of the correlation matrix, is $1.19493303075 \cdot 10^{-5}$. Now, let us take a look at the largest eigenvalues of the correlation matrix to assess whether they decay quickly. If this is the case, proper orthogonal decomposition based reduced order modelling is computationally efficient.

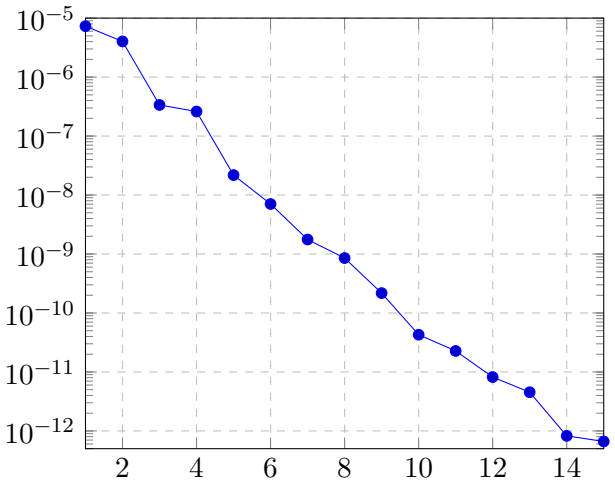


Figure 4.2: First 15 largest eigenvalues of the correlation matrix of the heat equation

In Figure 4.2, we can observe when looking at the 15 largest eigenvalues that the eigenvalues of the correlation matrix decay rapidly, e.g. there are more than 6 orders of magnitude between the largest and the 15th largest eigenvalue. Therefore, it makes sense to use the POD for reduced order modelling. We will now analyze the first 10 largest eigenvalues, their partial energies, which are the sum of the first r eigenvalues, and their energy ratios (see (4.4)), where

$$\text{energy ratio} = \frac{\text{partial energy}}{\text{total energy}}.$$

POD size	i.th eigenvalue	Partial energy	Energy ratio
1	$7.292 \cdot 10^{-6}$	$7.292 \cdot 10^{-6}$	61.02 %
2	$4.028 \cdot 10^{-6}$	$1.132 \cdot 10^{-5}$	94.73 %
3	$3.368 \cdot 10^{-7}$	$1.166 \cdot 10^{-5}$	97.55 %
4	$2.607 \cdot 10^{-7}$	$1.192 \cdot 10^{-5}$	99.73 %
5	$2.179 \cdot 10^{-8}$	$1.1939 \cdot 10^{-5}$	99.92 %
6	$7.049 \cdot 10^{-9}$	$1.1946 \cdot 10^{-5}$	99.98 %
7	$1.762 \cdot 10^{-9}$	$1.1948 \cdot 10^{-5}$	99.990 %
8	$8.543 \cdot 10^{-10}$	$1.19490 \cdot 10^{-5}$	99.998 %
9	$2.174 \cdot 10^{-10}$	$1.194925 \cdot 10^{-5}$	99.9993 %
10	$4.281 \cdot 10^{-11}$	$1.194929 \cdot 10^{-5}$	99.9997 %

Figure 4.3: Comparison of the first 10 largest eigenvalues, their partial energies and their energy ratios for the heat equation

In Figure 4.3, we see that the partial energies are a good approximation to the total energy, even for less than 10 POD vectors, and that the energy ratios converge quickly to 100 %. Remember that in our simulations, we used a finite element basis of size 49,665. With the proper orthogonal decomposition, we discovered that we can reduce the size of the basis to less than 10 basis functions and still preserve

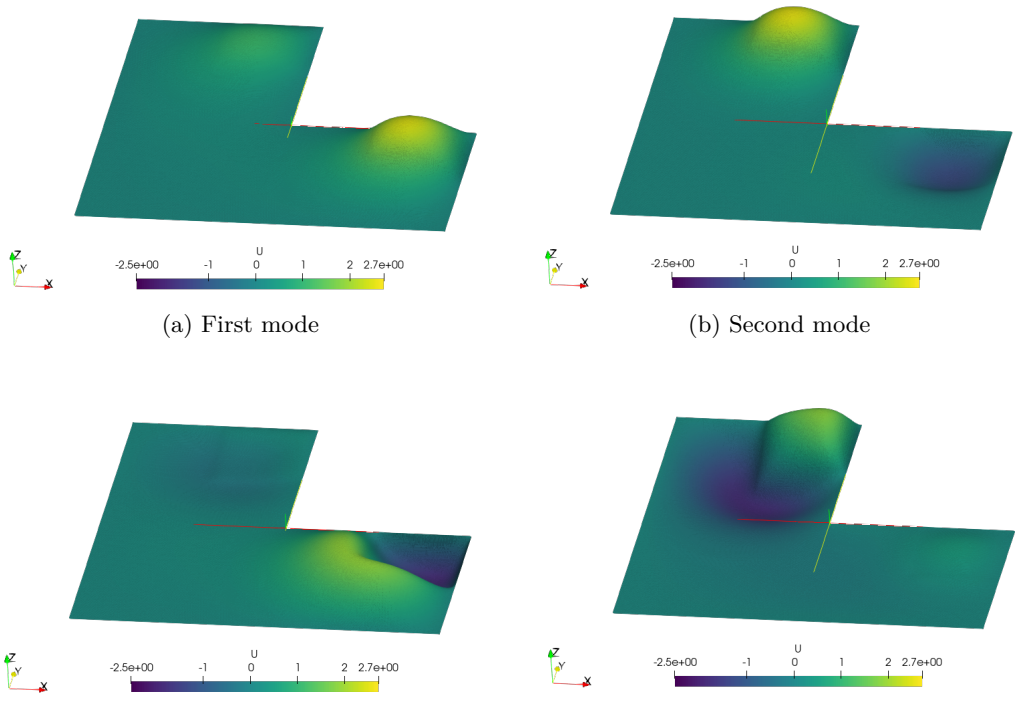
the majority of the energy.

What is a good choice for the POD basis size? One could either choose a fixed size of POD basis or what is usually done in practice, one could increase the size of the POD basis until the energy ratio is bigger than some threshold $\delta_{\mathcal{E}}$, e.g. $\delta_{\mathcal{E}} = 99\ \%$. The latter approach has also been used in Algorithm 3. We will now investigate how the size of the POD basis depends on the choice of the threshold $\delta_{\mathcal{E}}$.

$\delta_{\mathcal{E}}$	POD size	Energy ratio
90 %	2	94.73 %
95 %	3	97.55 %
99 %	4	99.73 %
99.9 %	5	99.92 %
99.99 %	7	99.990 %
99.999 %	9	99.9993 %

Figure 4.4: Size of the POD basis dependent on the threshold $\delta_{\mathcal{E}}$ for the energy ratio

In Figure 4.4, we see that depending on the choice for the threshold $\delta_{\mathcal{E}}$ for the energy ratio, only very few POD modes are needed to reach the desired energy ratio, e.g. for $\delta_{\mathcal{E}} = 90\ \%$ we need 2 basis vectors and for $\delta_{\mathcal{E}} = 99\ \%$ we need 4 basis vectors. Let us therefore take a look at the first four POD modes.



(a) First mode (b) Second mode

(c) Third mode (d) Fourth mode

Figure 4.5: First four POD modes of the heat equation

From the finite element simulations we know that the solution to our heat equation problem has two peaks which alternately rise and dissipate. Therefore, we would expect to only have two POD modes, which each represent one of the peaks. However, the first two POD modes capture one of the peaks and additionally have a small contribution of the other peak. Nevertheless, these two basis vectors already capture around 95 % of the energy ratio. Therefore, the remaining POD vectors don't play a big role anymore. This can be seen at the example of the third and fourth POD modes, which are only

nonzero near the peaks from the finite element solution, but don't carry a lot of information about the dynamics of the system anymore.

4.6.3 POD-ROM results

Finally, we need to compare the efficiency of the finite element method with proper orthogonal decomposition based reduced order modelling. The direct numerical solver (FEM) took 13.459 seconds for the computations. We will now compare this to the computation times of the POD-ROM method for different sizes of the POD basis. Each time step has been solved with a direct solver.

POD size	Absolute time	Relative time	Error
1	1.092 s	8.11 %	$4.66 \cdot 10^{-6}$
2	1.082 s	8.04 %	$6.29 \cdot 10^{-7}$
3	1.109 s	8.24 %	$2.92 \cdot 10^{-7}$
4	1.122 s	8.34 %	$3.18 \cdot 10^{-8}$
5	1.162 s	8.63 %	$9.96 \cdot 10^{-9}$
6	1.149 s	8.54 %	$2.91 \cdot 10^{-9}$
7	1.155 s	8.58 %	$1.15 \cdot 10^{-9}$
8	1.175 s	8.73 %	$2.98 \cdot 10^{-10}$
9	1.189 s	8.83 %	$8.02 \cdot 10^{-11}$
10	1.201 s	8.92 %	$3.74 \cdot 10^{-11}$

Figure 4.6: Comparison of FEM simulation times with computation time of POD-ROM depending on the size of the POD basis

In Figure 4.6, the absolute time denotes the total computation time for determining the POD basis vectors and using them for reduced order modelling. Relative time is the quotient of the absolute time of the POD-ROM method and the computation time for the finite element simulation. The error between the finite element snapshots and the snapshots from the reduced order model has been derived from $(\hat{\mathbf{P}}_D^r)$ and is given by

$$\sum_{j=1}^{n} \alpha_j \left\| \boldsymbol{\eta}^{h,j} - \sum_{i=1}^{r} \eta_i^{r,j} \boldsymbol{\psi}_i \right\|_W^2 = \sum_{i=r+1}^{n} \lambda_i,$$

where $\eta_i^{r,j} := \eta_i^r(t_j)$. Since we are using the trapezoidal rule for the quadrature of the time integral and the time step size k is constant, the quadrature weights α_j read

$$\alpha_j = \begin{cases} \frac{1}{k} & \text{for } 1 < j < n, \\ \frac{1}{2k} & \text{for } j \in \{1, n\}. \end{cases}$$

Overall, we can see in Figure 4.6 that a bigger POD basis leads to a slight increase in computation time and that the error between the finite element solution and its proper orthogonal decomposition based approximation decays rapidly. In our simulations the POD-ROM method took only between 8.04 % and 8.92 % of the computation time of the finite element simulation. This means that in our numerical experiments reduced order modelling is on average 11.77 times faster then the direct numerical simulation of the heat equation with the finite element method.

Chapter 5

The FEM for the Navier-Stokes equations

In this chapter, we will investigate the finite elemtent simulations of the Navier-Stokes equations. We start by introducing the strong formulation of the Navier-Stokes equations. Next, we describe the Navier-Stokes 2D-2 benchmark from Schäfer/Turek [61], its problem statement and quantities of interest. Afterwards, we discuss the finite element spaces, which can be used for the fluid velocity and pressure. Therein, we point out that for these function spaces the inf-sup condition needs to be satisfied. Multiplying the Navier-Stokes equations with test functions, integrating by parts over Ω and applying the boundary conditions, we then arrive at the nonlinear weak formulation $A(\boldsymbol{U})(\boldsymbol{\Psi}) = F(\boldsymbol{\Psi})$. We then continue by deriving the Newton method for this nonlinear problem. Finally, we conclude with a discussion of the numerical results from the finite element computations. A classical numerical analysis book on the topic of the Navier-Stokes equations is [27]. The interested reader can find more information on the finite element analysis of the Navier-Stokes equations in [58] and [71]. An implementation in the finite element software deal.II [2] with a similar problem statement has been presented in [13].

5.1 The Navier-Stokes equations

Starting from physical conservation laws, namely the conservation of mass and the conservation of momentum, Reynolds' transport theorem demonstrates that incompressible, viscous fluid flow with constant density and temperature can be modeled by the Navier-Stokes equations

$$\begin{aligned} \rho \partial_t \boldsymbol{v} - \nabla \cdot \boldsymbol{\sigma} + \rho(\boldsymbol{v} \cdot \nabla)\boldsymbol{v} &= 0, \\ \nabla \cdot \boldsymbol{v} &= 0. \end{aligned}$$

From conservation of angular momentum, it follows that the Cauchy stress tensor of an incompressible Newtonian fluid

$$\boldsymbol{\sigma} = -pI + \mu(\nabla \boldsymbol{v} + \nabla \boldsymbol{v}^T)$$

needs to be symmetric. Plugging the definition of the stress tensor into the Navier-Stokes equations leads to the following formulation:

Formulation 5.1.1 (Navier-Stokes equations)**.** *Find the vector-valued velocity* $\boldsymbol{v} : (0, T) \times \Omega \to \mathbb{R}^d$ *with* $d \in \{2, 3\}$ *and the scalar-valued pressure* $p : (0, T) \times \Omega \to \mathbb{R}$ *such that*

$$\rho \partial_t \boldsymbol{v} + \nabla p - \mu \nabla \cdot (\nabla \boldsymbol{v} + \nabla \boldsymbol{v}^T) + \rho(\boldsymbol{v} \cdot \nabla)\boldsymbol{v} = 0, \tag{5.1a}$$

$$\nabla \cdot \boldsymbol{v} = 0. \tag{5.1b}$$

Here, ρ denotes the fluid density, ν is the kinematic viscosity and $\mu = \rho\nu$ defines the dynamic viscosity.

5.1.1 Reynolds number

The Reynolds number is a dimensionless number, which characterizes the type of fluid flow.

Definition 5.1.2 (Reynolds number)**.** *The Reynolds number is defined as*

$$Re := \frac{LV}{\nu} = \frac{\rho LV}{\mu},$$

where L is a characteristic length and V is a characteristic velocity.

For Reynolds numbers between 1 and 10^5 the fluid flow is laminar [58] and for large Reynolds numbers the flow becomes turbulent (chaotic).

5.2 Problem setup of the Navier-Stokes benchmark

For our numerical simulations, we consider the test case 2D-2 from the Schäfer/Turek Navier-Stokes benchmarks [61]. The 2D-2 benchmark problem describes the two dimensional laminar flow around a circular cylinder, where the fluid flow is nonstationary but when it is fully developed becomes periodic. In the following figure, the domain Ω and the different boundary components Γ_{in}, Γ_{out}, Γ_{wall} and Γ_{circle} are depicted.

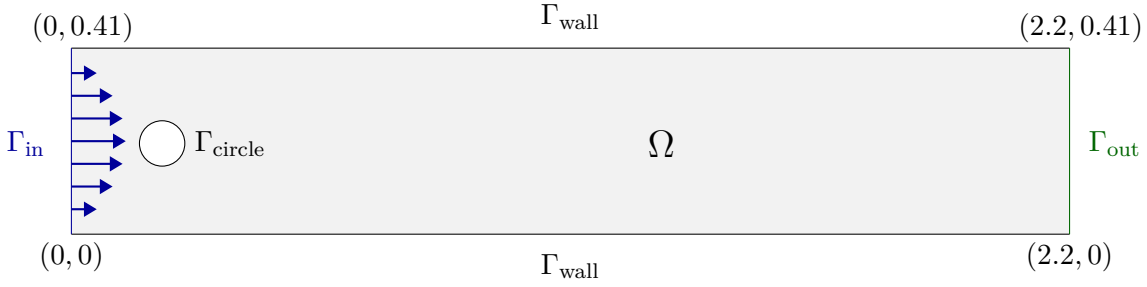


Figure 5.1: Domain of the Navier-Stokes benchmark problem

The parameters of the benchmark configuation are listed in the next table.

parameter	symbol	value
kinematic viscosity	ν	10^{-3} m^2/s
fluid density	ρ	1 kg/m^3
height of pipe	h	0.41 m
diameter of circle	d	0.1 m

Figure 5.2: Parameters of the Navier-Stokes benchmark problem

The domain is defined as $\Omega := (0, 2.2) \times (0, 0.41) \setminus B_r(0.2, 0.2)$ with $r = 0.05$, where $B_r(x, y)$ denotes the ball around (x, y) with radius r. On the boundary $\partial\Omega$ we prescribe Dirichlet and Neumann boundary conditions. We apply the no-slip condition, i.e. homogeneous Dirichlet boundary conditions $\boldsymbol{v} = \boldsymbol{0}$ on $\Gamma_{\text{wall}} \cup \Gamma_{\text{circle}}$, where $\Gamma_{\text{wall}} = (0, 2.2) \times \{0\} \cup (0, 2.2) \times \{0.41\}$ and $\Gamma_{\text{circle}} = \partial B_r(0.2, 0.2)$. On the boundary $\Gamma_{\text{in}} = \{0\} \times (0, 0.41)$, we enforce a steady parabolic inflow profile, which is a type of inhomogeneous Dirichlet boundary conditions, i.e. $\boldsymbol{v} = \boldsymbol{g}$ on Γ_{in}. For the 2D-2 benchmark the inflow parabola is given by

$$\boldsymbol{g}(0, y) = \begin{pmatrix} \frac{4v_{\max} y(h-y)}{h^2} \\ 0 \end{pmatrix},$$

with maximum velocity $v_{\max} = 1.5$ m/s. The ouflow boundary $\Gamma_{\text{out}} = \{2.2\} \times (0, 0.41)$ has Neumann boundary conditions. It has been discussed in [31] and was also shown in [71] that the do-nothing condition $\boldsymbol{\sigma} \cdot \boldsymbol{n} = \boldsymbol{0}$ on Γ_{out} leads to an incorrect outflow profile of the pressure. Hence, instead we work with the modified do-nothing condition $\mu \partial_n \boldsymbol{v} - p\boldsymbol{n} = \boldsymbol{0}$ on Γ_{out}.

In our benchmark problem, the characteristic length is the diameter of the circle, i.e. $L := d = 0.1$ m, and the characteristic velocity is the mean velocity in x-direction on the inflow boundary, i.e. $V(t) = \bar{v}_x(t) = 1$ m/s, since

$$\begin{aligned}\bar{v}_x(t) &:= \frac{1}{h} \int_0^h v_x \left(t, \begin{pmatrix} 0 \\ y \end{pmatrix} \right) \, \mathrm{d}y = \frac{1}{h} \int_0^h g_x(0, y) \, \mathrm{d}y \\ &= \frac{1}{h} \int_0^h \frac{6 \text{ m/s} \cdot y(h-y)}{h^2} \, \mathrm{d}y = 1 \text{ m/s}.\end{aligned}$$

Overall, we have thus shown that the Reynolds number of our problem statement is

$$Re = \frac{LV}{\nu} = \frac{d\bar{v}_x}{\nu} = \frac{0.1 \text{ m} \cdot 1 \text{ m/s}}{10^{-3} \text{ m}^2/\text{s}} = 100.$$

For the benchmark problem, the quantities of interest are the drag and the lift coefficients (C_D and C_L), as well as the Strouhal number (St) and the pressure difference (Δp). The drag force F_D and the lift force F_L can be computed by a line integral over the boundary of the circle, namely

$$\begin{pmatrix} F_D \\ F_L \end{pmatrix} = \int_{\Gamma_{\text{circle}}} \boldsymbol{\sigma} \cdot \boldsymbol{n} \, \mathrm{d}s.$$

Then, the drag and lift coefficients are given by

$$C_D = \frac{2F_D}{\rho d \bar{v}_x^2} \quad \text{and} \quad C_L = \frac{2F_L}{\rho d \bar{v}_x^2}.$$

The Strouhal number St is defined as

$$St = \frac{df}{\bar{v}_x^2},$$

where f is the frequency of the flow. For $t \geq 25$ s the fluid flow is fully developed and time-periodic with period $1/f \approx 0.33$ s. The period can be determined programmatically by measuring the time between two consecutive maxima of the lift coefficient. The pressure difference is the difference between the pressure on the front and back of the circle, i.e.

$$\Delta p := p(0.15, 0.2) - p(0.25, 0.2).$$

5.3 Finite element spaces

The Navier-Stokes equations are an example of a mixed system. In contrast to the heat equation, we now don't need to find an appropriate function space for only one solution variable, but we need to construct function spaces for the fluid velocity $\boldsymbol{v}$ and for the fluid pressure p. Similar to the treatment of the heat equation, after integration by parts the highest derivative of the fluid velocity will be its gradient. We thus only require $\boldsymbol{v} \in [H^1(\Omega)]^2$. For the fluid pressure, we require even lower regularity. After integration by parts, it suffices that the pressure is square integrable, i.e. $p \in L_2(\Omega)$. We need to account for Dirichlet boundary conditions in the function space, hence we choose

$$V^v := \{\boldsymbol{v} \in [H^1(\Omega)]^2 \mid \boldsymbol{v} = \boldsymbol{0} \text{ on } \Gamma_{\text{in}} \cup \Gamma_{\text{wall}} \cup \Gamma_{\text{circle}}\}.$$

We prescribe homogeneous Neumann boundary conditions for the pressure on $\partial\Omega$ and thus work with

$$V^p := L^2(\Omega),$$

since the do-nothing condition implicitly normalizes the pressure to give a unique solution [31, 58].

For the uniqueness of the pressure solution, the inf-sup or Ladyzhenskaya-Babuska-Brezzi (LBB) condition

$$\inf_{q\in V^p}\sup_{\varphi\in V^v}\frac{(q,\nabla\cdot\varphi)}{\|q\|_{L^2(\Omega)}\|\varphi\|_{H^1(\Omega)}}\geq\gamma>0$$

needs to be satisfied. This stability estimate still needs to be fulfilled when V^p and V^v are being replaced by the finite element spaces $V_h^p\subset V^p$ and $V_h^v\subset V^v$. In practice this means that the velocity space needs to be sufficiently larger than the pressure space, e.g. one cannot use linear finite elements for the velocity and the pressure, since this leads to artificial oscillations in the pressure [71].

A popular finite element discretization of mixed elements that satisfies the inf-sup condition is the Taylor-Hood element $Q_2\times Q_1$. This has been proven e.g. in [27][Corollary 4.1]. With the Taylor-Hood element, the fluid velocity is being approximated by quadratic finite elements and the fluid pressure is being solved with linear finite elements.

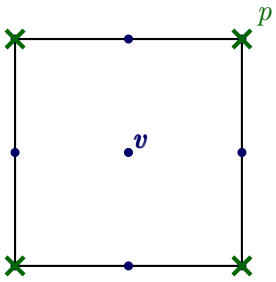


Figure 5.3: Degrees of freedom of the Taylor-Hood element

In Figure 5.3, the degrees of freedom of the quadratic finite elements for the velocity have been marked with blue dots and the degrees of freedom of the linear finite elements for the pressure have been marked with green crosses.

5.4 Weak formulation

To derive the weak formulation of the Navier-Stokes equations, we multiply equations (5.1) by test functions $\boldsymbol{\psi}^v\in V^v$ and $\psi^p\in V^p$ respectively, and integrate over the domain Ω.

Find $\{\boldsymbol{v},p\}\in\{\boldsymbol{g}+V^v\}\times V^p$ such that the initial conditions are satisfied
and for almost all time steps $t\in(0,T)$ it holds that

$$\rho(\partial_t\boldsymbol{v},\boldsymbol{\psi}^v)+\left(\nabla p-\mu\nabla\cdot(\nabla\boldsymbol{v}+\nabla\boldsymbol{v}^T),\boldsymbol{\psi}^v\right)+\rho\left((\boldsymbol{v}\cdot\nabla)\boldsymbol{v},\boldsymbol{\psi}^v\right)=0 \tag{5.2}$$

$$(\nabla\cdot\boldsymbol{v},\boldsymbol{\psi}^p)=0 \tag{5.3}$$

Next, using integration by parts and the fact that ψ^v satisfies homogeneous Dirichlet boundary conditions on $\partial\Omega\setminus\Gamma_{\text{out}}$, we get that

$$\rho(\partial_t\boldsymbol{v},\boldsymbol{\psi}^v)+\mu\left(\nabla\boldsymbol{v}+\nabla\boldsymbol{v}^T,\nabla\boldsymbol{\psi}^v\right)-(p,\nabla\cdot\boldsymbol{\psi}^v)-\int_{\Gamma_{\text{out}}}\left[\mu(\nabla\boldsymbol{v}+\nabla\boldsymbol{v}^T)\cdot\boldsymbol{n}-p\boldsymbol{n}\right]\cdot\boldsymbol{\psi}^v\,\mathrm{d}s+\rho\left((\boldsymbol{v}\cdot\nabla)\boldsymbol{v},\boldsymbol{\psi}^v\right)=0.$$

Using the 'modified' do-nothing condition [31]

$$\mu\nabla\boldsymbol{v}\cdot\boldsymbol{n}-p\boldsymbol{n}=0,$$

the weak form simplifies to

$$\rho(\partial_t\boldsymbol{v},\boldsymbol{\psi}^v)+\mu\left(\nabla\boldsymbol{v}+\nabla\boldsymbol{v}^T,\nabla\boldsymbol{\psi}^v\right)-(p,\nabla\cdot\boldsymbol{\psi}^v)-\int_{\Gamma_{\text{out}}}\left[\mu\nabla\boldsymbol{v}^T\cdot\boldsymbol{n}\right]\cdot\boldsymbol{\psi}^v\,\mathrm{d}s+\rho\left((\boldsymbol{v}\cdot\nabla)\boldsymbol{v},\boldsymbol{\psi}^v\right)=0. \tag{5.4}$$

Discretizing the time derivative with the One-Step-θ scheme and adding equations (5.3) and (5.4), we get the semi-linear form

$$\begin{aligned} A(\boldsymbol{U})(\boldsymbol{\Psi}) := \rho(\boldsymbol{v}, \boldsymbol{\psi}^v) + k\theta\mu \left(\nabla \boldsymbol{v} + \nabla \boldsymbol{v}^T, \nabla \boldsymbol{\psi}^v\right) - k\theta(p, \nabla \cdot \boldsymbol{\psi}^v) \\ - k\theta \int_{\Gamma_{\text{out}}} \left[\mu \nabla \boldsymbol{v}^T \cdot \boldsymbol{n}\right] \cdot \boldsymbol{\psi}^v \, \mathrm{d}s + k\theta\rho \left((\boldsymbol{v} \cdot \nabla)\boldsymbol{v}, \boldsymbol{\psi}^v\right) + k(\nabla \cdot \boldsymbol{v}, \psi^p) \end{aligned} \tag{5.5}$$

and the right hand side functional

$$\begin{aligned} F(\boldsymbol{\Psi}) := \rho(\boldsymbol{v}^{n-1}, \boldsymbol{\psi}^v) - k(1-\theta)\mu \left(\nabla \boldsymbol{v}^{n-1} + \nabla \boldsymbol{v}^{n-1\,T}, \nabla \boldsymbol{\psi}^v\right) + k(1-\theta)(p^{n-1}, \nabla \cdot \boldsymbol{\psi}^v) \\ + k(1-\theta) \int_{\Gamma_{\text{out}}} \left[\mu \nabla \boldsymbol{v}^{n-1\,T} \cdot \boldsymbol{n}\right] \cdot \boldsymbol{\psi}^v \, \mathrm{d}s - k(1-\theta)\rho \left((\boldsymbol{v}^{n-1} \cdot \nabla)\boldsymbol{v}^{n-1}, \boldsymbol{\psi}^v\right), \end{aligned} \tag{5.6}$$

where $\boldsymbol{U} := \{\boldsymbol{v}, p\} \in \{\boldsymbol{g} + V^v\} \times V^p$ and $\boldsymbol{\Psi} := \{\boldsymbol{\psi}^v, \psi^p\} \in V^v \times V^p$. Here, we used the notation $\boldsymbol{v} := \boldsymbol{v}^n$ and $p := p^n$ for solutions that need to be computed in the current time step and $\boldsymbol{v}^{n-1}$ and p^{n-1} for solutions from the last time step, which already have been computed. In the implementation the pressure term is treated fully implicitly, i.e. we have the term $-k(p, \nabla \cdot \boldsymbol{\psi}^v)$ in the semi-linear form and $(p^{n-1}, \nabla \cdot \boldsymbol{\psi}^v)$ doesn't arise in the right hand side anymore.

5.5 Solution of the weak formulation with Newton's method

To solve the nonlinear problem $A(\boldsymbol{U})(\boldsymbol{\Psi}) = F(\boldsymbol{\Psi})$, we can first reformulate it as a root finding problem, where we are trying to find a root of the residual $F(\boldsymbol{\Psi}) - A(\boldsymbol{U})(\boldsymbol{\Psi})$. From introductions to numerical analysis [60] and finite element analysis courses [72], it is known that such variational root finding problems can be solved by Newton's method. Therein, in a step of the residual-based Newton's method one needs to solve the problem: Find $\delta\boldsymbol{U} := \{\delta\boldsymbol{v}, \delta p\} \in V^v \times V^p$ such that

$$A'(\boldsymbol{U})(\delta\boldsymbol{U}, \boldsymbol{\Psi}) = -A(\boldsymbol{U})(\boldsymbol{\Psi}) + F(\boldsymbol{\Psi}) \quad \forall \boldsymbol{\Psi} \in V^v \times V^p.$$

Hence, we now need to compute the Fréchet derivative of the semi-linear form $A(\boldsymbol{U})(\boldsymbol{\Psi})$ with respect to $\boldsymbol{U}$ in the direction of $\delta\boldsymbol{U}$. Note that for a linear functional $f : V^v \to V^v$ the Frechét derivative is given by

$$f'(\boldsymbol{v})(\delta\boldsymbol{v}) = f(\delta\boldsymbol{v}),$$

due to the linearity of f. Hence, it only remains to compute the Frechét derivative of

$$f : V^v \to V^v, \quad \boldsymbol{v} \mapsto (\boldsymbol{v} \cdot \nabla)\boldsymbol{v}.$$

We have that

$$\begin{aligned} f(\boldsymbol{v} + h\,\delta\boldsymbol{v}) &= \left((\boldsymbol{v} + h\,\delta\boldsymbol{v}) \cdot \nabla\right)(\boldsymbol{v} + h\,\delta\boldsymbol{v}) \\ &= (\boldsymbol{v} \cdot \nabla)\boldsymbol{v} + h\left[(\delta\boldsymbol{v} \cdot \nabla)\boldsymbol{v} + (\boldsymbol{v} \cdot \nabla)\delta\boldsymbol{v}\right] + h^2(\delta\boldsymbol{v} \cdot \nabla)\delta\boldsymbol{v}. \end{aligned}$$

We observe that the first term is $f(\boldsymbol{v})$ and the last term is quadratic in h. Hence, the Frechét derivative reads

$$f'(\boldsymbol{v})(\delta\boldsymbol{v}) = (\delta\boldsymbol{v} \cdot \nabla)\boldsymbol{v} + (\boldsymbol{v} \cdot \nabla)\delta\boldsymbol{v}.$$

The derivative of the semi-linear form is thus given by

$$\begin{aligned} A'(\boldsymbol{U})(\delta\boldsymbol{U}, \boldsymbol{\Psi}) := \rho(\delta\boldsymbol{v}, \boldsymbol{\psi}^v) + k\theta\mu \left(\nabla\delta\boldsymbol{v} + \nabla\delta\boldsymbol{v}^T, \nabla\boldsymbol{\psi}^v\right) - k\theta(\delta p, \nabla \cdot \boldsymbol{\psi}^v) \\ -k\theta \int_{\Gamma_{\text{out}}} \left[\mu \nabla\delta\boldsymbol{v}^T \cdot \boldsymbol{n}\right] \cdot \boldsymbol{\psi}^v \, \mathrm{d}s + k\theta\rho \left((\delta\boldsymbol{v} \cdot \nabla)\boldsymbol{v} + (\boldsymbol{v} \cdot \nabla)\delta\boldsymbol{v}, \boldsymbol{\psi}^v\right) + k(\nabla \cdot \delta\boldsymbol{v}, \psi^p). \end{aligned}$$

Finally, we can formulate the full algorithm of Newton's method, which we will use for the numerical experiments.

Algorithm 4 Residual-based Newton method with backtracking line search

Input: Initial guess $\boldsymbol{U}_h^{n,0} \in \{\boldsymbol{g} + V_h^v\} \times V_h^p$
Output: FEM solution $\boldsymbol{U}_h^n \in \{\boldsymbol{g} + V_h^v\} \times V_h^p$ at time t_n.

1: **for** j = 0, 1, 2, ... **do**
2: Find $\delta \boldsymbol{U}_h^n \in V_h^v \times V_h^p$ such that

$$A'(\boldsymbol{U}_h^{n,j})(\delta \boldsymbol{U}_h^n, \boldsymbol{\Psi}_h) = -A(\boldsymbol{U}_h^{n,j})(\boldsymbol{\Psi}_h) + F(\boldsymbol{\Psi}_h) \quad \forall \boldsymbol{\Psi}_h \in V_h^v \times V_h^p,$$
$$\boldsymbol{U}_h^{n,j+1} = \boldsymbol{U}_h^{n,j} + \delta \boldsymbol{U}_h^n.$$

3: Check convergence criterion

$$\|R(\boldsymbol{U}_h^{n,j+1})\|_\infty \leq \|R(\boldsymbol{U}_h^{n,j})\|_\infty, \tag{5.7}$$

where

$$\|R(\boldsymbol{U}_h^{n,j})\|_\infty := \max_{i \in [1, \dim X_h]} \left| F(\boldsymbol{\Psi}_i) - A(\boldsymbol{U}_h^{n,j})(\boldsymbol{\Psi}_i) \right|.$$

4: **if** (5.7) is violated **then**
5: Compute for $k = 1, \ldots, k_{\max}$ a new solution ▷ line search

$$\boldsymbol{U}_h^{n,j+1} = \boldsymbol{U}_h^{n,j} + \lambda^k \, \delta \boldsymbol{U}_h^n$$

until (5.7) is fulfilled.
6: **if** (5.7) is fulfilled **then**
7: Check the stopping criterion

$$\|R(\boldsymbol{U}_h^{n,j+1})\|_\infty \leq TOL. \tag{5.8}$$

8: **if** (5.8) is fulfilled **then**
9: $\boldsymbol{U}_h^n := \boldsymbol{U}_h^{n,j+1}$.
10: **else**
11: $j \to j + 1$ and **go to** 2.

For the numerical tests, we used $k_{\max} = 10$ as the maximum number of line search steps and $\lambda = 0.6$ as the line search damping parameter. We set the maximum number of Newton steps to 10 and the tolerance TOL to 10^{-16}.

Since we are dealing with a time-dependent problem, the initial guess can be chosen as the solution from the last time step, i.e.

$$U_h^{n,0} := U_h^{n-1}.$$

We then need to prescribe the non-homogeneous Dirichlet conditions representing the parabolic inflow on this initial guess, as well as the homogeneous Dirichlet conditions on Γ_{wall}. However, the boundary conditions for the update $\delta \boldsymbol{U}_h^n$ become homogeneous, i.e. we enforce $\delta \boldsymbol{U}_h^n = \boldsymbol{0}$ on $\Gamma_{\text{in}} \cup \Gamma_{\text{wall}} \cup \Gamma_{\text{circle}}$.

We observe that in Algorithm 4 we need to compute the directional derivative $A'(\boldsymbol{U}_h^{n,j})(\delta \boldsymbol{U}_h^n, \boldsymbol{\Psi}_h)$ in each step of the Newton method. This is computationally expensive, since we need to assemble the Jacobian matrix in every step. Instead, in our program we used the approximation

$$A'(\boldsymbol{U}_h^{n,j+1})(\cdot, \boldsymbol{\Psi}_h) \approx A'(\boldsymbol{U}_h^{n,j})(\cdot, \boldsymbol{\Psi}_h),$$

when the reduction rate

$$\theta_j := \frac{\|R(\boldsymbol{U}_h^{n,j+1})\|_\infty}{\|R(\boldsymbol{U}_h^{n,j})\|_\infty}$$

was sufficiently small. We used these so called simplified-Newton steps, when $\theta_j \leq \theta_{\max} := 0.1$.

5.6 Numerical results

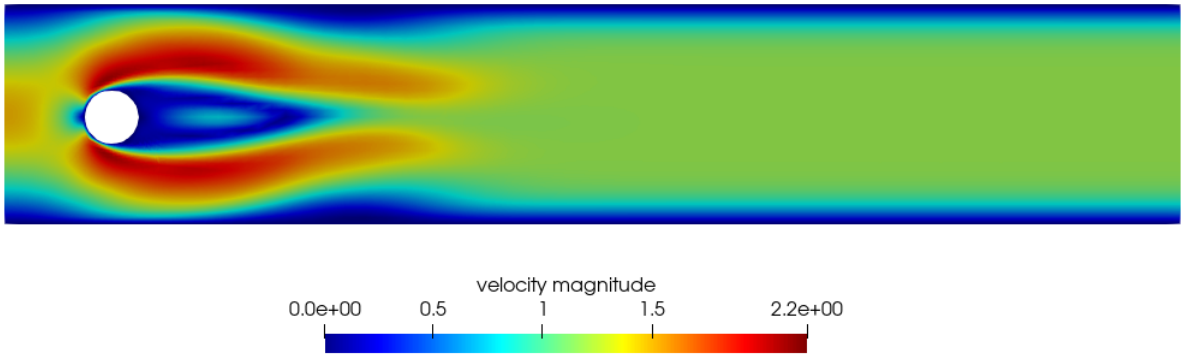
For the finite element simulations, the input mesh has been refined globally 3 times. This resulted in 2,560 cells, 21,024 degrees of freedom for the fluid velocity and 2,696 degrees of freedom for the pressure. In total, there have been 23,720 degrees of freedom in the FEM simulation. Until the initial solution of the flow has developed, we used a coarser time step size of 0.05 s from $t = 0$ s to $t = 3.5$ s. Thereafter, the finer time step size 0.005 s has been used until the final time $t = 35$ s. This gave us a total of 6,372 snapshots. We chose $\theta = \frac{1}{2}$, which gave us the Crank-Nicholson scheme for the discretization of the time derivative. Each time step has been solved with the sparse direct solver UMFPACK. The following computations have been done on an AMD Ryzen 7 2700X with 16 GB RAM.

After the FEM simulation, we computed the quantities of interest: the Strouhal number (St), the maximal drag coefficient (max C_D), the maximal lift coefficient (max C_L) and the maximal pressure difference (Δp). It can be seen in Figure 5.4 that our results mostly coincide with the experimental results from the benchmark paper [61] and only the maximal drag coefficient is slightly lower than the results from the literature.

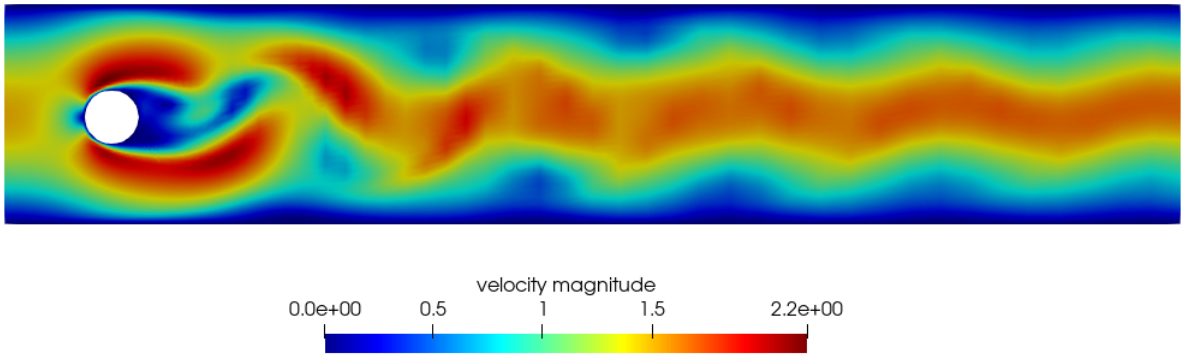
Quantity of interest	Our results	Benchmark results [61]
St	0.3018	[0.2950, 0.3050]
max C_D	3.1755	[3.2200, 3.2400]
max C_L	0.9899	[0.9900, 1.0100]
Δp	2.5079	[2.4600, 2.5000]

Figure 5.4: Quantities of interest results of the FEM solution of the 2D-2 benchmark

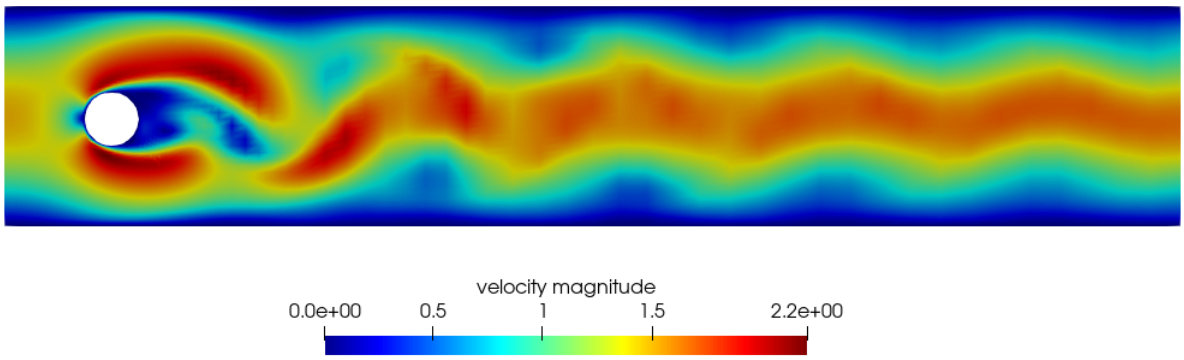
In Figure 5.5, Figure 5.6 and Figure 5.7, we summarize our observations of the finite element simulation of the Navier-Stokes 2D-2 benchmark. In the first few seconds of the simulation, the fluid flow is still not fully developed, which can be seen at the snapshots of the velocity magnitude and the pressure at $t = 0.75$ s. After a few seconds, we can observe vortex shedding on the right of the circle. After 25 seconds of simulations, the fluid flow is fully developed and periodic in time. One period of the flow is the length of time between two consecutive maxima of the lift coefficient. For our analysis, we take a closer look at the time interval [26.395 s, 26.725 s], whose end points are exactly the times at which the lift coefficient is maximal. In Figure 5.7, we plot the lift coefficient, the drag coefficient and the pressure difference for this time interval. Furthermore, in Figure 5.5 and Figure 5.6, we also show snapshots of the FEM solution, when the lift is maximal ($t = 26.395$ s) and minimal ($t = 26.56$ s) in the time interval of interest.



(a) Initial solution (t = 0.75 s)

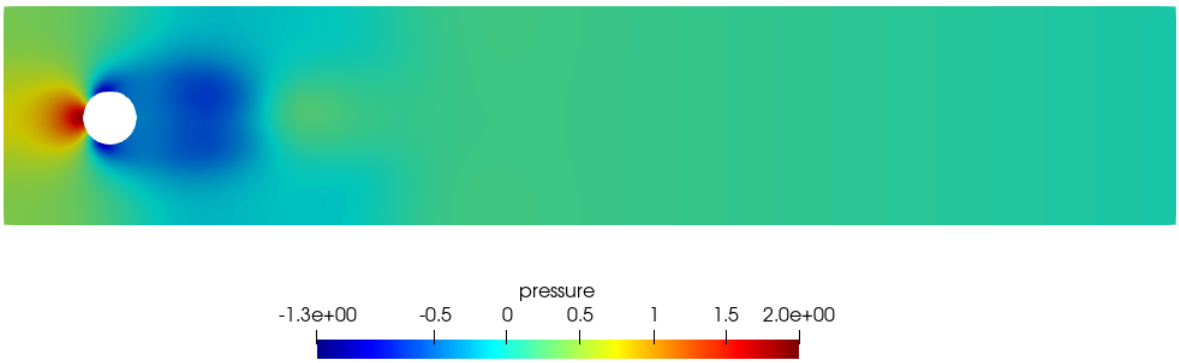


(b) Maximal lift of period of fluid flow (t = 26.395 s)

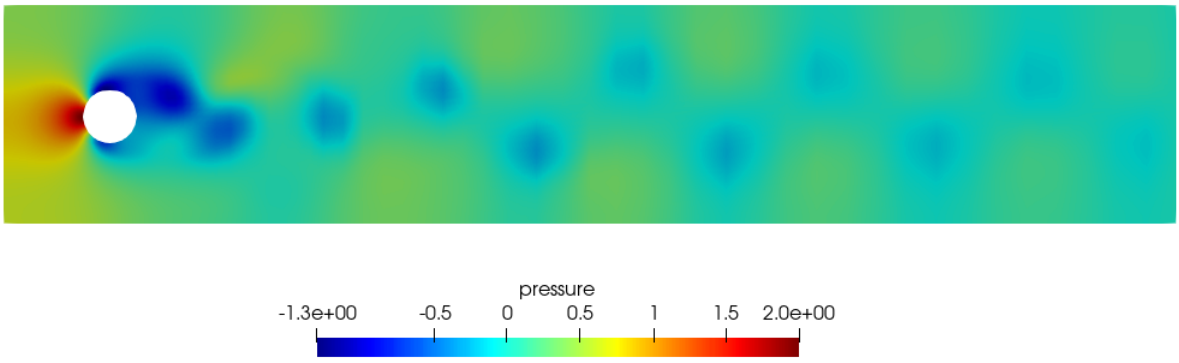


(c) Minimal lift of period of fluid flow (t = 26.56 s)

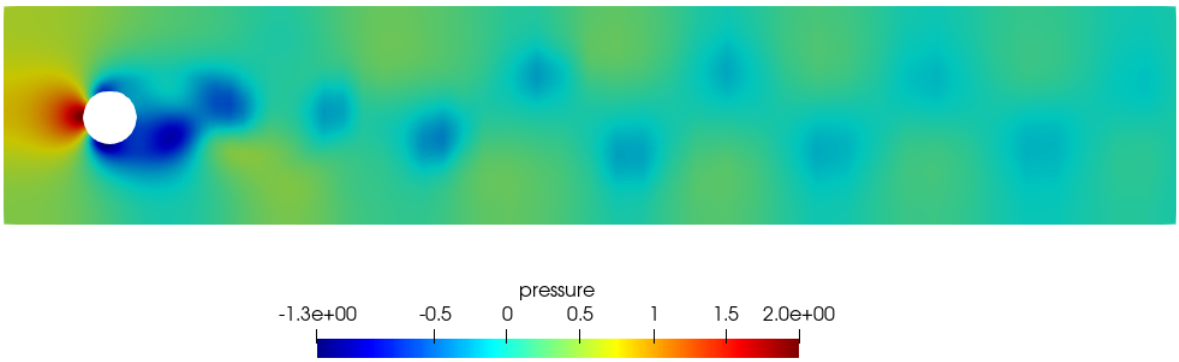
Figure 5.5: Snapshots of the velocity magnitude of the FEM solution of the 2D-2 benchmark



(a) Initial solution (t = 0.75 s)

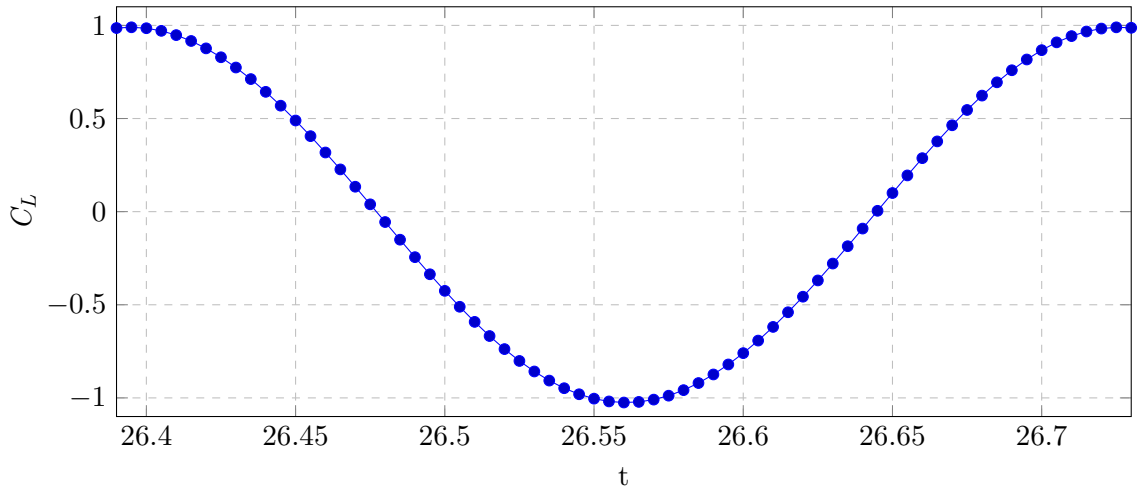


(b) Maximal lift of period of fluid flow (t = 26.395 s)

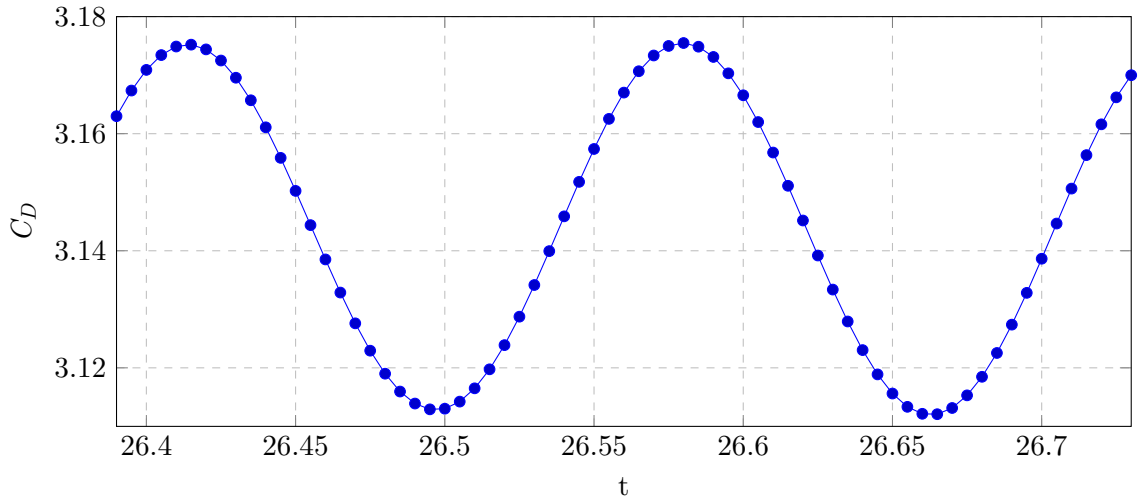


(c) Minimal lift of period of fluid flow (t = 26.56 s)

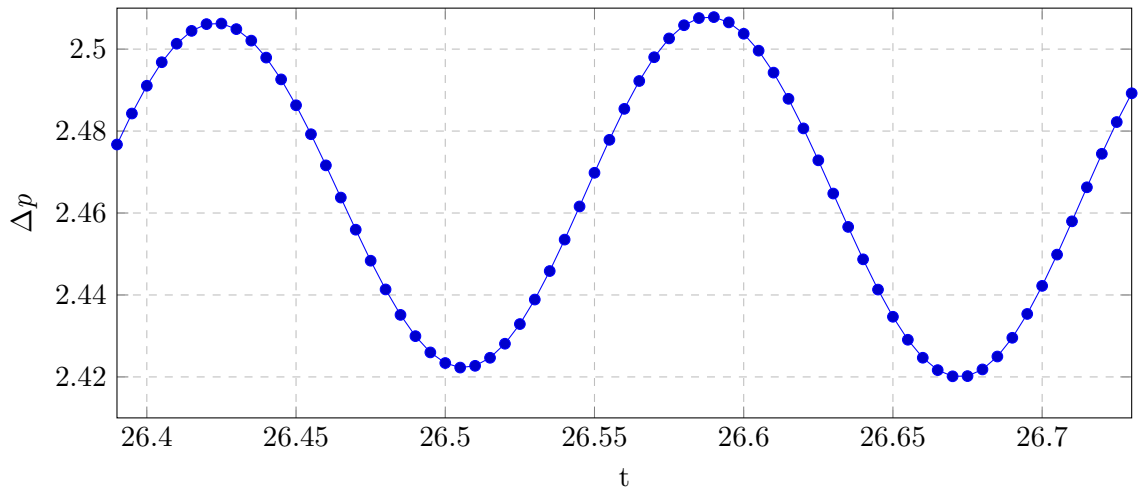
Figure 5.6: Snapshots of the pressure of the FEM solution of the 2D-2 benchmark



(a) Lift coefficient



(b) Drag coefficient



(c) Pressure difference

Figure 5.7: Quantities of interest of the FEM solution of the 2D-2 benchmark during one period of the flow

Chapter 6

The POD for the Navier-Stokes equations

In this chapter, we will investigate the usage of the proper orthogonal decomposition for reduced order modelling of the Navier-Stokes equations. Herefore, we will consider the 2D-2 benchmark from Chapter 5, which simulates the flow around a cylinder in two dimensions. Using the results from the finite element simulations, which have been described in Chapter 5, in the first step we compute the POD basis of the velocity and of the pressure with the method of snapshots from Chapter 4. In the second step, we apply the POD-ROM method to the momentum equation of the Navier-Stokes equations and get a reduced order model for the velocity, wherein the pressure term vanishes due to the incompressibility condition. In the third step, different approaches for the computation of the pressure are being discussed, including a neural network based method, which is novel to the best of the author's knowledge. Finally, we discuss our solution of the numerical experiments and compare it with the results from the literature. The interested reader can find more information on POD-ROM for the Navier-Stokes equations in [14], [26][Chapter 5] and [70][Chapter 6].

6.1 POD of velocity and pressure

Using the finite element method setup from Chapter 5, we computed the 401 snapshots from initial time 25 s to the final time 27 s with time step size 0.005 s via direct numerical simulation. In the following, we will discuss how we separately computed the POD basis vectors for velocity and pressure from these high fidelity snapshots. The velocity and pressure have also been computed with a decoupled approach in [14, 26, 70].

6.1.1 POD of velocity

We will now compute the POD basis vectors for the velocity with the method of snapshots as explained in Algorithm 3. The only novelty compared to the POD computation of the heat equation is that in fluid mechanics one often uses a centering approach.

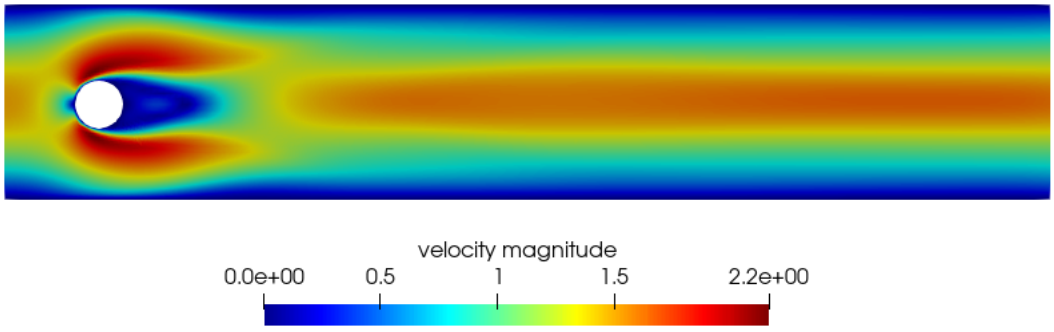


Figure 6.1: Magnitude of the mean velocity of the snapshots of the Navier-Stokes equations

Before using the method of snapshots, we first compute the mean of the velocity snapshots $\bar{\boldsymbol{\eta}}^h = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\eta}^{h,i}$ and the mean flow $\bar{\boldsymbol{v}}(\boldsymbol{x}) = \sum_{i=1}^{m} \bar{\eta}_i^h \boldsymbol{\psi}_i^v(\boldsymbol{x})$, where $\{\boldsymbol{\psi}_i^v\}_{i=1}^m$ are the FEM basis functions of the velocity. Then, we can use the ansatz

$$\boldsymbol{v}^r(t,\boldsymbol{x}) = \bar{\boldsymbol{v}}(\boldsymbol{x}) + \sum_{i=1}^{r} \eta_i^r(t)\boldsymbol{\phi}_i^v(\boldsymbol{x}) \tag{6.1}$$

for reduced order modelling of the velocity, where $\{\boldsymbol{\phi}_i^v\}_{i=1}^r$ with $r \ll m$ are the POD modes of the velocity. To compute these POD modes of the velocity with the method of snapshots, we first need to center the snapshots with the mean flow, i.e. we first need to subtract the mean $\bar{\boldsymbol{\eta}}^h$ from the snapshots $\boldsymbol{\eta}^{h,i}$. We can then apply the proper orthogonal decomposition to the correlation matrix of the centered snapshots as discussed in Chapter 4. The total energy of the velocity problem, i.e. the sum of the eigenvalues of the correlation matrix, is 0.826184407011. Let us now analyze the largest few eigenvalues of the correlation matrix of the velocity and verify that they decrease rapidly, which motivates the usage of a reduced order model for the velocity.

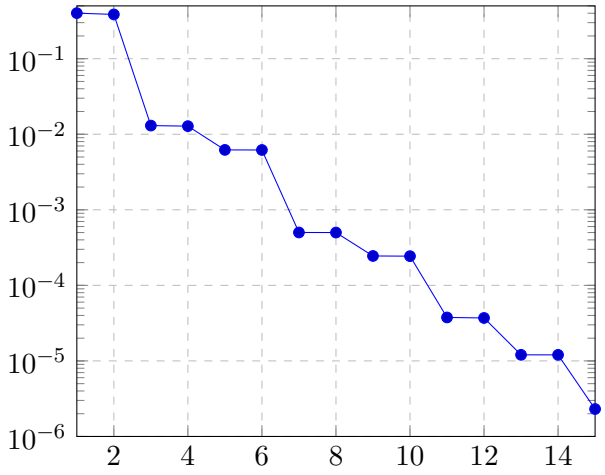


Figure 6.2: First 15 largest eigenvalues of the correlation matrix of the velocity of the Navier-Stokes equations

In Figure 6.2, we observe that the eigenvalues of the correlation matrix of the velocity fall sharply, which validates the POD ansatz (6.1) for truth approximation purposes. We will now look at the first 10 largest eigenvalues, their partial energies, which are the sum of the first r eigenvalues, and their energy ratios (see (4.4)).

POD size	i.th eigenvalue	Partial energy	Energy ratio
1	0.4008560512	0.4008560512	48.52 %
2	0.385506742831	0.786362794031	95.18 %
3	0.0130260162293	0.79938881026	96.76 %
4	0.012805095564	0.812193905824	98.31 %
5	0.00620677419644	0.81840068002	99.06 %
6	0.00618942896932	0.82459010899	99.81 %
7	0.000501109577591	0.825091218567	99.87 %
8	0.000499931133157	0.8255911497	99.93 %
9	0.000244898617558	0.825836048318	99.96 %
10	0.000243503640934	0.826079551959	99.987 %

Figure 6.3: Comparison of the first 10 largest eigenvalues, their partial energies and their energy ratios for the velocity of the Navier-Stokes equations

In Figure 6.3, we note that the partial energies are a good approximation to the total energy of the velocity and the energy ratios approach 100 %, but not as quickly as it was the case for the heat equation (see Figure 4.3). This indicates that the Navier-Stokes equation are a more complicated problem than the heat equation and require a few more POD basis vectors for reduced order modelling. In practice, the POD basis size is determined by taking the smallest POD basis size for which the energy ratio is bigger than some threshold $\delta_{\mathcal{E}}$. We will now investigate how the size of the POD basis of the velocity depends on the choice of this threshold $\delta_{\mathcal{E}}$.

$\delta_{\mathcal{E}}$	POD size	Energy ratio
90 %	2	95.18 %
95 %	2	95.18 %
99 %	5	99.06 %
99.9 %	8	99.93 %
99.99 %	11	99.992 %
99.999 %	14	99.9993 %

Figure 6.4: Size of the POD basis dependent on the threshold $\delta_{\mathcal{E}}$ for the energy ratio of the velocity of the Navier-Stokes equations

In Figure 6.4, we see that depending on the choice of the threshold $\delta_{\mathcal{E}}$ for the energy ratio of the velocity, only very few POD modes are needed to reach the desired energy ratio, e.g. for $\delta_{\mathcal{E}} = 90$ % we need 2 basis vectors, for $\delta_{\mathcal{E}} = 99$ % we need 5 basis vectors and for $\delta_{\mathcal{E}} = 99.99$ % we need 11 basis vectors. At the end of this section, we show the magnitude of the first five velocity POD modes of the Navier-Stokes equations. We observe that the magnitudes of the first four velocity POD modes, which are shown in Figure 6.9, visually resemble the POD modes of the Navier-Stokes equations that have been depicted in [14][Fig. 4].

6.1.2 POD of pressure

We will now repeat the steps from the last subsection and compute the POD basis vectors for the pressure with the method of snapshots and using a centering approach. Herefore, we first compute the mean of the pressure snapshots $\bar{\boldsymbol{\alpha}}^h = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\alpha}^{h,i}$ and the mean pressure $\bar{p}(\boldsymbol{x}) = \sum_{i=1}^{m_p} \bar{\alpha}_i^h \psi_i^p(\boldsymbol{x})$, where $\{\psi_i^p\}_{i=1}^{m_p}$ are the FEM basis functions of the pressure. Then, we can use the ansatz

$$p^r(t, \boldsymbol{x}) = \bar{p}(\boldsymbol{x}) + \sum_{i=1}^{r_p} \alpha_i^r(t)\phi_i^p(\boldsymbol{x}) \tag{6.2}$$

for reduced order modelling of the pressure, where $\{\phi_i^p\}_{i=1}^{r_p}$ with $r_p \ll m_p$ are the POD modes of the pressure. The POD basis size of the pressure does not need to be the same as the POD basis size of the velocity, but for our numerical experiments we work with $r = r_p$.

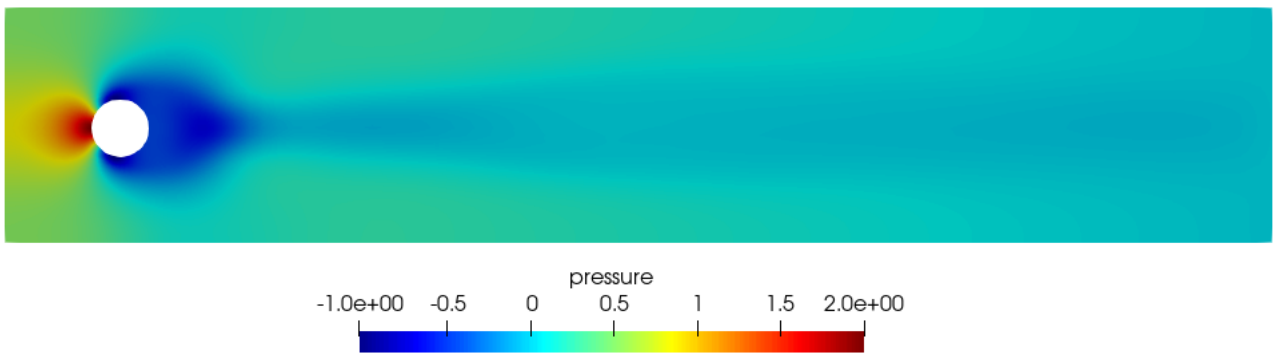


Figure 6.5: Mean pressure of the snapshots of the Navier-Stokes equations

To calculate the POD modes of the pressure, we again first center the snapshots with the mean pressure, i.e. we subtract the mean $\bar{\boldsymbol{\alpha}}^h$ from the snapshots $\boldsymbol{\alpha}^{h,i}$, and then use the proper orthogonal decomposition on the correlation matrix of the centered snapshots. The total energy of the pressure problem, i.e. the sum of the eigenvalues of the correlation matrix, is 0.123471339707. Next, we visualize the largest eigenvalues of the correlation matrix of the pressure and check if they decline quickly.

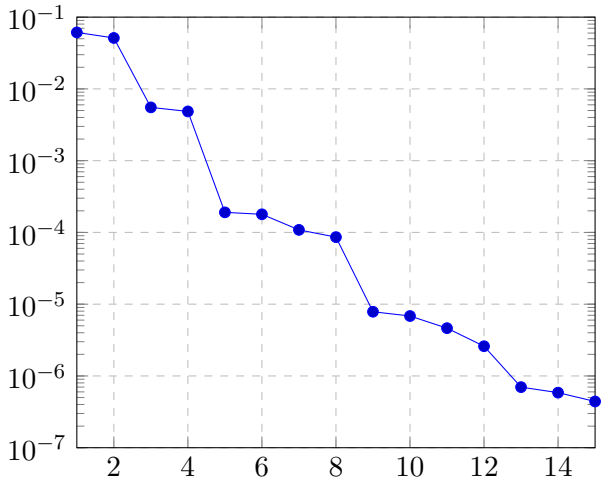


Figure 6.6: First 15 largest eigenvalues of the correlation matrix of the pressure of the Navier-Stokes equations

In Figure 6.6, we notice that the eigenvalues of the correlation matrix of the pressure descend rapidly in value. Hence, the POD ansatz (6.2) should yield good approximations to the pressure from the FEM simulations for even very few POD modes. We continue by analyzing the first 10 largest eigenvalues, their partial energies, which are the sum of the first r eigenvalues, and their energy ratios (see (4.4)).

POD size	i.th eigenvalue	Partial energy	Energy ratio
1	0.0612280781364	0.0612280781364	49.59 %
2	0.0512750051872	0.112503083324	91.12 %
3	0.00552804515282	0.118031128476	95.59 %
4	0.00485226040942	0.122883388886	99.52 %
5	0.000190400390044	0.123073789276	99.68 %
6	0.000178923435037	0.123252712711	99.82 %
7	0.00010849725874	0.12336120997	99.91 %
8	$8.60398321476 \cdot 10^{-5}$	0.123447249802	99.98 %
9	$7.84770203766 \cdot 10^{-6}$	0.123455097504	99.987 %
10	$6.84251904709 \cdot 10^{-6}$	0.123461940023	99.992 %

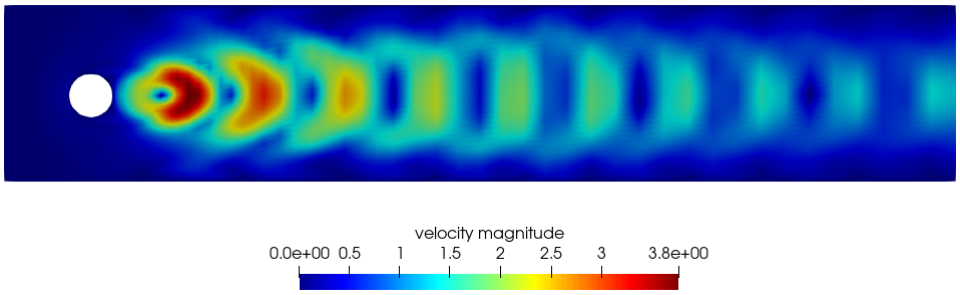
Figure 6.7: Comparison of the first 10 largest eigenvalues, their partial energies and their energy ratios for the pressure of the Navier-Stokes equations

In Figure 6.7, we find that the partial energies are a good approximation to the total energy of the pressure and the energy ratios converge towards 100 %. However, analogous to the observations for the velocity POD basis, this convergence is not as fast as it was the case for the heat equation (see Figure 4.3). We show in the following how the size of the POD basis of the pressure depends on the threshold $\delta_{\mathcal{E}}$, which is the minimal desired energy ratio.

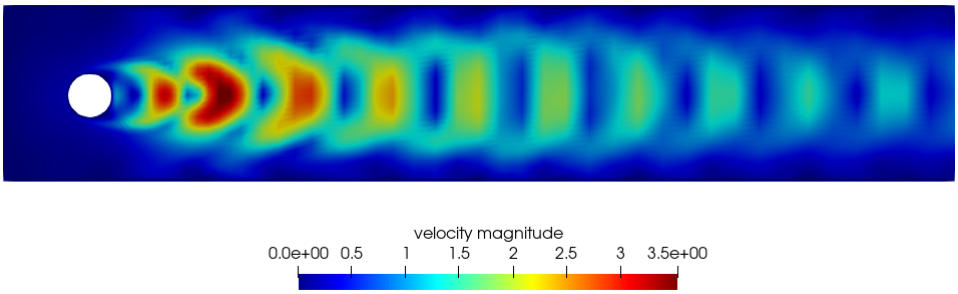
$\delta_{\mathcal{E}}$	POD size	Energy ratio
90 %	2	91.12 %
95 %	3	95.59 %
99 %	4	99.52 %
99.9 %	7	99.91 %
99.99 %	10	99.992 %
99.999 %	14	99.9993 %

Figure 6.8: Size of the POD basis dependent on the threshold $\delta_{\mathcal{E}}$ for the energy ratio of the pressure of the Navier-Stokes equations

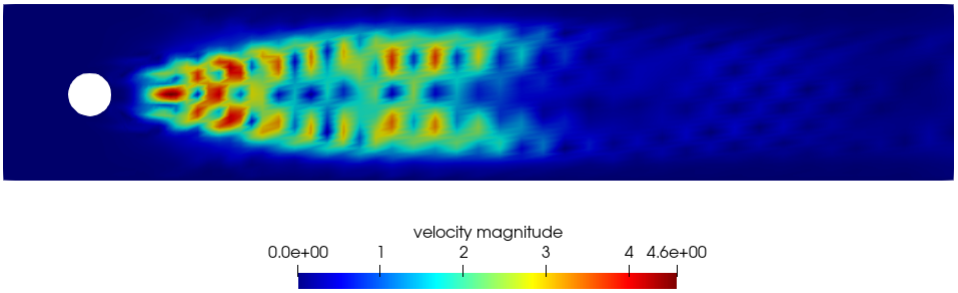
In Figure 6.8, we see that depending on the choice for the threshold $\delta_{\mathcal{E}}$ for the energy ratio, only very few POD modes are needed to reach the desired energy ratio, e.g. for $\delta_{\mathcal{E}} = 90$ % we need 2 basis vectors, for $\delta_{\mathcal{E}} = 99$ % we need 4 basis vectors and for $\delta_{\mathcal{E}} = 99.99$ % we need 10 basis vectors. On the final page of this section, we show the first five pressure POD modes. On closer inspection, we detect that the first four pressure POD modes, which are shown in Figure 6.10, look similar to the POD modes of the Navier-Stokes equations that have been published in [14][Fig. 5]. However, some of our basis vectors differ from the results from the literature by a scaling of -1, which then only changes the sign of the coefficients in the POD basis expansion (6.2).



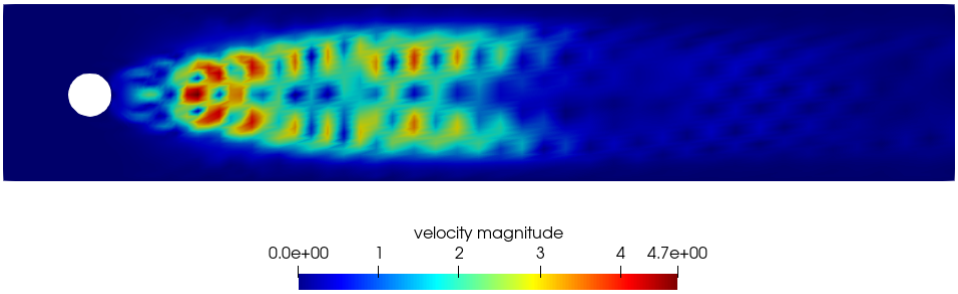
(a) Magnitude of the first mode



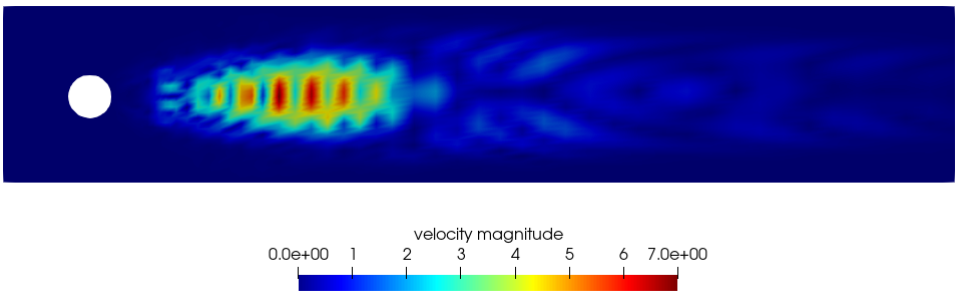
(b) Magnitude of the second mode



(c) Magnitude of the third mode

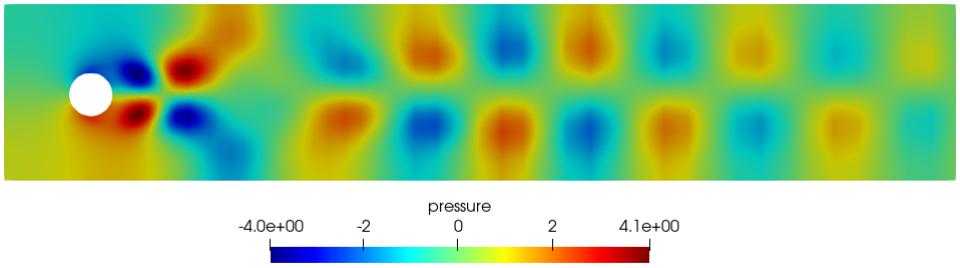


(d) Magnitude of the fourth mode

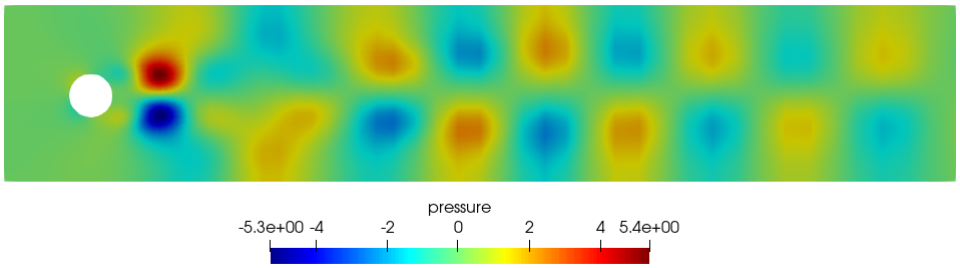


(e) Magnitude of the fifth mode

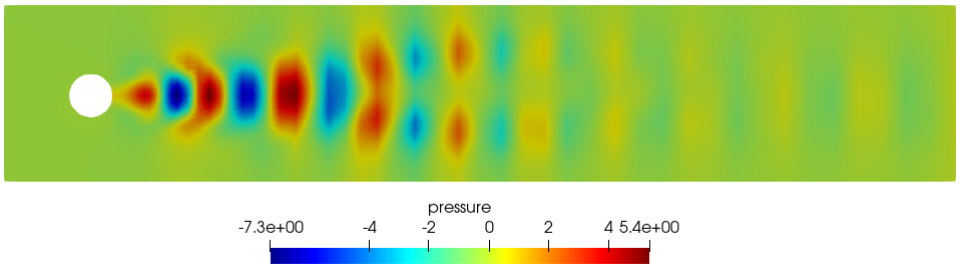
Figure 6.9: Magnitude of the first five velocity POD modes of the Navier-Stokes equations



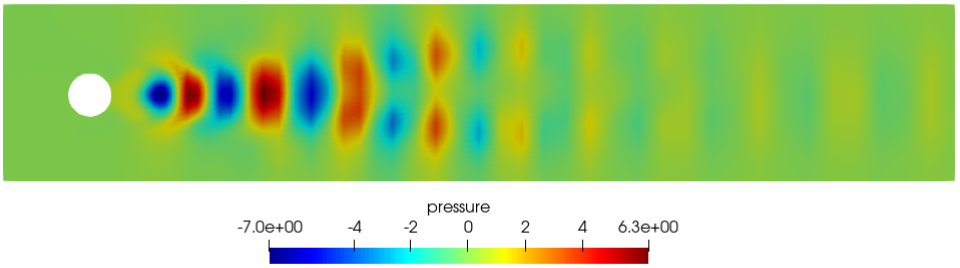
(a) First mode



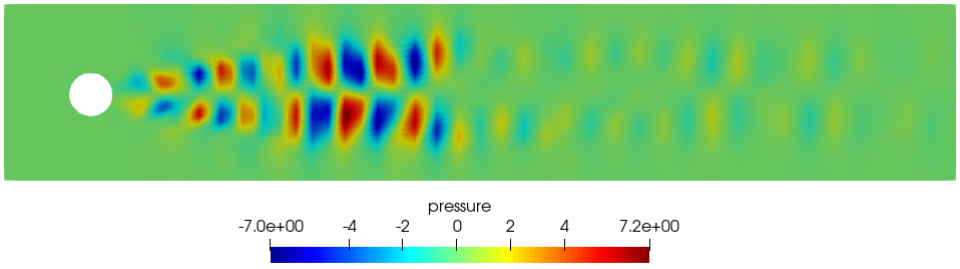
(b) Second mode



(c) Third mode



(d) Fourth mode



(e) Fifth mode

Figure 6.10: First five pressure POD modes of the Navier-Stokes equations

6.2 Velocity POD-ROM

Plugging the POD ansatz (6.1) into the weak formulation of the Navier-Stokes equations and using the velocity POD modes $\{\boldsymbol{\phi}_i^v\}_{i=1}^r$ as test functions yields

$$\rho(\partial_t \boldsymbol{v}^r, \boldsymbol{\phi}_i^v) + \mu(\nabla \boldsymbol{v}^r + \nabla \boldsymbol{v}^{r\,T}, \nabla \boldsymbol{\phi}_i^v) - \mu \int_{\Gamma_{\text{out}}} \left[\nabla \boldsymbol{v}^{r\,T} \cdot \boldsymbol{n}\right] \cdot \boldsymbol{\phi}_i^v \, \mathrm{d}s + \rho((\boldsymbol{v}^r \cdot \nabla)\boldsymbol{v}^r, \boldsymbol{\phi}_i^v) = 0 \quad \forall 1 \leq i \leq r.$$

In the finite element discretization, we also had the terms $(p, \nabla \cdot \boldsymbol{\psi}^v)$ and $(\nabla \cdot \boldsymbol{v}, \psi^p)$ in the weak formulation, but the velocity POD modes are a linear combination of the velocity snapshots $\{\boldsymbol{\eta}^{h,i}\}_{i=1}^n$ from the FEM simulation and it is assumed that they satisfy the incompressibility condition. However, this assumption is only idealized [14][Subsection 3.1]. Hence, the POD vectors $\boldsymbol{\phi}_i^v$ for $1 \leq i \leq r$ by construction also satisfy the incompressiblity condition and we thus have that

$$\begin{aligned} \nabla \cdot \boldsymbol{\phi}_i^v &= 0 \quad \forall 1 \leq i \leq r, \\ \nabla \cdot \boldsymbol{v}^r &= 0. \end{aligned}$$

Therefore, the reduced weak formulation doesn't contain the pressure term and the term for the incompressibility of the fluid. For more difficult problems in fluid mechanics, e.g. for fluid structure interaction problems, we still need to account for the pressure in the reduced order model. Then, the reduced weak form needs to be supplemented by some stabilization terms, since even though the FEM function spaces satisfy the inf-sup condition, this doesn't necessarily hold true anymore for the reduced spaces [54]. To solve the reduced weak form, we again apply the One-Step-θ scheme for time discretization. By defining $\hat{\boldsymbol{v}}(\boldsymbol{x}) := \boldsymbol{v}^r(t^n, \boldsymbol{x}) - \bar{\boldsymbol{v}}(\boldsymbol{x})$, a step of the time discrete formulation with time step size k reads:

Find $\hat{\boldsymbol{v}}(\boldsymbol{x}) = \sum_{i=1}^r \eta_i^{r,n} \boldsymbol{\phi}_i^v(\boldsymbol{x})$ such that

$$a(\hat{\boldsymbol{v}})(\boldsymbol{\phi}_i^v) = l(\boldsymbol{\phi}_i^v) \quad \forall 1 \leq i \leq r, \tag{6.3}$$

where

$$\begin{aligned} a(\hat{\boldsymbol{v}})(\boldsymbol{\phi}) := \rho(\hat{\boldsymbol{v}}, \boldsymbol{\phi}) + k\theta\mu(\nabla \hat{\boldsymbol{v}} + \nabla \hat{\boldsymbol{v}}^T, \nabla \boldsymbol{\phi}) - k\theta\mu \int_{\Gamma_{\text{out}}} \left[\nabla \hat{\boldsymbol{v}}^T \cdot \boldsymbol{n}\right] \cdot \boldsymbol{\phi} \, \mathrm{d}s \\ + k\theta\rho((\hat{\boldsymbol{v}} \cdot \nabla)\hat{\boldsymbol{v}}, \boldsymbol{\phi}) + k\theta\rho((\bar{\boldsymbol{v}} \cdot \nabla)\hat{\boldsymbol{v}}, \boldsymbol{\phi}) + k\theta\rho((\hat{\boldsymbol{v}} \cdot \nabla)\bar{\boldsymbol{v}}, \boldsymbol{\phi}) \end{aligned}$$

and

$$\begin{aligned} l(\boldsymbol{\phi}) := \rho(\boldsymbol{v}^{r,n-1}, \boldsymbol{\phi}) - k(1-\theta)\mu \left(\nabla \boldsymbol{v}^{r,n-1} + \nabla \boldsymbol{v}^{r,n-1\,T}, \nabla \boldsymbol{\phi}\right) \\ + k(1-\theta)\mu \int_{\Gamma_{\text{out}}} \left[\nabla \boldsymbol{v}^{r,n-1\,T} \cdot \boldsymbol{n}\right] \cdot \boldsymbol{\phi} \, \mathrm{d}s \\ - k(1-\theta)\rho((\boldsymbol{v}^{r,n-1} \cdot \nabla)\boldsymbol{v}^{r,n-1}, \boldsymbol{\phi}) - k\theta\rho((\bar{\boldsymbol{v}} \cdot \nabla)\bar{\boldsymbol{v}}, \boldsymbol{\phi}) \\ - \rho(\bar{\boldsymbol{v}}, \boldsymbol{\phi}) - k\theta\mu(\nabla \bar{\boldsymbol{v}} + \nabla \bar{\boldsymbol{v}}^T, \nabla \boldsymbol{\phi}) + k\theta\mu \int_{\Gamma_{\text{out}}} \left[\nabla \bar{\boldsymbol{v}}^T \cdot \boldsymbol{n}\right] \cdot \boldsymbol{\phi} \, \mathrm{d}s. \end{aligned}$$

In this formulation $\boldsymbol{v}^{r,n-1}(\boldsymbol{x}) = \bar{\boldsymbol{v}}(\boldsymbol{x}) + \sum_{i=1}^r \eta_i^{r,n-1} \boldsymbol{\phi}_i^v(\boldsymbol{x})$ denotes the solution of the velocity from the previous time step. We observe that the reduced weak formulation (6.3) is still a nonlinear problem. Hence, we compute the Fréchet derivative of the semi-linear form $a(\hat{\boldsymbol{v}})(\boldsymbol{\phi})$, which reads

$$\begin{aligned} a'(\hat{\boldsymbol{v}})(\delta\hat{\boldsymbol{v}}, \boldsymbol{\phi}) = \rho(\delta\hat{\boldsymbol{v}}, \boldsymbol{\phi}) + k\theta\mu(\nabla\delta\hat{\boldsymbol{v}} + \nabla\delta\hat{\boldsymbol{v}}^T, \nabla\boldsymbol{\phi}) - k\theta\mu \int_{\Gamma_{\text{out}}} \left[\nabla\delta\hat{\boldsymbol{v}}^T \cdot \boldsymbol{n}\right] \cdot \boldsymbol{\phi} \, \mathrm{d}s \\ + k\theta\rho((\delta\hat{\boldsymbol{v}} \cdot \nabla)\hat{\boldsymbol{v}}, \boldsymbol{\phi}) + k\theta\rho((\hat{\boldsymbol{v}} \cdot \nabla)\delta\hat{\boldsymbol{v}}, \boldsymbol{\phi}) \\ + k\theta\rho((\bar{\boldsymbol{v}} \cdot \nabla)\delta\hat{\boldsymbol{v}}, \boldsymbol{\phi}) + k\theta\rho((\delta\hat{\boldsymbol{v}} \cdot \nabla)\bar{\boldsymbol{v}}, \boldsymbol{\phi}). \end{aligned} \tag{6.4}$$

In reduced order modelling, we try to avoid computations which are of order m, which is the number of degrees of freedom in the finite element simulation. Hence, our goal is to assemble as much of the linear equation system as possible in the offline phase of the reduced order modelling and then have a

quick assembly in the online phase of the simulation. For most of the terms of the Fréchet derivative (6.4), we can simply assemble the corresponding finite element matrices, e.g. the mass matrix, and then project it into the space spanned by the POD vectors. Only the linearized terms $((\delta\hat{\boldsymbol{v}}\cdot\nabla)\hat{\boldsymbol{v}},\boldsymbol{\phi})$ and $((\hat{\boldsymbol{v}}\cdot\nabla)\delta\hat{\boldsymbol{v}},\boldsymbol{\phi})$ cannot be assembled in this way. For these terms, we instead compute the rank 3 tensor $C\in\mathbb{R}^{r\times r\times r}$ with entries $C_{ijk}:=((\boldsymbol{\phi}_j^v\cdot\nabla)\boldsymbol{\phi}_k^v,\boldsymbol{\phi}_i^v)$ for $1\leq i,j,k\leq r$.

6.3 Pressure reconstruction

Until now, we have only paid attention to the velocity in our POD-ROM modelling. In many applications, we are interested in the computation of goal functionals, like drag or lift, which require the knowledge of the pressure solution. In [14], the authors analyzed three different methods on reduced order models for the Navier-Stokes equations which include the computation of the pressure. The first two methods use the reduced weak formulation from before to compute the pressure and then proceed to determine the pressure by solving a Poisson equation. The third method is a coupled method for simultaneous computation of the fluid velocity and pressure, by adding stabilization terms to the Navier-Stokes equations. The stablization is necessary to ensure that the POD spaces satisfy the inf-sup condition. In this thesis, we will only consider the first two methods, which have been presented in [14] and in [26][Chapter 5]. Additionally, we also consider a neural network aproach, which learns a mapping between the velocity and pressure POD spaces, and is novel to the best of the author's knowledge.

To reconstruct the pressure, we commence by considering the strong formulation of the Navier-Stokes equations (5.1) and taking the divergence of the momentum equation. This yields

$$-\Delta p=\rho\nabla\cdot[(\boldsymbol{v}\cdot\nabla)\boldsymbol{v}]\quad\text{in }\Omega,$$

since all other terms vanish due to the incompressibility condition $\nabla\cdot\boldsymbol{v}=0$. We can pass to the POD spaces and rewrite the right hand side as

$$-\Delta p^r=\rho\nabla\cdot[(\boldsymbol{v}^r\cdot\nabla)\boldsymbol{v}^r]=\rho\left((\partial_1 v_1^r)^2+2\partial_2 v_1^r\partial_1 v_2^r+(\partial_2 v_2^r)^2\right)\quad\text{in }\Omega. \tag{6.5}$$

In the following, we will discuss two different methods from [14], which try to solve this Poisson equation for the pressure.

6.3.1 Method based on velocity modes

The first method is based on the POD modes of the velocity and was first introduced in [51]. Herein, we plug the ansatz (6.1) into the Poisson problem (6.5) and define $\eta_0^r(t)\equiv 1$ and $\boldsymbol{\phi}_0^v(\boldsymbol{x})=\bar{\boldsymbol{v}}(\boldsymbol{x})$. This yields

$$-\Delta p^r=\rho\sum_{i=0}^{r}\sum_{j=0}^{r}\eta_i^r(t)\eta_j^r(t)\left\{\sum_{k=1}^{2}\sum_{l=1}^{2}\partial_k(\phi_i^v)_l\;\partial_l(\phi_j^v)_k\right\}.$$

Note that here the time dependent coefficients $\eta_i^r(t)$ and $\eta_j^r(t)$ are known from the velocity ROM. Hence, the solution to the reduced pressure equation reads

$$p^r(t,\boldsymbol{x})=\rho\sum_{i=0}^{r}\sum_{j=0}^{r}\eta_i^r(t)\eta_j^r(t)p_{ij}(\boldsymbol{x}),$$

where p_{ij} for $1\leq i,j\leq r$ solve the Poisson equation

$$-\Delta p_{ij}=\sum_{k=1}^{2}\sum_{l=1}^{2}\partial_k(\phi_i^v)_l\;\partial_l(\phi_j^v)_k\qquad\text{in }\Omega.$$

At first glance, this approach seems to be a viable model order reduction method. The pressure coefficients p_{ij} for $1 \leq i, j \leq r$ can be computed in the offline stage and be used for the recovery of the pressure at time step t^n from the velocity solution in the online stage. However, the authors of [14] point out that the recovery of the pressure in the online stage requires $O(r^2 m_p)$ operations, where m_p is the number of degrees of freedom of the pressure in the FEM simulation, since one needs to compute a linear combination of the functions p_{ij}. Furthermore, the offline stage, which involves the computation of the coefficients p_{ij} for $1 \leq i, j \leq r$ takes longer than the computation of the POD modes for the pressure. This motivates the usage of a pressure mode based approach for model order reduction. Finally, in their numerical experiments Caiazzo et al [14] observed that the range of the drag coefficient in velocity mode based computations of the pressure was significantly lower than the range of the drag coefficient in the FEM simulation. Due to the lack of computational efficiency and the lack of accuracy, we will not investigate velocity mode based approaches for the pressure reconstruction and instead focus on pressure mode based approaches.

6.3.2 Method based on pressure modes

In this method, we again solve (6.5) in each time step, but we now express the pressure with the help of its POD modes, which was proposed in [36]. To derive the weak formulation, we use the boundary conditions which have been described in [26][Subsection 2.2.2, Subsection 5.4.2]. On the Dirichlet boundaries of the velocity, we apply Neumann boundary conditions to the pressure. On the Nemann boundaries of the velocity, we apply Dirichlet boundary conditions to the pressure. We thus have the boundary conditions

$$\begin{aligned} \partial_n p &= \left(\nabla \cdot (\nabla \boldsymbol{v}^r + \nabla \boldsymbol{v}^{r\,T}) - (\boldsymbol{v}^r \cdot \nabla)\boldsymbol{v}^r\right) \cdot \boldsymbol{n} && \text{on } \partial\Omega \setminus \Gamma_{\text{out}}, \\ p &= 0 && \text{on } \Gamma_{\text{out}}. \end{aligned}$$

Plugging the pressure ansatz (6.2) into the Poisson equation (6.5) and applying integration by parts, we get that

$$\begin{aligned} \sum_{j=1}^{r_p} (\nabla \phi_j^p, \nabla \phi_i^p) \alpha_j^{r,n} &= \rho \left(\nabla \cdot [(\boldsymbol{v}^r \cdot \nabla)\boldsymbol{v}^r], \phi_i^p\right) - (\nabla \bar{p}, \nabla \phi_i^p) \\ &\quad + \int_{\partial\Omega \setminus \Gamma_{\text{out}}} \left(\nabla \cdot (\nabla \boldsymbol{v}^r + \nabla \boldsymbol{v}^{r\,T}) - (\boldsymbol{v}^r \cdot \nabla)\boldsymbol{v}^r\right) \cdot \boldsymbol{n}\ \phi_i^p \ \mathrm{d}s \quad \forall 1 \leq i \leq r_p, \end{aligned}$$

where $\alpha_j^{r,n} := \alpha_j^r(t^n)$ and $\boldsymbol{v}^r := \boldsymbol{v}^{r,n} = \boldsymbol{v}^r(t^n)$. This pressure mode based reduced order model produced goal functional results that closely matched the results from the high fidelity simulations in [14, 26]. In the next subsection, we will investigate whether we can achieve solutions of similar accuracy by using a machine learning approach, which doesn't require the solution of the Poisson equation (6.5).

6.3.3 Neural network based pressure reconstruction

To recover the reduced pressure from the reduced velocity, we consider feedforward neural networks $\mathrm{NN} : \mathbb{R}^r \to \mathbb{R}^{r_p}$, where r is the number of velocity POD modes and r_p is the number of pressure POD modes.

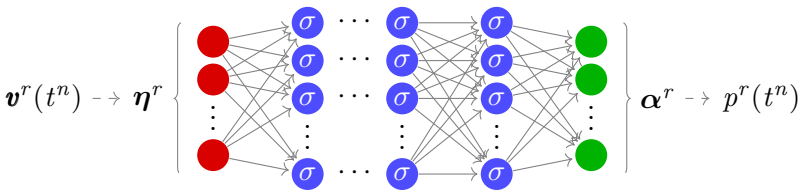


Figure 6.11: Fully connected neural network which is being used for the reconstruction of the reduced pressure from the reduced velocity

The neural networks can be expressed as

$$\mathrm{NN}(x) = T^{(L)} \circ \sigma \circ T^{(L-1)} \circ \cdots \circ \sigma \circ T^{(1)}(x),$$

where $T^{(i)} : \mathbb{R}^{n_{i-1}} \to \mathbb{R}^{n_i}, y \mapsto W^{(i)}y + b^{(i)}$ are affine transformations for $1 \leq i \leq L$, with weight matrices $W^{(i)} \in \mathbb{R}^{n_i \times n_{i-1}}$ and bias vectors $b^{(i)} \in \mathbb{R}^{n_i}$. Here, n_i denotes the number of neurons in the i.th layer with $n_0 = r$ and $n_L = r_p$. $\sigma : \mathbb{R} \to \mathbb{R}$ is a nonlinear activation function, which is the hyperbolic tangent function in this thesis. The activation function is being applied element-wise to the output of the affine transformation. Using neural networks to recover the pressure is motivated by the universal approximation theorem [19, 34, 56], which states that continuous functions can be approximated to arbitrary precision by single hidden layer neural networks.

To train the parameters of the neural network, i.e. the weights and biases of the affine transformations, we first used the snapshots of the FEM simulation and computed the reduced velocities $\boldsymbol{v}^r$ and the reduced pressures p^r. The reduced velocities and reduced pressures can be expressed as a linear combination of their respective POD modes and the mean (see (6.1) and (6.2)). Since these reduced solutions can be fully described by their coefficients, we use the coefficients of the velocity $\boldsymbol{\eta}^r \in \mathbb{R}^r$ as the input to the neural network and the coefficients of the pressure $\boldsymbol{\alpha}^r \in \mathbb{R}^{r_p}$ as the target output of the neural network. In this way, we are trying to learn the mapping

$$\mathrm{NN}(\boldsymbol{\eta}^r) = \boldsymbol{\alpha}^r$$

for all input-output pairs $(\boldsymbol{\eta}^r, \boldsymbol{\alpha}^r) \in \mathbb{R}^r \times \mathbb{R}^{r_p}$ in the training dataset. To learn such a mapping, during training we try to minimize the mean squared error loss

$$\ell\left(\left\{\boldsymbol{\eta}^{r,i}\right\}_{i=1}^n, \left\{\boldsymbol{\alpha}^{r,i}\right\}_{i=1}^n\right) = \frac{1}{n}\sum_{i=1}^n \left\|\mathrm{NN}\left(\boldsymbol{\eta}^{r,i}\right) - \boldsymbol{\alpha}^{r,i}\right\|^2$$

using the Adam optimizer [41], an adaptive gradient descent method, with a learning rate of 10^{-4}. We use the full dataset in each training step and stop the training after 20,000 epochs or when the loss hasn't decreased within the last 1,000 epochs. In Figure 6.12, we visualize how the loss decreases during training for different sizes of the POD.

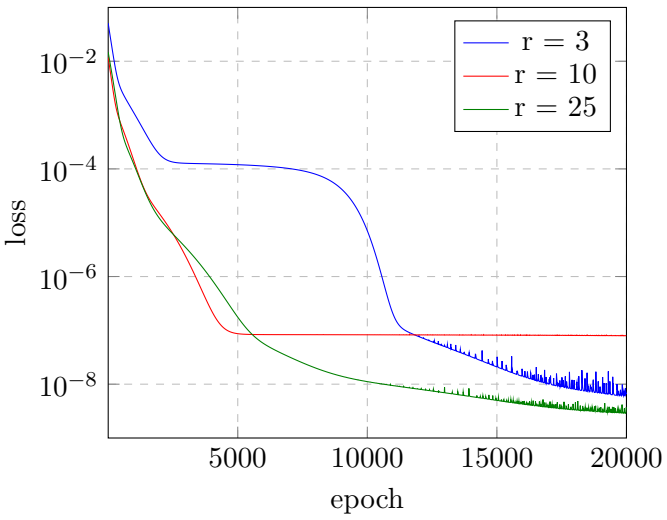


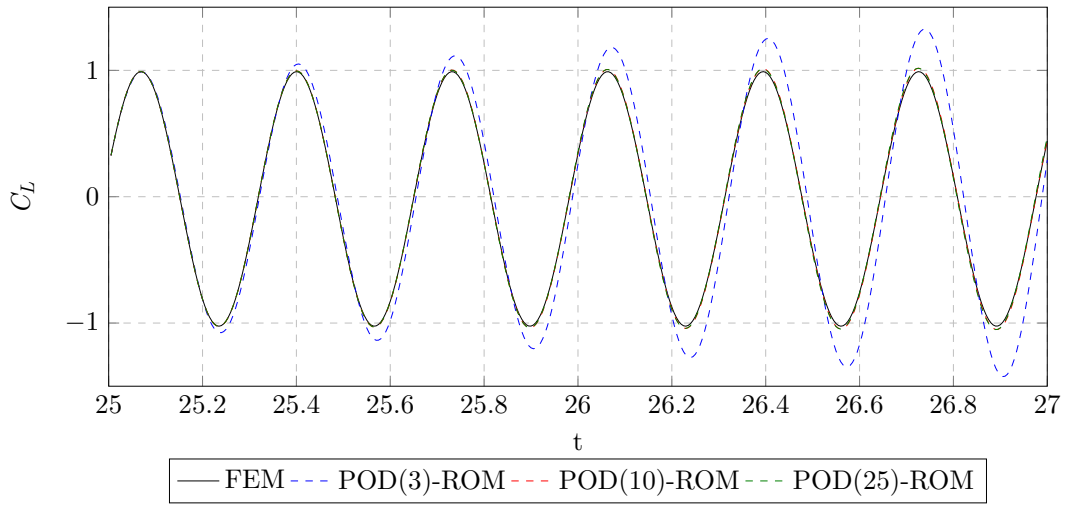
Figure 6.12: Comparison of the loss during training for different sizes of the POD basis

We observe that the loss of the neural seems to gradually decline below 10^{-8} for the different POD sizes and only for $r = 10$ the parameters of the neural network might be stuck close to a local minimum.

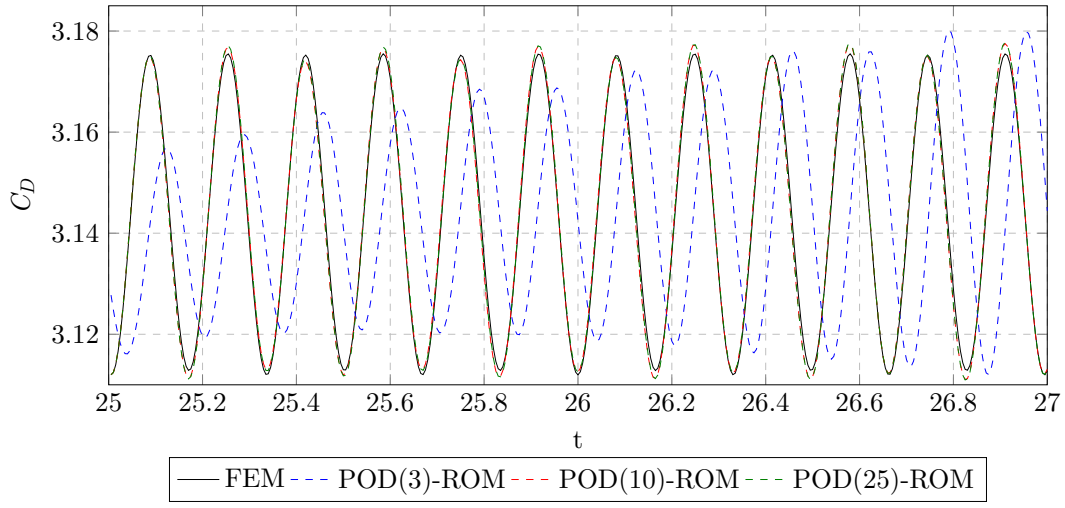
The neural networks have been implemented in LibTorch, the PyTorch [55] C++ application programming interface, using a seed of 1.

6.4 Numerical results

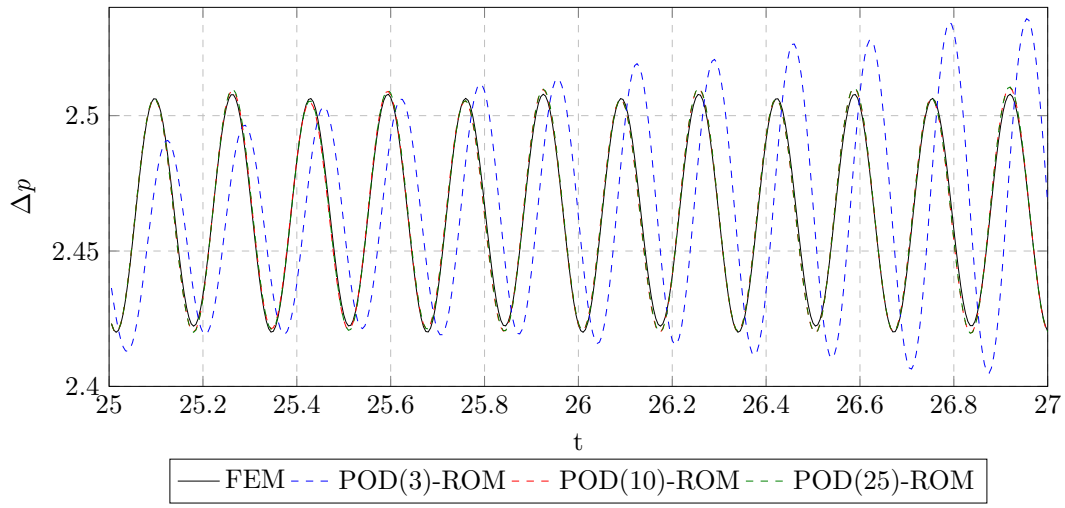
Finally, we will take a closer look at the numerical results of the reduced order model and assess its computational efficiency and robustness.



(a) Lift coefficient



(b) Drag coefficient



(c) Pressure difference

Figure 6.13: Comparison of quantities of interest of the FEM solution and the POD-ROM solution for different sizes of the POD basis

In Figure 6.13, we plot the drag and lift coefficients and the pressure difference between 25 s and 27 s for the finite element solution and the reduced order model solution with different POD sizes. We observe that a POD basis of size 3 is not sufficiently big to be able to capture the correct dynamics of the physical system and there is a large error between this POD-ROM solution and the FEM solution. On the other hand, the reduced order model simulations with a bigger POD basis size, i.e. $r = 10$ or $r = 25$, yield much more accurate predictions of the goal functionals, which are almost indistinguishable from the FEM results. Moreover, the difference between the results for a basis of size 10 and a basis of size 25 is very small, which hints at an observation that we will make later on: If the POD basis is big enough, i.e. $r = 10$ or $r = 25$, adding a few basis vectors in the POD leads only to minor improvements in the accuracy of the reduced order model. Finally, we mention that the observations of the goal functional results can also be found in the literature, e.g. in [14][Fig. 7].

Thus far, we have noticed that the values of the goal functionals of the reduced order models look promising, but how is the efficiency of the reduced order models compared to the finite element simulation? The direct numerical solver (FEM) took 396.270 seconds for the computations. We will now compare this to the computation times of the POD-ROM for different sizes of the POD basis.

POD size	NN time	Absolute time	Relative time
1	8 s	40.457 s	10.21 %
2	8 s	39.856 s	10.06 %
3	8 s	40.290 s	10.17 %
4	8 s	40.785 s	10.29 %
5	8 s	41.013 s	10.35 %
6	8 s	41.698 s	10.52 %
7	9 s	42.540 s	10.74 %
8	9 s	41.350 s	10.43 %
9	9 s	43.085 s	10.87 %
10	10 s	43.396 s	10.95 %
11	4 s	37.660 s	9.50 %
12	4 s	38.573 s	9.73 %
13	4 s	38.204 s	9.64 %
14	6 s	40.716 s	10.27 %
15	8 s	42.508 s	10.73 %
16	9 s	44.493 s	11.23 %
17	10 s	45.328 s	11.44 %
18	10 s	45.471 s	11.47 %
19	11 s	46.467 s	11.73 %
20	11 s	47.013 s	11.86 %
21	11 s	47.899 s	12.09 %
22	12 s	48.456 s	12.23 %
23	11 s	47.374 s	11.95 %
24	11 s	49.166 s	12.41 %
25	11 s	48.795 s	12.31 %

Figure 6.14: Comparison of FEM simulation times with computation time of POD-ROM depending on the size of the POD basis for the Navier-Stokes equations

In Figure 6.14, the absolute time denotes the total computation time for determining the POD basis vectors and reduced order modelling. Relative time is the quotient of the absolute time of the POD-ROM method and the computation time of the finite element simulation. We observe that a bigger POD basis leads to a small increase in the computation time, which for our reduced order models is between 9.50% and 12.31% of the computation time of finite element simulation. This means that in our experiments the ROM was on average 9.15 times faster than the direct numerical simulation.

We have established that the POD-ROM leads to a computational speed up, but how much accuracy have we lost in this trade-off? To answer this question, we will work with the root mean squared error as an evaluation metric, which is defined as

$$RMSE := \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\tilde{y}_i - y_i)^2},$$

where $\{y_i\}_{i=1}^n$ are the values of the goal functional of the FEM solution and $\{\tilde{y}_i\}_{i=1}^n$ are the values of the goal functional of the POD-ROM solution. In Figure 6.17, we have plotted the root mean squared error for the drag and lift coefficient and the pressure difference between the FEM solution and the POD-ROM solution for different sizes of the POD basis. We observe that the root mean squared error in the goal functionals is the largest for a POD basis which contains between 1 and 3 vectors. If we then increase the POD basis size, the error decays rapidly until about a POD basis size of 15, at which point the error seems to stagnate. This signifies that a POD size of 15 is sufficient to achieve nearly optimal root mean squared error in the goal functionals and even smaller POD bases, which consist of only 8 or 10 vectors, already yield accurate results. Taking a look into the literature, we notice that our root mean squared error results are of the same magnitude as the results reported in [14][Fig. 8]

Finally, we should also pay attention to the neural network architecture, i.e. the size of the neural network, which we are using for the pressure reconstruction. For the previous numerical experiments, we have been using a neural network with one hidden layer and 30 neurons in this hidden layer. At this point we need to ask the question whether an increase in the number of hidden layers or the number of neurons therein leads to a decrease in the error between the FEM and the POD-ROM solution. For these numerical experiments we will use a fixed POD basis size of 25 vectors.

Neurons in the hidden layer(s)	NN time	Absolute time	Relative time
(30)	11 s	49.147 s	12.40 %
(30, 30)	18 s	61.931 s	15.63 %
(60, 60)	21 s	65.296 s	16.48 %

Figure 6.15: Comparison of FEM simulation times with computation time of POD-ROM depending on the number of hidden layers and the number of neurons therein

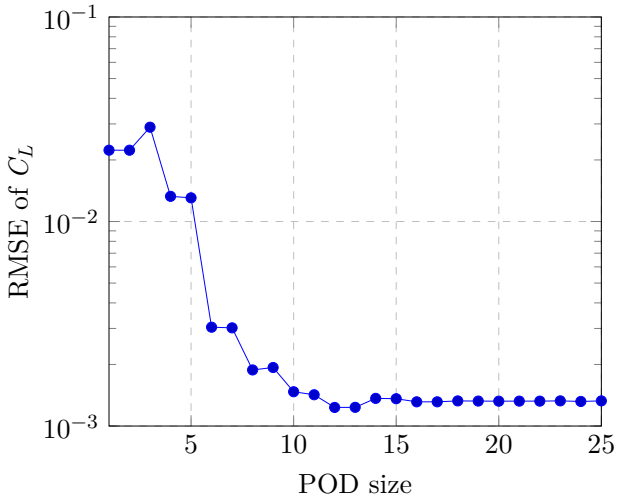
In Figure 6.15, we observe that more hidden layers and a higher number of neurons in these hidden layers leads to an increase in training time of the neural network and a longer computation time of the POD-ROM method, since bigger neural networks also take longer for inference, i.e. it takes longer to make predictions with more complicated networks.

Neurons in the hidden layer(s)	Drag RMSE	Lift RMSE	Pressure difference RMSE
(30)	$1.3260 \cdot 10^{-3}$	$1.4504 \cdot 10^{-2}$	$1.6944 \cdot 10^{-3}$
(30, 30)	$1.3288 \cdot 10^{-3}$	$1.4402 \cdot 10^{-2}$	$1.6990 \cdot 10^{-3}$
(60, 60)	$1.3246 \cdot 10^{-3}$	$1.4507 \cdot 10^{-2}$	$1.6942 \cdot 10^{-3}$

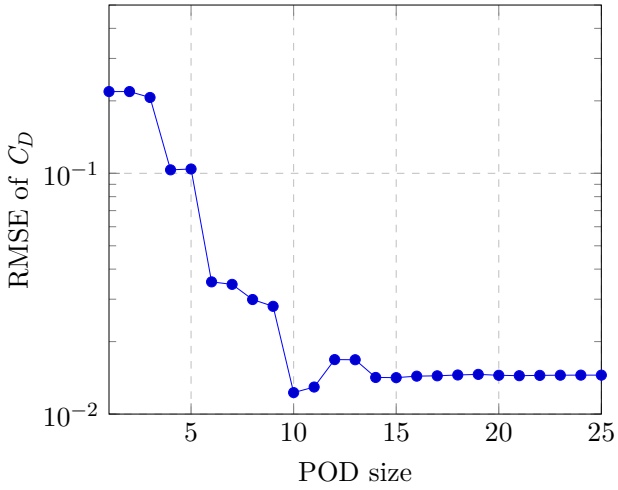
Figure 6.16: Root mean squared error of the goal functionals of the FEM and the ROM solution depending on the number of hidden layers and the number of neurons therein

We repeated the experiment for different sized neural networks to investigate whether larger networks produce better approximations of the goal functionals. However, in Figure 6.16 we don't observe an improvement in the root means squared error between the goal functionals of the FEM simulation and the goal functionals of the POD-ROM simulation, if we use larger networks. Hence, a neural network with one hidden layer containing 30 neurons seems to be the best choice from these three neural

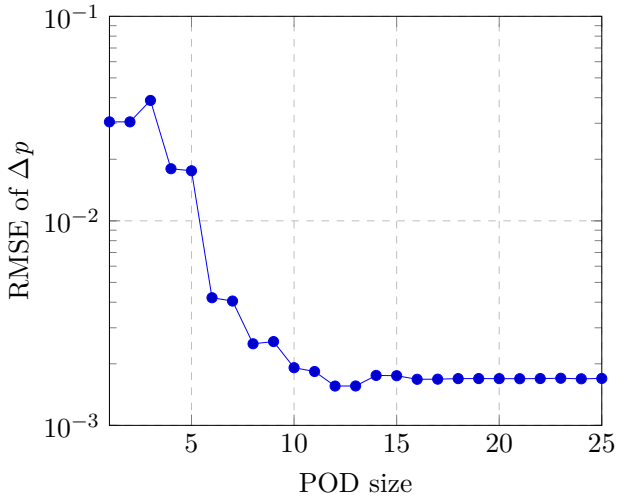
network architectures, since it has the fastest computation and inference time, and it is sufficiently accurate when compared to the other neural networks.



(a) Error of lift coefficient



(b) Error of drag coefficient



(c) Error of pressure difference

Figure 6.17: Root mean squared error of the goal functionals of the FEM and the ROM solution depending on the size of the POD basis

Chapter 7

Conclusion and outlook

7.1 Conclusion

In this thesis, we investigated a proper orthogonal decomposition based reduced order model for the time-dependent Navier-Stokes equations. The proper orthogonal decomposition leverages the data from the direct numerical simulations and creates a new basis for reduced order modelling. If the eigenvalues of the correlation matrix of the snapshots decay rapidly, it is sufficient to use only a few basis vectors for reduced order modelling and we achieve a significant speedup over the high fidelity simulation. In our numerical experiments of the heat equation and the Navier-Stokes equations, this condition has been satified. Thus, we observed that a POD basis with around 10 basis vectors yields accurate results and produces a tenfold speedup compared to the FEM simulation. In this thesis, we considered separate reduced order models for the velocity and pressure of the Navier-Stokes equations. Multiple methods for the pressure reconstruction have been presented, including a neural network based approach which is new to the best of our knowledge. Moreover, it has been demonstrated experimentally that our neural network based pressure reconstruction has a similar computational efficiency and accuracy as the methods from the literature.

7.2 Outlook

Based on the observations made in this thesis and current trends in the reduced order modelling community, we present several ideas for possible future developments:

Hyperreduction techniques for nonlinear partial differential equations

Nonlinear partial differential equations are inherently different to solve with numerical methods and this can lead to the reduced order model not being fully independent of the dimension of finite element space [30][Subsection 2.3.3]. In this thesis, for the treatment of the nonlinearity from the Navier-Stokes equations, we assembled a rank 3 tensor $C \in \mathbb{R}^{r\times r\times r}$ for the convection term. To avoid an expensive evaluation of the nonlinear terms in the reduced order model, in the literature either the Navier-Stokes equations are being linearized [14] or hyperreduction techniques for the nonlinear term are being used, where the evaluation of the nonlinear term only depends on the dimension r of the POD space. Popular choices are empirical interpolation methods, such as EIM [8] (empirical interpolation method), DEIM [16, 17] (discrete empirical interpolation method) and Q-DEIM [21] (QR decomposition based DEIM). Alternative methods are missing point estimation [3], best points interpolation [50] and Gauss-Newton with approximated tensor quantities [1].

Deep learning based reduced order modelling

It has been shown in this thesis that reduced order models for the Navier-Stokes equations can be created with the help of the proper orthogonal decomposition. However, there are a few drawbacks to this approach. As mentioned above, the evaluation of the nonlinearity typically requires additional hyperreduction techniques. Furthermore, in a monolithic reduced order model for the Navier-Stokes equations the reduced spaces don't satisfy the inf-sup condition anymore and one needs to add stabilization terms to the reduced weak form of the Navier-Stokes equations [54]. Using a POD-ROM ansatz becomes even more challenging when we consider more complicated problems, e.g. fluid structure interaction. On the other hand, deep learning methods have found many applications in computational fluid dynamics in recent years [44, 10]. Therefore, we start to see more and more deep learning based approaches, which try to circumvent the limitations of traditional model order reduction techniques [46, 24].

Reduced order modelling with snapshots from adaptively refined meshes

Solving fluid dynamics problems with many million unknowns is computationally very expensive. Therefore, in this work we employed a proper orthogonal decomposition based reduced order model, which reduces the number of unknowns to around 10 unknowns. However, one could also try to reduce the number of unknowns in the direct numerical simulation by employing adaptive finite elements [7]. In [29], the authors extend the proper orthogonal decomposition based reduced order modelling to be able to use snapshots from adaptively refined meshes.

Appendix A

Technical proofs

A.1 Theorem 3.3.1

Proof. The proof is based on [67][Theorem 1.1].

PART 1: $\{\boldsymbol{\psi}_i\}_{i=1}^r$ solves $(\mathbf{P}^r)$

$(\mathbf{P}^r)$ is a maximization problem with equality constraints. We can thus use the Lagrange formalism to verify that the left singular vectors $\{\boldsymbol{\psi}_i\}_{i=1}^r$ solve the optimization problem $(\mathbf{P}^r)$.

Part 1a): First-order necessary conditions

We introduce the Lagrange function

$$\mathcal{L} : \underbrace{\mathbb{R}^m \times \cdots \times \mathbb{R}^m}_{r \text{ times}} \times \mathbb{R}^{r\times r}, \quad \left(\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_r, \Lambda := (\lambda_{ij})_{i,j=1}^r\right) \mapsto \sum_{i=1}^r \sum_{j=1}^n |(\boldsymbol{u}_j, \boldsymbol{\psi}_i)_{\mathbb{R}^m}|^2 + \sum_{i,j=1}^r \lambda_{ij} \left(\delta_{ij} - (\boldsymbol{\psi}_i, \boldsymbol{\psi}_j)_{\mathbb{R}^m}\right).$$

Then, the first-order necessary optimality conditions for $(\mathbf{P}^r)$ are the Karush-Kuhn-Tucker conditions (KKT conditions), which are given by

$$\partial_{\boldsymbol{\psi}_i} \mathcal{L}(\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_r, \Lambda) = \mathbf{0} \qquad \text{for } 1 \le i \le r, \tag{A.1}$$

$$\partial_{\lambda_{ij}} \mathcal{L}(\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_r, \Lambda) = \delta_{ij} - (\boldsymbol{\psi}_i, \boldsymbol{\psi}_j)_{\mathbb{R}^m} = 0 \qquad \text{for } 1 \le i, j \le r. \tag{A.2}$$

Notice that $\partial_{\lambda_{ij}} \mathcal{L} = 0$ makes sure that $\{\boldsymbol{\psi}_i\}_{i=1}^r$ is admissible, i.e. in this case that $\{\boldsymbol{\psi}_i\}_{i=1}^r$ is orthonormal. Therefore, let us take a closer look at the conditions $\partial_{\boldsymbol{\psi}_i} \mathcal{L} = \mathbf{0}$ for $1 \le i \le r$. Then we have for $1 \le k \le r$ that

$$\partial_{\boldsymbol{\psi}_k} \mathcal{L}(\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_r, \Lambda) \boldsymbol{\delta\psi}_k = 0 \quad \forall \boldsymbol{\delta\psi}_k \in \mathbb{R}^m.$$

Computing the derivative then yields

$$\begin{aligned}
\partial_{\boldsymbol{\psi}_k} \mathcal{L}(\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_r, \Lambda) \delta\boldsymbol{\psi}_k &= 2 \sum_{i=1}^r \sum_{j=1}^n (\boldsymbol{u}_j, \boldsymbol{\psi}_i)_{\mathbb{R}^m} (\boldsymbol{u}_j, \boldsymbol{\delta\psi}_k)_{\mathbb{R}^m} \delta_{ik} \\
&\quad - \sum_{i,j=1}^r \lambda_{ij} (\boldsymbol{\psi}_i, \boldsymbol{\delta\psi}_k)_{\mathbb{R}^m} \delta_{jk} - \sum_{i,j=1}^r \lambda_{ij} (\boldsymbol{\delta\psi}_k, \boldsymbol{\psi}_j)_{\mathbb{R}^m} \delta_{ki} \\
&= 2 \sum_{j=1}^n (\boldsymbol{u}_j, \boldsymbol{\psi}_k)_{\mathbb{R}^m} (\boldsymbol{u}_j, \boldsymbol{\delta\psi}_k)_{\mathbb{R}^m} - \sum_{i=1}^r (\lambda_{ik} + \lambda_{ki}) (\boldsymbol{\psi}_i, \boldsymbol{\delta\psi}_k)_{\mathbb{R}^m} \\
&= \left(2 \sum_{j=1}^n (\boldsymbol{u}_j, \boldsymbol{\psi}_k)_{\mathbb{R}^m} \boldsymbol{u}_j - \sum_{i=1}^r (\lambda_{ik} + \lambda_{ki}) \boldsymbol{\psi}_i, \boldsymbol{\delta\psi}_k \right)_{\mathbb{R}^m}.
\end{aligned}$$

Using that $\partial_{\boldsymbol{\psi}_k}\mathcal{L}(\boldsymbol{\psi}_1,\dots,\boldsymbol{\psi}_r,\Lambda)\boldsymbol{\delta\psi}_k = 0$, we get that

$$\sum_{j=1}^{n}(\boldsymbol{u}_j,\boldsymbol{\psi}_k)_{\mathbb{R}^m}\boldsymbol{u}_j = \frac{1}{2}\sum_{i=1}^{r}(\lambda_{ik}+\lambda_{ki})\boldsymbol{\psi}_i \tag{A.3}$$

for all $1 \le k \le r$. Since $Y = [\boldsymbol{u}_1,\dots,\boldsymbol{u}_n]$, we get by direct computation that

$$YY^T\boldsymbol{\psi} = Y\begin{pmatrix}(\boldsymbol{u}_1,\boldsymbol{\psi})_{\mathbb{R}^m}\\ \vdots \\ (\boldsymbol{u}_n,\boldsymbol{\psi})_{\mathbb{R}^m}\end{pmatrix} = \sum_{j=1}^{n}(\boldsymbol{u}_j,\boldsymbol{\psi})_{\mathbb{R}^m}\boldsymbol{u}_j \quad \forall\boldsymbol{\psi}\in\mathbb{R}^m. \tag{A.4}$$

Plugging this into (A.3) yields

$$YY^T\boldsymbol{\psi}_k = \frac{1}{2}\sum_{i=1}^{r}(\lambda_{ik}+\lambda_{ki})\boldsymbol{\psi}_i \tag{A.5}$$

for all $1 \le k \le r$. We will now prove by induction that the condition $\partial_{\boldsymbol{\psi}_i}\mathcal{L} = \mathbf{0}$ for $1 \le i \le r$ can be rewritten as the eigenvalue problem

$$YY^T\boldsymbol{\psi}_i = \lambda_i\boldsymbol{\psi}_i \quad \text{for } 1 \le i \le r. \tag{A.6}$$

Induction start: We have $r = 1$ and by (A.5) with $k = 1$ it holds that

$$YY^T\boldsymbol{\psi}_1 = \lambda_1\boldsymbol{\psi}_1,$$

where $\lambda_1 := \lambda_{1,1}$.

Induction assumption: Next, we assume that for $r \ge 1$ the condition $\partial_{\boldsymbol{\psi}_i}\mathcal{L} = \mathbf{0}$ for $1 \le i \le r$ can be rewritten as

$$YY^T\boldsymbol{\psi}_i = \lambda_i\boldsymbol{\psi}_i \quad \text{for } 1 \le i \le r.$$

Induction step: Now we need to show that this statement still holds when we go from r to $r+1$. Due to the induction assumption, it only remains to show that

$$YY^T\boldsymbol{\psi}_{r+1} = \lambda_{r+1}\boldsymbol{\psi}_{r+1}.$$

By (A.5), we then have that

$$YY^T\boldsymbol{\psi}_{r+1} = \frac{1}{2}\sum_{i=1}^{r+1}(\lambda_{i,r+1}+\lambda_{r+1,i})\boldsymbol{\psi}_i. \tag{A.7}$$

Since $\{\boldsymbol{\psi}_i\}_{i=1}^{r+1}$ also needs to satisfy $\partial_{\lambda_{ij}}\mathcal{L} = 0$ for $1 \le i,j \le r+1$, $\{\boldsymbol{\psi}_i\}_{i=1}^{r+1}$ is orthonormal and in particular

$$(\boldsymbol{\psi}_{r+1},\boldsymbol{\psi}_j)_{\mathbb{R}^m} = 0 \quad \text{for } 1 \le j \le r.$$

Multiplying this equation by λ_j and using the induction assumption, we get

$$0 = \lambda_j(\boldsymbol{\psi}_{r+1},\boldsymbol{\psi}_j)_{\mathbb{R}^m} = (\boldsymbol{\psi}_{r+1},\, YY^T\boldsymbol{\psi}_j)_{\mathbb{R}^m}.$$

Using that YY^T is symmetrical and equation (A.7), we finally arrive at

$$\begin{aligned}0 = (YY^T\boldsymbol{\psi}_{r+1},\boldsymbol{\psi}_j)_{\mathbb{R}^m} &= \frac{1}{2}\sum_{i=1}^{r+1}(\lambda_{i,r+1}+\lambda_{r+1,i})\underbrace{(\boldsymbol{\psi}_i,\boldsymbol{\psi}_j)_{\mathbb{R}^m}}_{=\delta_{ij}}\\ &= \frac{1}{2}(\lambda_{j,r+1}+\lambda_{r+1,j}),\end{aligned}$$

which means that $\lambda_{r+1,i} = -\lambda_{i,r+1}$ for $1 \leq i \leq r$. Plugging these relations of the Lagrange multipliers into (A.7) gets simplified to

$$\begin{aligned} YY^T\boldsymbol{\psi}_{r+1} &= \frac{1}{2}\sum_{i=1}^{r}(\lambda_{i,r+1} + \lambda_{r+1,i})\boldsymbol{\psi}_i + \lambda_{r+1,r+1}\boldsymbol{\psi}_{r+1} \\ &= \frac{1}{2}\sum_{i=1}^{r}(\lambda_{i,r+1} - \lambda_{i,r+1})\boldsymbol{\psi}_i + \lambda_{r+1,r+1}\boldsymbol{\psi}_{r+1} \\ &= \lambda_{r+1,r+1}\boldsymbol{\psi}_{r+1}, \end{aligned}$$

which proves the induction step, if we set $\lambda_{r+1} := \lambda_{r+1,r+1}$.

From the singular value decomposition, it follows that the left singular vectors $\{\boldsymbol{\psi}_i\}_{i=1}^r$ satisfy the eigenvalue problem

$$YY^T\boldsymbol{\psi}_i = \lambda_i\boldsymbol{\psi}_i \quad \text{for } 1 \leq i \leq r.$$

Furthermore, $\Psi = [\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_m]$ is an orthogonal matrix. Hence, $\{\boldsymbol{\psi}_i\}_{i=1}^r$ is also orthonormal. We have thus shown that the left singular vectors $\{\boldsymbol{\psi}_i\}_{i=1}^r$ fulfill the KKT conditions (A.1) and (A.2).

Part 1b): Second-order necessary conditions

A necessary condition for a maximum is that the Hessian matrix of the Lagrange function is negative semidefinite on $\ker \nabla c(\boldsymbol{\psi})$, i.e

$$\tilde{\boldsymbol{\psi}}^T\nabla^2_{\boldsymbol{\psi}\boldsymbol{\psi}}\mathcal{L}(\boldsymbol{\psi}, \Lambda)\tilde{\boldsymbol{\psi}} \leq 0 \qquad \forall\tilde{\boldsymbol{\psi}} := \begin{pmatrix} \tilde{\boldsymbol{\psi}}_1 \\ \vdots \\ \tilde{\boldsymbol{\psi}}_r \end{pmatrix} \in \ker \nabla c(\boldsymbol{\psi}) \subset \mathbb{R}^{mr}.$$

Here, we use the notation

$$\boldsymbol{\psi} := \begin{pmatrix} \boldsymbol{\psi}_1 \\ \vdots \\ \boldsymbol{\psi}_r \end{pmatrix} \in \mathbb{R}^{mr}$$

for the big vector that contains the first r left singular vectors of Y. The functions $c_{ij}(\boldsymbol{\psi}) = \delta_{ij} - (\boldsymbol{\psi}_i, \boldsymbol{\psi}_j)_{\mathbb{R}^m}$ for $1 \leq i, j \leq r$ describe the constraints and $\nabla c(\boldsymbol{\psi})$ is defined as

$$\nabla c(\boldsymbol{\psi}) := \begin{pmatrix} \nabla c_{1,1}(\boldsymbol{\psi}) \\ \nabla c_{1,2}(\boldsymbol{\psi}) \\ \vdots \\ \nabla c_{r,r}(\boldsymbol{\psi}) \end{pmatrix} \in \mathbb{R}^{r^2 \times mr}.$$

The derivatives of these constraint functions read

$$\partial_{\boldsymbol{\psi}_k} c_{ij}(\boldsymbol{\psi}) = -\boldsymbol{\psi}_i^T\delta_{jk} - \boldsymbol{\psi}_j^T\delta_{ik}$$

for $1 \leq i, j, k \leq r$. Therefore, it holds that

$$\begin{aligned} \sum_{k=1}^{r}\partial_{\boldsymbol{\psi}_k} c_{ij}(\boldsymbol{\psi})\tilde{\boldsymbol{\psi}}_k &= \sum_{k=1}^{r} -(\boldsymbol{\psi}_i, \tilde{\boldsymbol{\psi}}_k)_{\mathbb{R}^m}\delta_{jk} - (\boldsymbol{\psi}_j, \tilde{\boldsymbol{\psi}}_k)_{\mathbb{R}^m}\delta_{ik} \\ &= -(\boldsymbol{\psi}_i, \tilde{\boldsymbol{\psi}}_j)_{\mathbb{R}^m} - (\boldsymbol{\psi}_j, \tilde{\boldsymbol{\psi}}_i)_{\mathbb{R}^m}. \end{aligned}$$

For $\tilde{\boldsymbol{\psi}} \in \ker \nabla c(\boldsymbol{\psi})$, we have $\nabla c(\boldsymbol{\psi})\tilde{\boldsymbol{\psi}} = \mathbf{0}$, which is equivalent to

$$(\nabla c_{ij}(\boldsymbol{\psi}), \tilde{\boldsymbol{\psi}})_{\mathbb{R}^{mr}} = 0 \quad \text{for } 1 \leq i, j \leq r.$$

We have already evaluated this inner product, namely

$$\begin{aligned} 0 \overset{!}{=} (\nabla c_{ij}(\boldsymbol{\psi}), \tilde{\boldsymbol{\psi}})_{\mathbb{R}^{mr}} &= \sum_{k=1}^{r} \partial_{\boldsymbol{\psi}_k} c_{ij}(\boldsymbol{\psi}) \tilde{\boldsymbol{\psi}}_k \\ &= -(\boldsymbol{\psi}_i, \tilde{\boldsymbol{\psi}}_j)_{\mathbb{R}^m} - (\boldsymbol{\psi}_j, \tilde{\boldsymbol{\psi}}_i)_{\mathbb{R}^m}. \end{aligned}$$

We observe that $(\boldsymbol{\psi}_i, \tilde{\boldsymbol{\psi}}_j)_{\mathbb{R}^m} = -(\boldsymbol{\psi}_j, \tilde{\boldsymbol{\psi}}_i)_{\mathbb{R}^m}$ and in particular

$$\left|(\boldsymbol{\psi}_i, \tilde{\boldsymbol{\psi}}_j)_{\mathbb{R}^m}\right|^2 = \left|(\boldsymbol{\psi}_j, \tilde{\boldsymbol{\psi}}_i)_{\mathbb{R}^m}\right|^2 \tag{A.8}$$

holds true for $1 \leq i, j \leq r$.

From Part 1a), we recall from (A.5) that

$$\partial_{\boldsymbol{\psi}_k} \mathcal{L}(\boldsymbol{\psi}, \Lambda) = 2\, Y Y^T \boldsymbol{\psi}_k - \sum_{j=1}^{n} (\lambda_{jk} + \lambda_{kj}) \boldsymbol{\psi}_j$$

for $1 \leq k \leq r$. Differentiating this again yields

$$\partial_{\boldsymbol{\psi}_l} \partial_{\boldsymbol{\psi}_k} \mathcal{L}(\boldsymbol{\psi}, \Lambda) = 2\, Y Y^T \delta_{lk} - (\lambda_{lk} + \lambda_{kl}) I \tag{A.9}$$

for $1 \leq l, k \leq r$, where $I \in \mathbb{R}^{m \times m}$ is the identity matrix. The Hessian matrix of the Lagrange function reads

$$\nabla^2_{\boldsymbol{\psi}\boldsymbol{\psi}} \mathcal{L}(\boldsymbol{\psi}, \Lambda) = \begin{pmatrix} \partial_{\boldsymbol{\psi}_1} \partial_{\boldsymbol{\psi}_1} \mathcal{L}(\boldsymbol{\psi}, \Lambda) & \partial_{\boldsymbol{\psi}_1} \partial_{\boldsymbol{\psi}_2} \mathcal{L}(\boldsymbol{\psi}, \Lambda) & \cdots & \partial_{\boldsymbol{\psi}_1} \partial_{\boldsymbol{\psi}_r} \mathcal{L}(\boldsymbol{\psi}, \Lambda) \\ \partial_{\boldsymbol{\psi}_2} \partial_{\boldsymbol{\psi}_1} \mathcal{L}(\boldsymbol{\psi}, \Lambda) & \partial_{\boldsymbol{\psi}_2} \partial_{\boldsymbol{\psi}_2} \mathcal{L}(\boldsymbol{\psi}, \Lambda) & \cdots & \partial_{\boldsymbol{\psi}_2} \partial_{\boldsymbol{\psi}_r} \mathcal{L}(\boldsymbol{\psi}, \Lambda) \\ \vdots & \vdots & \ddots & \vdots \\ \partial_{\boldsymbol{\psi}_r} \partial_{\boldsymbol{\psi}_1} \mathcal{L}(\boldsymbol{\psi}, \Lambda) & \partial_{\boldsymbol{\psi}_r} \partial_{\boldsymbol{\psi}_2} \mathcal{L}(\boldsymbol{\psi}, \Lambda) & \cdots & \partial_{\boldsymbol{\psi}_r} \partial_{\boldsymbol{\psi}_r} \mathcal{L}(\boldsymbol{\psi}, \Lambda) \end{pmatrix} \in \mathbb{R}^{mr \times mr}.$$

We now show the negative semidefiniteness of the Hessian matrix on $\ker \nabla c(\boldsymbol{\psi})$. Let

$$\tilde{\boldsymbol{\psi}} = \begin{pmatrix} \tilde{\boldsymbol{\psi}}_1 \\ \vdots \\ \tilde{\boldsymbol{\psi}}_r \end{pmatrix} \in \ker \nabla c(\boldsymbol{\psi}) \subset \mathbb{R}^{mr}.$$

Then we have that

$$\begin{aligned} \tilde{\boldsymbol{\psi}}^T \nabla^2_{\boldsymbol{\psi}\boldsymbol{\psi}} \mathcal{L}(\boldsymbol{\psi}, \Lambda) \tilde{\boldsymbol{\psi}} &= \left(\begin{pmatrix} \tilde{\boldsymbol{\psi}}_1 \\ \vdots \\ \tilde{\boldsymbol{\psi}}_r \end{pmatrix}, \begin{pmatrix} \sum_{j=1}^r \partial_{\boldsymbol{\psi}_1} \partial_{\boldsymbol{\psi}_j} \mathcal{L}(\boldsymbol{\psi}, \Lambda) \tilde{\boldsymbol{\psi}}_j \\ \vdots \\ \sum_{j=1}^r \partial_{\boldsymbol{\psi}_r} \partial_{\boldsymbol{\psi}_j} \mathcal{L}(\boldsymbol{\psi}, \Lambda) \tilde{\boldsymbol{\psi}}_j \end{pmatrix} \right)_{\mathbb{R}^{mr}} \\ &\overset{(A.9)}{=} \left(\begin{pmatrix} \tilde{\boldsymbol{\psi}}_1 \\ \vdots \\ \tilde{\boldsymbol{\psi}}_r \end{pmatrix}, \begin{pmatrix} \sum_{j=1}^r \left[2\, Y Y^T \tilde{\boldsymbol{\psi}}_j \delta_{1j} - (\lambda_{1j} + \lambda_{j1}) \tilde{\boldsymbol{\psi}}_j \right] \\ \vdots \\ \sum_{j=1}^r \left[2\, Y Y^T \tilde{\boldsymbol{\psi}}_j \delta_{rj} - (\lambda_{rj} + \lambda_{jr}) \tilde{\boldsymbol{\psi}}_j \right] \end{pmatrix} \right)_{\mathbb{R}^{mr}}. \end{aligned}$$

Notice that we have shown in the induction in part 1a) of the proof that $\lambda_{ij} = -\lambda_{ji}$ for $i \neq j$. Using this fact and setting $\lambda_i := \lambda_{ii}$ for $1 \leq i \leq r$, the previous inner product simplifies to

$$\begin{aligned} \tilde{\boldsymbol{\psi}}^T \nabla^2_{\boldsymbol{\psi}\boldsymbol{\psi}} \mathcal{L}(\boldsymbol{\psi}, \Lambda) \tilde{\boldsymbol{\psi}} &= \left(\begin{pmatrix} \tilde{\boldsymbol{\psi}}_1 \\ \vdots \\ \tilde{\boldsymbol{\psi}}_r \end{pmatrix}, \begin{pmatrix} 2\, Y Y^T \tilde{\boldsymbol{\psi}}_1 - 2\lambda_1 \tilde{\boldsymbol{\psi}}_1 \\ \vdots \\ 2\, Y Y^T \tilde{\boldsymbol{\psi}}_r - 2\lambda_r \tilde{\boldsymbol{\psi}}_r \end{pmatrix} \right)_{\mathbb{R}^{mr}} \\ &= 2 \sum_{k=1}^{r} (\tilde{\boldsymbol{\psi}}_k, (\, Y Y^T - \lambda_k) \tilde{\boldsymbol{\psi}}_k)_{\mathbb{R}^m}. \end{aligned}$$

The left singular vectors $\{\boldsymbol{\psi}_i\}_{i=1}^m$ build an orthonormal basis of $\mathbb{R}^m$, which allows us to write the vectors $\tilde{\boldsymbol{\psi}}_i$ in their Fourier series representation

$$\tilde{\boldsymbol{\psi}}_i = \sum_{j=1}^{m} (\tilde{\boldsymbol{\psi}}_i, \boldsymbol{\psi}_j)_{\mathbb{R}^m} \boldsymbol{\psi}_j \tag{A.10}$$

for $1 \leq i \leq r$. Using this Fourier series representation and the eigenvalue relation (A.6), we get that

$$\begin{aligned}
\tilde{\boldsymbol{\psi}}^T \nabla^2_{\boldsymbol{\psi}\boldsymbol{\psi}} \mathcal{L}(\boldsymbol{\psi}, \Lambda) \tilde{\boldsymbol{\psi}} &= 2 \sum_{k=1}^{r} \sum_{i,j=1}^{m} (\tilde{\boldsymbol{\psi}}_k, \boldsymbol{\psi}_i)_{\mathbb{R}^m} (\tilde{\boldsymbol{\psi}}_k, \boldsymbol{\psi}_j)_{\mathbb{R}^m} (\boldsymbol{\psi}_i, (YY^T - \lambda_k)\boldsymbol{\psi}_j)_{\mathbb{R}^m} \\
&\overset{(A.6)}{=} 2 \sum_{k=1}^{r} \sum_{i,j=1}^{m} (\lambda_j - \lambda_k) (\tilde{\boldsymbol{\psi}}_k, \boldsymbol{\psi}_i)_{\mathbb{R}^m} (\tilde{\boldsymbol{\psi}}_k, \boldsymbol{\psi}_j)_{\mathbb{R}^m} \underbrace{(\boldsymbol{\psi}_i, \boldsymbol{\psi}_j)_{\mathbb{R}^m}}_{=\delta_{ij}} \\
&= 2 \sum_{k=1}^{r} \sum_{i=1}^{m} (\lambda_i - \lambda_k) \left| (\tilde{\boldsymbol{\psi}}_k, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \right|^2 .
\end{aligned}$$

We can now use the property (A.8) of $\ker \nabla c(\boldsymbol{\psi})$ and observe that

$$\sum_{k=1}^{r} \sum_{i=1}^{r} (\lambda_i - \lambda_k) \left| (\tilde{\boldsymbol{\psi}}_k, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \right|^2 = 0,$$

since $\lambda_i - \lambda_k = 0$ for $1 \leq i = k \leq r$ and

$$(\lambda_i - \lambda_k) \left| (\tilde{\boldsymbol{\psi}}_k, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \right|^2 + (\lambda_k - \lambda_i) \left| (\tilde{\boldsymbol{\psi}}_i, \boldsymbol{\psi}_k)_{\mathbb{R}^m} \right|^2 = 0$$

for $1 \leq i < k \leq r$, due to (A.8). Finally, since the eigenvalues $\{\lambda_i\}_{i=1}^m$ are sorted in descending order, we get the negative definiteness of the Hessian matrix:

$$\tilde{\boldsymbol{\psi}}^T \nabla^2_{\boldsymbol{\psi}\boldsymbol{\psi}} \mathcal{L}(\boldsymbol{\psi}, \Lambda) \tilde{\boldsymbol{\psi}} = 2 \sum_{k=1}^{r} \sum_{i=r+1}^{m} \underbrace{(\lambda_i - \lambda_k)}_{\leq 0} \left| (\tilde{\boldsymbol{\psi}}_k, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \right|^2 \leq 0.$$

This completes the proof that $\{\boldsymbol{\psi}_i\}_{i=1}^r$ satisfies the second-order necessary conditions for a maximum.

Part 1c): $\{\boldsymbol{\psi}_i\}_{i=1}^r$ is a maximum of $(\mathbf{P}^r)$

To prove that $\{\boldsymbol{\psi}_i\}_{i=1}^r$ is a maximum of $(\mathbf{P}^r)$, we will compute the solution to this optimization problem inductively. For $1 \leq l \leq r-1$, let the solution $\{\boldsymbol{\psi}_i\}_{i=1}^l$ to the optimization problem $(\mathbf{P}^l)$ be given. Then we compute the $(l+1)$.th POD vector by considering the equivalent maximization problem

$$\max_{\tilde{\boldsymbol{\psi}}_{l+1} \in \mathbb{R}^m} \sum_{j=1}^{n} |(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_{l+1})_{\mathbb{R}^m}|^2 \quad \text{s.t.} \quad \begin{cases} (\tilde{\boldsymbol{\psi}}_{l+1}, \boldsymbol{\psi}_j)_{\mathbb{R}^m} = 0 \quad \forall 1 \leq j \leq l, \\ \|\tilde{\boldsymbol{\psi}}\|_{\mathbb{R}^m} = 1. \end{cases} \tag{$\mathbf{P}^{l+1}_{\text{iter}}$}$$

Induction start: We start by showing that the first left singular vector $\boldsymbol{\psi}_1$ solves $\mathbf{P}^1 = \mathbf{P}^1_{\text{iter}}$. For this let $\bar{\tilde{\boldsymbol{\psi}}}_1 \in \bar{\mathbb{R}}^{\bar{m}}$ denote another normed vector. We need to show that

$$\sum_{j=1}^{n} |(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_1)_{\mathbb{R}^m}|^2 \leq \sum_{j=1}^{n} |(\boldsymbol{u}_j, \boldsymbol{\psi}_1)_{\mathbb{R}^m}|^2.$$

Since the left singular vectors $\{\boldsymbol{\psi}_i\}_{i=1}^m$ build an orthonormal basis of $\mathbb{R}^m$, we can use the Fourier series representation (A.10) of $\tilde{\boldsymbol{\psi}}_1$, which gives us

$$\begin{aligned}
\sum_{j=1}^{n} |(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_1)_{\mathbb{R}^m}|^2 &= \sum_{j=1}^{n} \left| \left(\boldsymbol{u}_j, \sum_{i=1}^{m} (\tilde{\boldsymbol{\psi}}_1, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \boldsymbol{\psi}_i \right)_{\mathbb{R}^m} \right|^2 \\
&= \sum_{j=1}^{n} \sum_{i=1}^{m} \sum_{k=1}^{m} \left(\boldsymbol{u}_j, (\tilde{\boldsymbol{\psi}}_1, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \boldsymbol{\psi}_i \right)_{\mathbb{R}^m} \left(\boldsymbol{u}_j, (\tilde{\boldsymbol{\psi}}_1, \boldsymbol{\psi}_k)_{\mathbb{R}^m} \boldsymbol{\psi}_k \right)_{\mathbb{R}^m} \\
&= \sum_{j=1}^{n} \sum_{i=1}^{m} \sum_{k=1}^{m} (\boldsymbol{u}_j, \boldsymbol{\psi}_i)_{\mathbb{R}^m} (\boldsymbol{u}_j, \boldsymbol{\psi}_k)_{\mathbb{R}^m} (\tilde{\boldsymbol{\psi}}_1, \boldsymbol{\psi}_i)_{\mathbb{R}^m} (\tilde{\boldsymbol{\psi}}_1, \boldsymbol{\psi}_k)_{\mathbb{R}^m}.
\end{aligned}$$

Combining (A.4) and (A.6), we get that

$$\sum_{j=1}^{n} (\boldsymbol{u}_j, \boldsymbol{\psi}_k)_{\mathbb{R}^m} \boldsymbol{u}_j = \lambda_k \boldsymbol{\psi}_k \tag{A.11}$$

for $1 \leq k \leq m$. We can thus further simplify the previous equation.

$$\begin{aligned}
\sum_{j=1}^{n} |(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_1)_{\mathbb{R}^m}|^2 &= \sum_{i=1}^{m}\sum_{k=1}^{m} \Bigg(\underbrace{\sum_{j=1}^{n} (\boldsymbol{u}_j, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \boldsymbol{u}_j}_{=\lambda_i \boldsymbol{\psi}_i}, \boldsymbol{\psi}_k \Bigg)_{\mathbb{R}^m} (\tilde{\boldsymbol{\psi}}_1, \boldsymbol{\psi}_i)_{\mathbb{R}^m} (\tilde{\boldsymbol{\psi}}_1, \boldsymbol{\psi}_k)_{\mathbb{R}^m} \\
&= \sum_{i=1}^{m}\sum_{k=1}^{m} \lambda_i \underbrace{(\boldsymbol{\psi}_i, \boldsymbol{\psi}_k)_{\mathbb{R}^m}}_{=\delta_{ik}} (\tilde{\boldsymbol{\psi}}_1, \boldsymbol{\psi}_i)_{\mathbb{R}^m} (\tilde{\boldsymbol{\psi}}_1, \boldsymbol{\psi}_k)_{\mathbb{R}^m} \\
&= \sum_{i=1}^{m} \lambda_i \left| (\tilde{\boldsymbol{\psi}}_1, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \right|^2
\end{aligned}$$

Note that λ_1 is the biggest eigenvalue and $\tilde{\boldsymbol{\psi}}_1$ is normed, hence

$$\sum_{j=1}^{n} |(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_1)_{\mathbb{R}^m}|^2 = \sum_{i=1}^{m} \lambda_i \left| (\tilde{\boldsymbol{\psi}}_1, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \right|^2 \leq \lambda_1 \sum_{i=1}^{m} \left| (\tilde{\boldsymbol{\psi}}_1, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \right|^2 = \lambda_1 \|\tilde{\boldsymbol{\psi}}_1\|_{\mathbb{R}^m}^2 = \lambda_1.$$

This proves the induction start, since we will show in equation (A.12) that

$$\lambda_1 = \sum_{j=1}^{n} |(\boldsymbol{u}_j, \boldsymbol{\psi}_1)_{\mathbb{R}^m}|^2.$$

Induction assumption: Next, we assume that for $r \geq 1$ the solution to the optimization problem $(\mathbf{P}^r)$ is given by the first r left singular vectors $\{\boldsymbol{\psi}_i\}_{i=1}^r$.

Induction step: Now we need to show that this statement still holds when we go from r to $r+1$. For this we need to verify that the $(r+1)$.th left singular vector $\boldsymbol{\psi}_{r+1}$ solves $\mathbf{P}_{\text{iter}}^{r+1}$. The procedure of this proof is analogous to the induction start. Let $\tilde{\boldsymbol{\psi}}_{r+1} \in \mathbb{R}^m$ denote another normed vector which is orthogonal to the vectors $\{\boldsymbol{\psi}_i\}_{i=1}^r$. We need to show that

$$\sum_{j=1}^{n} |(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_{r+1})_{\mathbb{R}^m}|^2 \leq \sum_{j=1}^{n} |(\boldsymbol{u}_j, \boldsymbol{\psi}_{r+1})_{\mathbb{R}^m}|^2.$$

Doing the same calculations as in the induction start, it can be shown that

$$\sum_{j=1}^{n} |(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_{r+1})_{\mathbb{R}^m}|^2 = \sum_{i=1}^{m} \lambda_i \left| (\tilde{\boldsymbol{\psi}}_{r+1}, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \right|^2.$$

Since $\tilde{\boldsymbol{\psi}}_{r+1}$ is orthogonal to the vectors $\{\boldsymbol{\psi}_i\}_{i=1}^r$ and the eigenvalues $\{\lambda_i\}_{i=1}^m$ are sorted in descending order, we get that

$$\begin{aligned}
\sum_{j=1}^{n} |(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_{r+1})_{\mathbb{R}^m}|^2 &= \sum_{i=1}^{m} \lambda_i \left| (\tilde{\boldsymbol{\psi}}_{r+1}, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \right|^2 \\
&= \sum_{i=r+1}^{m} \lambda_i \left| (\tilde{\boldsymbol{\psi}}_{r+1}, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \right|^2 \\
&\leq \lambda_{r+1} \sum_{i=r+1}^{m} \left| (\tilde{\boldsymbol{\psi}}_{r+1}, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \right|^2.
\end{aligned}$$

Since $\tilde{\boldsymbol{\psi}}_{r+1}$ is normed and due to (A.12), it holds that

$$\sum_{j=1}^{n} |(\boldsymbol{u}_j, \tilde{\boldsymbol{\psi}}_{r+1})_{\mathbb{R}^m}|^2 \leq \lambda_{r+1} \sum_{i=1}^{m} \left|(\tilde{\boldsymbol{\psi}}_{r+1}, \boldsymbol{\psi}_i)_{\mathbb{R}^m}\right|^2 = \lambda_{r+1} \|\tilde{\boldsymbol{\psi}}_{r+1}\|_{\mathbb{R}^m}^2 = \lambda_{r+1} = \sum_{j=1}^{n} |(\boldsymbol{u}_j, \boldsymbol{\psi}_{r+1})_{\mathbb{R}^m}|^2.$$

This completes the proof that $\{\boldsymbol{\psi}_i\}_{i=1}^r$ is a maximum of $(\mathbf{P}^r)$.

PART 2: $\arg\max(\mathbf{P}^r) = \sum_{i=1}^r \lambda_i$

Let $\boldsymbol{\psi}_i$ denote the i.th left singular vector with $1 \leq i \leq r$. It holds that

$$\begin{aligned}\sum_{j=1}^{n} |(\boldsymbol{u}_j, \boldsymbol{\psi}_i)_{\mathbb{R}^m}|^2 &= \sum_{j=1}^{n} (\boldsymbol{u}_j, \boldsymbol{\psi}_i)_{\mathbb{R}^m} (\boldsymbol{u}_j, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \\ &= \sum_{j=1}^{n} \left((\boldsymbol{u}_j, \boldsymbol{\psi}_i)_{\mathbb{R}^m} \boldsymbol{u}_j, \boldsymbol{\psi}_i\right)_{\mathbb{R}^m}.\end{aligned}$$

We can now again use equation (A.11) and the fact that $\boldsymbol{\psi}_i$ is normed, which gives us that

$$\sum_{j=1}^{n} |(\boldsymbol{u}_j, \boldsymbol{\psi}_i)_{\mathbb{R}^m}|^2 = \lambda_i (\boldsymbol{\psi}_i, \boldsymbol{\psi}_i)_{\mathbb{R}^m} = \lambda_i \|\boldsymbol{\psi}_i\|_{\mathbb{R}^m}^2 = \lambda_i. \tag{A.12}$$

In particular, we thus get by Part 1 that

$$\arg\max(\mathbf{P}^r) = \sum_{i=1}^{r} \sum_{j=1}^{n} |(\boldsymbol{u}_j, \boldsymbol{\psi}_i)_{\mathbb{R}^m}|^2 = \sum_{i=1}^{r} \lambda_i.$$

Overall, we have thus proven Theorem 3.3.1. □

Bibliography

[1] D. Amsallem, M. J. Zahr, and C. Farhat. Nonlinear model order reduction based on local reduced-order bases. International Journal for Numerical Methods in Engineering, 92(10):891–916, 2012.

[2] D. Arndt, W. Bangerth, B. Blais, T. C. Clevenger, M. Fehling, A. V. Grayver, T. Heister, L. Heltai, M. Kronbichler, M. Maier, P. Munch, J.-P. Pelteret, R. Rastak, I. Thomas, B. Turcksin, Z. Wang, and D. Wells. The `deal.II` library, version 9.2. Journal of Numerical Mathematics, 28(3):131–146, 2020.

[3] P. Astrid, S. Weiland, K. Willcox, and T. Backx. Missing Point Estimation in Models Described by Proper Orthogonal Decomposition. IEEE Transactions on Automatic Control, 53:2237–2251, 2008.

[4] J. A. Atwell, J. Borggaard, and B. King. Reduced order controllers for Burgers' equation with a nonlinear observer. International Journal of Applied Mathematics and Computer Science, 11:1311–1330, 2001.

[5] N. Aubry, P. Holmes, J. Lumley, and E. Stone. The dynamics of coherent structures in the wall region of a turbulent boundary layer. Journal of Fluid Mechanics, 192:115–173, 1988.

[6] 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, 2013.

[7] W. Bangerth and R. Rannacher. Adaptive Finite Element Methods for Differential Equations. Birkhäuser Verlag„ 2003.

[8] M. Barrault, Y. Maday, N. Nguyen, and A. Patera. An 'empirical interpolation' method: application to efficient reduced-basis discretization of partial differential equations. Comptes Rendus Mathematique, 339:667–672, 2004.

[9] P. Benner, S. Grivet-Talocia, A. Quarteroni, G. Rozza, W. Schilders, and L. M. Silveira. Model Order Reduction. Volume 3: Applications. De Gruyter, Dec. 2020. Publication Title: Volume 3 Applications.

[10] S. L. Brunton, B. R. Noack, and P. Koumoutsakos. Machine Learning for Fluid Mechanics. Annual Review of Fluid Mechanics, 52(1):477–508, 2020.

[11] M. Buffoni, S. Camarri, A. Iollo, and M. V. Salvetti. Low-dimensional modelling of a confined three-dimensional wake flow. Journal of Fluid Mechanics, 569:141–150, 2006.

[12] J. Burkardt, M. Gunzburger, and H.-C. Lee. POD and CVT-based reduced-order modeling of Navier–Stokes flows. Computer Methods in Applied Mechanics and Engineering, 196(1):337–355, 2006.

[13] L. Bystricky. Using deal.II to solve problems in computational fluid dynamics. Master's thesis, The Florida State University, 2016.

[14] A. Caiazzo, T. Iliescu, V. John, and S. Schyschlowa. A numerical investigation of velocity–pressure reduced order models for incompressible flows. Journal of Computational Physics, 259:598–616, 2014.

[15] W. Cazemier, R. Verstappen, and A. Veldman. Proper orthogonal decomposition and low-dimensional models for driven cavity flows. Physics of Fluids, 10(7):1685–1699, July 1998. Relation: http://www.rug.nl/informatica/organisatie/overorganisatie/iwi Rights: University of Groningen. Research Institute for Mathematics and Computing Science (IWI).

[16] S. Chaturantabut and D. Sorensen. Nonlinear Model Reduction via Discrete Empirical Interpolation. SIAM J. Sci. Comput., 32:2737–2764, 2010.

[17] S. Chaturantabut and D. Sorensen. A State Space Error Estimate for POD-DEIM Nonlinear Model Reduction. SIAM J. Numer. Anal., 50:46–63, 2012.

[18] L. Cordier and M. Bergmann. Réduction de dynamique par Décomposition Orthogonale aux Valeurs Propres (POD). 2006.

[19] G. Cybenko. Approximation by superpositions of a sigmoidal function. Mathematics of Control, Signals and Systems, 2(4):303–314, 1989.

[20] deal.II authors. The step-26 tutorial program. `https://www.dealii.org/current/doxygen/deal.II/step_26.html#Thetestcase`. Accessed: 2021-05-18.

[21] Z. Drmač and S. Gugercin. A New Selection Operator for the Discrete Empirical Interpolation Method—Improved A Priori Error Bound and Extensions. SIAM Journal on Scientific Computing, 38(2):A631–A648, 2016.

[22] F. Durst. Grundlagen der Strömungsmechanik: eine Einführung in die Theorie der Strömung von Fluiden. Springer-Verlag, 2007.

[23] L. C. Evans. Partial differential equations. American Mathematical Society, 2010.

[24] S. Fresca and A. Manzoni. Real-time simulation of parameter-dependent fluid flows through deep learning-based reduced order models. Fluids, 6(7), 2021.

[25] D. Galbally, K. Fidkowski, K. Willcox, and O. Ghattas. Non-linear model reduction for uncertainty quantification in large-scale inverse problems. 2009.

[26] S. Giere. Numerical and Analytical Aspects of POD-Based Reduced-Order Modeling in Computational Fluid Dynamics. PhD thesis, 2016.

[27] V. Girault and P.-A. Raviart. Finite element methods for Navier-Stokes equations: Theory and Algorithms. Springer-Verlag, Berlin; New York, 1986.

[28] G. Golub, C. Van Loan, P. Van Loan, and C. Van Loan. Matrix Computations. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 1996.

[29] C. Gräßle, M. Hinze, J. Lang, and S. Ullmann. POD model order reduction with space-adapted snapshots for incompressible flows. Adv. Comput. Math., 45:2401–2428, 2019.

[30] C. Gräßle, M. Hinze, and S. Volkwein. 2 Model order reduction by proper orthogonal decomposition, pages 47–96. De Gruyter, 2020.

[31] J. G. Heywood, R. Rannacher, and S. Turek. Artificial boundaries and flux and pressure conditions for the incompressible Navier–Stokes equations. International Journal for Numerical Methods in Fluids, 22(5):325–352, 1996.

[32] N. Higham. Matrix Nearness Problems and Applications. 1989.

[33] P. Holmes, J. L. Lumley, and G. Berkooz. Turbulence, Coherent Structures, Dynamical Systems and Symmetry. Cambridge Monographs on Mechanics. Cambridge University Press, 1996.

[34] K. Hornik. Approximation capabilities of multilayer feedforward networks. Neural Networks, 4(2):251 – 257, 1991.

[35] H. Hotelling. Analysis of a complex of statistical variables into principal components. Journal of Educational Psychology, 24:417–441,498–520, 1933.

[36] A. Imran, A. Nayfeh, and C. Ribbens. On the stability and extension of reduced-order Galerkin models in incompressible flows. a numerical study of vortex shedding. Theoretical and Computational Fluid Dynamics, 23:213–237, 07 2009.

[37] A. Iollo, A. Dervieux, J. Désidéri, and S. Lanteri. Two stable POD-based approximations to the Navier–Stokes equations. Computing and Visualization in Science, 3:61–66, 2000.

[38] A. Iollo, S. Lanteri, and J.-A. Désidéri. Stability Properties of POD–Galerkin Approximations for the Compressible Navier–Stokes Equations. Theoretical and Computational Fluid Dynamics, 13(6):377–396, Mar 2000.

[39] I. T. Jolliffe. Principal Component Analysis. Springer Series in Statistics. Springer-Verlag, New York, 2 edition, 2002.

[40] K. Karhunen. Zur Spektraltheorie stochastischer Prozesse. Annales Academiae scientiarum Fennicae. Series A. 1, Mathematica-physica. 1946.

[41] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization, arXiv:1412.6980, 2017.

[42] B. Koc, C. Mou, H. Liu, Z. Wang, G. Rozza, and T. Iliescu. Verifiability of the Data-Driven Variational Multiscale Reduced Order Model, 2021.

[43] K. Kunisch and S. Volkwein. Galerkin proper orthogonal decomposition methods for parabolic problems. Numerische Mathematik, 90(1):117–148, Nov. 2001.

[44] J. N. Kutz. Deep learning in fluid dynamics. Journal of Fluid Mechanics, 814:1–4, 2017.

[45] T. Lassila, A. Manzoni, A. Quarteroni, and G. Rozza. Model Order Reduction in Fluid Dynamics: Challenges and Perspectives, pages 235–273. Springer International Publishing, Cham, 2014.

[46] K. Lee and K. T. Carlberg. Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. Journal of Computational Physics, 404:108973, 2020.

[47] M. Loève. Probability Theory. Van Nostrand, 1955.

[48] J. L. Lumley. The structure of inhomogeneous turbulent flows. Atmospheric Turbulence and Radio Wave Propagation, pages 166–178, 1967.

[49] Z. Luo, J. Chen, P. Sun, and X. Yang. Finite element formulation based on proper orthogonal decomposition for parabolic equations. Science in China Series A: Mathematics, 52(3):585–596, Mar 2009.

[50] N. Nguyen, A. Patera, and J. Peraire. A 'best points' interpolation method for efficient approximation of parametrized function. International Journal for Numerical Methods in Engineering, 73:521 – 543, 01 2008.

[51] B. Noack, P. Papas, and P. Monkewitz. The need for a pressure-term representation in empirical Galerkin models of incompressible shear flows. Journal of Fluid Mechanics, 523, 01 2005.

[52] B. Noble, J. Daniel, and K. M. R. Collection. Applied Linear Algebra. Prentice Hall, 1977.

[53] J. Nocedal and S. J. Wright. Numerical Optimization. Springer, New York, NY, USA, second edition, 2006.

[54] M. Nonino, F. Ballarin, and G. Rozza. A Monolithic and a Partitioned, Reduced Basis Method for Fluid–Structure Interaction Problems. Fluids, 6(6), 2021.

[55] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 32. Curran Associates, Inc., 2019.

[56] A. Pinkus. Approximation theory of the MLP model in neural networks. Acta Numerica, 8:143–195, 1999.

[57] R. Preisendorfer and C. Mobley. Principal Component Analysis in Meteorology and Oceanography. Developments in atmospheric science. Elsevier, 1988.

[58] R. Rannacher. Finite Element Methods for the Incompressible Navier-Stokes Equations, pages 191–293. Birkhäuser Basel, Basel, 2000.

[59] M. Reed and B. Simon. I: Functional Analysis. Methods of Modern Mathematical Physics. Elsevier Science, 1981.

[60] T. Richter and T. Wick. Einführung in die Numerische Mathematik - Begriffe, Konzepte und zahlreiche Anwendungsbeispiele. Springer-Verlag, Berlin Heidelberg New York, 2017.

[61] M. Schäfer, S. Turek, F. Durst, E. Krause, and R. Rannacher. Benchmark Computations of Laminar Flow Around a Cylinder, pages 547–566. Vieweg+Teubner Verlag, Wiesbaden, 1996.

[62] L. Sirovich. Turbulence and the dynanics of coherent structures. Part I: Coherent structures. Quarterly of Applied Mathematics, 45(3):561–571, 1987. Full publication date: October 1987.

[63] L. Sirovich. Turbulence and the dynanics of coherent structures. Part II: Symmetries and transformations. Quarterly of Applied Mathematics, 45(3):573–582, 1987. Full publication date: October 1987.

[64] L. Sirovich. Turbulence and the dynanics of coherent structures. Part III: Dynamics and scaling. Quarterly of Applied Mathematics, 45(3):583–590, 1987. Full publication date: October 1987.

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

[66] F. Tröltzsch. Optimale Steuerung partieller Differentialgleichungen: Theorie, Verfahren und Anwendungen. Vieweg Studium. Vieweg+Teubner Verlag, 2010.

[67] S. Volkwein. Lecture notes in "Model Reduction using Proper Orthogonal Decomposition", 2011.

[68] S. Volkwein. Lecture notes in "Proper Orthogonal Decomposition: Theory and Reduced-Order Modelling". Aug. 2013.

[69] Z. Wang, I. Akhtar, J. Borggaard, and T. Iliescu. Proper orthogonal decomposition closure models for turbulent flows: A numerical comparison. Computer Methods in Applied Mechanics and Engineering, 237-240:10–26, 2012.

[70] D. Wells. Stabilization of POD-ROMs. PhD dissertation, Virginia Tech, 2015.

[71] T. Wick. Lecture notes in "Modeling, Discretization, Optimization, and Simulation of Fluid-Structure Interaction". Nov. 2019.

[72] T. Wick. Lecture notes in "Numerical Methods for Partial Differential Equations". 2020.

Statement of Authorship

I hereby declare that I am the sole author of this master thesis and that I have not used any sources other than those listed in the bibliography and identified as references. In particular, I declare that all parts of this work that have been taken from other sources have been marked as such. I further declare that I have not submitted this thesis to any other institution in order to obtain a degree.

Place, Date

Julian Roth

Selbstständigkeitserklärung

Hiermit versichere ich, dass ich diese Arbeit selbstständig verfasst und keine weiteren als die angegebenen Hilfsmittel verwendet habe. Insbesondere versichere ich, dass ich alle wörtlichen und sinngemäßen Übernahmen aus anderen Werken als solche kenntlich gemacht habe. Die Prüfungsleistung wurde bisher keiner anderen Prüfungsbehörde vorgelegt.

Ort, Datum

Julian Roth