Master Thesis

Reduced-Order Bicriterial Optimal Control of Evolution Equations

submitted by

Felix Spura

at the

Universität Konstanz

Department of Mathematics and Statistics

Konstanz, 12.04.2019

Supervisor and 1st Reviewer: Prof. Dr. Stefan Volkwein, University of Konstanz
2nd Reviewer: Jun.-Prof. Dr. Gabriele Ciaramella, University of Konstanz

Erklärung der Selbstständigkeit

Ich versichere hiermit, dass ich die vorliegende Masterarbeit mit dem Thema:

Reduced-Order Bicriterial Optimal Control of Evolution Equations

selbstständig verfasst und keine anderen Hilfsmittel als die angegebenen benutzt habe. Die Stellen, die anderen Werken dem Wortlaut oder dem Sinne nach entnommen sind, habe ich in jedem einzelnen Falle durch Angaben der Quelle, auch der benutzten Sekundärliteratur, als Entlehnung kenntlich gemacht.

Die Arbeit wurde bisher keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Konstanz, den 12. April 2019

Felix Spura

Contents

Introduction

As a special form of optimization problems, optimal control problems are aimed at finding an optimal input of control variables with respect to one or more objectives. Furthermore, the control variables oftentimes influence a certain state of the system by means of a differential equation. In particular, an objective might be to achieve a certain desired state or to minimize the cost that is caused by applying the control.
Moreover, in real-world applications like engineering, there is often a trade-off between multiple objectives. In most practical examples, there is some kind of cost that is supposed to be as low as possible. Besides, we often have a performance objective which could be one of many possible examples, such as achieving a certain desired result in as little time as possible. So it is natural to have at least two, often contradictory objectives in a real-world optimization problem. Additionally, in the context of multiple objectives a concept of optimality has to be defined, since finding a solution that is optimal for each individual objective in the classical sense is, most of the time, impossible. Therefore, we will use the notion of Pareto optimality throughout this thesis.

Having chosen an optimality concept, it is often unclear upfront how the different objectives have to be weighted such that a desirable compromise can be reached. Therefore, by providing a whole set of Pareto optimal solutions, we seek to offer a basis for a so-called decision-maker to find this kind of compromise. While determining such a set of Pareto optimal solutions it is important to ensure adequate coverage of the whole spectrum of possible solutions. To this end the choice of a suitable solution method is beneficial. Many approaches to treat multiobjective optimization problems use some kind of scalarization to obtain a number of scalar optimization problems, which can be treated more easily by well-known scalar optimization methods. The solution method used in this thesis is the Euclidean reference point method, which has the capability to approximate the Pareto optimal solutions uniformly across the set of solutions as opposed to the straightforward weighted sum method (see e.g. [1]), for which it is hard to predict how far apart the Pareto optimal solutions are.

Another aspect of optimal control problems is the possibly high dimension of the equations that are due to the discretization in numerical computations. This becomes already apparent for problems with three space dimensions that are discretized by the finite element method. If one desires a reasonably fine approximation of space, the computational effort becomes really high. Therefore, model order reduction techniques come into consideration. Especially in this thesis, proper orthogonal decomposition (POD) is employed, which has favourable properties to treat evolution problems like the one occurring in this thesis. In particular, by design, POD attempts to capture the most dominant characteristics of the solutions of the treated differential equation. A central issue while applying POD is to have a solid understanding of how the approximation error behaves and if it can be controlled in any way. If the error can be controlled, POD can be

a powerful tool, because it greatly reduces the dimension of the treated problem, which leads to a much lower computational effort. In the numerical experiments, it will be explored, how a time-efficient POD strategy can be constructed and at the same time, a sufficiently high quality of the solutions can be ensured.
The focus of this thesis is supposed to be on numerical experiments treating a bicriterial optimal control problem which is governed by a diffusion equation with time-dependent convection term including three space dimensions. Similar problems in two space dimensions have already been investigated in [2] and [3]. Most of the theoretical results concerning the Euclidean reference point method and the optimal control by a time-independent convection term were developed in [2] and tested numerically. These theoretical results and will therefore be used again in this thesis. In [3] the necessary adaptions for a time-dependent convection term were made and tested numerically. In the present thesis, we will extend these findings to the case of three space dimensions and explore associated difficulties. Furthermore, we will take a closer look at various POD strategies, analyse the results and refine those strategies gradually. In the process, new a-priori estimates that were worked out in collaboration with the author of [3] will be applied to construct a more efficient POD strategy.

The following overview shows how this thesis is structured.
In **Chapter 1**, we state basic definitions and results that do not belong to the actual topic of this thesis directly but are supposed to facilitate the understanding of some of the main results.
Chapter 2 is a compilation of relevant theoretical results, cited from [2] that will later be used to treat bicriterial optimization problems. The Euclidean reference point method is introduced as the method of choice to solve bicriterial optimization problems. Based on this method a basic algorithm that can be used to compute Pareto optimal solutions of a bicriterial optimization problem is provided.
Chapter 3 gives a condensed overview of the basic theory of proper orthogonal decomposition which is aimed at applying it to the state and adjoint equation in both the theoretical and the numerical considerations of the central optimal control problem of this thesis.
Chapter 4 builds the foundation to treat the specific optimal control problem of this thesis in the context of bicriterial optimal control problems governed by a certain evolution equation as well as how to apply POD to it. Multiple a-priori and a-posteriori error estimates regarding the POD approximation of the optimal control problem are presented which can also be applied for the numerical treatment of the optimal control problem.
Chapter 5 consists of numerical investigations which examine the practical application of algorithms to compute the Pareto front as well as employing POD in this concrete case. Three different models in three space dimensions are considered successively while various strategies to control the size of the POD basis and also to update the POD basis are tested and refined in the process. Meanwhile, the focus is put on both time efficiency and sufficient accuracy of the used algorithms with the aim of gaining further insight on how two choose different parameters in the POD strategy. Besides, possible improvements of the basic algorithm to compute the Pareto front by adjusting the employed step sizes are explored.
Finally, in the **Conclusion** we review the most important insights we have gained in the preceding chapters.

Chapter 1

Auxiliary Results and Definitions

To begin with, we state some general definitions and results, which will be used repeatedly in the subsequent chapters. Throughout this chapter, let $n \in \mathbb{N}$.

1.1 Order on $\mathbb{R}^n$

Definition 1.1. Let $\bar{\mathbb{R}} := \mathbb{R} \cup \{-\infty, \infty\}$.

(i) For $x, y \in \bar{\mathbb{R}}^n$ we declare

$$x \leq y :\Leftrightarrow x_i \leq y_i \text{ for } i \in \{1, \dots, n\},$$
$$x < y :\Leftrightarrow x_i < y_i \text{ for } i \in \{1, \dots, n\}.$$

Additionally, we define

$$x \lneqq y :\Leftrightarrow x \leq y \text{ and } x \neq y.$$

(ii) Accordingly, for $x \in \mathbb{R}^n$ we write

$$\mathbb{R}^n_{\geq x} := \{y \in \mathbb{R}^n \mid y \geq x\} \subset \mathbb{R}^n,$$
$$\mathbb{R}^n_{\gneqq x} := \{y \in \mathbb{R}^n \mid y \gneqq x\} \subset \mathbb{R}^n.$$

and we define $\mathbb{R}^n_{\leq x}$, $\mathbb{R}^n_{\lneqq x}$ analogously. Furthermore, we use the notations $\mathbb{R}^n_{\geq} := \mathbb{R}^n_{\geq 0}$, $\mathbb{R}^n_{\gneqq} := \mathbb{R}^n_{\gneqq 0}$ and likewise $\mathbb{R}^n_{\leq}$ and $\mathbb{R}^n_{\lneqq}$.

(iii) Consequently, we define for $x, y \in \bar{\mathbb{R}}^n$ with $x \leq y$:

$$(x, y) := \{z \in \mathbb{R}^n \mid x < z < y\},$$
$$[x, y] := \{z \in \mathbb{R}^n \mid x \leq z \leq y\}.$$

One sided intervals of this kind are defined accordingly.

1.2 Inequalities

The following two inequalities are cited from [4, p. 27].

Lemma 1.2 (Gronwall's inequality)**.** *For $T > 0$ let $v : [0, T] \to \mathbb{R}$ be a non-negative, differentiable function satisfying*

$$v'(t) \leq \varphi(t)v(t) + \chi(t) \quad \textit{for all } t \in [0, T],$$

where φ and χ are real-valued, non-negative, integrable functions on $[0,T]$. Then it holds

$$v(t) \leq \exp\left(\int_0^t \varphi(s)\,ds\right)\left(v(0) + \int_0^t \chi(s)\,ds\right) \quad \text{for all } t \in [0,T].$$

In particular, if $v \leq \varphi v$ in $[0,T]$ and $v(0) = 0$ hold, then $v = 0$ in $[0,T]$.

Lemma 1.3 (Young's inequality)**.** *For every $a, b \in \mathbb{R}$ and $\varepsilon > 0$ we have*

$$ab \leq \frac{\varepsilon a^2}{2} + \frac{b^2}{2\varepsilon}.$$

1.3 Abstract Evolution Equations

Let $(V, \langle\cdot,\cdot\rangle_V)$ and $(H, \langle\cdot,\cdot\rangle_H)$ be two real, separable Hilbert spaces, such that $V \subset H$ dense with compact embedding. Overall, we assume $V \subset H \subset V'$ to be a Gelfand triple. For more details about the Gelfand triple in the context of optimal control, see for example [5, p. 118]. Besides, let $T > 0$.

Definition and Remark 1.4. Let us define

$$W(0,T) := \{\varphi \in L^2(0,T;V) \mid \varphi_t \in L^2(0,T;V')\}$$

along with the inner product

$$\langle\varphi,\psi\rangle_{W(0,T)} := \int_0^T \langle\varphi(t),\psi(t)\rangle_V + \langle\varphi_t(t),\psi_t(t)\rangle_{V'}\,\mathrm{dt} \quad \text{for all } \varphi,\psi \in W(0,T)$$

Then, $\big(W(0,T), \langle\cdot,\cdot\rangle_{W(0,T)}\big)$ is a Hilbert space (see [6, p. 473]).

The following result is taken from [6, p. 473], where it was also proved.

Theorem 1.5. *Any $\varphi \in W(0,T)$ satisfies $\varphi \in C(0,T;H)$ and the embedding*

$$W(0,T) \hookrightarrow C(0,T;H)$$

is continuous, i.e. there is a constant $C_W > 0$, such that

$$\|\varphi\|_{C(0,T;H)} \leq C_W \|\varphi\|_{W(0,T)} \quad \text{for all } \varphi \in W(0,T).$$

Under appropriate assumptions, it is possible to show the well-posedness of abstract evolution problems such as the weak formulation of the state equation occurring in this thesis. In a more general form, the evolution equation reads

$$\begin{aligned} \langle y_t(t),\varphi\rangle_{V',V} + a(t;y(t),\varphi) &= \langle f(t),\varphi\rangle_{V',V} \quad \text{for all } \varphi \in V \\ \langle y(0),\varphi\rangle_H &= \langle y_0,\varphi\rangle_H \qquad \text{for all } \varphi \in H, \end{aligned} \tag{AEE}$$

which is supposed to hold almost everywhere (a.e.) in $(0,T)$. Additionally, $a(t;\cdot,\cdot) : V \times V \to \mathbb{R}$ is assumed to be a bilinear form for almost all (f.a.a.) $t \in (0,T)$. Moreover, we suppose $f : (0,T) \to V'$ and $y_0 \in H$. Here, $\langle\cdot,\cdot\rangle_{V',V}$ stands for the dual pairing between V and its dual space V'.

Remark 1.6. (i) For $y \in W(0,T)$ it holds

$$\langle y_t(t), \varphi \rangle_{V',V} = \frac{d}{dt} \langle y(t), \varphi \rangle_H \quad \text{for all } \varphi \in V.$$

(see [6, p. 477]).

(ii) The initial condition of (AEE) can be interpreted using the embedding $W(0,T) \hookrightarrow C(0,T;H)$ from Theorem 1.5.

More specifically, the well-posedness of (AEE) requires the following assumption.

Assumption 1. *Suppose that the bilinear form $a(t;\cdot,\cdot) : V \times V \to \mathbb{R}$ fulfils the following requirements:*

1. *The function $t \mapsto a(t,\cdot,\cdot)$ is measurable,*
2. *there is a constant $C_a > 0$ which does not depend on t such that*

$$| \, a(t;\varphi,\psi) \, | \leq C_a \, \|\varphi\|_V \, \|\psi\|_V \quad \text{for all } \varphi, \psi \in V,$$

3. *there are constants $\gamma > 0$ and $\eta \geq 0$ that do not depend on t such that $a(t;\cdot,\cdot)$ is coercive, that is*

$$a(t;\varphi,\varphi) \geq \gamma \, \|\varphi\|_V - \eta \, \|\varphi\|_H \quad \text{for all } \varphi, \psi \in V$$

holds.

Now, the following theorem, which is based on [6, pp. 512-513, Theorems 1 and 2], yields the desired result about the solvability of (AEE).

Theorem 1.7. *Let $a(t,\cdot,\cdot)$ be a bilinear form satisfying Assumption 1. Then, for any $f \in L^2(0,T;V')$ and $y_0 \in H$ there exists a unique solution $y \in W(0,T)$ of* (AEE) *which depends continuously on the data. More specifically,*

$$\|y\|_{W(0,T)} \leq C \left(\|f\|_{L^2(0,T;V')} + \|y_0\|_H \right) \tag{1.1}$$

holds for a constant $C > 0$ which does not depend on f and y_0.

Proof. A proof regarding the existence and uniqueness of the solution can be found in [6, pp. 512-520]. The inequality (1.1) can be shown by standard variational techniques and energy estimates similar to those in [6, pp. 520-521]. □

Chapter 2

Bicriterial Optimization by the Euclidean Reference Point Method

In order to deal with optimization problems that have more than one objective function, an appropriate optimality concept, namely the so-called Pareto optimality will be introduced. Using this concept, we will learn that a set of solutions has to be found instead of a single optimal solution. Similarly to [3], we introduce the concepts of multiobjective optimization problems only for the bicriterial case, since this is the relevant case for the applications in this thesis. With regard to the analysis of multiobjective optimization problems, we recall the definitions concerning the order on $\mathbb{R}^n$ from Chapter 1, which will be applied for the case $n = 2$. All results in this chapter are cited from [2, Chapter 3, pp. 13-37], where the general case, which also holds for more than two objectives, is covered. Likewise, the proofs to the results in this chapter can also be found there. However, we only state the results that are necessary to treat the specific optimal control problem in this thesis.

Let $(U, \langle\cdot,\cdot\rangle_U)$ be a real Hilbert space and $U_{\text{ad}} \subset U$ non-empty, convex and closed. In this chapter, we will treat the following *Bicriterial Optimization Problem*

$$\min_{u \in U_{\text{ad}}} f(u) \tag{BO}$$

with the function $f : U_{\text{ad}} \to \mathbb{R}^2, f(u) := (f_1(u), f_2(u))^T$.

Definition 2.1. We call

(1) the set $U_{\text{ad}} \subset U$ *admissible set* and a vector $u \in U_{\text{ad}}$ *admissible*,

(2) the image of the admissible set $Y := f(U_{\text{ad}}) \subset \mathbb{R}^2$ *objective admissible region* and a vector $y \in Y$ *objective vector*.

2.1 Optimality Concept

Since in most applications, there is a conflict between the objective functions, we cannot expect to minimize both f_1 and f_2 at the same time. That is why we introduce a more appropriate optimality concept, which will be used throughout this thesis, namely the Pareto optimality.

Definition 2.2. An admissible vector $\bar{u} \in U_{\text{ad}}$ is called *Pareto optimal* for (BO) if there is no $u \in U_{\text{ad}} \backslash \{\bar{u}\}$ with $f(u) \lneqq f(\bar{u})$. Besides, we call $P_U := \{u \in U_{\text{ad}} \mid u \text{ is Pareto optimal}\} \subset U$ the *Pareto set* and $P_Y := f(P_U) \subset \mathbb{R}^2$ the *Pareto front*.

Remark 2.3. (i) Taking into account the previous definition of optimality, solving the optimization problem (BO) essentially means to determine a set of Pareto optimal solutions, i.e. the Pareto set, which usually contains more than one element.

(ii) The purpose of calculating a set of Pareto optimal solutions is to present this set to a *decision maker* who can use his deeper insight of the problem to pick a solution which is the best solution according to his criteria.

For the analysis of the present multiobjective optimization problem, we define a lower bound of the Pareto front.

Definition 2.4. By

$$y^{\mathrm{id}} := \inf_{u \in U_{\mathrm{ad}}} \begin{pmatrix} f_1(u) \\ f_2(u) \end{pmatrix} := \begin{pmatrix} \inf_{u \in U_{\mathrm{ad}}} f_1(u) \\ \inf_{u \in U_{\mathrm{ad}}} f_2(u) \end{pmatrix}$$

we define the *ideal objective vector* of (BO).

2.2 Scalarization by the Euclidean Reference Point Method

There are many different approaches on how to compute Pareto optimal solutions. For example, several of these methods are covered in [1]. One approach that is often used, is to scalarize the multiobjective optimization problem, such that it can be solved by well-known optimization techniques for scalar problems. Usually solving one scalar optimization problem yields one Pareto optimal solution, so in order to approximate the Pareto set, a series of different scalar problems has to be solved. In this thesis, only the scalarization method called the Euclidean reference point method is introduced, since this method will be used for the later numerical experiments. Therefore we only state the results which are necessary to construct the numerical optimization algorithm in the last section of this chapter. All results of this section are based on [2, pp. 26-34].

Definition 2.5. We write $P_Y + \mathbb{R}^2_{\leq} := \{z + x \mid z \in P_Y \text{ and } x \in \mathbb{R}^2_{\leq}\}$. For $z \in P_Y + \mathbb{R}^2_{\leq}$ we define

$$F_z : U \to \mathbb{R}, \quad F_z(u) := \frac{1}{2} \left(f_1(u) - z_1\right)^2 + \frac{1}{2} \left(f_2(u) - z_2\right)^2 . \tag{2.1}$$

Then, we call

$$\min_{u \in U_{\mathrm{ad}}} F_z(u) \tag{RPP}$$

the *Euclidean reference point problem* for the reference point z.

Remark 2.6. Evidently, minimizing the scalar function F_z is equivalent to determining an objective vector $y \in Y$ that is closest to the reference point z with respect to the Euclidean distance.

Notation 2.7. For a reference point $z \in \mathbb{R}^2$, we denote by $(\mathrm{RPP})_z$ the corresponding Euclidean reference point problem.

2.2.1 Theoretical Results

In this section we formulate analytical results that serve as the foundation for the algorithm to compute the Pareto front at the end of this chapter.

Assumption 2. *Suppose that f_1 and f_2 are strictly convex, continuous and bounded from below. If U_{ad} is unbounded, we require additionally that $\lim_{\|u\|_U \to \infty} f_i(u) = \infty$ holds for $i \in \{1, 2\}$.*

The following result is a citation from [2] where it appeared as Theorem 3.35 along with its proof.

Theorem 2.8. *Let $z \in P_Y + \mathbb{R}^2_{\leq}$ be a reference point and let Assumption 2 be satisfied. Then, the Euclidean reference point problem* (RPP) *has a unique solution $\bar{u}_z \in U_{ad}$ which is Pareto optimal.*

We proceed by stating an important result, that ensures that all Pareto optimal solutions can be found by solving an Euclidean reference point problem. The following result is a citation from [2] where it appeared as Theorem 3.43 along with its proof.

Theorem 2.9. *Let Assumption 2 be satisfied. Then, for any $\bar{y} = f(\bar{u}) \in P_Y$, there exists $z \in P_Y + \mathbb{R}^2_{\lneq}$ such that $\bar{u}$ is the unique solution of* $(\text{RPP})_z$.

The following results show that the solution of (RPP) and its image are continuously dependent on the corresponding reference point.

Assumption 3. *Let the objectives f_1 and f_2 be twice continuously differentiable with positive definite second derivatives $\nabla^2 f_1$ and $\nabla^2 f_2$. Furthermore, suppose that there exists $i_p \in \{1, 2\}$ such that $\nabla^2 f_{i_p}$ is uniformly positive definite with coercivity constant C_{i_p}.*

The following result is taken from [2] where it appeared as Lemma 3.46 along with its proof.

Lemma 2.10. *Let Assumptions 2 and 3 be satisfied. We define the set $\mathcal{Z} := \{z \in P_Y + \mathbb{R}^2_{\leq} \mid z_{i_p} < y^{id}_{i_p} - \kappa\}$ for any $\kappa > 0$. Then, the mapping $\mathcal{Z} \to U_{ad}$, $z \mapsto \bar{u}_z$, where $\bar{u}_z$ is the unique minimizer of F_z, is Lipschitz continuous with respect to the constant $\frac{1}{2}\left(\kappa \cdot C_{i_p}\right)^{-\frac{1}{2}}$.*

The following result is a citation from [2] where it appeared as Theorem 3.47 along with its proof.

Theorem 2.11. *Let Assumption 2 be satisfied. Then, the mapping $P_Y + \mathbb{R}^2_{\leq} \to P_Y$, $z \mapsto f(\bar{u}_z)$, where $\bar{u}_z$ is the unique minimizer of F_z, is Lipschitz continuous with Lipschitz constant 1.*

The following result is taken from [2] where it appeared as Lemma 3.42 along with its proof.

Lemma 2.12. *Let Assumption 2 be satisfied and let $z \in P_Y + \mathbb{R}^2_{\leq}$ be arbitrary. Besides, let $\bar{u}$ be the unique minimizer of F_z. Then, $\bar{u}$ is also the unique minimizer of $F_{\tilde{z}}$ for each $\tilde{z} = f(\bar{u}) + \lambda(z - f(\bar{u}))$ with $\lambda \geq 0$, i.e. for all reference points being located on the ray that is starting in $f(\bar{u})$ and goes through z.*

The two preceding results imply a geometrical property of the Pareto front. The following result is a citation of [2, Corollary 3.52]. The corresponding proof can also be found there.

Corollary 2.13. *Let Assumption 2 be satisfied. Besides, let $z^1, z^2 \in P_Y + \mathbb{R}^2_{\leq}$ be arbitrary reference points and assume that $\bar{u}^1, \bar{u}^2 \in U_{ad}$ are the unique minimizers of F_{z^1} and F_{z^2}, respectively. Then it holds*

$$\left\| f(\bar{u}^1) - f(\bar{u}^2) \right\|_{\mathbb{R}^2} \leq dist(z^2, \mathcal{R}),$$

where we denote by $\mathcal{R} \subset \mathbb{R}^2$ the ray that starts in $f(\bar{u}^1)$ and goes through z^1.

2.2.2 A-Posteriori Error Estimate

We would also like to cite an error estimate regarding the reference point problem $(\mathrm{RPP})_z$, which serves as the basis for the error estimate that is used in the numerical considerations of this thesis. The following result is a citation from [2] where it appeared as Theorem 3.51 along with its proof.

Theorem 2.14. *Let Assumptions 2 and 3 be satisfied. Additionally, let us assume that the functions f_1, f_2 are quadratic, that is they fulfil*

$$f_i(u+h) = f_i(u) + \langle \nabla f_i(u), h\rangle_U + \frac{1}{2}\langle \nabla^2 f_i(u)h, h\rangle_U$$

for all $u \in U_{ad}$, $h \in U$ with $u + h \in U_{ad}$ and for $i \in \{1, 2\}$. Furthermore, let $z \in P_Y + \mathbb{R}^2_{\leqq}$ and $\bar{u} \in U_{ad}$ be the unique solution of $(\mathrm{RPP})_z$. Then, for any $u_p \in U_{ad}$ satisfying $f(u_p) \geq z$ and $f_{i_p}(\bar{u}) + f_{i_p}(u_p) > 2z_{i_p}$ we obtain the estimates

$$\|\bar{u} - u_p\|_U \leq \left[C_{i_p}\left(\frac{f_{i_p}(\bar{u}) + f_{i_p}(u_p)}{2} - z_{i_p}\right)\right]^{-1} \|\xi\|_U$$

and

$$\|f(\bar{u}) - f(u_p)\|_{\mathbb{R}^2} \leq \frac{1}{2}\left[C_{i_p}\left(\frac{f_{i_p}(\bar{u}) + f_{i_p}(u_p)}{2} - z_{i_p}\right)\right]^{-\frac{1}{2}} \|\xi\|_U ,$$

where $\xi \in U$ is determined such that

$$\langle \nabla F_z(u_p) + \xi, u - u_p\rangle_U \geq 0 \text{ for all } u \in U_{ad}$$

holds.

2.3 Algorithm to Compute the Pareto Front

In order to solve a bicriterial optimization problem (BO) numerically, we have to compute approximations of the Pareto set P_U and Pareto front P_Y, i.e. their discrete counterparts $\tilde{P}_U := \{\bar{u}^0, \bar{u}^1, \ldots, \bar{u}^{N+1}\} \subset P_U$ and $\tilde{P}_Y := f(\tilde{P}_U) \subset P_Y$. We already know by Theorem 2.9 that all of these points can be calculated by solving an Euclidean reference point problem for a suitable reference point $z \in P_Y + \mathbb{R}^2_{\leqq}$. Apart from that, Theorem 2.8 ensures the solvability of the reference point problems if the respective reference points are chosen appropriately. To this end, the reference points are constructed by an algorithm that will be explained in detail below. The procedure is taken from [2, pp. 34-37] and [7, p. 230], where a more elaborate description can be found.
The individual pareto points will be computed iteratively, whereby the first and the last optimal control have to be calculated separately since we do not know how the set $P_Y + \mathbb{R}^2_{\leqq}$ looks upfront and therefore we cannot choose an appropriate reference point from this set to compute the first Pareto optimal point. Additionally, its useful to have the last Pareto optimal point available as a stopping condition. Consequently, we calculate the first and the last Pareto optimal point from

$$\min_{u \in U_{\mathrm{ad}}} f_1(u) \tag{2.2}$$

i.e. minimizing the first objective and from

$$\min_{u \in U_{\text{ad}}} f_2(u). \tag{2.3}$$

i.e. minimizing the second objective, respectively. Indeed, $f(\bar{u}^0)$ and $f(\bar{u}^{N+1})$ can be viewed as the outer points of the Pareto front P_Y if the objective space is depicted by a two-dimensional plane. From this point of view the Pareto front P_Y takes a path from $f(\bar{u}^0)$ to $f(\bar{u}^{N+1})$. Now we would like to compute $z^1, \ldots, z^N$ sequentially, such that solving the associated Euclidean reference point problems yields $f(\bar{u}^i)$, $i = 1, \ldots, N$, that approximate P_Y ideally in an equidistant manner. Considering the theoretical results, Theorem 2.11 supports the idea that a good approximation of the Pareto front can be achieved by an appropriate choice of the reference points. In this case, the reference points will be constructed as follows: From Lemma 2.12 we conclude that $\delta^p := z^n - f(\bar{u}^n) \in \mathbb{R}^2$ is perpendicular to P_Y in $f(\bar{u}^n)$ and consequently $\delta^x := (-\delta_2^p, \delta_1^p)^\top \in \mathbb{R}^2$ is tangential to P_Y at the same point and is directed downwards with respect to P_Y. Using these vectors, a given reference point z^n and the associated Pareto optimum $\bar{u}^n$, we determine z^{N+1} by

$$z^{N+1} := f(\bar{u}^n) + h_x \cdot \frac{\delta^x}{\|\delta^x\|_{\mathbb{R}^2}} + h_p \cdot \frac{\delta^p}{\|\delta^p\|_{\mathbb{R}^2}} \quad \text{for } n = 2, \ldots, N \tag{2.4}$$

With this new reference point, the next solution $\bar{u}^{n+1}$ will be determined using the initial value $\bar{u}^n$, which is supported by the continuous dependency of the solutions on the reference points, see Lemma 2.10.
Moreover, the parameters h_x and h_p will be chosen to determine how far the next reference point is chosen in tangential direction with respect to P_Y and how far it is away from the Pareto front in perpendicular direction, respectively. Besides, since the calculation of the reference points is recursive, the first reference point has to be determined differently. Therefore, we set

$$z^1 := f(\bar{u}^0) - \begin{pmatrix} h_p \\ h_x \end{pmatrix}. \tag{2.5}$$

Regarding the approximation fineness, we conclude from Lemma 2.13:

$$\left\| f(\bar{u}^{n+1}) - f(\bar{u}^n) \right\|_{\mathbb{R}^2} \leq |h_x| \,.$$

This enables us to control the distance between consecutive Pareto optimal points. Using the above iteration of reference points, we will stop the iteration if z^{n+1} fulfils $z_1^{n+1} \geq f_1\left(\bar{u}^L\right)$, where $\bar{u}^L$ is the unique solution to (2.3) and the last Pareto optimal solution from the perspective of computing the Pareto front in a downwards direction like described above. To sum up, we formulate the procedure to compute the Pareto front as a pseudo code in Algorithm 1. This algorithm was already presented in [2, Algorithm 1] and [7, Algorithm 1], respectively.

Algorithm 1 Algorithm to compute the Pareto front

Require: Maximum number $N_{\max} \in \mathbb{N}$ of Pareto points, step size parameters $h_x, h_p > 0$;

Solve (2.2) and (2.3) in order to obtain $\bar{u}^0$ and $\bar{u}^L$;
Set $\tilde{P}_U = \{\bar{u}^0\}$ and $\tilde{P}_Y = \{f(\bar{u}^0)\}$;
Set $n = 0$ and compute z^1 by (2.5);
while $n < N_{\max}$ and $z_1^{n+1} < f_1\left(\bar{u}^L\right)$ **do**
    Set $n = n + 1$;
    Solve $(\mathrm{RPP})_{z^n}$ with initial point $\bar{u}^{(n-1)}$;
    Set $\tilde{P}_U = \tilde{P}_U \cup \{\bar{u}^n\}$ and $\tilde{P}_Y = \tilde{P}_Y \cup \{f(\bar{u}^n)\}$;
    Compute z^{n+1} by (2.4);
Return $\tilde{P}_U = \tilde{P}_U \cup \{\bar{u}^L\}$ and $\tilde{P}_Y = \tilde{P}_Y \cup \{f(\bar{u}^L)\}$

Chapter 3

Proper Orthogonal Decomposition for Abstract Evolution Equations

In this chapter, we will introduce a model order reduction technique for evolution equations of the form

$$\begin{aligned} \langle y_t(t), \varphi \rangle_{V',V} + a(t; y(t), \varphi) &= \langle f(t,u), \varphi \rangle_{V',V} \quad \text{for all } \varphi \in V \text{ a.e.} \\ \langle y(0), \varphi \rangle_H &= \langle y_0, \varphi \rangle_H \quad \text{for all } \varphi \in H \end{aligned} \tag{AEE}$$

where u is an element of a vector space U. The general idea is to consider the above problem on a suitable, low-dimensional subspace $V^\ell \subset V$. Oftentimes a finite element approach is used, which is a discretization of V that does not take into account any properties of the evolution problem. However, there is another dimension reduction technique called *proper orthogonal decomposition* (POD) that usually achieves a much lower dimension than the finite element method while maintaining sufficient accuracy. This is possible because, by construction, POD uses information about the solution of the evolution problem to generate the lower dimensional subspace V^ℓ, which then captures the most dominant characteristics of the problem. Having said this, since in a numerical application POD usually requires a finite element solution first, using POD can only be advantageous if the evolution equation has to be solved repeatedly for different instances of u. This is typically the case for optimal control problems, that have such an evolution equation as a constraint.
First, we formulate the continuous version of POD, which will be needed for the later analysis of an optimal control problem. Besides, we also need a discrete version of POD, which is suited for the inherently discrete data of numeric computations, so we will cover this in the second part of this chapter. This chapter is based on [4, Section 1.2], so the proofs of the cited results can be found there.

3.1 Continuous POD

Let $(V, \langle \cdot, \cdot \rangle_V)$ and $(H, \langle \cdot, \cdot \rangle_H)$ be real, separable Hilbert spaces and assume that V is dense in H by compact embedding. For fixed $\wp \in \mathbb{N}$ let $y^1, \dots, y^\wp \in L^2(0,T;V)$ be given. Besides, we define the subspace

$$\mathcal{V} := \operatorname{span}\{y^k(t) \mid 1 \le k \le \wp, t \in [0,T]\} \subset V.$$

Definition 3.1. In the context of POD, we call $y^1, \dots, y^\wp \in L^2(0,T;V)$ *snapshots* and the associated set $\mathcal{V}$ *snapshot subspace*. Furthermore, we declare $d := \dim(\mathcal{V}) \in \mathbb{N} \cup \{\infty\}$.

Remark 3.2. Typically, the snapshots $y^1, \dots, y^\wp \in L^2(0,T;V)$ are solutions or derivatives of solutions of the investigated evolution problem.

Next, we declare the central terms of the POD method.

Definition 3.3. Let the snapshots $y^1, \dots, y^\wp \in L^2(0,T;V)$ be given. We call the solution of the optimization problem

$$\begin{aligned} \min_{\psi_1,\dots,\psi_l \in V} \sum_{k=1}^{\wp} \int_0^T \left\| y^k(t) - \sum_{i=1}^{\ell} \langle y^k(t), \psi_i \rangle_V \psi_i \right\|_V^2 dt \\ \text{subject to (s.t.) } \langle \psi_i, \psi_j \rangle_V = \delta_{ij}, \quad 1 \le i,j \le \ell, \end{aligned} \tag{$CPOD^\ell$}$$

POD basis $\{\bar{\psi}_i\}_{i=1}^\ell$ of rank ℓ for any $\ell \in \{1,\dots,d\}$. The resulting vector space $V^\ell := \operatorname{span}\{\bar{\psi}_1, \dots, \bar{\psi}_\ell\}$ is called *POD space* of rank ℓ.

Remark 3.4. According to the previous definition, a POD basis $\{\bar{\psi}_i\}_{i=1}^\ell$ minimizes the sum of the mean square errors between the snapshots y^k and their projections onto V^ℓ.

Now, we introduce one possible approach to solve $(CPOD^\ell)$.
The following result is taken from [4] where it appeared as Lemma 1.13 along with its proof.

Definition and Theorem 3.5. Let $y^1, \dots, y^\wp \in L^2(0,T;V)$ be snapshots. Then, the operator

$$\mathcal{R} : V \to \mathcal{V}, \quad \mathcal{R}\psi = \sum_{k=1}^{\wp} \int_0^T \langle y^k(t), \psi \rangle_V y^k(t)\, dt. \tag{3.1}$$

is linear, compact, non-negative and self-adjoint.

The following result provides crucial insight on how to solve the optimization problem $(CPOD^\ell)$. Evidently, this can be achieved by solving an eigenvalue problem.
The following result is a citation of [4, Theorem 1.15]. The corresponding proof can also be found there.

Theorem 3.6. *Let $y^1, \dots, y^\wp \in L^2(0,T;V)$ be snapshots. Let the operator $\mathcal{R}$ be defined by* (3.1). *Then, there exist non-negative eigenvalues $\{\bar{\lambda}_i\}_{i\in\mathbb{N}}$ and associated orthonormal eigenfunctions $\{\bar{\psi}_i\}_{i\in\mathbb{N}}$ satisfying*

$$\mathcal{R}\bar{\psi}_i = \bar{\lambda}_i \bar{\psi}_i, \quad \bar{\lambda}_1 \ge \dots \ge \bar{\lambda}_d > \bar{\lambda}_{d+1} = \dots = 0. \tag{3.2}$$

For every $\ell \in \{1,\dots,d\}$ the first ℓ eigenfunctions $\{\bar{\psi}_i\}_{i=1}^\ell$ solve $(CPOD^\ell)$. Besides, these functions fulfil

$$\sum_{k=1}^{\wp} \int_0^T \left\| y^k(t) - \sum_{i=1}^{\ell} \langle y^k(t), \bar{\psi}_i \rangle_V \bar{\psi}_i \right\|_V^2 dt = \sum_{i=\ell+1}^{d} \bar{\lambda}_i. \tag{3.3}$$

Remark 3.7. A closer look at (3.3) reveals, that the accuracy of the POD approximation depends on the chosen number of POD basis functions and on the decay rate of the eigenvalues $\{\bar{\lambda}_i\}_{i\in\mathbb{N}}$.

3.2 Discrete POD

In numerical applications the continuous version of POD is usually not applicable since realistically we do not have functions $y^1, \dots, y^\wp$ that can be evaluated everywhere on $[0,T]$, but only discrete approximations thereof. Fortunately, there is also a discrete version of POD available (see [4, Section 1.2.1]), which we will summarize here. Throughout this chapter, let $m \in \mathbb{N}$.
A common scenario is, that the discrete snapshots $y^1, \dots, y^\wp : \{t_1, \dots, t_n\} \to \mathbb{R}^m$ are finite element solutions of the treated evolution equation using different inputs. We proceed in the same way as in the continuous case.

Definition 3.8. In the context of discrete POD, we call the vectors $y_1^k, \dots, y_n^k \in \mathbb{R}^m$ for $1 \leq k \leq \wp$ *discrete snapshots* and define a *discrete snapshot space* $\mathcal{V}^n$ by

$$\mathcal{V}^n := \operatorname{span}\{y_j^k \mid 1 \leq k \leq \wp, 1 \leq j \leq n\}. \tag{3.4}$$

Additionally, let $d^n \in \{1, \dots, n\wp\}$ be the dimension of this space.

Like in the continuous case, we are looking for a suitable low-dimensional subspace V^ℓ of the snapshot space $\mathcal{V}^n$.

Definition 3.9. For $1 \leq k \leq \wp$ let $y_1^k, \dots, y_n^k \in \mathbb{R}^m$ be discrete snapshots and let $\alpha_1^n, \dots, \alpha_n^n > 0$ be positive weighting parameters. Furthermore, let $W \in \mathbb{R}^{m \times m}$ be a symmetric and positive definite matrix, such that $\langle \cdot, \cdot \rangle_W := \langle \cdot, W \cdot \rangle_{\mathbb{R}^m}$ is an inner product on $\mathbb{R}^m$. Then, for $\ell \in \{1, \dots, d^n\}$ we call the solution $\{\bar{\psi}_i\}_{i=1}^\ell$ of the optimization problem

$$\begin{aligned} \min_{\psi_1, \dots, \psi_\ell \in \mathbb{R}^m} \sum_{k=1}^{\wp} \sum_{j=1}^{n} \alpha_j^n \left\| y_j^k - \sum_{i=1}^{\ell} \langle y_j^k, \psi_i \rangle_W \psi_i \right\|_W^2 \\ \text{s.t. } \langle \psi_i, \psi_j \rangle_W = \delta_{ij}, \quad 1 \leq i, j \leq \ell. \end{aligned} \tag{$DPOD^\ell$}$$

discrete POD basis of rank ℓ. The resulting linear space $V^\ell := \operatorname{span}\{\bar{\psi}_1, \dots, \bar{\psi}_\ell\}$ is called *discrete POD space* of rank ℓ.

Remark 3.10. (i) Comparing the previous definition to the corresponding definition in the continuous case, we observe that the weights $\alpha_1^n, \dots, \alpha_n^n$ achieve some kind of numerical integration of the integral in the continuous case. A straightforward choice are the trapezoidal weights:

$$\alpha_1^n = \frac{T}{2(n-1)}, \quad \alpha_j^n = \frac{T}{n-1} \text{ for } 2 \leq j \leq n-1, \quad \alpha_n^n = \frac{T}{2(n-1)}.$$

(ii) A weighted inner product $\langle \cdot, \cdot \rangle_W$ is used for modeling the inner product on the finite element space. Specifically, in the numerical part of this thesis, we choose $W = M + A$, where M is the mass matrix and W is the stiffness matrix corresponding to the present evolution problem, respectively.

The procedure to solve the minimization problem is again analogous to the continuous case. The following result is a citation from [4] where it appeared as Lemma 1.3 along with its proof.

Definition and Theorem 3.11. Let $y_1^k, \dots, y_n^k \in \mathbb{R}^m$ be discrete snapshots for $1 \le k \le \wp$. We define the operator

$$\mathcal{R}^n : \mathbb{R}^m \to \mathcal{V}^n, \quad \mathcal{R}^n \psi = \sum_{k=1}^{\wp} \sum_{j=1}^{n} \alpha_j^n \langle y_j^k, \psi \rangle_W y_j^k. \tag{3.5}$$

Then, $\mathcal{R}^n$ is linear, compact, non-negative and self-adjoint.

The following result is based on Theorem 1.6 and Theorem 1.8 from [4].

Theorem 3.12. *For $1 \le k \le \wp$ let $y_1^k, \dots, y_n^k \in \mathbb{R}^m$ be discrete snapshots. Furthermore, let $\mathcal{R}^n$ be defined by (3.5). Then, there exist non-negative eigenvalues $\{\bar{\lambda}_i^n\}_{i=1}^m$ and associated orthonormal eigenvectors $\{\bar{\psi}_i^n\}_{i=1}^m$ satisfying*

$$\mathcal{R}^n \bar{\psi}_i^n = \bar{\lambda}_i^n \bar{\psi}_i^n, \quad \bar{\lambda}_1^n \ge \dots \ge \bar{\lambda}_{d^n}^n > \bar{\lambda}_{d^n+1}^n = \dots = 0. \tag{3.6}$$

Besides, for every $\ell \in \{1, \dots, d^n\}$ the first ℓ eigenvectors $\{\bar{\psi}_i^n\}_{i=1}^{\ell}$ solve $(DPOD^\ell)$. Moreover, we have

$$\sum_{k=1}^{\wp} \sum_{j=1}^{n} \alpha_j^n \left\| y_j^k - \sum_{i=1}^{\ell} \langle y_j^k, \bar{\psi}_i^n \rangle_W \bar{\psi}_i^n \right\|_W^2 = \sum_{i=\ell+1}^{d^n} \bar{\lambda}_i^n. \tag{3.7}$$

Remark 3.13. There are a few different approaches to determine a POD basis $\{\bar{\psi}_i^n\}_{i=1}^{\ell}$ of rank ℓ in a numerical context (cf. [4, pp. 10-13]). We only mention the method which is used later in the numerical experiments of this thesis, which is the so-called *method of snapshots*. In particular, we consider the case $\wp = 2$, which will be the standard choice for the numerical experiments.

For $1 \le k \le 2$ let $y_1^k, \dots, y_n^k \in \mathbb{R}^m$ be discrete snapshots. We define the matrix $Y := [y_1^1 \mid \dots \mid y_n^1 \mid y_1^2 \mid \dots \mid y_n^2] \in \mathbb{R}^{m \times 2n}$ with rank d^n as well as the diagonal matrix $D := \operatorname{diag}(\alpha_1^n, \dots, \alpha_n^n) \in \mathbb{R}^{n \times n}$. Besides, let $\hat{Y} := W^{1/2} Y \tilde{D}^{1/2} \in \mathbb{R}^{m \times 2n}$.
Then, we solve the symmetric $2n \times 2n$ eigenvalue problem

$$\hat{Y}^\top \hat{Y} \varphi_i^n = \bar{\lambda}_i^n \varphi_i^n, \quad \text{with } \langle \varphi_i^n, \varphi_j^n \rangle_{\mathbb{R}^{2n}} = \delta_{ij}, \quad 1 \le i, j \le \ell$$

for the ℓ largest eigenvalues $\bar{\lambda}_1^n \ge \dots \ge \bar{\lambda}_\ell^n > 0$. Based on this, we compute the basis functions

$$\bar{\psi}_i^n = W^{-1/2} \psi_i^n = \frac{1}{\sqrt{\bar{\lambda}_i^n}} W^{-1/2} \hat{Y} \varphi_i^n = \frac{1}{\sqrt{\bar{\lambda}_i^n}} Y \tilde{D}^{1/2} \varphi_i^n$$

for $1 \le i \le \ell$. Note that since we have $\hat{Y}^\top \hat{Y} = \tilde{D}^{1/2} Y^\top W Y \tilde{D}^{1/2}$, the numerically expensive computation of $W^{1/2}$ and $W^{-1/2}$ is not needed. Therefore, this method is espacially favourable for $2n \le m$. However, we should keep in mind that this approach might be numerically unstable since computing the product $\hat{Y}^\top \hat{Y}$ increases the condition number compared to $\hat{Y}$.

Chapter 4

Bicriterial Optimal Control of Evolution Equations

In this chapter we will show how a bicriterial optimal control problem, which includes an evolution equation of the form (AEE) as the state equation, can be treated. In an exemplary manner, the procedure is demonstrated for one specific optimal control problem that serves as a model for a heating, ventilation and air conditioning (HVAC) process. As an application of the theory presented in Chapter 2 there will be two objectives that take into account the heating costs and the achievement of a desired temperature, respectively. More specifically, we start by formulating this problem in detail. The current chapter is largely based on Chapter 5 of [2], while the adaptions for the time-dependent convection term are taken from Chapter 5 of [3]. More precise sources will be stated when the respective results are formulated.

4.1 The Bicriterial Optimal Control Problem

We specify a bicriterial optimal control problem by

$$\min J(u,y) = \begin{pmatrix} \frac{1}{2}\|y - y_Q\|^2_{L^2(0,T;L^2(\Omega))} \\ \frac{1}{2}\|u\|^2_{L^2(0,T;\mathbb{R}^m)} \end{pmatrix} \tag{OCP}$$

subject to (s.t.)

$$\begin{aligned} y_t(t,x) - \kappa\Delta y(t,x) + \beta(t,x)\cdot\nabla y(t,x) &= \textstyle\sum_{i=1}^m u_i(t)\chi_i(x) && \text{for } (t,x) \in (0,T)\times\Omega \\ \tfrac{\partial y}{\partial \eta}(t,x) + \alpha_j y(t,x) &= \alpha_j y_a^j(t) && \text{for } (t,x) \in (0,T)\times\Gamma_j \\ y(0,x) &= y_0(x) && \text{for } x \in \Omega \end{aligned} \tag{STEQ}$$

for all $j \in \{1,\dots,s\}$ and

$$u_a(t) \le u(t) \le u_b(t) \quad \text{f.a.a. } t \in [0,T]. \tag{BCON}$$

The first constraint (STEQ), called the *state equation* takes the form of a partial differential equation, namely a heat equation with a potentially time-dependent convection function b for the state y, which is supposed to describe the temperature distribution in the room. Specifically, the heating process is influenced by the control u on the right-hand side of the state equation. We assume m heaters in the room, whose locations are described by the indicator functions $\chi_1,\dots,\chi_m$. The respective heating strategies $u_1,\dots,u_m$ are space-independent. The Robin boundary conditions of the state equation are supposed to model the temperature exchange with the world outside of the room Ω. Additionally, we impose the bilateral constraints (BCON) on the control u. For deeper insights on the whole model, we refer to the numerical

experiments in Chapter 5, where different variations of the above optimal control problem are investigated numerically.

For further analysis of the stated optimal control problem, we establish additional assumptions.

Assumption 4.
- *For an $n \in \mathbb{N}$ with $n \geq 2$, let $\Omega \subset \mathbb{R}^n$ be bounded with a C^1-boundary $\Gamma = \partial\Omega$, which consists of the disjoint subsets $\Gamma_1, \dots, \Gamma_s$, where $s \in \mathbb{N}$. That is $\bigcup_{i=1}^{s} \Gamma_i = \Gamma$.*
- *We require $\alpha_i \geq 0$ and $y_a^i \in L^2(0,T)$, $i \in \{1, \dots, s\}$*
- *For the convection term β we assume $\beta \in L^\infty(0,T; L^\infty(\Omega; \mathbb{R}^n))$.*
- *Let $W_1, \dots, W_m \subset \Omega$ be disjoint, measurable sets with measure greater than zero such that the characteristic functions $\chi_i := \chi_{W_i}$ fulfil $\chi_i \in L^2(\Omega)$ for $i \in \{1, \dots, m\}$.*
- *Set $U := L^2(0,T; \mathbb{R}^m)$ as the control space.*
- *Let $\kappa > 0$, $y_0 \in L^2(\Omega)$, $u_a, u_b \in L^\infty(0,T; \mathbb{R}^m)$*

Remark 4.1. A slight difference to the corresponding assumptions in [3] is, that we allow for different outside temperatures y_a^i at different parts Γ_i of the boundary for $i \in \{1, \dots, s\}$. This adaption is required for the application in the numerical experiments.

For the remaining part of this chapter, we require that Assumption 4 is satisfied.

4.2 The Reduced Bicriterial Optimal Control Problem

With the goal of formulating an objective function that only depends on the control u, we investigate the solvability of the state equation (STEQ) given Assumption 4. Indeed, this aspect was already covered in Section 5.1 of [2] for a time-independent convection term and adapted in Section 5.2 of [3] for the time-dependent case, so we will omit the derivation process and only quote the relevant results from these sources.
The weak formulation of STEQ takes the following form (cf. [3, Definition and Remark 5.1]):

Definition and Remark 4.2. Setting $V := H^1(\Omega)$, $H := L^2(\Omega)$, $W(0,T) := \{\varphi \in L^2(0,T;V) \mid \varphi_t \in L^2(0,T;V')\}$ and defining the mappings $a : (0,T) \times V \times V \to \mathbb{R}$ and $f : (0,T) \to V'$ by

$$a(t,\varphi,\psi) := \kappa \int_\Omega \nabla\varphi(x) \cdot \nabla\psi(x)\,dx + \int_\Omega \beta(t,x) \cdot \nabla\varphi(x)\psi(x)\,dx + \kappa \sum_{i=1}^{s} \alpha_i \int_{\Gamma_i} \varphi(x)\psi(x)\,dS(x)$$

and

$$\langle f(t), \varphi\rangle_{V',V} := \sum_{i=1}^{m} \int_\Omega u_i(t)\chi_i(x)\varphi(x)\,dx + \kappa \sum_{i=1}^{s} \alpha_i \int_{\Gamma_i} \varphi(x) y_a^i(t)\,dS(x),$$

we find a weak representation of (STEQ):

$$\begin{aligned} \langle y_t(t), \varphi\rangle_{V',V} + a(t; y(t), \varphi) &= \langle f(t), \varphi\rangle_{V',V} \quad \text{for all } \varphi \in V \text{ a.e.} \\ \langle y(0), \varphi\rangle_H &= \langle y_0, \varphi\rangle_H \qquad \text{for all } \varphi \in H \end{aligned} \tag{WS}$$

where a function $y \in W(0,T)$ is supposed to be determined. The problem (WS) is called the *weak formulation* of (STEQ).

The above definitions of the spaces V and H shall be maintained throughout this chapter.
It is well-known that under appropriate conditions, an equation of the form (WS) has a unique solution (cf. Theorem 1.7, which is cited from [6]). Indeed, it has already been verified that the required conditions hold given assumption 4, see [2] for the case of a time-independent convection term and [3] for the case of a time-dependent convection term. Consequently, we get the following result, which is taken from [3] where it appeared as Theorem 5.5. Note that the adaption in the affected proofs due to the change mentioned in Remark 4.1 is minor and therefore not carried out explicitly here.

Theorem 4.3. *Let Assumption 4 be satisfied. Then, there exists a unique solution $y \in W(0,T)$ of the weak formulation* (WS)*. Moreover, there is a constant $C > 0$ such that the a-priori estimate*

$$\|y\|_{W(0,T)} \leq C \left(\|f\|_{L^2(0,T;V')} + \|y_0\|_H \right) \tag{4.1}$$

holds.

The previous theorem motivates the following definition.

Definition 4.4. We define the linear continuous operator

$$\mathcal{T} : L^2(0,T;V') \times H \to W(0,T), \quad \mathcal{T}(f, y_0) := y,$$

which maps a right-hand side f and an initial value y_0 to the unique solution y of (WS).

The next definition is made to analyse how the solution of (WS) depends on the control $u \in U$.

Definition and Remark 4.5. Let $f(t) = \mathcal{F}(t) + (\mathcal{B}u)(t)$, where

$$\mathcal{F} : (0,T) \to V', \quad \langle \mathcal{F}(t), \varphi \rangle_{V',V} := \kappa \sum_{i=1}^{s} \alpha_i \int_{\Gamma_i} \varphi(x) y_a^i(t) \, dS(x)$$

and

$$\mathcal{B} : U \to L^2(0,T;V'), \quad \langle (\mathcal{B}u)(t), \varphi \rangle_{V',V} := \sum_{i=1}^{m} \int_{\Omega} u_i(t) \chi_i(x) \varphi(x) \, dx.$$

for $\varphi \in V$. Thus, we have split the right-hand side f of the state equation (WS) into two parts such that one of them depends on u and the other part does not.

The following result is a citation of [3, Lemma 5.8]. The corresponding proof can also be found there.

Lemma 4.6. *The operator $\mathcal{B}$ depends linearly on the control $u \in U$ and satisfies*

$$\|\mathcal{B}u\|_{L^2(0,T;V')} \leq \|\chi\|_{L^2(\Omega;\mathbb{R}^m)} \|u\|_U . \tag{4.2}$$

Definition and Remark 4.7. Let $y \in W(0,T)$ be the unique solution to (WS) for a $u \in U$. We call $\hat{y} := \mathcal{T}(\mathcal{F}, y_0)$ the *u-independent part* of the solution. We define the linear operator

$$\mathcal{S} : U \to W(0,T) \hookrightarrow L^2(0,T;H), \quad \mathcal{S}u := \mathcal{T}(\mathcal{B}u, 0).$$

Accordingly, we call $\mathcal{S}u$ the *u-dependent part* of the solution. Thus, we obtain $y = \hat{y} + \mathcal{S}u$. Here, the embedding $W(0,T) \hookrightarrow L^2(0,T;H)$ follows by applying Theorem 1.5 since we have $C(0,T;H) \subset L^2(0,T;H)$.

The following result is taken from [2] where it appeared as Lemma 5.6 and Lemma 5.7 along with the corresponding proofs.

Lemma 4.8. *The solution operator $\mathcal{S}$ is injective, continuous and satisfies*

$$\|\mathcal{S}\|_{L(U;L^2(0,T;H))} \leq C \|\chi\|_{L^2(\Omega;\mathbb{R}^m)},$$

where $C > 0$ is the same constant as in (4.1).

Proof. Although the convection term was time-independent in [2], the solution operators $\mathcal{S}$ and $\mathcal{T}$ have the same relevant properties in the present case. So the proof is exactly the same as in the cited source. □

Making use of the well-posedness of the state equation, we can express (OCP) as an equivalent optimal control problem only depending on the control u.

Definition 4.9. We set $U_{\text{ad}} := \{u \in U \mid u_a(t) \leq u(t) \leq u_b(t) \text{ f.a.a. } t \in [0,T]\}$ as the admissible set and define the reduced objective function

$$\hat{J} : U \to \mathbb{R}^2, \quad \hat{J}(u) := \begin{pmatrix} \hat{J}_1(u) \\ \hat{J}_2(u) \end{pmatrix} := \begin{pmatrix} \frac{1}{2}\|\hat{y} + \mathcal{S}u - y_Q\|^2_{L^2(0,T;H)} \\ \frac{1}{2}\|u\|^2_U \end{pmatrix}.$$

Then, we call

$$\min_{u \in U_{\text{ad}}} \hat{J}(u) \tag{ROCP}$$

the *reduced bicriterial optimal control.*

Taking into account the properties of the solution operator $\mathcal{S}$, it is evident that solving (ROCP) is equivalent to solving the optimization problem (OCP) from which it originates.
Apart from that, if we recall the situation of Chapter 2, we find that the reduced optimal control problem (ROCP) fits into the framework of the then defined bicriterial optimization problem. Moreover, it is possible to verify the main assumptions on the objective functions $\hat{J}_1$, $\hat{J}_2$ and the admissible set U_{ad} that were made in Chapter 2. This has already been done in [2, pp. 53-55], so we only cite those results here.
The following result is a citation of [2, Lemma 5.9]. The corresponding proof can also be found there.

Lemma 4.10. *The set U_{ad} is non-empty, closed and convex.*

The following result summarizes Lemma 5.10, Lemma 5.11 and Corollary 5.12 from [2], where they were also proved.

Lemma 4.11. *The functions $\hat{J}_1$ and $\hat{J}_2$ are strictly convex, continuous, bounded from below and quadratic as well as twice continuously Fréchet differentiable. Moreover, the second derivatives $\nabla^2\hat{J}_1$, $\nabla^2\hat{J}_2$ are positive definite and $\nabla^2\hat{J}_2$ is uniformly positive definite with coercivity constant $C_{J_2} = 1$.*

Hereby, we have verified all required assumptions such that the relevant results from Chapter 2 can be applied to the present problem (ROCP). Accordingly, we define the Euclidean reference point problem corresponding to (ROCP).

Definition 4.12. For a reference point $z \in \mathbb{R}^2$ we define the Euclidean reference point problem by

$$\min_{u\in U_{\text{ad}}} \frac{1}{2}\left(\hat{J}_1(u) - z_1\right)^2 + \frac{1}{2}\left(\hat{J}_2(u) - z_2\right)^2.$$

Again, we utilize the function $F_z : U \to \mathbb{R},\ F_z(u) := \frac{1}{2}\left(\hat{J}_1(u) - z_1\right)^2 + \frac{1}{2}\left(\hat{J}_2(u) - z_2\right)^2$. In accordance with Chapter 2, we denote this problem by $(\text{RPP})_z$.

4.3 Adjoint Equation

Numerical algorithms that solve scalar optimization problems often make use of the gradient and even the second derivative of the objective function. So, in view of the eventual numerical experiments, we would like to have a representation of $\nabla\hat{J}_1(u)$ and $\nabla^2\hat{J}_1(u)h$, respectively, that is numerically evaluable in an efficient manner for given $u \in U_{ad}$ and $h \in U$. With regard to optimal control problems, this can often be achieved by introducing an adjoint equation. It turns out that this approach also works for the present optimal control problem (ROCP). Thus, we proceed by introducing the adjoint equation. The results in this section are based on those from [2, Section 5.4], whereas the adaptions for the time-dependent bilinear form a are taken from [3, Section 5.4]. A detailed derivation, why the adjoint equation helps to express the derivatives of the objective function in the present case can be found in [2].

Definition 4.13. Corresponding to a differential equation of the form (WS), the objective function $\hat{J}_1$ and a control $u \in U_{\text{ad}}$, we call the end value problem

$$\begin{aligned} -\frac{d}{dt}\langle p(t), \varphi\rangle_{V',V} + a(t;\varphi,p(t)) &= \langle y_Q(t) - \hat{y}(t) - \mathcal{S}u(t), \varphi\rangle_{V',V} \quad \text{for all } \varphi \in V \text{ a.e.} \\ \langle p(T), \varphi\rangle_H &= \langle 0, \varphi\rangle_H \qquad\qquad\qquad\quad \text{for all } \varphi \in H \end{aligned} \tag{WA}$$

the adjoint equation.

The following result is taken from [3] where it appeared as Theorem 5.19 along with its proof.

Theorem 4.14. *For every $u \in U$ there exists a unique solution $p \in W(0,T)$ of the adjoint equation* (WA) *which satisfies*

$$\|p\|_{W(0,T)} \le C\,\|y_Q - \hat{y}(t) - \mathcal{S}u\|_{L^2(0,T;V')}$$

for a constant $C > 0$.

Definition and Remark 4.15. As for the state equation, we can use the previous theorem to define the linear and continuous operator $\tilde{\mathcal{T}} : L^2(0,T;V') \to W(0,T)$ which maps any right-hand side in (WA) to the solution of the equation.
Then, analogously to Definition and Remark 4.5, we split the solution of (WA) in two parts, such that one part depends on the control u and the other one does not. In particular, we define $\mathcal{A} : U \to W(0,T) \hookrightarrow L^2(0,T;H)$, $\mathcal{A}(u) := \tilde{\mathcal{T}}(-\mathcal{S}u)$ and $\hat{p} := \tilde{\mathcal{T}}(y_Q - \hat{y})$. Here, we can infer that $\mathcal{A}$ is linear and continuous since both $\mathcal{S}$ and $\tilde{\mathcal{T}}$ are linear and continuous. Consequently, it follows $\tilde{\mathcal{T}}(y_Q - \hat{y} - \mathcal{S}u) = \hat{p} + \mathcal{A}u$ by applying linearity.

Finally, we achieve the desired expression of the derivatives of $\hat{J}_1$ by using the solution of the adjoint equation in the next lemma. The corresponding result for an adjoint equation with a time-independent bilinear form $a(\cdot,\cdot)$ appeared as Corollary 5.19 in [2].

Lemma 4.16. *Let $u \in U$ and $\hat{p} + \mathcal{A}u$ be the corresponding solution of the adjoint equation* (WA)*. Then, the derivates of $\hat{J}_1$ take the following form.*

$$\nabla \hat{J}_1(u) = -\mathcal{B}^*(\hat{p} + \mathcal{A}u),$$

$$\nabla^2 \hat{J}_1(u) = -\mathcal{B}^*\mathcal{A}.$$

Proof. The proof can be done exactly in the same way as for the time-independent bilinear form a in the cited source. □

We also have a result available, showing how $\mathcal{B}^*$ can be evaluated. The following result is a citation from [2] where it appeared as Lemma 5.20 along with its proof.

Lemma 4.17. *Let the operator $\mathcal{B}$ be given as in Definition and Remark 4.5. Then, the following expression of the adjoint operator $\mathcal{B}^*$ holds.*

$$\mathcal{B}^* : L^2(0,T;V) \to U, \quad \mathcal{B}^* v(t) = \begin{pmatrix} \int_\Omega \chi_1(x) v(t,x)\,dx \\ \vdots \\ \int_\Omega \chi_m(x) v(t,x)\,dx \end{pmatrix}.$$

4.4 The POD Approximated Model

Having shown that the present optimal control problem (ROCP) can be treated with the techniques presented in Chapter 2, particularly with Algorithm 1, we would still like to take a closer look at how to apply POD to (ROCP). That is because, when using Algorithm 1 in practice, we have to solve both the state and the adjoint equation many times (cf. Chapter 5). So a reduction of the problem dimension could potentially save a lot of computation time. Unless stated otherwise, the results in this section are taken from [2] and the respective proofs can also be found there. The results in this section are based on those from [2, Section 5.5], whereas the adaptions for the time-dependent convection term are taken from [3, Section 5.5].

4.4.1 Reduced-Order State Equation

First, let us assume that $\hat{y}$, that is the part of the solution to the weak formulation (WS) which does not depend on u has already been computed and that a POD space $V^\ell \subset V$ of rank ℓ is given.

Definition 4.18. For $u \in U$ we introduce a reduced-order model of the u-dependent part of (WS) which is defined by

$$\begin{aligned} \langle y_t^\ell(t), \varphi\rangle_{(V^\ell)',V^\ell} + a(t; y^\ell(t), \varphi) &= \langle \mathcal{B}u(t), \varphi\rangle_{(V^\ell)',V^\ell} \quad \text{for all } \varphi \in V^\ell \text{ a.e.} \\ \langle y^\ell(0), \varphi\rangle_H &= \langle 0, \varphi\rangle_H \qquad\qquad \text{for all } \varphi \in H. \end{aligned} \tag{WS$^\ell$}$$

The following result already appeared as Theorem 5.24 in [3].

Theorem 4.19. *Suppose Assumption 4 is satisfied. Then, there exists a unique solution $y^\ell \in H^1(0,T;V^\ell)$ of the reduced-order model (WS$^\ell$) which obeys the estimate*

$$\left\|y^\ell\right\|_{H^1(0,T;V)} \leq C \left\|\mathcal{B}u\right\|_{L^2(0,T;(V^\ell)')} \tag{4.3}$$

for a constant $C > 0$ that is independent of ℓ.

Proof. For a proof of a similar version of this theorem, we refer to [8, pp. 354-356]. □

Remark 4.20. By making a Galerkin ansatz with the computed POD basis, we can write (WS$^\ell$) as a linear system of ordinary differential equations, that looks very much like a system that arises from the discretization by the finite element method.
Consequently, we can proceed with the time discretization analogously to the finite element system, e.g. by the well-known Crank-Nicolson method.

Definition and Remark 4.21. As for the full-order state equation we define the solution operator

$$\mathcal{S}^\ell : U \to H^1(0,T;V^\ell) \hookrightarrow L^2(0,T;H), \quad \mathcal{S}^\ell(u) := y^\ell,$$

which yields the unique solution y^ℓ of (WS$^\ell$) for any $u \in U$. Then, by linearity $\tilde{y}^\ell := \hat{y} + \mathcal{S}^\ell u$ fulfils

$$\begin{aligned} \langle \tilde{y}_t^\ell(t), \varphi\rangle_{(V^\ell)',V^\ell} + a(t; \tilde{y}^\ell(t), \varphi) &= \langle \mathcal{F}(t) + \mathcal{B}u(t), \varphi\rangle_{(V^\ell)',V^\ell} \quad \text{for all } \varphi \in V^\ell \text{ a.e.} \\ \langle \tilde{y}^\ell(0), \varphi\rangle_H &= \langle y_0, \varphi\rangle_H \qquad\qquad \text{for all } \varphi \in H. \end{aligned}$$

With the present definition of $\tilde{y}^\ell$ we can generally not expect that $\tilde{y}^\ell(t)$ belongs to V^ℓ, but to the space $\hat{y} + V^\ell$. It turns out that this approach avoids the approximation error in the part of the solution that does not depend on u. This has been investigated more thoroughly in the numerical experiments of [4].

The following result already appeared as Lemma 5.23 in [2].

Lemma 4.22. *The operator $\mathcal{S}^\ell$ is continuous and satisfies*

$$\left\|\mathcal{S}^\ell\right\|_{L(U,L^2(0,T;H))} \leq C \left\|\chi\right\|_{L^2(\Omega;\mathbb{R}^m)},$$

with C being the constant from (4.3). Notably, C is independent of ℓ.

Proof. The claim follows directly from the estimate (4.3) for the solution of the POD approximated state equation and the estimate (4.2) of the operator $\mathcal{B}$. □

As opposed to the situation for the original state equation, in general, the operator $\mathcal{S}^\ell$ is not injective for an arbitrary POD space. Therefore, we have to make additional assumptions.

Assumption 5. *Suppose the POD space V^ℓ is chosen such that $\mathcal{B}$ seen as an element of $L(U; L^2(0,T;(V^\ell)'))$ is injective.*

For the remaining part of this chapter, we require Assumption 5 to be satisfied. With this assumption, the injectivity of $\mathcal{S}^\ell$ can be verified easily. The following result is taken from [3] where it appeared as Lemma 5.28 along with its proof.

Lemma 4.23. *The operator $\mathcal{S}^\ell$ is injective.*

Next, we would like to introduce an a-priori error bound for $\|\mathcal{S}^\ell u - \mathcal{S}u\|_{L^2(0,T;V)}$. To this end, we need to cite the following result from [9] which considers an alternative projection onto the POD space V^ℓ. It appeared in the cited source as a part of Theorem 5.2 and was also proved there.

Lemma 4.24. *Let $(V^\ell)_{\ell\in\mathbb{N}}$ be the POD spaces computed using the snapshots $y^1, \dots, y^\wp \in L^2(0,T;V)$. Then, it holds*

$$\sum_{k=1}^{\wp} \int_0^T \left\| y^k(t) - \mathrm{Q}^\ell y^k(t) \right\|_V^2 \, dt = \sum_{i=\ell+1}^{\infty} \bar{\lambda}_i \left\| \bar{\psi}_i - \mathrm{Q}^\ell \bar{\psi}_i \right\|_V^2 ,$$

where $\{\bar{\psi}_i\}_{i\in\mathbb{N}}$ denotes the POD basis corresponding to the POD spaces $(V^\ell)_{\ell\in\mathbb{N}}$ and $\{\bar{\lambda}_i\}_{i\in\mathbb{N}}$ are the associated eigenvalues given by (3.2). Besides, the orthogonal projection Q^ℓ is defined by

$$\mathrm{Q}^\ell : H \to V^\ell, \quad \mathrm{Q}^\ell v := \inf_{w\in V^\ell} \|v - w\|_H . \tag{4.4}$$

If, in addition, $\left\|\mathrm{Q}^\ell\right\|_{L(V)}$ is bounded independent of ℓ, then for each $k = 1, \dots, \wp$ we have

$$\lim_{\ell\to\infty} \left\| y^k - \mathrm{Q}^\ell y^k \right\|_{L^2(0,T;V)} = 0.$$

Since the projection Q^ℓ will be used in the numerical tests, we also mention how to compute it in practice.

Remark 4.25. In the situation of Lemma 4.24 and for $v \in H$, the orthogonal projection $\mathrm{Q}^\ell v \in V^\ell$ can be obtained by solving

$$M^\ell d = b^\ell,$$

with $d \in \mathbb{R}^\ell$, $M^\ell := \left(\langle \bar{\psi}_i, \bar{\psi}_j \rangle_H\right) \in \mathbb{R}^{\ell\times\ell}$, $b^\ell := \left(\langle v, \bar{\psi}_i \rangle_H\right) \in \mathbb{R}^\ell$ first. Then, it holds

$$\mathrm{Q}^\ell v = \sum_{i=1}^{\ell} d_i \bar{\psi}_i.$$

Now, we are able to present the mentioned error bound which includes the projection Q^ℓ. The proof of the following theorem is based on the techniques used for example in the proofs of Theorems 6.3 and 6.5 in [9] and was worked out in collaboration with the author of [3], where this result already appeared as Theorem 5.33 along with its proof.

Theorem 4.26. *Let $u \in U$ and assume that $(V^\ell)_{\ell\in\mathbb{N}}$ are POD spaces which are constructed based on the snapshot $y^1 = \mathcal{S}u$. Besides, we define $y^\ell := \mathcal{S}^\ell u$ and $y := \mathcal{S}u$. Then, there exists a constant $C = C(C_a, \gamma, \eta, T)$ such that the estimate*

$$\left\|y^\ell - y\right\|^2_{L^2(0,T;V)} \le C \sum_{i=\ell+1}^{\infty} \bar\lambda_i \left\|\bar\psi_i - \mathrm{Q}^\ell\bar\psi_i\right\|^2_V \tag{4.5}$$

is satisfied. Here, $\{\bar\psi_i\}_{i\in\mathbb{N}}$ is the POD basis and $\{\bar\lambda_i\}_{i\in\mathbb{N}}$ are the eigenvalues determined by (3.2) using the mentioned snapshot. Moreover, Q^ℓ is the orthogonal projection defined by (4.4). If, in addition, $\|\mathrm{Q}^\ell\|_{L(V)}$ is bounded independently of ℓ, we have

$$\lim_{\ell\to\infty} \left\|y^\ell - y\right\|_{L^2(0,T;V)} = 0.$$

Proof. To begin with, we define $\theta^\ell(t) := y^\ell(t) - \mathrm{Q}^\ell(y(t)) \in V^\ell$ and $\rho^\ell(t) := \mathrm{Q}^\ell(y(t)) - y(t) \in V$ such that we can write

$$y^\ell(t) - y(t) = y^\ell(t) - \mathrm{Q}^\ell(y(t)) + \mathrm{Q}^\ell(y(t)) - y(t) =: \theta^\ell(t) + \rho^\ell(t)$$

for almost all $t \in (0,T)$. Applying Lemma 4.24 with one snapshot $y^1 = \mathcal{S}u$, we get immediately

$$\left\|\rho^\ell\right\|^2_{L^2(0,T;V)} = \left\|y - \mathrm{Q}^\ell(y)\right\|^2_{L^2(0,T;V)} = \sum_{k=1}^{\wp} \int_0^T \left\|y^k(t) - \mathrm{Q}^\ell y^k(t)\right\|^2_V \, dt = \sum_{i=\ell+1}^{\infty} \bar\lambda_i \left\|\bar\psi_i - \mathrm{Q}^\ell\bar\psi_i\right\|^2_V.$$

Next, using the reduced-order equation (WS^ℓ) and Remark 1.6, we get

$$\begin{aligned} \langle \mathcal{B}u(t), \varphi\rangle_H &= \frac{d}{dt}\langle y(t), \varphi\rangle_H + a(t; y(t), \varphi) && (4.6)\\ &= \frac{d}{dt}\langle \mathrm{Q}^\ell(y(t)), \varphi\rangle_H + \frac{d}{dt}\langle \theta^\ell(t), \varphi\rangle_H + a(t; \mathrm{Q}^\ell(y(t)), \varphi) + a(t; \theta^\ell(t), \varphi) && (4.7)\end{aligned}$$

and thus

$$\frac{d}{dt}\langle \theta^\ell(t), \varphi\rangle_H + a(t; \theta^\ell(t), \varphi) = \langle \mathcal{B}u(t), \varphi\rangle_H - \frac{d}{dt}\langle \mathrm{Q}^\ell(y(t)), \varphi\rangle_H - a(t; \mathrm{Q}^\ell(y(t)), \varphi) \tag{4.8}$$

for all $\varphi \in V^\ell$ and f.a.a. $t \in (0,T)$. Now, we make use of the fact that the projection $\mathrm{Q}^\ell : H \to V^\ell \subset H$ is self adjoint (see [9, p. 874], proof of Theorem 6.5). We infer

$$\frac{d}{dt}\langle \mathrm{Q}^\ell(y(t)), \varphi\rangle_H = \frac{d}{dt}\langle y(t), \mathrm{Q}^\ell(\varphi)\rangle_H = \frac{d}{dt}\langle y(t), \varphi\rangle_H \quad \text{for all } \varphi \in V^\ell \text{ a.e.} \tag{4.9}$$

since $\mathrm{Q}^\ell(\varphi) = \varphi$ holds for all $\varphi \in V^\ell$. Applying (4.9) and (WS) for the u-dependent part to (4.8), we get

$$\frac{d}{dt}\langle \theta^\ell(t), \varphi\rangle_H + a(t; \theta^\ell(t), \varphi) = -a(t; \rho^\ell(t), \varphi) \quad \text{for all } \varphi \in V^\ell \text{ a.e.} \tag{4.10}$$

We continue by choosing $\varphi = \theta^\ell(t) \in V^\ell$ in (4.10) and applying the coercivity of $a(t;\cdot,\cdot)$ as well as Young's inequality for an $\varepsilon > 0$:

$$\begin{aligned}\frac{1}{2}\frac{d}{dt}\left\|\theta^\ell(t)\right\|_H^2 + \gamma\left\|\theta^\ell(t)\right\|_V^2 - \eta\left\|\theta^\ell(t)\right\|_H^2 &\leq \langle\theta_t^\ell(t),\theta^\ell(t)\rangle_H + a(t;\theta^\ell(t),\theta^\ell(t)) \\ &\leq \left|-a(t;\rho^\ell(t),\theta^\ell(t))\right| \\ &\leq C_a\left\|\rho^\ell(t)\right\|_V\left\|\theta^\ell(t)\right\|_V \\ &\leq \frac{C_a^2}{2\varepsilon}\left\|\rho^\ell(t)\right\|_V^2 + \frac{\varepsilon}{2}\left\|\theta^\ell(t)\right\|_V^2\end{aligned} \tag{4.11}$$

Therefore, we find by rearranging and choosing $\varepsilon = 2\gamma$

$$\frac{d}{dt}\left\|\theta^\ell(t)\right\|_H^2 \leq \frac{C_a^2}{2\gamma}\left\|\rho^\ell(t)\right\|_V^2 + 2\eta\left\|\theta^\ell(t)\right\|_H^2.$$

This enables us to apply Gronwall's Lemma 1.2, which implies

$$\begin{aligned}\left\|\theta^\ell(t)\right\|_H^2 &\leq \exp\left(\int_0^t 2\eta\,ds\right)\left(\left\|\theta^\ell(0)\right\|_H^2 + \frac{C_a^2}{2\gamma}\int_0^t\left\|\rho^\ell(s)\right\|_V^2\,ds\right) \\ &\leq \exp\left(2\eta T\right)\frac{C_a^2}{2\gamma}\int_0^T\left\|\rho^\ell(s)\right\|_V^2\,ds \\ &\leq \exp\left(2\eta T\right)\frac{C_a^2}{2\gamma}\left\|y - \mathrm{Q}^\ell(y)\right\|_{L^2(0,T;V)}^2\end{aligned} \tag{4.12}$$

for almost all $t \in (0,T)$. We proceed by using (4.11) once more with $\varepsilon = \gamma$. This yields by rearranging

$$\left\|\theta^\ell(t)\right\|_V^2 \leq \frac{1}{\gamma}\left(-\frac{d}{dt}\left\|\theta^\ell(t)\right\|_H^2 + 2\eta\left\|\theta^\ell(t)\right\|_H^2 + \frac{C_a^2}{\gamma}\left\|\rho^\ell(t)\right\|_V^2\right).$$

We integrate both sides of this inequality over $(0,T)$ and apply (4.12) to the result:

$$\begin{aligned}\left\|\theta^\ell\right\|_{L^2(0,T;V)}^2 &\leq \frac{1}{\gamma}\left(-\int_0^T\frac{d}{dt}\left\|\theta^\ell(s)\right\|_H^2\,ds + 2\eta\int_0^T\left\|\theta^\ell(s)\right\|_H^2\,ds + \frac{C_a^2}{\gamma}\int_0^T\left\|\rho^\ell(s)\right\|_V^2\right) \\ &= \frac{1}{\gamma}\left(-\left\|\theta^\ell(T)\right\|_H^2 + \left\|\theta^\ell(0)\right\|_H^2 + 2\eta\int_0^T\left\|\theta^\ell(s)\right\|_H^2\,ds + \frac{C_a^2}{\gamma}\left\|\rho^\ell\right\|_{L^2(0,T;V)}^2\right) \\ &\leq \frac{2\eta}{\gamma}\int_0^T\left\|\theta^\ell(s)\right\|_H^2\,ds + \frac{C_a^2}{\gamma^2}\left\|\rho^\ell\right\|_{L^2(0,T;V)}^2 \\ &\leq \exp\left(2\eta T\right)\frac{C_a^2\eta T}{\gamma^2}\left\|\rho^\ell\right\|_{L^2(0,T;V)}^2 + \frac{C_a^2}{\gamma^2}\left\|\rho^\ell\right\|_{L^2(0,T;V)}^2.\end{aligned}$$

With the preceding estimate, we can finish the proof of the estimate (4.5). More specifically, it

holds

$$\begin{aligned}\left\|y^\ell - y\right\|^2_{L^2(0,T;V)} &\leq 2\left\|\theta^\ell\right\|^2_{L^2(0,T;V)} + 2\left\|\rho^\ell\right\|^2_{L^2(0,T;V)} \\ &\leq 2\exp(2\eta T)\,\frac{C_a^2\eta T}{\gamma^2}\left\|\rho^\ell\right\|^2_{L^2(0,T;V)} + \frac{2C_a^2}{\gamma^2}\left\|\rho^\ell\right\|^2_{L^2(0,T;V)} + 2\left\|\rho^\ell\right\|^2_{L^2(0,T;V)} \\ &\leq C\sum_{i=\ell+1}^{\infty}\bar{\lambda}_i\left\|\bar{\psi}_i - \mathrm{Q}^\ell\bar{\psi}_i\right\|^2_V\end{aligned}$$

Now, $\lim_{\ell\to\infty}\left\|y^\ell - y\right\|_{L^2(0,T;V)} = 0$ follows directly from this estimate and Lemma 4.24. □

Remark 4.27. Note that the a-priori estimate above depends on the solution $y := \mathcal{S}u$ of the state equation, since this solution is used to construct the POD space, so it is not very time-efficient to use it in order to control the error in numerical applications. However, estimates of this type are still useful as a heuristic to determine the number of used POD basis functions. This will be demonstrated in the numerical part of this thesis.

4.4.2 Reduced-Order Objective Function

By means of the solution operator $\mathcal{S}^\ell$ of the reduced-order model (WS^ℓ), we can also define a POD approximated objective function and the corresponding Euclidean reference point problem.

Definition 4.28. For a given POD space $V^\ell \subset V$ of rank ℓ we define the POD approximated objective function by

$$\hat{J}^\ell : U \to \mathbb{R}^2, \quad \hat{J}^\ell(u) := \begin{pmatrix} \hat{J}_1^\ell(u) \\ \hat{J}_2(u) \end{pmatrix} := \begin{pmatrix} \frac{1}{2}\left\|\hat{y} + \mathcal{S}^\ell u - y_Q\right\|^2_{L^2(0,T;H)} \\ \frac{1}{2}\left\|u\right\|^2_U \end{pmatrix}.$$

Thereby, we obtain the *reduced-order* bicriterial optimal control problem

$$\min_{u\in U_{\mathrm{ad}}} \hat{J}^\ell(u). \tag{ROCP$^\ell$}$$

Additionally, we declare $Y^\ell := \hat{J}^\ell(U_{\mathrm{ad}}) \subset \mathbb{R}^2$ as the objective admissible region with respect to $\hat{J}^\ell$.

Notation 4.29. Henceforth, we denote the original bicriterial optimal control problem (ROCP) by the *full problem* and its reduced-order counterpart (ROCP$^\ell$) by the *POD approximated problem*. In the same way, we will refer to the full and POD approximated versions of the state and adjoint equation.

Next, we can use the POD approximated cost function to formulate a reduced-order Euclidean reference point problem.

Definition 4.30. For a reference point $z \in \mathbb{R}^2$ we define the reduced-order Euclidean reference point problem associated with (ROCP$^\ell$) by

$$\min_{u\in U_{\mathrm{ad}}} \frac{1}{2}\left(\hat{J}_1^\ell(u) - z_1\right)^2 + \frac{1}{2}\left(\hat{J}_2(u) - z_2\right)^2.$$

Analogously to the full problem, we employ the function

$$F_z^\ell : U \to \mathbb{R}, \quad F_z^\ell(u) := \frac{1}{2}\left(\hat{J}_1^\ell(u) - z_1\right)^2 + \frac{1}{2}\left(\hat{J}_2(u) - z_2\right)^2.$$

Consistent with previous notations, we denote this problem by $(\mathrm{RPP})_z^\ell$.

The following result ensures that the results from Chapter 2 are applicable to the POD approximated problem. It appeared in [2] as a part of Lemma 5.30 and was also proved there.

Lemma 4.31. *Let Assumption 5 be satisfied. Then the functions $\hat{J}_1^\ell, \hat{J}_2$ are quadratic and fulfil Assumptions 2 and 3.*

With regard to the approximation quality, we cite the following result about the convergence of the POD approximated objective function $\hat{J}_1^\ell$ from [2] where it appeared as Theorem 5.31 along with its proof.

Theorem 4.32. *Let $(V^\ell)_{\ell\in\mathbb{N}}$ be POD spaces generated by an arbitrary family of snapshots.*

(i) If $u \in U_{ad}$ such that $\mathcal{S}u \in H^1(0,T;V)$ holds, then $\hat{J}_1^\ell(u) \to \hat{J}_1(u)$ as $\ell \to \infty$.

(ii) If $(u^\ell)_{\ell\in\mathbb{N}} \subset U_{ad}$ is a sequence such that $u^\ell \to u$ $(\ell \to \infty)$ for $u \in U_{ad}$ and $\mathcal{S}u \in H^1(0,T;V)$ hold, then $\left|\hat{J}_1^\ell(u^\ell) - \hat{J}_1(u^\ell)\right| \to 0$ and $\hat{J}_1^\ell(u^\ell) \to \hat{J}_1(u)$ as $\ell \to \infty$.

4.4.3 Reduced-Order Adjoint Equation

Similar to the procedure that we used for the full optimal control problem (ROCP), we would also like to introduce an adjoint equation for the reduced-order model, which will once again be used to make the gradient of the objective function $\hat{J}_1^\ell$ more easily computable. Therefore, let $V^\ell \subset V$ be a given POD space of rank ℓ and let $\hat{p}$ be the u-independent part of the solution of (WA).

Definition 4.33. For $u \in U$ we call the end value problem

$$\begin{aligned} \langle -p_t^\ell(t), \varphi\rangle_{(V^\ell)',V^\ell} + a(t;\varphi,p^\ell(t)) &= \langle -\mathcal{S}^\ell u(t), \varphi\rangle_{(V^\ell)',V^\ell} \quad \text{for all } \varphi \in V^\ell \text{ a.e.} \\ \langle p^\ell(T), \varphi\rangle_H &= \langle 0, \varphi\rangle_H \qquad\qquad \text{for all } \varphi \in H \end{aligned} \tag{WA^ℓ}$$

a *reduced-order model* of the u-dependent part of (WA).

This time again, the solvability of the equation can be shown. The following result appeared in [3] as Theorem 5.40.

Theorem 4.34. *Let Assumption 4 be satisfied. Then, there exists a unique solution $p^\ell \in H^1(0,T;V^\ell)$ of (WA^ℓ), which satisfies the estimate*

$$\left\|p^\ell\right\|_{H^1(0,T;V)} \le C \left\|\mathcal{S}^\ell u\right\|_{L^2(0,T;(V^\ell)')}.$$

Notably, the constant $C > 0$ is independent of ℓ.

Proof. The assertions follow from a more general result about the solvability of finite dimensional evolution equations like that from [8, p. 354] in the same way as the solvability of the full adjoint equation resulted from the more general Theorem 1.7 about the solvability of evolution equations. □

Definition and Remark 4.35. Analogously to Definition and Remark 4.15 we declare the solution operator

$$\mathcal{A}^\ell : U \to H^1(0,T;V^\ell) \hookrightarrow L^2(0,T;H)$$

such that $\mathcal{A}^\ell u$ is the unique solution of (WA$^\ell$). Taking into account linearity, we infer that $\tilde{p}^\ell := \hat{p} + \mathcal{A}^\ell u$ fulfils

$$\begin{aligned} \langle -\tilde{p}_t^\ell(t), \varphi\rangle_{(V^\ell)',V^\ell} + a(t;\varphi,\tilde{p}^\ell(t)) &= \langle y_Q - \hat{y} - \mathcal{S}^\ell u(t), \varphi\rangle_{(V^\ell)',V^\ell} && \text{for all } \varphi \in V^\ell \text{ a.e} \\ \langle \tilde{p}^\ell(T), \varphi\rangle_H &= \langle 0, \varphi\rangle_H && \text{for all } \varphi \in H \end{aligned}$$

for all $u \in U$. Similarly to the state equation, we generally have $\tilde{p}^\ell(t) \notin V^\ell$. This kind of splitting of the solution is done for the same reason as for the state equation. Keeping this in mind, we define by $\hat{p}^\ell \in H^1(0,T;V^\ell)$ the reduced-order solution of

$$\begin{aligned} \langle -\hat{p}_t^\ell(t), \varphi\rangle_{(V^\ell)',V^\ell} + a(t;\varphi,\hat{p}^\ell(t)) &= \langle y_Q - \hat{y}, \varphi\rangle_{(V^\ell)',V^\ell} && \text{for all } \varphi \in V^\ell \text{ a.e} \\ \langle \hat{p}^\ell(T), \varphi\rangle_H &= \langle 0, \varphi\rangle_H && \text{for all } \varphi \in H. \end{aligned}$$

This solution exists and is unique due to the same arguments as in Theorem 4.34.

The following result appeared in [3] as Lemma 5.43.

Lemma 4.36. *The solution operator $\mathcal{A}^\ell$ is continuous and satisfies*

$$\left\|\mathcal{A}^\ell\right\|_{L(U,L^2(0,T;H))} \leq C \left\|\chi\right\|_{L^2(\Omega;\mathbb{R}^m)},$$

with a constant $C > 0$ that is independent of ℓ.

Proof. The assertions result directly from Theorem 4.34 and Lemma 4.19. □

Proceeding in a similar way as for the full problem, we are able to find a representation of the derivatives of $\hat{J}_1^\ell$ by making use of the adjoint equation. The following result is taken from [2] where it appeared as Corollary 5.36. Note that the representation of the first derivative has been corrected by replacing $\hat{p}$ with $\hat{p}^\ell$.

Lemma 4.37. *For $u \in U$ we can express the gradient of $\hat{J}_1^\ell$ by*

$$\nabla \hat{J}_1^\ell(u) = -\mathcal{B}^*(\hat{p}^\ell + \mathcal{A}^\ell u)$$

and its second derivative by

$$\nabla^2 \hat{J}_1^\ell(u) = -\mathcal{B}^*\mathcal{A}^\ell.$$

With this additional knowledge about the adjoint equation, we can state convergence results regarding the gradient of $\hat{J}_1^\ell$. The following result is taken from [2] where it appeared as Theorem 5.39 along with its proof.

Theorem 4.38. *Let $(V^\ell)_{\ell\in\mathbb{N}}$ be POD spaces generated by an arbitrary family of snapshots.*

(i) If $u \in U_{ad}$ satisfies $\mathcal{S}u, \mathcal{A}u \in H^1(0,T;V)$, then $\nabla \hat{J}_1^\ell(u) \to \nabla \hat{J}_1(u)$ as $\ell \to \infty$.

(ii) If $(u^\ell)_{\ell\in\mathbb{N}} \subset U_{ad}$ with $u^\ell \to u$, $\ell \to \infty$ for a $u \in U_{ad}$ such that $\mathcal{S}u, \mathcal{A}u \in H^1(0,T;V)$ holds, then $\left\|\nabla \hat{J}_1^\ell(u^\ell) - \nabla \hat{J}_1(u^\ell)\right\|_U \to 0$ and $\nabla \hat{J}_1^\ell(u^\ell) \to \nabla \hat{J}_1(u)$ as $\ell \to \infty$.

Furthermore, we can formulate an a-priori error bound for $\left\|p^\ell - p\right\|^2_{L^2(0,T;V)}$, which is analogous to Theorem 4.26. The proof of the following theorem is based on the techniques used for example in the proofs of Theorems 6.3 and 6.5 in [9] and was worked out in collaboration with the author of [3], where this result already showed up as Theorem 5.48 along with its proof.

Theorem 4.39. *Assume $u \in U$ and that $(V^\ell)_{\ell\in\mathbb{N}}$ are POD spaces, which are constructed based on the snapshots $y^1 = \mathcal{S}u$, $y^2 = \mathcal{A}u$. Besides, we define $p^\ell := \mathcal{A}^\ell u$ and $p := \mathcal{A}u$. Then, there exists a constant $C = C(C_a, \gamma, \eta, T, C_V)$ such that the estimate*

$$\left\|p^\ell - p\right\|^2_{L^2(0,T;V)} \le C \sum_{i=\ell+1}^{\infty} \bar{\lambda}_i \left\|\bar{\psi}_i - \mathrm{Q}^\ell \bar{\psi}_i\right\|^2_V \tag{4.13}$$

is satisfied. Here, $\{\bar{\psi}_i\}_{i\in\mathbb{N}}$ is the POD basis and $\{\bar{\lambda}_i\}_{i\in\mathbb{N}}$ are the eigenvalues determined by (3.2) using the mentioned snapshots. Moreover, Q^ℓ is the orthogonal projection defined by (4.4). If, in addition, $\left\|\mathrm{Q}^\ell\right\|_{L(V)}$ is bounded independently of ℓ, we have

$$\lim_{\ell\to\infty} \left\|p^\ell - p\right\|_{L^2(0,T;V)} = 0.$$

Proof. We proceed very similarly to the proof of Theorem 4.26. We define $\theta^\ell(t) := p^\ell(t) - \mathrm{Q}^\ell(p(t)) \in V^\ell$ and $\rho^\ell(t) := \mathrm{Q}^\ell(p(t)) - p(t) \in V$ such that

$$p^\ell(t) - p(t) = p^\ell(t) - \mathrm{Q}^\ell(p(t)) + \mathrm{Q}^\ell(p(t)) - p(t) =: \theta^\ell(t) + \rho^\ell(t),$$

holds for almost all $t \in (0,T)$. Applying Lemma 4.24 with the two snapshots $y^1 = \mathcal{S}u$ and $y^2 = \mathcal{A}u$, we get immediately

$$\begin{aligned}
\left\|\rho^\ell\right\|^2_{L^2(0,T;V)} &\le \left\|y - \mathrm{Q}^\ell y\right\|^2_{L^2(0,T;V)} + \left\|\rho^\ell\right\|^2_{L^2(0,T;V)} \\
&= \left\|y - \mathrm{Q}^\ell y\right\|^2_{L^2(0,T;V)} + \left\|p - \mathrm{Q}^\ell p\right\|^2_{L^2(0,T;V)} \\
&= \sum_{i=\ell+1}^{\infty} \bar{\lambda}_i \left\|\bar{\psi}_i - \mathrm{Q}^\ell \bar{\psi}_i\right\|^2_V
\end{aligned}$$

using the notation $y = \mathcal{S}u$. In the following, let us also write $y^\ell = \mathcal{S}^\ell u$ like before. We continue by using the POD adjoint equation (WA^ℓ) for $p^\ell = \mathrm{Q}^\ell(p) + \theta^\ell$ to infer

$$-\langle \mathcal{S}^\ell u(t), \varphi\rangle_{(V^\ell)',V^\ell} = -\frac{d}{dt}\langle p^\ell(t)), \varphi\rangle_H + a(t;\varphi,p^\ell(t)) \tag{4.14}$$

$$= -\frac{d}{dt}\langle \mathrm{Q}^\ell(p(t)), \varphi\rangle_H - \frac{d}{dt}\langle \theta^\ell(t), \varphi\rangle_H + a(t;\varphi,\mathrm{Q}^\ell(p(t))) + a(t;\varphi,\theta^\ell(t)) \tag{4.15}$$

and thus

$$-\frac{d}{dt}\langle\theta^\ell(t),\varphi\rangle_H+a(t;\varphi,\theta^\ell(t))=-\langle\mathcal{S}^\ell u(t),\varphi\rangle_{(V^\ell)',V^\ell}+\frac{d}{dt}\langle\mathrm{Q}^\ell(p(t)),\varphi\rangle_H-a(t;\varphi,\mathrm{Q}^\ell(p(t))) \quad (4.16)$$

for all $\varphi\in V^\ell$ and f.a.a. $t\in(0,T)$. Once more, we use that the orthogonal projection $\mathrm{Q}^\ell: H\to H$ is self adjoint to obtain

$$\frac{d}{dt}\langle\mathrm{Q}^\ell(p(t)),\varphi\rangle_H=\frac{d}{dt}\langle p(t),\mathrm{Q}^\ell(\varphi)\rangle_H=\frac{d}{dt}\langle p(t),\varphi\rangle_H \quad \text{for all } \varphi\in V^\ell \text{ a.e.} \quad (4.17)$$

Applying (4.17) and (WA) for the u-dependent part to (4.16), we get

$$\begin{aligned}-\frac{d}{dt}\langle\theta^\ell(t),\varphi\rangle_H+a(t;\varphi,\theta^\ell(t))=&-\langle\mathcal{S}^\ell u(t),\varphi\rangle_{(V^\ell)',V^\ell}+\langle\mathcal{S}u(t),\varphi\rangle_{(V^\ell)',V^\ell}\\&+a(t;\varphi,p(t))-a(t;\varphi,\mathrm{Q}^\ell(p(t))),\end{aligned}$$

which is equivalent to

$$-\frac{d}{dt}\langle\theta^\ell(t),\varphi\rangle_H+a(t;\varphi,\theta^\ell(t))=\langle y(t)-y^\ell(t),\varphi\rangle_{(V^\ell)',V^\ell}+a(t;\varphi,p(t)-\mathrm{Q}^\ell(p(t))).$$

We continue by choosing $\varphi=\theta^\ell(t)\in V^\ell$ in the previous equation and apply the coercivity of $a(t;\cdot,\cdot)$ as well as Young's inequality twice:

$$\begin{aligned}-\frac{1}{2}\frac{d}{dt}\left\|\theta^\ell(t)\right\|_H^2+\gamma\left\|\theta^\ell(t)\right\|_V^2-\eta\left\|\theta^\ell(t)\right\|_H^2\le&\left\|y(t)-y^\ell(t)\right\|_H\left\|\theta^\ell(t)\right\|_H+C_a\left\|\theta^\ell(t)\right\|_V\left\|\rho^\ell(t)\right\|_V\\\le&\frac{1}{2}\left\|y(t)-y^\ell(t)\right\|_H^2+\frac{1}{2}\left\|\theta^\ell(t)\right\|_H^2\\&+\frac{\tilde\varepsilon}{2}\left\|\theta^\ell(t)\right\|_V^2+\frac{C_a^2}{2\tilde\varepsilon}\left\|\rho^\ell(t)\right\|_V^2.\end{aligned} \quad (4.18)$$

Now, by taking $\varepsilon=2\gamma$, we can eliminate $\gamma\left\|\theta^\ell(t)\right\|_V^2$ on both sides of the inequality such that

$$-\frac{d}{dt}\left\|\theta^\ell(t)\right\|_H^2\le\left\|y(t)-y^\ell(t)\right\|_H^2+\frac{C_a^2}{2\gamma}\left\|\rho^\ell(t)\right\|_V^2+(1+2\eta)\left\|\theta^\ell(t)\right\|_H^2 \quad (4.19)$$

follows. Then, we integrate (4.19) on both sides over the interval $(T-t,T)$ to obtain

$$\begin{aligned}\left\|\theta^\ell(T-t)\right\|_H^2-\left\|\theta^\ell(T)\right\|_H^2\le&\int_{T-t}^T\left\|y(s)-y^\ell(s)\right\|_H^2+\frac{C_a^2}{2\gamma}\left\|\rho^\ell(s)\right\|_V^2\,ds\\&+\int_{T-t}^T(1+2\eta)\left\|\theta^\ell(s)\right\|_H^2\,ds\end{aligned}$$

f.a.a. $t\in(0,T)$. We proceed by performing the substitution $r:=T-s$ in the previous inequality, which yields

$$\begin{aligned}\left\|\theta^\ell(T-t)\right\|_H^2\le&\left\|\theta^\ell(T)\right\|_H^2+\int_0^t\left\|y(T-r)-y^\ell(T-r)\right\|_H^2+\frac{C_a^2}{2\gamma}\left\|\rho^\ell(T-r)\right\|_V^2\,dr\\&+\int_0^t(1+2\eta)\left\|\theta^\ell(T-r)\right\|_H^2\,dr.\end{aligned}$$

This enables us to use Gronwall's Lemma 1.2 on $v(t) := \left\|\theta^\ell(T-t)\right\|_H^2$ such that we can estimate f.a.a. $t \in (0,T)$

$$\begin{aligned}\left\|\theta^\ell(T-t)\right\|_H^2 &\leq \exp\left(\int_0^t (1+2\eta)\,ds\right)\left(\left\|\theta^\ell(T)\right\|_V^2 + \int_{T-t}^T \left\|y(s)-y^\ell(s)\right\|_H^2 + \frac{C_a^2}{2\gamma}\left\|\rho^\ell(s)\right\|_V^2\,ds\right)\\ &\leq \exp\left((1+2\eta)t\right)\left(C_V^2\int_0^T \left\|y(s)-y^\ell(s)\right\|_V^2\,ds + \frac{C_a^2}{2\gamma}\int_0^T \left\|\rho^\ell(s)\right\|_V^2\,ds\right)\\ &\leq \exp\left((1+2\eta)T\right)\left(C_V^2\left\|y-y^\ell\right\|_{L^2(0,T;V)}^2 + \frac{C_a^2}{2\gamma}\left\|\rho^\ell\right\|_{L^2(0,T;V)}^2\right),\end{aligned} \tag{4.20}$$

where we have also made use of the embedding $V \hookrightarrow H$ to estimate the norm in H with the embedding constant C_V. Now, we can use (4.18) again with $\varepsilon = \gamma$ and therefore we obtain by rearranging

$$\left\|\theta^\ell(t)\right\|_V^2 \leq \frac{1}{\gamma}\left(\frac{d}{dt}\left\|\theta^\ell(t)\right\|_H^2 + (1+2\eta)\left\|\theta^\ell(t)\right\|_H^2 + \left\|y(t)-y^\ell(t)\right\|_H^2 + \frac{C_a^2}{\gamma}\left\|\rho^\ell(t)\right\|_V^2\right)$$

At this point, we are able to estimate the $\|\cdot\|_H$-terms on the right-hand side of the above estimate by using (4.20). However, we integrate the previous estimate over $(0,T)$ first.

$$\begin{aligned}\left\|\theta^\ell\right\|_{L^2(0,T;V)}^2 \leq& \frac{1}{\gamma}\left\|\theta^\ell(T)\right\|_H^2 - \frac{1}{\gamma}\left\|\theta^\ell(0)\right\|_H^2 + \frac{(1+2\eta)}{\gamma}\int_0^T \left\|\theta^\ell(s)\right\|_H^2\,ds\\ &+ \frac{C_V^2}{\gamma}\left\|y-y^\ell\right\|_{L^2(0,T;V)}^2 + \frac{C_a^2}{\gamma^2}\left\|\rho^\ell\right\|_{L^2(0,T;V)}^2\\ \leq& DC_V^2T\left\|y-y^\ell\right\|_{L^2(0,T;V)}^2 + D\frac{C_a^2T}{2\gamma}\left\|\rho^\ell\right\|_{L^2(0,T;V)}^2\\ &+ \frac{C_V^2}{\gamma}\left\|y-y^\ell\right\|_{L^2(0,T;V)}^2 + \frac{C_a^2}{\gamma^2}\left\|\rho^\ell\right\|_{L^2(0,T;V)}^2.\end{aligned}$$

Specifically, we have $D := \frac{\exp\left((1+2\eta)T\right)(1+2\eta)}{\gamma}$. Eventually, we use the relation

$$\left\|p^\ell - p\right\|_{L^2(0,T;V)}^2 \leq 2\left\|\theta^\ell\right\|_{L^2(0,T;V)}^2 + 2\left\|\rho^\ell\right\|_{L^2(0,T;V)}^2$$

again, where we now have an estimate for both of the terms on the right-hand side of this inequality. Overall, this shows the inequality (4.13). Finally, $\lim_{\ell\to\infty}\left\|p^\ell - p\right\|_{L^2(0,T;V)} = 0$ follows from the estimate we have just proved and Lemma 4.24. □

Remark 4.40. (i) The results from Theorem 4.26 remain valid if we use the snapshots $y^1 = \mathcal{S}u$ and $y^2 = \mathcal{A}u$ for a given $u \in U$ in order to compute the POD spaces mentioned in the theorem.

(ii) Defining alternatively $p^\ell := \mathcal{A}^\ell u + \hat{p}^\ell$ and $p := \mathcal{A}u + \hat{p}$, the estimate (4.13) of the previous theorem for $\left\|p^\ell - p\right\|_{L^2(0,T;V)}^2$ still holds if we replace the second snapshot y^2 by $\mathcal{A}u + \hat{p}$ in order to compute the POD spaces. The corresponding proof can be carried out analogously if we define θ^ℓ and ρ^ℓ like in the above proof, but use p, p^ℓ as declared in this remark instead.

4.4.4 A-Priori Estimates and Convergence

In this section, we investigate the behaviour of the solution of the POD approximated Euclidean reference point problem for increasing dimension of the used POD space. Notably, this solution converges to the solution of the full Euclidean reference point problem and an estimate of the difference between these solutions can be formulated if certain assumptions are fulfilled. The convergence result and its ancillary statements have already been shown in [2] for an optimal control problem that is analogous to the present problem except that there was only a time-independent convection term. However, the solution operators have the same relevant properties in the present case, so the proofs remain valid for the situation in this thesis. The mentioned proofs can be found in [2, pp. 64-66].
In the following, unless stated otherwise, let the POD spaces $(V^\ell)_{\ell\in\mathbb{N}}$ be computed by an arbitrary family of snapshots.
The following result is cited from [2] where it appeared as Lemma 5.40 along with its proof.

Lemma 4.41. *Let $z \in \bigcap_{\ell\in\mathbb{N}} \left(P_{Y^\ell} + \mathbb{R}^2_{\leq}\right)$ be a reference point and let $\bar{u}^\ell$ denote the minimizer of F_z^ℓ for all $\ell \in \mathbb{N}$. Then, $(\bar{u}^\ell)_{\ell\in\mathbb{N}} \subset U_{ad}$ is bounded.*

The following result is taken from [2] where it appeared as Theorem 5.41 along with its proof.

Theorem 4.42. *Let $z \in \bigcap_{\ell\in\mathbb{N}} \left(P_{Y^\ell} + \mathbb{R}^2_{\leq}\right) \cap \left(P_Y + \mathbb{R}^2_{\leq}\right)$ be a reference point and let $\bar{u}^\ell$ denote the minimizer of F_z^ℓ for all $\ell \in \mathbb{N}$ as well as $\bar{u}$ the minimizer of F_z. If $\mathcal{S}\bar{u}, \mathcal{A}\bar{u} \in H^1(0,T;V)$ and additionally $\hat{J}_2(\bar{u}) > z_2$ is fulfilled, it holds*

$$\lim_{\ell\to\infty} \left\|\bar{u}^\ell - \bar{u}\right\|_U = 0.$$

We also have further convergence results about the reduced objective function and its gradient available.
The following result is a citation of [2, Corollary 5.42], where the corresponding proof is also given.

Corollary 4.43. *Let $z \in \bigcap_{\ell\in\mathbb{N}} \left(P_{Y^\ell} + \mathbb{R}^2_{\leq}\right) \cap \left(P_Y + \mathbb{R}^2_{\leq}\right)$ be a reference point and let $\bar{u}^\ell$ denote the minimizer of F_z^ℓ for all $\ell \in \mathbb{N}$ as well as $\bar{u}$ the minimizer of F_z. If $\mathcal{S}\bar{u}, \mathcal{A}\bar{u} \in H^1(0,T;V)$, the following statements hold:*

(i) $\hat{J}_1(\bar{u}^\ell) \to \hat{J}_1(\bar{u})$ *and* $\hat{J}_1^\ell(\bar{u}^\ell) \to \hat{J}_1(\bar{u})$ *as* $\ell \to \infty$.

(ii) $\nabla\hat{J}_1(\bar{u}^\ell) \to \nabla\hat{J}_1(\bar{u})$ *and* $\nabla\hat{J}_1^\ell(\bar{u}^\ell) \to \nabla\hat{J}_1(\bar{u})$ *as* $\ell \to \infty$.

(iii) $\nabla F_z(\bar{u}^\ell) \to \nabla F_z(\bar{u})$ *and* $\nabla F_z^\ell(\bar{u}^\ell) \to \nabla F_z(\bar{u})$ *as* $\ell \to \infty$.

Following the Theorems 4.26 and 4.39 it is possible to derive a similar a-priori error estimate for the control space. The proof of the following theorem was worked out in collaboration with the author of [3], where this result already showed up as Theorem 5.53 along with its proof.

Theorem 4.44. *Let $z \in \bigcap_{\ell\in\mathbb{N}} \left(P_{Y^\ell} + \mathbb{R}^2_{\leq}\right) \cap \left(P_Y + \mathbb{R}^2_{\leq}\right)$ be a reference point and let $\bar{u}^\ell$ be the minimizer of F_z^ℓ for all $\ell \in \mathbb{N}$ as well as $\bar{u}$ the minimizer of F_z. Additionally, we assume $\mathcal{S}\bar{u}, \mathcal{A}\bar{u} \in H^1(0,T;V)$. Suppose that the POD spaces $(V^\ell)_{\ell\in\mathbb{N}}$ are computed using the snapshots*

$y^1 = \mathcal{S}\bar{u}$ and $y^2 = \mathcal{A}\bar{u} + \hat{p}$ with $\hat{p} \in H^1(0,T;V)$. Besides, let $\hat{J}_1^\ell(\bar{u}) \geq z_1$ and $\hat{J}_2(\bar{u}^\ell) > z_2$. Then, there exists a constant $C = C(C_a, \gamma, \eta, T, C_V, z) > 0$ such that the error estimate

$$\left\|\bar{u}^\ell - \bar{u}\right\|^2_{L^2(0,T;V)} \leq C \left(\sum_{i=\ell+1}^{\infty} \bar{\lambda}_i \left\|\bar{\psi}_i - \mathrm{Q}^\ell \bar{\psi}_i\right\|_V^2 \right) \tag{4.21}$$

holds. Here, $\{\bar{\psi}_i\}_{i\in\mathbb{N}}$ is the POD basis and $\{\bar{\lambda}_i\}_{i\in\mathbb{N}}$ the eigenvalues which are determined by (3.2) using the mentioned snapshots. Moreover, Q^ℓ is the orthogonal projection defined by (4.4). If, in addition, $\|\mathrm{Q}^\ell\|_{L(V)}$ is bounded independently of ℓ, we have

$$\lim_{\ell\to\infty} \left\|\bar{u}^\ell - \bar{u}\right\|_U = 0.$$

Proof. To begin with, we use an estimate that was derived at the end of the proof of Theorem 5.41 in [2] and we rewrite the right-hand side of it.

$$\begin{aligned}
&\left(\frac{\hat{J}_1^\ell(\bar{u}^\ell) + \hat{J}_1(\bar{u})}{2} - z_1\right) \left\|\mathcal{S}^\ell(\bar{u}^\ell - \bar{u})\right\|^2_{L^2(0,T;H)} + \left(\frac{\hat{J}_2(\bar{u}) + \hat{J}_2(\bar{u}^\ell)}{2} - z_2\right) \left\|\bar{u}^\ell - \bar{u}\right\|^2_U \\
&\leq \left(\hat{J}_1(\bar{u}) - z_1\right) \left(\hat{J}_1(\bar{u}) - \hat{J}_1^\ell(\bar{u}) + \langle \nabla\hat{J}_1(\bar{u}) - \nabla\hat{J}_1^\ell(\bar{u}), \bar{u}^\ell - \bar{u}\rangle_U\right) \\
&+ \left(\hat{J}_1^\ell(\bar{u}^\ell) - z_1\right) \left(\hat{J}_1^\ell(\bar{u}) - \hat{J}_1(\bar{u})\right) \\
&= \left(\hat{J}(\bar{u}) - z_1\right) \langle \nabla\hat{J}_1(\bar{u}) - \nabla\hat{J}_1^\ell(\bar{u}), \bar{u}^\ell - \bar{u}\rangle_U + \left(\hat{J}_1(\bar{u}) - \hat{J}_1^\ell(\bar{u}^\ell)\right) \left(\hat{J}_1(\bar{u}) - \hat{J}_1^\ell(\bar{u})\right) \\
&= \left(\hat{J}(\bar{u}) - z_1\right) \langle \nabla\hat{J}_1(\bar{u}) - \nabla\hat{J}_1^\ell(\bar{u}), \bar{u}^\ell - \bar{u}\rangle_U \\
&+ \left(\hat{J}_1(\bar{u}) - \hat{J}_1^\ell(\bar{u}) + \hat{J}_1^\ell(\bar{u}) - \hat{J}_1^\ell(\bar{u}^\ell)\right) \left(\hat{J}_1(\bar{u}) - \hat{J}_1^\ell(\bar{u})\right) \\
&= \left(\hat{J}(\bar{u}) - z_1\right) \langle \nabla\hat{J}_1(\bar{u}) - \nabla\hat{J}_1^\ell(\bar{u}), \bar{u}^\ell - \bar{u}\rangle_U + \left(\hat{J}_1(\bar{u}) - \hat{J}_1^\ell(\bar{u})\right)^2 \\
&+ \left(\hat{J}_1^\ell(\bar{u}) - \hat{J}_1^\ell(\bar{u}^\ell)\right) \left(\hat{J}_1(\bar{u}) - \hat{J}_1^\ell(\bar{u})\right).
\end{aligned} \tag{4.22}$$

We continue by estimating the first summand of the expression above. At this, we apply the Lemmata 4.16 and 4.37 to rewrite the derivatives of the objective functions. Additionally, we write $p^\ell := \hat{p}^\ell + \mathcal{A}^\ell u$, $p := \hat{p} + \mathcal{A}u$.

$$\begin{aligned}
&\left(\hat{J}(\bar{u}) - z_1\right) \langle \nabla\hat{J}_1(\bar{u}) - \nabla\hat{J}_1^\ell(\bar{u}), \bar{u}^\ell - \bar{u}\rangle_U \\
&\leq \left(\hat{J}(\bar{u}) - z_1\right) \|\nabla\hat{J}_1(\bar{u}) - \nabla\hat{J}_1^\ell(\bar{u})\|_U \|\bar{u}^\ell - \bar{u}\|_U \\
&= \left(\hat{J}(\bar{u}) - z_1\right) \| - \mathcal{B}^*(\hat{p} + \mathcal{A}\bar{u}) + \mathcal{B}^*(\hat{p}^\ell + \mathcal{A}^\ell\bar{u})\|_U \|\bar{u}^\ell - \bar{u}\|_U \\
&= \left(\hat{J}(\bar{u}) - z_1\right) \|\mathcal{B}^*(p^\ell - p)\|_U \|\bar{u}^\ell - \bar{u}\|_U \\
&\leq \left(\hat{J}(\bar{u}) - z_1\right) \|\mathcal{B}^*\|_{L(L^2(0,T;V),U)} \|p^\ell - p\|_{L^2(0,T;H)} \|\bar{u}^\ell - \bar{u}\|_U \\
&\leq \frac{C_1}{2\varepsilon_1} \|p^\ell - p\|^2_{L^2(0,T;H)} + \frac{\varepsilon_1}{2} \|\bar{u}^\ell - \bar{u}\|^2_U.
\end{aligned} \tag{4.23}$$

For the last estimate, we have used Young's inequality and the boundedness of $\|\mathcal{B}^*\|_{L(L^2(0,T;V),U)}$, which follows from the representation in Lemma 4.17. We proceed by estimating the last summand of (4.22) where we apply that the objective functions are quadratic.

$$\begin{aligned}
&\left(\hat{J}_1^\ell(\bar{u}) - \hat{J}_1^\ell(\bar{u}^\ell)\right)\left(\hat{J}_1(\bar{u}) - \hat{J}_1^\ell(\bar{u})\right)\\
=&\left(\langle \nabla \hat{J}_1^\ell(\bar{u}^\ell), \bar{u} - \bar{u}^\ell\rangle_U + \frac{1}{2}\langle \nabla^2 \hat{J}_1^\ell(\bar{u}^\ell)(\bar{u} - \bar{u}^\ell), \bar{u} - \bar{u}^\ell\rangle_U\right)\left(\hat{J}_1(\bar{u}) - \hat{J}_1^\ell(\bar{u})\right)\\
=&\left(\hat{J}_1(\bar{u}) - \hat{J}_1^\ell(\bar{u})\right)\langle \nabla \hat{J}_1^\ell(\bar{u}^\ell), \bar{u} - \bar{u}^\ell\rangle_U + \frac{1}{2}\left(\hat{J}_1(\bar{u}) - \hat{J}_1^\ell(\bar{u})\right)\|\mathcal{S}^\ell(\bar{u} - \bar{u}^\ell)\|^2_{L^2(0,T;H)}\\
\leq&\left|\hat{J}_1(\bar{u}) - \hat{J}_1^\ell(\bar{u})\right|\|\nabla \hat{J}_1^\ell(\bar{u}^\ell)\|_U\|\bar{u} - \bar{u}^\ell\|_U + \frac{1}{2}\left(\hat{J}_1(\bar{u}) - J_1^\ell(\bar{u})\right)\|\mathcal{S}^\ell(\bar{u} - \bar{u}^\ell)\|^2_{L^2(0,T;H)}\\
\leq&\frac{1}{2\varepsilon_2}\left(\hat{J}_1(\bar{u}) - \hat{J}_1^\ell(\bar{u})\right)^2 + \frac{\varepsilon_2}{2}\|\nabla \hat{J}_1^\ell(\bar{u}^\ell)\|_U^2\|\bar{u} - \bar{u}^\ell\|_U^2 + \frac{1}{2}\left(\hat{J}_1(\bar{u}) - \hat{J}_1^\ell(\bar{u})\right)\|\mathcal{S}^\ell(\bar{u} - \bar{u}^\ell)\|^2_{L^2(0,T;H)}
\end{aligned} \tag{4.24}$$

Furthermore, if we plug the estimates (4.23) and (4.24) into the estimate from the beginning of the proof and summarize the respective terms, we get

$$\begin{aligned}
&\left(\frac{\hat{J}_1^\ell(\bar{u}^\ell) + \hat{J}_1^\ell(\bar{u})}{2} - z_1\right)\left\|\mathcal{S}^\ell(\bar{u}^\ell - \bar{u})\right\|^2_{L^2(0,T;H)}\\
+&\left(\frac{\hat{J}_2(\bar{u}) + \hat{J}_2(\bar{u}^\ell)}{2} - z_2 - \frac{\varepsilon_1}{2} - \frac{\varepsilon_2}{2}\left\|\nabla \hat{J}_1^\ell(\bar{u}^\ell)\right\|^2_U\right)\left\|\bar{u}^\ell - \bar{u}\right\|^2_U\\
\leq&\frac{C_1}{2\varepsilon_1}\|p^\ell - p\|^2_{L^2(0,T;H)} + \left(1 + \frac{1}{\varepsilon_2}\right)\left(\hat{J}_1(\bar{u}) - \hat{J}_1^\ell(\bar{u})\right)^2.
\end{aligned}$$

Here, we can omit the first term since $\hat{J}_1^\ell(\bar{u}^\ell) + \hat{J}_1^\ell(\bar{u}) - 2z_1 \geq 0$ holds. Apart from that, since $\hat{J}_2(\bar{u}) + \hat{J}_2(\bar{u}^\ell) - 2z_2 > 0$ is satisfied by assumption and since we have from Theorem 4.43(ii) that $\|\nabla \hat{J}_1^\ell(\bar{u}^\ell)\|_U^2 \leq C$ holds for all $\ell \in \mathbb{N}$ and a constant $C > 0$, we can choose $\varepsilon_1, \varepsilon_2$ small enough such that

$$C_2\left\|\bar{u}^\ell - \bar{u}\right\|^2_U \leq \left(\frac{\hat{J}_2(\bar{u}) + \hat{J}_2(\bar{u}^\ell)}{2} - z_2 - \frac{\varepsilon_1}{2} - \frac{\varepsilon_2}{2}\left\|\nabla \hat{J}_1^\ell(\bar{u}^\ell)\right\|^2_U\right)\left\|\bar{u}^\ell - \bar{u}\right\|^2_U$$

holds for a constant $C_2 > 0$. This implies

$$\left\|\bar{u}^\ell - \bar{u}\right\|^2_U \leq \frac{1}{C_2}\left(\frac{C_1}{2\varepsilon_1}\left\|p^\ell - p\right\|^2_{L^2(0,T;H)} + \left(1 + \frac{1}{2\varepsilon_2}\right)\left(\hat{J}_1(\bar{u}) - \hat{J}_1^\ell(\bar{u})\right)^2\right). \tag{4.25}$$

Next, we consider the second term of the previous estimate. We find

$$\begin{aligned}\left|\hat{J}_1(\bar{u}) - \hat{J}_1^\ell(\bar{u})\right| =& \frac{1}{2}\left|\|\hat{y} + \mathcal{S}\bar{u} - y_Q\|^2_{L^2(0,T;H)} - \left\|\hat{y} + \mathcal{S}^\ell\bar{u} - y_Q\right\|^2_{L^2(0,T;H)}\right| \\ =& \frac{1}{2}\left(\|\hat{y} + \mathcal{S}\bar{u} - y_Q\|_{L^2(0,T;H)} + \left\|\hat{y} + \mathcal{S}^\ell\bar{u} - y_Q\right\|_{L^2(0,T;H)}\right)\cdot \\ & \left|\|\hat{y} + \mathcal{S}\bar{u} - y_Q\|_{L^2(0,T;H)} - \left\|\hat{y} + \mathcal{S}^\ell\bar{u} - y_Q\right\|_{L^2(0,T;H)}\right| \\ \leq& \left(C_3\,\|\chi\|_{L^2(\Omega;\mathbb{R}^m)}\,\|\bar{u}\|_U + \|\hat{y} - y_Q\|_{L^2(0,T;H)}\right)\cdot \\ & \left|\|\hat{y} + \mathcal{S}\bar{u} - y_Q\|_{L^2(0,T;H)} - \left\|\hat{y} + \mathcal{S}^\ell\bar{u} - y_Q\right\|_{L^2(0,T;H)}\right|\end{aligned}$$

for a constant $C_3 > 0$. Here the first factor can be bounded by a constant that is independent of ℓ whereas the second factor can be estimated by the reverse triangle inequality, that is

$$\left|\|\hat{y} + \mathcal{S}\bar{u} - y_Q\|_{L^2(0,T;H)} - \left\|\hat{y} + \mathcal{S}^\ell\bar{u} - y_Q\right\|_{L^2(0,T;H)}\right| \leq \left\|\mathcal{S}\bar{u} - \mathcal{S}^\ell\bar{u}\right\|_{L^2(0,T;H)}.$$

Finally, if we insert this into the inequality (4.25) we get

$$\left\|\bar{u}^\ell - \bar{u}\right\|^2_U \leq C_4\left\|p^\ell - p\right\|^2_{L^2(0,T;H)} + C_5\left\|\mathcal{S}\bar{u} - \mathcal{S}^\ell\bar{u}\right\|^2_{L^2(0,T;H)}, \tag{4.26}$$

where we can apply Theorem 4.26 and Remark 4.40(ii) to complete the proof of the a-priori estimate. Then, $\lim_{\ell\to\infty}\left\|\bar{u}^\ell - \bar{u}\right\|_U = 0$ follows from the estimate we have just proved and Lemma 4.24.

□

Remark 4.45. Although we used the second snapshot $y^2 = \mathcal{A}u + \hat{p}$ in the previous theorem, the estimate (4.21) still works reasonably well in practice if we only use $\mathcal{A}u$ instead. This will be demonstrated in the numerical tests of this thesis. We prefer to use $\mathcal{A}u$ as a snapshot because it reduces the approximation error of the POD basis (cf. numerical tests in [4]).

4.4.5 A-Posteriori Error Analysis

In this section, we present a-posteriori error estimates, which are ultimately supposed to provide a measure of quality for the solutions of $(\mathrm{RPP})_z^\ell$ in comparison to the solutions of the full problem $(\mathrm{RPP})_z$. That is we would like to estimate $\left\|\bar{u}^\ell - \bar{u}\right\|_U$. The stated results are taken from [2, Section 5.5.5], where a more detailed explanation and the respective proofs can be found.
The following result is an adaption of Theorem 2.14 to the bicriterial optimal control problem of the current chapter and it already appeared in [2] as Corollary 5.43.

Corollary 4.46. *Let $z \in P_Y + \mathbb{R}^2_{\leq}$ be a reference point. Moreover, let $\bar{u} \in U_{ad}$ be the minimizer of F_z as well as $u_p \in U_{ad}$. If $\hat{J}(u_p) \geq z$ and $\hat{J}_2(u_p) > z_2$ hold, then we have the following a-posteriori estimates:*

$$\|u_p - \bar{u}\|_U \leq \left(\frac{\hat{J}_2(u_p) - z_2}{2}\right)^{-1}\|\xi\|_U$$

and

$$\left\|\hat{J}(u_p) - \hat{J}(\bar{u})\right\|_{\mathbb{R}^2} \leq \frac{1}{2}\left(\frac{\hat{J}_2(u_p) - z_2}{2}\right)^{-\frac{1}{2}} \|\xi\|_U ,$$

where $\xi \in U$ is given such that

$$\langle \nabla F_z(u_p) + \xi, u - u_p\rangle_U \geq 0 \quad \text{for all } u \in U_{ad} \tag{4.27}$$

is fulfilled.

To receive the desired estimate from this theorem, we will choose $u_p = \bar{u}^\ell$. Apart from that, the following result shows how $\xi \in U$ can possibly be chosen. The basic idea behind the construction of ξ is explained for example in [10].
The following result appeared in [2] as Lemma 5.44.

Lemma 4.47. *Let $u_p \in U_{ad}$ be arbitrary. Defining $\xi = \xi(u_p) \in U$ by*

$$\xi_i(t) := \begin{cases} -\min(0, \nabla F_z(u_p)_i(t)) & \text{a.e. in } \{t \in [0,T] \mid (u_p)_i(t) = (u_a)_i(t)\}, \\ -\max(0, \nabla F_z(u_p)_i(t)) & \text{a.e. in } \{t \in [0,T] \mid (u_p)_i(t) = (u_b)_i(t)\}, \\ -\nabla F_z(u_p)_i(t) & \text{otherwise} \end{cases} \tag{4.28}$$

for $i = 1, \ldots, m$, (4.27) holds.

Now, it is possible to show that under certain assumptions the mentioned a-posteriori error estimate for $\left\|\bar{u}^\ell - \bar{u}\right\|_U$ converges to 0 as $\ell \to \infty$. The following result is taken from [2] where it appeared as Theorem 5.46 along with its proof.

Theorem 4.48. *Let $z \in \bigcap_{\ell\in\mathbb{N}} \left(P_{Y^\ell} + \mathbb{R}^2_{\leq}\right) \cap \left(P_Y + \mathbb{R}^2_{\leq}\right)$ be a reference point and let $\bar{u}^\ell$ denote the minimizer of F_z^ℓ for all $\ell \in \mathbb{N}$ as well as $\bar{u}$ the minimizer of F_z. Moreover let $\mathcal{S}\bar{u}, \mathcal{A}\bar{u} \in H^1(0,T;V)$ and $\xi^\ell := \xi(\bar{u}^\ell)$ be given by (4.28). Then, it holds*

$$\lim_{\ell\to\infty} \left\|\xi^\ell\right\|_U = 0.$$

Besides, if $\hat{J}_2(\bar{u}) > z_2$, $\hat{J}(\bar{u}^\ell) \geq z$ and $\hat{J}_2(\bar{u}^\ell) > z_2$ hold for all $\ell \in \mathbb{N}$, we have

$$\left\|\bar{u}^\ell - \bar{u}\right\|_U \leq \left(\frac{\hat{J}_2(\bar{u}^\ell) - z_2}{2}\right)^{-1} \left\|\xi^\ell\right\|_U \to 0 \quad (\ell \to \infty)$$

and

$$\left\|\hat{J}(\bar{u}^\ell) - \hat{J}(\bar{u})\right\|_{\mathbb{R}^2} \leq \frac{1}{2}\left(\frac{\hat{J}_2(\bar{u}^\ell) - z_2}{2}\right)^{-\frac{1}{2}} \left\|\xi^\ell\right\|_U \to 0 \quad (\ell \to \infty).$$

Remark 4.49. The convergence results of the previous theorem serve as motivation to use the mentioned estimate of $\left\|\bar{u}^\ell - \bar{u}\right\|_U$ as a measure of the quality of the solutions $\bar{u}^\ell$, since the estimate converges to zero as $\ell \to \infty$. This gives reason to believe that increasing the number of POD basis functions ℓ also improves the approximation of $\bar{u}$ by $\bar{u}^\ell$ at some point.

4.4.6 POD-Based Algorithm to Compute the Pareto Front

In order to prepare the use of POD in our numerical experiments, we would like to specify how to incorporate POD into Algorithm 1. In particular this is achieved in the following way. First the Pareto optimal controls $\bar{u}^0$ and $\bar{u}^L$ are obtained by minimizing $\hat{J}_1$ and $\hat{J}_2$, respectively. Then, we determine the corresponding state y and the adjoint p, which serve as the snapshots for the construction of the POD basis. Consequently, we compute the POD basis $\{\bar{\psi}_i\}_{i=1}^{\ell_{max}}$ of a certain maximum size ℓ_{max} using y and p. For the solution of further reference point problems we choose $\ell < \ell_{max}$ and use the resulting POD basis of size ℓ. Thereby, different strategies to choose ℓ exist. We will investigate some of these strategies in the numerical tests of this thesis. The complete procedure to compute the Pareto front while applying POD is presented in Algorithm 2 in a condensed form.
This algorithm was already used in a similar version in [2], but not stated explicitely. In a form close to the following, it already appeared in [7] as Algorithm 2.

Algorithm 2 POD-based Algorithm to compute the Pareto front

Require: Maximum number $N_{\max} \in \mathbb{N}$ of Pareto points, step size parameters $h_x, h_p > 0$;
1: Minimize the individual objectives $\hat{J}_1$ and $\hat{J}_2$ seperately to obtain $\bar{u}^0$ and $\bar{u}^L$, respectively;
2: Set $\tilde{P}_U = \{\bar{u}^0\}$ and $\tilde{P}_Y = \{\hat{J}(\bar{u}^0)\}$;
3: Compute POD basis $\{\bar{\psi}_i\}_{i=1}^{\ell}$ from $(DPOD^{\ell})$ using the state y and the dual p associated with $\bar{u}^0$;
4: Set $n = 0$ and compute z^1 using (2.5);
5: **while** $n < N_{\max} - 1$ and $z_1^{n+1} < \hat{J}_1(\bar{u}^L)$ **do**
6: Set $n = n + 1$;
7: Solve $(\mathrm{RPP})_{z^n}^{\ell}$ with starting point $\bar{u}^{n-1}$;
8: Set $\tilde{P}_U = \tilde{P}_U \cup \{\bar{u}^n\}$ and $\tilde{P}_Y = \tilde{P}_Y \cup \{\hat{J}(\bar{u}^n)\}$;
9: Compute z^{n+1} using (2.4);
10: Return $\tilde{P}_U = \tilde{P}_U \cup \{\bar{u}^L\}$ and $\tilde{P}_Y = \tilde{P}_Y \cup \{\hat{J}(\bar{u}^L)\}$

One possible way to choose the number of POD basis functions ℓ is to estimate the approximation error of the POD solution by using an a-posteriori estimate like those presented in the previous section and to extend the POD basis if the result of the estimate was not satisfactory. In particular for our numerical experiments we will use the following a-posteriori estimate for the error in the control space:

$$\mu_U(\bar{u}^1, z^1) := \left(\frac{\hat{J}_2(\bar{u}^1) - z_2^1}{2} \right)^{-1} \left\| \xi(\bar{u}^1) \right\|_U$$

$$\mu_Y(\bar{u}^1, z^1) := \frac{1}{2} \left(\frac{\hat{J}_2(\bar{u}^1) - z_2^1}{2} \right)^{-\frac{1}{2}} \left\| \xi(\bar{u}^1) \right\|_U$$

for $n = 1$ and

$$\mu_U(\bar{u}^n, z^n) := \left(\frac{\mu_J(\bar{u}^{n-1}, z^{n-1}) + \hat{J}_2(\bar{u}^n)}{2} - z_2^n \right)^{-1} \|\xi(\bar{u}^n)\|_U \tag{4.29}$$

$$\mu_Y(\bar{u}^n, z^n) := \frac{1}{2}\left(\frac{\mu_J(\bar{u}^{n-1}, z^{n-1}) + \hat{J}_2(\bar{u}^n)}{2} - z_2^n\right)^{-\frac{1}{2}} \|\xi(\bar{u}^n)\|_U$$

for $n = 2, \ldots, N_{\max} - 2$. Specifically, $\mu_J(\bar{u}^{n-1}, z^{n-1})$ is defined by

$$\mu_J(\bar{u}^{n-1}, z^{n-1}) := \frac{h_p\left(\hat{J}_2(\bar{u}^{n-1}) - \mu_Y(\bar{u}^{n-1}, z^{n-1}) - z_2^{n-1}\right)}{\left\|\hat{J}(\bar{u}^{n-1}) - z^{n-1}\right\|_{\mathbb{R}^2} + \mu_Y(\bar{u}^{n-1}, z^{n-1})}$$

Note that these estimates can be derived from Theorem 2.14 by using information about previous Pareto and reference points in order to infer $\mu_J(\bar{u}^{n-1}, z^{n-1}) \leq \hat{J}_2(\bar{u}^n) - z_2^n$. These modified a-posteriori estimates were deduced in [2, p. 84] and were found to have better efficiency than the estimates in Corollary 4.46. In the following chapter, we will explain in more detail, how the error estimation by μ_U is implemented in order to control the number of used POD basis functions.

Chapter 5

Numerical Experiments

As opposed to the bicriterial optimal control problems investigated in [2] and [3], in the present master thesis we focus on problems in which the room is modeled by a set $\Omega \subset \mathbb{R}^3$, i.e. we consider three space dimensions instead of two. This enables us to investigate models that are closer to real-world conditions since by heating up a room we usually think of a three-dimensional space, which is not completely filled by heaters. Additionally, we take into consideration inhomogeneous boundary conditions, which allow for heat exchange of the modeled room with the outer world. With homogeneous boundary conditions, the applicability of the model is very limited since the convection only moves around the air inside the room.
Naturally, in 3D the occurring linear systems of equations stemming from the finite element discretization become much larger than in 2D if we use the same maximum edge length for the finite element triangulation. That is why POD becomes all the more useful and important in this case. Thus, we will put special emphasis on developing an efficient and sufficiently accurate strategy to choose the length of the POD basis.

5.1 Numerical Implementation

This section provides details about the numerical implementation of the solution algorithms. The complete implementation is achieved in MATLAB R2016b and is based on the code used for a 2D problem in [2] and [7]. The computations are performed on a standard PC with Windows 10, an Intel(R) Core(TM) i3-4005U CPU @ 1.7Ghz, 4GB RAM.

5.1.1 Discretization of the State and Adjoint Equation

The discretization of the three-dimensional space is achieved by linear finite elements which are generated by the MATLAB PDE toolbox using a maximum edge length of 0.1. As a result, we get a mesh consisting of 1650 nodes and 7667 elements. The discretization is visualised in Figure 5.1.
With this higher number of nodes compared to the two-dimensional model, it turns out that the direct solution of the arising systems of linear equations requires a much higher computational effort, so it becomes appropriate to use iterative solvers instead. In particular, we will use the *generalized minimal residual* (gmres) method by applying the MATLAB function `gmres`, since the systems of equations resulting from the discretization of the state equation will not be symmetric. For the two-dimensional problem, we choose the same configuration as in [2]. The time interval is discretized equidistantly into 100 subintervals. We use the Crank-Nicolson method for time integration. Note that in contrast to the theoretical investigations in Chapter 4, we now denote the finite element discretized problem by the full problem and the POD approximation thereof by the POD (approximated) problem. Regarding POD, we use the u-

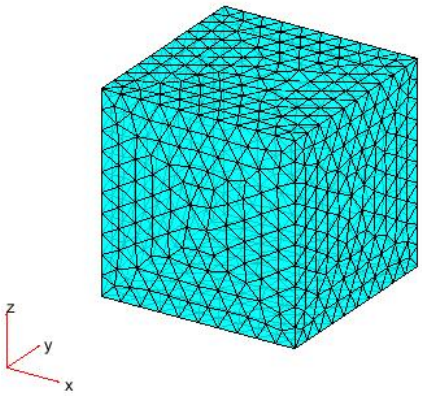


Figure 5.1: Finite element discretization of space

dependent part of the solution to the state equation, denoted by y and the u-dependent part of the solution to the adjoint equation, denoted by p, as the two snapshots to compute the POD basis throughout this chapter. More specifically, for the first POD basis this always refers to the solutions of the full problem whereas for subsequent POD bases it depends on the employed algorithm if the respective solutions to the full or to the POD problem are meant to be used. We also recall that the respective POD basis is constructed according to the method of snapshots, see Remark 3.13.

5.1.2 Computation of the Pareto Front

For the computation of the Pareto front we will use, until further notice, Algorithm 1 if the full problem is considered and Algorithm 2 as a baseline if the POD problem is treated. To begin with, we proceed as in [2] by using the weight $\omega = (\omega_1, \omega_2) = (1, 0.02)$ on the objectives $(\hat{J}_1, \hat{J}_2)$ to calculate the top Pareto point, which facilitates the optimization by regularization of the problem. Later on, we will investigate the difficulties arising when $\omega = (1, 0)$ is used instead. Note that the minimization of $\hat{J}_2$ does not require any calculation since $u = 0$ satisfies the constraints we specified earlier. After having computed the first and the last Pareto point, we determine the remaining Pareto points by solving Euclidean reference point problems.

Solution of the Reference Point Problems

The occurring scalar optimization problems, i.e. the Euclidean reference point problems, are solved by the projected Newton-CG method. For the necessary evaluations of the gradient and of the second derivative, we use the representations we derived in Chapter 4 by making use of the adjoint equation.

We maintain the same stopping condition for the optimization routine as in [2], i.e. we stop if

the a-posteriori estimates

$$\|u_n - \bar{u}\|_U \leq \left(\frac{\hat{J}_2(u_n) - z_2}{2} \right)^{-1} \|\xi(u_n)\|_U \leq \varepsilon_U$$

and

$$\left\| \hat{J}(u_n) - \hat{J}(\bar{u}) \right\|_{\mathbb{R}^2} \leq \frac{1}{2} \left(\frac{\hat{J}_2(u_n) - z_2}{2} \right)^{-\frac{1}{2}} \|\xi(u_n)\|_U \leq \varepsilon_Y$$

(cf. Corollary 4.46) are fulfilled, where $\bar{u}$ is the unknown optimal solution of $(\mathrm{RPP})_z$, u_n is the the current iterate and $\xi(u_n)$ given by (cf. Lemma 4.47)

$$\xi_i(u_n)(t) := \begin{cases} -\min(0, \nabla F_z(u_n)_i(t)) & \text{a.e. in } \{t \in [0,T] \mid (u_n)_i(t) = (u_a)_i(t)\}, \\ -\max(0, \nabla F_z(u_n)_i(t)) & \text{a.e. in } \{t \in [0,T] \mid (u_n)_i(t) = (u_b)_i(t)\}, \\ -\nabla F_z(u_n)_i(t) & \text{otherwise.} \end{cases}$$

Here, we have to replace the gradient ∇F_z by ∇F_z^ℓ in case we are solving the POD approximated problem $(\mathrm{RPP})_z^\ell$ and $\bar{u}$ will be the unknown optimal solution to $(\mathrm{RPP})_z^\ell$. For the following numerical experiments we use the thresholds $\varepsilon_U := 10^{-4}$ and $\varepsilon_Y := 10^{-5}$ for the error in the control and in the objective space, respectively.
For the computation of the reference points, we initially use certain fixed values for the step sizes h_x and h_p. Subsequently, we will investigate the effect of varying h_p on the optimization performance.

5.2 Model Including Three Space Dimensions

We start by formulating the bicriterial optimal control problem with adjustments for a three-dimensional space domain. This model serves as a baseline for the whole chapter. The three-dimensional model is distinguished from the two-dimensional model by the way the controls, i.e. the heaters, are positioned. In order to achieve a more realistic three-dimensional model, the heaters should not occupy the whole room, as a generalization of the two-dimensional model in [2] would suggest. Instead, we place three individual heaters that are distributed in our three-dimensional space, more precisely in the domains $A_1 = (0, 0.125) \times (0.05, 0.45) \times (0.05, 0.4)$, $A_2 = (0, 0.125) \times (0.55, 0.95) \times (0.05, 0.4)$ and $A_3 = (0.25, 0.75) \times (0, 0.125) \times (0.05, 0.4)$, (cf. Figure 5.2b). In detail we define the optimization problem:

$$\min J(u, y) = \begin{pmatrix} \frac{1}{2} \|y - y_Q\|^2_{L^2(0,1;L^2(\Omega))} \\ \frac{1}{2} \|u\|^2_{L^2(0,1;\mathbb{R}^3)} \end{pmatrix} \tag{NOC}$$

subject to

$$\begin{aligned} y_t(t,x) - \kappa \Delta y(t,x) + c_\beta \beta(t,x) \cdot \nabla y(t,x) &= \textstyle\sum_{i=1}^3 u_i(t) \chi_i(x) && \text{for } (t,x) \in (0,1) \times \Omega \\ \tfrac{\partial y}{\partial \eta}(t,x) &= 0 && \text{for } (t,x) \in (0,1) \times \Gamma \\ y(0,x) &= y_0(x) && \text{for } x \in \Omega \end{aligned} \tag{SEC}$$

and

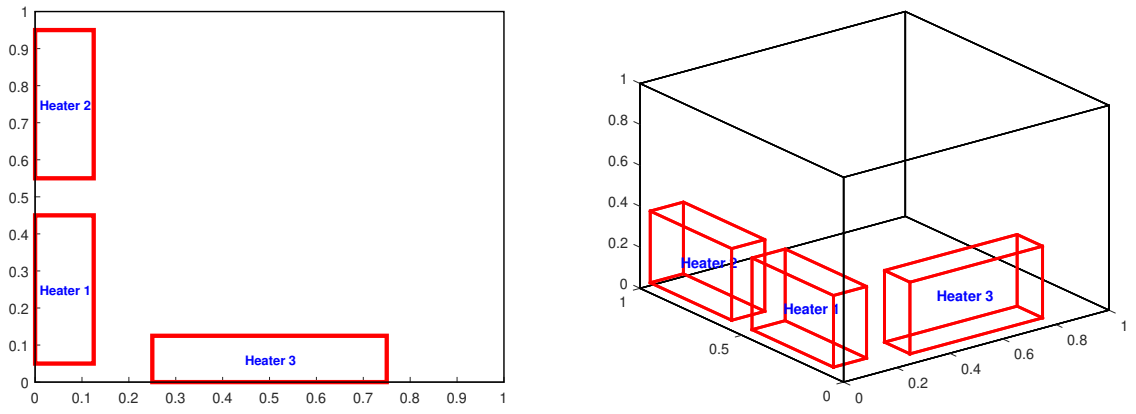
$$u_a(t) \leq u(t) \leq u_b(t) \quad \text{for almost all } t \in [0,1]. \tag{BCon}$$

In addition, we use the following parameter and function values in this example:

- The room Ω is given by the three-dimensional unit cube, i.e. $\Omega = (0,1)^3$.
- As the convection term $\beta(t,x)$ with $(t,x) \in (0,1) \times \Omega$, we use a solution of a Navier-Stokes equation which is described by a stream which goes in a parabolic way from the rear left to the front right of the room (cf. Figure 5.11). In case of the time-dependent convection there is a vortex moving along the mentioned stream over time through the room. The vortex ends up approximately at time $t = 0.1$ on the right-hand side of the front wall of the room. The parameter $c_\beta \geq 0$ is a constant which is used to control the strength of the convection.
- We fix the bilateral control constraints at $u_a = 0$ and $u_b = 3$.
- As an initial condition, we assume a space-independent temperature of 16° everywhere in the room, i.e. $y_0(x) = 16$ for all $x \in \Omega$.
- We choose the desired temperature as $y_Q(t,x) = 16 + 2t$ for all $(t,x) \in (0,1) \times \Omega$, which means that it increases linearly from 16° at the starting time $t = 0$ to 18° at the end of the observed time interval $T = 1$.

With these assumptions, we are able to verify Assumption 4 of Chapter 4, such that we can formulate the objective function to only depend on u in the same way as we did in Chapter 4. Thereby, we obtain the reduced objective function $\hat{J}$ again. Furthermore, we will use the same notations as in Chapter 4.

5.3 Model Including a Time-Independent Convection Term



(a) Heaters in 2D

(b) Heaters in 3D

Figure 5.2: Positioning of the heaters

To begin with, we consider the stated optimal control problem (NOC) with a time-independent convection term, i.e. $b(t,x) = b(x)$, see Figure 5.3.

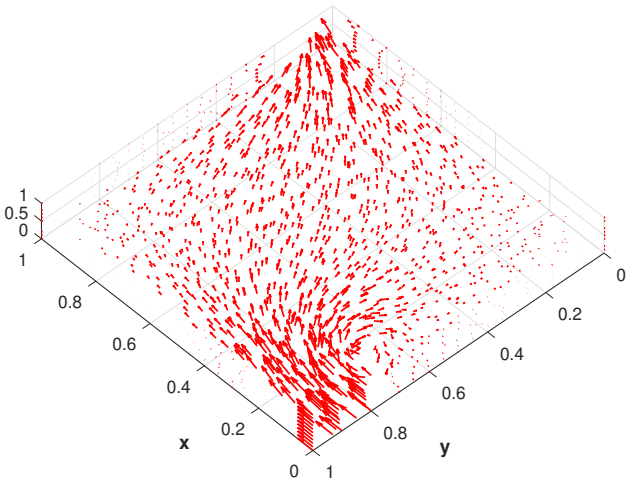


Figure 5.3: Time-independent convection

Considering the finite element discretization of the state and adjoint equation (cf. weak formulation (WS)), we note that for the 3D model the resulting systems of linear equations of size 1650×1650 are too large to handle in adequate time for direct solvers like MATLAB "backslash". This problem is not discussed in detail in this thesis but iterative methods being on average more than five times faster for a single solution of the state equation led to the decision of chosing iterative solvers for the systems of linear equations. Furthermore, if a convection term is included, the occurring matrices are no longer symmetric and therefore have to be treated by an iterative method that can handle such matrices. That is why we use the `gmres` method provided by MATLAB with a restart after ten iterations, which yielded good performance in our test runs. For the present problems preconditioning did not seem to speed up the numerical solution. It is also worth mentioning that `gmres` did perform satisfactory for each problem considered in this chapter. However, it will not be stated explicitely since this is not a main focus of the numerical experiments. Apart from rare cases, the algorithm did not need more than 30 total iterations to achieve convergence.
For the purpose of comparison, we also consider a 2D model which corresponds to the defined 3D model by being its projection onto a plane, see Figure 5.2a. More specifically, the room is given as the 2D unit square and the heaters are distributed in the domains $A_1 = (0, 0.125) \times (0.05, 0.45)$, $A_2 = (0, 0.125) \times (0.55, 0.95)$ and $A_3 = (0.25, 0.75) \times (0, 0.125)$. It should be noted that by projecting the heaters onto a plane we lose information about the height of the heaters which will also have an impact on the numerical results. By an initial test run of solving the full problem with Algorithm 1, we obtain the Pareto front depicted in Figure 5.4.
Note that we define $P^n := \hat{J}(\bar{u}^n)$ for $n \in \{0, \dots, N_P + 1\}$, where N_P is the number of computed inner Pareto points for the respective bicriterial optimization problem. Turning to the results of the test run the individual Pareto points range between $P^0 = (0.592, 1.8166)$ and $P^{27} = (0.667, 0)$. So far we have used a perpendicular step length $h_x = 0.1$ and a parallel step length $h_p = 0.5$. In order to achieve a finer approximation of the Pareto front, we will employ $h_x = 0.05$, $h_p = 0.25$ for this specific problem. Hereby we obtain the Pareto front in Figure 5.5b

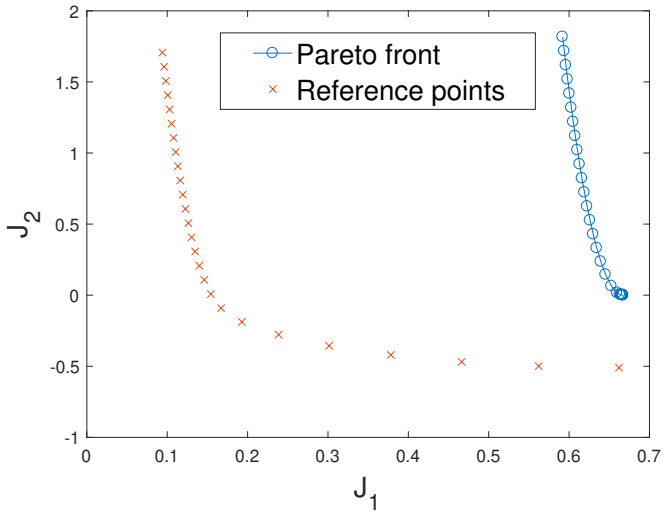
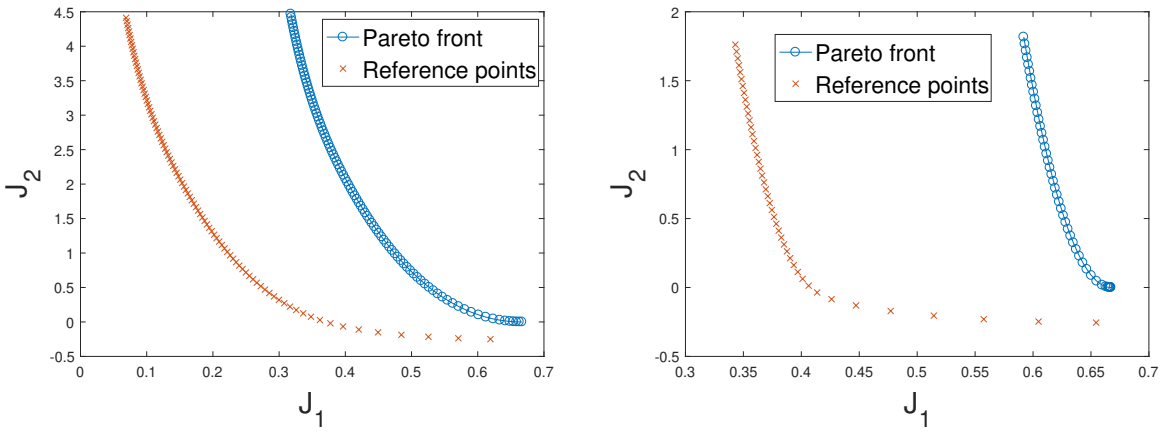


Figure 5.4: Pareto front for $h_x = 0.1$, $h_p = 0.5$

with Pareto points ranging from $P^0 = (0.592, 1.816)$ to $P^{45} = (0.667, 0)$. For comparison we have a plot of the Pareto front of the respective 2D problem with Pareto points ranging from $P^0 = (0.31782, 4.4675)$ to $P^{98} = (0.6667, 0)$.



(a) Paretofront 2D

(b) Paretofront 3D

Figure 5.5: Comparison of the Pareto fronts for $h_x = 0.05$, $h_p = 0.25$

Apparently, for the 2D problem, a better approximation of the desired temperature y_Q can be achieved at the upper left end of the Pareto front than for the 3D problem. Furthermore, for the same heating costs at the top Pareto point of the 3D problem of 1.816 we get a significantly better approximation of the desired temperature for the 2D problem. Namely, for the 2D problem, the Pareto point $(0.418, 1.78)$ is calculated. That is we achieve a an approximation of the desired temperature which is better by 0.174 in terms of $\hat{J}_1$ for approximately the same

heating costs. This is mainly due to the heaters occupying a relatively larger part of the room in terms of the volume of the whole room for the 2D problem. As already mentioned, the height of the heaters in the 3D problem cannot be projected onto a 2D plane properly. Indeed, if we calculate the Pareto front of a 3D problem with heaters of height one, that is the full height of the room, we obtain a more similar result, see Figure 5.6. In this case the outer Pareto points are $P^0 = (0.36872, 4.6201)$ and $P^{100} = (0.66725, 0)$, respectively.

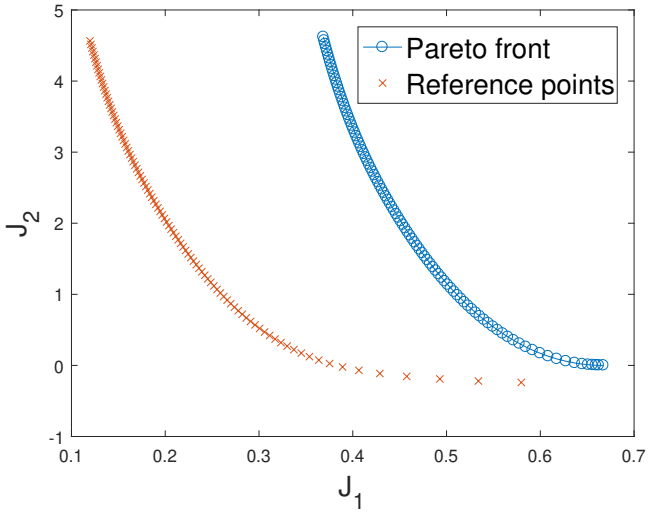
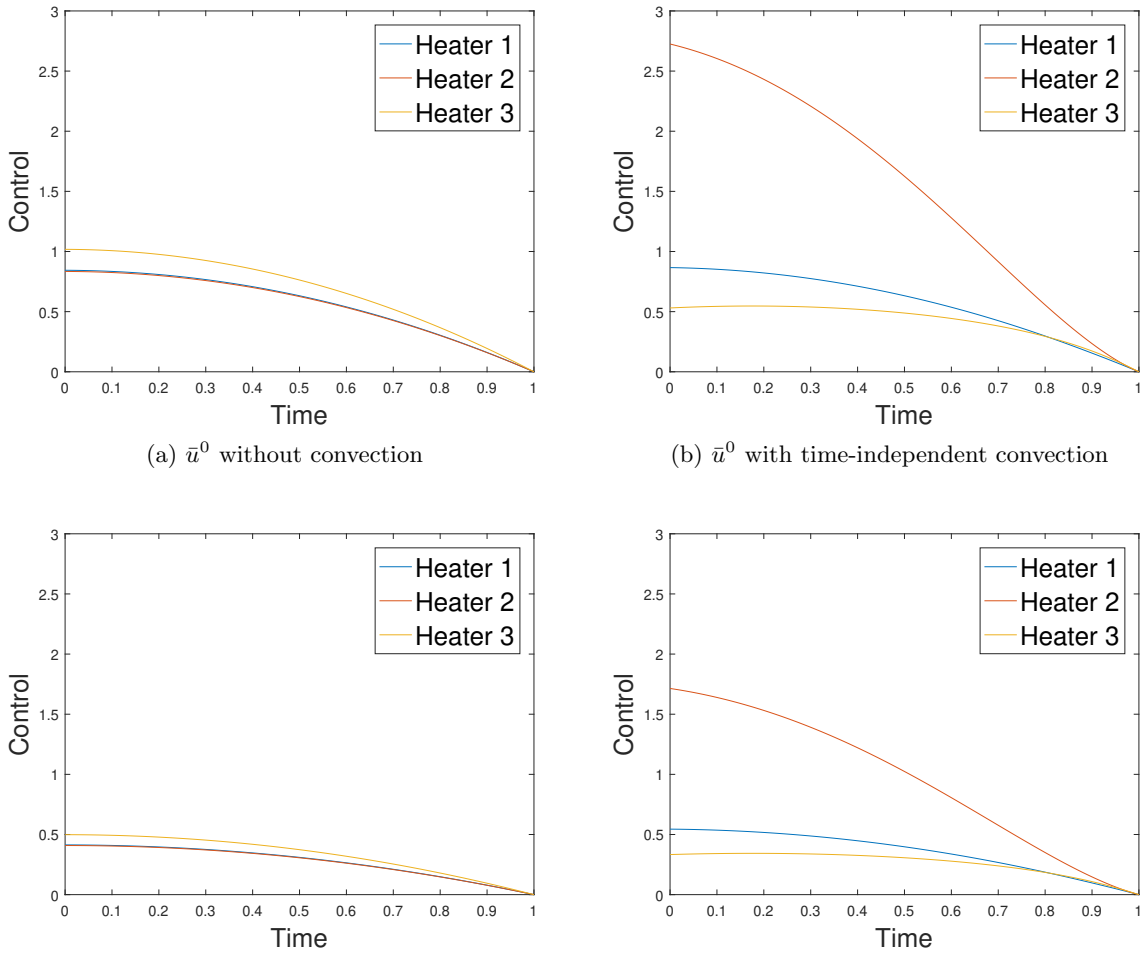


Figure 5.6: Paretofront using heaters of height 1

We should keep this difference in mind for future comparisons of the 2D- and the 3D problem. Besides, this experiment has confirmed the plausible fact that larger heaters, allow for a stronger heating effect, in this case, a better fulfillment of the first objective. In order to maintain a realistic scenario, we keep the heaters at the size specified in section 5.2.

Now we would like to investigate the influence of the convection on the heating strategies. In Figure 5.7 we compare the heating strategies of the problems without convection and with convection by taking a look at the controls corresponding to the top Pareto point, i.e. the controls that achieve the best possible approximation of the desired temperature in terms of $\hat{J}_1 + 0.02\hat{J}_2$. Additionally, the controls corresponding to a Pareto point in the middle of the Pareto front are visualised for the respective problems.



(a) $\bar{u}^0$ without convection

(b) $\bar{u}^0$ with time-independent convection

(c) $\bar{u}^{10}$ without convection Pareto point

(d) $\bar{u}^{22}$ with time-independent convection

Figure 5.7: Comparison of controls for different Pareto points

We are able to observe a considerable difference. Without convection, heaters one and two are heating almost to the same extent. On the other hand, with convection, heater two has to heat by far the most. Additionally, with convection, the heating strategies are not necessarily monotonically decreasing anymore, which becomes most apparent from the heating strategy of heater three. The difference between the heating strategies for the problem with and without convection is due to the heat flow from the rear left to the front right of the room, which is caused by the convection. More specifically, heater two is in the area of the room where the

stream of the convection starts, so it is advantageous in terms of the first objective $\hat{J}_1$ that heater two heats by far the most and from its position the heated air is transported through the room by the convection.
POD basis extension algorithms similar to the following Algorithm 3 already appeared in multiple sources for example in [2] as Algorithm 2, where it was applied to a related 2D problem with floor heating. With respect to this algorithm, we recall the definition of μ_u in (4.29).

Algorithm 3 POD basis extension algorithm

Require: Maximum number $N_{\max} \in \mathbb{N}$ of Pareto points, step size parameters $h_p, h_x > 0$, initial number of POD basis functions ℓ_0, maximum number of POD basis functions ℓ_{max}, a-posteriori bound μ;

Set check = 0;
while check = 0 **do**
  Solve $(\text{RPP})^{\ell}_{z^n}$ using POD basis $\{\bar{\psi}_i\}_{i=1}^{\ell}$ of rank ℓ and starting point $\bar{u}^{n-1}$;
  Compute the a-posteriori estimate $\mu_U(\bar{u}^n, z^n)$;
  if $\mu_U(\bar{u}^n, z^n) < \mu$ **or** $\ell \geq \ell_{max}$ **then**
    Set check = 1;
  else
    Set $\ell = \ell + 1$;

Now we investigate the 3D problem further with regard to the choice of the POD basis. First, we test the basis extension routine described in Algorithm 3 with the parameters $\mu = 4 \cdot 10^{-4}$ and $\ell_0 = 2$ and $\ell_{max} = 50$. Let us also set a minimum number of POD basis functions $\ell_{min} = 2$, which is supposed to hold throughout this chapter unless stated otherwise. In order to determine the effect of the convection term on the number of required POD basis elements, we perform a test run of the problem without convection term using the step size $h_x = 0.05$, as well as of the present problem with convection term. In Table 5.1 we immediately observe that the problem with convection requires considerably more basis extensions in order to stay below the error threshold μ.

Table 5.1: Required basis extensions

	basis extensions
without convection	5
with convection	27

This behaviour is of course expected since adding a convection term introduces more complex dynamics to the problem, which require more POD basis elements to be approximated adequately. This notion is also supported by the behaviour, in particular, the decline, of the eigenvalues of the two problems. That is the eigenvalues of the problem without convection decrease considerably faster than for the problem with convection. Notably, the greatest eigenvalue of the problem with convection is about ten times as large as the greatest eigenvalue of the problem without convection while for the tenth eigenvalue, the corresponding eigenvalues already differ by a factor of 10^3 (cf. Figure 5.8). Of course, this leads to the problem with convection requiring more POD basis functions to achieve the desired accuracy. Indeed, relation (3.7) supports this observation, since in this context faster decreasing eigenvalues result in the sum $\sum_{i=\ell+1}^{d} \bar{\lambda}_i^n$

being smaller for a smaller value of ℓ in contrast to the problem with convection. So the error stemming from POD, which is expressed by the left-hand side of (3.7), should behave in the same way.

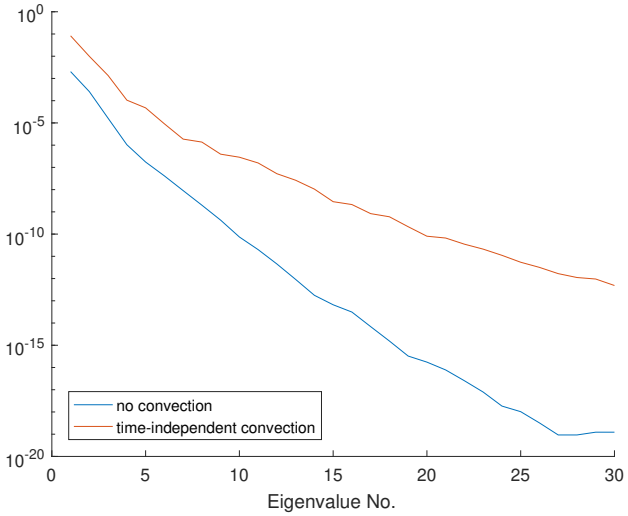


Figure 5.8: Behaviour of eigenvalues

Furthermore, we observe the computation times of the problems with and without convection in Table 5.2 for both the full problem and the reduced-order problem created by the previously used POD strategy. Note that the computation time for the full problem with convection is more than four times as long as for the full problem without convection. For the corresponding POD problems, we find that for the problem with convection even more time is saved in relation to the full problem, namely 82% as opposed to only 71% for the respective problems without convection.

Table 5.2: Comp. time in seconds

	FULL	POD	rel. time saving
without convection	210	61	71 %
with convection	862	159	82 %

Since the number of Pareto points is different for the two problems, we also take a look at the average computation time per Pareto point, see Table 5.3.

Table 5.3: Computation time per Pareto point

	FULL	POD
without convection	9.5	2.76
with convection	18.72	3.46
additional effort	97 %	25 %

It becomes obvious that the difference in total computation time is not only due to the number

of Pareto points, but that the average computation time for each Pareto point rises significantly if we include a convection term. While for the full problem the computation time per Pareto point almost doubles for the problem with convection, for the POD problems the computation time only increases by 25%. Overall the use of POD appears to be even more beneficial for the problem with added convection.
We have seen that the inclusion of a convection term into the problem leads to a significantly higher number of POD basis functions. Since for every basis extension the current Pareto point has to be recalculated, basis extensions can be costly with regard to computation time if they are used excessively. Thus, it becomes desirable to limit the number of basis extensions. One possible approach might be to choose a greater number of initial POD basis functions. However, we do not want to choose such a value arbitrarily, but we would like to develop a strategy to choose this value according to certain properties of the problem, such that the strategy might be valid for other problems, too. We try out a criterion, which has already been applied to a similar 2D problem successfully, see [11]. Specifically, we choose ℓ_0 such that

$$\mathcal{K}(\ell_0) := \sum_{i=\ell_0+1}^{\ell_{max}} \lambda_i \|\psi_i - \mathrm{Q}^{\ell_0}\psi_i\|^2_{H^1(\Omega)} < \varepsilon \tag{5.1}$$

is fulfilled for a suitable $\varepsilon < \mu$, with μ being the previously used a-posteriori bound. This approach is based on Theorem 4.44, where an a-priori estimate for the error in the control space was proved. Although the stated theorem is not applicable directly, since the available POD space is not computed from the optimal control $\bar{u}$, it can still serve as guidance. That is because the optimal control $\bar{u}$ depends continuously on the reference point problem, so we can reasonably assume that if the available control, that is the solution of the previous reference point problem, does not differ too much from the actual optimal control $\bar{u}$, we might still use the stated estimate as an approximation. Besides, we do not know the magnitude of the constant in the estimate (4.21), so we have to determine an appropriate value experimentally. It should also be noted that we use slightly different snapshots than in the assumptions of Theorem 4.44 (cf. Remark 4.45). Since we suspect this difference to have limited impact, we will just examine the usefulness of the described criterion numerically. Apart from this, let us recall that Q^ℓ can be computed by means of Remark 4.25.
Furthermore, to define the criterion more specifically, we choose $\ell_{max} = 50$, since calculating the full sum for all of the 1650 eigenvalues would mean an unacceptably high computational effort. Here, we also take into account that the eigenvalues λ_i become very small for $i > 50$ and also drop off very rapidly (cf. Figure 5.8), so that the further addends only have a minor impact anyway. We choose $\ell_{min} = 2$ as the minimal starting value for ℓ_0. Now we compare the required POD basis extensions corresponding to the described strategy in Table 5.4.
Obviously, ℓ_0 increases if ε decreases. Thus it is not surprising to see that the number of basis extensions decreases with decreasing ε, if we bear in mind that most of the time increasing the number of used POD basis elements improves the accuracy of the solution. Here, the time to determine ℓ_0 by the above criterion has no significant impact on the computation time. Evidently, in case there is no convection, the optimal choice of ε is already reached at 10^{-8} which leads to seven initial POD basis functions and no execution of basis extensions. We are able to visualise the number of initial POD basis elements derived by the aforementioned strategy by calculating the value of the function $\mathcal{K}(\ell) = \sum_{i=\ell+1}^{\ell_{max}} \lambda_i \|\psi_i - \mathrm{Q}_H^\ell \psi_i\|^2_{H^1(\Omega)}$ for various ℓ. The result is plotted in Figure 5.9.

Table 5.4: Test concerning the choice of ε

ε	ℓ_0	basis extensions	comp. time
10^{-5}	6	29	154.18
10^{-6}	10	18	135.39
10^{-7}	13	16	136.74
10^{-8}	16	13	131.09
10^{-9}	19	10	125.40
10^{-10}	23	6	118.76
10^{-11}	29	2	111.87
10^{-12}	31	1	105.84

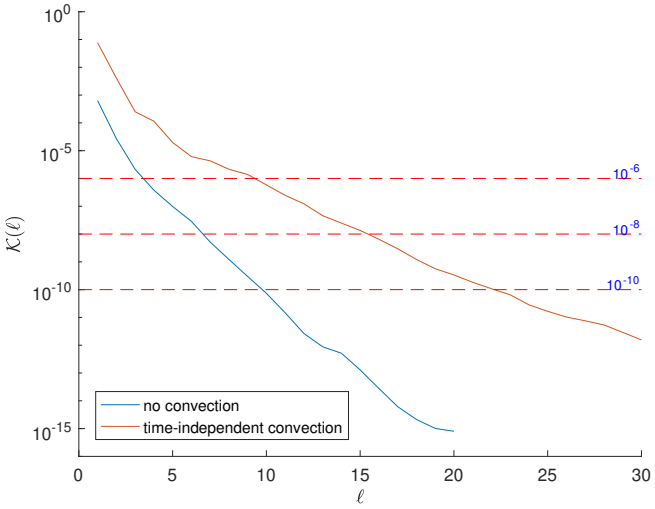


Figure 5.9: Behaviour of $\mathcal{K}$ for the problem both with and without convection

Apparently, the value of $\mathcal{K}$ starts at a lower value for the problem without convection than for the problem with convection and also declines faster afterwards. So to a certain extent, the criterion given by $\mathcal{K}$ takes into account that the problem without convection requires a much smaller number of POD basis functions compared to the problem with convection in order to achieve the same accuracy with respect to the a-posteriori estimate. This is due to the fact that for the same bound ε a much smaller ℓ_0 is calculated. Additionally, we can infer that in general, it is not advisable to choose an ε that is too small since we might use far more POD basis functions than necessary as a result. One possible way to reach a compromise is to perform POD basis updates, i.e. calculating a new POD basis at an appropriate point during the computation of the Pareto front instead of performing an excessive number of basis extensions that have diminishing returns in the sense of improving the accuracy of the approximation.

There is another similar criterion which can be used as a heuristic to determine a suitable number of initial POD basis functions, namely if we choose ℓ_0 such that $\mathcal{E}(\ell_0) := \frac{\sum_{i=1}^{\ell_0} \lambda_i}{\sum_{i=1}^{\ell_{max}} \lambda_i} > 1 - \varepsilon$ holds. For an interesting comparison we plot both of the mentioned criteria in Figure 5.10 as a function

of ℓ. Evidently, $\mathcal{K}$ falls off faster than $1-\mathcal{E}$ initially, then both criteria take a parallel course which

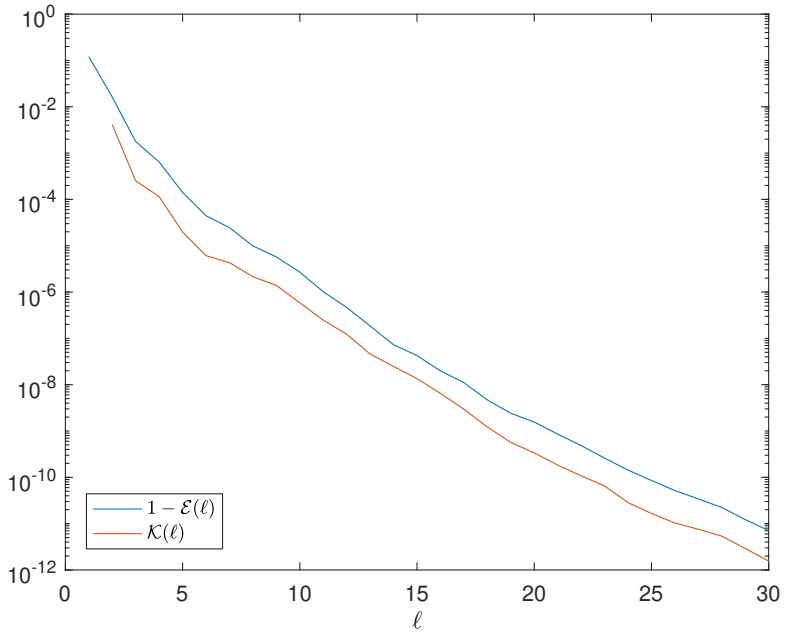


Figure 5.10: Comparison of two criteria

means that for the same value of ℓ they differ by a certain, approximately constant factor which is in this case around five. It appears that by using $1-\mathcal{E}$ instead of $\mathcal{K}$ we might achieve similar results with respect to the choice of ℓ_0 by adjusting for the aforementioned factor. Accordingly, for the same choice of ε we find that $1-\mathcal{E}$ leads to a number of POD basis elements that is higher by about two to three than if we use $\mathcal{K}$. Having said that, in the present thesis we will mainly focus our investigations on the criterion $\mathcal{K}$ and how it can be applied appropriately.
Moving on, we would like to extend our POD strategy by basis updates, i.e. calculating a new POD basis at suitable times during the computation of the Pareto front. Upfront we have the expectation that POD updates might prevent the used POD basis from becoming excessively large during the process of basis extension and apart from this they might be helpful for problems in which basis extensions are not able to achieve the desired approximation quality of the solutions no matter how long the basis becomes. The latter problem might occur if there are strong dynamics included in the problem, e.g. by a convection such that in the course of the calculation of the Pareto front the available POD basis does not represent the current problem adequately anymore. With a new, more suitable POD basis it is even possible that fewer basis elements are required for the same accuracy in terms of the a-posteriori-estimate.
A POD basis update algorithm similar to the following Algorithm 4 appeared in [3] as Algorithm 4 and was tested on a related 2D problem. The algorithm shows the procedure for the n-th reference point problem. Note that in lines 13-14 of Algorithm 4 the new POD basis is constructed based on the state and adjoint that had to be computed for the a-posteriori estimate $\mu_U(\bar{u}^\ell, z^n)$. This means that no additional full problem is solved at this stage. Only if this shortened procedure is not successful, a full reference point problem is solved to determine a

new POD basis, see lines 17-18 of Algorithm 4. If solving the full problem can be avoided, this approach saves a considerable amount of computation time.

Algorithm 4 POD basis update algorithm

Require: $\mu > 0$, $\varepsilon_0 > 0$, $0 < \ell_{min} < \ell_{max}$, $\ell \in [\ell_{min}, \ell_{max}]$;
1: Determine ℓ with ε_0 by criterion (5.1);
2: Set check $= 0$, updateflag $= 0$;
3: **while** check $= 0$ **do**
4: Solve $(\text{RPP})^{\ell}_{z^n}$ with starting point $\bar{u}_0 = \bar{u}^{n-1}$ to obtain $\bar{u}^{\ell}$;
5: Compute the a-posteriori estimate $\mu_U(\bar{u}^{\ell}, z^n)$;
6: **if** $\mu_U(\bar{u}^{\ell}, z^n) < \mu$ **then**
7: Set check $= 1$;
8: **else**
9: **if** $\ell < \ell_{max}$ **then**
10: Set $\ell = \ell + 1$;
11: **else**
12: **if** updateflag $= 0$ **then**
13: Compute POD basis using the state y and the adjoint p associated
14: with $\bar{u}^{\ell}$;
15: Set updateflag $= 1$;
16: **else if** updateflag $= 1$ **then**
17: Solve full problem $(\text{RPP})_{z^n}$;
18: Compute POD basis using state y and its adjoint p corresponding to $\bar{u}$;
19: Determine ℓ with ε_0 by criterion (5.1) for the new POD basis, set check $= 1$.

Specifically, putting POD basis updates into practice, we test Algorithm 4 where the initial ℓ_0 is determined by criterion (5.1) with $\varepsilon = 10^{-8}$. Using different values for ℓ_{max}, we compare the computation time in Table 5.5. In column ℓ_0 the POD basis length at the start and after each basis update, which has been determined by means of (5.1), is stated.

Table 5.5: Test regarding the choice of ℓ_{max}

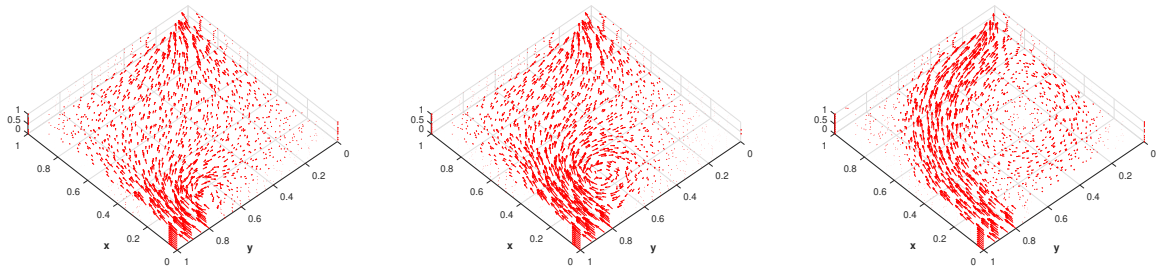
ℓ_{max}	ℓ_0	basis extensions	basis updates	comp. time
16	15,14,14	3	2	129.66
18	16,14	2	1	116.81
20	16,14	4	1	120.52
22	16,14	6	1	126.33
24	16,14	8	1	130.22

For every value of ℓ_{max} that was tested in Table 5.5, at least one basis update was performed. Apparently, this is also more efficient in terms of computation time if we compare the computation times with that in Table 5.4, since for every tested value of ℓ_{max} the computation time is lower than without basis update. The reason for this is that the number of basis extensions could be reduced significantly. In particular, $\ell_{max} = 18$ appears to be the best choice, whereby two basis extensions are performed at Pareto point number 26 and one basis update is performed

at Pareto point number 27. In contrast, the choice $\ell_{max} = 16$ turns out to be too small which results in more than one basis update being performed. Besides, as far as the choice of ε is concerned, although we could achieve an even shorter computation time (cf. Table 5.4) with a very small ε, this choice might be too specific to the present problem. Therefore, with no prior knowledge, it equates to guessing the right number of POD basis functions right away. If we have to treat a different problem that has not yet been investigated extensively, it might be better to choose a slightly bigger ε, so that the number of POD basis functions does not become too large already at the start. Consequently, $\varepsilon = 10^{-8}$ is one possible choice that appears to be a good compromise. For comparison in [11], where a related 2D problem was considered, $\varepsilon = 2 \cdot 10^{-6}$ turned out to be a relatively efficient choice. For general problems, ε might have to be adapted after having gained some insight into the problem by the means of numerical tests. With regard to further investigations of POD basis strategies, we examine a more complex problem in the following section.

5.4 Model Including a Time-Dependent Convection Term

In this section, we investigate the bicriterial optimal control problem of the previous Section 5.3 again, the only difference being that the convection term b in the state equation is now time-dependent. The relation of the convection of this section to the one of the previous section is that the time-independent convection corresponds to the time-dependent convection at time $t = 0$. For investigations of a related problem with time-dependent convection in two space dimensions, we refer to [3]. We would like to put emphasis on the application of POD in this case in comparison to the example of the previous section. To gain a better understanding of how the convection behaves, we visualise it at three particular time instances in Figure 5.11.

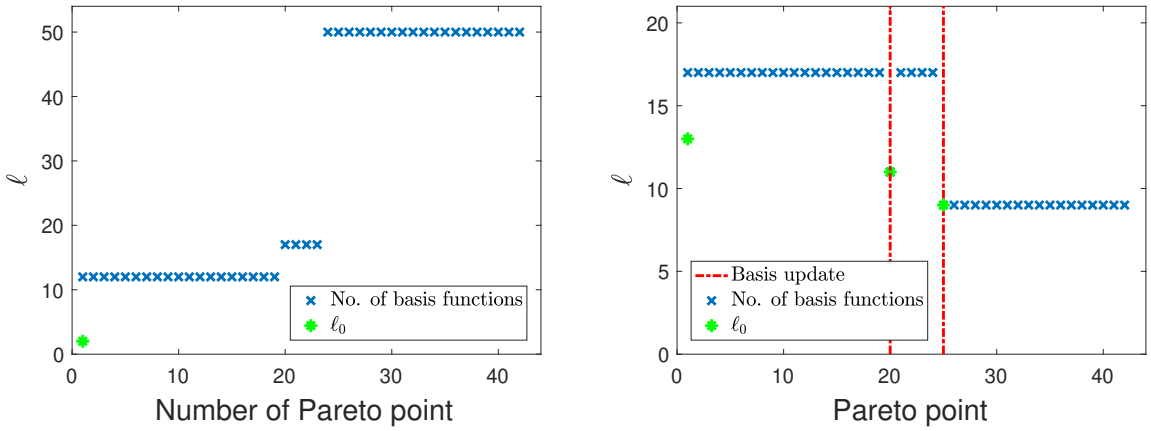


(a) Convection at $t = 0$ (b) Convection at $t = 0.01$ (c) Convection at $t = 0.5$

Figure 5.11: Convection at three time instances

Unless stated otherwise, we use the same step size parameters for the computation of the Pareto front as for the problem with time-independent convection term, i.e. $h_x = 0.05$, $h_p = 0.25$. To begin with, we test Algorithm 4 with the parameters $\ell_{max} = 50$, $\varepsilon_0 = 10^{-6}$ and Algorithm 3 with the parameters $\ell_{max} = 50$, $\ell_0 = 2$ on this problem. For the problem with time-dependent convection, we observe that we are not able to achieve the desired accuracy in terms of the a-posteriori estimate without performing at least one basis update. That is basis extensions alone did not suffice, even if the original POD basis was extended up to a length of 50 elements

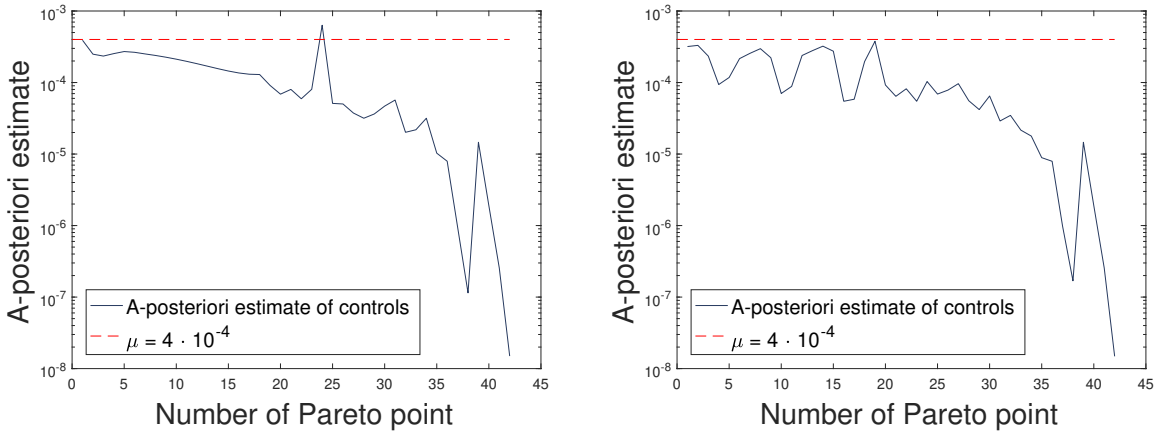
if Algorithm 3 is employed (cf. Figure 5.12). In Figure 5.13a we see a plot of the a-posteriori estimate at the respective Pareto points. In particular, we can observe that the bound $\mu = 4{\cdot}10^{-4}$ is exceeded at Pareto point number 25 for Algorithm 3.



(a) Basis extension algorithm

(b) Basis update algorithm

Figure 5.12: Basis extension and updates



(a) A-posteriori estimate basis extension

(b) A-posteriori estimate basis update

Figure 5.13: Comparison of two POD basis strategies

At a certain number of POD basis functions we cannot expect much improvement anymore since the eigenvalues become very small (cf. Figure 5.14). This is due to relation (3.7) which indicates that the error does not decrease much if an eigenvalue with very small value is left out from the sum. In contrast, a basis update strategy is able to achieve further improvement, which means it is able to keep the a-posteriori estimate below the chosen threshold, as shown in Figure 5.13b. Note that if any basis updates are performed during the computation of the Pareto front, statements about eigenvalues always refer to the eigenvalues which were calculated

for the initial POD basis unless it is stated otherwise.
Once more, the behaviour of the eigenvalues in comparison to the previously investigated problems is very interesting (cf. Figure 5.14). While the eigenvalues for the time-dependent and the time-independent problem are very similar up to the sixth eigenvalue, they fall significantly slower for the time-dependent problem afterwards. In particular, the difference at the 15th eigenvalue is already of a factor of 50 and the 20th eigenvalue of the time-dependent problem is already 250 times as large as for the time-independent problem. This behaviour reflects the more complex dynamics of the problem with time-dependent convection. Because of this we already expect the time-dependent problem to require more POD basis functions in order to achieve the same approximation quality as for the time-independent problem. We recall the error representation by (3.7) which gives some intuition how the rate of decline of the eigenvalues is related to the requirement of a sufficient number of POD basis functions for the desired accuracy.

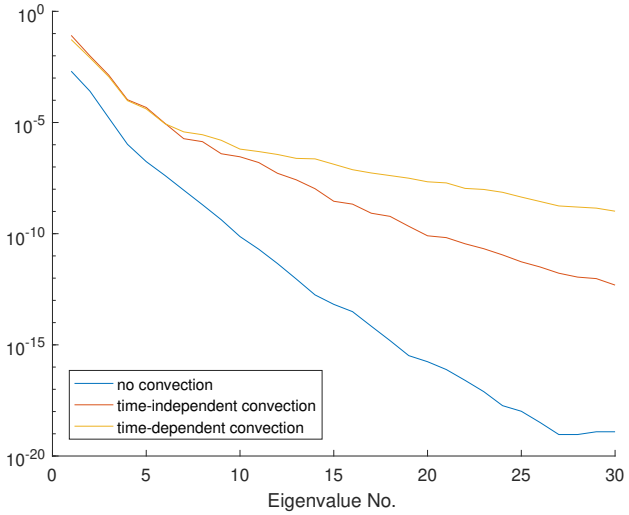


Figure 5.14: Behaviour of eigenvalues

If we take a look at the criterion which was used to determine ℓ_0, we are able to observe similar results as for the eigenvalues. Therefore, we visualise $\mathcal{K}(\ell)$ for $\ell_{max} = 50$ in Figure 5.15. We expect that for the same choice of ε we get a significantly greater ℓ_0 for the time-dependent problem. In particular, $\varepsilon = 10^{-8}$ results in $\ell_0 = 16$ for the time-independent case and judging from the plot, we estimate ℓ_0 to be at about 27 for the time-dependent case, but certainly quite a lot higher than for the time-independent problem.
Now, we would like to take a look at the computed Pareto front for the current time-dependent problem. A first test run yields the Pareto front in Figure 5.16 with Pareto points ranging between the outer points $P^0 = (0.5956, 1.7243)$ and $P^{43} = (0.6669, 0)$. The Pareto front looks very similar to the one of the problem with time-independent convection. Looking at the Pareto front in a descending direction up to a value of about 0.5 for the functional $\hat{J}_2$, the Pareto front is relatively steep and has little curvature, i.e. for a certain difference in heating costs, expressed by $\hat{J}_2$, the difference in approximation of the desired temperature, in the sense of $\hat{J}_1$, remains pretty much constant in this part of the Pareto front. Below the described part of the

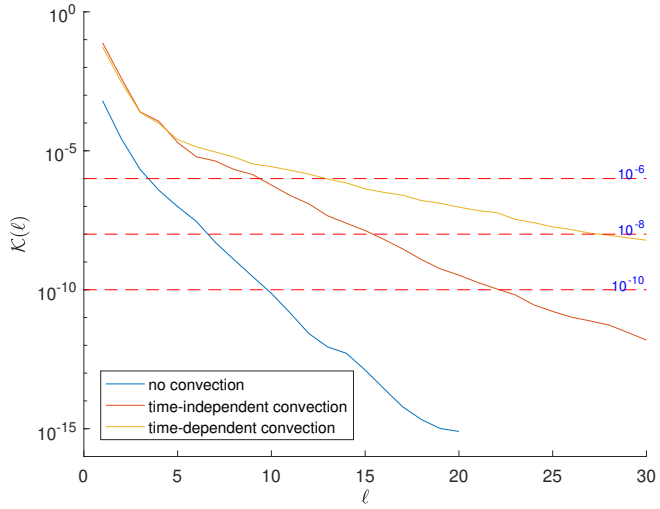


Figure 5.15: Behaviour of $\mathcal{K}$ for the three tested examples

Pareto front, that is below a value of about 0.5 for $\hat{J}_2$, the curvature increases. Specifically, if the heating costs are decreased further, the approximation of the desired temperature worsens disproportionally. Conversely, by increasing the heating costs up to a value of 0.2 for $\hat{J}_2$, we achieve the best improvement in terms of approximation of the desired temperature relative to the rest of the Pareto front.

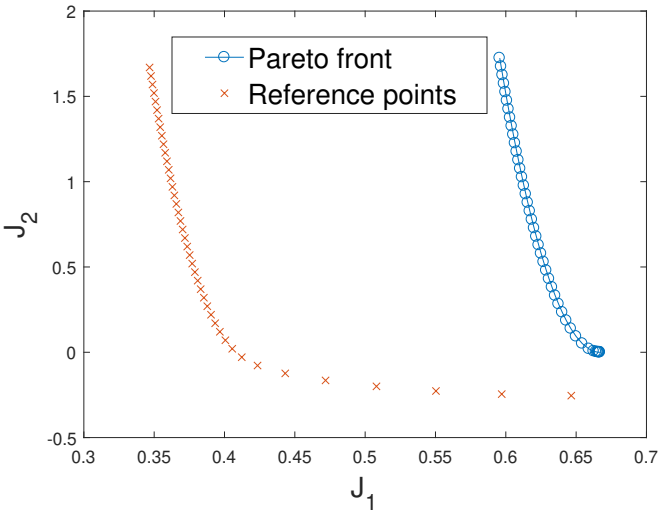
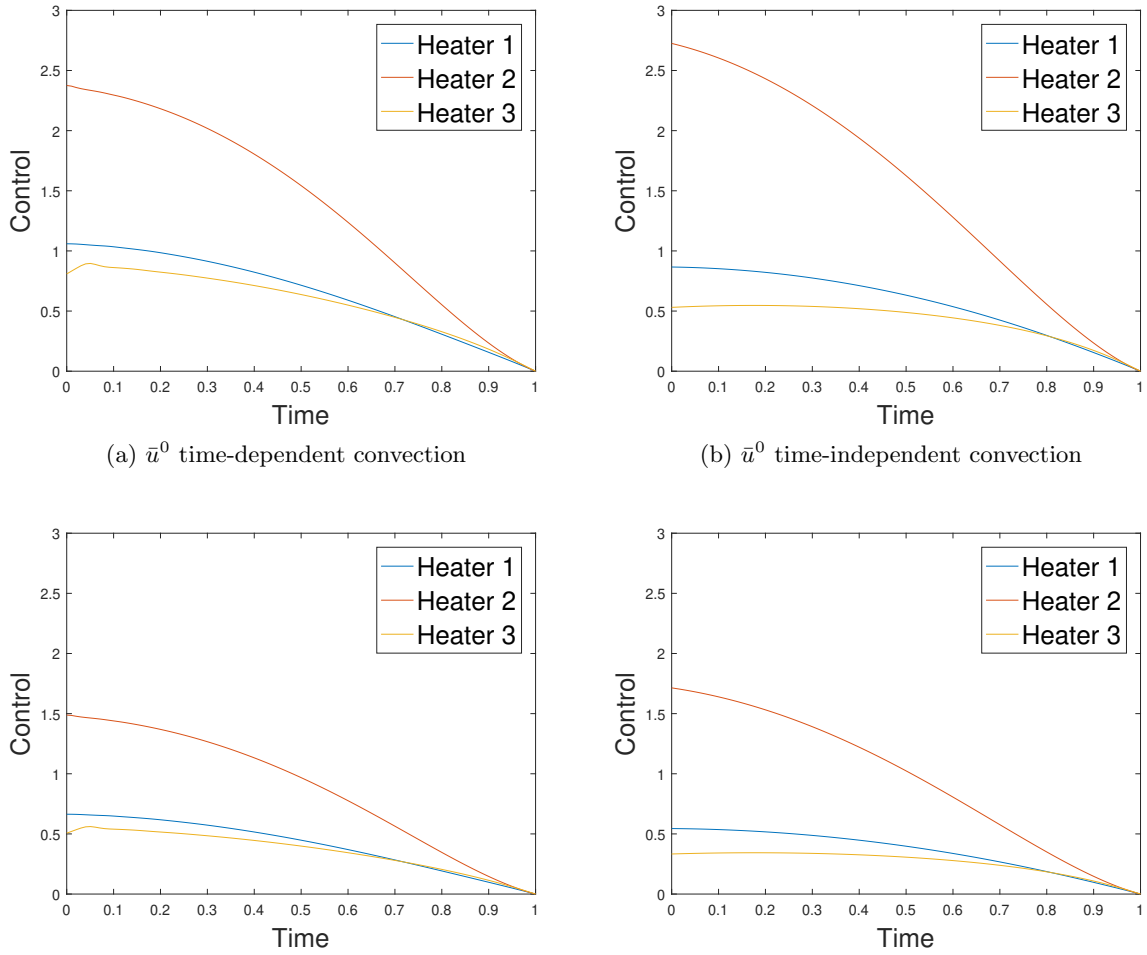


Figure 5.16: Pareto front with time-dependent convection

Once more, in Figure 5.17 we take a look at the controls corresponding to Pareto point P^0 and to a respective Pareto point in the middle of the Pareto front. Apparently, the heating strategies have changed noticeably compared to the time-independent case. With time-dependent convection, we can observe an initially wavelike behaviour of the heating strategy for heater three.

Overall heater two is heating slightly less in presence of the time-dependent convection, whereas the other two heaters are heating slightly more compared to the time-independent case.



(a) $\bar{u}^0$ time-dependent convection

(b) $\bar{u}^0$ time-independent convection

(c) $\bar{u}^{21}$ time-dependent convection

(d) $\bar{u}^{22}$ time-independent convection

Figure 5.17: Comparison of the controls of different Pareto points

Turning back to the application of POD to the problem with time-dependent convection we would like to get an impression of how the length of the POD basis affects the a-posteriori error. Therefore, we conduct the following procedure: First, we compute the whole Pareto front and the corresponding reference points by using the full finite element problem to obtain the optimal controls $\bar{u}^n_{Full}$ and the reference points z^n for $n \in \{1, \dots, N_P\}$, where N_P is the total number of solved reference point problems. At the same time, we save the reference points in order to use them for the comparison of solutions corresponding to different lengths of POD bases. This approach is necessary since different starting values for the length of the POD basis would yield different reference points and would therefore render the results incomparable. Now, with these predefined reference points, we compute the Pareto front with the POD problem for different constant numbers of POD basis functions, i.e. $\ell \in \{5, 10, 15, 20, 30\}$ to obtain the solutions

$\bar{u}^n_{POD}$ for $n \in \{1, \dots, N_P\}$. Additionally, we compute the a-posteriori estimate $\mu_U(\bar{u}^n_{POD}, z^n)$ according to (4.29) at each Pareto point. Finally, we compare the values of the a-posteriori estimate for the different chosen values of ℓ. In the following, let the *true error* denote the difference between the finite element solution and the POD solution. So we consider the finite element solution as the true solution in the sense that by model order reduction with POD we cannot expect a more accurate solution than the finite element solution. That is because on the one hand the POD basis is calculated from the finite element solution of the state and the adjoint equation and on the other hand the analytic solution of the problem is not available, so there is no way for us to assess if the present POD solution is closer to the analytic solution than the finite element solution.
In Figure 5.18a the described experiment is visualised. Apparently, $\ell = 5$ are too few POD basis functions to capture the complexity of the problem adequately. In this case the threshold for the a-posteriori error is exceeded for more than half of the reference point problems. For $\ell = 10$ the a-posteriori error has already improved considerably. The error stays below the desired bound for most of the Pareto points. In case $\ell = 15$, the a-posteriori estimate is smaller than $4 \cdot 10^{-4}$ for all but one Pareto point. If ℓ is even greater we do not achieve significant improvement any more. An important observation is that no matter how large we choose ℓ, we cannot achieve that the a-posteriori estimate stays below the desired bound for every Pareto point. This is where basis updates become a useful tool, since it possibly allows for further improvement in terms of the a-posteriori estimate.
Moreover, we would not only like to know how the length of the POD basis affects the a-posteriori estimate but also how it is related to the true error. For this reason, we take a closer look at the efficiency of the a-posteriori estimate which is defined below.

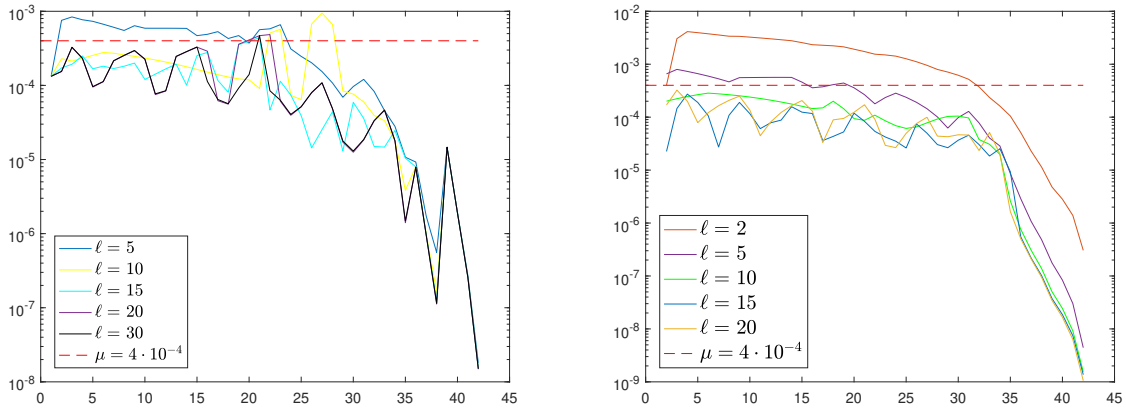
Definition 5.1. Let $a, b \in \mathbb{R}_{\geq 0}$ with $a \leq b$, b being an overestimate of a. We denote the ratio

$$\eta := \frac{b}{a} \tag{5.2}$$

by the *estimate efficiency*. Consequently it holds $\eta \in [1, \infty)$, where $\eta = 1$ represents the best possible efficiency.

Now, we apply this notion of efficiency to the a-posteriori estimate. One way to achieve this is to calculate the true error in the control space, i.e. the difference of the control, computed from the full finite element system to the corresponding control computed from the POD problem. Then we compare this error with the a-posteriori estimate, more precisely we calculate the quotient $\frac{\mu_U(\bar{u}^n_{POD}, z^n)}{\|\bar{u}^n_{POD} - \bar{u}^n_{Full}\|_U}$, where $\bar{u}^n_{POD}$ and $\bar{u}^n_{Full}$ have been defined above. The mentioned quotient defines an estimate efficiency in the sense of the last definition. Again, we conduct this experiment for different ℓ.
It turns out that exceeding the a-posteriori bound for one reference point problem during previous tests might have been a result of the a-posteriori estimate being inefficient, although we cannot tell this with absolute certainty since the reference points are different from the previous experiment. Nevertheless, for this experiment the true error lies below the threshold $\mu = 4 \cdot 10^{-4}$ already for $\ell = 10$ POD basis functions (cf. Figure 5.19). Therefore, no basis update is necessary. However, the computation time is not increased significantly by a single basis update, so its adverse effect is also very limited.

Furthermore, we observe in Figure 5.19 that the efficiency of the a-posteriori estimate is satisfactory for the majority of reference point problems. At worst the estimate is about ten times as large as the true error. In most cases, the efficiency is better than two. Besides, the efficiency getting worse beyond the 35th Pareto point is not critical since there is very little control input in this part of the Pareto front which results in the true error being very small anyway (cf. Figure 5.18b). So overall the a-posteriori estimate still performs pretty well in terms of assessing the quality of POD solutions by providing an upper bound for the error and controlling the execution of basis extensions. In case of doubt, the efficiency of the a-posteriori estimate has to be analysed similarly to the just presented procedure. It is worth mentioning that in comparison to the 2D problems that were investigated in [2] and [3], the efficiency of the a-posteriori estimate is relatively bad. This might be due to the heaters occupying a much smaller part of the whole space in the 3D case than for the 2D problems which possibly makes the control input harder to capture by the POD basis and therefore makes the error in the control space more difficult to handle. It remains to be investigated if there are situations where basis updates are more beneficial than for the example we have just studied.
As an additional insight, we found out that basis extensions are only reasonable up to a certain basis length. That is, at some point we cannot expect further improvement in terms of both the a-posteriori estimate and the true error by extending the POD basis. For $\ell > 20$ both the a-posteriori estimate and the true error are identical to those with $\ell = 20$ for almost every reference point problem. Therefore, the error for $\ell > 20$ is not visualised separately.



(a) A-posteriori estimate for various values of ℓ

(b) True error in the control space

Figure 5.18: A-posteriori estimate next to the error in the control space

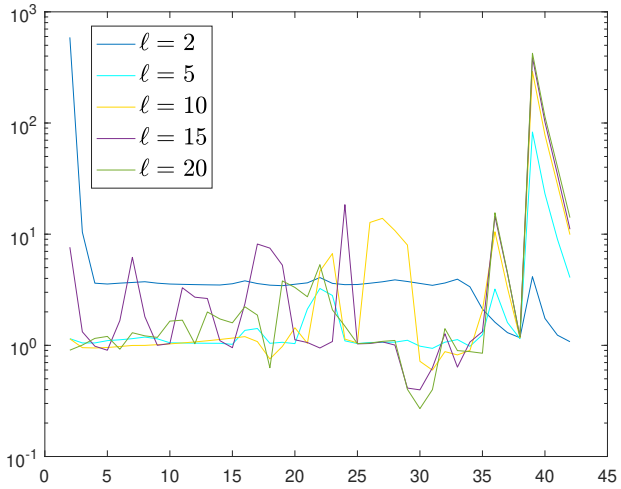


Figure 5.19: Efficiency of the a-posteriori estimate

5.5 Model with Inhomogeneous Boundary Conditions

To get even closer towards a realistic model, we proceed by adding inhomogeneous boundary conditions to the problem of the previous Section 5.4. Specifically, we consider the following state equation.

$$\begin{aligned} y_t(t,x) - \kappa\Delta y(t,x) + c_b b(t,x)\cdot\nabla y(t,x) &= \textstyle\sum_{i=1}^m u_i(t)\chi_i(x) && \text{für } (t,x)\in(0,1)\times\Omega \\ \tfrac{\partial y}{\partial \eta}(t,x) + \alpha_i y(t,x) &= \alpha_i y_a^i(t) && \text{für } (t,x)\in(0,1)\times\Gamma_i \\ y(0,x) &= y_0(x) && \text{für } x\in\Omega, \end{aligned} \tag{SEIB}$$

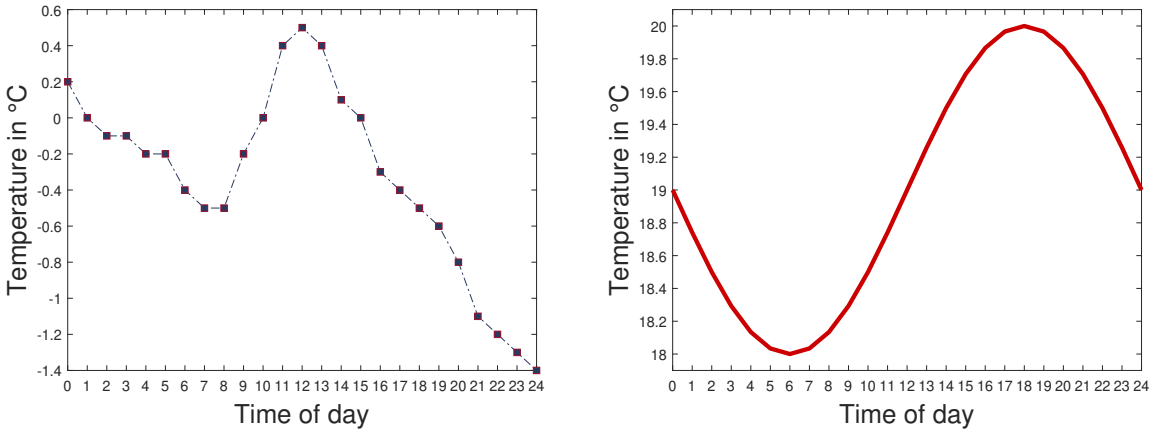
We assume these equations to hold for $i \in \{1,\dots,s\}$, where $s \in \mathbb{N}$ is the number of boundary segments, which will be described in more detail below along with the corresponding isolation coefficients α_i and the outside temperatures y_a^i. In comparison to the problem of the preceding section, the terms $\alpha_i y(t,x)$ and $\alpha_i y_a^i(t)$ were added, where α_i models the isolation of the walls of our room and $y_a^i(t)$ stands for the outside temperature at the respective part of the boundary Γ_i. Apart from the changes that are mentioned in this section, we maintain the same assumptions on the problem as described in Section 5.2 while considering the time-dependent convection term from Section 5.4.

Previously the convection was only able to move around the air inside the room, now, with inhomogeneous boundary conditions the outside temperature has an influence on the inside temperature. The magnitude of this influence is determined by the difference between the inside and the outside temperature and by the isolation coefficients α_i (cf. boundary condition of (SEIB)). Now we are able to model windows and also walls that are not perfectly isolated as well as other distinct parts of the wall of a room like doors. The prevalent impact of the outside temperature makes the problem more realistic and interesting. So in the following, we will model open windows by setting $\alpha_i = 1$ on parts of the boundary of the room. We place two of those windows in the areas $W_1 := \{0\}\times[0.8,1]\times[0.55,0.85]$ and $W_2 := \{0\}\times[0,0.2]\times[0.55,0.85]$, respectively. Furthermore, we would like to implement inner walls which border on other rooms

or building parts, as well as outer walls that border on the exterior of the building directly. In order to save heating costs, we assume better isolation at the outer walls ($\alpha = 0.1$) than at the inner walls ($\alpha = 0.2$). If we visualise this room like in Figure 5.22, the walls $F1, F2, F4$ are supposed to be inner walls and $F3, F5, F6$ are supposed to be outer walls. Moreover, we implement a door in the area $[0.6, 0.9] \times \{1\} \times [0, 0.8]$ by setting $\alpha = 0.3$. Besides, we choose different outside temperatures at the outer walls and at the inner walls, respectively. For the inner walls we choose an outside temperature $y_a(t) = 19 + \sin(2\pi(t - 1/2))$ for $t \in [0, 1]$, whereby t is supposed to correspond to a time during a regular day. In our model, the time interval $[0, 1]$ is supposed to correspond to an entire day from midnight to midnight. So the temperature takes a sinusoidal course with the lowest temperature of 18 degree occurring at $t = 1/4$ which corresponds to six in the morning and the highest temperature of 20 degree occurring at $t = 3/4$ which corresponds to six in the afternoon. This should simulate a typical heating routine, where the heaters are turned on at six in the morning and are turned off at six in the afternoon. For the outer walls, we choose an outside temperature that could occur realistically during a real winter day in Konstanz. Therefore we use historical data of a weather station (source: website of the *Climate Data Center des deutschen Wetterdienstes*, "https://www.dwd.de/DE/leistungen/cdcftpmesswerte/cdcftpmesswerte.html?nn=16102"). For all subsequent numerical tests, we will choose the outside temperature at different times according to Table 5.6 and Figure 5.20b, respectively.

Table 5.6: Outside temperature at every full hour of the day

Time of the day	0	1	2	3	4	5	6	7	8	9	10	11	12
°C	0.2	0	-0.1	-0.1	-0.2	-0.2	-0.4	-0.5	-0.5	-0.2	0	0.4	0.5
Time of the day	13	14	15	16	17	18	19	20	21	22	23	24	
°C	0.4	0.1	0	-0.3	-0.4	-0.5	-0.6	-0.8	-1.1	-1.2	-1.3	-1.4	



(a) Outside temperature at the outer walls

(b) Outside temperature at the inner walls

Figure 5.20: Outer temperatures of the model

The entire model is visualised in Figure 5.22. Taking another look at Figure 5.11, we recognise

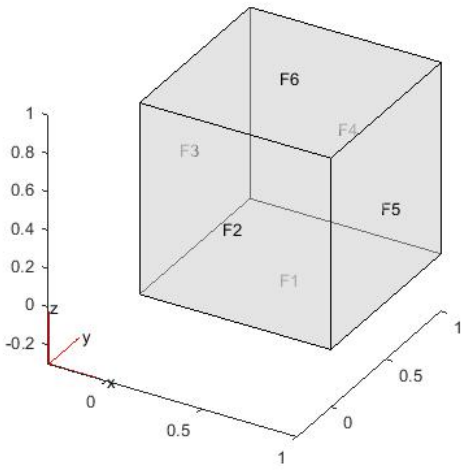


Figure 5.21: Spatial layout of the walls

that the convection is essentially a flow from one window to the other. Additionally, there is a vortex as depicted in Figure 5.11. However, the convection is constant over the entire height of the room and does not just act at the height of the windows.

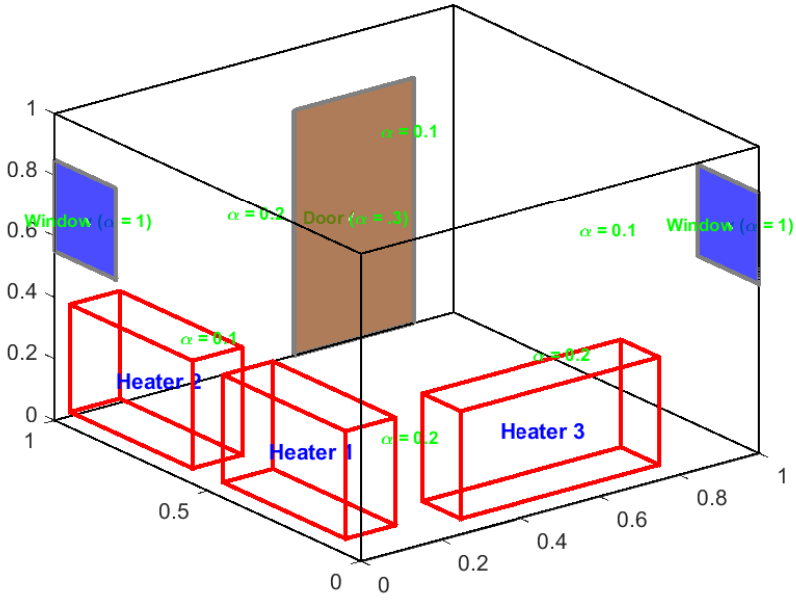


Figure 5.22: Model with inhomogeneous boundary conditions

We conduct a test run with the step sizes $h_x = 0.15, h_p = 0.75$, which yields the Pareto front in Figure 5.23. The Pareto points range in between the outer Pareto points $P^0 = (4.366, 4.9542)$

and $P^{41} = (4.596, 0)$, which correspond to the minimal values of $\hat{J}_1(u) + 0.02\hat{J}_2(u)$ and $\hat{J}_2(u)$, respectively. We notice immediately that the optimal value of the functional $\hat{J}_1$, which corresponds to the top Pareto point, is considerably greater than for the previously tested models, which means that the desired temperature is not reached as closely as for the models with homogeneous boundary conditions. This is obviously due to the influence of the cold air from the outside.

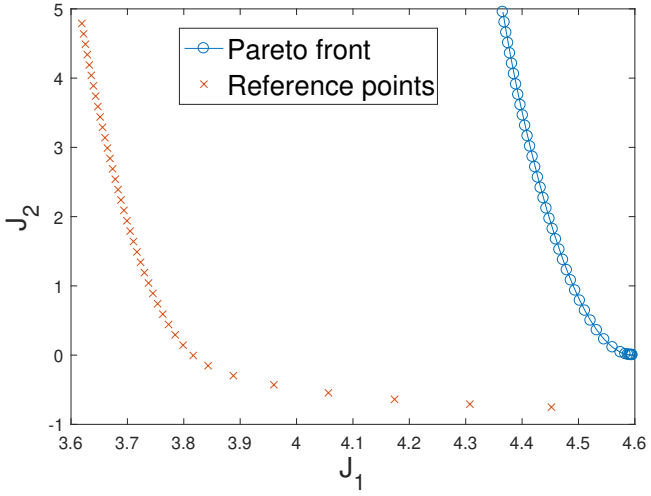


Figure 5.23: Pareto front with inhomogeneous boundary condition

Taking a closer look at the corresponding state $\hat{y} + Su_1$ at time $t = 1$ (cf. Figure 5.24)), we realise that the room is actually cooling down although the heaters are turned on, which means the controls are greater than zero. More specifically, the room cools down the most close to the windows of the model and at places where the cold air is moved by the convection. Only close to the heaters the temperature can be maintained higher. Consequently, the heaters in this model are too small and therefore not powerful enough to compensate for the cold air coming from the outside on a typical winter day. However, this corresponds to observations in real life, where a usual heated room cools down if the windows are kept open entirely for an extended period of time.

Considering the controls corresponding to the top Pareto point (see Figure 5.25) it is apparent that, apart from u_2, they lie significantly below the upper bound $u_b = 3$, although heating more would obviously lead to a better approximation of the desired temperature. The reason for this is the weight 0.02 on the heating costs, i.e. on the functional $\hat{J}_2$. Apparently, by heating more $\hat{J}_2$ would increase to such an extent that during the optimization higher values for the respective controls are prevented by the weight on $\hat{J}_2$.

Now, we conduct a short comparison of the performance of the optimization algorithm for the three different examples we have investigated so far. The results for the computation of the full finite element problems can be seen in Table 5.7. While the problems with homogeneous boundary conditions take a similar time to compute, the problem with inhomogeneous boundary conditions needs 47% more time than the corresponding problem with homogeneous boundary conditions, although a comparable number of Pareto points is computed. Taking into con-

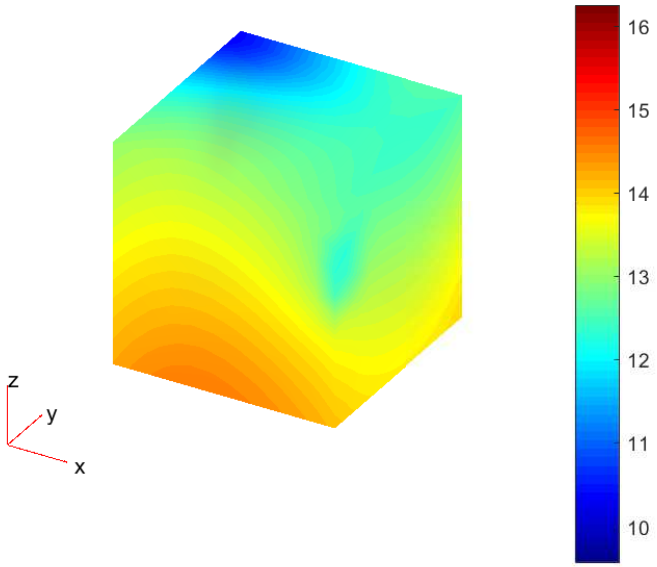


Figure 5.24: State at the top Pareto point for $t = 1$

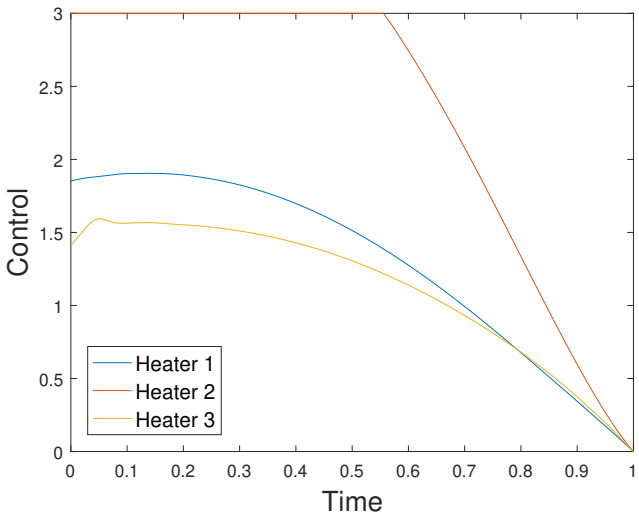


Figure 5.25: Controls corresponding to the top Pareto point

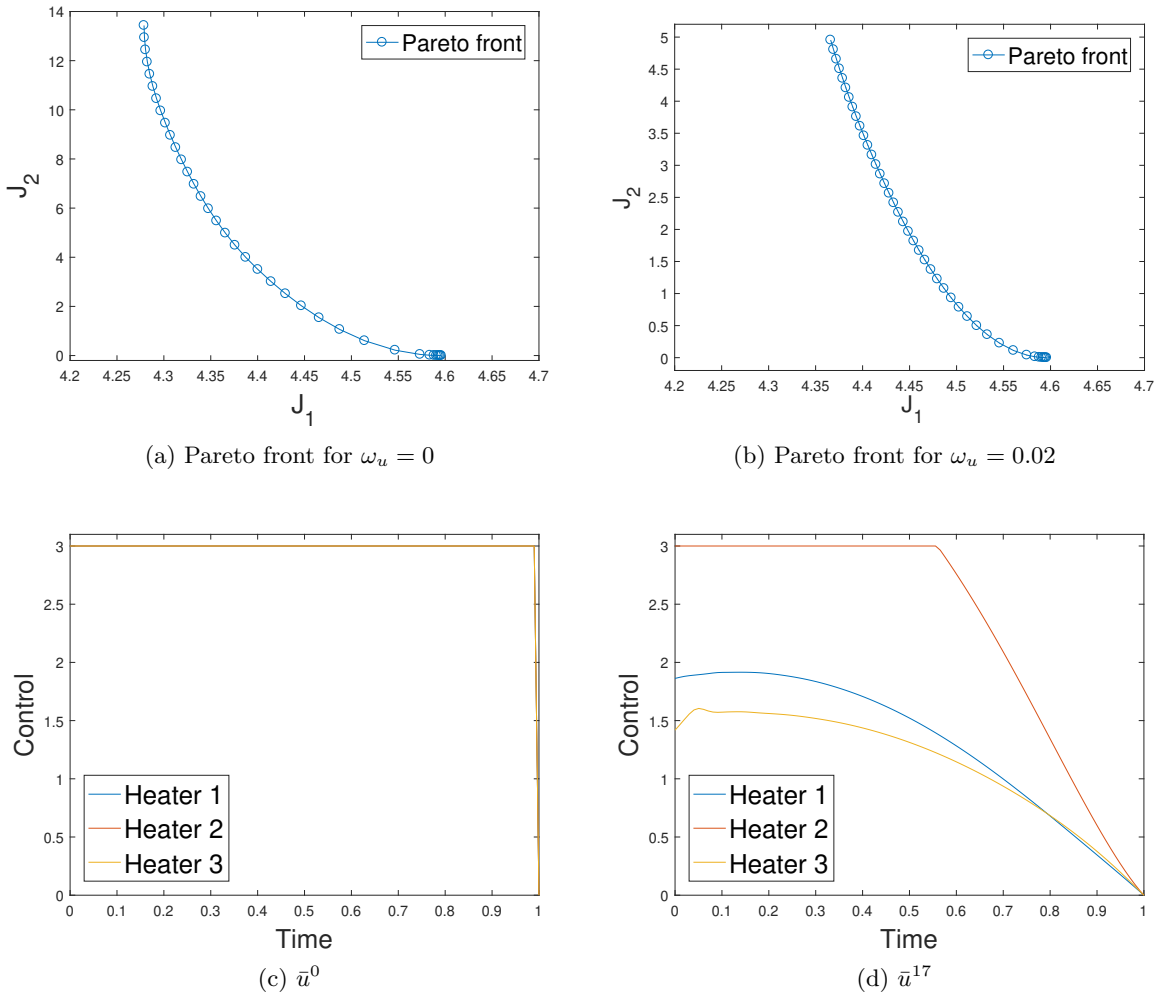
sideration the average number of optimization iterations in column four, we notice that the number of iterations is almost the same for both the problem with and without inhomogeneous boundary conditions. Actually, if we take a closer look at the optimization, it turns out that for the problem with inhomogeneous boundary conditions, the optimization step size has to be reduced repeatedly by the algorithm as opposed to the problem with homogeneous boundary

conditions. This is computationally expensive and explains the apparent difference in computation time. Evidently, the Pareto optimal points are harder to compute for the problem with inhomogeneous boundary conditions.

Table 5.7: Algorithm performance for different models

Model	Number of Pareto points	Average number of iterations per Pareto point	Average comp. time per Pareto point	Comp. time
time-indep. conv.	46	4	18.74	862s
time-dep. conv.	44	3.74	17.77	782s
time-dep. conv., inh. bound. cond.	42	3.75	27.45	1153s

As a further step, to achieve a better approximation of the desired temperature y_Q, we try to recompute the Pareto front while omitting the weight on $\hat{J}_2$ for the first Pareto point, so we minimize $\hat{J}_1$ without paying attention to the heating costs. Taking into account that for this modified problem the heating costs will probably be considerably higher for the top Pareto point we enlarge the step sizes h_p and h_x: We set $h_x = 0.5$ and $h_p = 2.5$. Indeed, by this procedure, we get a better approximation of the desired temperature. Now the top Pareto point is $P^0 = (4.2789, 13.4318)$. So the value of $\hat{J}_1$ has decreased by about 0.09, whereas the heating costs have increased by more than eight in the sense of $\hat{J}_2$. Consequently, we have to accept considerably higher heating costs in order to achieve a relatively small improvement of $\hat{J}_1$. If we recall the previous problem in which a small weight was applied to the second objective, we now have a better understanding, why the controls were heating that much less than the upper bound at the top Pareto point (cf. Figure 5.25). Namely because at that point further heating effort does not achieve much better fulfillment of the first objective. This observation is also reflected by the shape of the Pareto front, in particular by the part of the Pareto front which has been computed additionally to the previous example, i.e. the part where $\hat{J}_2 > 5$. That is to say that this part of the Pareto front is very steep in comparison to the lower part of the Pareto front. Another aspect of setting $\omega_2 = 0$ is that the computation of 36 Pareto points took 3155 seconds for the full problem with an average number of Newton-CG iterations per Pareto point of 9.71. If we compare this to the results in Table 5.7, we notice that the computational effort is more than three times as high if the average computation time per Pareto point is considered and that the average number of Newton-CG iterations per Pareto point is more than 2.5 times as high as opposed to using the weight $\omega_2 = 0.02$ to determine the top Pareto point. Later on, we will look for a strategy to achieve a faster computation of the Pareto front.
With this modified objective functional we actually achieve to use the heaters to their full capacity, that is all controls are almost constant at the upper bound $u_b = 3$. This is apparent from the plot of the controls corresponding to the top Pareto point, see Figure 5.26c. For comparison, we perform a test run with upper bound $u_b = 5$. As expected, the heating costs, that is the value of $\hat{J}_2$, corresponding to the top Pareto point increase considerably. In this case the top Pareto point is $P^1 = (4.0756, 37.3106)$. This means that to achieve an improvement of $\hat{J}_1$ by 0.2, we have to accept heating costs that are higher by 24. Again, the controls lie almost completely at the upper bound, while the desired temperature is still far from being reached which suggests that heating more could still improve the approximation of the desired



(a) Pareto front for $\omega_u = 0$

(b) Pareto front for $\omega_u = 0.02$

(c) $\bar{u}^0$

(d) $\bar{u}^{17}$

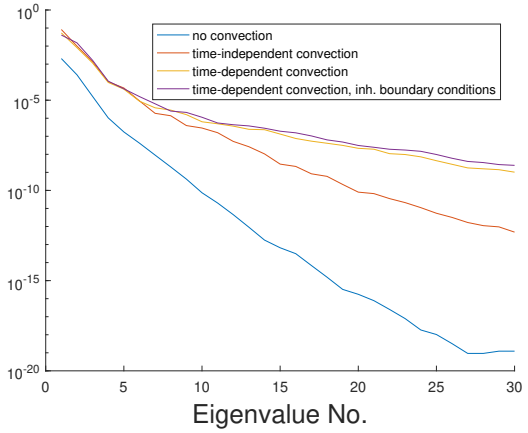
Figure 5.26: Results for $\omega_u = 0$

temperature. However, in reality, heaters also have limited heating capability, so we refrain from increasing the upper bound further at this point. For further experiments we stick with $u_b = 3$.

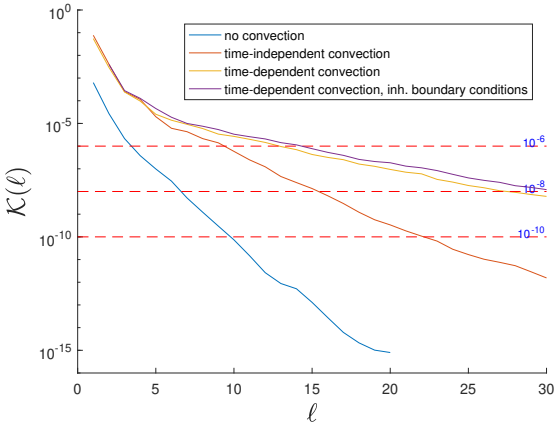
Overall we can conclude from the shape of the Pareto front, that the use of heating diminishes more while the heating output increases. This is reflected by the Pareto front being relatively steep and showing little curvature in the top part and increasing curvature in its lower part.

Now, we would like to investigate how adding the inhomogeneous boundary conditions affects the use of POD. To begin with, we calculate the eigenvalues corresponding to the initial POD basis and compare them to those of the previous models. Note that for this comparison between different models we used the weight 0.02 on the second objective for all problems, such that the results are better comparable. Nevertheless, it can be observed that for a given problem, the eigenvalues corresponding to different POD bases behave very similarly unless the control input becomes very small, cf. Figure 5.27c. This observation suggests that the qualitative behaviour of the eigenvalues is in some sense characteristic of a given problem. Taking a look at Figure

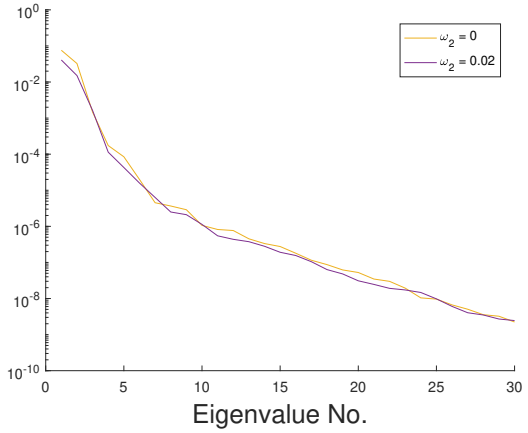
5.27a, we notice that the eigenvalues behave similarly to those of the problem with homogeneous boundary conditions. On a logarithmic scale, these two sets of eigenvalues appear pretty much parallel, that means they differ by an approximately constant factor. If we compare the two criteria used to determine the number of POD basis functions, omitting the weight on $\hat{J}_2$ again, we recognise that they take a similar course for $\ell \geq 5$, the difference being a factor of about ten. So we expect that using the same ε while applying criterion $\mathcal{K}$ leads to a lower number of initial POD basis functions than using $\mathcal{E}$.



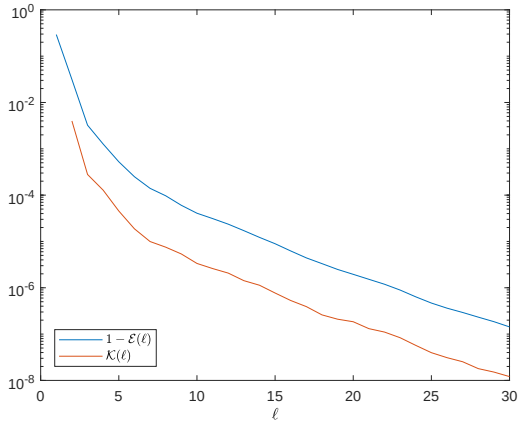
(a) Behaviour of the eigenvalues for different models



(b) $\mathcal{K}(\ell)$ for different models



(c) Eigenvalues for different weights on $\hat{J}_2$

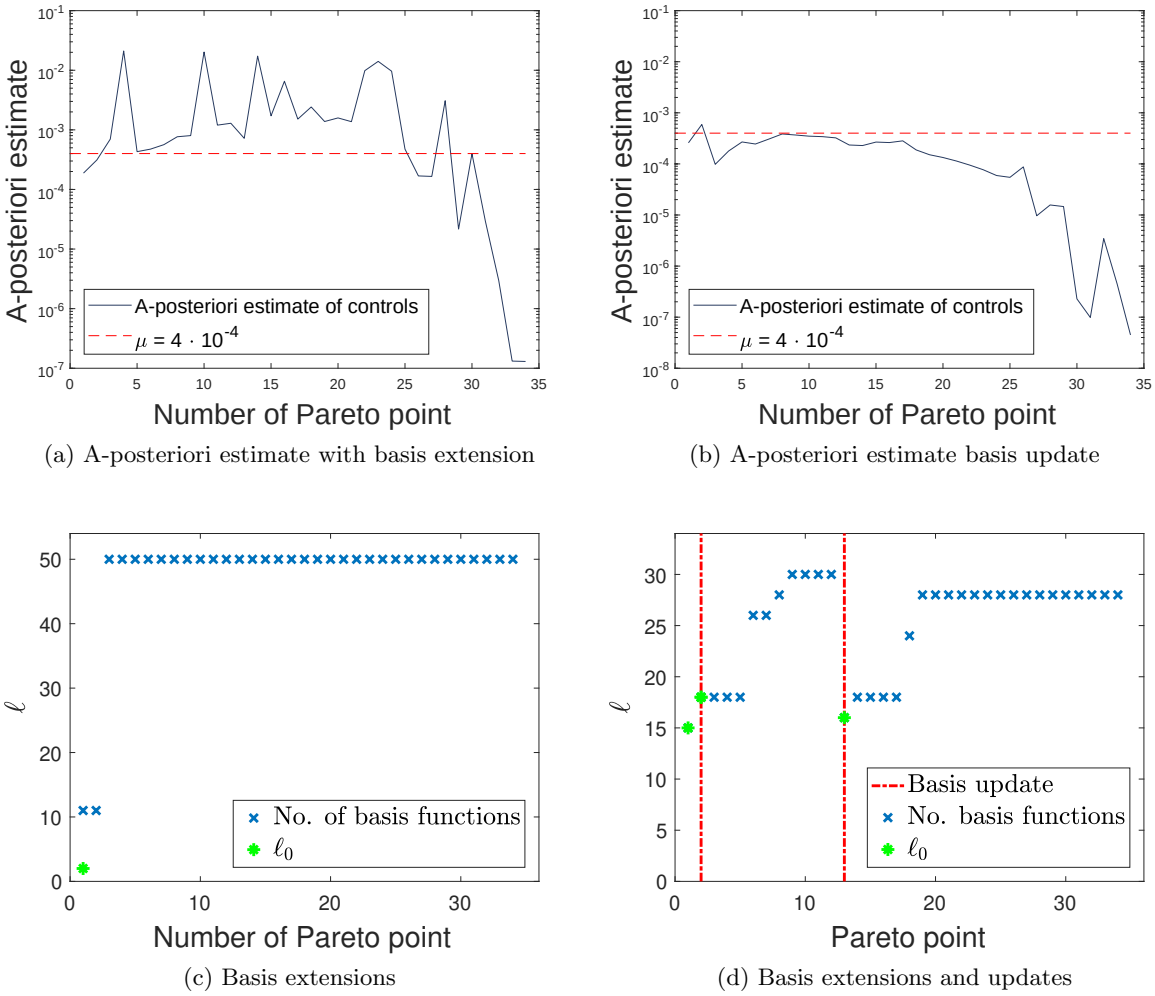


(d) Comparison of two criteria

Figure 5.27: Behaviour of eigenvalues and different error estimates

To begin with, we test our simple basis extension Algorithm 3 with parameters $\ell_0 = 2$, $\ell_{max} = 50$ to check if basis updates might be necessary at all. Indeed, we observe in Figure 5.28a that even if the POD basis is extended up to a length of 50 POD basis functions the bound for the a-posteriori estimate is exceeded significantly. For comparison we also test the basis update Algorithm 4 with parameters $\ell_{max} = 50$ and $\varepsilon = 10^{-6}$ for the calculation of ℓ_0 according to (5.1). Looking at the results for the basis update algorithm in Figure 5.28b, we recognise noticable improvement.

For all but one Pareto point the a-posteriori bound is satisfied and for the remaining reference point problem the full system is used according to Algorithm 4, in order to achieve a satisfactory result. Until further notice, we maintain the aforementioned parameters for the basis update algorithm.



(a) A-posteriori estimate with basis extension

(b) A-posteriori estimate basis update

(c) Basis extensions

(d) Basis extensions and updates

Figure 5.28: Comparison of basis extension and basis update algorithm

A side effect of omitting the weight on the second objective for the computation of the top Pareto point, is that using the current step sizes $h_x = 0.5, h_p = 2.5$, we need a lot of Newton iterations and therefore extensive computation time for the Euclidean reference point problems corresponding to the upper part of the Pareto front (cf. Figure 5.29a). That is why we turn towards the investigation of another parameter, namely the step size h_p. This step size determines how far away the reference points are chosen parallel to the Pareto front. Interestingly, the optimization algorithm seems to converge in fewer optimization steps, if we increase h_p and leave the other step size h_x the same. We conduct a test to find out more details about this effect. More precisely, we keep $h_x = 0.5$ constant and run the test for different values of h_p.

Moreover, we perform this test for both the full and the POD problem, whereby the basis update Algorithm 4 with $\ell_{max} = 50$ and ℓ_0 being determined by criterion (5.1) with $\varepsilon = 10^{-6}$ is used. The maximum number of Newton-CG iterations per reference point problem is set to 30. Consequently, we get the results shown in Table 5.8 for the full problem and in Table 5.9 for the POD problem. Obviously, a higher value for h_p leads to a higher number of Pareto points. This is mainly because of the curvature increasing strongly in the bottom part of the Pareto front and therefore many Pareto points being very close to each other in that area. This effect is reinforced the further away the reference points are from the Pareto front, which is largely determined by h_p, since this causes the reference point to pass through a longer arc around the Pareto front. For this reason, when running the test for different values of h_p, we only take into account the first 20 Pareto points starting at the top if we compare the number of optimization iterations or the computation time per Pareto point. This is important, because the aforementioned large number of Pareto points lying very close to each other, which occurs for relatively high values of h_p, can skew the averages we want to compare. One aspect of this problem is that the reference point problems at the bottom of the Pareto front usually take only one or two newton iterations to solve. Another thing to consider is that the upper 20 Pareto points are very similar for different values of h_p (cf. Figure 5.30c), since the distance between the Pareto points in this part of the Pareto front is mainly determined by the other step size parameter h_x. For the purpose of visualisation, we compare the computation time in Figure 5.29a and the number of Newton-CG iterations in Figure 5.29b for the respective reference point problems and for two different values of h_p. Among other things, we recognise from these plots that the first 20 reference point problems are most interesting for the purpose of comparison between different values of h_p. Beyond these first 20 reference point problems both the number of required iterations and the computation time decrease quickly. This applies very similarly for different values of h_p.

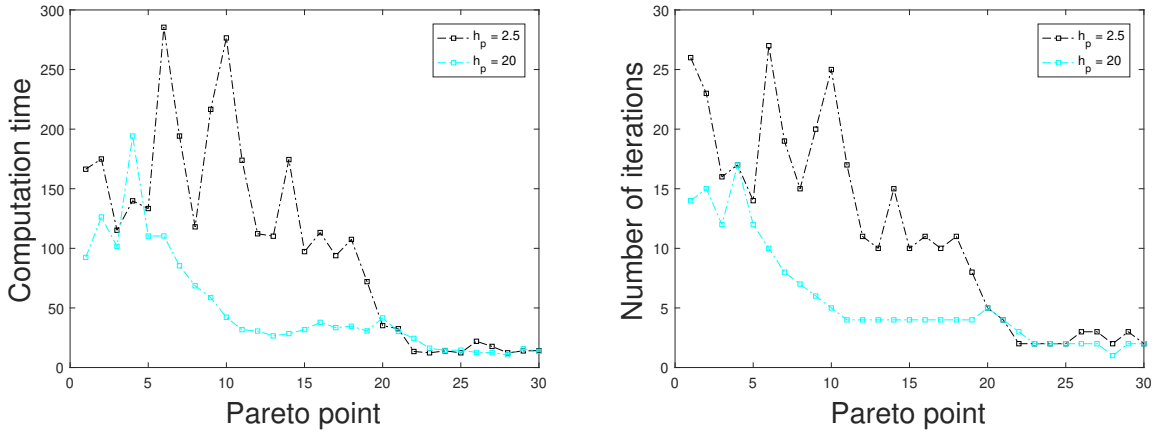
Table 5.8: Test for different h_p solving the full problem

h_p	Number of Pareto points	Average number of iterations per Pareto point	Average comp. time per Pareto point	Comp. time
0.5	30	21.5	182.28s	3791s
1	32	20.1	174.37s	3642s
2.5	36	15.5	145.52s	3155s
5	44	11.4	121.91s	2718s
10	60	9.5	88.89s	2202s
20	91	7.4	65.88s	2037s
40	101	5.3	52.53s	1904s

From Table 5.8 we recognise that the total computation time decreases every time we increase h_p, although the number of computed Pareto points is increased thereby. Clearly, the number of necessary Newton-CG iterations also decreases as well as the average computation time per reference point problem. If POD is used, the general trend is similar, especially if we consider computation time and Newton iterations per Pareto point. However, with POD, the computation time gets worse if we increase h_p from 20 to 40. The reason for this is that in this case the optimization routine apparently does not converge to a sufficiently accurate solution in terms

Table 5.9: Test for different h_p solving the POD-problem

h_p	Number of Pareto points	Average number of iterations per Pareto point	Average comp. time per Pareto point	Comp. time	Basis updates
0.5	30	20.9	18.93s	469.3s	4
1	32	18.1	20.38s	469.9s	4
2.5	36	16.8	15.04s	367.6s	2
5	44	11.6	13.21s	343.9s	2
10	60	9.4	11.62s	340.3s	1
20	91	7.55	9.1s	365.2s	3
40	101	5.3	10.42s	393.1s	2



(a) Computation time per Pareto point

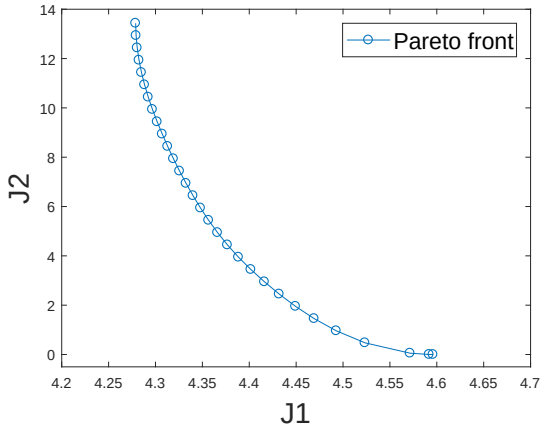
(b) Newton-CG iterations per Pareto point

Figure 5.29: Impact of h_p on the full problem

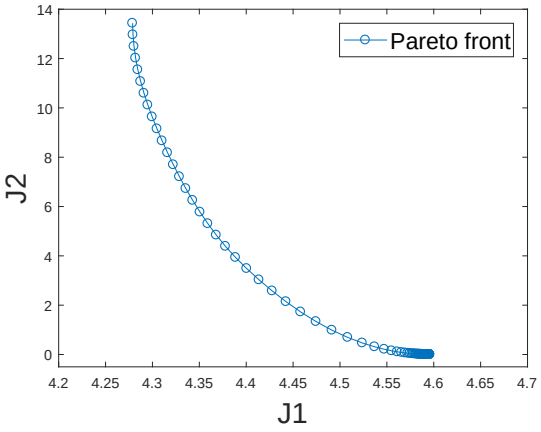
of the a-posteriori estimate for some of the reference point problems. This causes the POD Algorithm 4 to use the full system to solve the reference point problem. Still, the average computation time per Pareto point is better than for $h_p = 10$. These observations suggest that $h_p = 20$ is a favourable choice if POD is used and for the full system it is still far better than the original choice $h_p = 2.5$, so until further notice we will use fixed $h_p = 20$ for both our experiments with the full problem and the POD problem.

As opposed to the described problem at the bottom part of the Pareto front for larger h_p, the fineness of the approximation in the upper part of the Pareto front remains almost unchanged if h_p is modified, because this adverse effect is largely dependent on the current curvature of the Pareto front. This is immediately apparent from Figure 5.30 where the Pareto fronts for $h_p = 0.5$ and $h_p = 20$, as well as the distance between consecutive Pareto points is shown. For the first 20 Pareto points the distance between the Pareto points only differs by 5% on average. It is only afterwards that the deviation increases which is due to the increasing curvature. This realization about the distance between the Pareto points in the upper part of the Pareto front also suggests that the improved convergence is not due to the Pareto points lying closer to each other, because

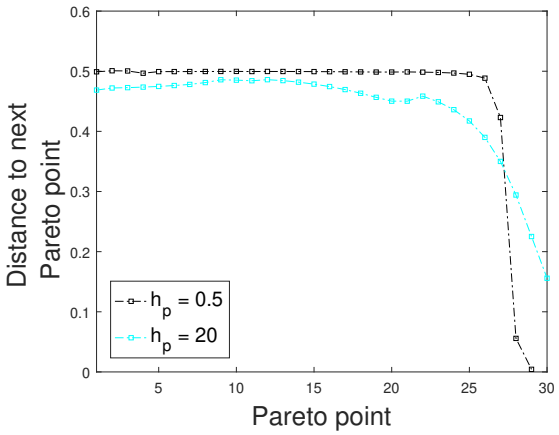
the distance does not change significantly. So far, we do not have a theoretical explanation for the faster convergence with an increased h_p other than it seems to somehow improve the conditioning of the resulting reference point problem with regard to the optimization algorithm. For the upper part the described behaviour of the distance between consecutive Pareto points is desirable because we would like to keep approximately the same level of approximation fineness and at the same time, we still would like to improve the convergence speed of the solution of the reference point problems by using the aforementioned approach of increasing h_p. However, the fineness of the approximation of the Pareto front is best controlled by the parameter h_x.



(a) Pareto front for $h_p = 0.5$



(b) Pareto front for $h_p = 20$

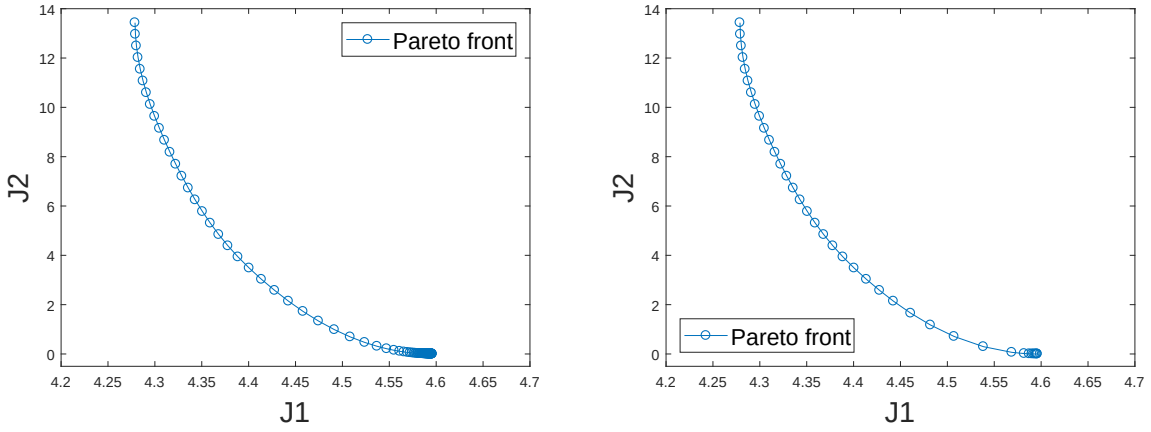


(c) Comparison of the distance between Pareto points

Figure 5.30: Comparison of the Pareto fronts and the distance between Pareto points for different h_p

In order to benefit from the faster convergence with increased h_p on the one hand but on the other hand to avoid computing unnecessarily many Pareto points at the bottom of the Pareto front, we propose a routine to reduce h_p under certain conditions. Namely, we specify that the algorithm should reduce h_p after the solution of the current reference point problem has only

taken few optimization iterations, for example, two or fewer iterations. This is a reasonable criterion since in the bottom part of the Pareto front the optimization needs fewer iterations than in the upper part of the Pareto front. Generally speaking, while passing the Pareto front downwards, the number of iterations decreases gradually (cf. Figure 5.29b). The last few reference point problems usually only need a single iteration. To reduce the probability of the optimization algorithm needing few iterations only being a coincidence, we will only reduce h_p if the optimization needed few iterations for two consecutive reference point problems. If we test this routine, setting the mentioned threshold of iterations to two, we get the Pareto front in Figure 5.31b.



(a) Pareto front for fixed $h_p = 20$

(b) Pareto front with adaptive h_p

Figure 5.31

In Figure 5.31 we take a look at the respective Pareto fronts for the two different strategies. In particular, Figure 5.31a shows the Pareto front for fixed $h_p = 20$. It is clearly noticeable that a lot of Pareto points, lying very close to each other, are computed at the bottom of the Pareto front. In total 91 Pareto points are calculated and the computation time is 322 seconds. In contrast, we see the Pareto front, where h_p is reduced according to the discussed criterion, in Figure 5.31b. This time, 38 Pareto point are calculated and the program computation time is 229 seconds. Furthermore, h_p was reduced after the computation of the 25th Pareto point, which is apparent in Figure 5.31b by the approximation getting coarser at some point. Additionally, far fewer Pareto points are calculated at the bottom of the Pareto front, so that an unnecessarily dense approximation in this part of the Pareto front is avoided. If one would like to get a finer approximation in the bottom part of the Pareto front, one could take into consideration to modify h_x along with h_p. Summing up, by adapting h_p in the aforementioned way, we managed to reduce the number of unnecessarily computed Pareto points and are still profiting from the better convergence in the upper part of the Pareto front.
To generalize the procedure, one might try to adapt h_p automatically, depending on the convergence speed for the current reference point problem. More precisely, we monitor the number of needed optimization iterations for the current reference point problem. If this number exceeds a certain threshold, e.g. 20 iterations, the optimization is terminated, we increase h_p and restart the optimization for the current reference point problem. This process is repeated until

convergence is achieved within the desired number of iterations or until an upper bound h_p^{max} for h_p is reached. Although this approach cannot guarantee success with regard to convergence speed, it does help for most of the tested examples whereas it almost never has a negative effect on convergence speed, if the upper bound h_p^{max} is chosen carefully.

To improve our chances of success we should also ensure not to choose a too small number of maximum iterations since for a small number of maximum iterations it might not even be possible to reach that goal. Furthermore, the purpose of increasing h_p is mainly to help with issues of very slow convergence or no convergence at all. Curiously, for too small values of h_p one often observes that in the top part of the Pareto front the optimization does not converge at all, no matter how many iterations it performs. In these cases, the optimization routine seems to run in circles and no satisfactory descent is achieved. Possibly, small values of h_p render the problem ill-conditioned. Specifically, if POD is used, one should ensure that a sufficient number of POD basis functions is used, since too few POD basis functions might also have an adverse effect on the speed of convergence of the optimization routine. We try to avoid this by choosing an adequate ℓ_0 with criteria we have already discussed.

Overall, we can combine this routine to increase h_p in the upper part of the Pareto front and the previously discussed strategy to reduce h_p in the bottom part of the Pareto front in order to get an algorithm that on the hand improves speed of convergence, adapted to the present problem, and on the other hand avoids the problem of computing unnecessarily many Pareto points in the bottom part of the Pareto front.

We summarize the described procedure in Algorithm 5. In Algorithm 5, $\hat{J} = (\hat{J}_1, \hat{J}_2)$ denotes the objective functional with two components. Note that the algorithm is not restricted to the specific objective we have defined in this chapter. Furthermore we write $z^i = (z_1^i, z_2^i)$, $i = 1, \ldots, N_{max}$ for the reference points, where N_{max} is a set maximum number of reference points. In addition, every time we solve a reference point problem, we check if the maximum number of optimization iterations is reached. If this is the case, we set maxitflag $= 1$, otherwise we set maxitflag $= 0$. After solving the n-th reference point problem, we save the respective number of needed optimization iterations as $it(n)$. Algorithm 5 is an extension of Algorithm 1 and we will only state it for the full problem here. The respective algorithm for the POD problem can be constructed in the same fashion as Algorithm 2 was deduced from Algorithm 1.

For a test run with Algorithm 5 for the POD problem we set the maximum number of iterations $it_{max} = 20$, the initial h_p to 2.5, as well as $h_p^{max} = 40$. With these parameters, we observe that h_p is increased to $h_p = 20$ during the computation of the second Pareto point and from there on h_p remains constant until it is reduced to $2.5 = 5h_x$ after the computation of the 24th Pareto point by the rules of the algorithm. Interestingly, we found out earlier that $h_p = 20$ is a favourable choice in terms of average iterations per Pareto point and average computation time per Pareto point. Of course the h_p determined by the algorithm depends on the chosen maximum number of iterations it_{max}. This result indicates, that $it_{max} = 20$ is achievable for this problem and the number of iterations is a useful criterion to determine a more favourable h_p. Taking a closer look at the results, we find that in total 38 Pareto points were calculated and we obtain the Pareto front exactly like in Figure 5.31b, since h_p is already increased for the first reference point problem. In addition, the computation time is 302 seconds, which is considerably faster than for all fixed h_p (cf. Table 5.9), even if increasing h_p in the beginning leads to several recomputations of the same Pareto point. This improvement is due to a very fast average computation time for the first 20 Pareto points of 5.68 seconds, which is only slightly

Algorithm 5 Algorithm to compute the Pareto front with adaption of h_p

Require: Maximum number $N_{\max} \in \mathbb{N}$ of Pareto points, step size parameters $h_p, h_x > 0$, upper bound $h_p^{max} > h_p$, maximum iterations per reference point problem it_{max};

Minimize the individual objectives $\hat{J}_1$ and $\hat{J}_2$ seperately to obtain $\bar{u}^0$ and $\bar{u}^L$, respectively;
Set $n = 0$ and compute z^1 using h_p, h_x;
while $n < N_{\max}$ and $z_1^{n+1} < \hat{J}_1\left(\bar{u}^L\right)$ **do**
    $n := n + 1$;
    Set maxitflag $= 1$;
    while maxitflag $= 1$ **do**
        Compute $\bar{u}^n$, $it(n)$ and maxitflag from the n-th reference point problem;
        if maxitflag $= 1$ **then**
            if $2h_p \leq h_p^{max}$ **then**
                Set $h_p = 2h_p$;
                Recompute reference point problem z^n with updated h_p;
            else
                Set maxitflag $= 0$;
    if $n \geq 3$ and $it(n-1) \leq 2$ and $it(n) \leq 2$ **then**
        Set $h_p = 5h_x$;
    Set $\tilde{P}_U = \tilde{P}_U \cup \{\bar{u}^n\}$ and $\tilde{P}_Y = \tilde{P}_Y \cup \{f(\bar{u}^n)\}$;
    Compute reference point z^{n+1} using h_p, h_x;
Return $\tilde{P}_U = \tilde{P}_U \cup \{\bar{u}^L\}$ and $\tilde{P}_Y = \tilde{P}_Y \cup \{f(\bar{u}^L)\}$.

worse than for fixed $h_p = 20$ (cf. Table 5.9) and at the same time the number of unnecessary Pareto points at the bottom of the Pareto front is greatly reduced.
Another way to keep the algorithm from computing unnecessarily many Pareto points that are very close to each other in the bottom part of the Pareto front is to adapt h_p to the curvature of the Pareto front. In particular, we assume that the Pareto front can be locally described by a circle with radius $\frac{1}{\kappa}$, where κ is the curvature of this circle. Additionally, without loss of generality let us assume that the center of this circle is $(0, 0)$. Taking into account the way consecutive Pareto points are calculated, we get the following relation for the angle φ between consecutive reference points z^i, z^{i+1}, with respect to the center of the circle:

$$\cos\varphi = \frac{\langle z^i, z^{i+1}\rangle}{\|z^i\|\|z^{i+1}\|} = \frac{1/\kappa + h_p}{((1/\kappa + h_p)^2 + h_x^2)^2}$$

Therefore we get for the distance between $\hat{J}(\bar{u}^i)$ and $\hat{J}(\bar{u}^{i+1})$:

$$\|\hat{J}(\bar{u}^i) - \hat{J}(\bar{u}^{i+1})\| = \frac{2}{\kappa}\sin\left(\frac{1}{2}\arccos\left(\frac{1/\kappa + h_p}{((1/\kappa + h_p)^2 + h_x^2)^2}\right)\right)$$

By construction of the algorithm for computing the Pareto front the maximum distance between $\hat{J}(\bar{u}^i)$ and $\hat{J}(\bar{u}^{i+1})$ is h_x. So based on that we require that the distance between consecutive Pareto points does not get too small in relation to h_x. That is we would like to ensure

$$\frac{\|\hat{J}(\bar{u}^i) - \hat{J}(\bar{u}^{i+1})\|}{h_x} = \frac{1}{h_x}\frac{2}{\kappa}\sin\left(\frac{1}{2}\arccos\left(\frac{1/\kappa + h_p}{((1/\kappa + h_p)^2 + h_x^2)^2}\right)\right) \geq \varepsilon_{tol}. \tag{5.3}$$

where the curvature κ has to be computed first. This is achieved by

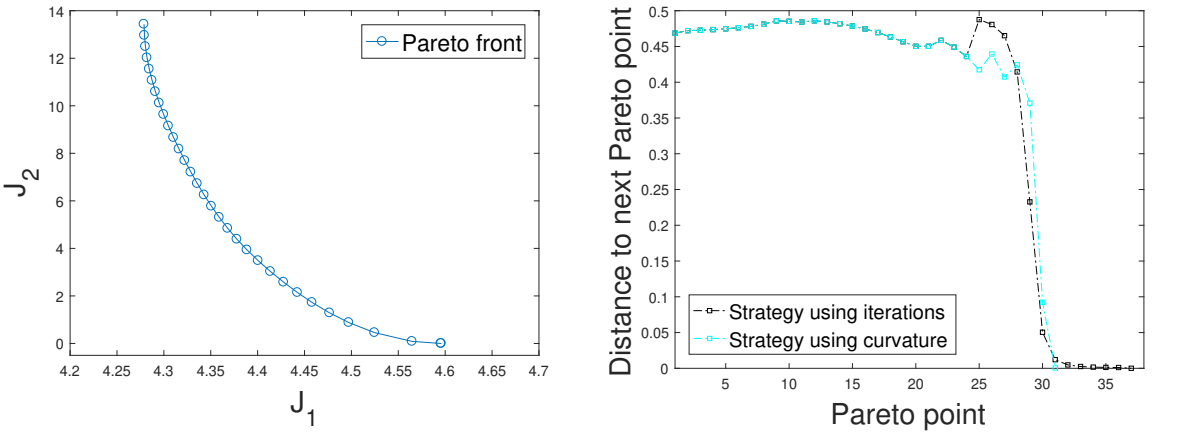
$$\kappa = \frac{1}{\hat{J}_1(\bar{u}^i) - \hat{J}_1(\bar{u}^{i-1})} \left(\frac{-(\hat{J}_1(\bar{u}^i) - z_1^i)}{(\hat{J}_2(\bar{u}^i) - z_2^i)} + \frac{(\hat{J}_1(\bar{u}^{i-1}) - z_1^{i-1})}{(\hat{J}_2(\bar{u}^{i-1}) - z_2^{i-1}))} \right) \tag{5.4}$$

which uses an approximation of the second derivative of the Pareto front by difference quotients, whereby the first derivative is already known as the term $\frac{-(\hat{J}_1(\bar{u}^i)-z_1^i)}{(\hat{J}_2(\bar{u}^i)-z_2^i)}$. More specifically, while computing the Pareto front, we check (5.3) at every Pareto point, until this term falls below a specified threshold. In this case we determine a new h_p such that (5.3) holds for an ε_{curv} and continue computing the Pareto front with this new value for h_p. We summarize the described procedure in a short algorithm, see Algorithm 6. So in practice we can replace lines 14 to 15 of Algorithm 5 by Algorithm 6 as an alternative.

Algorithm 6 Algorithm to adjust h_p

Require: Parameters $\varepsilon_{tol} \leq \varepsilon_{curv}$;
1: Compute the current curvature κ by (5.4);
2: **if not** (5.3) for ε_{tol} **then**
3: Set h_p such that (5.3) holds for ε_{curv};

In practice, we choose ε_{tol} smaller than ε_{curv} by some margin such that h_p is not adjusted too often, for example, if the left-hand side of (5.3) becomes just slightly smaller than ε_{tol} and thus h_p would have to be adjusted only by a small value and, considering the shape of the present Pareto front, also for every further reference point problem. If we do a test run for this alternative procedure with $\varepsilon_{tol} = 0.85$, $\varepsilon_{curv} = 0.9$, we get a Pareto front of 32 Pareto points, see Figure 5.32a), which shows that at the bottom of the Pareto front there are even less Pareto points than with Algorithm 5 and the Pareto points appear more evenly spaced there. In this case, the Pareto front is computed even faster, namely in 294 seconds, which is probably due to less Pareto points being computed. Furthermore, comparing the distance between consecutive Pareto points for these two algorithms, we observe that the distance between Pareto points is equal for both algorithms up to the 24th Pareto point. For the three Pareto points afterwards, the distance between Pareto points is larger for Algorithm 5. Opposed to this the distance between Pareto points is larger for each remaining Pareto point for Algorithm 6. This can be explained if we look at when h_p was increased for the respective algorithms.
For the Algorithm 6, h_p is adjusted at the 27th, 29th, 30th and 31st Pareto point to $9.49, 3.45, 1.2872$ and 0.1, respectively. Note that 0.1 is actually the lower limit for h_p. This limit was set manually in order to prevent too small values of h_p resulting from Algorithm 6 which can cause problems with the solution of the corresponding reference point problem. For the present optimal control problems, such small values of h_p would only occur at the very bottom of the Pareto front, such that setting this lower bound is not very limiting for the algorithm. Now considering Algorithm 5 h_p is only decreased once at Pareto point 24. So for Algorithm 6 the distance decreases, until the relative distance (cf. (5.3)) falls below ε_{tol}, then h_p is adjusted, such that the distance between Pareto points increases again. This is repeated until no such h_p exists anymore, which is, for example, true if the remaining distance to the last Pareto point is less than $\varepsilon_{curv} \cdot h_x$. In comparison, the adaption of h_p in Algorithm 5 is not able to adjust to the decreasing distance between consecutive Pareto points, so it is to be expected that the distance

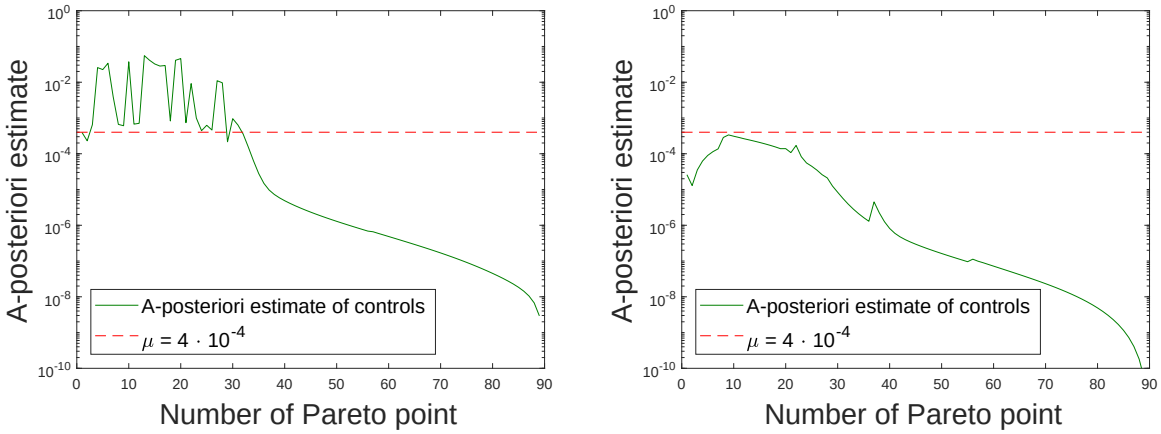


(a) Pareto front for Algorithm 6

(b) Distance between Pareto points for the two adaptive algorithms

Figure 5.32: Results for Algorithm 6 and comparison to Algorithm 5

between Pareto points becomes smaller than for Algorithm 6 at some point. Summing up the adaption of h_p in Algorithm 5 appears to be less flexible, since it does not take into account the present shape of the Pareto front and makes the assumption, that the number of iterations decreases when traversing the Pareto front downwards. The latter is true for all problems considered in the present thesis, but certainly not for general multiobjective optimization problems. Besides, by not taking into account the shape of the Pareto front it is not possible to achieve an even distribution of Pareto points at the bottom of the Pareto front. Evidently, Algorithm 6 performs better regarding the mentioned weaknesses of Algorithm 5. However, the adaption of h_p in Algorithm 5 still performs sufficiently well for the present problems and its advantage is its simplicity. In particular, no additional computations are required as opposed to the procedure in Algorithm 6. For general multiobjective optimization problems, one first has to gain some insight into the problem at hand in order to choose an appropriate strategy to adapt h_p.
Since the adaption of h_p depends on the number of needed iterations for the respective reference point problem, the choice of h_p in the discussed Algorithm 5 may differ for different sizes of the POD basis or compared to the full problem. Therefore, we use fixed $h_p = 20$ for the following tests involving POD strategies in order to ensure comparability of the respective results. This choice of h_p is based on the observations that suggest this to be a favourable value if we would like to achieve relatively fast convergence for both the full and the POD systems.
With the mentioned parameter choice we perform another computation of the Pareto front while paying attention to the a-posteriori estimate this time, see Figure 5.33. This is supposed to serve as a baseline for further investigations.
Comparing these results with previous tests in Figure 5.28, we notice that changing h_p also has a positive effect on the values of the a-posteriori estimate. Especially for the basis update strategy from Algorithm 4 this can be recognised: Here, the a-posteriori estimate stays below the desired threshold for every Pareto point and for none of the reference point problems the algorithm has to switch to the full problem. For the basis extension Algorithm 3, the a-posteriori estimate still lies above the threshold for many Pareto points.

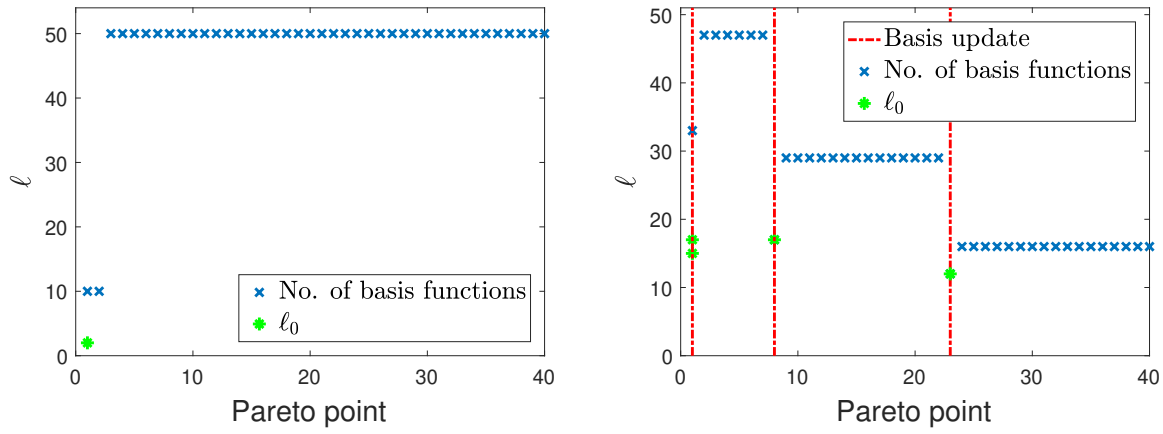


(a) A-posteriori estimate basis extension

(b) A-posteriori estimate basis update

Figure 5.33: A-posteriori estimate for different POD basis strategies

In Figure 5.34, we visualise basis extensions and basis updates that have been performed at the respective Pareto points. For the basis extension Algorithm 3, the basis is extended to ten POD basis function at the first Pareto point and afterwards it is extended to the maximum length of 50 at the fourth Pareto point. Looking at Figure 5.35a we recognise that despite numerous basis extensions, the desired error threshold is exceeded, whereas in Figure 5.34b we realise that with a basis update this problem can be avoided. In particular, the basis update is performed at the same Pareto point, where the error threshold was exceeded with the basis extension algorithm. Eventually, with the basis update algorithm, the calculated 17 basis functions suffice to keep the error sufficiently small.



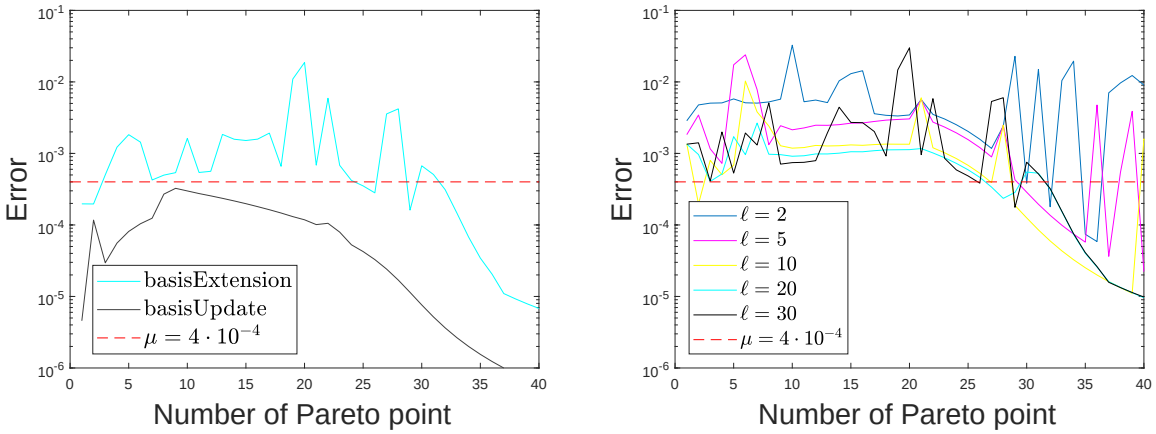
(a) Basis adaptions using basis extensions

(b) Basis adaptions using basis update

Figure 5.34: Basis adaptions using different algorithms

The results are similar if we take a look at the true error $\|\bar{u}^n_{POD} - \bar{u}^n_{Full}\|_U$ für $n = 1, \ldots, N_P$,

where $\bar{u}^n_{POD}$ is the computed optimal control for the n-th POD reference point problem, $\bar{u}^n_{Full}$ is the computed optimal control of the n-th full reference point problem and N_P denotes the total number of Pareto optimal points. We note that the number of calculated Pareto points is the same for both the full and the POD problem, so we can compare the individual Pareto points and the associated controls. This approach is feasible despite differing reference points for the different problems, since we would like to investigate how good the results of an algorithm are if the reference point problems are calculated by itself. This is important because in general, it is impractical to solve the full problem every time before the POD problem only to calculate accurate reference points. Proceeding like this would defeat the purpose of using POD to save computation time as opposed to the full problem. Looking at the results in Figure 5.35a, we see that the true error for the basis extension algorithm is, despite being lower than the a-posteriori estimate, still above the desired threshold. For the basis update strategy, however, the true error is below the threshold for all Pareto points . Furthermore, if we look at the error in the control space for different fixed numbers of POD basis functions in Figure 5.35b, we realise that basis extensions are not sufficient to reduce the error below the desired threshold.



(a) Error in control space for POD basis strategies

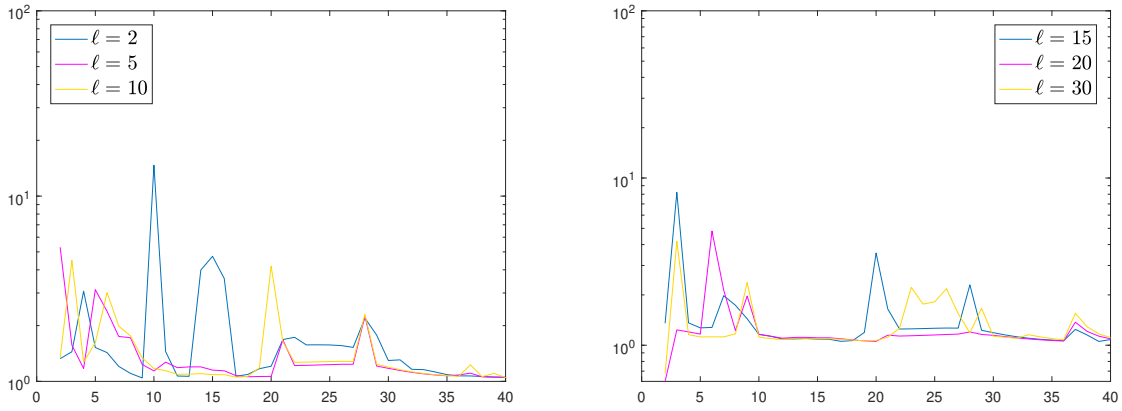
(b) Error in control space for fixed ℓ

Figure 5.35: Error in control space with POD

We observe a significant reduction of the error only up to a fixed number of $\ell = 15$ POD basis functions. For $\ell = 20$, the error improves only slightly compared to $\ell = 15$ and at a few points it even increases. If ℓ is increased further, which is not shown in the figure, the error is very similar to that with $\ell = 20$, so apparently, we cannot expect to reduce the error much beyond $\ell = 20$ POD basis functions. Compared to the 2D problems in [2] and [3] the correlation between POD basis length and error in control space is not as straightforward. However, in the present situation, this problem can be overcome by combining basis extensions with basis updates which are performed at appropriate points in time. Additionally, the previous observations suggest that the maximum number of POD basis functions should not be chosen too high in order to avoid useless basis extensions. Especially if we want to improve our algorithm with regard to computation time, these findings will be benficial.

Next, we would like to investigate the efficiency of the a-posteriori estimate. To achieve this, we

proceed like in Section 5.4 to calculate the quotient $\frac{\mu_U(\bar{u}^n_{POD},z^n)}{\|\bar{u}^n_{POD}-\bar{u}^n_{Full}\|_U}$. Note that we use the same reference points for the full problem and for the POD problem once again. Here, we choose a fixed $h_p = 20$ and we will only consider the first 40 Pareto points, since, as previously observed, a lot of Pareto points lying very close to each other are calculated at the bottom of the Pareto front. For this part of the Pareto front, the error gets very small anyway (cf. Figure 5.33), so that the efficiency of the a-posteriori estimate does not play a role anymore. Looking at the results in Figure 5.36, we find that the efficiency is worst for $\ell = 2$ and actually improves for greater ℓ. For $\ell = 15$ the efficiency appears to be relatively good, for 80% of the first 40 Pareto points the efficiency is below 1.4. If $\ell = 20$, the same holds even for 1.2. However, the efficiency gets worse if $\ell \geq 30$, more precisely, the number of outliers increases apparently. Apart from that, we learn for a possible POD strategy that we also should not start with a too high number of POD basis functions in order to benefit from the range of ℓ with the best efficiency. Summing up, these investigations have shown that although the efficiency of the a-posteriori estimate for the present problem varies for different numbers of POD basis functions, it is in general still good, that is close to one, for most of the Pareto points. Additionally, the error is not underestimated at more than one point which means that the a-posteriori estimate is at least a reliable upper bound for the deviation of the POD solution from the solution of the full problem. Another indication that the possible inefficiencies of the a-posteriori estimate are not too problematic in practice, is the observation that the true error could be controlled very successfully by the a-posteriori estimate (cf. Figure 5.35a). Consequently, the desired accuracy in terms of the true error could be achieved successfully using the a-posteriori estimate. Moreover, we have seen that the true error could not be reduced below the desired threshold without the use of basis updates, that is exceeding the a-posteriori threshold by the basis extension algorithm was not only due to possible inefficiencies of the a-posteriori estimate.



(a) Efficiency for $\ell = 2, 5, 10$ POD basis function

(b) Efficiency for $\ell = 15, 20, 30$ POD basis function

Figure 5.36: Efficiency of the a-posteriori estimate for different fixed ℓ

If we take another look at our basis update strategy, there are several parameters that affect the program computation time. For example, we would like to keep the number of basis extensions low, because for every basis extension the current Pareto point has to be recomputed,

which might cause substantial additional effort if used excessively. Among others these three parameters are influential: The initial number of POD basis functions ℓ_0, because too few basis functions usually do not allow for sufficient accuracy of the solutions which leads to the algorithm performing time-consuming basis extensions until the desired accuracy with regard to the a-posteriori estimate of the solution is achieved. The second parameter is the maximum number of POD basis functions ℓ_{max} up to which basis extensions are performed. As already mentioned, according to our observations, above a certain number of POD basis functions it is unlikely to achieve improvement by further basis extension, which is the reason why they would be very inefficient in that situation. The third parameter we are considering is the number of POD basis elements, by which the POD basis is extended during each basis extension. Since a POD basis extension, according to our observations, only rarely causes the error to increase, it is neither necessary nor efficient with regard to computation time to extend the POD basis only in small steps. Possibly, it is more favourable to make only a limited number of tries to extend the POD basis before we proceed by performing a basis update. With further tests we would like to establish a good compromise between the mentioned parameters. For the following tests we also determine ℓ_{max}, that is the maximum number of POD basis functions, up to which the POD basis is extended, by the same criterion (5.1) as we did for the initial number of POD basis functions ℓ_0. The only difference is that we choose ε_{max} smaller than ε_0, which was used to determine ℓ_0. Apart from that, we need an upper bound ℓ^{ub} for the computation of ℓ_0 and ℓ_{max}. Since we could not observe any improvement already at $\ell = 30$ with regard to the error in control space, we set $\ell^{ub} = 50$ to leave some room for error, so we can reasonably assume that our number of POD basis elements is not a limiting factor. Furthermore, we test different values for the number of POD basis functions by which the POD basis is extended during one single basis extension step of our algorithm. We denote this number by the parameter ℓ_{incr}.

We summarize the POD strategy with the discussed parameters in Algorithm 7. Since we do not only want to consider fixed absolute values for ℓ_{incr}, there is also a parameter par_{incr} in the algorithm, which will be explained below.

The following Algorithm 7 is based on the previously used Algorithm 4 and shows the procedure for the n-th reference point problem.

In the following, we use a fixed $h_p = 20$ initially, then we reduce it analogously to the method described in Algorithm 5. We test a variety of combinations of the three parameters ε_0, ε_{max} and ℓ_{incr}. The goal is not to determine the single best combination of parameters but rather to find indications of what parameter choices are beneficial with regard to program computation time and possibly figure out the reasons why certain parameter choices work well. More precisely, we choose $\varepsilon_0 \in \{10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}, 10^{-9}, 10^{-10}\}$, $\varepsilon_{max} \in \{10^{-6}, 10^{-7}, 10^{-8}, 10^{-9}, 10^{-10}, 10^{-11}, 10^{-12}\}$ and $par_{incr} \in \{$'nobext','max', $\frac{1}{2}, \frac{1}{3}, \frac{1}{4}, 2, 5\}$. Of course only combinations with $\varepsilon_0 > \varepsilon_{max}$ are considered, since our procedure is only feasible for $\ell_{max} > \ell_0$. If par_{incr} is an integer we set $\ell_{incr} = par_{incr}$, otherwise par_{incr} is to be understood as follows: $par_{incr} =$ *'max'* means that only one basis extension per POD basis will be performed. More specifically, if the a-posteriori threshold is exceeded, the length of the POD basis is extended to the maximum length ℓ_{max} immediately. For $0 < par_{incr} < 1$, the procedure is similar: For the current POD basis, a maximum of $\frac{1}{par_{incr}}$ basis extensions will be performed. In practice this is achieved by determining ℓ_0 and ℓ_{max} after a POD basis has been computed and then setting $\ell_{incr} = \lceil par_{incr}(\ell_{max} - \ell_0) \rceil$. The last special case is $par_{incr} =$ *'nobext'*, which means that no basis extensions are performed at all, that is for each POD basis, only a fixed ℓ is computed based

Algorithm 7 POD basis update algorithm

Require: $\mu > 0$, $0 < \ell_{min} < \ell_{ub}$, $\varepsilon_0 > \varepsilon_{max}$, par_{incr};
Determine ℓ, ℓ_{max} with ε_0, ε_{max} by criterion (5.1) and ℓ_{incr} by par_{incr};
Set check = 0;
while check = 0 **do**
    Solve $(\mathrm{RPP})^{\ell}_{z^n}$ with starting point $\bar{u}_0 = \bar{u}^{n-1}$ to obtain $\bar{u}^{\ell}$;
    Compute the a-posteriori estimate $\mu_U(\bar{u}^{\ell}, z^n)$;
    if $\mu_U(\bar{u}^{\ell} < \mu$ **then**
        Set check = 1;
    else
        if $\ell < \ell_{max} - \ell_{incr} + 1$ **then**
            Set $\ell = \ell + \ell_{incr}$;
        else
            if $\ell < \ell_{max}$ **then**
                Set $\ell = \ell_{max}$;
            else
                if updateflag = 0 **then**
                    Compute POD basis using the state y and the adjoint p associated
                    with $\bar{u}^{\ell}$;
                    Set updateflag = 1;
                else if updateflag = 1 **then**
                    Solve full problem $(\mathrm{RPP})_{z^n}$;
                    Compute POD basis using state y and its adjoint p corresponding to $\bar{u}$;
                Determine ℓ, ℓ_{max} with ε_0, ε_{max} by criterion (5.1) and ℓ_{incr} from par_{incr} for
                the new POD basis, set check = 1.

on ε_{max} and if the a-posteriori threshold is exceeded, a basis update is performed instantly. By testing the described choices of par_{incr}, we would like to investigate if computation time can be saved by performing fewer basis extensions or if, due to eventual inefficiencies of the a-posteriori estimate, it might be reasonable to conduct multiple tries of basis extension per POD basis. In this regard, we have observed inefficiencies of the a-posteriori estimate (cf. Figure 5.36) and moreover, there are occasional Pareto points, at which the error in control space or the a-posteriori estimate both increase, instead of decreasing (cf. Figure 5.35) when ℓ is increased. For the aforementioned ranges of parameters, we list the ten fastest parameter combinations and the ten slowest parameter combinations with respect to the computation time in Table 5.10 and in Table 5.11, respectively.

First of all, we notice that for the better configurations, both the number of basis extensions and the number of basis updates are relatively low. However, the number of basis extensions is in no case zero, which suggests that it is not necessarily the most efficient choice to do no basis extensions at all, which is enforced if we choose par_{incr} = *'nobext'*. If we take a look at the results for this parameter choice, we recognise the reason for this shortcoming: No matter how ε_{max} is chosen, at least six basis updates are performed. So if we do not have the option to do basis extensions, this has to be compensated by an increased number of basis updates, which are far more computationally expensive in comparison to basis extensions. Apart from

Table 5.10: The ten best combinations of the parameters ε_0, ε_{max}, par_{incr}

ε_0	ε_{max}	par_{incr}	Computation time	Basis extensions	Basis updates
10^{-6}	10^{-8}	*'max'*	146.72s	5	4
10^{-6}	10^{-8}	1/2	147.33s	7	3
10^{-4}	10^{-9}	*'max'*	147.34s	4	3
10^{-8}	10^{-10}	*'max'*	148.47s	3	3
10^{-4}	10^{-8}	1/2	148.64s	8	3
10^{-6}	10^{-8}	1/2	148.78s	7	3
10^{-5}	10^{-8}	1/2	149.71s	7	3
10^{-8}	10^{-12}	*'max'*	150.80s	3	3
10^{-5}	10^{-8}	*'max'*	151.35s	7	3
10^{-4}	10^{-12}	*'max'*	151.62s	4	3

Table 5.11: The ten worst combinations of the parameters ε_0, ε_{max}, par_{incr}

ε_0	ε_{max}	par_{incr}	Comp. time	Basis extensions	Basis updates
10^{-4}	10^{-6}	1/4	384.16s	82	20
10^{-4}	10^{-6}	2	368.48s	85	15
10^{-4}	10^{-6}	1/3	341.54s	60	19
10^{-5}	10^{-6}	1/3	325.79s	63	18
10^{-4}	10^{-7}	2	301.58s	77	7
10^{-5}	10^{-6}	1/4	298.20s	51	13
10^{-5}	10^{-6}	2	293.01s	50	13
10^{-6}	10^{-12}	2	289.51s	54	3
10^{-6}	10^{-10}	2	277.81s	50	3
10^{-5}	10^{-12}	2	270.06s	49	2

this extreme case, concerning the choice of par_{incr}, it is obvious that an excessive number of basis extensions should be avoided. For the best parameter configurations in our test, this is achieved by performing one to two basis extensions for each individual POD basis that is calculated during the computation of the Pareto front. On the other hand, looking at the worst configurations, we observe that the basis extension is often performed in very small steps, in particular, the basis extensions by two elements occurs six times among the worst configurations. Clearly, the reason for these configurations being slow is in part the high number of executed basis extensions. Regarding ε_0, this parameter is chosen to be at least 10^{-8}, however, there is apparently no upper bound for it, possibly because the choice of a too small ℓ_0 is compensated by fast basis extensions. For ε_{max} the values are not exceeding 10^{-8}, which corresponds to an ℓ_{max} between 30 and 35 for this problem. Most likely for higher values of ε_{max} the resulting ℓ_{max} is too small, such that an excessive number of basis updates has to be executed. This observation is confirmed by taking a look at the worst configurations: Here, multiple cases of $\varepsilon_{max} \in \{10^{-6}, 10^{-7}\}$ occur, which entails seven basis updates for $\varepsilon_{max} = 10^{-7}$ and even at least 13 basis updates for $\varepsilon_{max} = 10^{-6}$. So the algorithm attempts to achieve the desired accuracy, which is difficult because the determined ℓ_{max} is too low, by performing more basis updates.

However, we recognise immediately from the program computation time that this strategy is not particularly efficient.
Summing up, we are able to identify the following aspects to have a positive effect on the program computation time:

- Limiting the number of basis extensions,
- the choice of a sufficiently small ε_{max} (here $\varepsilon_{max} \leq 10^{-8}$).

In contrast, the following aspects have a negative effect on program computation time:

- Performing basis extensions in small steps,
- performing no basis extensions at all,
- choosing ε_{max} too large.

Considering our observations regarding the efficiency of the a-posteriori estimate, the fact that for the investigated 3D problems the improvement of the error in the control space by basis extensions is not as straightforward as in the 2D case with floor heating (cf. [2, Section 6.3], [3, Section 6.2.2]) and the tests regarding parameter choices, it is advisable to allow a limited number of basis extensions and additionally make use of basis updates. Taking into account the aspects we just listed, we can achieve a time efficient algorithm that is still able to produce accurate solutions in terms of the error in the control space.

Conclusion

In this thesis, the results from [2, 3] for a bicriterial optimal control problem governed by time-independent and a time-dependent convection-diffusion equation, respectively, were extended from the case of two space dimensions to three space dimensions. In the process, POD was employed as a method of model-order reduction.
In Chapter 4 we were able to directly apply an already available result from [3] to obtain the well-posedness of the linear heat equation with time-dependent convection term for the case of three space dimensions, such that a reduced optimal control problem could be formulated. Then, the resulting optimal control problem was shown to fit into the framework which was outlined in Chapter 2, so that the available theoretical results were applicable. Overall, the application of the theoretical results from [2, 3] including POD was straightforward since all of the relevant results hold true for any space dimension greater or equal than two. In particular, the POD-convergence results and the error estimates known from [2] were found to remain valid, which proved to be very useful for the numerical experiments. Remarkably, in collaboration with the author of [3], we were able to show new a-priori estimates for the error in the state (see Theorem 4.26), adjoint (see Theorem 4.39) and control space (see Theorem 4.44), the latter of which served as the starting point for various POD basis strategies in the numerical part of the present thesis.
For the primary focus of this thesis, we conducted a variety of numerical experiments on the optimal control problem in three space dimensions. It turned out that, as expected, the computational effort is considerably higher for the problem in three space dimensions than for the problem in two space dimensions. Besides, the systems of linear equations emerging from the three-dimensional model required the use of iterative solvers like the gmres method. Additionally, the application of POD became essential in order to reduce the computational effort to a reasonable extent. Overall, three different examples of three-dimensional models were considered while different strategies to control the number of POD basis functions were investigated. In particular, the POD a-posteriori estimate that was derived and applied in [2] was an essential tool to observe the error caused by the POD approximation and to control the length of the POD basis based on this estimate. In this context, one idea was to determine an appropriate number of POD basis functions upfront by using an a-priori estimate of the POD error in the control space, which was derived in Chapter 4. Indeed, we were able to save computation time using this inequality by avoiding an unnecessarily high number of POD basis extensions and moreover this approach even proved to be adaptive to the dynamics to some degree by leading to a higher number of initial POD basis functions for the problems with time-dependent convection as opposed to the problem with time-independent convection or without convection. Another observation was that we found the POD a-posteriori estimate to be less efficient than for the two-dimensional problems treated in [3] and [2], but it still performed well enough to justify its application as a criterion to extend the POD basis since it proved to be a reliable upper bound for the error in the control space. Similarly, the correlation between the number

of used POD basis functions and the error in the control space turned out to be not as clear as for the two-dimensional problems. Therefore, basis extensions were, in general, less efficient, but remained as one possible tool in order to improve the accuracy of the POD approximation. However, these findings showed the importance of employing basis updates additionally. In particular, POD basis updates turned out to be more useful than for two-dimensional problems such as those investigated in [3]. Evidently, the error in the control space could be reduced by utilizing POD basis updates in comparison to just maintaining the original POD basis along with performing basis extensions. Beyond that, it was not even possible to keep the error in the control space below the defined threshold by only performing basis extensions without any basis updates for the more complicated models we considered. Finally, having investigated the use of basis extensions and basis updates extensively, we were able to guidelines of how to choose different parameters in the utilized POD strategies in order to obtain a time-efficient and sufficiently accurate POD-based algorithm that includes both basis extensions and basis updates as well as appropriate methods to choose the number of POD basis functions.
Apart from the described POD basis strategies, the effect of omitting the weight on the second objective function $\hat{J}_2$ was investigated. Namely, this turned out to make the convergence in the upper part of the Pareto front much harder. We found out that increasing the step size h_p helped to improve the convergence and we proposed algorithms to adapt the value of h_p based on the required number of Newton-CG iterations and also based on the curvature of the Pareto front, such that the convergence in the upper part of the Pareto front was improved considerably and the distance between Pareto points in the bottom part of the Pareto front did not become too small. Overall, adjusting the step size h_p either by hand or by the mentioned algorithms helped improve convergence for the individual reference point problems and to reduce the computation time of the whole Pareto front considerably. This was found to be true for both the full finite element problem and the POD problem.
All things considered, the strategies concerning POD and the step size h_p were able to greatly reduce the computation time of the Pareto front as opposed to the original finite element problem while still yielding high-quality solutions, since we were able to control the error caused by the POD approximation successfully.

Bibliography

[1] M. Ehrgott. *Multicriteria Optimization*. Springer Berlin, Heidelberg, second edition, 2005.

[2] S. Banholzer. Pod-based bicriterial optimal control of convection-diffusion equations, 2017. Master Thesis.

[3] E. Makarov. Pod-based bicriterial optimal control of time-dependent convection-diffusion equations with basis update, 2018. Master Thesis.

[4] M. Gubisch and S. Volkwein. Proper orthogonal decomposition for linear-quadratic optimal control. In P. Benner, A. Cohen, M. Ohlberger, and K. Willcox, editors, *Model reduction and approximation : theory and algorithms*, number 15 in Computational science and engineering, pages 3–63. SIAM, Philadelphia, 2017.

[5] F. Tröltzsch. *Optimale Steuerung partieller Differentialgleichungen - Theorie, Verfahren und Anwendungen*. Vieweg+Teubner, Wiesbaden, second edition, 2009.

[6] R. Dautray and J.-L. Lions. *Mathematical Analysis and Numerical Methods for Science and Technology. Volume 5: Evolution Problems I*. Springer-Verlag, Berlin, 1992.

[7] Stefan Banholzer, Dennis Beermann, and Stefan Volkwein. Pod-based error control for reduced-order bicriterial pde-constrained optimization. *Annual Reviews in Control*, 44:226–237, 2017.

[8] L.C. Evans. *Partial Differential Equations*. American Mathmatical Society, Providence, Rhode Island, 2008.

[9] John R. Singler. New pod error expressions, error bounds, and asymptotic results for reduced order models of parabolic pdes. *SIAM Journal on Numerical Analysis*, 52(2):852–876, 2014.

[10] E. Kammann, F. Troeltzsch, and S. Volkwein. A posteriori error estimation for semilinear parabolic optimal control problems with application to model reduction by pod. *ESAIM: Mathmatical Modelling and Numerical Analysis*, 2013.

[11] S. Banholzer, E. Makarov, and S. Volkwein. Pod-based multiobjective optimal control of time-variant heat phenomena. Technical Report 370, 2017.