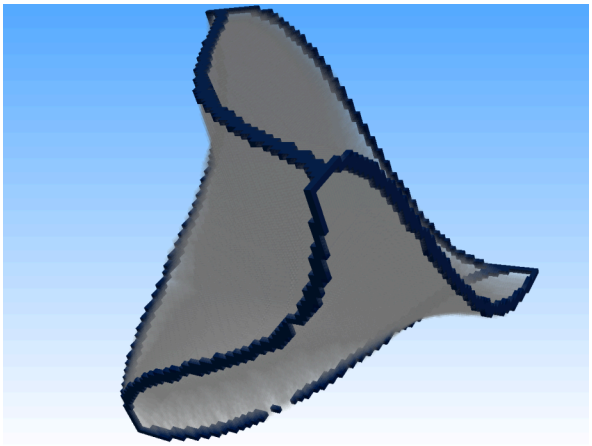
UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft

Exploiting Structure in Multiobjective Optimization and Optimal Control



Von der Fakultät für
Elektrotechnik, Informatik und Mathematik
der Universität Paderborn
zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften
– Dr. rer. nat. –

genehmigte Dissertation

von

Sebastian Peitz

Paderborn 2017

Gutachter: Prof. Dr. Michael Dellnitz
Prof. Dr. Sina Ober-Blöbaum
Prof. Dr. Stefan Volkwein

Tag der mündlichen Prüfung: 08.08.2017

Acknowledgements

This dissertation would have not been possible without the persistent support, encouragement and assistance of people from both the academic and social sphere, and I am grateful to all of them.

First of all, I would like to express my gratitude to my advisor Prof. Dr. Michael Dellnitz for his guidance and support, for introducing me to the fascinating field of multiobjective optimization, for his enthusiasm, and for the freedom and the multitude of opportunities he has given me during my research.

I am grateful to Prof. Dr. Sina Ober-Blöbaum from the University of Oxford for introducing me to optimal control and many interesting related topics and for her extensive supervision during the last years.

I would like to thank Prof. Dr. Stefan Volkwein from the University of Konstanz for introducing me to the field of reduced order modeling and for the fruitful cooperation. My visits to Konstanz have been both enlightening and enjoyable.

I gratefully acknowledge the support I have received within the leading edge cluster *Intelligent Technical Systems OWL (it's OWL)* as well as the *DFG Priority Programme 1962 - Non-smooth and Complementarity-based Distributed Parameter Systems: Simulation and Hierarchical Optimization*. Special thanks goes to my co-workers Dennis Beermann, Julian Eckstein, Dr. Patrick Friedel, Manuel Gräler, Dr. Ulrich Köhler, and Kai Schäfer.

Many thanks go to my past and present colleagues at Paderborn University: Adrian Ziessler, Bennet Gebken, Raphael Gerlach, Dr. Kathrin Flaßkamp, Dr. Sebastian Hage-Packhäuser, Dr. Mirko Hessel-von Molo, Dr. Christian Horenkamp, Dr. Karin Mora, and Bianca Thiere. I am grateful for many professional and personal discussions, their helpful advice – in particular while making the transition from engineering to applied mathematics – and the pleasant and enjoyable work atmosphere they have created.

I would like to express my gratitude to my parents for their constant encouragement and their support of all my decisions. Finally, I would like to thank Nina-Madeleine Brummel for every single day we spend together.

Abstract

Multiobjective optimization plays an increasingly important role in modern applications, where several criteria are often of equal importance. The task in multiobjective optimization and multiobjective optimal control is therefore to compute the set of optimal compromises (the Pareto set) between the conflicting objectives.

Since – in contrast to the solution of a single objective optimization problem – the Pareto set generally consists of an infinite number of solutions, the computational effort can quickly become challenging. This is even more the case when many problems have to be solved, when the number of objectives is high, or when the objectives are costly to evaluate. Consequently, this thesis is devoted to the identification and exploitation of structure both in the Pareto set and the dynamics of the underlying model as well as to the development of efficient algorithms for solving problems with additional parameters, with a high number of objectives or with PDE-constraints. These three challenges are addressed in three respective parts.

In the first part, predictor-corrector methods are extended to entire Pareto sets. When certain smoothness assumptions are satisfied, then the set of parameter dependent Pareto sets possesses additional structure, i.e. it is a manifold. The tangent space can be approximated numerically which yields a direction for the predictor step. In the corrector step, the predicted set converges to the Pareto set at a new parameter value. The resulting algorithm is applied to an example from autonomous driving.

In the second part, the hierarchical structure of Pareto sets is investigated. When considering a subset of the objectives, the resulting solution is a subset of the Pareto set of the original problem. Under additional smoothness assumptions, the respective subsets are located on the boundary of the Pareto set of the full problem. This way, the "skeleton" of a Pareto set can be computed and due to the exponential increase in computing time with the number of objectives, the computations of these subsets are significantly faster which is demonstrated using an example from industrial laundries.

In the third part, PDE-constrained multiobjective optimal control problems are addressed by reduced order modeling methods. Reduced order models exploit the structure in the system dynamics, for example by describing the dynamics of only the most energetic modes. The model reduction introduces an error in both the function

values and their gradients, which has to be taken into account in the development of algorithms. Both scalarization and set-oriented approaches are coupled with reduced order modeling. Convergence results are presented and the numerical benefit is investigated. The algorithms are applied to semi-linear heat flow problems as well as to the Navier-Stokes equations.

Zusammenfassung

Mehrzieloptimierung bekommt einen immer größeren Stellenwert in modernen Anwendungen, in denen verschiedene Zielkriterien häufig von gleich großer Bedeutung sind. Ziel der Mehrzieloptimierung bzw. der Mehrzieloptimalsteuerung ist es daher, die Menge optimaler Kompromisse (die Paretomenge) für die in Konflikt stehenden Ziele zu berechnen.

Im Gegensatz zur Lösung von Einzieloptimierungsproblemen besitzt die Paretomenge im Allgemeinen unendlich viele gleichwertige Lösungen. Der numerische Aufwand kann daher sehr schnell zu einer großen Herausforderung werden. Dies tritt umso mehr zutage, wenn zusätzliche Faktoren hinzukommen, wie zum Beispiel das Lösen einer Vielzahl von Problemen, die Berücksichtigung vieler Zielfunktionen oder ein besonders hoher Rechenaufwand zur Berechnung der Zielfunktionswerte. Diese Arbeit widmet sich daher der Identifikation sowie der Ausnutzung von Strukturen, sowohl in der Paretomenge als auch in der zugrunde liegenden Dynamik, sowie der Entwicklung effizienter Algorithmen für parameterabhängige Probleme, Probleme mit vielen Zielfunktionen oder durch partielle Differentialgleichungen beschriebene Probleme. Die drei genannten Problemklassen werden in drei separaten Teilen betrachtet.

Im ersten Teil werden weit verbreitete predictor-corrector-Methoden auf die Fortsetzung ganzer Paretomengen erweitert. Falls gewisse Glattheitsannahmen erfüllt sind, besitzt die Menge parameterabhängiger Paretomengen eine zusätzliche Struktur, sie ist in diesem Falle eine Mannigfaltigkeit. Der Tangentialraum kann numerisch approximiert werden, um eine Richtung für den predictor-Schritt zu ermitteln. Im corrector-Schritt wird anschließend Konvergenz zur Paretomenge für einen neuen Parameterwert erreicht. Der resultierende Algorithmus wird an einem Beispiel aus dem autonomen Fahren validiert.

Im zweiten Teil wird die hierarchische Struktur von Paretomengen untersucht. Durch die Betrachtung einer Teilmenge der Zielfunktionen lässt sich eine Teilmenge der ursprünglichen Paretomenge berechnen. Unter zusätzlichen Glattheitsannahmen liegen die jeweiligen Teilmengen auf dem Rand dieser übergeordneten Paretomenge. Auf diese Weise lässt sich das “Skelett” einer Paretomenge berechnen. Da der Rechenaufwand exponentiell mit der Anzahl an Zielfunktionen wächst, können diese Teilmengen um mehrere Größenordnungen schneller berechnet werden, was an einem Beispiel aus der industriellen Wäscherei veranschaulicht wird.

Im dritten Teil werden Mehrzieloptimalsteuerungsprobleme behandelt, deren Dynamik durch partielle Differentialgleichungen beschrieben wird, und mit Verfahren aus der Modellreduktion gekoppelt. Bei der Modellreduktion werden Strukturen in der Systemdynamik ausgenutzt, indem beispielsweise nur die Dynamik der energiereichsten Moden betrachtet wird. Diese reduzierten Modelle resultieren in einem Fehler sowohl in der Zielfunktion selbst als auch im Gradienten, was bei der Entwicklung von Algorithmen berücksichtigt werden muss. Es werden sowohl Skalarisierungsmethoden als auch mengenbasierte Ansätze mit Modellreduktion gekoppelt und das Konvergenzverhalten sowie die numerische Effizienz untersucht. Die Ergebnisse werden anhand semilinearer Wärmeleitungsprobleme und der Navier-Stokes-Gleichungen verdeutlicht.

Contents

1 Introduction

Multiobjective optimization is everywhere. In daily life as well as in technical applications, there is hardly ever only one goal of interest. For example, if we want to go on vacation, we want to go to a beautiful place with perfect weather conditions, ideally offering opportunities for recreation as well as various activities. Moreover, the destination should be within easy reach and the journey should be affordable. Since not all of these goals can be optimally satisfied at the same time, we are forced to choose a compromise.

The same situation occurs in technical applications. Due to the ever increasing complexity of technical systems and design requirements, there are nowadays few problems where only one objective is of importance. For example, in transportation one wants to reach a destination as fast as possible while minimizing the energy consumption. This should ideally be achieved while providing an optimal comfort and simultaneously maintaining maximal security. The example illustrates that many objectives are often equally important. Similar to the vacation example, these different objectives generally contradict each other which is why we are forced to accept a trade-off between them. This results in a *multiobjective optimization problem (MOP)*:

$$\min_{\boldsymbol{u}} \boldsymbol{J}(\boldsymbol{u}) = \min_{\boldsymbol{u}} \begin{pmatrix} J_1(\boldsymbol{u}) \\ \vdots \\ J_k(\boldsymbol{u}) \end{pmatrix},$$

where multiple objectives have to be minimized at the same time. Similar to scalar optimization problems, we want to find an optimal solution to this problem but in a multiobjective optimization problem, this solution consists of the *set of optimal compromises*. This means that solutions are optimal if we cannot find other solutions that are superior in all objectives. For example, if one car is faster, cheaper, more energy efficient and more comfortable than another one, there is no reason to choose the second option (unless some additional objective is considered where it is the superior choice). The task is therefore to compute the set of optimal compromises between the conflicting objectives, the so-called *Pareto set*.

There exists a large variety of algorithms for the computation of Pareto optimal solutions, either for single optima or for the entire set of optimal compromises. These

can be divided into *scalarization methods* (see e.g. [Ehr05] for an overview), *continuation methods* [Hil01, MS17], *evolutionary approaches* [CLV07] and *set-oriented methods* [DSH05, SWOBD13]. Moreover, combinations of these methods have been investigated by various researchers. In [SMDT03], for example, concepts from evolutionary computation are combined with set-oriented approaches whereas in memetic algorithms (see e.g. [KC00, BdJ05, LSCS10, Bos12]), deterministic approaches are used to accelerate local convergence.

If we want to optimally control a dynamical system – i.e. a system whose state varies over time – then we have to solve an optimal control problem. In this situation, the optimization variable is a function (for example of time) instead of a finite-dimensional a parameter. This can naturally be extended to a *multiobjective optimal control problem (MOCP)* by formulating several criteria. The most important distinction between optimization and optimal control is that since a dynamical system shall be controlled, we have to take the system dynamics into account as a constraint:

$$\min_{\boldsymbol{y},\boldsymbol{u}} \boldsymbol{J}(\boldsymbol{y},\boldsymbol{u}) = \min_{\boldsymbol{y},\boldsymbol{u}} \begin{pmatrix} \int_{t_0}^{t_e} C_1(\boldsymbol{y}(t),\boldsymbol{u}(t))\,\mathrm{d}t + \Phi_1(t_e) \\ \vdots \\ \int_{t_0}^{t_e} C_k(\boldsymbol{y}(t),\boldsymbol{u}(t))\,\mathrm{d}t + \Phi_k(t_e) \end{pmatrix}$$

$$\begin{aligned} \text{s.t.} \qquad & \dot{\boldsymbol{y}}(t) = \boldsymbol{F}(\boldsymbol{y}(t),\boldsymbol{u}(t)), && t \in [t_0, t_e], \\ & \boldsymbol{y}(t_0) = \boldsymbol{y}_0. \end{aligned}$$

These problems can either be solved directly by discretizing both the state and the control or indirectly utilizing optimality conditions based on the Pontryagin Maximum Principle or the Lagrange functional [BBB+01]. The system dynamics $\boldsymbol{F}$ is often described by ordinary or partial differential equations and depending on the type of dynamical system, the computational effort to numerically solve these problems can quickly become very large. Similar to optimization, there are many scenarios where multiple objectives are of interest in control problems. If the vehicle is controlled by the engine torque profile over time, for example, the transportation problem from above (fast versus energy efficient transportation) is an MOCP.

Multiobjective optimization is a powerful tool in many situations and in different stages of a design process. During the development of a new product, knowledge of the Pareto set can help the developer (also referred to as *decision maker*, cf. [Mie12]) to judge the current design and decide whether trade-offs are beneficial for the overall outcome. Even an unsatisfactory outcome can be very helpful in the sense that the design concept can be modified early in the development process. During the operation of a system, knowledge of the Pareto set can be utilized to increase the system's

flexibility. This way, the priorization of the different objectives may be varied, e.g. in order to react to changes in the system state (such as a low battery state of charge), to changing constraints (such as the speed limit) or to changes in the environment (e.g. changing weather conditions). This online adaptivity is a key enabler for the development of intelligent (*self-optimizing* [GFDK09]) systems.

In conclusion, the decision making process, independent of whether realized by a human operator or by a control system using sensor data, can be significantly improved by solving multiobjective optimization problems instead of artificially synthesizing equally important criteria into one objective (e.g. by a *weighted sum*). The benefits of putting optimization before decision making are described very well in [BDMS08]: "Converting a multiobjective optimization problem into a simplistic single-objective problem puts decision making before optimization, that is, before alternatives are known. [...] articulating preferences without a good knowledge of alternatives is difficult, and thus the resulting optimum may not correspond to the solution the user would have selected from the set of Pareto optimal solutions. Treating the problem as a true multiobjective problem means putting the preference articulation stage after optimization [...]. This will help the user gain a much better understanding of the problem and the available alternatives, thus leading to a more conscious and better choice."

Being aware that in general multiple objectives are of interest, multiobjective optimization has an increased value compared to single objective optimization in many situations. However, we also have to accept a trade-off since the fact that the objectives are conflicting often results in an infinite number of optimal compromises and hence, the computing time is significantly higher. The problem of increased computational complexity is even amplified if one (or more) of the following three factors apply:

(1) many (parameter dependent) problems need to be solved,

(2) many objectives are present,

(3) the underlying model is costly to solve.

The second point is specific to multiobjective problems and while the first and the third point are obviously equally valid for scalar problems, there exist additional challenges in the situation where multiple criteria are present.

There are numerous examples for problems which exhibit one or more of the above-mentioned challenges. Let us consider *autonomous driving*, for example. If one wants to optimally control an electric vehicle with respect to multiple criteria in real-time, the available amount of time is far too short for computing the entire

Pareto set online, rendering such an approach infeasible. A popular method is therefore to a priori scalarize the objectives (e.g. by a weighted sum) and thereby transform the problem into a scalar optimal control problem that can to be solved online. This results in the well-known and widely used *model predictive control (MPC)* approach, see [GP17] and Section 2.2.2 of this thesis for details. However, as motivated above, the strategy of a priori scalarizing the problem may result in undesirable solutions since a small increase in one objective may allow for a much larger decrease in another one, a trade-off a decision maker might happily accept. An alternative approach (denoted as *explicit MPC* [AB09]) to satisfy the real-time requirement is to solve problems for many different parameter values in an *offline phase* and then select the respective solution from a library in the *online phase*. When extending this concept to multiple objectives, we face the challenge of building a library which consists of a large number of Pareto sets such that the currently valid set can be selected from the library in the online phase. Hence, we have to solve many parameter dependent MOPs.

The increased computational effort due to a large number of objectives mainly stems from the curse of dimensionality. The Pareto set typically is an object of dimension $k-1$, where k is the number of objectives. Hence, each additional objective increases the dimension by one and the computational effort grows exponentially. An example for such a problem occurs in industrial laundries. There several tons of laundry are processed every day and hence, significant amounts of resources, namely energy, water and cleaning detergents are required. Therefore, it is desirable to utilize optimization and optimal control techniques to operate such a laundry in an intelligent manner. One of the most important parts is the cleaning process itself, where the optimal configuration depends on the type of laundry and the type of contamination. In 1959, Herbert Sinner [Sin59] developed a concept for laundering which is still applied in modern laundries. It states that in order to realize satisfactory cleaning, the four influential factors temperature, chemistry (i.e. the amount of cleaning detergents), washing time and mechanics (i.e. the rotational velocity of the laundry) have to be chosen in the right way. Based on this, the process of cleaning different types of contamination can be modeled according to experimental data. Since there exists a variety of types of contamination (e.g. fat, wine, curry, oil, or blood) which should all be cleaned as best as possible, we have to solve an MOP with many objectives.

The third challenge is present in all MOCPs where the system dynamics are governed by partial differential equations (PDEs) which are used to model a wide range of physical problems from heat flow over electro-magnetism to fluid dynamics. The typical procedure to numerically solve a PDE is to introduce a discretization in both space and time. The spatial domain is discretized by a numerical mesh based on a finite difference, finite volume or finite element method, thereby transforming the infinite-dimensional into a (potentially very large) finite-dimensional system (i.e. a

system of coupled ordinary differential equations) [FP02]. With increasing computational capacities, the size of problems that can be solved has tremendously increased during the last decades [SvdVR08]. However, many technical applications result in problems that even nowadays are very difficult or even impossible to solve. This is for example the case when one wants to directly solve the Navier-Stokes equations in complex domains. Consequently, solving optimal control problems involving PDEs is a great challenge, and considering multiple criteria furthermore increases the complexity.

Motivated by the three challenges above and the corresponding exemplary applications, this thesis is concerned with exploiting structural aspects, either of multiobjective optimization problems in general or of the specific problem at hand, in order to reduce the computational cost. To this end, the three challenges mentioned above will be addressed separately in three respective chapters. For the first two, the fact that under additional smoothness assumptions the Pareto set is a *manifold* [Hil01] will be utilized. For the third challenge, we consider multiobjective optimal control problems constrained by PDEs. If in this situation the system dynamics exhibits a structure (e.g. *coherent structures* in fluid flow), then the PDE can be replaced by a *reduced order model (ROM)* [HLBR12] which can be solved much faster.

In Chapter 2, the mathematical concepts utilized throughout this thesis will be introduced. The focus lies on multiobjective optimization, including a survey of existing methods and convergence proofs for the *reference point method* and the *subdivision algorithm*, both of which will frequently be used in the subsequent chapters. Furthermore, optimal control and model predictive control are introduced before the chapter concludes with an introduction to model order reduction in general and ROM-based optimal control of PDEs in particular.

Chapter 3 is concerned with the situation where a large number of problems has to be solved, which is motivated in Section 3.1 by a project within the leading edge cluster *Intelligent Technical Systems OWL (it's OWL)* in the context of autonomously driven electric vehicles. The corresponding MOP depends on an additional parameter $\boldsymbol{p}$ (such as the initial velocity of the electric vehicle):

$$\min_{\boldsymbol{u}} \boldsymbol{J}(\boldsymbol{u}, \boldsymbol{p}) = \min_{\boldsymbol{u}} \begin{pmatrix} J_1(\boldsymbol{u}, \boldsymbol{p}) \\ \vdots \\ J_k(\boldsymbol{u}, \boldsymbol{p}) \end{pmatrix}.$$

For the analysis of dynamical systems (e.g. path following and bifurcations) as well as for solving multiobjective optimization problems, continuation methods (see e.g. [AG03]) are a powerful and widely used tool. The key enabler for these methods is that by applying the *Implicit Function Theorem*, one can show that the solution depends continuously on (or is even differentiable with respect to) the parameter of

interest. Consequently, one can determine the tangent space at a current parameter value and perform a predictor step along a direction within the tangent space. In the corrector step a point is computed which solves the problem under consideration for a new parameter value. In the situation considered here, this new "point" is the entire Pareto set. Before addressing the corresponding numerical challenges, it will first be shown in Section 3.2 that if the objective functions are differentiable with respect to the parameter, the set of points satisfying the necessary optimality conditions depends continuously on the parameter. Then a predictor-corrector method will be presented by which continuation of entire Pareto sets can be realized. Since it is difficult to compute the tangent space of the Pareto set, it is approximated via finite differences. However, since the corrector step locally converges to the Pareto set, this approach works very well which will be shown using academic examples as well as an application from autonomous driving (Section 3.3).

The challenge of a large number of objectives will be addressed in Chapter 4. Due to the ever increasing complexity of applications, the number of objectives that are considered has increased during the last years. Whereas in the beginning of multiobjective optimization, two or maximally three objectives were formulated, four to twenty objectives are often considered today [FPL05]. Since the computational effort grows exponentially with the number of objectives, solving problems with many objectives quickly becomes very challenging [SLC11] which is why a new term named *many-objective optimization* (see [VBB14] for an overview) has been coined for this branch of multiobjective optimization, and many researchers are developing algorithms specifically tailored to handle a large number of objectives [PF07, BZ11, YLLZ13]. Due to the exponential increase, every objective that can be neglected significantly reduces the time required to compute the Pareto set. This has been addressed in [SDT+13], for example, where objectives of minor interest are identified using *Proper Orthogonal Decomposition (POD)*. The method proposed here works in a similar fashion. It will be shown in Section 4.1 that considering only a subset of objectives leads to the computation of a subset of the original Pareto set. Moreover, if this set is a manifold, then neglecting objectives results in Pareto optimal solutions that lie on the boundary of the original set. This way, a skeleton of the Pareto set of the original problem can be computed very efficiently. Finally, interior points can be obtained by means of a generalized ϵ-constraint algorithm presented in Section 4.2. The improvement in computational efficiency will be demonstrated using academic examples (Section 4.3) as well as the application from industrial laundries introduced earlier (Section 4.4).

Finally, Chapter 5 addresses the third challenge where the increased computational cost is caused by the underlying dynamical system itself. More specifically, we are going to consider MOCPs constrained by PDEs. In this case, reduced order modeling techniques can be applied which exploit the structure of the underlying dynamics. For non-linear problems, ROMs obtained via *POD* and *Galerkin projec-*

tion [HLBR12] have been very successful in many applications. In this approach data obtained from the infinite-dimensional state space is projected onto a basis $\{\psi_i\}_{i=1}^{\ell}$ which consists of only a small number of elements:

$$y(\boldsymbol{x}, t) \approx \sum_{i=1}^{\ell} z_i(t)\psi_i(\boldsymbol{x}).$$

This way, the infinite-dimensional problem can be replaced by a system of ordinary differential equations for the coefficients $z_i(t)$.

Reduced order modeling approaches can be further divided into methods where

(1) a model is created only once,

(2) the reduction of the ROM-based optimization is validated by regularly evaluating the full model,

(3) error analysis is utilized to estimate the error of the underlying ROM and to perform necessary adaptations.

All three of these approaches will be considered here, where the complexity of the dynamical system varies from a linear heat equation to the two-dimensional, incompressible Navier-Stokes equations. In Section 5.1, concept (a) will be pursued in order to solve an otherwise intractable flow control problem. Different algorithms for MOCPs will be compared with respect to the quality of the solution as well as the computational cost. Furthermore, it will be shown that in particular for systems with complex dynamics, it is highly advisable to implement means to control the error of the ROM. Therefore in Section 5.2 a trust region framework for POD-based optimal control of PDE-constrained problems [Fah00] – which falls within category (b) – will be combined with scalarization methods. This way convergence of the reduced problem can be guaranteed while a considerable reduction of the computational cost is achieved. Finally, in order to enable set-oriented approaches to utilize ROMs, the subdivision algorithm developed in [DSH05] will be extended to inexact models in Section 5.3. It will be shown that inexactness in the gradients of the objectives leads to a reduction of the cone of valid descent directions and consequently, only a superset of the Pareto set can be obtained. However, the "distance" between the exact and the inexact Pareto front can be controlled by bounding the error of the ROM. The interplay between reduced order modeling and set-oriented multiobjective optimal control, which is the subject of the research project *Multiobjective Optimal Control of Partial Differential Equations Using Reduced-Order Modeling* within the *DFG Priority Programme 1962 - Non-smooth and Complementarity-based Distributed Parameter Systems: Simulation and Hierarchical Optimization*, will finally be investigated in Section 5.4

Parts of this thesis grew out of several preceding publications to which the author has made substantial contributions. They are referenced at the beginning of the respective chapters.

2 Theoretical Background

The purpose of this chapter is to introduce and review the mathematical concepts that will be used during the subsequent chapters of this thesis, starting with multiobjective optimization in Section 2.1, which is the common foundation for all following results. *Optimal control* and *model predictive control (MPC)* are introduced in Section 2.2 . When dealing with objective functionals instead of functions, optimal control methods based on the *calculus of variations* need to be utilized. This is for example the case when the optimization variable is a function of time. In order to stabilize systems with complex dynamics, open loop and closed loop control are combined which leads to the concept of MPC. In situations where the underlying dynamical system is costly to solve, as is the case for partial differential equations (PDEs), model order reduction techniques are a popular approach to reduce the computational effort. Consequently, the basic principles of *reduced order modeling (ROM)* will be addressed in Section 2.3.

2.1 Multiobjective Optimization

In most applications from industry or economy, multiple objectives are of interest. In the case of manufacturing, for example, one wants to produce a product as cost efficient as possible while maintaining high quality. When designing airplanes, it is of interest to minimize the weight in order to save energy. At the same time, the structural stability needs to be maximized to ensure a sufficient level of security. These are only two examples of a dilemma which occurs in many situations. The optimal value of the different objectives can seldom be achieved at the same time since they are *conflicting*. This gives rise to a *multiobjective optimization problem (MOP)*. In a situation where multiple objectives are present, the solution does in general not consist of isolated points but of the *set of optimal compromises* between these objectives. The aim of multiobjective optimization is to compute this set, also known as the *Pareto set* named after Vilfredo Pareto (1848 – 1923) [Par71]. All elements contained in the Pareto set have in common that one objective can only be improved by accepting a trade-off in at least one other objective. In other words, from a Pareto optimal solution it is impossible to simultaneously improve all objectives. In the literature, multiobjective optimization is also referred to as *multicriteria optimization*, *vector optimization* or *Pareto optimization*.

In this section the concept of multiobjective optimization will be introduced, starting with Pareto optimality in Section 2.1.1. Gradients and descent directions are revisited in the multiobjective context in Section 2.1.2 and some geometrical properties are recalled in Section 2.1.3 before summarizing the most important methods for solving MOPs in Sections 2.1.4 and 2.1.5. The next step after the Pareto set has been computed, the *decision making* (cf. [Mie12]) will (apart from a small exception in Chapter 3) not be covered in this thesis.

2.1.1 Pareto Optimality

In multiobjective optimization we want to minimize multiple objectives at the same time. Consequently, the fundamental difference to scalar optimization is that the objective function $\boldsymbol{J} : \mathbb{R}^n \to \mathbb{R}^k$ is vector-valued. Hence, the general problem is of the form

$$\begin{aligned} \min_{\boldsymbol{u}\in\mathbb{R}^n} \boldsymbol{J}(\boldsymbol{u}) = \min_{\boldsymbol{u}\in\mathbb{R}^n} & \begin{pmatrix} J_1(\boldsymbol{u}) \\ \vdots \\ J_k(\boldsymbol{u}) \end{pmatrix} \\ \text{s.t.} \qquad g_i(\boldsymbol{u}) &\leq 0, \quad i = 1, \dots, l, \\ h_j(\boldsymbol{u}) &= 0, \quad j = 1, \dots, m, \end{aligned} \tag{2.1}$$

where $\boldsymbol{u} \in \mathbb{R}^n$ is the *control variable* and $\boldsymbol{g} : \mathbb{R}^n \to \mathbb{R}^l$, $\boldsymbol{g}(\boldsymbol{u}) = (g_1(\boldsymbol{u}), \dots, g_l(\boldsymbol{u}))^\top$ and $\boldsymbol{h} : \mathbb{R}^n \to \mathbb{R}^m$, $\boldsymbol{h}(\boldsymbol{u}) = (h_1(\boldsymbol{u}), \dots, h_m(\boldsymbol{u}))^\top$, are inequality and equality constraints, respectively. The space of the control variables is also called the *decision space* and the objective function maps $\boldsymbol{u}$ to the *objective space*. Alternatively, problem (2.1) can be written as

$$\min_{\boldsymbol{u}\in\mathcal{U}} \boldsymbol{J}(\boldsymbol{u}), \tag{MOP}$$

where $\mathcal{U} = \{\boldsymbol{u} \in \mathbb{R}^n \mid g_i(\boldsymbol{u}) \leq 0,\ i = 1, \dots, l \text{ and } h_j(\boldsymbol{u}) = 0,\ j = 1, \dots, m\}$ is the *feasible set*. In the unconstrained case, we have $\mathcal{U} = \mathbb{R}^n$.

Remark 2.1.1. *In this thesis, all vector-valued quantities as well as mappings to vector-valued spaces are written in bold notation.*

Remark 2.1.2. *It is common in finite-dimensional optimization to use the notation $\boldsymbol{x}$ for the control or optimization variable and $\boldsymbol{F}$ for the objective function. In contrast to that, $\boldsymbol{u}$ and $\boldsymbol{J}$ are more common for control problems. In order to unify the notation, the latter will be used throughout this thesis for all optimization and optimal control problems.*

In contrast to single objective optimization problems, there exists no total order of the objective function values in $\mathbb{R}^k, k \geq 2$ (unless they are not conflicting). Therefore, the comparison of values is defined in the following way [Mie12]:

Definition 2.1.3. *Let $\boldsymbol{v}, \boldsymbol{w} \in \mathbb{R}^k$. The vector $\boldsymbol{v}$ is* less than *$\boldsymbol{w}$ (denoted by $\boldsymbol{v} <_p \boldsymbol{w}$), if $v_i < w_i$ for all $i \in \{1, \ldots, k\}$. The relation $\leq_p$ is defined in an analogous way.*

Example 2.1.4. *Consider the points $\boldsymbol{u}_1 = (3, 2)$ and $\boldsymbol{u}_2 = (1, 3)$ and $\boldsymbol{u}_3 = (2, 1)$ (cf. Figure 2.1). Then neither $\boldsymbol{u}_2 <_p \boldsymbol{u}_1$ nor $\boldsymbol{u}_1 <_p \boldsymbol{u}_2$. However, $\boldsymbol{u}_3 <_p \boldsymbol{u}_1$ since $u_{3,1} < u_{1,1}$ and $u_{3,2} < u_{1,2}$.*

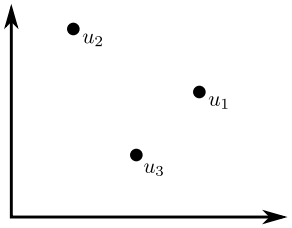


Figure 2.1: Vector-valued comparison of points: The points $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ are not ordered since $\boldsymbol{u}_2 \not<_p \boldsymbol{u}_1$ and $\boldsymbol{u}_1 \not<_p \boldsymbol{u}_2$. On the other hand $\boldsymbol{u}_3 <_p \boldsymbol{u}_1$.

A consequence of the lack of a total order is that we cannot expect to find isolated optimal points. Instead, the solution to (MOP) is the set of optimal compromises (also called the *set of non-dominated points*):

Definition 2.1.5. *Consider the multiobjective optimization problem (MOP). Then*

(a) a point $\boldsymbol{u}^*$ dominates *a point $\boldsymbol{u}$, if $\boldsymbol{J}(\boldsymbol{u}^*) \leq_p \boldsymbol{J}(\boldsymbol{u})$ and $\boldsymbol{J}(\boldsymbol{u}^*) \neq \boldsymbol{J}(\boldsymbol{u})$.*

(b) a feasible point $\boldsymbol{u}^*$ is called globally Pareto optimal *if there exists no feasible point $\boldsymbol{u} \in \mathcal{U}$ dominating $\boldsymbol{u}^*$. The image $\boldsymbol{J}(\boldsymbol{u}^*)$ of a globally Pareto optimal point $\boldsymbol{u}^*$ is called a* globally Pareto optimal value. *If this property holds in a neighborhood $U(\boldsymbol{u}^*) \subset \mathcal{U}$, then $\boldsymbol{u}^*$ is called* locally Pareto optimal.

(c) the set of non-dominated feasible points is called the Pareto set *$\mathcal{P}_S$, its image the* Pareto front *$\mathcal{P}_F$.*

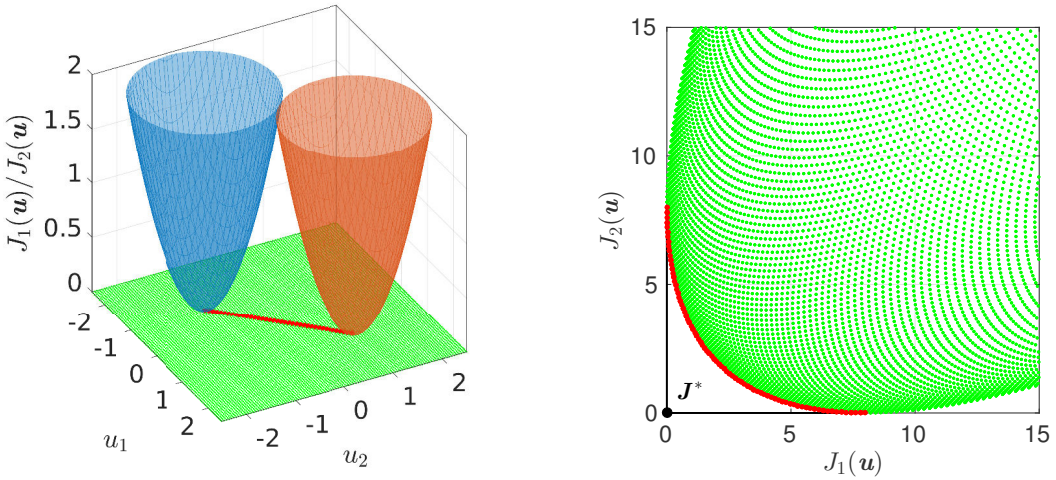
Remark 2.1.6. *The concept of non-dominance can be extended to sets, which will be utilized in the gradient-free version of the subdivision algorithm (cf. Section 2.1.5): a set $\mathcal{B}^*$* dominates *a set $\mathcal{B}$ if for every $\boldsymbol{u} \in \mathcal{B}$ there exists at least one $\boldsymbol{u}^* \in \mathcal{B}^*$ dominating $\boldsymbol{u}$.*

Remark 2.1.7. *Pareto optimal points are also known as* efficient points *[Ehr05, Mie12] or* non-inferior points *[VH83] in the literature.*

A consequence of Definition 2.1.5 is that for each point that is contained in the Pareto set (the red line in Figure 2.2 (a)), one can only improve one objective by accepting a trade-off in at least one other objective. Figuratively speaking, in a two-dimensional problem, we are interested in finding the "lower left" boundary of the feasible set in objective space (cf. Figure 2.2 (b)). A lower bound for each of the points contained in the Pareto front is the so-called *utopian point* $\boldsymbol{J}^*$ with

$$J_i^* = \min_{\boldsymbol{u} \in \mathcal{U}} J_i(\boldsymbol{u})$$

such that $\boldsymbol{J}^* \leq_p \boldsymbol{J}(\boldsymbol{u})\ \forall \boldsymbol{u} \in \mathcal{U}$. Several algorithms for solving MOPs make use of $\boldsymbol{J}^*$.



(a) (b)

Figure 2.2: The red lines are the Pareto set (a) and Pareto front (b) of an exemplary multiobjective optimization problem (two paraboloids) of the form $\min_{\boldsymbol{u} \in \mathbb{R}} \boldsymbol{J}(\boldsymbol{u})$, $\boldsymbol{J} : \mathbb{R}^2 \to \mathbb{R}^2$. The point $\boldsymbol{J}^* = (0, 0)^\top$ is called the utopian point.

Similar to single objective optimization, a necessary condition for optimality is based on the gradients of the objective functions. The first order conditions were independently discovered by Karush in 1939 [Kar39] and by Kuhn and Tucker in 1951 [KT51, Kje00]. Due to this, they are widely known as the *Karush-Kuhn-Tucker (KKT)* conditions. The theorem with multiple objectives is as follows:

Theorem 2.1.8 ([KT51])**.** *Let $\boldsymbol{u}^*$ be a Pareto optimal point of problem (2.1) and assume that $\nabla h_j(\boldsymbol{u}^*)$ for $j = 1, \ldots, m$, and $\nabla g_s(\boldsymbol{u}^*)$ for $s = 1, \ldots, l$, are linearly independent. Then there exist non-negative scalars $\alpha_1, \ldots, \alpha_k \geq 0$ with $\sum_{i=1}^{k} \alpha_i = 1$,*

$\boldsymbol{\gamma} \in \mathbb{R}^m$ *and* $\boldsymbol{\mu} \in \mathbb{R}^l$ *such that*

$$\begin{aligned} \sum_{i=1}^{k} \alpha_i \nabla J_i(\boldsymbol{u}^*) + \sum_{j=1}^{m} \gamma_j \nabla h_j(\boldsymbol{u}^*) + \sum_{s=1}^{l} \mu_s \nabla g_s(\boldsymbol{u}^*) &= \mathbf{0}, \\ h_j(\boldsymbol{u}^*) &= 0, \quad j = 1, \dots, m, \\ g_s(\boldsymbol{u}^*) &\leq 0, \quad s = 1, \dots, l, \\ \mu_s g_s(\boldsymbol{u}^*) &= 0, \quad s = 1, \dots, l, \\ \mu_s &\geq 0, \quad s = 1, \dots, l. \end{aligned} \tag{KKT}$$

Observe that (KKT) is only a necessary condition for a point $\boldsymbol{u}^*$ to be Pareto optimal and the set of points satisfying (KKT) is called the set of *substationary points* $\mathcal{P}_{S,\mathsf{sub}}$ [Mie12]. Obviously, $\mathcal{P}_{S,\mathsf{sub}}$ is a superset of the Pareto set $\mathcal{P}_S$.

Remark 2.1.9. *An intuitive interpretation of the multiobjective KKT condition is given in [Hil01] using the well-known* weighted sum *method (cf. Section 2.1.4). By introducing* $J_{\boldsymbol{\alpha}}(\boldsymbol{u}) = \sum_{i=1}^{k} \alpha_i J_i(\boldsymbol{u})$ *for a fixed weight vector* $\boldsymbol{\alpha} \in \mathbb{R}^k$*, problem* (2.1) *is transformed into a scalar optimization problem. The gradient of* $J_{\boldsymbol{\alpha}}(\boldsymbol{u})$ *is simply* $\nabla J_{\boldsymbol{\alpha}}(\boldsymbol{u}) = \sum_{i=1}^{k} \alpha_i \nabla J_i(\boldsymbol{u})$ *and the first order condition of this scalar problem is equal to* (KKT)*.*

2.1.2 Gradients and Descent Directions in Multiobjective Optimization

In this section we will revisit some results considering gradients for multiobjective optimization problems. While it is widely acknowledged in scalar optimization that utilizing gradient information is beneficial, this is different for multiobjective optimization [Bos12]. Apart from scalarization methods (cf. Section 2.1.4), many other solution approaches are gradient-free. Nevertheless, several authors have addressed gradients for MOPs. In [FS00, SSW02, Dés12], algorithms are developed where a single descent direction is computed in which all objectives decrease. An extension of Newton's method to MOPs with quadratic convergence is developed in [FGS09] and an algorithm by which the entire set of descent directions can be determined is presented in [Bos12].

In what follows, we will only consider unconstrained MOPs, i.e. $\mathcal{U} = \mathbb{R}^n$. The

corresponding KKT condition is then as follows:

$$\begin{aligned} \sum_{i=1}^{k} \alpha_i \nabla J_i(\boldsymbol{u}^*) &= \mathbf{0}, \\ \boldsymbol{\alpha} &\geq_p \mathbf{0}, \\ \sum_{i=1}^{k} \alpha_i &= 1. \end{aligned} \tag{KKTu}$$

In this situation, if $\boldsymbol{u} \notin \mathcal{P}_{S,\mathsf{sub}}$ then (KKTu) can be utilized to identify a descent direction $\boldsymbol{q}(\boldsymbol{u})$ for which all objectives are decreasing, i.e.:

$$-\nabla J_i(\boldsymbol{u}) \cdot \boldsymbol{q}(\boldsymbol{u}) > 0, \quad i = 1, \ldots, k. \tag{2.2}$$

One way to compute a descent direction satisfying (2.2) is to solve the following auxiliary optimization problem:

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^k} \left\{ \left\| \sum_{i=1}^{k} \alpha_i \nabla J_i(\boldsymbol{u}) \right\|_2^2 \;\middle|\; \boldsymbol{\alpha} \geq_p \mathbf{0}, \sum_{i=1}^{k} \alpha_i = 1 \right\}. \tag{QOP}$$

Using (QOP) we obtain the following result:

Theorem 2.1.10 ([SSW02])**.** *Consider the unconstrained multiobjective optimization problem* (MOP) *with* $\mathcal{U} = \mathbb{R}^n$ *and define* $\boldsymbol{q} : \mathbb{R}^n \to \mathbb{R}^n$ *as*

$$\boldsymbol{q}(\boldsymbol{u}) = -\sum_{i=1}^{k} \widehat{\alpha}_i \nabla J_i(\boldsymbol{u}), \tag{2.3}$$

where $\widehat{\boldsymbol{\alpha}}$ *is a solution of* (QOP)*. Then either* $\boldsymbol{q}(\boldsymbol{u}) = \mathbf{0}$ *and* $\boldsymbol{u}$ *satisfies* (KKTu)*, or* $\boldsymbol{q}(\boldsymbol{u})$ *is a descent direction for all objectives* $J_1(\boldsymbol{u}), \ldots, J_k(\boldsymbol{u})$ *in* $\boldsymbol{u}$*. Moreover,* $\boldsymbol{q}(\boldsymbol{u})$ *is locally Lipschitz continuous.*

Remark 2.1.11. *Similar to scalar optimization, there exist in general infinitely many valid descent directions* $\boldsymbol{q}(\boldsymbol{u})$ *and by solving* (QOP)*, we obtain one particular direction. As stated above, there are alternative ways to compute such a direction, see e.g. [FS00] for the computation of a single direction or [Bos12], where the entire set of descent directions is determined.*

Example 2.1.12. *An example for the descent direction* $\boldsymbol{q}(\boldsymbol{u})$ *is shown in Figure 2.3 using a simple one-parametric problem with* $\boldsymbol{J} : \mathbb{R} \to \mathbb{R}^2$*:*

$$\min_{u \in \mathbb{R}} \boldsymbol{J}(u) = \min_{u \in \mathbb{R}} \begin{pmatrix} u^4 - u^3 - 4u^2 \\ (u-1)^2 \end{pmatrix}. \tag{2.4}$$

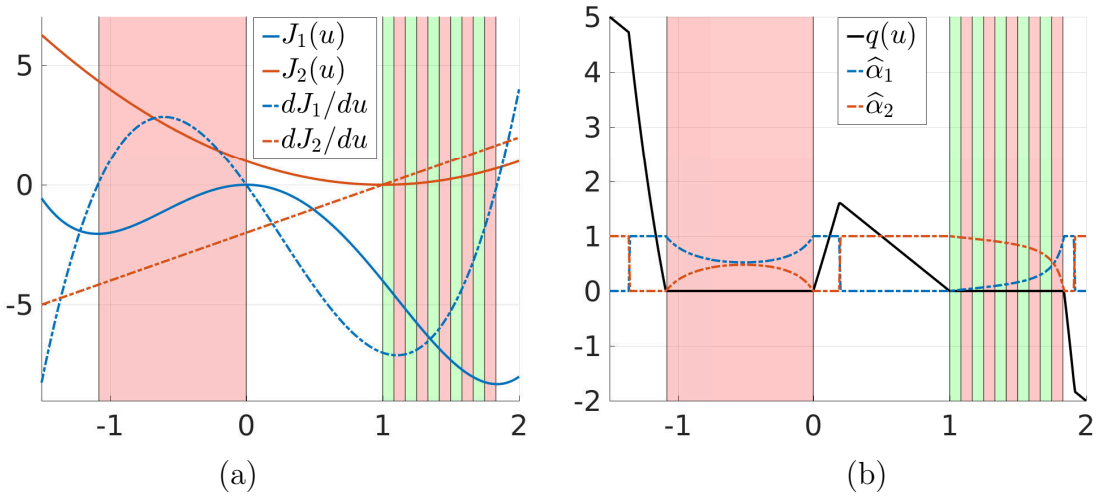


Figure 2.3: Example (2.4). (a) The two objective functions and the respective derivatives. (b) The descent direction $q(u)$ and the corresponding KKT weights $\widehat{\alpha}_1$ and $\widehat{\alpha}_2$. In both figures, the set of substationary points is marked in red and the Pareto set is marked in green.

Observe that the respective parts of the set of substationary points are bounded by points where one of the scalar objectives possesses an extreme point. (This will be addressed in detail in Chapter 4.) Correspondingly, the weights $\widehat{\boldsymbol{\alpha}}$ at these points are either $(1,0)$ or $(0,1)$. On the set of substationary points, the weights vary smoothly (if the objective functions are smooth enough [Hil01]). This fact is utilized in the weighted sum method, *cf. Section 2.1.4. Outside the set, the weights may possess jumps, e.g. when the norm of the two gradients changes from $\|\nabla J_1\| > \|\nabla J_2\|$ to $\|\nabla J_2\| > \|\nabla J_1\|$, see Figure 2.3 (b) at $u = -1.36$, $u = 0.19$ and $u = 1.92$, respectively.*

Using a descent direction for all objectives, we can utilize gradient-based methods known from scalar optimization theory (see e.g. [NW06]) which often results in accelerated convergence to an optimal solution. Line search methods are a very popular iterative approach where in addition to a direction $\boldsymbol{p} \in \mathbb{R}^n$ (i.e. a gradient-based direction), a step length $h \in \mathbb{R}$ has to be computed such that a new iterate

$$\boldsymbol{u}^{(j+1)} = \boldsymbol{u}^{(j)} + h^{(j)}\boldsymbol{p}^{(j)}, \tag{2.5}$$

is an improvement of the current iterate $\boldsymbol{u}^{(j)}$. The descent direction $\boldsymbol{p}$ is chosen in such a way that for constants $\sigma, \tau > 0$,

$$\frac{\boldsymbol{q}^\top(\boldsymbol{u}^{(j)})\boldsymbol{p}^{(j)}}{\|\boldsymbol{q}(\boldsymbol{u}^{(j)})\|\|\boldsymbol{p}^{(j)}\|} \geq \sigma, \qquad \frac{\|\boldsymbol{q}(\boldsymbol{u}^{(j)})\|}{\|\boldsymbol{p}^{(j)}\|} \geq \tau, \tag{2.6}$$

where $\boldsymbol{q}(\boldsymbol{u}^{(j)})$ is the direction computed by solving (QOP). If (2.6) is satisfied and $h^{(j)}$ is a feasible step size (e.g. the Armijo or the Powell step size [NW06]), then one can show that every accumulation point $\boldsymbol{u}^{(*)}$ of the sequence (2.5) satisfies condition (KKTu) of the unconstrained problem. This result was proved in [DSH05], it is stated here for completeness. In order to simplify the notation, we introduce the linear combination $J_{\boldsymbol{\alpha}}(\boldsymbol{u}) = \sum_{i=1}^{k} \alpha_i J_i(\boldsymbol{u})$ of the objective functions of (MOP), where $\boldsymbol{\alpha} \in [0,1]^k$.

Theorem 2.1.13 ([DSH05]). *Suppose that $\boldsymbol{u}^{(*)}$ is an accumulation point of the sequence $(\boldsymbol{u}^{(j)})_{j=0,1,\dots}$ created by (2.5). We assume that the derivative of each function $J_{\boldsymbol{\alpha}}$ is Lipschitz continuous with (uniform) Lipschitz constant L. Then $\boldsymbol{u}^{(*)}$ is a substationary point for the multiobjective optimization problem (MOP) with $\mathcal{U} = \mathbb{R}^n$.*

Proof. First observe that if one of the $\boldsymbol{u}^{(j)}$ is a substationary point, then we are done (cf. Theorem 2.1.10). Thus, without loss of generality, we may assume that $\boldsymbol{u}^{(j)} \to \boldsymbol{u}^{(*)}$ for $j \to \infty$ and that none of the $\boldsymbol{u}^{(j)}$ is substationary. Consider the corresponding sequence $\widehat{\boldsymbol{\alpha}}^{(j)} = (\widehat{\alpha}_1^{(j)}, \dots, \widehat{\alpha}_k^{(j)})$ of solution vectors of the optimization problem (QOP) in step j of the iteration procedure. Since $\alpha_1^{(j)}, \dots, \alpha_k^{(j)} \in [0,1]$, we may assume that $\widehat{\boldsymbol{\alpha}}^{(j)} \to \widehat{\boldsymbol{\alpha}}$ for $j \to \infty$. Otherwise, we restrict the following considerations to a subsequence. We show now that the sequence $(\boldsymbol{u}^{(j)})$ converges to a stationary point for $J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u})$, thus proving the desired result. Using the fact that $h^{(j)}$ is an Armijo or a Powell step size in $\boldsymbol{u}^{(j)}$, we have by classical results on iteration schemes for optimization problems (see e.g. [DS83]) that there exists a constant $\Theta > 0$ such that

$$J_{\widehat{\boldsymbol{\alpha}}^{(j)}}(\boldsymbol{u}^{(j)}) - J_{\widehat{\boldsymbol{\alpha}}^{(j)}}(\boldsymbol{u}^{(j+1)}) \geq \\ \Theta \min\left(-\left(\nabla J_{\widehat{\boldsymbol{\alpha}}^{(j)}}(\boldsymbol{u}^{(j)})\right)^{\top} \boldsymbol{p}^{(j)}, \left(\frac{\left(\nabla J_{\widehat{\boldsymbol{\alpha}}^{(j)}}(\boldsymbol{u}^{(j)})\right)^{\top} \boldsymbol{p}^{(j)}}{\|\boldsymbol{p}^{(j)}\|} \right)^2 \right) \tag{2.7}$$

in each step of the iteration process. Observe that Θ does not depend on j by the assumption on the uniform Lipschitz continuity of $\nabla J_{\boldsymbol{\alpha}}$. Now, suppose that

$$J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(j)}) - J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(j+1)}) < \Theta \min\left(-\left(\nabla J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(j)})\right)^{\top} \boldsymbol{p}^{(j)}, \left(\frac{\left(\nabla J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(j)})\right)^{\top} \boldsymbol{p}^{(j)}}{\|\boldsymbol{p}^{(j)}\|} \right)^2 \right) \tag{2.8}$$

for infinitely many j. By our assumption on the descent direction we have

$$\frac{-\left(\nabla J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(j)})\right)^{\top} \boldsymbol{p}^{(j)}}{\|\nabla J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(j)})\| \|\boldsymbol{p}^{(j)}\|} \geq \sigma, \qquad \|\boldsymbol{p}^{(j)}\| \geq (\tau/2) \|\nabla J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(j)})\|, \tag{2.9}$$

for all $j \geq j_0$. Combining these estimates with (2.7) and (2.8), we obtain

$$\begin{aligned} 0 &= \lim_{j\to\infty} \min \left[-\left(\nabla J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(j)})\right)^\top \boldsymbol{p}^{(j)}, \left(\frac{\left(\nabla J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(j)})\right)^\top \boldsymbol{p}^{(j)}}{\|\boldsymbol{p}^{(j)}\|} \right)^2 \right] \\ &\geq (\Theta\sigma/4) \min(\tau, \sigma) \|\nabla J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(*)})\|^2, \end{aligned}$$

and $\boldsymbol{u}^{(*)}$ is substationary as desired.

It remains to consider the case where

$$J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(j)}) - J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(j+1)}) \geq \Theta \min \left[-\left(\nabla J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(j)})\right)^\top \boldsymbol{p}^{(j)}, \left(\frac{\left(\nabla J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(j)})\right)^\top \boldsymbol{p}^{(j)}}{\|\boldsymbol{p}^{(j)}\|} \right)^2 \right]$$

for all $j \geq j_1$. Here, we obtain in an analogous way

$$J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(j)}) - J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(j+1)}) \geq (\Theta\sigma/4) \min(\tau, \sigma) \|\nabla J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(*)})\|^2$$

for all $j \geq \max(j_0, j_1)$. Letting $j \to \infty$, it follows that

$$\|\nabla J_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{u}^{(*)})\| = 0.$$

□

Remark 2.1.14. *The iteration step (2.5) can also be interpreted as a dynamical system of which the attracting set is a superset of the the set of substationary points $\mathcal{P}_{S,\mathsf{sub}}$. If $\mathcal{P}_{S,\mathsf{sub}}$ is bounded and connected, then the attractor is precisely $\mathcal{P}_{S,\mathsf{sub}}$. This will be utilized in the subdivision algorithm presented in Section 2.1.5.*

2.1.3 Manifold Conditions for Pareto Sets

It was shown by Hillermeier [Hil01] that under certain smoothness assumptions (i.e. the objective functions and the constrains are twice continuously differentiable), the set of substationary points $\mathcal{P}_{S,\mathsf{sub}}$ is locally a $(k-1)$-dimensional manifold. This additional structure has vast implications for the development of algorithms for MOPs. A consequence is that in many situations, the set of substationary points is connected. Hence, for two parameters $\boldsymbol{u}^{(1)}, \boldsymbol{u}^{(2)}$ that are close to each other, the corresponding points on the Pareto front are also close. This can be utilized when the Pareto set is approximated by a finite set of Pareto optimal points, see Section 2.1.4. Another implication is that a hyperplane tangential to the Pareto set exists at every point. This is the foundation of continuation methods. In this section we want to recall which additional conditions the objective function $\boldsymbol{J}$ has to satisfy for $\mathcal{P}_{S,\mathsf{sub}}$ to be a manifold.

Taking another look at (KKT), the points satisfying the optimality conditions can be computed by solving a zero finding problem. For this, we introduce a new set of constraints $\widetilde{\boldsymbol{h}}(\boldsymbol{u})$ which consists of the equality constraints $\boldsymbol{h}(\boldsymbol{u})$ and active inequality constraints $\boldsymbol{g}_{\mathcal{I}}(\boldsymbol{u})$, where $\mathcal{I} = \{i \in \{1, \dots, l\} \mid g_i(\boldsymbol{u}) = 0\}$ denotes the set of active inequalities in $\boldsymbol{u}$. Using this notation, (KKT) can be rewritten as:

$$\sum_{i=1}^{k} \alpha_i \nabla J_i(\boldsymbol{u}^*) + \sum_{j=1}^{m+|\mathcal{I}|} \widetilde{\gamma}_j \nabla \widetilde{h}_j(\boldsymbol{u}^*) = \boldsymbol{0},$$
$$\widetilde{h}_j(\boldsymbol{u}^*) = 0, \quad j = 1, \dots, m + |\mathcal{I}|.$$

Then introducing $\boldsymbol{H} : \mathbb{R}^{n+m+|\mathcal{I}|+k} \to \mathbb{R}^{n+m+|\mathcal{I}|+1}$, (KKT) is equivalent to

$$\boldsymbol{H}(\boldsymbol{u}^*, \widetilde{\boldsymbol{\gamma}}^*, \boldsymbol{\alpha}^*) = \begin{pmatrix} \sum_{i=1}^{k} \alpha_i^* \nabla J_i(\boldsymbol{u}^*) + \sum_{j=1}^{m+|\mathcal{I}|} \widetilde{\gamma}_j^* \nabla \widetilde{h}_j(\boldsymbol{u}^*) \\ \widetilde{\boldsymbol{h}}(\boldsymbol{u}^*) \\ \sum_{i=1}^{k} \alpha_i^* - 1 \end{pmatrix} = \boldsymbol{0}. \tag{2.10}$$

Consequently, the set of points satisfying (KKT) can be computed by finding a curve $\boldsymbol{c}(\boldsymbol{\phi}) = \boldsymbol{H}^{-1}(\boldsymbol{0})$, where $\boldsymbol{\phi} \in \mathbb{R}^{k-1}$ is a parametrization of $\boldsymbol{c}$.

Applying the Implicit Function Theorem (see e.g. [KP03]), one can show that the set of substationary points is a $(k-1)$-dimensional manifold:

Theorem 2.1.15 ([Hil01]). *Let $\mathcal{M} := \{(\boldsymbol{u}^*, \widetilde{\boldsymbol{\gamma}}^*, \boldsymbol{\alpha}^*) \in \mathbb{R}^{n+m+|\mathcal{I}|+k} \mid \boldsymbol{H}(\boldsymbol{u}^*, \widetilde{\boldsymbol{\gamma}}^*, \boldsymbol{\alpha}^*) = \boldsymbol{0}$ and $\boldsymbol{\alpha}^* >_p \boldsymbol{0}\}$. If the Jacobian $\boldsymbol{H}'$ has full rank in one point $(\boldsymbol{u}^*, \widetilde{\boldsymbol{\gamma}}^*, \boldsymbol{\alpha}^*)$, i.e.*

$$rank(\boldsymbol{H}'(\boldsymbol{u}^*, \widetilde{\boldsymbol{\gamma}}^*, \boldsymbol{\alpha}^*)) = n + m + |\mathcal{I}| + 1, \tag{2.11}$$

then $\mathcal{M}$ is a $(k-1)$-dimensional differentiable submanifold of $\mathbb{R}^{n+m+|\mathcal{I}|+k}$ in a neighborhood of $(\boldsymbol{u}^*, \widetilde{\boldsymbol{\gamma}}^*, \boldsymbol{\alpha}^*)$.

If all points $(\boldsymbol{u}^*, \widetilde{\boldsymbol{\gamma}}^*, \boldsymbol{\alpha}^*) \in \mathcal{M}$ satisfy the rank condition (2.11) then $\mathcal{M}$ is a $(k-1)$-dimensional differentiable submanifold of $\mathbb{R}^{n+k}$.

Proof. The proof follows directly from the Implicit Function Theorem, see [Hil01]. □

Remark 2.1.16. *Theorem 2.1.15 is also valid in many situations where $\boldsymbol{J}(\mathcal{U})$ is non-convex. This is determined by the eigenvalues of the Hessian of $J_{\boldsymbol{\alpha}}$. If all eigenvalues are greater than zero in a neighborhood $U(\boldsymbol{u}^*)$ of $\boldsymbol{u}^*$, $\boldsymbol{J}(\boldsymbol{u}^*)$ is locally convex and $\boldsymbol{u}^*$ is locally Pareto optimal. If at least one eigenvalue is less than zero, then $\boldsymbol{u}^*$ is a saddle point of the scalar function $J_{\boldsymbol{\alpha}}$. In this situation, Pareto optimality has to be confirmed by higher order information. Nonetheless, the rank condition is conserved. In between these situations, at least one eigenvalue is exactly zero and the Implicit*

Function Theorem is no longer valid. Here, the manifold property is conserved only if there exists at least one objective function with $\nabla J_i(\boldsymbol{u}^*) \cdot \boldsymbol{v} \neq 0$*, where* $\boldsymbol{v}$ *is the eigenvector corresponding to the zero eigenvalue. For a more detailed discussion, see [Hil01, pp. 68–75].*

The above result is used in many algorithms referred to as *continuation methods*, see e.g. [SDD05, CLS16, MS17]. This concept will be extended in Chapter 3 to continuation methods for entire Pareto sets and in Chapter 4, the resulting additional structure will be exploited to hierarchically compute Pareto sets by considering subsets of objectives.

2.1.4 Solution Methods

Many researchers in multiobjective optimization focus their attention on developing efficient algorithms for the computation of Pareto sets. The intention of this section is to provide an overview of widely used solution methods for MOPs and to discuss their respective advantages and disadvantages. This survey is by far not exhaustive since a lot of different approaches and variations of algorithms exist. Instead, the most important methods will be addressed, many of which will be used later on in this thesis. For a more comprehensive overview, the reader is referred to e.g. [Ehr05, Mie12], a review of more recent advances can be found in [CP07].

Algorithms for solving MOPs can be compiled into several fundamentally different categories of approaches. The first category is based on *scalarization techniques*, where ideas from single objective optimization theory are extended to the multiobjective situation. Consequently, the resulting solution method involves solving multiple scalar optimization problems consecutively. *Continuation methods* make use of the properties presented in Section 2.1.3. Another prominent approach is based on *evolutionary algorithms* [CLV07], where the underlying idea is to evolve an entire population of solutions during the optimization process. *Set-oriented methods* provide an alternative deterministic approach to the solution of MOPs. Utilizing subdivision techniques, the desired Pareto set is approximated by a nested sequence of increasingly refined box coverings [DSH05, Jah06, SWOBD13]. Since most of the results in this thesis are based on set-oriented algorithms, these will be described in more detail in the subsequent Section 2.1.5.

Scalarization Methods

All scalarization techniques have in common that the Pareto set is approximated by a finite set of Pareto optimal points which are computed by solving scalar sub-

problems. In many cases, the Pareto set possesses a structure (i.e. it is a manifold) and is thereby connected. Hence, two consecutive subproblems have solutions which are close to each other. This can be utilized to considerably accelerate the convergence. Nevertheless, MOPs are in general much more costly to solve than scalar optimization problems. Moreover, scalarization approaches are often inapplicable to problems with a large number of objectives since the parametrization quickly becomes tedious or even intractable.

The subproblems can be solved using any suitable method from scalar optimization, see e.g. [NW06] for an overview. Note that since the scalar optimization routines often are of local nature, these methods are also local and depend on the initial guesses. However, when the Pareto set is a manifold (cf. Section 2.1.3), these methods are promising to find the globally optimal Pareto set provided that an initial point can be accurately computed, e.g. by scalar optimization of one of the objectives. In the following, several popular scalarization techniques will be introduced, namely, the *weighted sum method*, the *ϵ-constraint method*, *normal boundary intersection* and the *reference point method*.

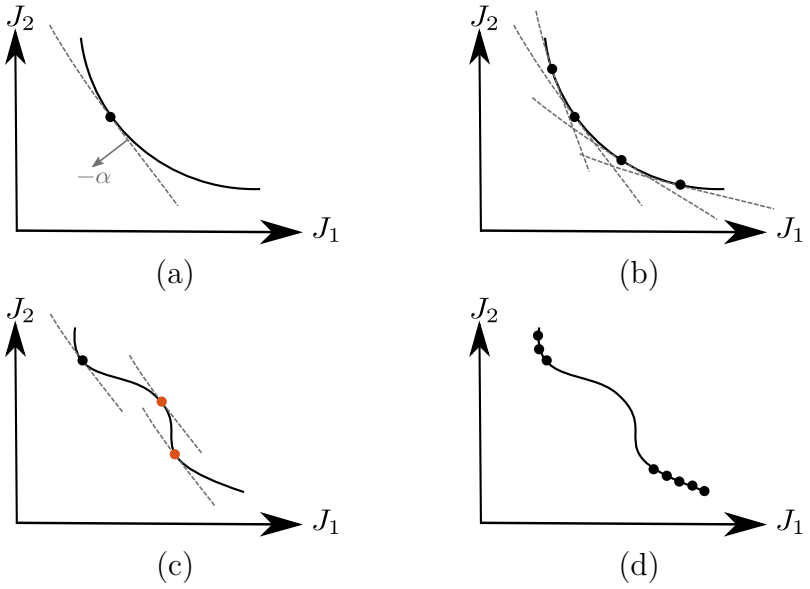


Figure 2.4: Weighted sum method. (a) Computation of one Pareto optimal point for a fixed weight vector $\boldsymbol{\alpha}$ which is the normal vector of a $(k-1)$-dimensional hyperplane. (b) Computation of multiple Pareto optimal points by varying $\boldsymbol{\alpha}$. (c)–(d) Non-convex sets $\boldsymbol{J}(\mathcal{U})$ lead to multiple Pareto optimal points with the same weight $\boldsymbol{\alpha}$. In particular, not all points can be computed using the weighted sum method.

In the *weighted sum method*, scalarization is achieved by introducing a weight vector $\boldsymbol{\alpha} \in \mathbb{R}^k$. The scalar objective is then a convex combination of the objective

functions $J_1, \ldots, J_k$:

$$\begin{aligned} \min_{\boldsymbol{u} \in \mathcal{U}} J_{\boldsymbol{\alpha}}(\boldsymbol{u}) &= \min_{\boldsymbol{u} \in \mathcal{U}} \sum_{i=1}^{k} \alpha_i J_i(\boldsymbol{u}), \\ \boldsymbol{\alpha} &\geq_p \mathbf{0}, \\ \sum_{i=1}^{k} \alpha_i &= 1. \end{aligned} \tag{2.12}$$

The vector $\boldsymbol{\alpha}$ defines a $(k-1)$-dimensional hyperplane tangential to the Pareto front (cf. Figure 2.4 (a)). By varying $\boldsymbol{\alpha}$, different Pareto optimal points can be computed and if the set $\boldsymbol{J}(\mathcal{U})$ is convex, the entire Pareto front can be approximated this way (cf. Figure 2.4 (b)). However, in the non-convex case, several Pareto optimal points can be described by the same value of $\boldsymbol{\alpha}$ such that gaps can occur (cf. Figures 2.4 (c) and (d)). Furthermore, varying $\boldsymbol{\alpha}$ in equidistant steps does not necessarily yield an equidistant covering of the Pareto front [DD97].

In the *ϵ-constraint method*, all but one objectives are transformed to constraints

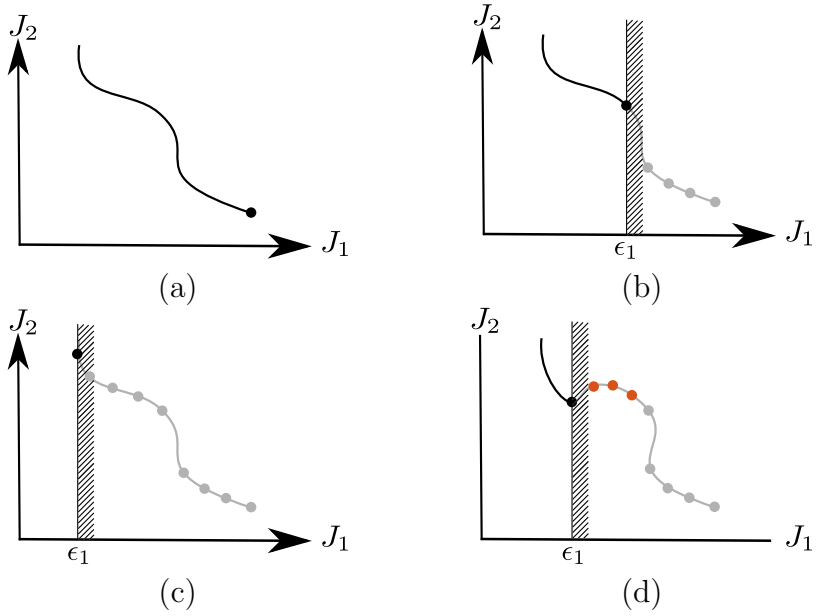


Figure 2.5: ϵ-constraint method ($\min_{\boldsymbol{u} \in \mathcal{U}} J_2(\boldsymbol{u})$ subject to $J_1(\boldsymbol{u}) \leq \epsilon_1$). (a) $\epsilon_1 = \infty$. (b) Intermediate value for ϵ_1. Large gaps in the Pareto front can occur. (c) The algorithm stops when the scalar minimum of J_1 is reached. (d) Previously Pareto optimal points can later on be dominated by newly computed points.

which leads to the problem

$$\begin{aligned} &\min_{\boldsymbol{u}\in\mathcal{U}} J_i(\boldsymbol{u}) \\ \text{s.t.} \quad &J_j(\boldsymbol{u}) \leq \epsilon_j, \quad j = 1,\dots,k,\ j \neq i, \end{aligned} \tag{2.13}$$

where the constraints $\boldsymbol{\epsilon} \in \mathbb{R}^{k-1}$ can be chosen. By setting $\epsilon_j = \infty$ for $j = 1,\dots,k,\ j \neq, i$, the scalar optimum of J_i is obtained when solving (2.13). By adjusting $\boldsymbol{\epsilon}$, a different (locally) Pareto optimal point is computed.

Using the ϵ-constraint method, it is possible to approximate Pareto sets where $\boldsymbol{J}(\mathcal{U})$ is not convex which is a clear advantage compared to the weighted sum method (cf. Figure 2.5). However, it is equally difficult to achieve an equidistant covering of the Pareto front. Moreover, convergence for the scalar problem (2.13) can become difficult with an increasing number of constraints [Ehr05]. An extension of the ϵ-constraint method where less than $k-1$ objectives are treated as constraints will be presented in Section 4.2.

Normal boundary intersection (NBI) was originally introduced in [DD98]. The general idea is to first compute the hyperplane spanned by the *convex hull of individual minima (CHIM)* $\boldsymbol{\Phi} : \mathbb{R}^n \times \mathbb{R}^k \to \mathbb{R}^k$:

$$\begin{aligned} \boldsymbol{\Phi}(\boldsymbol{u},\boldsymbol{\beta}) &= \sum_{i=1}^{k} \beta_i \left(J_i(\boldsymbol{u})\boldsymbol{e}_i - \boldsymbol{J}^*\right), \\ \boldsymbol{\beta} &\geq_p \mathbf{0}, \\ \sum_{i=1}^{k} \beta_i &= 1, \end{aligned}$$

where $\boldsymbol{\beta} \in \mathbb{R}^k$ is a parametrization of CHIM and $\boldsymbol{e}_i$ denotes the i^{th} unit vector. Each objective is shifted by the scalar minimum (i.e. the respective component of the utopian point $\boldsymbol{J}^*$) such that all shifted objectives are non-negative (cf. Figure 2.6 (a) and (b)). After discretizing the hyperplane by a fixed number of points each of which is prescribed by a fixed value $\boldsymbol{\beta}$, we compute the point along the normal vector $\hat{\boldsymbol{n}}$ pointing to the origin with the maximum distance t to CHIM (cf. Figure 2.6 (c)):

$$\begin{aligned} &\max_{t\in\mathbb{R}^{\geq 0},\boldsymbol{u}\in\mathcal{U}} t \\ \text{s.t.} \quad &\boldsymbol{\Phi}(\boldsymbol{u},\boldsymbol{\beta}) + t\hat{\boldsymbol{n}} = \boldsymbol{J}(\boldsymbol{u}). \end{aligned} \tag{2.14}$$

In the situation where parts of the Pareto front lie above the hyperplane, the objective function in (2.14) has to be replaced by a more sophisticated *achievement scalarizing function (ASF)* (cf. Figure 2.6 (d)). Several researchers have addressed this problem, see e.g. [Wie80, Wie86], an overview is presented in [MM02].

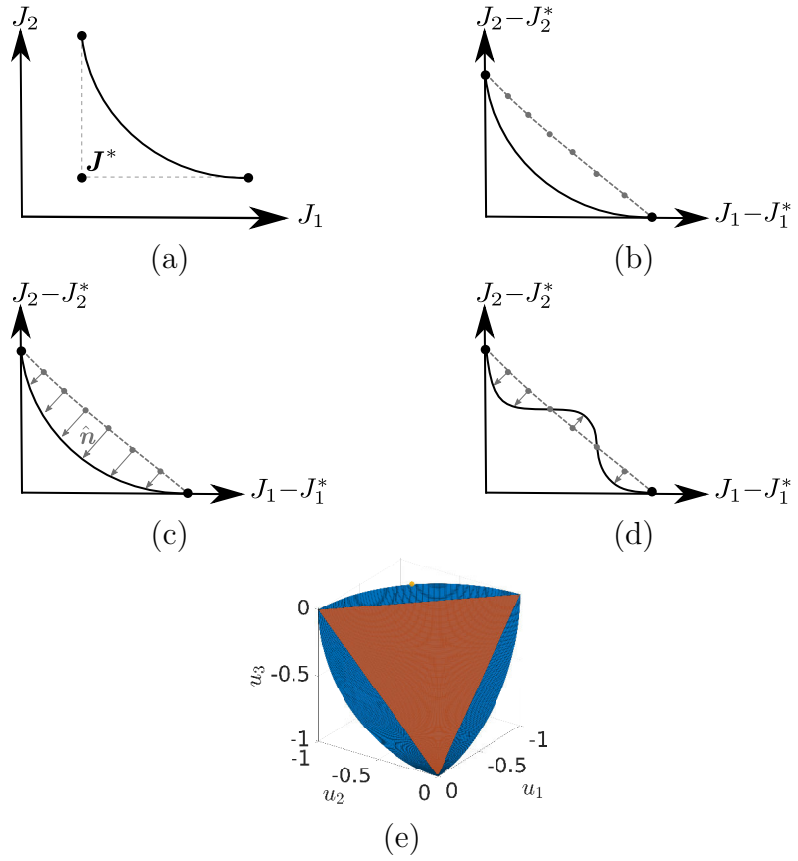


Figure 2.6: Normal boundary intersection. (a) Determination of the scalar minima J_i for $i = 1, \ldots, k$ yields $\boldsymbol{J}^*$. (b) Convex hull of individual minima (CHIM) and grid of starting points covering the CHIM. (c) Solution of scalar optimization problems. (d) Solution when parts of the Pareto front lie above the hyperplane defined by the CHIM. In this case the objective in problem (2.14) has to be replaced by a more advanced ASF. (e) Pareto front (blue) and CHIM (red) of Example (2.15), where not all Pareto optimal points can be computed, e.g. the yellow point.

NBI has advantages and disadvantages comparable to the ϵ-constraint method. While it is possible to solve problems where $\boldsymbol{J}(\mathcal{U})$ is non-convex, an equidistant discretization of the hyperplane spanned by the CHIM does not necessarily lead to a well-spread covering of the Pareto front. Moreover, this method is incapable of computing points that lie outside the "shadow" of the hyperplane. Consider for example the problem

$$\begin{aligned} \min_{\boldsymbol{u} \in \mathbb{R}^3} \quad & \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \\ \text{s.t.} \quad & \|\boldsymbol{u}\|_2 \leq 1. \end{aligned} \tag{2.15}$$

In this case, the individual minima are $(-1, 0, 0)$, $(0, -1, 0)$ and $(0, 0, -1)$ and the Pareto front is the part of the unit sphere lying in the negative octant (cf. Figure 2.6 (e)). Using NBI, it is not possible to compute all these points, a counterexample being the point $\left(-\sqrt{2}/2, -\sqrt{2}/2, 0\right)$.

The *reference point method* is very similar to normal boundary intersection. Here, the solution of (MOP) is again approximated by optimizing the distance between a Pareto optimal point and a reference point. In the beginning, one Pareto optimal point $\boldsymbol{u}^{(0)}$ has to be known. This can be achieved by solving a scalar optimization problem for some weighted sum of all objectives (including the scalar optimization with respect to any of the objectives of (MOP)). Then a so-called *target* $\boldsymbol{T}^{(1)} \in \mathbb{R}^k$ is chosen such that it lies outside the feasible set in objective space, e.g. by shifting the solution of the first Pareto point ($\boldsymbol{T}^{(1)} = \boldsymbol{J}(\boldsymbol{u}^{(0)}) - (h_\parallel, 0, \ldots, 0)^\top, h_\parallel > 0$). We then solve the scalar optimization problem

$$\min_{\boldsymbol{u}^{(i)} \in \mathcal{U}} \| \boldsymbol{T}^{(i)} - \boldsymbol{J}(\boldsymbol{u}^{(i)}) \|_2^2 \tag{2.16}$$

with $i = 1$. As a result, the corresponding optimal point $\boldsymbol{J}(\boldsymbol{u}^{(1)})$ lies on the boundary

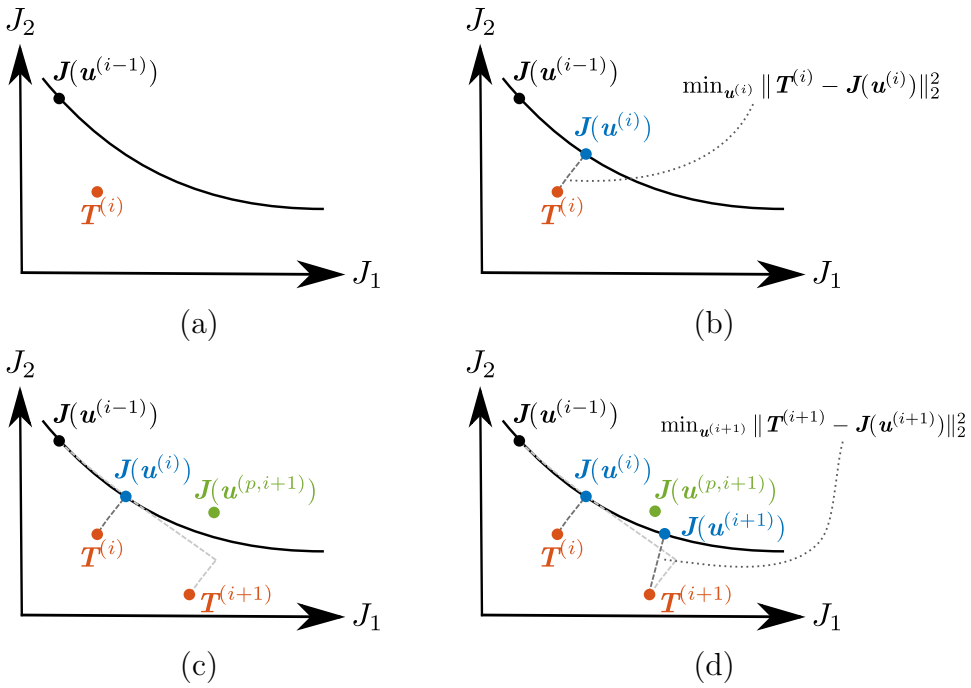


Figure 2.7: Reference point method. (a) Initial point $\boldsymbol{J}(\boldsymbol{u}^{(i-1)})$ and initial target $\boldsymbol{T}^{(i)}$. (b) Computation of next Pareto optimal point by solving the scalar optimization problem (2.16). (c) Computation of next target point $\boldsymbol{T}^{(i+1)}$ and prediction step $\boldsymbol{u}^{(p,i+1)}$ in decision space. (d) Solution of the next scalar problem.

of the feasible set and it is not possible to further improve all objectives at the same time. Thus, the point is (locally) Pareto optimal. This concept is also referred to as *compromise programming*, see [Ehr05] where a proof of Pareto optimality for problem (2.16) can be found. Since this will be utilized in Section 5.2 in the development of a multiobjective trust region method for PDE-constrained problems, it is stated here:

Theorem 2.1.17 ([Ehr05])**.** *Consider the weighted compromise programming problem*

$$\min_{u\in\mathcal{U}} \left(\sum_{i=1}^{k} (T_i - J_i(\boldsymbol{u}))^p \right)^{\frac{1}{p}} = \min_{u\in\mathcal{U}} \| \boldsymbol{T} - \boldsymbol{J}(\boldsymbol{u}) \|_p \tag{2.17}$$

for $1 \leq p < \infty$, where $\boldsymbol{T}$ is the target point which is less than or equal to the utopian point, i.e. $\boldsymbol{T} \leq_p \boldsymbol{J}^*$. Then if $\boldsymbol{u}^*$ is a solution to (2.17), $\boldsymbol{u}^*$ is Pareto optimal.

Proof. Suppose that $\boldsymbol{u}^*$ is a solution to (2.17) and that $\boldsymbol{u}^*$ is not Pareto optimal, i.e. $\boldsymbol{u}^* \notin \mathcal{P}_S$. Then there exists a $\boldsymbol{u}'$ dominating $\boldsymbol{u}^*$, i.e. $\boldsymbol{J}(\boldsymbol{u}') \leq_p \boldsymbol{J}(\boldsymbol{u}^*)$ and $J_j(\boldsymbol{u}') < J_j(\boldsymbol{u}^*)$ for at least one $j \in \{1, \ldots, k\}$. Consequently, $0 \leq J_i(\boldsymbol{u}') - T_i \leq J_i(\boldsymbol{u}^*) - T_i$ for all $i = 1, \ldots, k$ and $0 \leq J_j(\boldsymbol{u}') - T_j < J_j(\boldsymbol{u}^*) - T_j$ for at least one $j \in \{1, \ldots, k\}$. Thus, $\| \boldsymbol{T} - \boldsymbol{J}(\boldsymbol{u}') \|_p \leq \| \boldsymbol{T} - \boldsymbol{J}(\boldsymbol{u}^*) \|_p$ which is a contradiction to $\boldsymbol{u}^*$ being a solution to (2.17). □

Remark 2.1.18. *In practice, the requirement $\boldsymbol{T} \leq_p \boldsymbol{J}^*$ is often not satisfied, see Figure 2.7 for an example. However, if $\boldsymbol{T}$ is chosen in such a way that $\boldsymbol{T} \leq_p \boldsymbol{J}(\boldsymbol{u}^*)$, then $\boldsymbol{u}^*$ is still Pareto optimal in a neighborhood $U(\boldsymbol{u}^*)$.*

By adjusting the target position based on already known targets and Pareto optimal points, multiple points on the Pareto front (i.e. $\boldsymbol{J}(\boldsymbol{u}^{(2)}), \boldsymbol{J}(\boldsymbol{u}^{(3)}), \ldots$) are computed recursively (cf. Figure 2.7). For $\boldsymbol{J}$ being sufficiently smooth, the change in decision space is small when the target position changes only slightly and hence, the current solution is a good initial guess for the next scalar problem which considerably accelerates the convergence.

Similar to the other scalarization approaches, properly setting the targets to obtain a good approximation of the Pareto front (i.e. an approximation of the entire front by evenly distributed points) becomes complicated in higher dimensions. However, when dealing with two objectives, the targets can easily be determined using linear extrapolation as proposed in [RBW+09] (cf. Figure 2.7 and Algorithm 2.1). This also allows us to compute the whole front in at most two loops ($s = 1, 2$, line 2 in Algorithm 2.1). From the initial point, we first proceed in one direction, e.g. decreasing J_1. When at some point J_1 is increasing again (line 6 in Algorithm 2.1), we have reached the *extreme point* of the feasible set and the scalar optimum of J_1 (cf. the point $(0, 8)$ in Figure 2.2 (b) for an example). We then return to the initial

point and proceed in the opposite direction (lines 8, 9 in Algorithm 2.1) until the other extreme point is reached.

Algorithm 2.1 (Reference point method for $\boldsymbol{J} \in \mathbb{R}^2$)

Require: Initial solution $\boldsymbol{u}^{(0)}$, parameters $h_\parallel, h_\perp, h_p \in \mathbb{R}^{>0}$, index $i = 0$

1: Compute the first target point $\boldsymbol{T}^{(1)} = \boldsymbol{J}(\boldsymbol{u}^{(0)}) - (h_\parallel, 0)^\top$
2: **for** $s = 1, 2$ **do**
3: **loop**
4: $i = i + 1$
5: Solve scalar optimization problem $\min_{\boldsymbol{u}^{(i)}} \| \boldsymbol{T}^{(i)} - \boldsymbol{J}(\boldsymbol{u}^{(i)})\|_2^2$
6: **if** extremal point of Pareto front is passed $(J_s(\boldsymbol{u}^{(i)}) > J_s(\boldsymbol{u}^{(i-1)}))$ **then**
7: **if** $s = 1$ (first direction is completed) **then**
8: $\boldsymbol{u}^{(p,i+1)} = \boldsymbol{u}^{(0)}$ (Go back to the initial solution)
9: $\boldsymbol{T}^{(i+1)} = \boldsymbol{J}(\boldsymbol{u}^{(0)}) - h_\parallel \frac{\boldsymbol{J}(\boldsymbol{u}^{(1)}) - \boldsymbol{J}(\boldsymbol{u}^{(0)})}{\|\boldsymbol{J}(\boldsymbol{u}^{(1)}) - \boldsymbol{J}(\boldsymbol{u}^{(0)})\|_2} + h_\perp \frac{\boldsymbol{T}^{(1)} - \boldsymbol{J}(\boldsymbol{u}^{(1)})}{\|\boldsymbol{T}^{(1)} - \boldsymbol{J}(\boldsymbol{u}^{(1)})\|_2}$ (Go into the opposite direction)
10: **break**
11: **else**
12: **STOP**
13: **end if**
14: **else**
15: $\boldsymbol{T}^{(i+1)} = \boldsymbol{J}(\boldsymbol{u}^{(i)}) + h_\parallel \frac{\boldsymbol{J}(\boldsymbol{u}^{(i)}) - \boldsymbol{J}(\boldsymbol{u}^{(i-1)})}{\|\boldsymbol{J}(\boldsymbol{u}^{(i)}) - \boldsymbol{J}(\boldsymbol{u}^{(i-1)})\|_2} + h_\perp \frac{\boldsymbol{T}^{(i)} - \boldsymbol{J}(\boldsymbol{u}^{(i)})}{\|\boldsymbol{T}^{(i)} - \boldsymbol{J}(\boldsymbol{u}^{(i)})\|_2}$
16: $\boldsymbol{u}^{(p,i+1)} = \boldsymbol{u}^{(i)} + h_p \left(\boldsymbol{u}^{(i)} - \boldsymbol{u}^{(i-1)}\right)$ (Predictor step)
17: **end if**
18: **end loop**
19: **end for**

Continuation Methods

It has already been discussed in Section 2.1.3 that path following methods can be applied to approximate the set of substationary points if the Pareto set is a manifold. The idea is to compute a curve $\boldsymbol{c}(\boldsymbol{\phi}) = \boldsymbol{H}^{-1}(\boldsymbol{0})$ where $\boldsymbol{\phi} \in \mathbb{R}^{k-1}$ is a parametrization of $\boldsymbol{c}$ and $\boldsymbol{H}$ is defined according to (2.10). In this situation, *predictor corrector (PC)* approaches are frequently utilized. When differentiating $\boldsymbol{H}(\boldsymbol{c}(\boldsymbol{\phi})) = \boldsymbol{0}$ with respect to $\boldsymbol{\phi}$, we obtain

$$\boldsymbol{H}'(\boldsymbol{c}(\boldsymbol{\phi})) \cdot \boldsymbol{c}'(\boldsymbol{\phi}) = \boldsymbol{0}.$$

Consequently, the Pareto set can be linearized by computing kernel vectors of $\boldsymbol{H}'(\boldsymbol{c}(\boldsymbol{\phi}))$. This is achieved by performing a QR factorization [GvL13] of $\boldsymbol{H}'^\top$:

$$\boldsymbol{H}'(\boldsymbol{c}(\boldsymbol{\phi}))^\top = \boldsymbol{Q}\boldsymbol{R} = (\boldsymbol{q}_1, \ldots, \boldsymbol{q}_{n+m+k})\boldsymbol{R},$$

where the last $k-1$ entries of $\boldsymbol{Q}$ form an orthonormal basis of the tangent space of the Pareto set (Figure 2.8 (a)). After performing a predictor step with a suitable step length along the tangent space (Figure 2.8 (b)), a new solution is computed in the corrector step which again satisfies the KKT conditions (Figure 2.8 (c)). This is achieved by applying Newton's method, for example. The PC steps are then repeated in order to compute the next Pareto optimal point (cf. Figure 2.8 (d)).

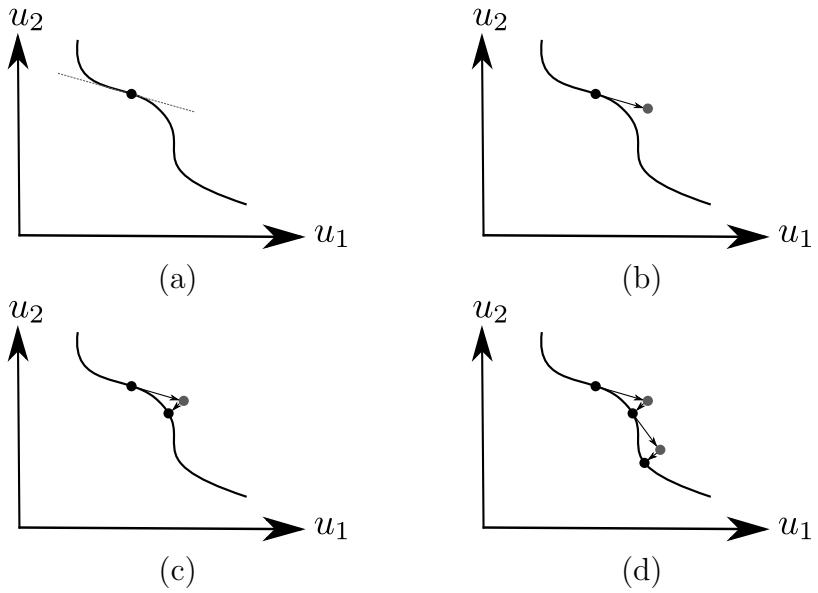


Figure 2.8: Continuation. (a) Local linearization of $\mathcal{P}_{S,\mathsf{sub}}$. (b) Predictor step along the linearization. (c) Corrector step to obtain the next Pareto optimal point. (d) Computation of the consecutive point.

The efficiency of the PC method can be increased by introducing adaptivity [AG03]. If a large number of iterations is required in the corrector step for example, this means that the predictor step presumably resulted in a point far away from the Pareto set, indicating a strong curvature. Consequently, the step length of the predictor step can be reduced for the computation of the next point.

If the set of substationary points $\mathcal{P}_{S,\mathsf{sub}}$ is high-dimensional, we observe problems with respect to the parametrization similar to scalarization techniques. It is then difficult to choose directions for the predictor steps in order to obtain a well-spread covering of $\mathcal{P}_{S,\mathsf{sub}}$. In this situation, interactive methods can be used in which a decision maker interactively determines which Pareto optimal points to compute next [CLS16, MS17]. Furthermore, continuation methods are restricted to situations where $\mathcal{P}_{S,\mathsf{sub}}$ is connected. Otherwise, it can only be computed in part.

Evolutionary Algorithms

Multiobjective Evolutionary Algorithms (*MOEAs*, see e.g. [CLV07] for an overview) have successfully been applied to a large variety problems. Although evolutionary algorithms will not be covered in this thesis, the basic principles are briefly explained here to acknowledge their relevance in multiobjective optimization. The basic idea is adopted from biological evolution, following the concept of *survival of the fittest*. The algorithms consist the stages reproduction, mutation, recombination, and selection. The general concept is as follows:

(1) Start with an initial population,

(2) Repeat until termination:

   (a) Create new individuals via *reproduction* and *recombination* from the previous population,

   (b) Evolve individuals via *mutation*,

   (c) Evaluate the *fitness function* (such as the objectives of an MOP or non-dominance properties) and *select* the fittest individuals.

Most MOEAs do not utilize gradients which makes them easy to use and also applicable to black box problems. However, this often has the consequence that a very large number of function evaluations is required and convergence is slow. Several authors have therefore proposed to combine MOEAs with gradient information [BdJ05, Bos12, HSK06]. These algorithms are also known as *memetic algorithms* [NCM12], where general MOEA concepts are combined with local search strategies known from gradient-based optimization.

2.1.5 The Subdivision Algorithm

Set-oriented methods [DSH05, SDD05, SWOBD13] are an alternative deterministic approach for the solution of MOPs. Instead of approximating the Pareto set by a finite number of points, a superset of the Pareto set is computed. This superset is composed of a finite number of subsets, represented by n-dimensional *boxes*, where n is the dimension of the decision space. As the diameter of the boxes tends to zero, this superset converges to the set of substationary points. Many of the results in the following chapters make use of these set-oriented methods and for this reason, the subdivision algorithm is presented in detail in this section, mainly following [DSH05].

The general concept of the algorithms presented here has been developed for the computation of attractors of dynamical systems [DH97] and was later adapted for the computation of zeros of functions in [DSS02] and the computation of global Pareto

sets in [DSH05]. At the end of the section a derivative-free alternative (the *sampling algorithm*) based on non-dominance testing is presented as well as a *recovering algorithm*, where the neighborhood of the current box collection is explored and additional boxes covering parts of the Pareto set are added. The main algorithm, the *subdivision algorithm*, is currently restricted to unconstrained MOPs such that we will use it in order to approximate the set of points satisfying the condition (KKTu) for unconstrained MOPs.

Consider a finite collection of dynamical systems of the type

$$\boldsymbol{u}^{(j+1)} = \boldsymbol{f}_l(\boldsymbol{u}^{(j)}), \quad l = 0, 1, \ldots, r, \tag{2.18}$$

where, for simplicity, each $\boldsymbol{f}_l : \mathbb{R}^n \to \mathbb{R}^n$ is assumed to be a diffeomorphism. In the context of zero finding [DSS02], the systems $\boldsymbol{f}_l$ correspond to damped Newton methods with different damping constants or step sizes. Denoting by $\Theta = \{1, 2, \ldots, r\}^{\mathbb{N}_0}$ the set of all sequences of the symbols $\{1, 2, \ldots, r\}$, we define the function $\boldsymbol{f}_{\boldsymbol{\vartheta}^j} = \boldsymbol{f}_{\vartheta_{j-1}} \circ \ldots \circ \boldsymbol{f}_{\vartheta_0}$ for $j \geq 1$ with $\boldsymbol{\vartheta}^j = (\vartheta_0, \vartheta_1, \ldots, \vartheta_{j-1})$ and $\vartheta_i \in \Theta$. Then the attracting set for the dynamical systems (2.18) is defined as follows:

Definition 2.1.19. *Let $\boldsymbol{f}_1, \ldots, \boldsymbol{f}_r : \mathbb{R}^n \to \mathbb{R}^n$ be diffeomorphisms and let $Q \subset \mathbb{R}^n$ be compact. The relative global attractor of $\boldsymbol{f}_1, \ldots, \boldsymbol{f}_r$ with respect to Q is defined as*

$$A_{Q,\boldsymbol{f}_1,\ldots,\boldsymbol{f}_r} = \bigcap_{\vartheta \in \Theta} \bigcap_{j \geq 1} \boldsymbol{f}_{\boldsymbol{\vartheta}^r}(Q) \cap Q.$$

Using a multilevel subdivision scheme, Algorithm 2.2 computes an outer approximation of the relative global attractor $A_{Q,\boldsymbol{f}_1,\ldots,\boldsymbol{f}_r}$ in the form of a sequence of sets $\mathcal{B}^{(0)}, \mathcal{B}^{(1)}, \ldots$, where each $\mathcal{B}^{(s)}$ is a subset of $\mathcal{B}^{(s-1)}$ and consists of finitely many subsets B of Q covering $A_{Q,\boldsymbol{f}_1,\ldots,\boldsymbol{f}_s}$. Within Algorithm 2.2, the box diameter

$$\operatorname{diam}(\mathcal{B}^{(s)}) = \max_{B \in \mathcal{B}^{(s)}} \operatorname{diam}(B)$$

tends to zero for $s \to \infty$ such that convergence to A_Q is obtained.

Theorem 2.1.20 ([DSS02])**.** *Let A_Q be the global attractor of $\boldsymbol{f}_1, \ldots, \boldsymbol{f}_r$ relative to the closed set Q. Let $\mathcal{B}^{(s)}$, $s = 0, 1, \ldots$, be a sequence of collections created by Algorithm 2.2 and denote by $Q^{(s)} = \bigcup_{B \in \mathcal{B}^{(s)}} B$ the corresponding sequence of compact subsets of $Q = Q^{(0)}$. Then*

$$\lim_{s \to \infty} d_h\left(Q^{(s)}, A_Q\right) = 0,$$

where d_h denotes the Hausdorff distance [RW98].

Algorithm 2.2 (Subdivision algorithm)

Let $\mathcal{B}^{(0)}$ be an initial collection of finitely many subsets of the compact set Q such that $\bigcup_{B\in\mathcal{B}^{(0)}} B = Q$. Then, for $s \geq 1$, $\mathcal{B}^{(s)}$ is inductively obtained from $\mathcal{B}^{(s-1)}$ in two steps:

(i) **Subdivision**. Construct from $\mathcal{B}^{(s-1)}$ a new collection of subsets $\hat{\mathcal{B}}^{(s)}$ such that

$$\bigcup_{B\in\hat{\mathcal{B}}^{(s)}} B = \bigcup_{B\in\mathcal{B}^{(s-1)}} B,$$
$$\operatorname{diam}(\hat{\mathcal{B}}^{(s)}) = \theta^{(s)}\operatorname{diam}(\mathcal{B}^{(s-1)}), \quad 0 < \theta_{\mathsf{min}} \leq \theta^{(s)} \leq \theta_{\mathsf{max}} < 1.$$

(ii) **Selection**. Define the new collection $\mathcal{B}^{(s)}$ by

$$\mathcal{B}^{(s)} = \left\{ B \in \hat{\mathcal{B}}^{(s)} \;\middle|\; \exists \hat{B} \in \hat{\mathcal{B}}^{(s)} \text{ such that } \boldsymbol{f}_{\vartheta^r}^{-1}(B) \cap \hat{B} \neq \emptyset \right\}.$$

If we now consider a dynamical system of the type (2.18) using the descent direction $\boldsymbol{q}(\boldsymbol{u})$ (given by Equation (2.3)) obtained by solving the quadratic problem (QOP), this leads to

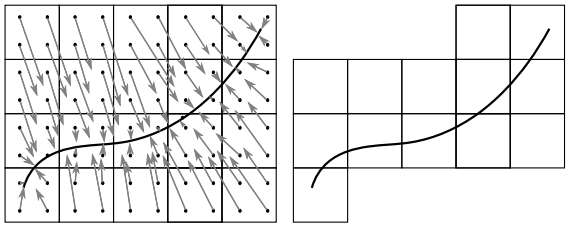
$$\boldsymbol{u}^{(j+1)} = \boldsymbol{u}^{(j)} + h^{(j)}\boldsymbol{q}^{(j)}(\boldsymbol{u}^{(j)}), \tag{2.19}$$

where $h^{(j)} \in \mathbb{R}$ is an appropriately chosen step length (e.g. such that it satisfies the Armijo rule [NW06] for all objectives). Note that this is equivalent to Equation (2.5) (with $\boldsymbol{p}^{(j)} = \boldsymbol{q}^{(j)}$) such that according to Theorem 2.1.13, each accumulation point of (2.19) satisfies (KKTu). From a dynamical systems point of view, the attractor A_Q of the system (2.19) is the set of substationary points if $\mathcal{P}_{S,\mathsf{sub}}$ is bounded and connected (cf. Remark 2.1.14). Otherwise, $\mathcal{P}_{S,\mathsf{sub}} \subset A_Q$. As a consequence, Algorithm 2.2 yields an outer approximation of this set:

Theorem 2.1.21 ([DSH05])**.** *Suppose that the set $\mathcal{P}_{S,\mathsf{sub}}$ of points $\boldsymbol{u}^* \in \mathbb{R}^n$ satisfying (KKTu) is bounded and connected. Let Q be a compact neighborhood of $\mathcal{P}_{S,\mathsf{sub}}$. Then the application of Algorithm 2.2 to Q with respect to the iteration scheme (2.19) leads to a sequence of coverings $\mathcal{B}^{(s)}$ which converges to the entire set $\mathcal{P}_{S,\mathsf{sub}}$, that is,*

$$d_h(\mathcal{P}_{S,\mathsf{sub}}, \mathcal{B}^{(s)}) \to 0, \quad \text{for } s = 0, 1, 2, \ldots.$$

We now take a brief look at the numerical realization of the subdivision algorithm. The elements $B \in \mathcal{B}^{(s)}$ are n-dimensional boxes. In the selection step, each box is represented by a prescribed number of sample points at which the dynamical system (2.19) is evaluated (cf. Figure 2.9 (a)). This involves determining the descent



(a) (b)

Figure 2.9: Global subdivision algorithm – selection step. (a) Evaluation of the dynamical system (2.19). (b) All boxes that do not possess a preimage within the collection are discarded.

direction by solving (QOP) and an appropriate step size $h \in \mathbb{R}$, e.g. via backtracking [NW06]. We then evaluate which boxes the sample points are mapped into and eliminate all "empty" boxes, i.e. boxes which do not possess a preimage within $\mathcal{B}^{(s)}$ (Figure 2.9 (b)). The remaining boxes are subdivided, followed by the next elimination step until a certain stopping criterion is met (e.g. a prescribed number of subdivision steps).

Sampling Algorithm

In many applications, gradients are unknown or difficult to compute. In this case, a gradient-free alternative of Algorithm 2.2 can be constructed using the concept of non-dominance (cf. Definition 2.1.5 and Remark 2.1.6). Algorithm 2.3 (denoted as the *sampling algorithm*) also consists of a subdivision and a selection step with the difference that the selection step is a set-valued non-dominance test. Sample points are inserted and the objectives are evaluated at each of the points. Then all boxes are discarded that contain only dominated points. Hence, we directly compute a superset of the global Pareto set $\mathcal{P}_S$, also for constrained MOPs. In the presence of inequality constraints, for example, we eliminate the boxes for which all sample points violate the constraints and then perform the non-dominance test for the remaining boxes. Equality constraints are simply modeled by introducing two inequality constraints. Finally, a combination of both the gradient-based and the gradient-free algorithm can be applied in order to accelerate convergence or to reduce the gradient-based algorithm to the computation of the Pareto set $\mathcal{P}_S$ instead of the set of substationary points $\mathcal{P}_{S,\mathsf{sub}}$.

Recovering Algorithm

In order to fill gaps in the box covering that may occur due to insufficient sampling, a recovering algorithm can be utilized. Very similar to the subdivision al-

Algorithm 2.3 (Sampling algorithm)

Let $\mathcal{B}^{(0)}$ be an initial collection of finitely many subsets of the compact set Q such that $\bigcup_{B\in\mathcal{B}^{(0)}} B = Q$. Then, for $s \geq 1$, $\mathcal{B}^{(s)}$ is inductively obtained from $\mathcal{B}^{(s-1)}$ in two steps:

(i) **Subdivision**. Construct from $\mathcal{B}^{(s-1)}$ a new collection of subsets $\hat{\mathcal{B}}^{(s)}$ such that

$$\bigcup_{B\in\hat{\mathcal{B}}^{(s)}} B = \bigcup_{B\in\mathcal{B}^{(s-1)}} B,$$
$$\operatorname{diam}(\hat{\mathcal{B}}^{(s)}) = \theta^{(s)}\operatorname{diam}(\mathcal{B}^{(s-1)}), \quad 0 < \theta_{\mathsf{min}} \leq \theta^{(s)} \leq \theta_{\mathsf{max}} < 1.$$

(ii) **Selection**. Define the new collection $\mathcal{B}^{(s)}$ by

$$\mathcal{B}^{(s)} = \left\{ B \in \hat{\mathcal{B}}^{(s)} \;\middle|\; \nexists \hat{B} \in \hat{\mathcal{B}}^{(s)} \text{ such that } \hat{B} \text{ dominates } B \right\},$$

where set-valued dominance is understood according to Remark 2.1.6.

gorithm, there exist a gradient-based and a gradient-free version of this procedure. The gradient-based version utilizes the continuation procedure described previously, i.e. the Pareto set is locally linearized. Then sampling points are inserted in the resulting hyperplane and corrected using a Newton step. All boxes that are hit by these points are added to the box collection such that the set of substationary points can be locally recovered. A detailed description of the procedure can be found in [SDD05]. The gradient-free alternative uses non-dominance testing, see [Sch04] for details. Here, all boxes that are immediate neighbors to the current box collection are inserted and the objectives are evaluated for a set of sample points. Then all dominated boxes are discarded. The recovering steps are repeated until no further boxes are added to the collection.

Note that both the gradient-based and the gradient-free version of the recovering algorithm are only capable of locally exploring the Pareto set. Hence, the algorithm is not global and also, one can no longer compute Pareto sets which are disconnected. However, the procedure can be used in combination with the subdivision or the sampling algorithm in order to increase the numerical efficiency. Moreover, if the numerical effort is too great for subdivision to be applicable, recovering can be used to locally compute Pareto sets.

2.2 Optimal Control and Model Predictive Control

The purpose of optimal control is to steer a dynamic process in such a way that some cost functional J is minimized. This implies that – in contrast to optimization – we have to compute an optimal function instead of a finite-dimensional parameter. In practice, this often is a control input over time such as the engine torque of a vehicle or the amount of heat introduced into a system. Moreover, the dynamics of the system have to be taken into account which are often described by ordinary or partial differential equations. One can easily formulate optimal control problems with multiple objectives by simply introducing a vector-valued cost functional $\boldsymbol{J}$ which results in a *multiobjective optimal control problem (MOCP)*. Depending on the solution approach, these problems can be solved in a similar fashion as MOPs.

The purpose of this section is to give an introduction to optimal control and various extensions that will be utilized throughout this thesis. However, since the main subject is multiobjective optimization, the results are presented only briefly. Instead, references to more detailed introductions are given in the various sections. First, the general (multiobjective) optimal control problem will be introduced in Section 2.2.1, including a short overview of well-known solution approaches.

In the context of real-time applicability, it is often necessary to apply *closed-loop* control instead of *open-loop* optimal control strategies in order to take plant-model mismatches or unforeseen events into account. To this end, optimal control approaches can be utilized within closed-loop control via *model predictive control (MPC)*. This approach will be introduced in Section 2.2.2. A related concept, *motion planning with motion primitives*, will be covered briefly in Section 2.2.3.

2.2.1 Optimal Control

The introduction to optimal control mainly follows [BBB+01] and [OB08], an even more detailed introduction can be found in [Lib12]. As stated in the introduction, the aim of optimal control is to optimally steer a process with respect to some objective while taking the system dynamics into account. The problems covered in this thesis are either described by ordinary differential equations (ODEs) or by partial differential equations (PDEs). There exist dynamical systems which have to be described differently, e.g. as discrete dynamical systems or via differential algebraic equations. However, these will not be covered here. In the case of ODEs, the dynamics on the time interval $[t_0, t_e]$ can be described by

$$\begin{aligned} \dot{\boldsymbol{y}}(t) &= \boldsymbol{F}(\boldsymbol{y}(t), \boldsymbol{u}(t)), \qquad t \in (t_0, t_e], \\ \boldsymbol{y}(t_0) &= \boldsymbol{y}_0, \end{aligned} \tag{ODE}$$

where $\boldsymbol{u} \in L^2((t_0, t_e); \mathbb{R}^{n_u})$ is the control function, $\boldsymbol{y} \in H^1((t_0, t_e); \mathbb{R}^{n_y})$ is the system state, $\dot{\boldsymbol{y}}$ is the time derivative $d\boldsymbol{y}/dt$ and $\boldsymbol{F} : \mathbb{R}^{n_y} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_y}$ is continuously differentiable with respect to $\boldsymbol{y}$ and $\boldsymbol{u}$. The Hilbert space L^2 is equipped with the inner product $(\boldsymbol{y}, \boldsymbol{z})_{L^2} = \int_{t_0}^{t_e} \boldsymbol{y}(t) \cdot \boldsymbol{z}(t)\, \mathrm{d}t$ and the norm $\|\boldsymbol{y}\|_{L^2} = \left(\int_{t_0}^{t_e} \boldsymbol{y}(t) \cdot \boldsymbol{y}(t)\, \mathrm{d}t \right)^{1/2}$ and H^1 is the standard Sobolev space $W^{1,2}$ (cf. [HPUU09]).

When considering PDEs, the state $\boldsymbol{y} \in L^2((t_0, t_e); H^1(\Omega)) \cap H^1((t_0, t_e); L^2(\Omega))$, additionally depends on the location $\boldsymbol{x}$. The domain of interest $\Omega \subset \mathbb{R}^{n_x}$ is a connected open set with the spatial dimension n_x and the boundary is denoted by $\Gamma = \partial\Omega$. By this, we have

$$\begin{aligned} \dot{\boldsymbol{y}}(\boldsymbol{x}, t) &= \boldsymbol{G}(\boldsymbol{y}(\boldsymbol{x}, t), \boldsymbol{u}(t)), && (\boldsymbol{x}, t) \in \Omega \times (t_0, t_e], \\ a(\boldsymbol{x}, t) \frac{\partial \boldsymbol{y}}{\partial \boldsymbol{n}}(\boldsymbol{x}, t) + b(\boldsymbol{x}, t) \boldsymbol{y}(\boldsymbol{x}, t) &= \boldsymbol{c}(\boldsymbol{x}, t), && (\boldsymbol{x}, t) \in \Gamma \times (t_0, t_e], \\ \boldsymbol{y}(\boldsymbol{x}, t_0) &= \boldsymbol{y}_0(\boldsymbol{x}), && \boldsymbol{x} \in \Omega, \end{aligned} \tag{PDE}$$

where $\boldsymbol{u} \in L^2((t_0, t_e); \mathbb{R}^{n_u})$ is again the control and $\boldsymbol{n}$ is the outward normal vector of the boundary Γ. The operator $\boldsymbol{G}$ is a partial differential operator describing the evolution of the system. For details, the reader is referred to [HPUU09, Trö10]. Since the state is space dependent, we additionally have to take boundary conditions (BCs) into account which are formulated as Robin type BCs. The coefficients $a(\boldsymbol{x}, t)$, $b(\boldsymbol{x}, t)$ and $\boldsymbol{c}(\boldsymbol{x}, t)$ are given by the problem definition. Note that these include both Dirichlet as well as Neumann BCs by neglecting one of the terms on the left hand side, respectively.

Remark 2.2.1. *By introducing a semi-discretization in space for the PDE (e.g. using a finite element discretization with n_y grid points), problem (PDE) is transformed into problem (ODE) with n_y degrees of freedom. Consequently, the following formulations are restricted to the ODE case.*

For the case of space independent states, the cost functional $\hat{\boldsymbol{J}} : \mathbb{R}^{n_y} \times \mathbb{R}^{n_u} \to \mathbb{R}^k$ with k objectives is of the general form

$$\hat{\boldsymbol{J}}(\boldsymbol{y}, \boldsymbol{u}) = \begin{pmatrix} \hat{J}_1(\boldsymbol{y}, \boldsymbol{u}) \\ \vdots \\ \hat{J}_k(\boldsymbol{y}, \boldsymbol{u}) \end{pmatrix}, \tag{2.20}$$

where

$$\hat{J}_i(\boldsymbol{y}, \boldsymbol{u}) = \int_{t_0}^{t_e} C_i(\boldsymbol{y}(t), \boldsymbol{u}(t))\, \mathrm{d}t + \Phi_i(t_e), \quad i = 1, \ldots, k. \tag{2.21}$$

The term $\Phi_i : \mathbb{R} \to \mathbb{R}$ represents the terminal cost and is called the *Mayer term* whereas the integral term (with $C_i : \mathbb{R}^{n_y} \times \mathbb{R}^{n_u} \to \mathbb{R}$) is the *Lagrange term*. Problems

where the cost functional contains both a Mayer and a Lagrange term are of *Bolza* form. A problem of Lagrange type can be transformed into a problem of Mayer type and vice versa [Ger12].

In many cases, there exists a unique solution $\boldsymbol{y}(t)$ or $\boldsymbol{y}(\boldsymbol{x}, t)$ for every $\boldsymbol{u}(t)$, i.e. there exists a solution operator $\mathcal{S} : \mathcal{U} \to \mathcal{Y}$ (see e.g. [Trö10]) and hence, we obtain a reduced cost functional $\boldsymbol{J}(\boldsymbol{u}) := \hat{\boldsymbol{J}}(\mathcal{S}\boldsymbol{u}, \boldsymbol{u})$. The existence of such a reduced functional will be assumed throughout the remainder of the thesis.

Based on the above formulations, we introduce the multiobjective optimal control problem for ODEs:

$$\begin{aligned} & \min_{\boldsymbol{u}} \boldsymbol{J}(\boldsymbol{u}) \\ \text{s.t.} \quad & (\text{ODE}), \\ & \boldsymbol{g}(\boldsymbol{u}(t)) \leq_p \boldsymbol{0}, \\ & \boldsymbol{h}(\boldsymbol{u}(t)) = \boldsymbol{0}, \\ & \boldsymbol{r}(\boldsymbol{y}(t_e)) = \boldsymbol{0}. \end{aligned} \tag{2.22}$$

Here, $\boldsymbol{g} : \mathbb{R}^{n_u} \to \mathbb{R}^l$ and $\boldsymbol{h} : \mathbb{R}^{n_u} \to \mathbb{R}^m$ are (inequality and equality) *path constraints* and $\boldsymbol{r} : \mathbb{R}^{n_y} \to \mathbb{R}^{m_f}$ are *final constraints*. Analog to problem (MOP), the constraints can be incorporated in the formulation of feasible sets $\mathcal{Y}$ and $\mathcal{U}$ for the state $\boldsymbol{y}$ and the control $\boldsymbol{u}$, respectively.

The problem with PDE constraints can be formulated analogously. In this case, the cost functionals are of the form

$$\hat{J}_i(\boldsymbol{y}, \boldsymbol{u}) = \int_\Omega \left(\int_{t_0}^{t_e} C_i(\boldsymbol{y}(\boldsymbol{x}, t), \boldsymbol{u}(t)) \, \mathrm{d}t + \Phi_i(\boldsymbol{x}, t_e) \right) \mathrm{d}\boldsymbol{x}, \quad i = 1, \ldots, k. \tag{2.23}$$

Necessary Optimality Conditions for Scalar Problems

The considerations in the following sections will be restricted to ODEs, assuming that (according to Remark 2.2.1) semi-discretized PDEs can also be addressed this way. In the single objective case, the corresponding necessary conditions for optimality go back to Lev Pontryagin and his well-known *Maximum Principle* [Lib12]. It states that an optimal triple $(\boldsymbol{y}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*)$, where $\boldsymbol{\lambda} \in H^1((t_0, t_e); \mathbb{R}^{n_y})$ is the so-called *adjoint* state (or co-state), maximizes the *control Hamiltonian* $\mathcal{H} : \mathbb{R}^{n_y} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_y} \to \mathbb{R}$, where

$$\mathcal{H}(\boldsymbol{y}(t), \boldsymbol{u}(t), \boldsymbol{\lambda}(t)) = -C(\boldsymbol{y}(t), \boldsymbol{u}(t)) + \boldsymbol{\lambda}^\top \boldsymbol{F}(\boldsymbol{y}(t), \boldsymbol{u}(t)),$$

with $C : \mathbb{R}^{n_y} \times \mathbb{R}^{n_u} \to \mathbb{R}$.

A more intuitive condition for optimality can be derived using the Lagrange formalism which is well known form constrained scalar optimization. As stated in

[IK08], the Lagrange formalism coincides with the Pontryagin Maximum Principle. The Lagrangian $\mathcal{L} : \mathbb{R}^{n_y} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_y} \to \mathbb{R}$ is defined as

$$\mathcal{L}(\boldsymbol{y}, \boldsymbol{u}, \boldsymbol{\lambda}) = \int_{t_0}^{t_e} \left(C(\boldsymbol{y}(t), \boldsymbol{u}(t)) + \boldsymbol{\lambda}^\top \left(\dot{\boldsymbol{y}}(t) - \boldsymbol{F}(\boldsymbol{y}(t), \boldsymbol{u}(t)) \right) \right) \mathrm{d}t + \Phi(t_e). \tag{2.24}$$

For an optimal triple $(\boldsymbol{y}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*)$ the Lagrangian becomes stationary:

$$\delta\mathcal{L}(\boldsymbol{y}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*) = 0,$$

where $\delta\mathcal{L}(\boldsymbol{y}, \boldsymbol{u}, \boldsymbol{\lambda})$ is the variation of $\mathcal{L}$ with respect to $\boldsymbol{y}$, $\boldsymbol{u}$ and $\boldsymbol{\lambda}$, respectively:

$$\delta\mathcal{L}(\boldsymbol{y}, \boldsymbol{u}, \boldsymbol{\lambda}) = \frac{\partial\mathcal{L}}{\partial\boldsymbol{y}}\delta\boldsymbol{y} + \frac{\partial\mathcal{L}}{\partial\boldsymbol{u}}\delta\boldsymbol{u} + \frac{\partial\mathcal{L}}{\partial\boldsymbol{\lambda}}\delta\boldsymbol{\lambda}.$$

The condition $\delta\mathcal{L}(\boldsymbol{y}, \boldsymbol{u}, \boldsymbol{\lambda}) = 0$ generally results in an optimality system (provided a constraint qualification condition holds [HPUU09]) which consists of the state equation, the adjoint equation, and the optimality condition. The latter is zero if the triple $(\boldsymbol{y}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*)$ satisfies the first order conditions and otherwise yields a descent direction for the objective which can be utilized in iterative approaches.

Solution Methods for Scalar Optimal Control Problems

Over the past 50 years, many researchers have contributed to the development of solution methods for scalar optimal control problems. These methods can be divided into two main categories, indirect and direct approaches, also referred to as *optimize-then-discretize* and *discretize-then-optimize*, respectively. For a more detailed introduction, the reader is referred to [BBB+01] and the references therein.

For the first approach, the optimality conditions presented above are formulated as a boundary value problem which is then discretized in order to solve it numerically. The formulation of the system is often challenging such that mathematical insight and knowledge about optimal control theory are important. Due to the demands on memory and CPU time, the optimality system is often solved iteratively: the state equation is solved forwards in time, the adjoint equation backwards in time, and the optimality condition then provides gradient information. Alternatively, *multiple shooting* or *collocation* methods can be used to indirectly solve the optimal control problem.

In the second approach, the problem is first discretized, i.e. a time and spatial grid are introduced. As a result, the optimal control problem is transformed into a high-dimensional optimization problem and methods from non-linear optimization can be applied. However, it is no longer possible to exploit the structure of the underlying dynamical system in order to derive methods tailored to a specific problem.

Direct methods can further be divided into *direct single* and *direct multiple shooting* as well as *Galerkin type* and *direct collocation methods*. For large scale applications, these approaches have proven to be more successful than indirect methods [BBB+01].

Parameter Dependent Optimal Control Problems

In many situations, the dynamical system additionally depends on a parameter $\boldsymbol{p} \in \mathbb{R}^{n_p}$. This parameter can for example have an influence on the initial condition or on the system dynamics in general. For the ODE case, the parameter dependent version is

$$\begin{aligned} \dot{\boldsymbol{y}}(t) &= \boldsymbol{F}(\boldsymbol{y}(t), \boldsymbol{u}(t), \boldsymbol{p}), \qquad t \in (t_0, t_e], \\ \boldsymbol{y}(t_0) &= \boldsymbol{y}_0(\boldsymbol{p}). \end{aligned} \tag{ODEp}$$

Consequently, the corresponding cost functional also depends on $\boldsymbol{p}$. If we are interested in solving optimal control problems for various parameter values, it would be advantageous if the problem exhibited additional structure. In fact, this has been addressed by several researchers for single objective optimal control, see e.g. [MP94, MP95, MM96, BM00]. Parameter dependent MOCPs will be investigated in Chapter 3.

One can show that the solution $\boldsymbol{y}$ of the dynamical system (ODEp) is Lipschitz continuous with respect to $\boldsymbol{p}$ if the vector field as well as the initial condition are also Lipschitz continuous:

Theorem 2.2.2 ([Ger12]). *Assume that for the dynamical system* (ODEp) *the Lipschitz condition*

$$\|\boldsymbol{F}(\boldsymbol{y}_1(t), \boldsymbol{u}(t), \boldsymbol{p}_1) - \boldsymbol{F}(\boldsymbol{y}_2(t), \boldsymbol{u}(t), \boldsymbol{p}_2)\| \leq L_1(\|\boldsymbol{y}_1(t) - \boldsymbol{y}_2(t)\| + \|\boldsymbol{p}_1 - \boldsymbol{p}_2\|)$$

holds with a Lipschitz constant L_1 and the initial condition $\boldsymbol{y}_0(\boldsymbol{p})$ is Lipschitz continuous with a Lipschitz constant L_2. Then the solution $\boldsymbol{y}(t)$ is also Lipschitz continuous with respect to $\boldsymbol{p}$.

Consequently, if the objective functional depends Lipschitz continuously on $\boldsymbol{y}$, the continuity with respect to $\boldsymbol{p}$ is carried over to $\boldsymbol{J}$.

Differentiability of the solution with respect to $\boldsymbol{p}$ can be shown if additional smoothness assumptions are met. This has been investigated in a series of papers for scalar as well as vector-valued parameters, also considering control-state constraints [MP94, MP95, MM96]. In short, the additional assumptions are that the objective functional J and the vector field $\boldsymbol{F}$ have to be at least C^2, the second derivative of the control Hamiltonian $\frac{\partial^2 \mathcal{H}}{\partial u^2}$ has to be positive definite and the Riccati equation of the linearized problem has to possess a C^1 solution. Together, these are denoted

as the second-order sufficient conditions (SSC). For details, the reader is referred to [MP94].

Multiobjective Optimal Control

When considering multiple objectives in an optimal control problem, solution methods can again be divided into indirect and direct approaches. To the best of the author's knowledge, indirect approaches have until now not been investigated or applied when directly addressing the vector-valued objective functional. Instead, by using a scalarization technique such as the weighted sum or the reference point method (cf. Section 2.1.4) the theory for scalar optimal control can be applied. When applying a direct approach on the other hand, the MOCP is transferred into a (potentially high-dimensional) MOP paving the way for techniques based on the vector of objectives $\boldsymbol{J}$, i.e. set-oriented, continuation or evolutionary algorithms. Algorithms for the solution of MOCPs have been proposed in e.g. [LHDvI10, OBRzF12, SWOBD13, DEF+16] for ODE constraints. In [LKBM05] as well as [ARFL09], non-linear PDE constraints are taken into account but the model is treated as a black box, i.e. no special treatment of the constraints is required. The first articles explicitly taking PDE constraints into account are [IUV13, ITV16], where MOCPs are solved with a weighted sum approach and model order reduction techniques subject to linear and semi-linear PDE constraints, respectively. In [BBV16, Ban16, BBV17], scalarization is realized via the reference point method. In Chapter 5, both direct and indirect approaches to MOCPs will be coupled with reduced order modeling in order to solve PDE-constrained MOCPs.

2.2.2 Model Predictive Control

Control theory has significantly been influenced by the advances in computational capacities during the last decades. For many systems, it is nowadays possible to use model-based optimal control algorithms to design sophisticated feedback laws. This concept is known as *model predictive control (MPC)* (see e.g. [GP17] for an extensive introduction). The general goal of MPC is to stabilize a system by using a combination of open and closed-loop control. Using a model of the system dynamics, an open-loop optimal control problem is solved in real-time over a so-called *prediction horizon*. The first part of this solution is then applied to the real system (plant) while the optimization is repeated to find a new control function on a prediction horizon which is slightly moved forward in time. For this reason, MPC is also referred to as *moving horizon control* or *receding horizon control*. As the optimal control problem (OCP) has to be solved online, a fixed upper bound for the computing time for solving the problem must not be violated. Due to this, MPC was initially

introduced in the process industry where the system dynamics are comparatively slow. However, advances in computer technology as well as algorithms have enabled the use of MPC in a large variety of applications [QB97].

In order to construct a feedback controller, the optimal control problem (2.22) is repeatedly solved online for varying time frames $[t_s, t_{s+p}]$ with $t_s = s \cdot t_h$ and $t_{s+p} = (s+p) \cdot t_h$, $s = 0, 1, 2, \ldots$. Here, several time horizons are introduced. The *sample time* t_h, over which the control input is constant, is the smallest time instance considered in the MPC framework. The optimal control problem is solved over a horizon of length t_p, which is the *prediction horizon*. In many situations, a third horizon, the so-called *control horizon* $t_c \leq t_p$ is defined on which the control can be varied within the optimal control problem. By choosing $t_c < t_p$, the complexity of the control problem can be reduced which is often critical for real-time applicability. If this is not an issue, one can set $t_c = t_p$. After having solved the first OCP, the first entry of the optimal control, $\boldsymbol{u}^*(t_s)$, is applied to the plant and the optimal control problem is solved again with a time frame shifted by t_h. The procedure is illustrated in Figure 2.10.

The concept of MPC was initially developed to stabilize a system [GP17], i.e. to drive the system state to a (potentially time dependent) reference state. However, stabilization is not always the main concern, e.g. when the system is already stable or when stability can be achieved by many different control inputs. In this situation, we can pursue additional objectives such as minimizing the energy consumption, which is known as *economic MPC* (see e.g. [RA09, DAR11, GP17]).

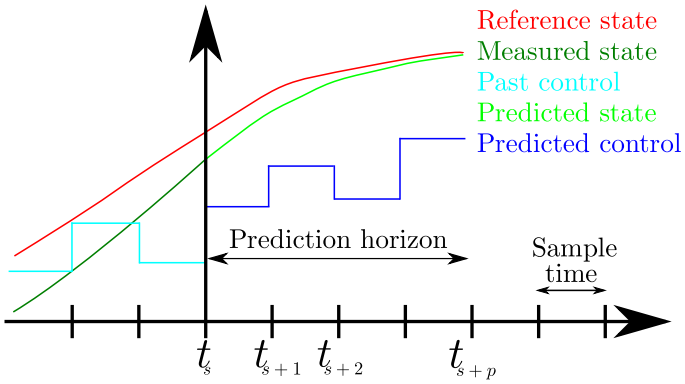


Figure 2.10: Sketch of the MPC methodology. While the first part of the predicted control is applied to the system, the next control is computed on a shifted horizon.

Due to the remarkable success of MPC, a large variety of algorithms has been established, where a first distinction can be made between linear and non-linear MPC. The first category refers to schemes in which linear models and quadratic objective

functions are used to predict the system dynamics. The resulting optimization problems are convex, i.e. global solutions can be computed very fast. Linear MPC approaches have been very successful in many different industrial applications (see e.g. [QB97] and [LC97] for overviews of applications and theory). The advantage of non-linear MPC [GP17] is that the typically non-linear system behavior can be approximated more accurately. Furthermore, more complicated optimality criteria and non-linear constraints can easily be incorporated in the problem formulation. However, the complexity and thereby the time to solve the resulting optimization problem increases such that it is often difficult to preserve real-time capability (see e.g. [EPS+16]).

Motivated by this, another extension is *explicit MPC* (see [AB09] for a survey), where the problem of real-time applicability is addressed by introducing an offline phase during which the open-loop optimal control problem is solved for a large number of possible situations, using e.g. multi-parametric non-linear programming. The solutions are then stored in a library such that they are directly available in the online phase. This concept will be used in Section 3.1 for the development of a multiobjective MPC algorithm for non-linear dynamical systems.

2.2.3 Motion Planning

An alternative for optimal strategy planning which has similarities to explicit MPC is the concept of *motion planning with motion primitives* going back to Frazzoli et al. [FDF05] (see also [Kob08, FOBK12]). Here, the challenge of online applicability is also addressed with a two-phase approach but in contrast to explicit MPC, trajectories for both the control and the state are obtained by combining several short pieces of simply controlled trajectories that are stored in a motion planning library. These motion primitives can be combined in order to create longer trajectories. In the online phase, the optimal sequence of motion primitives is determined from the motion planning library using e.g. graph search methods [Kob08]. To reduce the computational effort, this approach extensively relies on exploiting symmetries in the dynamical control system such that a motion primitive can be used in multiple situations, e.g. by performing a translation or rotation under which the dynamics are invariant. The invariances are formally described by a finite-dimensional Lie group G and its group action $\boldsymbol{\psi} : G \times \mathcal{Y} \to \mathcal{Y}$, where $\mathcal{Y}$ is the state space. A dynamical control system (ODE) is invariant under the group action $\boldsymbol{\psi}$, or equivalently, G is a symmetry group for the system (ODE), if

$$\boldsymbol{\psi}(g, \boldsymbol{\varphi}_{\boldsymbol{u}}(\boldsymbol{y}_0, t)) = \boldsymbol{\varphi}_{\boldsymbol{u}}(\boldsymbol{\psi}(g, \boldsymbol{y}_0), t) \quad \forall g \in G, \tag{2.25}$$

where $\boldsymbol{u}_0 \in \mathcal{U}$, $t \in [t_0, t_e]$ and $\boldsymbol{u} : [t_0, t_e] \to \mathcal{U}$ are piecewise-continuous control functions. This means that the group action on the state commutes with the flow

$\varphi_u(\boldsymbol{y}_0, t)$ of the dynamical control system (ODE).

Invariance leads to the concept of *equivalent trajectories*. Two trajectories are equivalent if they can be exactly superimposed through time translation and the action of the symmetry group. In the classical concept of motion primitives, all equivalent trajectories are summed up in an equivalence class, i.e. only a single representative is stored that can be used at many different points when transformed by the symmetry action. In other words, controlled trajectories that have been computed for a specific situation are also applicable in many different (equivalent) situations. In the multiobjective MPC algorithm presented in Section 3.1, the concept of motion primitives will be utilized to reduce the number of MOPs that have to be solved offline.

2.3 Reduced Order Modeling

Simulations are becoming more and more important in the design, optimization and control of complex dynamical systems. With the tremendous advances in computing, very large systems are nowadays easily solvable that would have been considered as too expensive only a few years ago. Due to this, simulation is today accepted as a third discipline besides theory and (physical) experiments [SvdVR08]. However, with increasing computational capabilities, systems of even larger complexity come into focus. In engineering, it is nowadays often of interest to simulate the dynamics of non-linear multi-physics systems such as fluid structure interaction or combined mechanical and electromagnetic behavior. These systems possess a large number of degrees of freedom that can easily reach the order of 10^6 to 10^8, in particular if the dynamics are described by partial differential equations that have to be discretized by a numerical mesh. If one wants to use these models for real-time control (such as *MPC*) or in a multi-query context such as optimization or parameter estimation, the capacities of even the most advanced computers quickly become insufficient.

To address this problem, *reduced order modeling (ROM)* is frequently applied. This is a powerful technique where large scale (also called *high-fidelity*) models are replaced by *surrogate models* of low dimension (see e.g. [ASG01, BMS05, QHS+05] for an overview). Many model order reduction techniques are data-driven [PW15], meaning that the large scale system has to be evaluated at least once to collect data. Based on this data, a much smaller model is created that approximates the solution of the original system. The reduction of the dimension results in a significant reduction of the computational effort which can be of several orders. At the same time, one has to accept an error in the reduced model. Hence, a trade-off between accuracy and speed-up has to be selected which depends on the specific situation the ROM is derived for. In the early stages of an optimization routine

or the development process of a new product, it may be necessary to evaluate many different parameters or designs such that low computing times are important while in other situations, it may be necessary to increase the accuracy, e.g. to avoid violations of critical constraints. In order to combine the strengths of detailed and efficient simulations, researchers have also started investigating *multi-fidelity methods* [REWH08, PWG16], where models of various degrees of detail are used hierarchically or alternatingly.

One of the most widely used techniques in model order reduction is *Proper Orthogonal Decomposition (POD)*. This method, also known as *Principal Component Analysis* or *Karhunen-Loève decomposition* [SvdVR08], was first introduced by Lumley [Lum67] to identify coherent structures in turbulent flows. The first ROMs based on POD were introduced by Sirovich in 1987 [Sir87]. Since the ROMs in this thesis are based on POD and Galerkin projection, this approach is presented in more detail in Sections 2.3.1 and 2.3.2, an introduction to ROM-based optimal control is presented in Section 2.3.3.

Besides POD, there are various other model order reduction techniques, each of which has specific strengths. The *Reduced Basis Method (RB)* (see e.g. [IR01, GMNP07, RHP08, HO08, AZF12]) is widely used for parameter dependent problems. The concept here is to collect data from the full system at a small number of parameter values and then construct a reduced basis with which the system can be evaluated at arbitrary parameter values. This method has successfully been combined with a weighted sum method to solve multiobjective optimal control problems constrained by semi-linear parabolic PDEs [IUV13]. Concepts from localized reduced-basis methods [AHKO12, OS15] will be adopted in Section 5.4 for set-oriented multiobjective optimal control of PDEs.

A relatively new approach to reduced order modeling is based on the Koopman operator (see e.g. [Mez13]). The Koopman operator is a linear but infinite-dimensional operator describing the dynamics of observables. In order to characterize it, one can compute its eigenvalues, eigenmodes and eigenfunctions. The eigenmodes are numerically computed by a method called *Dynamic Mode Decomposition (DMD)*, which was originally introduced by Schmidt [Sch10], see also [RMB+09, TRL+14, WKR15, KGPS16] for various extensions.

In contrast to POD, the DMD modes are not orthonormal. Furthermore, the eigenvalues are not related to the amount of information that is contained in the respective mode. On the other hand, similar to Fourier series analysis, each DMD mode is related to a single frequency [RBW+09] and the (complex) eigenvalues are related to the frequency (phase) and decay or growth (absolute value) of the corresponding mode. If the observable is defined as the values on all nodes of a grid of a discretization of a PDE (full-state observable [TRL+14]), then DMD modes have the same dimension as POD modes. First approaches utilizing Koopman

operator-based reduced order models in control problems have been presented in [PBK14, Mez16, PBK16, BBPK16].

2.3.1 Reduced Models via Galerkin Projection

Among the most widely used approaches for non-linear optimization based on ROMs is POD combined with Galerkin projection [KV99, SV10]. The general concept of a Galerkin projection (see e.g. [HLBR12] for an introduction in the context of fluid flows) is to find a finite-dimensional representation of an unknown function from an infinite-dimensional space (e.g. a temperature distribution $y(\boldsymbol{x}, t)$ which is the solution to a heat equation). This is realized by choosing elements $\psi_i(\boldsymbol{x})$ from a function space (e.g. the Hilbert space L^2) and representing $y(\boldsymbol{x}, t)$ in terms of $\{\psi_i(\boldsymbol{x})\}_{i=1}^{\infty}$:

$$y(\boldsymbol{x}, t) = \sum_{i=1}^{\infty} z_i(t)\psi_i(\boldsymbol{x}). \tag{2.26}$$

An adequate basis $\{\psi_i\}_{i=1}^{\infty}$ should satisfy several requirements [Rem96]:

(i) it should be complete in order to be able to exactly represent the solution $y(\boldsymbol{x}, t)$,

(ii) to allow for a unique solution, the elements $\psi_i(\boldsymbol{x})$ have to be linearly independent (this is the case if $\{\psi_i\}_{i=1}^{\infty}$ is an orthogonal system),

(iii) for a PDE-constrained problem, the Galerkin representation has to satisfy the same boundary conditions as the original solution.

The typical procedure to numerically solve a PDE is to discretize the spatial domain using a finite difference, finite volume (FV) or finite element method (FEM). This leads to a finite-dimensional system with a large number of degrees of freedom, making the problem costly to solve. In order to achieve a significant speedup, a reduced model has to satisfy another requirement:

(iv) in order to achieve high numerical efficiency, we want to compute a basis as small as possible, i.e. we want to find a truncation of (2.26) to ℓ basis functions:

$$y(\boldsymbol{x}, t) \approx \sum_{i=1}^{\ell} z_i(t)\psi_i(\boldsymbol{x}). \tag{2.27}$$

If such a representation has been found, the problem of determining y is transformed into the problem of determining the time-dependent coefficients $\boldsymbol{z} \in H^1((t_0, t_e); \mathbb{R}^{\ell})$. By inserting (2.26) into the equation describing the dynamics for y (i.e. an equation

of the form (PDE) or the corresponding weak formulation), a differential equation for $\boldsymbol{z}$ can be obtained[1]:

$$\begin{aligned} \dot{\boldsymbol{z}}(t) &= \boldsymbol{G}^r(\boldsymbol{z}(t), \boldsymbol{u}(t)), \qquad t \in (t_0, t_e], \\ \boldsymbol{z}(t_0) &= \boldsymbol{z}_0. \end{aligned} \tag{ROM}$$

In contrast to a standard spatial discretization approach by a numerical grid, the dimension ℓ is in many cases several orders smaller. In the next section POD will be discussed as a method to compute the basis $\boldsymbol{\psi}(\boldsymbol{x})$ and it will be shown that this yields an optimal compromise between the requirements (i) and (iv).

2.3.2 Proper Orthogonal Decomposition

Obviously, the requirements (i) and (iv) for a good basis are contradictory. Consequently, we search for a basis that is on the one hand capable of representing the solution with an error as small as possible and on the other hand is of small dimension ℓ, which can be addressed by POD [HLBR12]. The above requirements are formulated as an optimization problem, where the difference between the solution $y(\boldsymbol{x}, t)$ and its projection onto the reduced space has to be minimized (see also [Fah00, Vol11]):

$$\begin{aligned} \min_{\psi_1,\dots,\psi_\ell \in L^2} & \int_{t_0}^{t_e} \Big\| y(\cdot,t) - \sum_{i=1}^{\ell} \left(y(\cdot,t), \psi_i\right)_{L^2} \psi_i \Big\|_{L^2}^2 \,\mathrm{d}t \\ \text{s.t.} \quad & \left(\psi_i, \psi_j\right)_{L^2} = \delta_{i,j}, \quad 1 \le i,j \le \ell, \end{aligned} \tag{2.28}$$

where $\delta_{i,j}$ is the Kronecker delta. The procedure is practically realized by defining a time grid $[t_1, \dots, t_r]$ and taking r *snapshots* at these time instances. For this purpose, this approach is referred to as the *method of snapshots* [Sir87]. The optimization problem is thereby transformed to

$$\begin{aligned} \min_{\psi_1,\dots,\psi_\ell \in L^2} & \sum_{j=1}^{r} \Big\| y(\cdot,t_j) - \sum_{i=1}^{\ell} \left(y(\cdot,t_j), \psi_i\right)_{L^2} \psi_i \Big\|_{L^2}^2 \\ \text{s.t.} \quad & \left(\psi_i, \psi_j\right)_{L^2} = \delta_{i,j}, \quad 1 \le i,j \le \ell, \end{aligned}$$

[1] A more detailed description of the procedure, in particular the treatment of boundary conditions, is postponed to Section 5.1.

which is equivalent to

$$\begin{aligned} \max_{\psi_1,\dots,\psi_\ell \in L^2} & \sum_{i=1}^{\ell} \frac{1}{r} \sum_{j=1}^{r} \left(y(\cdot,t_j),\psi_i\right)_{L^2}^2 \\ \text{s.t.} \quad & \left(\psi_i,\psi_j\right)_{L^2} = \delta_{i,j}, \quad 1 \le i,j \le \ell, \end{aligned} \tag{2.29}$$

see e.g. [HLBR12]. Considering the case $\ell = 1$ [Fah00], this yields

$$\begin{aligned} \frac{1}{r}\sum_{j=1}^{r} \left(y(\cdot,t_j),\psi_1\right)_{L^2}^2 &= \frac{1}{r}\sum_{j=1}^{r} \left(\int_\Omega y(\boldsymbol{x},t_j)\psi_1(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}\right)\left(\int_\Omega y(\boldsymbol{z},t_j)\psi_1(\boldsymbol{z})\,\mathrm{d}\boldsymbol{z}\right) \\ &= \int_\Omega \psi_1(\boldsymbol{x}) \left[\int_\Omega \left(\frac{1}{r}\sum_{j=1}^{r} y(\boldsymbol{x},t_j)y(\boldsymbol{z},t_j)\right)\psi_1(\boldsymbol{z})\,\mathrm{d}\boldsymbol{z}\right]\mathrm{d}\boldsymbol{x} \\ &= \left(\psi_1, \mathcal{R}\psi_1\right)_{L^2}, \end{aligned}$$

with

$$\mathcal{R}\psi_1 = \int_\Omega \left(\frac{1}{r}\sum_{j=1}^{r} y(\boldsymbol{x},t_j)y(\boldsymbol{z},t_j)\right)\psi_1(\boldsymbol{z})\,\mathrm{d}\boldsymbol{z}.$$

The operator $\mathcal{R}$ is a linear, self-adjoint operator and hence possesses orthonormal eigenfunctions ψ_i with associated positive eigenvalues σ_i, $i = 1,\dots,r$. It is also referred to as the *two-point correlation* in the statistical analysis of turbulent flows [Pop00]. In [BHL93], it was shown that the ℓ eigenfunctions corresponding to the ℓ largest eigenvalues of $\mathcal{R}$ are the solution to problem (2.28) (see also [HLBR12] or [Fah00] for a proof using singular value decomposition). Furthermore, the eigenvalues can be utilized to determine the amount of information that is neglected by truncating the basis to size ℓ [Sir87]:

$$\epsilon(\ell) := \frac{\sum_{i=\ell+1}^{r}\sigma_i}{\sum_{j=1}^{r}\sigma_j}. \tag{2.30}$$

Besides optimality and orthonormality, the POD basis possesses several additional properties that are very useful for reduced order modeling [HLBR12, Fah00, TBD+17]:

(i) preservation of physical properties (when considering data from divergence-free flow fields, for example, the POD modes are also divergence-free),

(ii) computation of coherent structures containing the majority of the energy.

On the other hand, there are also weaknesses in the POD approach that should be taken into account when using a ROM:

(i) the technique does not distinguish between spatial and temporal structures. Consequently, POD modes generally correspond to a mix of frequencies,

(ii) POD arranges modes in terms of the amount of energy which does not necessarily correspond to the dynamical importance (in turbulence, for example, small scale motions can significantly influence large scale structures),

(iii) it is not always easy to determine the correct basis size. This is closely related to the previous point and results in a well-known drawback of the POD procedure, namely that dissipation, which frequently occurs on the small scales, is underrepresented,

(iv) for complex dynamical systems such as the Navier-Stokes equations, accuracy of the reduced order model can be very hard to achieve [NAM+03, SK04, GBZI04, NPM05, CBS05, BBI09, CAF09]. This can cause problems with the convergence in optimization routines such as those presented in [Fah00, BC08].

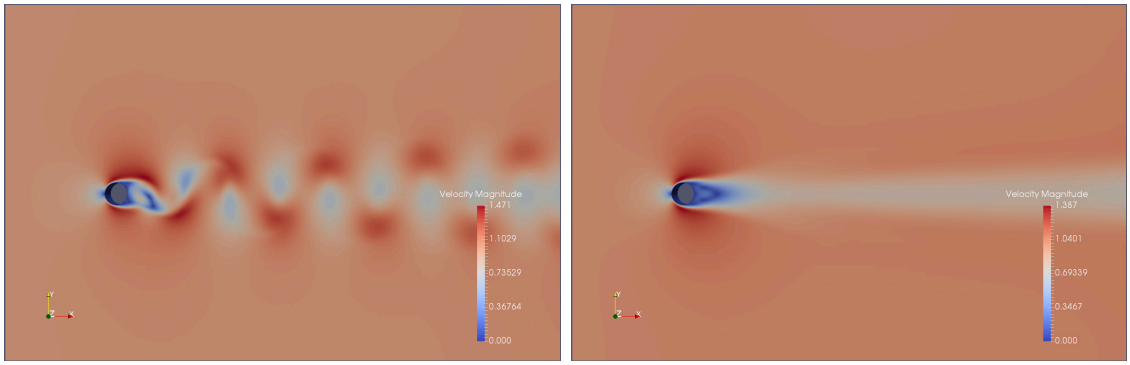
Nonetheless, POD has been and continues to be tremendously successful in a large variety of applications from image processing to reduced order modeling and optimal control. The key reasons are the optimality of the basis and that the truncation error can be estimated by the eigenvalues (cf. Equation (2.30)). This can be used to determine the error of ROMs (see e.g. [KV01, KV02]) or even of PDE-constrained optimal control problems [KV08, TV09].

In order to illustrate the POD procedure, let us consider the well-known example of the *von Kármán vortex street* which will also be studied in Section 5.1. There, the numerical scheme for obtaining the data will also be introduced in more detail. The von Kármán vortex street is a phenomenon occurring in two-dimensional fluid flows governed by the Navier-Stokes equations at moderate Reynolds numbers. In the wake of bluff bodies, vortices detach alternatingly from the upper and lower edge and a periodic solution emerges. When applying the POD procedure to numerical data either from experiments or numerical simulations, these are only available at discrete points $\boldsymbol{y}^d(t)$ in space[2]. Introducing a time grid $[t_1, \dots, t_r]$, these points can be arranged in the so-called *snapshot matrix* $\boldsymbol{S} = \left[\boldsymbol{y}^d(t_1), \dots, \boldsymbol{y}^d(t_r)\right] \in \mathbb{R}^{2n_y, r}$. In this example, the data has been obtained using a finite volume scheme and then interpolating the data on a finite element grid with n_y nodes. We collect $2n_y$ measurements (i.e. the two velocity components at n_y nodes) at r different time steps and solve the r-dimensional eigenvalue problem [KV02, Fah00]:

$$\mathcal{R}\boldsymbol{v}_i = \boldsymbol{S}^\top \mathcal{M} \boldsymbol{S}\boldsymbol{v}_i = \sigma_i \boldsymbol{v}_i, \qquad i = 1, \dots, r, \tag{2.31}$$

where $\mathcal{M} \in \mathbb{R}^{2n_y, 2n_y}$ is the finite element mass matrix. Using the eigenvalues and eigenvectors from (2.31), the POD modes can be computed. Making use of the finite

[2] The superscript ^{d} indicates that this quantity is defined on a finite-dimensional grid instead of a function space.



(a) (b)

Figure 2.11: (a) A snapshot of the well-known *von Kármán vortex street*. (b) Mean value of the flow field.

element basis functions $\{\phi_j(\boldsymbol{x})\}_{j=1}^{n_y}$, these can then be embedded in an L^2 function space:

$$\boldsymbol{\psi}_i^d = \frac{1}{\sqrt{\sigma_i}} \boldsymbol{S} \boldsymbol{v}_i,$$
$$\boldsymbol{\psi}_i(\boldsymbol{x}) = \begin{pmatrix} \sum_{j=1}^{n_y} \psi_{i,j}^d \phi_j(\boldsymbol{x}) \\ \sum_{j=1}^{n_y} \psi_{i,j+n_y}^d \phi_j(\boldsymbol{x}) \end{pmatrix}.$$

Note that $\sigma_i > 0$ for $i = 1, \ldots, r$ due to $\mathcal{R}$ being self-adjoint.

Remark 2.3.1. *Alternatively, the POD modes $\boldsymbol{\psi}^d$ con be computed applying singular value decomposition to the matrix $\left(\mathcal{M}^{1/2}\right)^\top \boldsymbol{S}$ [Fah00].*

In order to preserve the boundary conditions, we compute the POD on the fluctuations around the mean flow field (see Figure 2.11 (b)). These satisfy homogeneous boundary conditions as do the resulting POD modes, see Section 5.1 for details. By choosing a value for ϵ, typically 0.99 or 0.999, the basis size can be determined. For many applications, the eigenvalues decay fast such that a truncation to a small basis is possible. In this case, the first six modes capture already approximately 99.9% of the energy.

Figure 2.12 shows the first four POD modes. The modes occur in pairs, the second one slightly shifted downstream. This is due to symmetries (see the double eigenvalues in Figure 2.13 (a)) in the problem, namely in the horizontal axis through the cylinder as well as on the upper and lower boundary, respectively. The first four modes already account for roughly 98.6% of the information, cf. Figure 2.13 (a). The error, i.e. the ratio of the truncated eigenvalues, is visualized in Figure 2.13 (b).

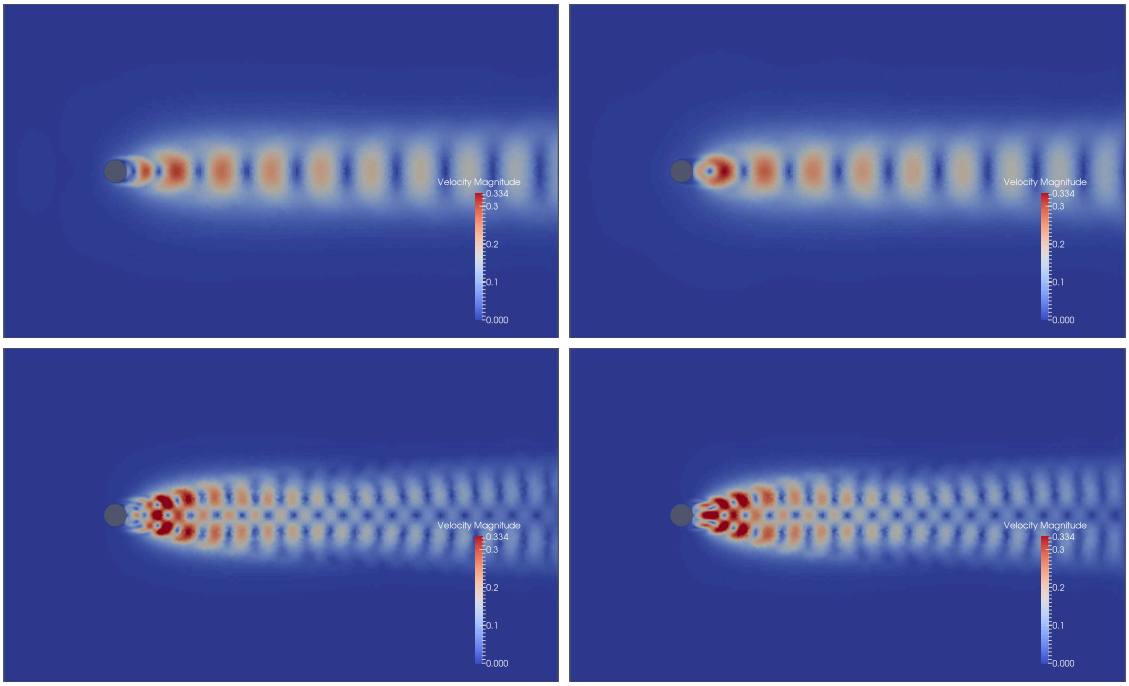
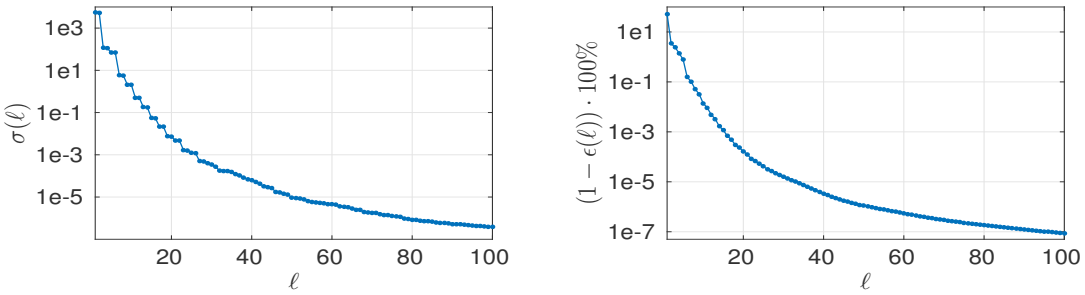


Figure 2.12: The first four POD modes of the fluctuating field of the *von Kármán vortex street*.



(a) (b)

Figure 2.13: (a) The first 100 eigenvalues σ_ℓ for the *von Kármán vortex street*. (b) The corresponding truncation error $\epsilon(\ell)$.

2.3.3 Optimal Control Based on Reduced Order Models

Motivated by the major achievements in the field of reduced order modeling, many researchers are going beyond merely reproducing the dynamics with ROMs but instead use them as a tool in a multi-query context such as parameter estimation [Las14], uncertainty quantification [GFWG10] or optimization (see e.g. [QHS+05, PWG16] for an overview). The first publications addressing MOCPs with PDE-constraints have recently appeared. Whereas in [ARFL09] the model is treated as a

black box and evolutionary algorithms are applied, a rigorous error analysis and convergence results can be found in [IUV13, ITV16, BBV16, Ban16, BBV17].

In principle, there are three different approaches how to implement a POD-based ROM in an optimization routine. The first one is to build a single ROM beforehand. The data is collected by sampling over a "wide range" of the control space, see e.g. [GPT99, Rav00, BCB05]. This method is the most efficient with respect to the reduction of computing time. However, it is often difficult to determine how to choose the reference control for which data is collected from the original system, i.e. how to appropriately cover the "wide range". Moreover, convergence to the solution of the original problem cannot be guaranteed.

The second approach is based on *trust region* approaches [Fah00, BC08, YM13, QGVW16]. Here, the reduced model is only valid within a specified distance to the reference control at which the data has been collected. Once the trust region boundary is reached, the full model is evaluated again and one can determine the efficiency of the objective reduction based on the surrogate model. Then a new ROM is obtained and the optimization proceeds using the updated model (see also Algorithm 5.3 in Section 5.2). For this approach, one can prove convergence to a local minimum of a single objective problem, see [Fah00] for a detailed description.

The third approach is to make use of the error estimate for the POD basis and perform an error analysis for the ROM-based optimal control problem [KV99, WP02, Row05, HV05, KV08, TV09]. This way, it is possible to specify error bounds on the state and adjoint equation, the objective functionals and respective gradients as well as the optimality condition. Moreover, this approach enables methods with adaptive basis sizes such that the desired bounds are satisfied, resulting in a convergent and highly efficient algorithm.

All three approaches will be investigated in Chapter 5 in the context of multiple objectives, more precisely in Sections 5.1, 5.2 and 5.4, respectively.

3 Continuation of Parameter Dependent Pareto Sets

This chapter addresses the problem of having to solve a large number of MOPs or MOCPs (e.g. in the presence of additional parameters) and how to exploit structure in the solutions to accelerate the computation. To motivate the necessity for efficient algorithms for these types of problems, an example from industry is presented in Section 3.1. There, an intelligent cruise control for autonomously driven electric vehicles is developed. In contrast to many other approaches for autonomous driving, multiple criteria are considered as equally important here. On the one hand, the vehicle should drive as fast as possible while on the other hand, energy efficiency has to be maximized. Since it is numerically infeasible to solve the corresponding MOCP in real-time, a new approach motivated by *explicit model predictive control* is presented, where a large number of solutions is computed in an offline phase and then stored in a library for later use. Since the number of problems grows exponentially with the number of parameters, the efficient computation of a new solution starting from an already known solution with slightly different parameter values is addressed in Section 3.2. In Section 3.3, the benefits of this approach will be validated, using again an example from autonomous driving.

3.1 Multiobjective Model Predictive Control of Electric Vehicles

Alternative drive technologies have gained more and more attention during the last years. This is mainly due to an increasing awareness of the impact of CO_2 emissions on climate change and the limitation of fossil fuels. In transportation, immense research activities have been conducted to develop new or improve existing concepts for electrically powered vehicles (EVs). Provided that the electric energy is harvested by renewable resources, EVs produce neither CO_2 nor NO_x and significantly less particulate matter compared to conventional and hybrid vehicles.

Up to now, EVs suffer from a reduced range compared to conventional vehicles powered by internal combustion engines. This is caused by the high battery cost as well as the limited lithium ion battery storage density. Therefore, strategies to increase the EV range without enlarging the battery play an important role for

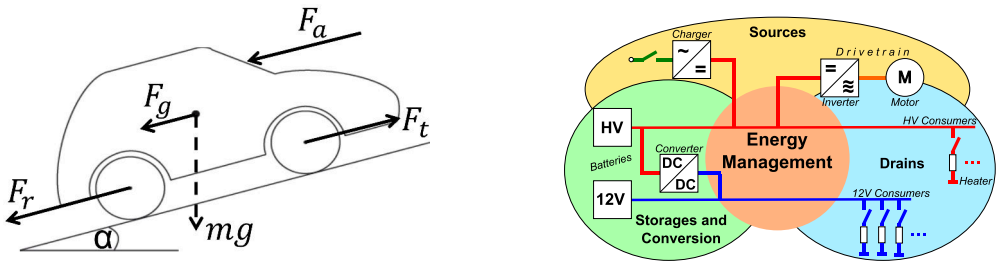
electro-mobility. There are various attempts to overcome the range limitations. Studies (see e.g. [BWC12]) have shown that the driving style has a large impact on energy consumption. Therefore, an intelligent controller acting on the drivetrain may enhance the range of EVs without changing any of the core components like the battery or the motor.

In this Section a new algorithm for multiobjective MPC of non-linear systems is presented. Problems with multiple criteria have been addressed by several authors using scalarization techniques (see e.g. [BM09a, BM09b] for a weighted sum or [ZFT12] for a reference point approach). For non-convex problems, scalarization and a-priori priorization may result in an unsatisfactory compromise such that we here want to compute the entire Pareto set in advance. To this end, elements from multiobjective optimal control, explicit MPC and motion planning with motion primitives are combined. The resulting algorithm consists of an offline phase during which MOCPs are solved and the corresponding Pareto sets and fronts are stored in a library for a wide range of possible scenarios (i.e. different speed limits, braking, accelerating). Invariances in the optimal control problem are exploited in order to reduce the number of problems that need to be solved. In the online phase, the currently active scenario is identified and the corresponding Pareto set is selected from the library. According to a decision maker's preference, an optimal compromise is then selected from the Pareto set and the first part of the solution is applied to the system. Similar to MPC, this is done repeatedly such that feedback control is realized. The difference to other approaches is the possibility to interactively choose between different objectives such that the system behavior can easily be modified. This can be very useful for autonomous driving, where one is interested in reaching a destination as fast as possible while minimizing the energy consumption.

In Section 3.1.1, the MOCP for the EV is formulated and solved offline for a test scenario. The results are computed with the reference point method (see Section 2.1.4) as well as the subdivision algorithm (see Section 2.1.5) in order to compare the algorithms. The multiobjective MPC algorithm within which a large number of MOCPs has to be solved is presented in Section 3.1.2. Due to the numerical cost, the method is currently restricted to constant control inputs. The results presented in this section have appeared in a series of papers [DEF+14, DEF+16, EPS+16, PSOB+17]. They were obtained in cooperation with *Hella KGaA Hueck & Co.* within the leading edge cluster *Intelligent Technical Systems OWL (it's OWL)*. The author has made significant contributions to the results presented therein.

3.1.1 Multiobjective Optimal Control of Electric Vehicles

All computations in the section are based on a very accurate EV model developed in [MKDB12]. The dynamics consists of a mechanical and an electrical subsystem (cf. Figure 3.1) which are coupled by efficiency maps. The subsystems are modeled using basic physical laws which results in a system of non-linear coupled ODEs. The control input $u(t)$ is the accelerator pedal position which can be set to values within 0 and 100. The model has been implemented in MATLAB/Simulink and has been validated against experimental data.



(a) (b)

Figure 3.1: (a) Mechanical subsystem of the EV model taking the forces acting on the vehicle into account. (b) Electrical subsystem consisting of motors, consumers, batteries, the drivetrain, etc.

As the initial scenario, we consider a fixed driving time of 250 seconds and an artificial track (see the height profile $h(t)$ in Figure 3.2 (b), bottom). We want to maximize the driven distance (i.e. the final position $p(t_e)$) as well as the final battery state of charge $S(t_e)$:

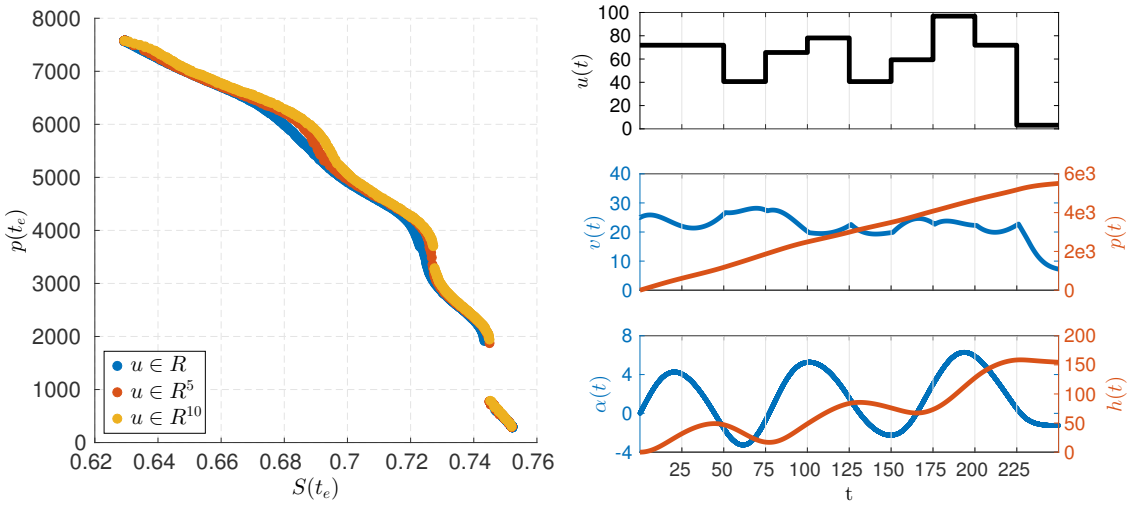
$$\begin{aligned} & \min_{u \in L^2} \begin{pmatrix} -S(t_e) \\ -p(t_e) \end{pmatrix}, \\ \text{s.t.} \quad & \dot{\boldsymbol{y}}(t) = \boldsymbol{F}(\boldsymbol{y}(t), u(t)), \qquad t \in (t_0, t_e], \\ & \boldsymbol{y}(t_0) = \boldsymbol{y}_0, \\ & u_l \leq u(t) \leq u_u, \qquad t \in (t_0, t_e], \end{aligned} \tag{3.1}$$

where the dynamics $\boldsymbol{F}$ is incorporated in the MATLAB/Simulink model.

In order to solve (3.1), we choose a direct approach and discretize the control into n steps over which the pedal position is constant, i.e. $u_i(t) \in [0, 100]$ for $t \in [t_{i-1}, t_i]$, $i = 1, \ldots, n$ with $t_0 < t_1 < \ldots < t_n = t_e$. The scalar problems in the reference point method (cf. Algorithm 2.1) are solved using an SQP method and the gradients are computed using finite differences. Note that the model is formally non-differentiable so that this step has to be treated with care. Thus, we first apply the gradient-free sampling algorithm (Algorithm 2.3) and use the results to verify the results

obtained with the reference point method (Algorithm 2.1). The numerical procedure is described in more detail in [DEF+14, DEF+16].

Figure 3.2 shows the Pareto front obtained with the sampling algorithm for different decision space dimensions as well as one EV simulation with a Pareto optimal pedal position profile and the resulting velocity profile. We observe that large values of u on positive slopes α and lower values on negative slopes are beneficial for the energy consumption. Solutions with a higher decision space dimension are obviously



(a) (b)

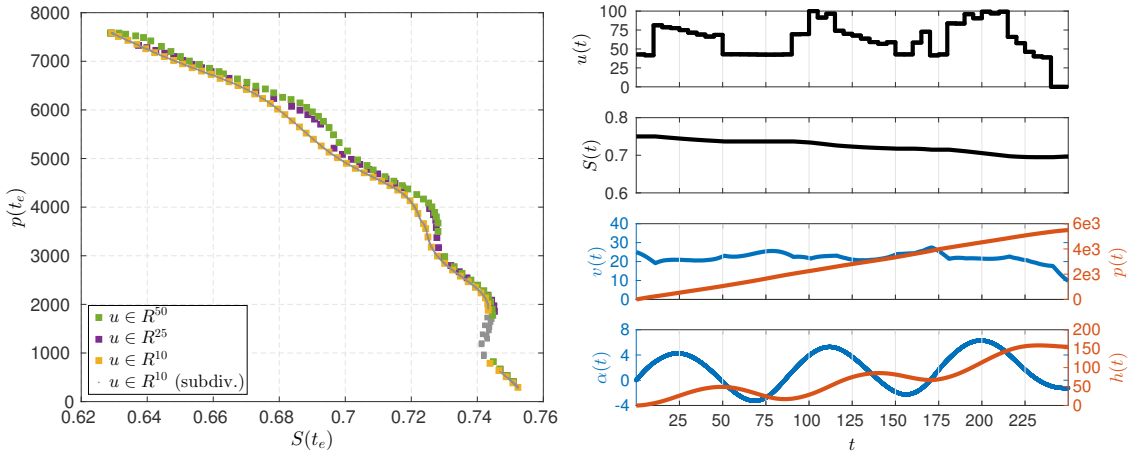
Figure 3.2: (a) Pareto front computed with the sampling algorithm for different decision space dimensions (the points are the images of the box centers). (b) EV simulation with a Pareto optimal pedal position profile ($\boldsymbol{u} \in \mathbb{R}^{10}$, $S(t_e) = 0.6914$, $p(t_e) = 5800$ m).

always at least as good as the lower dimensional solutions (cf. Figure 3.2 (a)). Additionally, the difference between the solutions is largest in the middle section. This is due to a higher variability in this part while near the ends of the front, the pedal position is close to the maximal or minimal value at all times.

When looking at the Pareto optimal points around $S(t_e) \approx 0.745$ (as well as $S(t_e) \approx 0.725$ for $\boldsymbol{u} \in \mathbb{R}^{10}$), we observe a gap which is caused by the EV's recuperation technique. The last point at the low distance part of the Pareto front corresponds to a stop at the top of a hill. Increasing the pedal position only slightly results in a final position with a negative inclination α. Since the EV can roll down the slope and recharge its battery via recuperation, a slight reduction of the objective $S(t_e)$ leads to a significant increase of the second objective $p(t_e)$, i.e. we can drive downhill without requiring further energy. This results in a gap in the front. The dents in Pareto front originate from the same physical effect. Due to the track

slope alternating between positive and negative values, the slope of the Pareto front is closely connected to the height profile.

A comparison of the results of the subdivision algorithm and the reference point method (cf. Figure 3.3 (a)) shows good agreement for the case $\boldsymbol{u} \in \mathbb{R}^{10}$, indicating that the reference point method also yields satisfactory results despite its local nature and the non-differentiability of the model. This allows us to compute Pareto sets for higher decision space dimensions. Note that the requirement $\boldsymbol{T} \leq_p \boldsymbol{J}^*$ is not satisfied here and Theorem 2.1.17 is not applicable. Hence, the reference point method effectively traces the boundary of the reachable set and there are also points included which are not Pareto optimal (cf. Remark 2.1.18). Consequently, one has to identify non-optimal solutions using a non-dominance test. These are plotted in gray at $S(t_e) \approx 0.74$.



(a) (b)

Figure 3.3: (a) Pareto front computed with the reference point method for different decision space dimensions. (b) Simulation with a Pareto optimal pedal position profile ($\boldsymbol{u} \in \mathbb{R}^{50}$, $S(t_e) = 0.6934$, $p(t_e) = 5750$ m).

As can be seen in Figure 3.3, the resulting improvements become smaller with increasing decision space dimension. Although the discretization becomes finer, the pedal position profile looks qualitatively very similar to the 10-dimensional solution (cf. Figures 3.2 (b) and 3.3 (b), respectively).

The results show that considering multiple criteria can be useful in the development of an intelligent cruise control. However, considering a fixed time frame and neglecting disturbances is obviously infeasible for autonomous driving. Therefore, the results presented here are embedded into a newly developed multiobjective MPC (MOMPC) framework in the next section.

3.1.2 The Offline-Online Multiobjective MPC Concept

In order to develop a closed-loop control for the EV with multiple objectives, the results from the previous section are coupled with an MPC framework. As presented in Section 2.2.2, in an MPC routine the optimal control problem is solved online repeatedly for varying time frames $[t_s, t_{s+p}]$ (with $t_s = s \cdot t_h$ and $t_{s+p} = (s+p) \cdot t_h$, $s = 1, 2, \ldots$). Despite initially being developed to stabilize a system [GP17], stabilization is not always the main concern. Considering the EV, for example, we only require a part of the state, namely the velocity, to remain within prescribed bounds, which gives us the opportunity to pursue additional objectives such as minimizing the energy consumption. This concept is known as *economic MPC* (see e.g. [RA09, DAR11]).

Since MOCPs are considerably more expensive to solve than scalar problems, it is computationally infeasible to directly include them in an MPC framework. A simple way to circumvent this problem is to scalarize the objective function by applying the weighted sum method, for example, and introducing a weighted objective $J_{\boldsymbol{\alpha}} = \sum_{i=1}^{k} \alpha_i J_i$. However, in this case an assumption on $\boldsymbol{\alpha}$ has to be made in advance which can in practice lead to unfavorable results. A slight increase in one objective might allow for a strong reduction in another one, for example. Hence, we want to have knowledge of the entire Pareto set during the MPC routine. To avoid large computing times during execution, we therefore split the computation in an *offline* and an *online phase*, similar to explicit MPC approaches [AB09] (as well as *reduced basis* concepts in reduced order modeling (cf. Section 5.4)).

The *offline phase* (Algorithm 3.1) consists of several steps. First, various *scenarios* are identified for which MOCPs need to be solved. The scenarios are determined by the system states and the constraints. Secondly, the dynamical control system is analyzed with respect to invariances in order to reduce the number of scenarios. In our approach, we extend the standard concept described in Section 2.2.3 by identifying symmetries in the solution of the MOCP with respect to the initial conditions $\boldsymbol{y}_0$:

$$\arg\min_{\boldsymbol{u}\in\mathcal{U}} \boldsymbol{J}(\boldsymbol{u}, \boldsymbol{y}_0) = \arg\min_{\boldsymbol{u}\in\mathcal{U}} \boldsymbol{J}(\boldsymbol{u}, \boldsymbol{\psi}(g, \boldsymbol{y}_0)) \quad \forall g \in G, \tag{3.2}$$

where the notation $\boldsymbol{J}(\boldsymbol{u}, \boldsymbol{y}_0)$ is introduced to indicate that the initial condition $\boldsymbol{y}_0$ is treated as an additional parameter. This means that we require the Pareto set to be invariant under group actions on the initial conditions. If the dynamics itself is also invariant under the same group action, then all trajectories contained in an equivalence class defined by the standard approach introduced in Section 2.2.3:

$$\boldsymbol{\psi}(g, \boldsymbol{\varphi}_{\boldsymbol{u}}(\boldsymbol{y}_0, t)) = \boldsymbol{\varphi}_{\boldsymbol{u}}(\boldsymbol{\psi}(g, \boldsymbol{y}_0), t) \quad \forall g \in G, \tag{2.25}$$

will also be contained in an equivalence class defined by (3.2). However, the first class may contain more solutions since we do not explicitly pose restrictions on the state but only require the respective Pareto sets to be identical.

Algorithm 3.1 (Offline Phase of the MOMPC algorithm)

1: Identify relevant scenarios
2: Symmetry analysis
3: Reduce the number of offline scenarios by exploiting symmetries
4: Solve an MOCP for each of the scenarios and store the solution in a library

By identifying invariances according to (3.2), the number of MOCPs can be reduced. If the system is invariant under translation of the initial position $p(t_0)$, for example, we do not need to solve multiple MOCPs that only differ in the position. Once these equivalence classes have been identified, we can reduce the number of possible scenarios accordingly. We then solve the resulting MOCPs on the prediction horizon t_p, introduce a parametrization $\boldsymbol{\rho} \in \mathbb{R}^{k-1}$ (which can then be chosen by the decision maker in the online phase) and store the Pareto sets and fronts in a library such that they can be used in the online phase. Since there is in general an infinite number of feasible initial conditions, there consequently exists an infinite number of scenarios that we have to consider. In practice, this obviously cannot be realized and we are restricted to a finite set of scenarios. In the online phase, we then select the scenario that is closest to the true initial condition. If a violation of the state constraints has to be avoided (for example, the EV is not allowed to go faster than the maximum speed), then a selection towards the "safe" side can be made. In case of the EV, we would consequently pick a solution corresponding to a velocity slightly higher than the actual velocity. This way, the maximally allowed acceleration would be bounded such that exceeding the speed limit is not possible.

The *online phase* is now basically a standard MPC approach (cf. Algorithm 3.2), the difference being that we select the solution of the control problem from a library instead of solving it in real-time, similar to explicit MPC approaches. The resulting algorithm thus provides a feedback law. In the offline phase, we define the scenarios in such a manner that the system cannot be steered out of the set of feasible states. This means that only controls $\boldsymbol{u}$ are valid that do not lead to a violation of the constraints. Additionally, we include scenarios which steer the system into the set of feasible states from any initial condition. In the literature, this is known as *viability*, cf. [GP17]. In case of the EV, for example, we have to include controls such that the velocity can be steered to values satisfying the constraints from any initial velocity.

The presented algorithm can be seen as an extension of extended MPC approaches to multiple objectives. We consider *economic* objectives (cf. [RA09]) and do not

Algorithm 3.2 (Online Phase of the MOMPC algorithm)

1: Measure the current system states that are necessary for the identification of the scenario
2: Choose the corresponding Pareto set from the library, i.e. with initial conditions closest to the current system state. (Due to the approximation, we cannot formally guarantee that the constraints are not violated. However, it is assumed that this is acceptable.)
3: Choose an optimal compromise $\boldsymbol{u}$ from the set, according to a decision maker's preference $\boldsymbol{\rho}$
4: Apply the first step (i.e. the sample time) of the solution $\boldsymbol{u}$ to the real system and go back to 1

focus on the stabilization of the system. This allows us to pursue multiple objectives between which a decision maker can choose dynamically, e.g. in order to react on changes in the environment or the system state itself. In contrast to weighting methods, the entire Pareto set is known, providing increased system knowledge.

Application to the Electric Vehicle

Due to the non-differentiability of the above EV model and in order to increase the numerical efficiency, slight simplifications of the model have been made in [EPS+16], namely by neglecting the influence of the temperature and disregarding the so-called *protection circuit* of the battery. These are justified since they only result in minor inaccuracies in the model when considering only short time horizons as in MPC. After some algebraic manipulations, this results in a system of four coupled nonlinear ODEs for the state variables *vehicle speed* v, *battery state of charge* S and the long term and short term voltage drops $U_{d,L}$ and $U_{d,S}$ which is of the form of (ODE):

$$\begin{aligned} \dot{\boldsymbol{y}}(t) &= \boldsymbol{F}(\boldsymbol{y}(t), u(t)), \qquad t \in (t_0, t_e], \\ \boldsymbol{y}(t_0) &= \boldsymbol{y}_0, \end{aligned}$$

with $\boldsymbol{y} = (v, S, U_{d,L}, U_{d,S})^\top \in H^1((t_0, t_e); \mathbb{R}^4)$ and $u \in L^2((t_0, t_e); \mathbb{R})$. The system is controlled by the torque $u(t)$ of the front wheels. Additionally, the battery current $I(t)$ is computed from the state $\boldsymbol{y}(t)$ via an algebraic equation and the position by integrating the velocity: $p(t) = \int_{t_0}^{t} v(\tau)\,\mathrm{d}\tau$. For the derivation and the exact formulation of the dynamical system, the reader is referred to [EPS+16].

Based on this system, problem (3.1) is extended to a more realistic scenario with fixed distances (i.e. variable final time) and bounds taking speed limits into

account:

$$\min_{u \in L^2} \begin{pmatrix} S(t_0) - S(t_e) \\ t_e - t_0 \end{pmatrix}, \tag{3.3}$$

$$\text{s.t.} \quad \dot{\boldsymbol{y}}(t) = \boldsymbol{F}(\boldsymbol{y}(t), u(t)), \qquad t \in (t_0, t_e], \tag{3.4}$$

$$v_{\mathsf{min}}(t) \leq v(t) \leq v_{\mathsf{max}}(t), \qquad t \in (t_0, t_e], \tag{3.5}$$

$$I_{\mathsf{min}}(t) \leq I(t) \leq I_{\mathsf{max}}(t), \qquad t \in (t_0, t_e], \tag{3.6}$$

$$\boldsymbol{y}(t_0) = \boldsymbol{y}_0, \; p(t_e) = p_e. \tag{3.7}$$

We set the final position p_e to 100 m, which means that the prediction horizon is based on the position instead of time. Correspondingly, the sample time is also specified with respect to the position, $t_s = 20\,\text{m}$. The conflicting objectives are to reach p_e as fast as possible (J_2) while minimizing the energy consumption (J_1). The battery current I is limited in order to avoid damage to the battery which results in implicit constraints on the control u. The velocity constraints are part of the scenarios which are defined in the first step of the offline phase.

The more invariances the MOCP possesses (in the sense of Equation (3.2)), the fewer problems need to be solved which significantly reduces the computational effort. Hence, we numerically analyze the system in this regard (cf. line 2 in Algorithm 3.1). Since the position p does not occur in the dynamical system (3.4), the dynamics is obviously invariant under translations in p. Moreover, when looking at the velocity v and the state of charge S (cf. Figures 3.4 (a) and 3.4 (b)), we see that the trajectories are almost invariant for a wide range of translated initial values of the state of charge $S(t_0)$ and otherwise fixed initial conditions. Note that this is not a strict invariance, almost invariance is interpreted as a low sensitivity with respect to $S(t_0)$. However, as argued above, we do not require invariances according to Equation (2.25) but according to the weaker condition (3.2). In contrast to that, when looking at Figure 3.4 (c), we observe that the dynamics is clearly not invariant under translations in the initial velocity $v(t_0)$. After performing the same analysis with regard to the other state variables $U_{d,L}$ and $U_{d,S}$, it can be concluded that we only need to define scenarios with respect to the initial velocity $v(t_0)$ and the active constraints $v_{\mathsf{min}}(t)$ and $v_{\mathsf{max}}(t)$.

A constraint on the velocity is given by the current speed limit $v_{\mathsf{max}}(p)$ which depends on the current vehicle position. Since we want to avoid interference with other vehicles by driving too slowly, we define a minimal velocity $v_{\mathsf{min}}(p) = 0.8 \cdot v_{\mathsf{max}}(p)$. (Here the velocities are functions of the position because they are given by the problem formulation this way. In the MOCP, they have to be reformulated as functions of time.) The *set of feasible states* is constrained by the velocities $v_{\mathsf{min}}(t) \leq v(t) \leq v_{\mathsf{max}}(t)$ which also determine the different scenarios. We distinguish between four cases (see Figure 3.5 (a)). While the cases constant velocity (box constraints) and stopping ($v = 0$ at the stop sign) are easily implemented, we introduce a linear

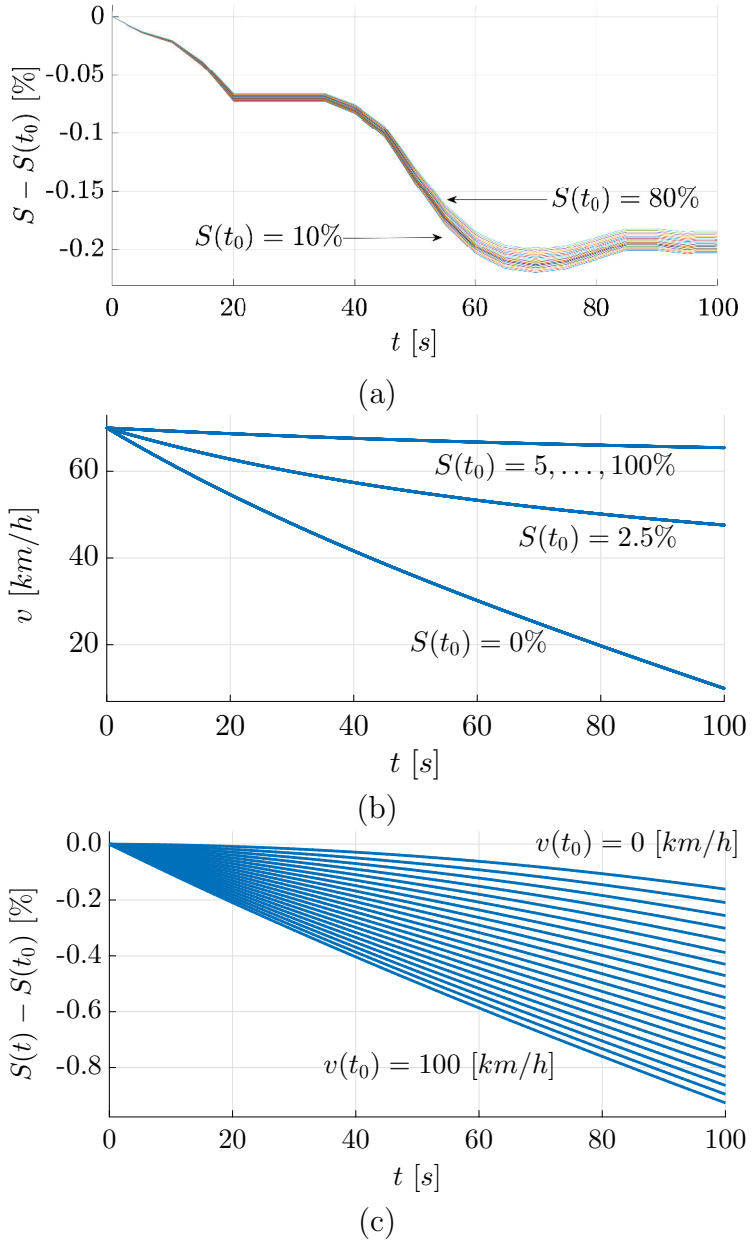


Figure 3.4: (a) Almost invariance of S with respect to the initial value $S(t_0)$. (b) Invariance of the velocity v with respect to the initial value $S(t_0)$ for $S(t_0) \geq 5\%$. (c) No invariance of the state of charge S with respect to the initial velocity $v(t_0)$.

constraint for the scenarios (ii) and (iii), respectively (see Figure 3.5 (b)) where, depending on the current velocity, a minimal increase or decrease $a(p) = (dv(p)/dp)$ must not be violated. An example is shown in Figure 3.6, where the Pareto set and the resulting velocity profiles are shown for the scenario $v(t_0) = 60$ km/h and

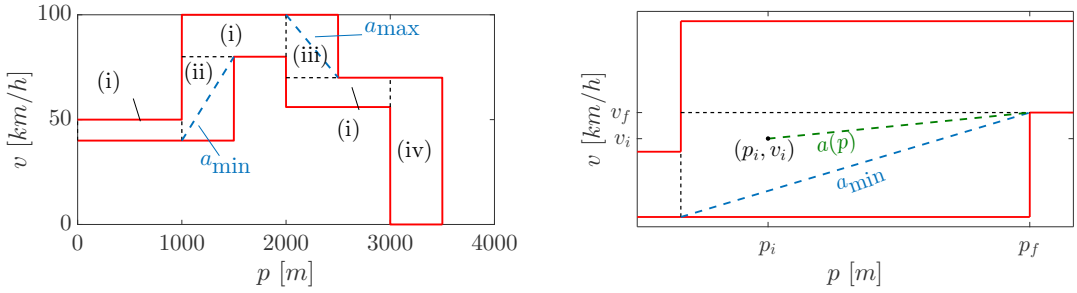


Figure 3.5: (a) Possible scenarios of boundary conditions. (i): constant velocity. (ii): acceleration. (iii): deceleration. (iv): stop sign. (b) Computation of lower bound a_{min} for the velocity gradient dv/dp.

$a_{\mathsf{min}} = 0.05\ \frac{km/h}{m}$. Note that the control u is constant over the prediction horizon in order to reduce the numerical effort. As mentioned before, only a finite number of initial conditions can be considered. Solving an MOCP for every step of 0.1 in the initial velocity leads to 1727 MOCPs in total.

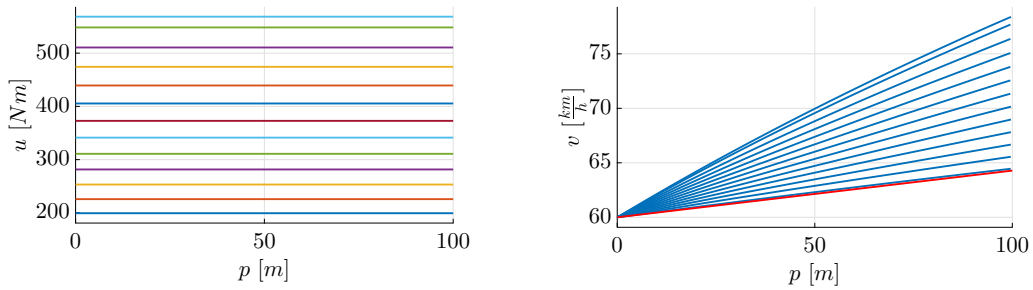


Figure 3.6: (a) Pareto set for an accelerating scenario with $v(t_0) = 60\ km/h$ and $a(0) = 0.05\ \frac{km/h}{m}$. (b) The corresponding trajectories $v(t)$.

Algorithm 3.2 is executed during the online phase. In each sample time, the current velocity and the constraints (for the current position) are evaluated in order to determine the active scenario. The corresponding Pareto set is then selected from the library and – according to the weighting parameter $\rho \in [0, 1]$ determined by the decision maker – an optimal compromise is chosen which is then applied to the system. On a standard computer, this operation takes in the order of 10^{-3} seconds in MATLAB.

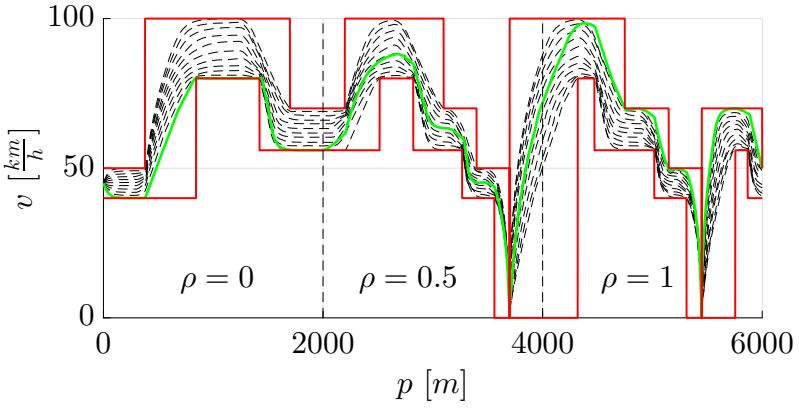


Figure 3.7: Different trajectories computed by the MPC approach. The dashed lines correspond to constant weights ρ whereas the green line uses dynamic weighting ($\rho = 0/0.5/1.0$, respectively).

3.1.3 Results

Several solutions with different weights ρ are shown in Figure 3.7 for an example track including two stop signs. The *set of feasible velocities* is bounded by the red lines. The dashed lines correspond to constant weights, varying from $\rho = 0$ (energy efficiency) to $\rho = 1$ (high velocity) and the solid green line is a solution where the weighting is changed from 0 over 0.5 to 1 during driving. The vehicle is obviously driving according to the decision maker's preference. This means that a closed-loop control has been realized for which the objectives can be adjusted dynamically. The adjustment can either be made manually or by a heuristic which takes into account the track, the battery state of charge and the current traffic. The objective function values for the entire track and different values of ρ are shown in Figure 3.8 (a).

In order to evaluate the quality of the solution, we compare it to a control computed via dynamic programming (DP, see [BS15] for an introduction and [SG09] for the algorithm). For computational reasons, the comparison is performed on a shorter track without stop signs and a relatively coarse discretization leading to a 100-dimensional problem. In the DP problem, a simplified linear model (cf. [EPS+16]) is used and the objective is a weighted sum of the MOCP (3.3) with $J = t_e + \beta E(t_e)$, where $E : \mathbb{R} \to \mathbb{R}$ is the consumed energy computed by integrating over the wheel torque and $\beta = 6 \cdot 10^{-5}$. In Figure 3.8 (b), we see that the solution obtained via DP is superior to the approach introduced in this section. This is not surprising since only a finite horizon is considered in MPC such that the results are at best sub-optimal [GP17] whereas the entire track is considered at once in DP. Conse-

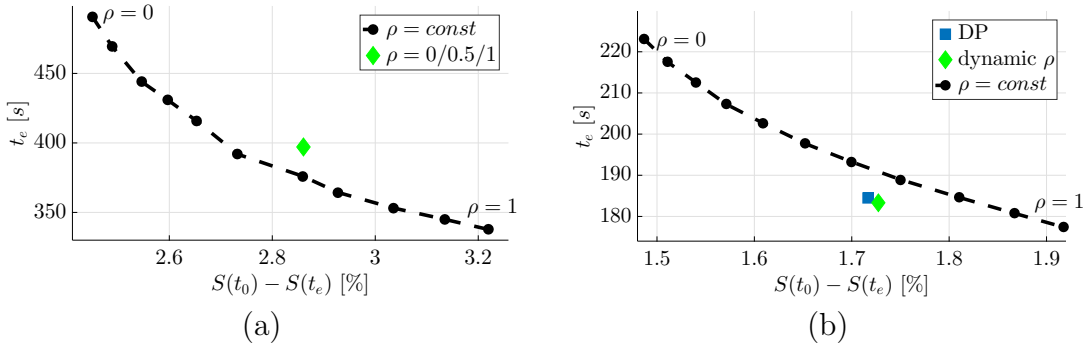


Figure 3.8: Function values for the scenarios depicted in Figure 3.7 and in Figure 3.9 for different weights ρ and in comparison to the Dynamic Programming solution.

quently, the DP algorithm is not real-time applicable and does not possess feedback behavior. Additionally, only constant torques over the prediction horizon have been considered until now. For future work, it will therefore be interesting to investigate the benefits of a refined discretization.

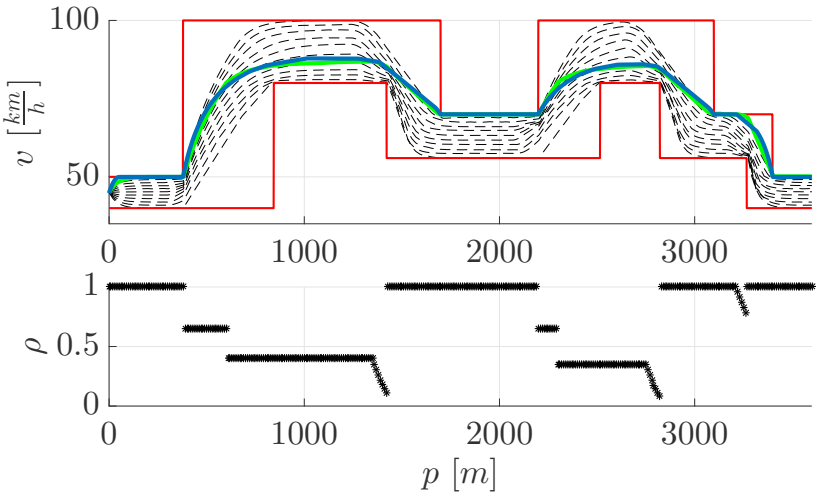


Figure 3.9: Validation of the approach versus a Dynamic Programming solution (blue). Green line: dynamic weighting according to the lower plot.

When using a simple heuristic for the preference ρ instead of fixed values (larger values for ρ at low velocities, lower values at high velocities and linear changes in ρ when approaching braking maneuvers, see Figure 3.9 (bottom)), we see that the quality of the solution can be significantly improved and is now comparable to the global optimum obtained by DP. The resulting trajectories as well as the function values J_1 and J_2 almost coincide, cf. Figures 3.9 (top) and 3.8 (b). By this, a second way to utilize the results is obtained. On the one hand, a decision maker can freely

select the preference and on the other hand, ρ can be determined by a heuristic, leading to solutions comparable to the global optimum.

Due to the computational cost, the decision space of the numerical example in this section was limited to constant controls. In order to increase the degree of the control input, it is necessary to increase the efficiency of solving the large number of MOCPs in the presence of additional parameters. This will be addressed in the next section.

3.2 Continuation of Pareto Sets

We have seen in the previous section that the multiobjective MPC algorithm results in the necessity to solve a large number of MOCPs for different parameter values. Similar to explicit MPC, the number of problems increases exponentially with the number of parameters. The problem of infeasible computational cost has previously been avoided by considering only constant controls. Since this is a severe restriction, this section is concerned with the development of a continuation method for entire Pareto sets with additional parameter dependencies. Having in mind direct approaches by which MOCPs can be transformed into MOPs, the problems considered here are restricted to a finite dimension. The parameter dependent problem can therefore be formulated as

$$\begin{aligned} \min_{\boldsymbol{u}\in\mathbb{R}^n} \boldsymbol{J}(\boldsymbol{u},\boldsymbol{p}) &= \min_{\boldsymbol{u}\in\mathbb{R}^n} \begin{pmatrix} J_1(\boldsymbol{u},\boldsymbol{p}) \\ \vdots \\ J_k(\boldsymbol{u},\boldsymbol{p}) \end{pmatrix} \\ \text{s.t.} \quad \boldsymbol{g}(\boldsymbol{u},\boldsymbol{p}) &\leq \boldsymbol{0}, \\ \boldsymbol{h}(\boldsymbol{u},\boldsymbol{p}) &= \boldsymbol{0}, \end{aligned} \tag{3.8}$$

with $\boldsymbol{p}\in\mathbb{R}^{n_p}$, $\boldsymbol{J}:\mathbb{R}^n\times\mathbb{R}^{n_p}\to\mathbb{R}^k$, $\boldsymbol{g}:\mathbb{R}^n\times\mathbb{R}^{n_p}\to\mathbb{R}^l$ and $\boldsymbol{h}:\mathbb{R}^n\times\mathbb{R}^{n_p}\to\mathbb{R}^m$. Here, $\boldsymbol{J}(\boldsymbol{u},\boldsymbol{p})$, $\boldsymbol{g}(\boldsymbol{u},\boldsymbol{p})$ and $\boldsymbol{h}(\boldsymbol{u},\boldsymbol{p})$ are introduced to account for the parameter dependency. Parameter dependent MOPs have been studied extensively in [Wit12]. However, continuation methods were applied to single points in the Pareto set. In [MS17, SLT+17], the authors follow a similar approach. Although the method presented here is mainly motivated by numerical experiments, a theoretical foundation for the applicability of the proposed method will be presented first.

The idea is very similar to the continuation approach presented in Section 2.1.4, where the set of points satisfying $\boldsymbol{H}=\boldsymbol{0}$ (Equation (2.10)) was sought. Here, we additionally consider a parameter $\boldsymbol{p}\in\mathbb{R}^{n_p}$ such that the set of substationary points

for a certain parameter $\mathcal{P}_{S,\mathsf{sub}}(\boldsymbol{p})$ value is defined by:

$$\boldsymbol{H}(\boldsymbol{u}^*, \widetilde{\boldsymbol{\gamma}}^*, \boldsymbol{\alpha}^*, \boldsymbol{p}) = \begin{pmatrix} \sum_{i=1}^{k} \alpha_i^* \nabla J_i(\boldsymbol{u}^*, \boldsymbol{p}) + \sum_{j=1}^{m+|\mathcal{I}|} \widetilde{\gamma}_j^* \nabla \widetilde{h}_j(\boldsymbol{u}^*, \boldsymbol{p}) \\ \widetilde{\boldsymbol{h}}(\boldsymbol{u}^*, \boldsymbol{p}) \\ \sum_{i=1}^{k} \alpha_i^* - 1 \end{pmatrix} = \boldsymbol{0}, \tag{3.9}$$

where the equality and the active inequality constraints have again been assembled in $\widetilde{\boldsymbol{h}}$. Consequently, we have

$$\mathcal{P}_{S,\mathsf{sub}}(\boldsymbol{p}) = \{ \ (\boldsymbol{u}^*, \widetilde{\boldsymbol{\gamma}}^*, \boldsymbol{\alpha}^*) \in \mathbb{R}^n \times \mathbb{R}^{m+|\mathcal{I}|} \times \mathbb{R}^k \mid \boldsymbol{H}(\boldsymbol{u}^*, \widetilde{\boldsymbol{\gamma}}^*, \boldsymbol{\alpha}^*, \boldsymbol{p}) = \boldsymbol{0} \}. \tag{3.10}$$

In what follows, we will assume that the requirements for $\mathcal{P}_{S,\mathsf{sub}}$ to be a manifold are satisfied, i.e. the objective functions and the constrains are twice continuously differentiable, cf. Section 2.1.3. Then for a fixed value $\boldsymbol{p}$, the corresponding set of substationary points $\mathcal{P}_{S,\mathsf{sub}}(\boldsymbol{p})$ is a $(k-1)$-dimensional manifold. Assuming that the function $\boldsymbol{H}$ is continuously differentiable with respect to $\boldsymbol{p}$, this result can be extended to the parameter $\boldsymbol{p}$:

Theorem 3.2.1. *Consider the parameter dependent MOP (3.8) and let $\boldsymbol{J}$, $\boldsymbol{g}$ and $\boldsymbol{h}$ be twice continuously differentiable with respect to $\boldsymbol{u}$ and once continuously differentiable with respect to $\boldsymbol{p}$. Denote by $\mathcal{M}_{\boldsymbol{p}} := \{(\boldsymbol{u}^*, \widetilde{\boldsymbol{\gamma}}^*, \boldsymbol{\alpha}^*, \boldsymbol{p}) \in \mathbb{R}^{n+m+|\mathcal{I}|+k+n_p} \mid \boldsymbol{H}(\boldsymbol{u}^*, \widetilde{\boldsymbol{\gamma}}^*, \boldsymbol{\alpha}^*, \boldsymbol{p}) = \boldsymbol{0}$ and $\boldsymbol{\alpha}_i^* >_p \boldsymbol{0}\}$ the set of points satisfying the first order conditions (KKT) for admissible parameter values $\boldsymbol{p}$.*

If the Jacobian $\boldsymbol{H}'$ has full rank in one point $(\boldsymbol{u}^*, \widetilde{\boldsymbol{\gamma}}^*, \boldsymbol{\alpha}^*, \boldsymbol{p})$, i.e.

$$rank(\boldsymbol{H}'(\boldsymbol{u}^*, \widetilde{\boldsymbol{\gamma}}^*, \boldsymbol{\alpha}^*, \boldsymbol{p})) = n + m + |\mathcal{I}| + 1, \tag{3.11}$$

then $\mathcal{M}_{\boldsymbol{p}}$ is a $(k - 1 + n_p)$-dimensional differentiable submanifold of $\mathbb{R}^{n+m+|\mathcal{I}|+k+n_p}$ in a neighborhood of $(\boldsymbol{u}^*, \widetilde{\boldsymbol{\gamma}}^*, \boldsymbol{\alpha}^*, \boldsymbol{p})$.

If all points $(\boldsymbol{u}^*, \widetilde{\boldsymbol{\gamma}}^*, \boldsymbol{\alpha}^*, \boldsymbol{p}) \in \mathcal{M}_{\boldsymbol{p}}$ satisfy the rank condition (3.11) then $\mathcal{M}_{\boldsymbol{p}}$ is a $(k - 1 + n_p)$-dimensional differentiable submanifold of $\mathbb{R}^{n+m+|\mathcal{I}|+k+n_p}$.

Proof. Due to the applicability of the Implicit Function Theorem, the proof is analog to the proof of Theorem 2.1.15. □

Obviously, there exist problems where the requirements for Theorem 3.2.1 are not satisfied. However, in this situation $\mathcal{P}_{S,\mathsf{sub}}(\boldsymbol{p})$ may still depend continuously on $\boldsymbol{p}$. This is for example the case when the objective functions as well as constraints are Lipschitz continuous with respect to the parameter. Consequently, so are their respective gradients and also the function $\boldsymbol{H}$, see also Theorem 2.2.2 on p. 37.

Remark 3.2.2. *Note that it is in general difficult to compute the tangent space of $\mathcal{M}_{\boldsymbol{p}}$ for the entire set of substationary points $\mathcal{P}_{S,\mathsf{sub}}(\boldsymbol{p})$. Instead, we will assume in the following that the above results allow to find a parametrization of $\mathcal{M}_{\boldsymbol{p}}$ by the parameter $\boldsymbol{p}$ and will develop numerical methods which exploit the continuity with respect to $\boldsymbol{p}$ and approximate the tangent space from numerical data.*

3.2.1 A Predictor-Corrector Method for Parameter Dependent MOPs

Motivated by the introductory considerations and Theorem 3.2.1, we want to utilize previous solutions in order to accelerate the computation of Pareto sets under slight variations of the parameter $\boldsymbol{p}$. Following standard approaches for numerical continuation methods (see also [AG03]), this is realized by a predictor corrector (PC) method. Based on already known solutions, a predictor step is performed, serving as an initial guess for the consecutive corrector step. Whereas in standard PC methods a single point is sought which satisfies $\boldsymbol{H} = \boldsymbol{0}$, we here have to compute the entire set of substationary points in the corrector step (cf. Algorithm 3.3). This is realized by applying the recovering algorithm described in Section 2.1.5. Alternatively, if the solution is approximated by a finite number of points, one could perform a descent step (e.g. based on (QOP)) for each of these points until the first order conditions are satisfied. Note however that in this situation, proper care has to be taken to avoid clustering and obtain a satisfactory approximation of $\mathcal{P}_{S,\mathsf{sub}}(\boldsymbol{p})$ by these points.

Algorithm 3.3 (Predictor-Corrector Method for Parameter Dependent MOPs)

1: Solve problem (3.8) with $\boldsymbol{p}^{(1)}$ and set the initial guess for the next problem, i.e. $\mathcal{P}^{\mathsf{init}}_{S,\mathsf{sub}}(\boldsymbol{p}^{(2)}) = \mathcal{P}_{S,\mathsf{sub}}(\boldsymbol{p}^{(1)})$
2: **for** $s = 2, 3 \ldots$ **do**
3: Corrector step s: Solve problem (3.8) with $\boldsymbol{p}^{(s)}$ using the recovering algorithm (Section 2.1.5) and the initial guess $\mathcal{P}^{\mathsf{init}}_{S,\mathsf{sub}}(\boldsymbol{p}^{(s)})$
4: Predictor step $s+1$: Compute the next initial guess $\mathcal{P}^{\mathsf{init}}_{S,\mathsf{sub}}(\boldsymbol{p}^{(s+1)})$ using Algorithm 3.4
5: **end for**

The predictor step consists of computing the tangent space of the object one wants to approximate and then selecting a direction within the tangent space and a step length, e.g. based on the curvature. Here, the tangent space of $\mathcal{P}_{S,\mathsf{sub}}(\boldsymbol{p})$ is approximated using numerical data. To be more precise, both the current and the previous sets of substationary points are first approximated by a finite number of points. Note that depending on the algorithm used for solving the MOP, the set is

already approximated this way in many cases. When applying set-oriented methods, the box centers can be used for this purpose. Then a predictor step is performed point-wise, based on two points from the respective sets that are closest to each other (cf. Algorithm 3.4). For the linearization to be valid, the distance between two parameter values s and $s-1$ has to be small, i.e. $\|\boldsymbol{p}^{(s)} - \boldsymbol{p}^{(s-1)}\| < \epsilon$ for some small $\epsilon > 0$.

Algorithm 3.4 (Predictor step for Parameter Dependent MOPs)

Require: Two solutions of problem (3.8), i.e. $\mathcal{P}_{S,\mathsf{sub}}(\boldsymbol{p}^{(s-1)})$ and $\mathcal{P}_{S,\mathsf{sub}}(\boldsymbol{p}^{(s)})$ corresponding to $\boldsymbol{p}^{(s-1)}$ and $\boldsymbol{p}^{(s)}$, respectively; parameter $h_1 > 0$

1: Approximate the two solution sets by a finite number of points $\{\boldsymbol{u}_{j_1}^{(s-1)}\}_{j_1=1}^{n_1}$ and $\{\boldsymbol{u}_{j_2}^{(s)}\}_{j_2=1}^{n_2}$

2: **for** $j_2 = 1, \ldots, n_2$ **do**

3: Find the point from the set $\{\boldsymbol{u}_{j_1}^{(s-1)}\}_{j_1=1}^{n_1}$ which is closest to $\boldsymbol{u}_{j_2}^{(s)}$:

$$j_{\mathsf{min}} = \operatorname*{arg\,min}_{j_1 \in \{1,\ldots,n_1\}} \|\boldsymbol{u}_{j_2}^{(s)} - \boldsymbol{u}_{j_1}^{(s)}\|_2$$

4: Compute the point

$$\boldsymbol{u}_{j_2}^{(s+1)} = \boldsymbol{u}_{j_2}^{(s)} + h\left(\boldsymbol{u}_{j_2}^{(s)} - \boldsymbol{u}_{j_{\mathsf{min}}}^{(s-1)}\right)$$

via linear extrapolation, where the step length is determined by

$$h = h_1 \frac{\|\boldsymbol{p}^{(s+1)} - \boldsymbol{p}^{(s)}\|_2}{\|\boldsymbol{p}^{(s)} - \boldsymbol{p}^{(s-1)}\|_2}$$

5: **end for**

6: Construct the initial guess $\mathcal{P}_{S,\mathsf{sub}}^{\mathsf{init}}(\boldsymbol{p}^{(s+1)})$ using the set of points $\{\boldsymbol{u}_j^{(s+1)}\}_{j=1}^{n_2}$

The procedure can be interpreted as a set-valued finite difference approach. Due to this, we can only compute one direction in the tangent space of $\mathcal{P}_{S,\mathsf{sub}}(\boldsymbol{p})$, meaning that continuation can be performed along a straight line within the parameter space:

$$\frac{\boldsymbol{p}^{(j+1)} - \boldsymbol{p}^{(j)}}{\|\boldsymbol{p}^{(j+1)} - \boldsymbol{p}^{(j)}\|_2} = \frac{\boldsymbol{p}^{(j)} - \boldsymbol{p}^{(j-1)}}{\|\boldsymbol{p}^{(j)} - \boldsymbol{p}^{(j-1)}\|_2} \qquad \text{for } j = 2, 3, \ldots$$

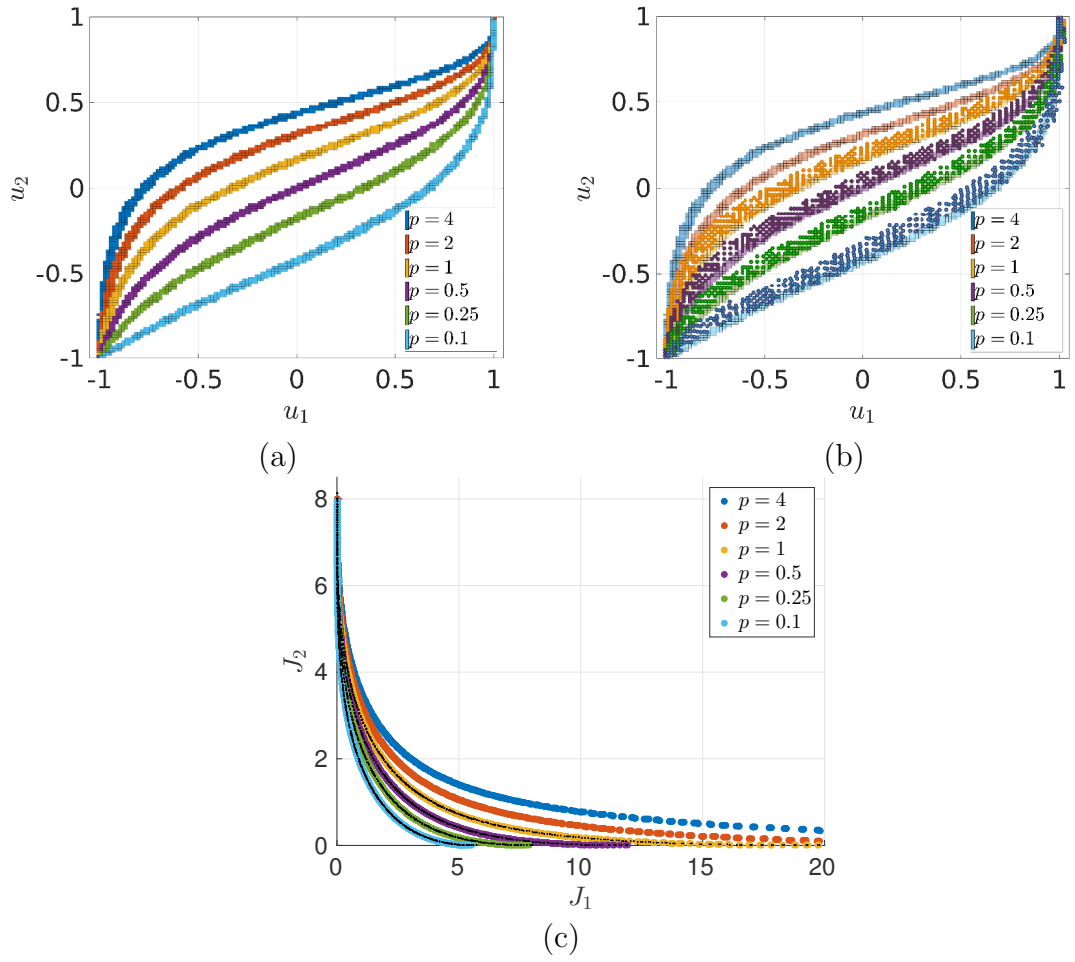


Figure 3.10: (a) The Pareto sets of example (3.12) for varying parameter values p. (b) Predictor steps computed by Algorithm 3.4 with $h_1 = 2$. The predicted points are plotted as circles, the solution after application of the corrector step is shown as a box covering in the same color. (c) The Pareto fronts corresponding to (a) and the image of the points predicted by Algorithm 3.4 (black dots).

Numerical Examples

In order to numerically analyze the behavior of the predictor step, let us first consider an academic example with $\boldsymbol{J} : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}^2$:

$$\min_{\boldsymbol{u} \in \mathbb{R}^2} \boldsymbol{J}(\boldsymbol{u}, p) = \min_{\boldsymbol{u} \in \mathbb{R}^2} \begin{pmatrix} (u_1 - 1)^2 + p(u_2 - 1)^4 \\ (u_1 + 1)^2 + (u_2 + 1)^2 \end{pmatrix}. \tag{3.12}$$

The solution to (3.12), computed with the sampling algorithm presented in Section 2.1.5, is shown in Figure 3.10 (a) for varying parameter values. Figure 3.10 (b) shows the points computed in the predictor step (Algorithm 3.4) for comparison. For this example, Theorem 3.2.1 applies and the Pareto set is globally a manifold. Consequently, the predictor step yields a set of points which lies very close to the corresponding Pareto set (see also Figure 3.10 (c) for a comparison in image space) and convergence of the corrector step is obtained after very few iterations.

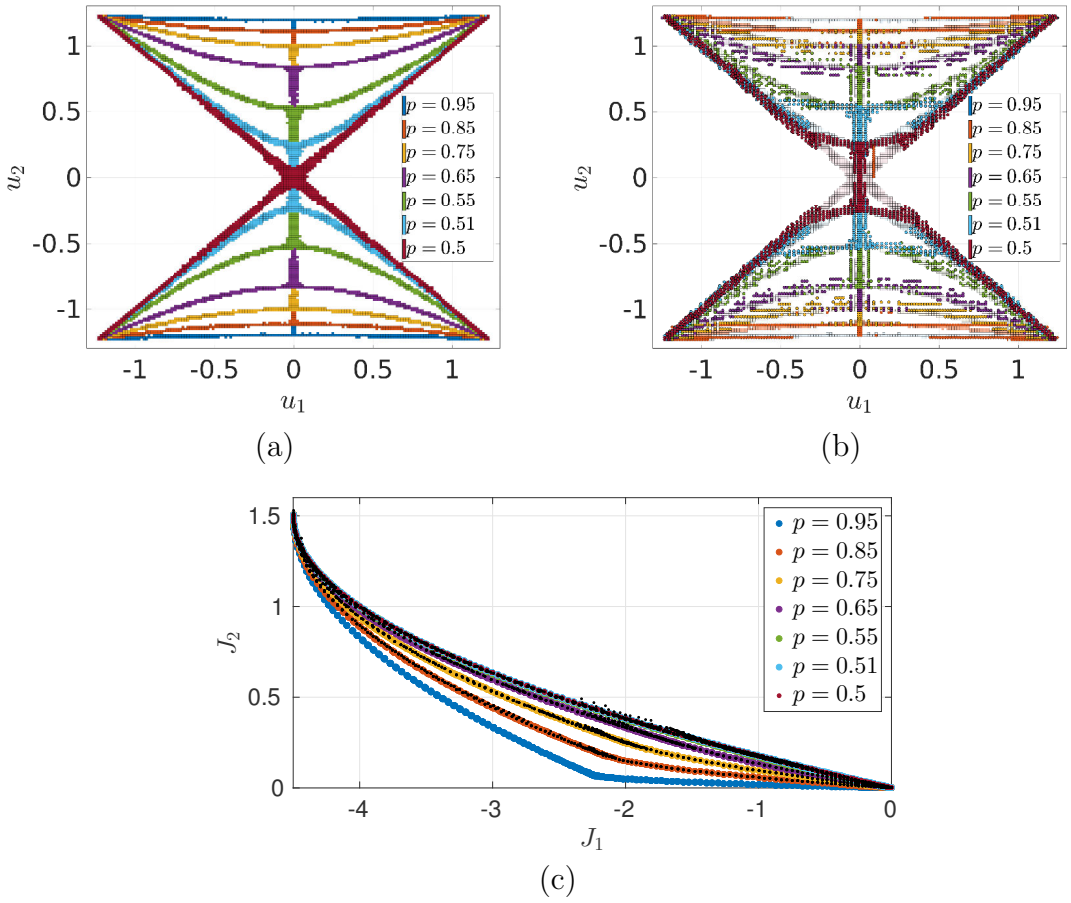


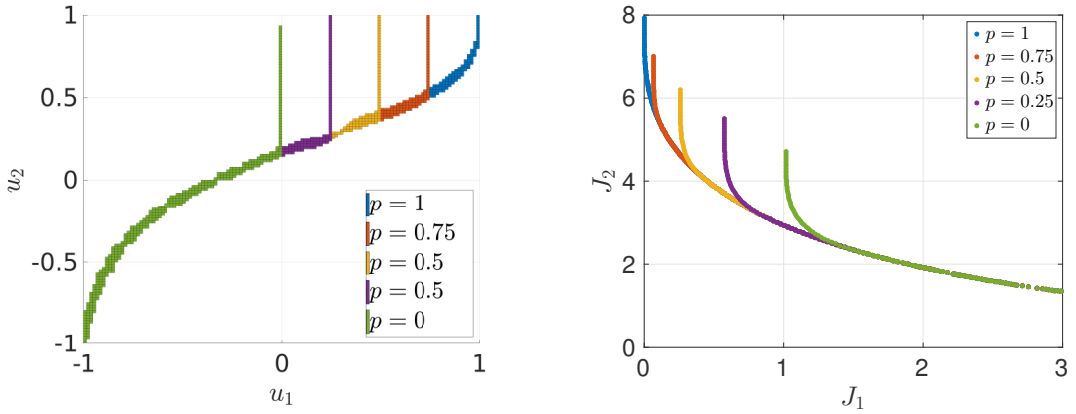
Figure 3.11: The Pareto set and Pareto front of example (3.13) for varying parameter values p.

Next, we again consider an objective function $\boldsymbol{J} : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}^2$:

$$\min_{\boldsymbol{u} \in \mathbb{R}^2} \boldsymbol{J}(\boldsymbol{u}, p) = \min_{\boldsymbol{u} \in \mathbb{R}^2} \begin{pmatrix} u_1^4 + u_2^4 - 3(u_1^2 + u_2^2) \\ p u_1^2 + (1-p) u_2^2 \end{pmatrix}. \tag{3.13}$$

On the symmetry axis $u_1 = 0$, the corresponding sets of substationary points are no longer globally manifolds and Theorem 3.2.1 does not apply. However, the continu-

ity with respect to the parameter is not influenced by this. The results are shown in Figure 3.11, where the predictor steps again show good agreement with the corresponding sets of substationary points. However, there are predictor points near the line $u_1 = 0$ which are relatively far away from the desired set. The reason for this lies in the violation of the manifold requirements and the corresponding non-smoothness in the set. The distance between the two closest points is measured in the 2-norm in Algorithm 3.4 which does not take into account the geometry of the Pareto set. The same effect may occur if the set locally possesses a strong curvature such that two points are identified as closest which they are not when considering a metric correctly accounting for the geometry. Nevertheless, since the corrector step converges locally to the Pareto set $\mathcal{P}_S(\boldsymbol{p})$, these far-off predictor points did not cause problems in the example considered here.



(a) (b)

Figure 3.12: (a) The Pareto sets of example (3.14) for varying parameter values p. (b) The corresponding Pareto fronts.

Remark 3.2.3. *In many applications, the constraints may be parameter dependent as well. In this situation, we do not need to recompute the entire Pareto set when varying* $\boldsymbol{p}$*. Starting with the* least constrained *solution, parts of the Pareto set rendered infeasible when tightening the constraints. However, the remainder of the Pareto set is still optimal for the new value of* $\boldsymbol{p}$*. Hence, we can keep this part and use continuation to recover the Pareto set of the new problem. If this set is locally a manifold, then the newly computed parts can only be found at the ends – where the manifold requirements are violated – or they are not connected to the already known part. In the first case, parameter variations in the constrains can be treated efficiently since the starting points for the recovering can easily be identified.*

Consider for example the problem

$$\begin{aligned} \min_{\boldsymbol{u}\in\mathbb{R}^2} \boldsymbol{J}(\boldsymbol{u}) &= \min_{\boldsymbol{u}\in\mathbb{R}^2} \begin{pmatrix} (u_1-1)^2 + (u_2-1)^4 \\ (u_1+1)^2 + (u_2+1)^2 \end{pmatrix} \\ s.t. \qquad & u_1 \leq p. \end{aligned} \tag{3.14}$$

The results for varying parameter values p are shown in Figure 3.12 where, beginning with $p = 1$, we only need to recompute the vertical parts of the solutions with $p < 1$. This will be utilized in the newly developed multiobjective ϵ-constraint method presented in Section 4.2.

3.3 Application to Autonomous Driving

In this section the PC method developed in the previous section is validated using another example from autonomous driving. Here, we are interested in optimally determining the steering angle for a vehicle with respect to secure and fast driving. To this end, we consider the well-known *bicycle model* [TL90, PLM06]. In this model, the dynamics of the vehicle is approximated by representing the two wheels on each axis by one wheel on the centerline (cf. Figure 3.13). When assuming a constant longitudinal velocity, this leads to a non-linear system of five coupled ODEs:

$$\begin{aligned} \dot{\boldsymbol{y}}(t) &= \begin{pmatrix} \dot{x_1}(t) \\ \dot{x_2}(t) \\ \dot{\Theta}(t) \\ \dot{v_y}(t) \\ \dot{r}(t) \end{pmatrix} = \begin{pmatrix} v_x(t)\cos(\Theta(t)) - v_y(t)\sin(\Theta(t)) \\ v_x(t)\sin(\Theta(t)) + v_y(t)\cos(\Theta(t)) \\ r \\ C_1(t)v_y(t) + C_2(t)r(t) + C_3(t)u(t), \\ C_4(t)v_y(t) + C_5(t)r(t) + C_6(t)u(t) \end{pmatrix}, \ t \in (t_0, t_e], \\ \boldsymbol{y}(t_0) &= \boldsymbol{y}_0, \end{aligned} \tag{3.15}$$

where $\boldsymbol{y} = (x_1, x_2, \Theta, v_y, r)^\top$ is the state consisting of the position $\boldsymbol{x} = (x_1, x_2)$, the angle Θ between the horizontal axis and the longitudinal vehicle axis, the lateral velocity v_y and the yaw rate r (cf. Figure 3.13). The vehicle is controlled by the

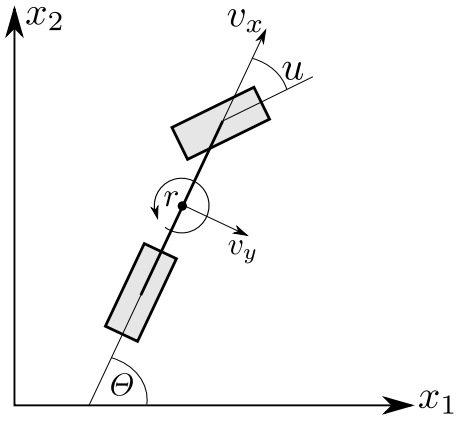


Figure 3.13: Bicycle model for the approximation of the vehicle dynamics.

front wheel angle u and the variables

$$
\begin{aligned}
C_1(t) &= \frac{C_{\alpha,f}\cos(u(t)) + C_{\alpha,f}}{m v_x(t)},\\
C_2(t) &= \frac{-L_f C_{\alpha,f}\cos(u(t)) + L_r C_{\alpha,f}}{I_z v_x(t)},\\
C_3(t) &= \frac{C_{\alpha,f}\cos(u(t))}{m}\\
C_4(t) &= \frac{-L_f C_{\alpha,f}\cos(u(t)) + L_r C_{\alpha,f}}{m v_x(t)} - v_x(t),\\
C_5(t) &= -\frac{L_f^2 C_{\alpha,f}\cos(u(t)) + L_r^2 C_{\alpha,f}}{I_z v_x(t)},\\
C_6(t) &= \frac{L_f C_{\alpha,f}\cos(u(t))}{I_z},
\end{aligned}
$$

have been introduced for abbreviation. The constants therein describe the vehicle's geometry, mass as well as tyre properties, see Table 3.1.

Table 3.1: Physical constants of the vehicle model.

Variable	Physical property	Numerical value
$C_{\alpha,f}$	Cornering stiffness coefficient (front)	65100
$C_{\alpha,r}$	Cornering stiffness coefficient (rear)	54100
L_f	Distance front wheel to center of mass	1
L_r	Distance rear wheel to center of mass	1.45
m	Vehicle mass	1275
I_z	Moment of inertia	1627

We consider the scenario of driving around a curve (cf. Figure 3.14 (e)), where the two concurrent objectives are to drive as far as possible within a given time $t_e - t_0$ (maximize speed) and to remain close to the middle of the road (maximize security). We set $t_0 = 0$ and $t_e = 10$ and in order to reduce the computational effort, the control u is approximated by five equidistant break points $(u(0) = u_1^d, \dots, u(10) = u_5^d)$, between which we interpolate linearly. This leads to the following MOCP with $\boldsymbol{J} : \mathbb{R}^5 \times \mathbb{R}^2 \to \mathbb{R}^2$,

$$\begin{aligned} \min_{\boldsymbol{u}^d \in \mathbb{R}^5} \boldsymbol{J}(\boldsymbol{u}^d, \boldsymbol{p}) &= \begin{pmatrix} -\int_{t_0}^{t_e} \boldsymbol{v}(t) \cdot \boldsymbol{s}(\boldsymbol{x}(t)) \, \mathrm{d}t \\ \int_{t_0}^{t_e} d(\boldsymbol{x}(t), \mathcal{X}_{mid})^2 \, \mathrm{d}t \end{pmatrix} \\ \text{s.t.} \quad & (3.15) \\ & d(\boldsymbol{x}(t), \mathcal{X}_{mid}) \leq d_{\mathsf{max}}, && t \in (t_0, t_e], \\ & u_l \leq u(t) \leq u_u, && t \in (t_0, t_e], \\ & \boldsymbol{x}(t_0) = \boldsymbol{p}, \end{aligned} \tag{3.16}$$

where the first objective is the integral of the velocity $\boldsymbol{v} = (v_x, v_y)$ in street direction $\boldsymbol{s}(\boldsymbol{x}(t))$ (i.e. parallel to the centerline) and $d(\boldsymbol{x}(t), \mathcal{X}_{mid})$ is the distance between the vehicle position and the closest point on the centerline $\mathcal{X}_{mid}$ (denoted as a dashed line in Figure 3.14 (e)). By $\boldsymbol{J}(\boldsymbol{u}^d, \boldsymbol{p})$ we denote the reduced cost functional as introduced in Section 2.2.1 with additional parameter dependency. The parameter enters in the initial condition $\boldsymbol{y}_0$, where the initial position $\boldsymbol{x}(t_0)$ is varied.

Both objectives in (3.16) require the computation of the minimal distance between the current position $\boldsymbol{x}(t)$ and the centerline $\mathcal{X}_{mid}$ (In the first objective, this is necessary to determine the orientation $\boldsymbol{s}(\boldsymbol{x}(t))$ of the centerline). Hence, the model is not differentiable and Theorem 3.2.1 is not applicable. Nevertheless, one may assume that on a track without strong curvature, this does not result in convergence issues since the objectives show no discontinuities with respect to the control variable. This can be observed numerically, see Figure 3.14 (d), where the variation of three different Pareto optimal solutions depending on the initial position $x_1(t_0)$ is shown. The trajectories indicate that the assumption on the continuity of the MOCP with respect to the initial position is justified in this situation.

Pareto sets for varying initial positions are shown in Figure 3.14 (a) and in (b), they are compared to the predictor step computed with Algorithm 3.4. We see that the predictor points agree remarkably well with the actual Pareto set as do their images with the Pareto fronts (cf. Figure 3.14 (c)) such that convergence can be obtained faster than by solving each problem individually. Furthermore, using the predictor step as a stand-alone algorithm in a real-time setting allows for a quick prediction of Pareto sets which have not been computed beforehand.

We observe in Figure 3.14 (c) that the Pareto set possesses gaps close to the maximal driven distance. This is very likely due to the number of sampling points

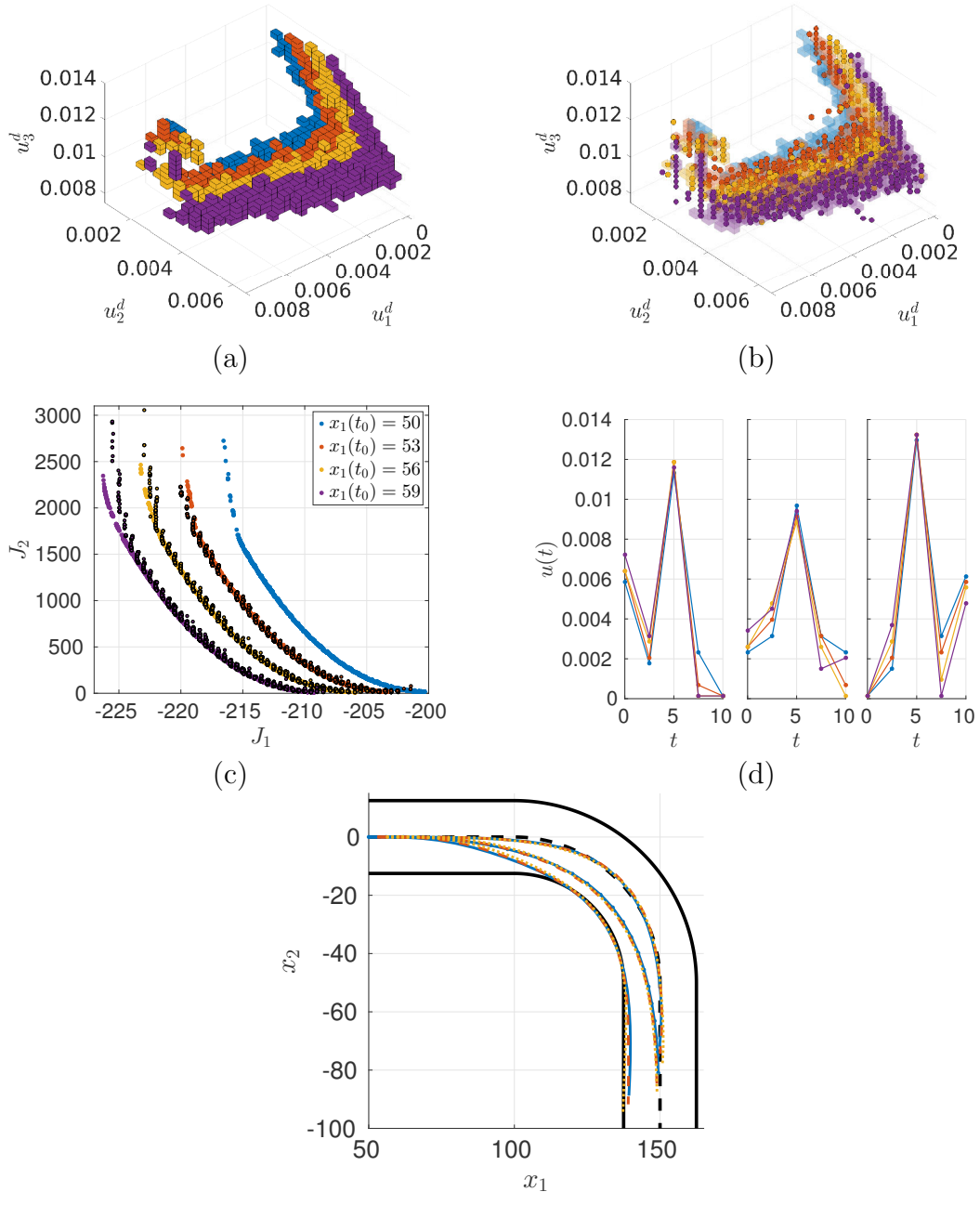
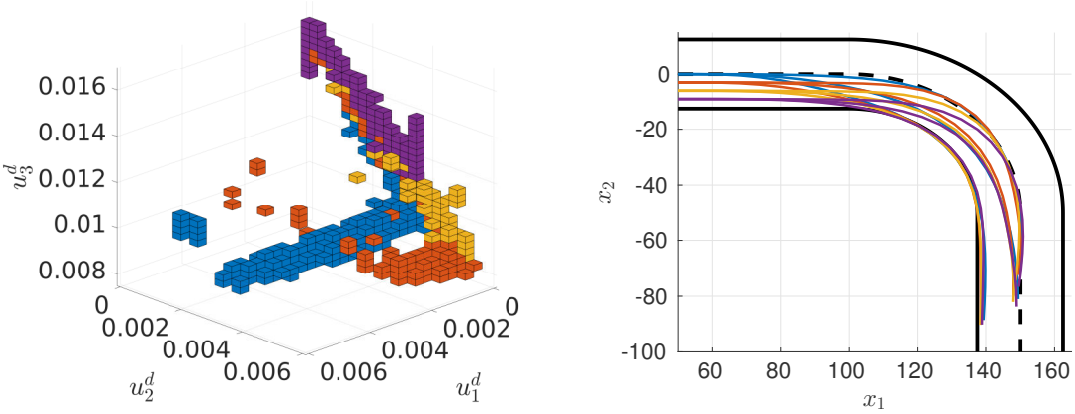


Figure 3.14: (a) The Pareto sets of (3.16) for varying initial positions $x_1(t_0)$ (Projection onto the first three variables (u_1^d, u_2^d, u_3^d)). (b) Points computed in the prediction step by Algorithm 3.4. (c) Pareto fronts corresponding to (a) and function values of the predictor points (black dots). (d) From left to right: Pareto points corresponding to the minimum of J_2, an intermediate solution and the minimum of J_1, respectively. The color corresponds to the parameter values in (a) – (c). (e) Vehicle trajectories corresponding to the controls shown in (d). The solution for $x_1(t_0) = 59$ is omitted for better visibility.



(a) (b)

Figure 3.15: (a) The Pareto sets of (3.16) for varying initial positions $x_2(t_0)$ (Projection onto the first three variables (u_1^d, u_2^d, u_3^d)). (b) Vehicle trajectories corresponding to three different optimal compromises (similar to Figure 3.14 (d)).

in the recovering algorithm. These solutions correspond to trajectories which are close to the boundary of the road (cf. Figure 3.14 (e)). This introduces numerical challenges for the set-oriented approach. When varying the initial position in x_2 direction, this effect is even more significant, see Figure 3.15.

Both in Figures 3.14 and 3.15, we see that some trajectories close to the border of the road violate the constraints at several points. The reason is that these trajectories correspond to the center points of the boxes (cf. Figure 3.14 (d)) which do not necessarily satisfy the constraints. Boxes are only discarded if every sample point within violates the constraints. Consequently, this issue can be resolved by refining the box covering or alternatively, by selecting a point within the box which is feasible.

4 Solving Many-Objective Optimization Problems via Subsets of Objectives

In this chapter the question how we can reduce the computational effort by exploiting the structure of Pareto sets in the presence of a large number of objectives is addressed. Many objectives are equally important in a wide range of applications. In the transportation example from the introduction, one wants to reach a destination as fast as possible while minimizing the energy consumption. Ideally, this should be achieved while providing an optimal comfort and maintaining maximal security at the same time.

The number of objectives considered in multiobjective optimization problems has grown. While two to three objectives were of interest in many applications in the beginning, this number has increased to four to twenty in recent years [FPL05]. This poses additional difficulties such that a new branch of research denoted as *many-objective optimization* has evolved, see e.g. [ITN08] for a short and [VBB14] for a more detailed review. The additional difficulties mainly stem from the curse of dimensionality. The Pareto set is typically a $(k-1)$-dimensional object (where k is the number of objectives) and hence, each additional objective increases the dimension by one so that the computational effort grows exponentially (see also [KL07, ITN08, SLC11]). Due to the complexity, computing the entire Pareto set of many-objective optimization problems has only been treated by evolutionary approaches until now, see e.g. [PF07, BZ11, YLLZ13]. Alternatively, there exist deterministic methods for interactively computing only those parts of the Pareto set which are of immediate interest [MS17, CLS16].

In order to reduce the complexity, one can try to reduce the number of objectives in the problem. Similar to the approach described in this chapter, such a reduction has been addressed in [SDT+13] in the context of evolutionary computation. The approach therein is to identify objectives of minor interest (e.g. via POD) and neglect them. In contrast to that, here the *hierarchical structure* of Pareto sets is exploited, meaning that considering a subset of objectives results in a subset of the Pareto set of the original problem. By this approach a *skeleton* of the desired set can be computed very efficiently by solving a small number of MOPs with a reduced number of objectives. This is particularly useful when function evaluations are expensive (see e.g. Chapter 5 for PDE-constrained problems) or when theoretical results rely on methods that are restricted to a small number of objectives. This is for

example the case in [BBV16, BBV17] where bicriterial MOPs constrained by PDEs are solved using reduced order modeling and error control. The remainder of the chapter is structured as follows. In Section 4.1, we will investigate the hierarchical structure of Pareto sets and prove that in the unconstrained case, the boundary of the set of substationary points is the union of the sets of substationary points of all subproblems with $k-1$ objectives. The results are utilized to develop a multiobjective version of the ϵ-constraint method in Section 4.2. Numerical examples are then presented in Sections 4.3 and 4.4.

Large parts of this chapter are also contained in [DPG] to which the author has made substantial contributions.

4.1 The Hierarchical Structure of Pareto Sets

In this chapter we will address both constrained and unconstrained MOPs as introduced in Chapter 2:

$$\min_{\boldsymbol{u}\in\mathcal{U}} \boldsymbol{J}(\boldsymbol{u}). \tag{MOP}$$

In order to exclude degenerate cases, the following rank assumption for the gradients of $\boldsymbol{J}$ is introduced:

Assumption 4.1.1. *Let $\boldsymbol{u}^*$ be a Pareto optimal point of (MOP). For every $k \geq 2$, the Jacobian matrix $\boldsymbol{J}'(\boldsymbol{u}^*)$, i.e.*

$$\boldsymbol{J}'(\boldsymbol{u}^*) = \begin{pmatrix} \nabla J_1(\boldsymbol{u}^*)^\top \\ \vdots \\ \nabla J_k(\boldsymbol{u}^*)^\top \end{pmatrix}, \tag{4.1}$$

satisfies the rank condition

$$\operatorname{rank}(\boldsymbol{J}'(\boldsymbol{u}^*)) = k-1.$$

Remark 4.1.2. *Note that this assumption is not restrictive. For a point $\boldsymbol{u}$ which is not substationary, $\boldsymbol{J}'(\boldsymbol{u})$ generally has full rank and a direction can be computed in which all objectives are descending (cf. (QOP) on p. 14). This can be repeated until a point $\boldsymbol{u}^*$ is found where – in the unconstrained case – the rank of $\boldsymbol{J}'(\boldsymbol{u}^*)$ is reduced according to the optimality condition (KKTu). The situation $\operatorname{rank}(\boldsymbol{J}'(\boldsymbol{u}^*)) < k-1$ can only occur if multiple gradients are linear dependent or equal to zero, i.e. if two objectives are non-conflicting. In this case, one can simply neglect one of these objectives and the above assumption is again satisfied for the reduced problem.*

Many solution approaches such as set-oriented methods can in theory also deal with a large number of objectives (i.e. $k \geq 4$) without any modifications to the algorithms. However, in practice the computational effort grows exponentially with the number of objectives. Consequently, it would be beneficial if we could gather information about the Pareto set by considering only subsets of objectives. In fact, we will start with the following simple observation:

Lemma 4.1.3. *Consider a constrained multiobjective optimization problem of the form (MOP) with k objectives where both the objectives and the constraints are at least once continuously differentiable. By considering only s objectives, where $1 \leq s < k$, we obtain (without loss of generality) the subproblem*

$$\min_{\boldsymbol{u} \in \mathcal{U}} \hat{\boldsymbol{J}}(\boldsymbol{u}) = \min_{\boldsymbol{u} \in \mathcal{U}} \begin{pmatrix} J_1(\boldsymbol{u}) \\ \vdots \\ J_s(\boldsymbol{u}) \end{pmatrix}, \tag{$\widehat{\text{MOP}}$}$$

for which the following holds:

(a) The set of substationary points $\widehat{\mathcal{P}}_{S,\mathsf{sub}}$ of ($\widehat{\text{MOP}}$) is a subset of the set of substationary points of (MOP), i.e. $\widehat{\mathcal{P}}_{S,\mathsf{sub}} \subseteq \mathcal{P}_{S,\mathsf{sub}}$,

(b) The Pareto set $\widehat{\mathcal{P}}_S$ of ($\widehat{\text{MOP}}$) is a subset of the Pareto set of (MOP), i.e. $\widehat{\mathcal{P}}_S \subseteq \mathcal{P}_S$.

Proof. (a) A substationary point $\boldsymbol{u}^*$ of ($\widehat{\text{MOP}}$) has to satisfy condition (KKT). Hence, there exist weights $\boldsymbol{\alpha}^* \in \mathbb{R}^s$ with $\boldsymbol{\alpha}^* \geq_p \mathbf{0}$ and $\sum_{i=1}^s \alpha_i^* = 1$ such that

$$\begin{aligned} \sum_{i=1}^{s} \alpha_i^* \nabla J_i(\boldsymbol{u}^*) + \sum_{j=1}^{m} \gamma_j \nabla h_j(\boldsymbol{u}^*) + \sum_{t=1}^{l} \mu_t \nabla g_t(\boldsymbol{u}^*) = \mathbf{0}, \\ h_j(\boldsymbol{u}^*) = 0, \quad j = 1, \ldots, m, \\ g_t(\boldsymbol{u}^*) \leq 0, \quad t = 1, \ldots, l, \\ \mu_t g_t(\boldsymbol{u}^*) = 0, \quad t = 1, \ldots, l, \\ \mu_t \geq 0, \quad t = 1, \ldots, l. \end{aligned}$$

Consequently, when considering additional objectives, the corresponding weights can be set to zero such that (KKT) is also satisfied for the problem with an increased number of objectives.
Part (b) is a simple consequence of the non-dominance property (cf. Definition 2.1.5). The Pareto set of ($\widehat{\text{MOP}}$) consists of all points which are non-dominated with respect to the objectives 1 to s. Adding additional objectives will hence not affect the non-dominance property of these points but only enlarge the Pareto set by points that are non-dominated with respect to the additional objectives. □

A result of Lemma 4.1.3 is that by solving a subproblem ($\widehat{\text{MOP}}$) of (MOP), we can compute a subset of the Pareto set. Due to the exponential increase in the computational complexity with respect to the number of objectives, these subproblems can be solved considerably faster while giving valuable insight into the original problem (MOP). In fact, it can be expected that solving a small number of subproblems is still faster than solving the original problem. Generally one can formulate $\binom{k}{s}$ subproblems which results in k problems for the case $s = k-1$ (i.e. when solving all problems where one objective is omitted) or in $\binom{k}{2} = \frac{1}{2}k(k-1)$ bi-objective problems (e.g. 10 problems for $k = 5$). Note that part (b) of the lemma is even valid for non-differentiable problems.

The above observation in itself can be very helpful in saving computational time. However, additional insight into the structure of the set $\mathcal{P}_{S,\mathsf{sub}}$ can be gained in the unconstrained case when the objective functions satisfy the additional rank condition from Assumption 4.1.1. In this situation, each subset $\widehat{\mathcal{P}}_{S,\mathsf{sub}} \subseteq \mathcal{P}_{S,\mathsf{sub}}$ is contained in the boundary of $\mathcal{P}_{S,\mathsf{sub}}$ which is stated in the following theorem.

Theorem 4.1.4. *Consider the general unconstrained multiobjective optimization problem* (MOP) *with* $\mathcal{U} = \mathbb{R}^n$*, and let all objectives be twice continuously differentiable.*

Let $\boldsymbol{u}^*$ *be a substationary point of the corresponding subproblem* ($\widehat{\text{MOP}}$) *with* $s = k-1$ *and assume that* $\boldsymbol{H}'$ *(with* $\boldsymbol{H}$ *according to* (4.4)*) has full rank in* $\boldsymbol{u}^*$ *such that* $\mathcal{M} = \boldsymbol{H}^{-1}(\boldsymbol{0})$ *is a* $(k-2)$*-dimensional differentiable submanifold of* $\mathbb{R}^{n+k}$ *in a neighborhood around* $(\boldsymbol{u}^*, \boldsymbol{\alpha}^*)$ *(according to Theorem 2.1.15, but with* $k-1$ *objectives). Furthermore, let Assumption 4.1.1 hold.*

Then $\boldsymbol{u}^*$ *is contained in the boundary* $\partial\mathcal{P}_{S,\mathsf{sub}}$ *of the set of substationary points* $\mathcal{P}_{S,\mathsf{sub}}$ *of* (MOP).

Proof. Since $\boldsymbol{u}^*$ is a substationary point of ($\widehat{\text{MOP}}$), the substationarity with respect to (MOP) follows directly from Lemma 4.1.3. Without loss of generality, assume that the k^{th} objective has been neglected. Denote by $\widehat{\mathcal{M}}_{\boldsymbol{u}^*} \subset \mathbb{R}^n$ the tangent space of $\widehat{\mathcal{P}}_{S,\mathsf{sub}}$ in $\boldsymbol{u}^*$. For every direction $\boldsymbol{v} \in \mathbb{R}^n \backslash \widehat{\mathcal{M}}_{\boldsymbol{u}^*}$ all objectives J_i, $i = 1, \ldots, s$, are increasing. If we now set $\boldsymbol{v} = \nabla J_k(\boldsymbol{u}^*)$, then J_k is also increased along $\boldsymbol{v}$. Due to the rank condition, $\boldsymbol{v} \notin \widehat{\mathcal{M}}_{\boldsymbol{u}^*}$. Hence, there is at least one direction $\boldsymbol{v}$ in which all objectives are increasing and $\boldsymbol{u}^*$ lies on the boundary of $\mathcal{P}_{S,\mathsf{sub}}$. □

The situation of Theorem 4.1.4 is visualized in Figure 4.1. While the optimality condition (KKTu) is not satisfied outside the set of substationary points (Figure 4.1 (a)), there exists a unique solution with $\boldsymbol{\alpha}^* >_p \boldsymbol{0}$ in the interior of $\mathcal{P}_{S,\mathsf{sub}}$ (Figure 4.1 (c)). All points on the boundary of $\mathcal{P}_{S,\mathsf{sub}}$ satisfy (KKTu) for the original problem as well as for a subproblem with $s < k$ (Figure 4.1 (b)). Due to the

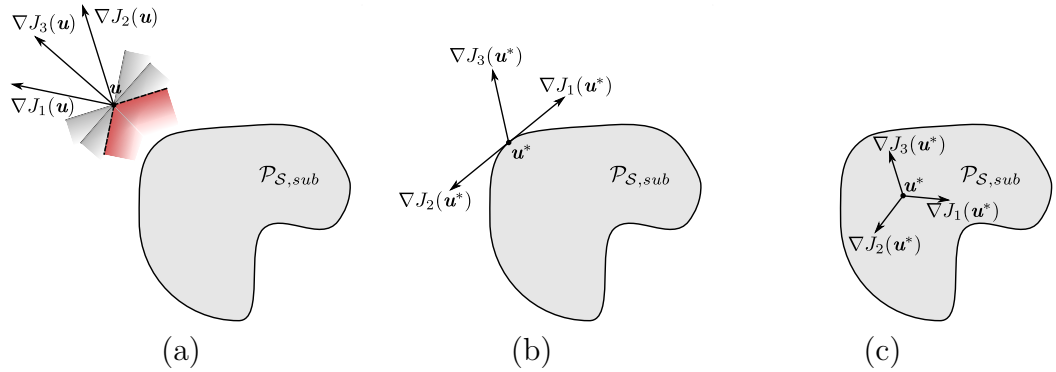


Figure 4.1: Sketch of an MOP with three objectives. (a) There exists no convex combination of the gradients satisfying (KKTu) outside $\mathcal{P}_{S,\mathsf{sub}}$. Instead, the union of the half spaces defined by the gradients defines the cone of descent directions for all objectives (shown in red). (b) On the boundary of $\mathcal{P}_{S,\mathsf{sub}}$, a subset of the gradients satisfies the KKT conditions, here for the objectives 1 and 2. (c) In the interior, there exists a convex combination of the gradients satisfying (KKTu) but here, all values of the corresponding weight vector $\boldsymbol{\alpha}^*$ are larger than zero.

rank condition, there exists a unique weight vector $\boldsymbol{\alpha}^* \in \mathbb{R}^k$ for each $\boldsymbol{u}^* \in \mathbb{R}^n$ such that

$$\boldsymbol{\alpha}^{*\top} \boldsymbol{J}'(\boldsymbol{u}^*) = \boldsymbol{\alpha}^{*\top} \begin{pmatrix} \nabla J_1(\boldsymbol{u}^*)^\top \\ \vdots \\ \nabla J_k(\boldsymbol{u}^*)^\top \end{pmatrix} = \mathbf{0}, \tag{4.2}$$

and we can deduce that at least one entry of $\boldsymbol{\alpha}^*$ has to be zero on the boundary of $\mathcal{P}_{S,\mathsf{sub}}$. For illustration, let us consider the following example:

Example 4.1.5. *Consider an MOP with* $\boldsymbol{J} : \mathbb{R}^2 \to \mathbb{R}^3$*:*

$$\min_{\boldsymbol{u} \in \mathbb{R}^2} \boldsymbol{J}(\boldsymbol{u}) = \min_{\boldsymbol{u} \in \mathbb{R}^2} \begin{pmatrix} -6u_1^2 + u_1^4 + 3u_2^2 \\ (u_1 - 0.5)^2 + 2(u_2 - 1)^2 \\ (u_1 - 1)^2 + 2(u_2 - 0.5)^2 \end{pmatrix}. \tag{4.3}$$

The norm of the descent direction $\boldsymbol{q}(\boldsymbol{u})$*, determined by solving the auxiliary problem* (QOP)*, is shown in Figure 4.2 along with the values for the weights* $\widehat{\boldsymbol{\alpha}}$*. Similar to Example 2.1.12 on p. 14, we observe continuous variations of* $\widehat{\boldsymbol{\alpha}}$ *within* $\mathcal{P}_{S,\mathsf{sub}}$ *(bounded by the white iso-lines) while outside jumps may occur. Moreover, one component of* $\widehat{\boldsymbol{\alpha}}$ *is always zero on the boundary of* $\mathcal{P}_{S,\mathsf{sub}}$ *such that these points are also substationary for the subproblem where only the objectives with non-zero weights* $\widehat{\alpha}_i$ *are considered.*

A further extension of Theorem 4.1.4 would be to show the other direction,

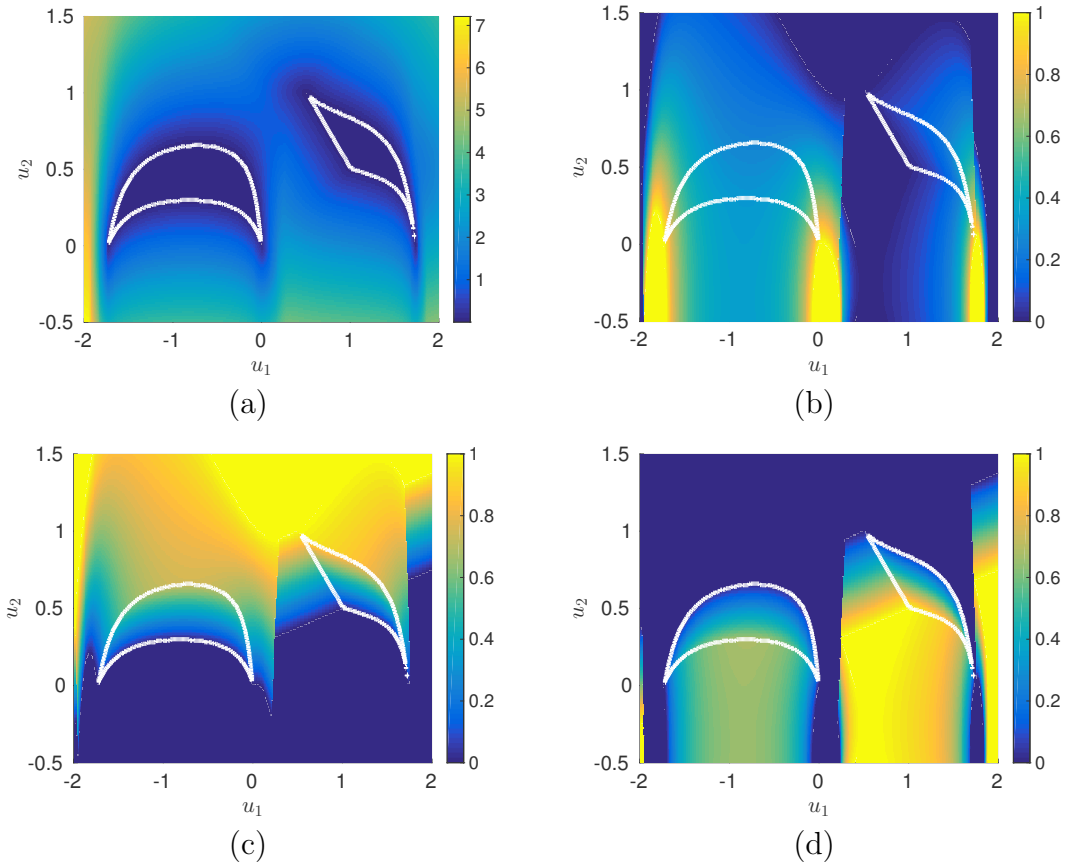


Figure 4.2: (a) Norm of the descent direction $\boldsymbol{q}(\boldsymbol{u})$, where the boundary of the set of substationary points is marked by the white iso-line. (b) – (d) The corresponding KKT weights $\widehat{\alpha}_1$, $\widehat{\alpha}_2$ and $\widehat{\alpha}_3$, respectively. On the boundary of $\mathcal{P}_{S,\mathsf{sub}}$ one component is always zero.

i.e. that for every substationary point of (MOP) which is located on the boundary, at least one component of $\boldsymbol{\alpha}^*$ has to be zero. This way, the entire boundary could be computed by taking the union of all subsets with $k-1$ objectives. This is difficult to prove since the boundary of $\mathcal{P}_{S,\mathsf{sub}}$ is in general not smooth (cf. Figure 4.5 on p. 91). However, when introducing additional assumptions, the desired result can be achieved. The more general case will be considered afterwards in Remark 4.1.8.

First we assume – according to the manifold conditions of Theorem 2.1.15 – that the objective functions are twice continuously differentiable and that the Jacobian matrix $\boldsymbol{H}'$ of

$$\boldsymbol{H}(\boldsymbol{u},\boldsymbol{\alpha}) = \begin{pmatrix} \sum_{i=1}^{k} \alpha_i \nabla J_i(\boldsymbol{u}) \\ \sum_{i=1}^{k} \alpha_i - 1 \end{pmatrix} \tag{4.4}$$

satisfies the full rank condition everywhere. For an optimal tuple $(\boldsymbol{u}^*, \boldsymbol{\alpha}^*)$ we have $\boldsymbol{H}(\boldsymbol{u}^*, \boldsymbol{\alpha}^*) = \mathbf{0}$ which is identical to (2.10) for the unconstrained case. Hence, the set of substationary points $\mathcal{M} = \boldsymbol{H}^{-1}(\mathbf{0})$ is a differentiable manifold. By the Implicit Function Theorem, Equation (4.4) defines a mapping

$$\boldsymbol{\Phi} : \Delta^{k-1} \to \mathcal{M}, \tag{4.5}$$

where Δ^{k-1} is the $k-1$-dimensional simplex excluding the boundary:

$$\Delta^{k-1} = \left\{ \boldsymbol{\alpha}^* \in \mathbb{R}^{k-1} \;\middle|\; \sum_{i=1}^{k-1} \alpha_i^* < 1,\ \boldsymbol{\alpha}^* >_p \mathbf{0} \right\},$$

and $\mathcal{M}$ is the manifold of substationary points. The simplex represents the first $k-1$ components of the weight vector $\boldsymbol{\alpha}^*$ which can be chosen freely. The last component is then defined as $\alpha_k^* = 1 - \sum_{i=1}^{k-1} \alpha_i^*$. Note that the components of $\boldsymbol{\alpha}^*$ are strictly greater than zero in Theorem 2.1.15 and hence, $\boldsymbol{\Phi}$ is restricted to the interior of the simplex. This is due to the fact that technical assumptions which make the Implicit Function Theorem applicable are violated, i.e. that an open neighborhood around $(\boldsymbol{u}^*, \boldsymbol{\alpha}^*)$ has to exist. However, if we additionally assume that $\boldsymbol{\Phi}$ is a diffeomorphism, the mapping can be extended to the closure

$$\overline{\Delta^{k-1}} = \left\{ \boldsymbol{\alpha}^* \in \mathbb{R}^{k-1} \;\middle|\; \sum_{i=1}^{k-1} \alpha_i^* \leq 1,\ \boldsymbol{\alpha}^* \geq_p \mathbf{0} \right\},$$

which will be shown in the following lemma.

Lemma 4.1.6. *Consider the general unconstrained multiobjective optimization problem (MOP) with $\mathcal{U} = \mathbb{R}^n$ and let $\boldsymbol{J}$ be twice continuously differentiable. Assume that $\boldsymbol{H}'$ (with $\boldsymbol{H}$ according to (4.4)) has full rank everywhere such that $\mathcal{M} = \boldsymbol{H}^{-1}(\mathbf{0})$ is a $(k-1)$-dimensional differentiable submanifold of $\mathbb{R}^{n+k}$ according to Theorem 2.1.15. Moreover, assume that the mapping $\boldsymbol{\Phi} : \Delta^{k-1} \to \mathcal{M}$ (Equation (4.5)) is a diffeomorphism.*

Then $\boldsymbol{\Phi}$ can be continuously extended to a homeomorphism on the closure $\overline{\boldsymbol{\Phi}} : \overline{\Delta^{k-1}} \to \overline{\mathcal{M}}$.

Proof. We have to show that $\overline{\boldsymbol{\Phi}}$ is bijective and that both $\overline{\boldsymbol{\Phi}}$ and $\overline{\boldsymbol{\Phi}^{-1}}$ are continuous. Since $\boldsymbol{\Phi}$ is a diffeomorphism, the inverse $\boldsymbol{\Phi}^{-1}$ is differentiable which implies uniform continuity. The simplex $\overline{\Delta^{k-1}}$ is complete and hence, according to [DMP03, Proposition 1.5.10], there exists a unique continuous extension $\overline{\boldsymbol{\Phi}^{-1}}$ of $\boldsymbol{\Phi}^{-1}$ to the closure $\overline{\mathcal{M}}$ of the manifold $\mathcal{M}$: For every $\boldsymbol{\beta} \in \overline{\mathcal{M}}$ there exists a Cauchy sequence $(\boldsymbol{\beta}^{(n)})$ with $\boldsymbol{\beta}^{(n)} \in \mathcal{M}$, $n = 1, 2, \ldots$, and $\lim_{n\to\infty}(\boldsymbol{\beta}^{(n)}) \to \boldsymbol{\beta}$. By uniform continuity,

$(\boldsymbol{\Phi}^{-1}(\boldsymbol{\beta}^{(n)}))$ is also a Cauchy sequence which results in

$$\boldsymbol{\Phi}^{-1}(\lim_{n\to\infty}(\boldsymbol{\beta}^{(n)})) = \lim_{n\to\infty}(\boldsymbol{\Phi}^{-1}(\boldsymbol{\beta}^{(n)})) =: \overline{\boldsymbol{\Phi}^{-1}}(\boldsymbol{\beta}).$$

From this we obtain completeness of $\overline{\mathcal{M}}$ and consequently, $\boldsymbol{\Phi}$ can be extended to $\overline{\boldsymbol{\Phi}}$ in the same way as $\boldsymbol{\Phi}^{-1}$ to $\overline{\boldsymbol{\Phi}^{-1}}$: For every $\boldsymbol{\alpha}^* \in \overline{\Delta^{k-1}}$ there exists a Cauchy sequence $(\boldsymbol{\alpha}^{(n)})$ with $\boldsymbol{\alpha}^{(n)} \in \Delta^{k-1}$, $n = 1, 2, \ldots$, and $\lim_{n\to\infty}(\boldsymbol{\alpha}^{(n)}) \to \boldsymbol{\alpha}^*$ which results in

$$\boldsymbol{\Phi}(\lim_{n\to\infty}(\boldsymbol{\alpha}^{(n)})) = \lim_{n\to\infty}(\boldsymbol{\Phi}(\boldsymbol{\alpha}^{(n)})) =: \overline{\boldsymbol{\Phi}}(\boldsymbol{\alpha}^*).$$

We have $\boldsymbol{\Phi}^{-1}(\boldsymbol{\Phi}(\boldsymbol{\alpha}^*)) = \mathbb{1}\ \forall\, \boldsymbol{\alpha}^* \in \Delta^{k-1}$ so that it remains to show that $\overline{\boldsymbol{\Phi}}$ is bijective on the boundary as well. This again follows from uniform continuity:

$$\begin{aligned}\overline{\boldsymbol{\Phi}^{-1}}(\overline{\boldsymbol{\Phi}}(\boldsymbol{\alpha}^*)) &= \overline{\boldsymbol{\Phi}^{-1}}(\boldsymbol{\Phi}(\lim_{n\to\infty}(\boldsymbol{\alpha}^{(n)})))\\ &= \overline{\boldsymbol{\Phi}^{-1}}(\lim_{n\to\infty}(\boldsymbol{\Phi}(\boldsymbol{\alpha}^{(n)}))) = \lim_{n\to\infty}(\boldsymbol{\Phi}^{-1}(\boldsymbol{\Phi}(\boldsymbol{\alpha}^{(n)}))) = \mathbb{1}.\end{aligned}$$

□

Using Lemma 4.1.6, we can now state the main result of this chapter, i.e. that the set of substationary points is bounded by the union of the sets of substationary points of all subproblems with $k-1$ objectives:

Theorem 4.1.7. *Consider the general unconstrained multiobjective optimization problem (MOP) with $\mathcal{U} = \mathbb{R}^n$ and let $\boldsymbol{J}$ be twice continuously differentiable. Assume that $\boldsymbol{H}'$ (with $\boldsymbol{H}$ according to (4.4)) has full rank everywhere such that $\mathcal{M} = \boldsymbol{H}^{-1}(\mathbf{0})$ is a $(k-1)$-dimensional differentiable submanifold of $\mathbb{R}^{n+k}$ according to Theorem 2.1.15. Moreover, assume that the mapping $\boldsymbol{\Phi} : \Delta^{k-1} \to \mathcal{M}$ (Equation (4.5)) is a diffeomorphism.*

Then the k sets of substationary points of the corresponding subproblems ($\widehat{\text{MOP}}$) with $s = k-1$ form the boundary of the set of substationary points $\mathcal{P}_{S,\mathsf{sub}}$ of (MOP), i.e. $\bigcup_{i=1}^{k} \widehat{\mathcal{P}}_{S,\mathsf{sub},i} = \partial\mathcal{P}_{S,\mathsf{sub}}$.

Proof. The proof follows directly from Theorem 4.1.4 and Lemma 4.1.6. According to Theorem 4.1.4, the sets $\widehat{\mathcal{P}}_{S,\mathsf{sub},i}$ lie on the boundary of $\mathcal{P}_{S,\mathsf{sub}}$ and not in the interior. Consequently, the corresponding weight vectors $\boldsymbol{\alpha}^*$ lie on the boundary of the simplex $\overline{\Delta^{k-1}}$ (where at least one entry of $\boldsymbol{\alpha}^*$ is zero). Since $\overline{\boldsymbol{\Phi}}$ is a homeomorphism according to Lemma 4.1.6, the boundary of $\overline{\Delta^{k-1}}$ is mapped to the boundary of $\overline{\mathcal{M}}$. □

Remark 4.1.8. *The assumption on $\boldsymbol{\Phi}$ being a diffeomorphism is strong in the sense that this excludes Pareto sets where the Pareto front is non-convex and in particular disconnected Pareto sets. Since $\boldsymbol{\Phi}$ is bijective, each point $(\alpha_1^*, \ldots, \alpha_{k-1}^*)$ has to*

uniquely define a point $\boldsymbol{u}^*$ on $\mathcal{P}_{S,\mathsf{sub}}$. This does not hold in the non-convex setting (see e.g. Figure 2.4 on p. 20) whereas under certain conditions, $\mathcal{P}_{S,\mathsf{sub}}$ is a manifold nonetheless, cf. Remark 2.1.16 on p. 18. However, the result that $\mathcal{P}_{S,\mathsf{sub}}$ is bounded by the union of the subsets $\widehat{\mathcal{P}}_{S,\mathsf{sub},i}$ for $i = 1, \ldots, k$ can be observed in other cases as well, see e.g. Figure 4.2 in Example 4.1.5.

If $\mathcal{P}_{S,\mathsf{sub}}$ is a manifold, then so is the interior of the subsets $\widehat{\mathcal{P}}_{S,\mathsf{sub},i}$ with $k-1$ objectives. According to Theorem 4.1.4, these subsets lie on the boundary of $\mathcal{P}_{S,\mathsf{sub}}$. Hence, all points on the boundary are contained in $(k-2)$-dimensional manifolds except for those where more than one entry of $\boldsymbol{\alpha}^*$ is zero. However, these points are contained in a set of measure zero with respect to the boundary. Consequently, the following theorem holds almost everywhere on the boundary of $\mathcal{P}_{S,\mathsf{sub}}$. It can be interpreted as the "opposite direction" of Theorem 4.1.4. There, a solution of the subproblem ($\widehat{\text{MOP}}$) has to lie on the boundary of $\mathcal{P}_{S,\mathsf{sub}}$ whereas here, a solution of (MOP) which lies on the boundary has to have an associated weight $\boldsymbol{\alpha}^*$ where at least one entry is zero.

Theorem 4.1.9. *Consider the general unconstrained multiobjective optimization problem (MOP) with $\mathcal{U} = \mathbb{R}^n$ and let $\boldsymbol{J}$ be twice continuously differentiable. Assume that $\boldsymbol{H}'$ (with $\boldsymbol{H}$ according to (4.4)) has full rank everywhere such that $\mathcal{M} = \boldsymbol{H}^{-1}(\boldsymbol{0})$ is a $(k-1)$-dimensional differentiable submanifold of $\mathbb{R}^{n+k}$ according to Theorem 2.1.15.*

Let $\boldsymbol{u}^*$ be a substationary point of (MOP) which is contained in the boundary $\partial\mathcal{P}_{S,\mathsf{sub}}$ and assume that the boundary is a $(k-2)$-dimensional differentiable submanifold of $\mathbb{R}^{n+k}$ in a neighborhood around $(\boldsymbol{u}^*, \boldsymbol{\alpha}^*)$. Then at least one component of the corresponding weight vector $\boldsymbol{\alpha}^*$ is zero.

Proof. Since $\boldsymbol{u}^*$ is located on the boundary of $\mathcal{P}_{S,\mathsf{sub}}$, no gradient $\nabla \boldsymbol{J}_i(\boldsymbol{u}^*)$, $i = 1, \ldots, k$ can point to the interior of $\mathcal{P}_{S,\mathsf{sub}}$. Otherwise, there would exist a descent direction opposite the inwards pointing gradient which would lead to a decrease of the corresponding objective value. Hence, the point $\boldsymbol{u}^*$ would not be located on the boundary. Consequently, all gradients either have to lie in the tangent space $\widehat{\mathcal{M}}_{\boldsymbol{u}^*} \subset \mathbb{R}^n$ of $\partial\mathcal{P}_{S,\mathsf{sub}}$ (which is also $(k-2)$-dimensional) or point outwards of $\mathcal{P}_{S,\mathsf{sub}}$.

Due to the rank assumption 4.1.1, maximally $k-1$ gradients can lie in $\widehat{\mathcal{M}}_{\boldsymbol{u}^*}$. The substationarity of $\boldsymbol{u}^*$ implies that there exists a convex combination to zero when appropriately choosing the corresponding entries of $\boldsymbol{\alpha}^*$. Since no gradient is pointing inwards, there exists no convex combination which takes the outwards pointing gradients into account and yields zero at the same time. Consequently, the entries of $\boldsymbol{\alpha}^*$ corresponding to vectors not contained in $\widehat{\mathcal{M}}_{\boldsymbol{u}^*}$ have to be zero. □

Corollary 4.1.10. *The results of Theorems 4.1.7 and 4.1.9 also hold in the situation where the set of substationary points is disconnected if the assumptions are satisfied everywhere in the respective parts.*

Theorems 4.1.7 and 4.1.9 imply that the set of substationary points is indeed structured hierarchically. When neglecting one objective, the resulting set of substationary points lies on the boundary of the solution set to the original problem. This can be repeated until $s = 1$ and we obtain the scalar minima of $J_1, \ldots, J_k$. If we want to address an MOP with four conflicting objectives, for example, the set is a three-dimensional manifold. It is bounded by four two-dimensional surfaces, each of which is again bounded by three one-dimensional line segments. Finally each of these line segments is bounded by two scalar minima, see Figure 4.5 in Section 4.3 for an illustration.

The hierarchical structure can be utilized to efficiently solve many-objective optimization problems. As a starting point, the Pareto set can be characterized by its boundaries. If we are interested in computing interior points, these boundaries can be used in continuation approaches [Hil01, SWOBD13, MS17] or evolutionary strategies [CLV07] as well-spread and already substationary initial guesses. An alternative approach based on the ϵ-constraint method and directly utilizing the hierarchical structure is presented in the next section. Numerical examples illustrating the concept as well as the savings in computational time will be shown in Section 4.3.

4.2 A Multiobjective Extension of the ϵ-Constraint Method

The results from Section 4.1 imply that it is numerically beneficial to gather information about (MOP) by solving cheaper subproblems ($\widehat{\text{MOP}}$), and we will see in Section 4.3 that this results in a remarkable speed-up. However, when we are interested in Pareto optimal solutions in the interior of the set of substationary points, we have to make adjustments to the subproblems that we solve. The approach presented here is inspired by the well-known ϵ-*constraint method* (cf. Section 2.1.4), where the following scalar problem is obtained by transforming all but one objectives into constraints:

$$\begin{aligned} &\min_{\boldsymbol{u}\in\mathcal{U}} J_i(\boldsymbol{u}) \\ \text{s.t.} \quad &J_j(\boldsymbol{u}) \leq \epsilon_j, \quad j = 1, \ldots, k,\ j \neq i. \end{aligned} \tag{2.13}$$

This way, a (locally) Pareto optimal point is computed for every feasible constraint $\boldsymbol{\epsilon} \in \mathbb{R}^{k-1}$.

Following this idea, we reformulate (MOP) in a similar way. We set $s = k - 1$,

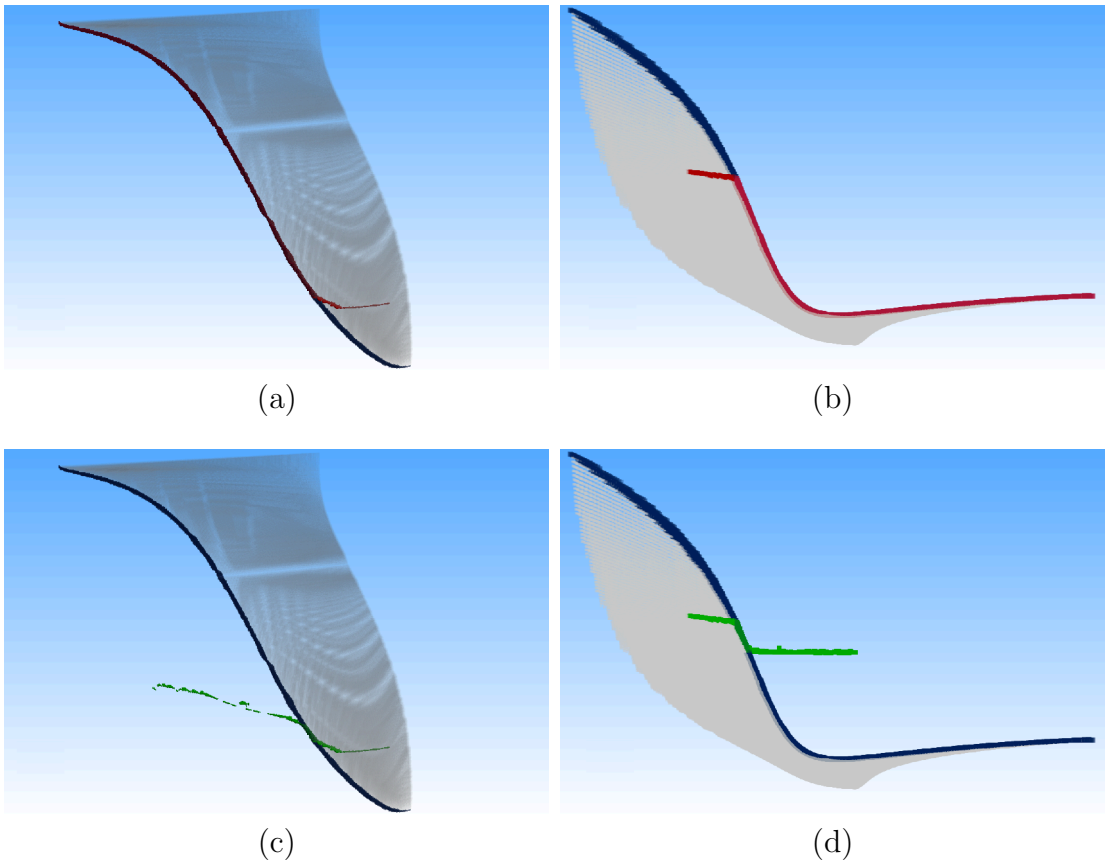


Figure 4.3: The Pareto set ((a) and (c)) and Pareto front ((b) and (d)) of example (4.8) approximated by the multiobjective ϵ-constraint method. The Pareto set and Pareto front of the original problem are shown in gray and for the two-dimensional subproblem with $\bar{\epsilon}_3 = \infty$ and $\underline{\epsilon}_3 = -\infty$ in blue. In (a) and (b), the solution to Problem (4.6) with $\bar{\epsilon}_3 = 12$ and $\underline{\epsilon}_3 = -\infty$ is shown in red. In (c) and (d), $\bar{\epsilon}_3 = 12$ and $\underline{\epsilon}_3 = 10$ which yields additional Pareto optimal points. However, these are infeasible due to the second constraint in Problem (4.7).

thereby neglecting one objective, and then introduce an additional constraint:

$$\begin{aligned} \min_{\boldsymbol{u}\in\mathcal{U}} \hat{\boldsymbol{J}}(\boldsymbol{u}) &= \min_{\boldsymbol{u}\in\mathcal{U}} \begin{pmatrix} J_1(\boldsymbol{u}) \\ \vdots \\ J_s(\boldsymbol{u}) \end{pmatrix} \\ \text{s.t.} \quad & J_k(\boldsymbol{u}) \leq \bar{\epsilon}_k, \end{aligned} \tag{4.6}$$

where $\hat{\boldsymbol{J}} : \mathcal{U} \to \mathbb{R}^{k-1}$ and $\bar{\epsilon}_k \in \{\mathbb{R} \cup \infty\}$ is an upper bound[1]. In the first computation,

[1]Note that since we are interested in interior points instead of points on the boundary in this

we set $\overline{\epsilon}_k = \infty$. The solution to (4.6) is thus a subset of the Pareto set according to Lemma 4.1.3. In the unconstrained case, this set lies on the boundary of the Pareto set if the conditions of Theorem 4.1.4 are satisfied. In the next iteration, we decrease $\overline{\epsilon}_k$ below the maximal value of J_k and hence compute another subset. Points with a lower value in J_k need to have increased values in at least one other objective and hence, they were dominated in the previous computation. By decreasing $\overline{\epsilon}_k$ in each loop, we can thus proceed and compute different subsets of the Pareto set in every iteration, see Algorithm 4.1 below.

However, this leads to the problem that we compute parts of the Pareto set multiple times, see Figure 4.3 (a) and (b), where the blue and red lines show solutions to (4.6) with different values for $\overline{\epsilon}_k$. By decreasing $\overline{\epsilon}_k$ parts of the previous solution are rendered infeasible and additional points are now Pareto optimal. However, as only parts of the previous solution are infeasible, the remaining part is computed twice. This can be circumvented in two different ways. On the one hand, we could start with a previous solution, remove all points that are infeasible due to the new constraint value and then proceed with a continuation algorithm (see also Remark 3.2.3 on p. 70). On the other hand, we can introduce a lower bound $\underline{\epsilon}_k$ and thereby ensure that the different Pareto sets are disjoint:

$$\begin{aligned} &\min_{\boldsymbol{u}\in\mathcal{U}} \hat{\boldsymbol{J}}(\boldsymbol{u}) = \min_{\boldsymbol{u}\in\mathcal{U}} \begin{pmatrix} J_1(\boldsymbol{u}) \\ \vdots \\ J_s(\boldsymbol{u}) \end{pmatrix} \\ \text{s.t.} \quad &\underline{\epsilon}_k \leq J_k(\boldsymbol{u}) \leq \overline{\epsilon}_k, \\ &\boldsymbol{u} \text{ satisfies the optimality conditions of (MOP).} \end{aligned} \tag{4.7}$$

Due to the lower bound $\underline{\epsilon}_k$, additional points can be optimal that are not part of the original Pareto set, see the green lines in Figure 4.3 (c) and (d). These have to be excluded from the solution by introducing the second constraint which can be realized by verifying the optimality condition of the full problem after the computation of the Pareto set. Both approaches are formalized in Algorithm 4.1, where alternative or different steps for the second approach are written in brackets.

Due to the exponential increase in computational effort with the number of objective functions and the quadratic scaling of non-dominance tests, this approach can help to significantly reduce the computing time. Moreover, a further increase in computational efficiency may be achieved by interpreting more than one objective as constraints. This increases the number of MOPs to be solved but reduces their respective complexity. For the case that $k-1$ objectives are treated as constraints, we obtain the classical ϵ-constraint method. Note that using this approach, one has to take into account that additional constraints can result in MOPs that are more

section, we can consider additional constraints, i.e. $\boldsymbol{u} \in \mathcal{U} \subset \mathbb{R}^n$.

Algorithm 4.1 (Multiobjective ϵ-Constraint Method)

Require: $\overline{\epsilon}_k, \underline{\epsilon}_k \in \mathbb{R}$, $\Delta\epsilon \in \mathbb{R}$, $i = 1$

1: Set $\overline{\epsilon}_k = \infty$ and $\underline{\epsilon}_k = -\infty$
2: Compute $\widehat{\mathcal{P}}^{(0)}_{S,\mathsf{sub}}$ by solving problem (4.6) (problem (4.7))
3: **while** $\widehat{\mathcal{P}}^{(i-1)}_{S,\mathsf{sub}} \neq \emptyset$ **do**
4: Identify maximum value $J_{k,\mathsf{max}}$ of J_k from $\widehat{\mathcal{P}}^{(i-1)}_{S,\mathsf{sub}}$
5: Set $\overline{\epsilon}_k = J_{k,\mathsf{max}} - \Delta\epsilon$
6: (Set $\underline{\epsilon}_k = \overline{\epsilon}_k - \Delta\epsilon$)
7: Compute $\widehat{\mathcal{P}}^{(i)}_{S,\mathsf{sub}}$ by solving problem (4.6) (problem (4.7))
8: Set $i = i + 1$
9: **end while**

difficult to solve as has been pointed out for the scalar ϵ-constraint method (see e.g. [Ehr05]).

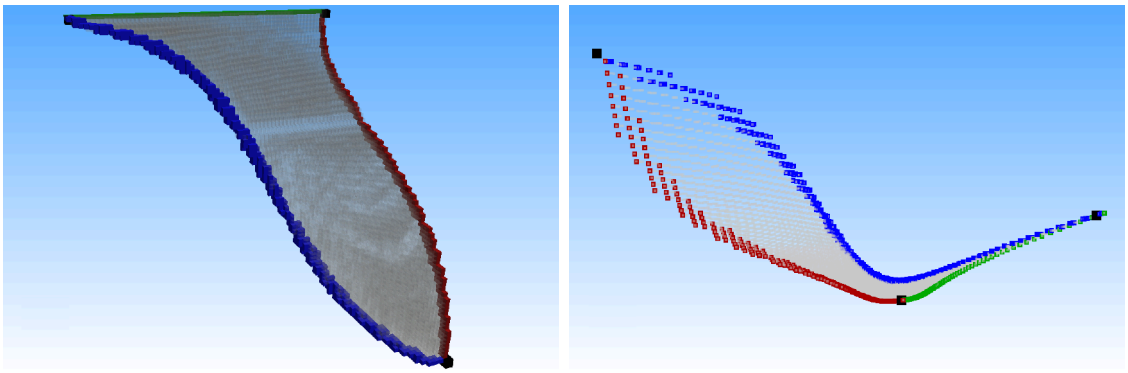
4.3 Numerical Examples

In order to demonstrate the significant decrease in computational effort, the concepts developed in Sections 4.1 and 4.2 will now be applied to several academic examples and one example from industry. All solutions have been computed using the sampling algorithm (Algorithm 2.3). In all computations the configuration of the algorithm (i.e. the initial box and the sampling) has not been changed within a problem. The increase in computational efficiency is therefore exclusively due to the reduced number of objectives in the subproblems. Moreover, it should be mentioned that the above considerations do not rely on the choice of a specific algorithm.

As a first example, let us consider the objective function $\boldsymbol{J} : \mathbb{R}^3 \to \mathbb{R}^3$:

$$\min_{\boldsymbol{u}\in\mathbb{R}^3} \boldsymbol{J}(\boldsymbol{u}) = \min_{\boldsymbol{u}\in\mathbb{R}^3} \begin{pmatrix} (u_1-1)^4 + (u_2-1)^2 + (u_3-1)^2 \\ (u_1+1)^2 + (u_2+1)^4 + (u_3+1)^2 \\ (u_1-1)^2 + (u_2+1)^2 + (u_3-1)^4 \end{pmatrix}. \tag{4.8}$$

The Pareto set and corresponding front of (4.8) are shown in Figure 4.4. The Pareto set is obviously structured hierarchically. The two-dimensional set of the original problem is bordered by three edges, the Pareto sets of the corresponding subproblems. These are again bordered by the respective minima of the two considered objectives. Solving all three two-dimensional subproblems in total requires only 25% of the time for solving the problem with 3 objectives.



(a) (b)

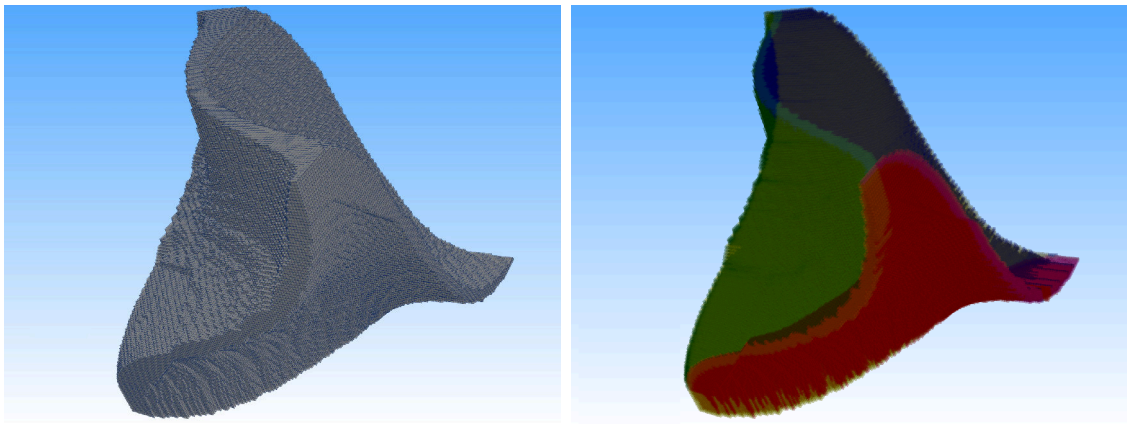
Figure 4.4: The Pareto set (a) and Pareto front (b) of example (4.8). The solution of the three-dimensional problem is shown in gray, the two-dimensional subproblems are colored in red, blue and green, respectively, and the scalar minima are black.

The second example is an MOP with four objectives, i.e. $\boldsymbol{J} : \mathbb{R}^3 \to \mathbb{R}^4$:

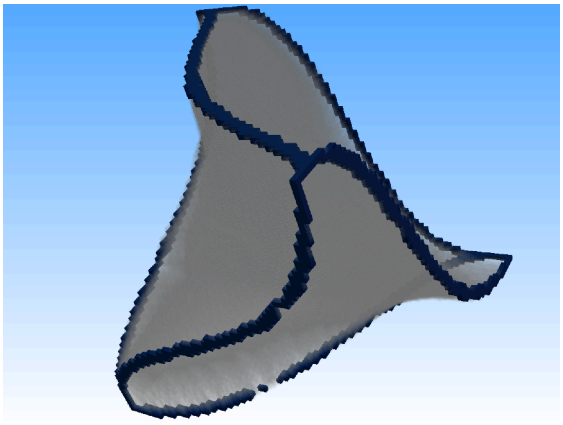
$$\min_{\boldsymbol{u}\in\mathbb{R}^3} \boldsymbol{J}(\boldsymbol{u}) = \min_{\boldsymbol{u}\in\mathbb{R}^3} \begin{pmatrix} (u_1-1)^4 + (u_2-1)^2 + (u_3-1)^2 \\ (u_1+1)^2 + (u_2+1)^4 + (u_3-1)^2 \\ (u_1-1)^2 + (u_2+1)^2 + (u_3+1)^4 \\ (u_1+1)^2 + (u_2-1)^4 + (u_3+1)^2 \end{pmatrix}. \tag{4.9}$$

The solution to this problem is shown in Figure 4.5, where again the hierarchical structure becomes visible. The three-dimensional Pareto set is bounded by four two-dimensional surfaces and these are again bounded by three edges each. The reduction of computational effort is even more significant for this problem. Solving the four three-objectives problems requires in total only approximately 2.3% of the time needed for solving the original problem whereas the six bi-objective problems require only approximately 0.04% of the time. This is a speed-up by a factor of over 2200.

If we want to exploit the hierarchical structure when computing interior points of the Pareto set, we can apply the multiobjective ϵ-constraint method presented in Section 4.2. We again consider example (4.8) and, according to Algorithm 4.1, set $\bar{\epsilon}_k = \infty$ and $\underline{\epsilon}_k = -\infty$. This leads to the computation of the Pareto set for the first two objectives, cf. Figures 4.3 and 4.6. The maximal value for J_3 is approximately 20 and thus, we consecutively solve problem (4.7) with $\bar{\epsilon}_k = 18, 16, 14, \ldots$ and $\underline{\epsilon}_k = 16, 14, 12, \ldots$, respectively. The result is shown in Figure 4.6, where the solutions to (4.7) with varying constraints are depicted in blue and the solution to the original Problem is shown in gray for comparison. Observe that the solution of the subproblems is dotted in some parts. This is due to numerical reasons and



(a) (b)

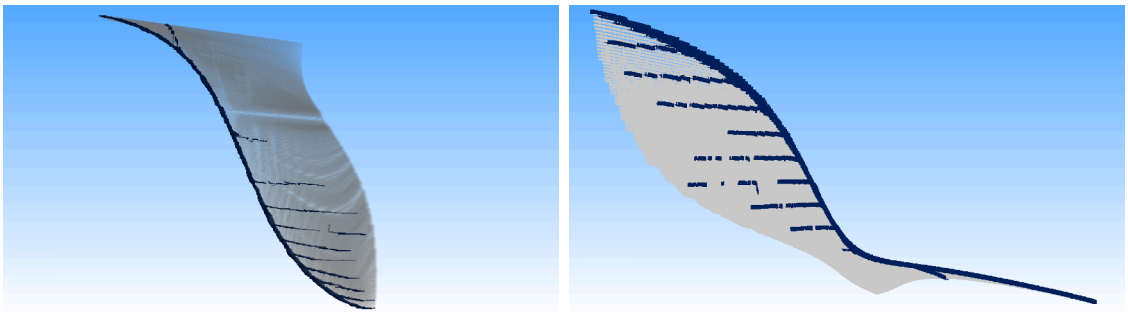


(c)

Figure 4.5: (a) The Pareto set of example (4.9). (b) The union of the Pareto sets of the four subproblems with three objectives (shown in different colors) forms the boundary of the original Pareto set. (c) The Pareto sets of the six subproblems with two objectives are shown in blue.

the box covering used in the sampling algorithm. In the subdivision procedure, it is often difficult to accurately capture the constraints with a reasonable number of sample points. This can result in the elimination of boxes in the earlier stages of the execution since relatively large boxes are identified as infeasible although this may not be true for all subsets of these boxes.

Let us now compare the efficiency of the two different problem formulations (4.6) and (4.7). In the first approach, solving ten 2D problems takes in total approximately 18.3% of the time required for solving the 3D problem. In the second approach, the time for solving the 2D problems is again reduced by a factor of roughly three, resulting in 6.2% of the time required for solving the 3D problem. However, in this case we additionally have to evaluate the optimality condition of



(a) (b)

Figure 4.6: (a) The Pareto set of example (4.8) approximated by the multiobjective ϵ-constraint method, i.e. by a sequence of subproblems (4.7) with varying constraint values $\underline{\epsilon}_3$ and $\bar{\epsilon}_3$. The solution to the original problem is shown in gray. (b) The corresponding Pareto front.

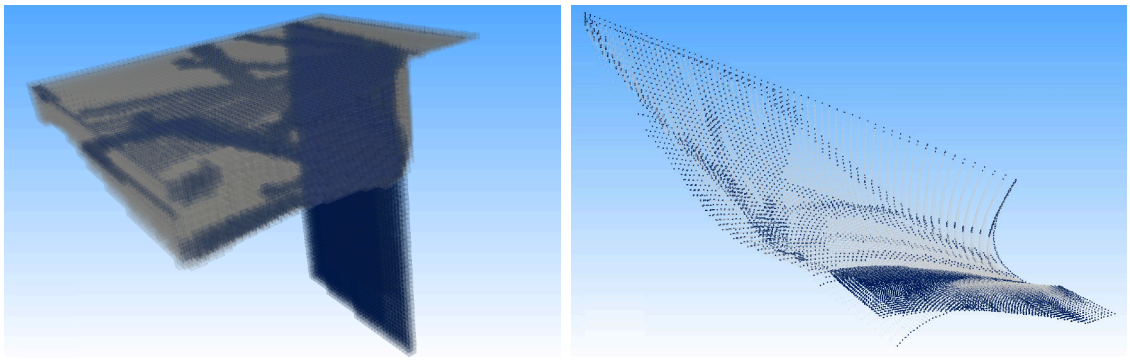
Table 4.1: Computational savings due to reduction of the number of objectives.

Problem	k	s	*# Subproblems*	*CPU time required in %*
(4.8)	3	2	3	25
(4.9)	4	3	4	2.3
(4.9)	4	2	6	0.04
(4.10)	14	2	91	7
(4.8) (ϵ-constr. (4.6))	3	2	10	18.3
(4.8) (ϵ-constr. (4.7))	3	2	10	10.7

the original problem (the second constraint in (4.7)) which in total results in approximately 10.7% of the time required for solving the 3D problem. An overview of the computational savings is given in Table 4.3.

4.4 Application: Industrial Laundry

As a last example, we consider an application from industry which has been investigated within the leading edge cluster *Intelligent Technical Systems OWL (it's OWL)*. In an industrial laundry, large amounts of laundry are processed every day. This results in a considerable demand for resources, namely energy, water and cleaning detergents, and it is advisable to utilize optimization and optimal control techniques to control such a laundry in an intelligent manner (see e.g. [PGH+16]). An important part in laundering is the cleaning process. Here, the optimal configuration depends on the type of laundry and the type of contamination. In 1959, Herbert Sinner [Sin59] developed a concept for laundering which is still applied in modern



(a) (b)

Figure 4.7: (a) The Pareto set of problem (4.10) for the parameters temperature, amount cleaning detergents and washing time. (b) Projection of the Pareto front on the objectives *cleaning of fat*, *cleaning of wine* and *cost*.

laundries. It states that in order to achieve satisfactory cleaning, the four influential factors temperature (u_1), chemistry (i.e. the amount of cleaning detergents, u_2), washing time (u_3) and mechanics (i.e. the rotational velocity of the laundry, u_4) have to be chosen in the right way. Based on this, the process of cleaning different types of contamination can be approximated by a quadratic model using experimental data. This has been done for thirteen different types of contamination (e.g. fat, wine, curry, oil, or blood). As a fourteenth objective, we consider the cost for the cleaning process which is simply modeled as a combination of the prizes for heating and for the washing detergents, i.e. $J_{14}(\boldsymbol{u}) = \varphi_1 u_1 + \varphi_2 u_2$. In the optimization, we neglect the influence of mechanics (i.e. u_4) since all objectives are non-conflicting in that regard. This leads to the following overall model with $\boldsymbol{J} : \mathbb{R}^3 \to \mathbb{R}^{14}$:

$$\min_{\boldsymbol{u}\in\mathbb{R}^3} \boldsymbol{J}(\boldsymbol{u}) = \min_{\boldsymbol{u}\in\mathbb{R}^3} \begin{pmatrix} \boldsymbol{u}^\top A_1 \boldsymbol{u} + b_1^\top \boldsymbol{u} + c_1 \\ \vdots \\ \boldsymbol{u}^\top A_{13} \boldsymbol{u} + b_{13}^\top \boldsymbol{u} + c_{13} \\ \varphi_1 u_1 + \varphi_2 u_2 \end{pmatrix}, \tag{4.10}$$

where the coefficients $A_i \in \mathbb{R}^{13,13}$, $b_i, c_i \in \mathbb{R}^{13}$, $i = 1, \ldots, 13$, are determined by experiments. Since we have fourteen objectives and only three parameters, the Pareto set does not possess the additional structure required for Theorems 4.1.4 or 4.1.7 to be applicable. However, we can still compute subsets of the Pareto set by considering subsets of objectives, cf. Lemma 4.1.3. When considering two objectives only, this leads to 91 bi-objective problems in total. Nevertheless, the computational saving is approximately 93% compared to the original problem. The Pareto set and the Pareto front of the original problem are shown in Figure 4.7 in gray and the unions of the 91 Pareto sets and Pareto fronts with two objectives in blue. We see that a well-spread approximation of the original set is obtained.

However, some parts are covered more coarsely than others which could be further improved by applying the ϵ-constraint method, solving additional subproblems (e.g. with three objectives) or by subsequently applying continuation or evolutionary algorithms.

5 Multiobjective Optimal Control of PDEs Using Reduced Order Modeling

This chapter addresses the third main source of large computational cost: the complexity of the model itself. The goal therefore is to exploit structure in the system dynamics to achieve a significant reduction of the computing time. We are going to consider control problems where the system dynamics is described by PDEs. There exists a large variety of systems that can be described by PDEs, from fairly simple problems such as the linear heat equation up to highly non-linear fluid flow problems at large Reynolds numbers governed by the Navier-Stokes equations. The typical procedure to numerically solve a PDE is to introduce a discretization both in space and time. The spatial domain is discretized by a numerical mesh based on a finite difference, finite volume or finite element method (see e.g. [FP02] for an introduction related to fluid dynamics). This transforms the infinite-dimensional into a (potentially very large) finite-dimensional system of coupled ODEs which is solved using time stepping schemes such as Runge-Kutta methods. With increasing computational capacities, the size of problems that can be solved has tremendously increased during the last decades. However, many technical applications result in problems that are still very difficult to solve or even infeasible. This is for example the case when one wants to directly solve the Navier-Stokes equations for the flow in an internal combustion engine or around an entire aircraft. Consequently, solving MOCPs constrained by PDEs is a great challenge.

In order to reduce the computational effort, the original model can be replaced by a reduced order model (ROM) with decreased complexity. As discussed in Section 2.3.3, there are three different approaches for incorporating ROMs in an optimization routine:

(1) creating a single ROM which sufficiently accurately approximates solutions for a wide range of controls,

(2) trust region methods where the improvement obtained with a ROM is evaluated by comparison with high fidelity solution,

(3) error analysis based on the truncation error of the POD basis.

In this chapter the three above-mentioned approaches to reduced order modeling based on POD are addressed and combined with multiobjective optimal control

methods in order to solve PDE-constrained MOCPs. Section 5.1 serves as an introduction and a motivation for the following chapters. There, a flow control problem based on the Navier-Stokes equations is considered and two different approaches how reduced order modeling and multiobjective optimal control can be combined in order to efficiently solve these otherwise infeasible problems are compared. Due to the computational effort and the lack of error bounds, a single ROM is computed in advance. The results are compared to the high-fidelity solution which shows good agreement but nonetheless strongly motivates more advanced approaches where convergence can be guaranteed. To this end, the trust region approach developed by Fahl [Fah00] will be extended to multiple objectives in Section 5.2 by combination with the reference point approach presented in Section 2.1.4. Finally, the subdivision algorithm presented in Section 2.1.5 will be extended to inexact function and gradient values in Section 5.3. In combination with error estimates for POD-based reduced order models, this paves the way for set-oriented multiobjective optimal control of PDEs, and a problem of this type will be addressed in Section 5.4.

5.1 Multiobjective Optimal Control of the Navier-Stokes Equations

Flow control is a very active area of research (see e.g. [BN15] for an overview). Due to the wide spectrum of potential applications (optimal mixing, drag reduction, HVAC (Heating, Ventilation and Air Conditioning), etc.) and the progress in computational capabilities, a lot of research is nowadays devoted to direct control of the Navier-Stokes equations [Lac61, GeH89, BMT01, Bew01, KB07].

However, since directly addressing the Navier-Stokes equations is a computational challenge, reduced order modeling plays an important role. Starting with Sirovich [Sir87], many researchers have investigated ROMs based on POD and Galerkin projection in order to improve their capability of accurately predicting the dynamical behavior, both on limit cycles and during transition phases [DKKO91, Rem00, MK02, RCM04, IR06, IR08, CFCA13, XFB+14]. The natural next step after being able to efficiently predict the dynamics with a ROM is to utilize this in a multi-query context such as parameter estimation [Las14] or optimal control, see e.g. [IR98, GPT99, Rav00, IR01, BCB05] for methods using one fixed POD basis or [Fah00, WP02, Row05, HV05, BC08] for adaptive methods where convergence with respect to the PDE-based solution is guaranteed. Flow control problems with concurrent objectives have been investigated before (see e.g. [MDL99, NZP04, KHW15]), but not in combination with reduced order modeling.

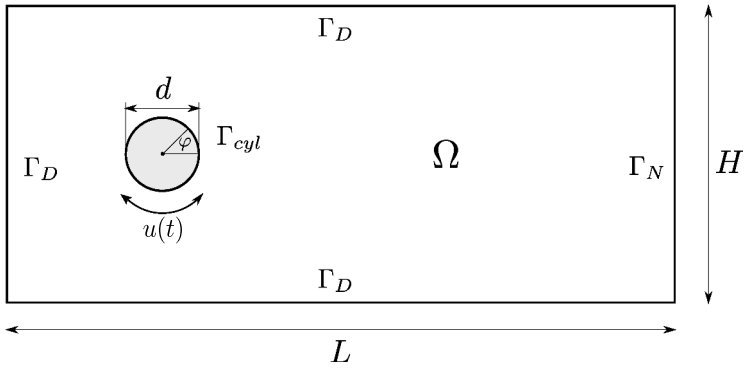


Figure 5.1: Sketch of the domain $\Omega \subset \mathbb{R}^2$. The length is $L = 25d$, the height $H = 15d$, and the cylinder center is placed at $(5d, 7.5d)$.

In this section the approach with one fixed basis will be adopted and combined with two fundamentally different algorithms for multiobjective optimal control problems, namely the reference point method (cf. Section 2.1.4) and the sampling algorithm (cf. Section 2.1.5). After introducing the problem formulation, the numerical scheme for obtaining the data and the PDE-based MOCP in Sections 5.1.1, 5.1.2 and 5.1.3, respectively, the ROM procedure for the Navier-Stokes equations with boundary control is introduced in Section 5.1.4. The adjoint approach utilized for the gradient-based reference point method is presented in Section 5.1.5. In Section 5.1.6, results are shown and the advantages and disadvantages of the two approaches are discussed, also with respect to the computational cost. In fact, it is shown that the different methods strongly differ in their respective efficiency and that gradient-based approaches have a better performance, provided that the gradient is approximated with sufficient accuracy. Parts of this section have previously appeared in [PD15, POBD15] to which the author has made substantial contributions.

5.1.1 Problem Formulation

The two-dimensional, viscous flow around a cylinder is one of the most extensively studied cases of separated flows in general as well as for flow control problems [GPT99, GBZI04, BCB05]. In this section the laminar case described by the incompressible Navier-Stokes equations at a Reynolds Number $Re = \frac{Y_\infty d}{\nu} = 200$ is considered (Re is computed with respect to the far field velocity Y_∞, the kinematic

viscosity ν and the cylinder diameter d):

$$\frac{\partial \boldsymbol{Y}(\boldsymbol{x},t)}{\partial t} + (\boldsymbol{Y}(\boldsymbol{x},t)\cdot\nabla)\,\boldsymbol{Y}(\boldsymbol{x},t) = -\frac{\nabla p(\boldsymbol{x},t)}{\rho} + \frac{1}{Re}\nabla^2\boldsymbol{Y}(\boldsymbol{x},t), \tag{5.1a}$$

$$\nabla\cdot\boldsymbol{Y}(\boldsymbol{x},t) = 0, \tag{5.1b}$$

$$(\boldsymbol{Y}(\boldsymbol{x},t_0), p(\boldsymbol{x},t_0)) = (\boldsymbol{Y}_0(\boldsymbol{x}), p_0(\boldsymbol{x}))\,, \quad \text{for } \boldsymbol{x}\in\Omega \text{ and } t\in(t_0,t_e], \tag{5.1c}$$

where $\boldsymbol{Y}\in H^1((t_0,t_e);L^2(\Omega))\cap L^2((t_0,t_e);H^2(\Omega))$ is the two-dimensional fluid velocity and $p\in H^1(\Omega\times(t_0,t_e);\mathbb{R})$ the pressure. H^k is the standard Sobolev space $W^{k,2}$ (cf. e.g. [HPUU09]). The spatial domain (cf. Figure 5.1) is denoted by Ω. Dirichlet boundary conditions (BC) are imposed at the inflow as well as the upper and lower walls Γ_D and a standard *no shear stress* condition [HRT96] at the outflow Γ_N:

$$\boldsymbol{Y}(\boldsymbol{x},t) = (Y_\infty, 0) \qquad \text{for } (\boldsymbol{x},t)\in\Gamma_D\times(t_0,t_e], \tag{5.1d}$$

$$p(\boldsymbol{x},t)\,\boldsymbol{n} = \frac{1}{Re}\frac{\partial\boldsymbol{Y}(\boldsymbol{x},t)}{\partial\boldsymbol{n}} \qquad \text{for } (\boldsymbol{x},t)\in\Gamma_N\times(t_0,t_e], \tag{5.1e}$$

where $\boldsymbol{n}\in\mathbb{R}^2$ is the outward normal vector of the boundary. On the cylinder Γ_{cyl}, we prescribe a time-dependent Dirichlet BC such that it performs a rotation around its center with the angular velocity $u(t)$:

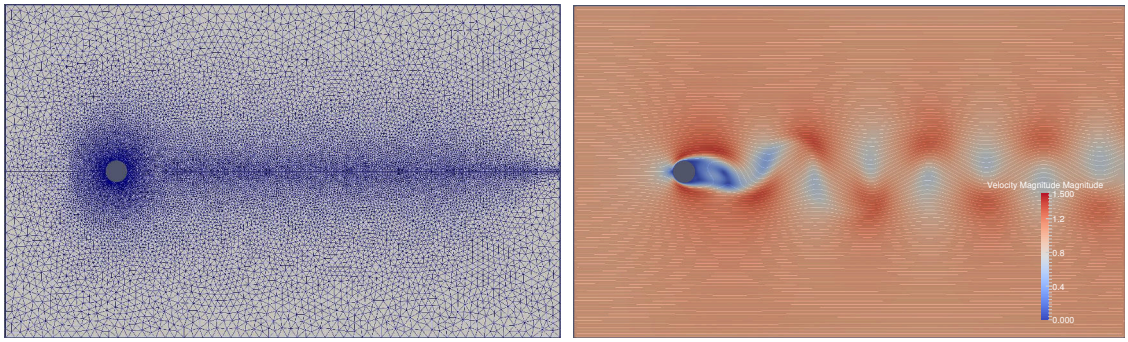
$$\boldsymbol{Y}(\boldsymbol{x},t) = \frac{d}{2}u(t)\begin{pmatrix}-\sin(\varphi)\\ \cos(\varphi)\end{pmatrix} \qquad \text{for } (\boldsymbol{x},t)\in\Gamma_{cyl}\times(t_0,t_e], \tag{5.1f}$$

with φ according to Figure 5.1. The cylinder rotation $u(t)\in L^2((t_0,t_e);\mathbb{R})$ serves as the control mechanism for the flow.

Following [Fah00, BCB05], the weak formulation of (5.1a) is introduced. Consider the space $V=\{\boldsymbol{\psi}\in H^1(\Omega;\mathbb{R}^2)\mid\nabla\cdot\boldsymbol{\psi}=0\}$ of divergence-free test functions. Then a function $\boldsymbol{Y}\in H^1(\Omega\times(t_0,t_e);\mathbb{R}^2)$ which satisfies

$$\left(\frac{\partial\boldsymbol{Y}}{\partial t}+(\boldsymbol{Y}\cdot\nabla)\,\boldsymbol{Y},\boldsymbol{\psi}\right) = \left(\frac{p}{\rho},\nabla\cdot\boldsymbol{\psi}\right) - \left[\frac{p}{\rho}\boldsymbol{\psi}\right] - \frac{1}{Re}\left((\nabla\boldsymbol{\psi},\nabla\boldsymbol{Y}) - \left[(\nabla\boldsymbol{Y})^\top\boldsymbol{\psi}\right]\right) \tag{5.2}$$

for all $\boldsymbol{\psi}\in V$ is called a *weak solution* of (5.1a). Here, $[\cdot]$ is the boundary integral (e.g. $[\boldsymbol{Y}]=\int_\Gamma\boldsymbol{Y}(\boldsymbol{x})\cdot\boldsymbol{n}\,\mathrm{d}\boldsymbol{x}$) and $(\cdot,\cdot)$ is the inner product for vector-valued quantities (e.g. $(\nabla\boldsymbol{\psi},\nabla\boldsymbol{Y})=\int_\Omega\sum_{i,j}\partial\boldsymbol{\psi}_i/\partial x_j\cdot\partial\boldsymbol{Y}_i/\partial x_j\,\mathrm{d}\boldsymbol{x}$). Note that (5.1b) is always satisfied by design of the test function space V.



(a) (b)

Figure 5.2: (a) FEM discretization of the domain Ω by a triangular mesh ($n_y = 17048$). (b) A snapshot of the solution to (5.1a) – (5.1f) for a non-rotating cylinder ($u(t) = 0$), the coloring is according to the velocity magnitude. The pattern is the well-known von Kármán vortex street.

5.1.2 Numerical discretization

The system (5.1a – 5.1f) is solved with the software package *OpenFOAM* [JJT07] using a finite volume (FV) discretization and the *PISO* scheme [FP02]. Since the computation of the reduced order model (evaluation of inner products etc.) is based on the finite element method (FEM), the solution is then mapped to a finite element mesh with $n_y = 17048$ degrees of freedom (Figure 5.2 (a)). This is done in the spirit of data driven modeling, where we collect data which does not necessarily have to be obtained by a numerical method. Finally, the velocity field can be written in terms of the FEM basis:

$$\boldsymbol{Y}(\boldsymbol{x}, t) = \begin{pmatrix} \sum_{j=1}^{n_y} Y_j^d(t)\phi_j(\boldsymbol{x}) \\ \sum_{j=1}^{n_y} Y_{j+n_y}^d(t)\phi_j(\boldsymbol{x}) \end{pmatrix}, \tag{5.3}$$

where $\{\phi_j\}_{j=1}^{n_y}$ are the FEM basis functions and $Y^d(t) \in \mathbb{R}^{2n_y}$ are the nodal values of the two velocity components, the superscript d denoting that this is a quantity defined on the grid nodes[1]. All FEM computations are performed with the *FEniCS* toolbox [LMW12] using linear basis functions.

For a non-rotating cylinder, i.e. $u(t) = 0$, the system possesses a periodic solution, the well-known *von Kármán vortex street* (Figure 5.2 (b)), where vortices separate alternatingly from the upper and lower edge of the cylinder. The effect is frequently observed in nature and is one of the most studied phenomena in fluid mechanics, also in the context of flow control, where the objective is to stabilize the flow and to reduce the drag.

[1] In the following, the nodal values of all quantities will be denoted by a superscript d.

5.1.3 Multiobjective optimal control problem

In many applications, the control cost is of great interest. This is immediately clear when the goal of the optimization is to save energy such that the control effort has to be taken into account. In scalar optimization problems, this is often done by adding an additional term of the form $\beta \int_{t_0}^{t_e} u^2(t)\,\mathrm{d}t$ to the cost functional where $\beta \in \mathbb{R}_{\geq 0}$ is a weighting or *regularization* parameter. Here, we want to consider two objectives separately, namely the minimization of the fluctuations

$$\boldsymbol{y}(\boldsymbol{x},t) = \boldsymbol{Y}(\boldsymbol{x},t) - \langle\, \boldsymbol{Y}(\boldsymbol{x},\cdot)\rangle$$

around the mean flow field

$$\langle\, \boldsymbol{Y}(\boldsymbol{x},\cdot)\rangle = \frac{1}{t_e - t_0}\int_{t_0}^{t_e} \boldsymbol{Y}(\boldsymbol{x},t)\,\mathrm{d}t,$$

i.e. the stabilization of the flow, and the minimization of the control cost. This leads to the following MOCP with $\hat{\boldsymbol{J}} : C([t_0,t_e];H^1(\Omega)) \cap L^2((t_0,t_e);H^2(\Omega)) \cap H^1((t_0,t_e);L^2(\Omega)) \times L^2((t_0,t_e);\mathbb{R}) \to \mathbb{R}^2$:

$$\begin{aligned} \min_{\boldsymbol{Y},u} \hat{\boldsymbol{J}}(\,\boldsymbol{Y},u) = \min_{\boldsymbol{Y},u} \begin{pmatrix} \int_{t_0}^{t_e} \|\boldsymbol{y}(\cdot,t)\|_{L^2}^2\,\mathrm{d}t \\ \|u\|_{L^2}^2 \end{pmatrix} \\ \text{s.t.} \qquad (5.1). \end{aligned}$$

In contrast to bounded domains, the proof of existence of a solution is an open problem for cases with *no shear stress* BCs, cf. [FGH98]. Nevertheless, based on numerical experiences [Ran00] we will from now on assume that there exists a unique solution $\boldsymbol{Y}(\boldsymbol{x},t)$ for each control u and hence (in accordance with the notation in Section 2.2.1), we denote by $\boldsymbol{Y}(u)$ the solution $\boldsymbol{Y}(\boldsymbol{x},t)$ for a fixed $u \in L^2((t_0,t_e);\mathbb{R})$ and consider the reduced cost functional $\boldsymbol{J} : L^2((t_0,t_e);\mathbb{R}) \to \mathbb{R}^2$ which leads to the reduced MOCP:

$$\min_u \boldsymbol{J}(u) = \min_u \begin{pmatrix} \int_{t_0}^{t_e} \|\boldsymbol{y}(\cdot,t)\|_{L^2}^2\,\mathrm{d}t \\ \|u\|_{L^2}^2 \end{pmatrix}. \tag{5.4}$$

5.1.4 Reduced Order Model

Problem (5.4) can now be solved using one of the methods presented in Section 2.1.4. However, all approaches require many evaluations of the cost functional and consequently of the system (5.1a – 5.1f). A FEM or FV discretization yields a large number of degrees of freedom n_y such that solving (5.4) quickly becomes computationally infeasible. Hence, we need to significantly reduce the cost for solving the dynamical system. This is achieved by means of reduced order modeling where we

replace the state equation by a reduced order model of coupled, non-linear ODEs. The more structure the dynamical system possesses, the smaller the size of the ROM and consequently, the system can be solved much faster.

The reduced order model is derived by introducing a Galerkin ansatz, cf. Section 2.3.1:

$$\boldsymbol{Y}(\boldsymbol{x},t) = \sum_{j=1}^{\ell} z_j(t)\boldsymbol{\psi}_j(\boldsymbol{x}), \qquad (\boldsymbol{x},t) \in \Omega \times (t_0,t_e], \tag{5.5}$$

where $\boldsymbol{z} \in H^1((t_0,t_e);\mathbb{R}^\ell)$ are the time-dependent coefficients and $\{\boldsymbol{\psi}_j\}_{j=1}^{\ell}$ are basis functions[2]. In contrast to FEM basis functions, these are in general global, i.e. they have full support. The objective is therefore to find a basis as small as possible ($\ell \ll n_y$) while allowing for a good approximation of the flow field.

We can derive a ROM of dimension ℓ using (5.5). By selecting a basis that is divergence-free, the mass conservation equation (5.1b) is automatically satisfied. (When collecting data from a divergence-free flow field, the resulting basis elements are also divergence-free, see Section 2.3.1, p. 45.) Inserting (5.5) into the weak formulation of the Navier-Stokes equations (5.2) yields a system of ℓ ODEs:

$$\dot{\boldsymbol{z}}(t) = \widetilde{\mathcal{B}}\boldsymbol{z}(t) + \widetilde{\mathcal{C}}\left(\boldsymbol{z}(t)\right),$$

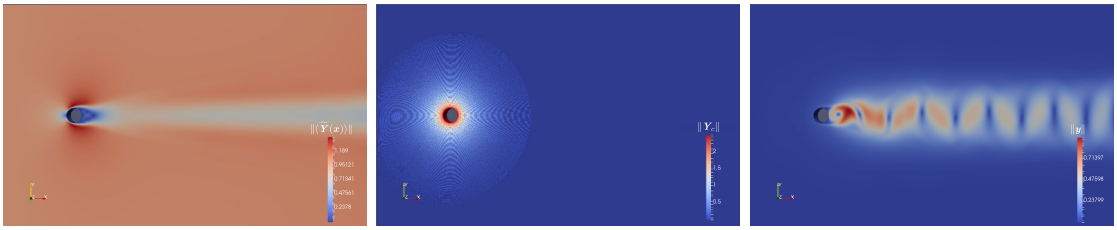
where the coefficient matrices are computed by evaluating the L^2 inner products:

$$\begin{aligned}
\left(\widetilde{\mathcal{B}}\right)_{ij} &= \frac{1}{Re}\left(\nabla\boldsymbol{\psi}_i, \nabla\boldsymbol{\psi}_j\right)_{L^2}, \\
\left(\widetilde{\mathcal{C}}\left(\boldsymbol{z}(t)\right)\right)_j &= \boldsymbol{z}(t)^\top \mathcal{K}_j \boldsymbol{z}(t), \\
\text{with } \left(\mathcal{K}_j\right)_{ik} &= \left(\left(\boldsymbol{\psi}_i \cdot \nabla\right)\boldsymbol{\psi}_k, \boldsymbol{\psi}_j\right)_{L^2}.
\end{aligned}$$

Remark 5.1.1. *In the above as well as all following formulations, the influence of the pressure term on the model coefficients has been neglected. In [NPM05], it has been shown that including the pressure term is favorable for the accuracy of the reduced model for open systems. Instead, an additional model stabilization is performed in order to increase the accordance with the flow field data which results in a further modification of the model coefficients. Consequently, following [CAF09], the influence of the pressure term is neglected.*

However, this model is not controllable in the sense that we can no longer influence the solution by choosing $u(t)$. Moreover, the basis functions have to be computed

[2]Note that now, both the state $\boldsymbol{Y}(\boldsymbol{x},t)$ as well as the modes $\boldsymbol{\psi}_j(\boldsymbol{x})$ are vector-valued.



(a) (b) (c)

Figure 5.3: Flow field decomposition such that $\boldsymbol{y}(\boldsymbol{x},t) = 0$ for $(\boldsymbol{x},t) \in \Gamma \times (t_0, t_e]$. (a) $\left\langle \widetilde{\boldsymbol{Y}}(\boldsymbol{x},\cdot) \right\rangle$. (b) $\boldsymbol{Y}_c$. (c) One snapshot of $\boldsymbol{y}$.

from data with homogeneous BCs ($\boldsymbol{Y}(\boldsymbol{x},t) = 0$ for $(\boldsymbol{x},t) \in \Gamma \times (t_0, t_e]$) such that they inherit these. Otherwise, it cannot be guaranteed that the Galerkin ansatz (5.5) satisfies the BCs for arbitrary values of $\boldsymbol{z}$ (cf. Section 2.3.1). The idea is therefore to introduce the Galerkin expansion for a modified flow field $\boldsymbol{y}(\boldsymbol{x},t)$ which satisfies homogeneous BCs for all $(\boldsymbol{x},t) \in \Gamma \times (t_0, t_e]$. In order to realize this, the following flow field decomposition is introduced [Rav00, BCB05]:

$$\boldsymbol{Y}(\boldsymbol{x},t) = \langle \boldsymbol{Y}(\boldsymbol{x},\cdot) \rangle + u(t)\, \boldsymbol{Y}_c(\boldsymbol{x}) + \boldsymbol{y}(\boldsymbol{x},t), \tag{5.6}$$

where $\boldsymbol{Y}_c$ is a so-called *control function* which is a solution to (5.1a) – (5.1c) with a constant cylinder rotation $u_c \in \mathbb{R}$ and homogeneous BCs elsewhere (cf. Figure 5.3 (b)). The choice of u_c for the computation of $\boldsymbol{Y}_c$ is arbitrary but has an influence on the resulting ROM such that the model performance might be influenced by this choice. Since the main intention of this section is to develop multiobjective optimal control algorithms for the Navier-Stokes equations, this problem is not addressed any further and a fixed value $u_c = 2$ is chosen such that the cylinder surface rotates with a surface velocity of 1. The decomposition for (5.6) is now computed in two steps:

$$\begin{aligned} \widetilde{\boldsymbol{Y}}(\boldsymbol{x},t) &= \boldsymbol{Y}(\boldsymbol{x},t)\big|_{u(t)=u_{\mathsf{ref}}(t)} - \frac{u_{\mathsf{ref}}(t)}{u_c}\, \boldsymbol{Y}_c(\boldsymbol{x}), \\ \boldsymbol{y}(\boldsymbol{x},t) &= \widetilde{\boldsymbol{Y}}(\boldsymbol{x},t) - \left\langle \widetilde{\boldsymbol{Y}}(\boldsymbol{x},\cdot) \right\rangle, \end{aligned}$$

where u_{ref} is the *reference control* for which the data is collected. The individual steps are visualized in Figure 5.3, where $\langle \widetilde{\boldsymbol{Y}}(\boldsymbol{x},\cdot) \rangle$ is shown in (a), $\boldsymbol{Y}_c$ in (b) and one snapshot of $\boldsymbol{y}$ in (c). This way, $\widetilde{\boldsymbol{Y}}$ satisfies homogeneous BCs on the cylinder Γ_{cyl} for all $t \in (t_0, t_e]$ and the fluctuating field $\boldsymbol{y}$, for which we introduce the Galerkin

expansion

$$\boldsymbol{y}(\boldsymbol{x},t) = \sum_{j=1}^{\ell} z_j(t)\boldsymbol{\psi}_j(\boldsymbol{x}), \tag{5.7}$$

satisfies homogeneous BCs everywhere[3].

Inserting (5.6) and (5.7) into (5.2) yields an extended ROM:

$$\begin{aligned} \dot{\boldsymbol{z}}(t) &= \mathcal{A} + \mathcal{B}\boldsymbol{z}(t) + \mathcal{C}\left(\boldsymbol{z}(t)\right) + \mathcal{D}\dot{u}(t) + \left(\mathcal{E} + \mathcal{F}\boldsymbol{z}(t)\right)u(t) + \mathcal{G}u^2(t), \\ z_j(t_0) &= \left(\boldsymbol{y}(\cdot,t_0),\boldsymbol{\psi}_j\right)_{L^2} =: z_{0,j}, \end{aligned} \tag{5.8}$$

where $\boldsymbol{z} \in H^1((t_0,t_e);\mathbb{R}^\ell)$. The size of the coefficient matrices $\mathcal{A}$ to $\mathcal{G}$ depends on the dimension ℓ of the ROM: $\mathcal{A},\ \mathcal{D},\ \mathcal{E},\ \mathcal{G} \in \mathbb{R}^\ell$; $\mathcal{B},\ \mathcal{F} \in \mathbb{R}^{\ell,\ell}$; $\left(\mathcal{C}(\boldsymbol{z}(t))\right)_j = \boldsymbol{z}(t)^\top \mathcal{K}_j \boldsymbol{z}(t)$ with $\mathcal{K}_j \in \mathbb{R}^{\ell,\ell}$ for $j = 1,\dots,\ell$. They can be explicitly computed using the following equations (see also [Fah00, BCB05] for details), where the time average of the modified flow field is denoted by $\boldsymbol{Y}_m = \langle \widetilde{\boldsymbol{Y}}(\boldsymbol{x},\cdot)\rangle$ for abbreviation:

$$\begin{aligned} \mathcal{A}_i &= -\left(\left(\boldsymbol{Y}_m\cdot\nabla\right)\boldsymbol{Y}_m,\boldsymbol{\psi}_i\right)_{L^2} - \frac{1}{Re}\left(\nabla\boldsymbol{Y}_m,\nabla\boldsymbol{\psi}_i\right)_{L^2}, \\ \mathcal{B}_{ij} &= -\left(\left(\boldsymbol{Y}_m\cdot\nabla\right)\boldsymbol{\psi}_j,\boldsymbol{\psi}_i\right)_{L^2} - \left(\left(\boldsymbol{\psi}_j\cdot\nabla\right)\boldsymbol{Y}_m,\boldsymbol{\psi}_i\right)_{L^2} - \frac{1}{Re}\left(\nabla\boldsymbol{\psi}_i,\nabla\boldsymbol{\psi}_j\right)_{L^2}, \\ \left(\mathcal{K}_j\right)_{ik} &= -\left(\left(\boldsymbol{\psi}_i\cdot\nabla\right)\boldsymbol{\psi}_k,\boldsymbol{\psi}_j\right)_{L^2}, \\ \mathcal{D}_i &= -\left(\boldsymbol{Y}_c,\boldsymbol{\psi}_i\right)_{L^2}, \\ \mathcal{E}_i &= -\left(\left(\boldsymbol{Y}_m\cdot\nabla\right)\boldsymbol{Y}_c,\boldsymbol{\psi}_i\right)_{L^2} - \left(\left(\boldsymbol{Y}_c\cdot\nabla\right)\boldsymbol{Y}_m,\boldsymbol{\psi}_i\right)_{L^2} - \frac{1}{Re}\left(\nabla\boldsymbol{Y}_c,\nabla\boldsymbol{\psi}_i\right)_{L^2}, \\ \mathcal{F}_{ij} &= -\left(\left(\boldsymbol{Y}_c\cdot\nabla\right)\boldsymbol{\psi}_j,\boldsymbol{\psi}_i\right)_{L^2} - \left(\left(\boldsymbol{\psi}_j\cdot\nabla\right)\boldsymbol{Y}_c,\boldsymbol{\psi}_i\right)_{L^2}, \\ \mathcal{G}_i &= -\left(\left(\boldsymbol{Y}_c\cdot\nabla\right)\boldsymbol{Y}_c,\boldsymbol{\psi}_i\right)_{L^2}. \end{aligned}$$

The initial value problem (5.8) is integrated in time using a Runge-Kutta method of fourth order.

Remark 5.1.2. *Since the time derivative of the control occurs in (5.8), we require higher regularity, i.e. $u \in H^1((t_0,t_e);\mathbb{R})$. Moreover, the second objective has to be altered slightly to avoid* bang bang *solutions [Ger12]. This point will be addressed in more detail in Section 5.1.5.*

Due to the truncation of the POD basis, the higher modes covering the small scale dynamics, i.e. the dynamics responsible for dissipation, are neglected. This can lead to incorrect long term behavior and hence, the model has to be modified to account

[3] Note that this does not hold exactly on the outflow boundary Γ_N. However, the resulting error is small and hence neglected [BCB05].

for this. Several researchers have addressed this problem and proposed different solutions, see e.g. [Rem00, SK04, NPM05, CAF09, BBI09]. Here, stabilization of the model (5.8) is achieved by solving an augmented optimization problem [GBZI04]. In this approach, the entries of the matrix $\mathcal{B}$ are manipulated in order to minimize the difference between the ROM solution and the projection of the data onto the reduced basis ($z_j^p(t) = (\boldsymbol{Y}(\cdot,t), \boldsymbol{\psi}_j)_{L^2}$).

After replacing the high fidelity by the surrogate model, the objectives in (5.4) can be replaced accordingly by equivalent formulations in terms of ROM-based quantities:

$$\min_u \begin{pmatrix} \int_{t_0}^{t_e} \int_\Omega \| \sum_{i=1}^{\ell} z_i(t)\boldsymbol{\psi}_i(\boldsymbol{x})\|_2^2 \,\mathrm{d}\boldsymbol{x}\,\mathrm{d}t \\ \ell \int_{t_0}^{t_e} u^2(t)\,\mathrm{d}t \end{pmatrix}. \tag{5.9}$$

Making use of the orthonormality of the POD basis, i.e. $(\boldsymbol{\psi}_i, \boldsymbol{\psi}_j)_{L^2} = \delta_{i,j}$, the first objective in (5.9) can be simplified:

$$\int_\Omega \Big\| \sum_{i=1}^{\ell} z_i(t)\boldsymbol{\psi}_i(\boldsymbol{x}) \Big\|_2^2 \,\mathrm{d}\boldsymbol{x} = \int_\Omega \left(\sum_{i=1}^{\ell} z_i(t)\boldsymbol{\psi}_i(\boldsymbol{x}) \right) \cdot \left(\sum_{i=1}^{\ell} z_i(t)\boldsymbol{\psi}_i(\boldsymbol{x}) \right) \mathrm{d}\boldsymbol{x} = \sum_{i=1}^{\ell} z_i^2(t),$$

which allows us to replace (5.9) by

$$\min_u \boldsymbol{J}^r(u) = \min_u \begin{pmatrix} \int_{t_0}^{t_e} \sum_{i=1}^{\ell} z_i^2(t)\,\mathrm{d}t \\ \ell \int_{t_0}^{t_e} u^2(t)\,\mathrm{d}t \end{pmatrix}, \tag{R-MOCP}$$

where the superscript r indicates that the objective functional is based on the reduced state $\boldsymbol{z}$.

Finally, it remains to determine an adequate basis size for the reduced model. The error $\epsilon(\ell)$ (cf. Equation (2.30) on p. 45) is only known for the reference control at which the data is collected. If one allows for multiple evaluations of the finite element solution, the ROM can be updated when necessary. This is realized using a trust-region approach which will be covered in the next section. An alternative approach is to use a reference control that yields "sufficiently rich" dynamics such that the model can be trusted for a large variety of controls [BCB05]. We here follow the latter approach and take 1201 snapshots in the interval $[0, 60]$ with $\Delta t = 0.05$ for a *chirping* reference control:

$$u_{chirp}(t) = -4\sin(2\pi t/120)\cos(2\pi t/3 - 18\sin(2\pi t/60)),$$

see also Figure 5.4. Demanding $\epsilon(\ell) \geq 99\%$ leads to a basis of size $\ell = 38$.

For the uncontrolled flow (i.e. $u(t) = 0$), Figure 5.5 shows the comparison between

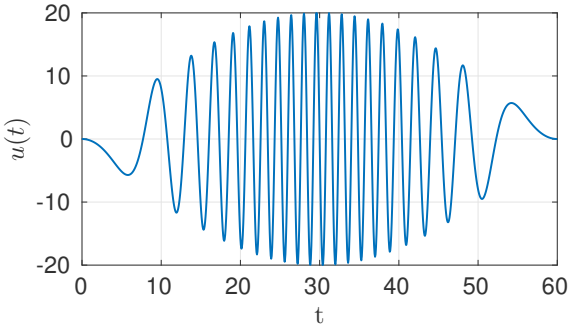


Figure 5.4: Chirping function [BCB05].

the solution obtained by two different ROMs and the projection of the high fidelity solution onto the corresponding POD bases. The model in Figure 5.5 (a) has been computed with a reference control $u_{\text{ref}} = 0$. We see that the agreement is very good which is not surprising since the data has been collected precisely from this solution. In Figure 5.5 (b), the model has been obtained at a reference control $u_{\text{ref}} = u_{\text{chirp}}$ and a larger difference can be observed. Nevertheless, the qualitative agreement is satisfactory for a variety of different controls as desired for optimal control purposes.

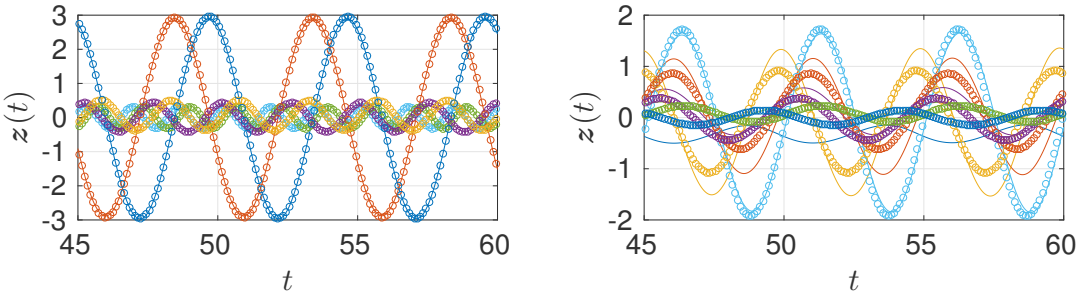


Figure 5.5: The coefficients $\boldsymbol{z}$ of the first six modes of (5.8) (—) and projection of the high fidelity solution (∘) for two ROMs obtained at different reference controls. (a) $u_{\text{ref}} = 0$. (b) $u_{\text{ref}} = u_{\text{chirp}}$.

Figure 5.6 shows the first four POD modes for the uncontrolled solution which we have already seen in the example in Section 2.3.1. In Figure 5.7 (a), the corresponding singular values are shown for both the uncontrolled flow and the flow controlled by the chirping function. The error, i.e. the ratio of the truncated eigenvalues, is visualized in (b).

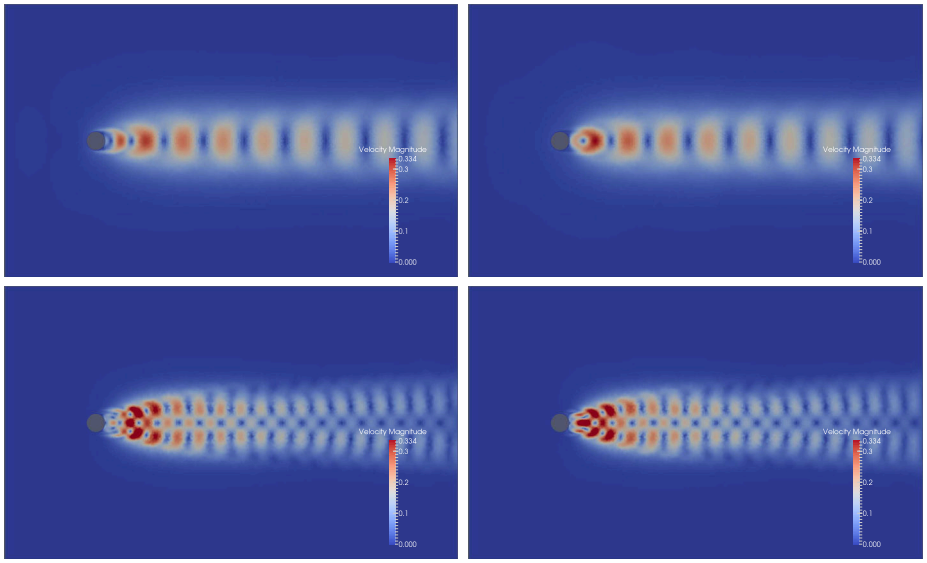
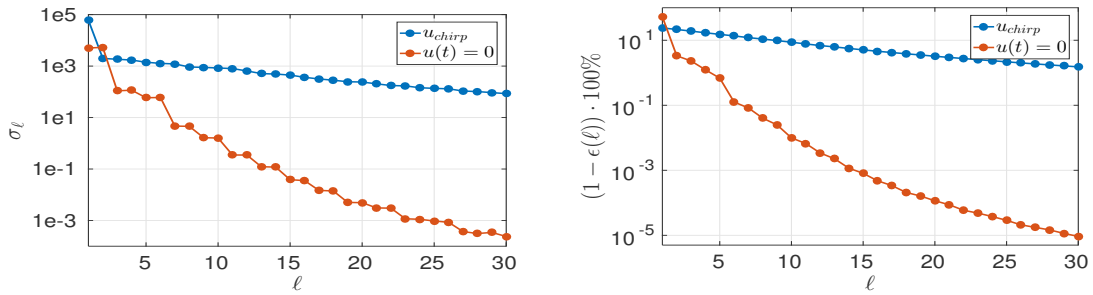


Figure 5.6: The first four POD modes for an uncontrolled solution.



(a) (b)

Figure 5.7: Eigenvalues (a) and error (b) at current basis size for an uncontrolled solution and a solution controlled by the chirping function.

5.1.5 Adjoint Systems

The gradient-based algorithm to solve the scalar optimization problems occurring in the reference point method (Section 2.1.4) is realized with an adjoint approach based on the Lagrange functional (cf. Section 2.2.1, p. 36). The first step is to transform (R-MOCP) into a scalar optimization problem where the euclidean distance to a

target $\boldsymbol{T}$ has to be minimized:

$$\min_u \overline{J^r}(u) = \min_u \|\boldsymbol{T} - \boldsymbol{J}^r(u)\|_2^2, \tag{5.10}$$

$$\text{with} \quad \overline{J^r}(u) = (T_1 - J_1^r(\boldsymbol{z}, u))^2 + (T_2 - J_2^r(\boldsymbol{z}, u))^2 = \left(T_1 - \int_{t_0}^{t_e} \sum_{i=1}^{\ell} z_i^2(t)\,\mathrm{d}t\right)^2 + \left(T_2 - \ell \int_{t_0}^{t_e} u^2(t)\,\mathrm{d}t\right)^2, \tag{5.11}$$

where $\boldsymbol{J}^r$ is according to (R-MOCP) and $\boldsymbol{z}$ satisfies the reduced order state equation (5.8).

The task is now to solve a sequence of scalar optimization problems with varying targets $\boldsymbol{T}$. Problem (5.10) is addressed by a line search strategy [NW06], where the direction is computed with a conjugate gradient method and the step length via backtracking such that it satisfies the Armijo rule. The required gradient information is computed with an adjoint approach. To this end, we introduce the adjoint state $\boldsymbol{\lambda} \in H^1((t_0, t_e); \mathbb{R}^\ell)$ and the Lagrange functional

$$\mathcal{L}(\boldsymbol{z}, \boldsymbol{\lambda}, u) = \int_{t_0}^{t_e} \overline{J^r}(u) - \boldsymbol{\lambda}^\top(t)\big(\dot{\boldsymbol{z}}(t) - \mathcal{A} - \mathcal{B}\boldsymbol{z}(t) - \mathcal{C}(\boldsymbol{z}(t)) - \mathcal{D}\dot{u}(t) - (\mathcal{E} + \mathcal{F}\boldsymbol{z}(t))u(t) - \mathcal{G}u^2(t)\big)\,\mathrm{d}t, \tag{5.12}$$

which is stationary for optimal values of u and the corresponding state $\boldsymbol{z}$ and adjoint state $\boldsymbol{\lambda}$. Using integration by parts, this leads to the following system of equations[4]:

$$\dot{\boldsymbol{z}}(t) = \mathcal{A} + \mathcal{B}\boldsymbol{z}(t) + \mathcal{C}(\boldsymbol{z}(t)) + \mathcal{D}\dot{u}(t) + (\mathcal{E} + \mathcal{F}\boldsymbol{z}(t))u(t) + \mathcal{G}u^2(t), \tag{5.13}$$

$$\dot{\boldsymbol{\lambda}}(t) = -\frac{\partial \overline{J^r}}{\partial \boldsymbol{z}}(t) - \left(\mathcal{B} + \frac{\partial \mathcal{C}(\boldsymbol{z})}{\partial \boldsymbol{z}}(t) + \mathcal{F}u(t)\right)^\top \boldsymbol{\lambda}(t), \tag{5.14}$$

$$0 = \frac{\partial \overline{J^r}}{\partial u}(t) + (\mathcal{E} + \mathcal{F}\boldsymbol{z}(t) + 2\mathcal{G}u(t))^\top \boldsymbol{\lambda}(t) - \mathcal{D}^\top \dot{\boldsymbol{\lambda}}(t) =: \nabla_u \overline{J^r}(t). \tag{5.15}$$

The above derivatives are abbreviations for the Fréchet derivatives

$$\frac{\partial \overline{J^r}}{\partial z_j}(t) = 4\left(\int_{t_0}^{t_e} \sum_{i=1}^{\ell} z_i^2(t)\,\mathrm{d}t - T_1\right) z_j(t),$$

$$\frac{\partial \overline{J^r}}{\partial u}(t) = 4\ell\left(\ell \int_{t_0}^{t_e} u^2(t)\,\mathrm{d}t - T_2\right) u(t),$$

and the initial and finial conditions are $\boldsymbol{z}(t_0) = \boldsymbol{z}_0$ and $\boldsymbol{\lambda}(t_e) = 0$, respectively.

[4] Note, that since this approach makes use of variational calculus, it is formally only applicable in the case of stronger assumptions, namely for $\boldsymbol{z}, \boldsymbol{\lambda}, u \in C^1$ (since we apply integration by parts).

Remark 5.1.3. *The optimality system (5.13) – (5.15) for the scalarized cost functional (5.11) can be derived as follows. In order to satisfy the necessary condition for optimality, we require all variations of the Lagrange functional (5.12) to be zero:*

$$\begin{aligned}
\delta\mathcal{L} &= \frac{\partial\mathcal{L}}{\partial \boldsymbol{z}}\delta\boldsymbol{z} + \frac{\partial\mathcal{L}}{\partial \dot{\boldsymbol{z}}}\delta\dot{\boldsymbol{z}} + \frac{\partial\mathcal{L}}{\partial \boldsymbol{\lambda}}\delta\boldsymbol{\lambda} + \frac{\partial\mathcal{L}}{\partial u}\delta u + \frac{\partial\mathcal{L}}{\partial \dot{u}}\delta\dot{u} = 0 \\
\Leftrightarrow \int_{t_0}^{t_e} &\left(\frac{\partial\overline{J^r}}{\partial \boldsymbol{z}} + \boldsymbol{\lambda}^\top\left(\mathcal{B} + \frac{\partial\mathcal{C}(\boldsymbol{z})}{\partial \boldsymbol{z}} + \mathcal{F}u\right)\right)\delta\boldsymbol{z} + \left(\frac{\partial\overline{J^r}}{\partial u} + \boldsymbol{\lambda}^\top\left(\mathcal{E} + \mathcal{F}\boldsymbol{z} + 2\mathcal{G}u\right)\right)\delta u + \\
&+ \left(\dot{\boldsymbol{z}} - \mathcal{A} - \mathcal{B}\boldsymbol{z} - \mathcal{C}(\boldsymbol{z}) - \mathcal{D}\dot{u} - (\mathcal{E} + \mathcal{F}\boldsymbol{z})u - \mathcal{G}u^2\right)\delta\boldsymbol{\lambda} + \boldsymbol{\lambda}^\top\mathcal{D}\delta\dot{u} - \boldsymbol{\lambda}^\top\delta\dot{\boldsymbol{z}}\,\mathrm{d}t = 0,
\end{aligned}$$

where the arguments have been omitted for abbreviation. Using integration by parts, this leads to

$$\begin{aligned}
&\int_{t_0}^{t_e}\left(\dot{\boldsymbol{\lambda}}^\top + \frac{\partial\overline{J^r}}{\partial \boldsymbol{z}} + \boldsymbol{\lambda}^\top\left(\mathcal{B} + \frac{\partial\mathcal{C}(\boldsymbol{z})}{\partial \boldsymbol{z}} + \mathcal{F}u\right)\right)\delta\boldsymbol{z} + \\
&+ \left(\dot{\boldsymbol{z}} - \mathcal{A} - \mathcal{B}\boldsymbol{z} - \mathcal{C}(\boldsymbol{z}) - \mathcal{D}\dot{u} - (\mathcal{E} + \mathcal{F}\boldsymbol{z})u - \mathcal{G}u^2\right)\delta\boldsymbol{\lambda} + \\
&+ \left(-\dot{\boldsymbol{\lambda}}^\top\mathcal{D} + \frac{\partial\overline{J^r}}{\partial u} + \boldsymbol{\lambda}^\top\left(\mathcal{E} + \mathcal{F}\boldsymbol{z} + 2\mathcal{G}u\right)\right)\delta u\,\mathrm{d}t - \boldsymbol{\lambda}^\top\delta\boldsymbol{z}\Big|_{t_0}^{t_e} + \boldsymbol{\lambda}^\top\mathcal{D}\delta u\Big|_{t_0}^{t_e} = 0.
\end{aligned}$$

Since we require each variation to be zero individually, we obtain the equations (5.13) – (5.15) and two boundary conditions at t_0 and t_e, respectively. We have a fixed value $\boldsymbol{z}(t_0) = \boldsymbol{z}_0$ such that $\delta\boldsymbol{z}(t_0) = 0$ and hence, we have no initial condition $\boldsymbol{\lambda}(t_0)$. In contrast, $\boldsymbol{z}(t_e)$ is arbitrary such that $\delta\boldsymbol{z}(t_e) \neq 0$ which results in the terminal condition $\boldsymbol{\lambda}(t_e) = 0$. Since we want to accept arbitrary initial values $u(t_0)$ and $\boldsymbol{\lambda}(t_0)$ is also free, we obtain an additional condition $\boldsymbol{\lambda}^\top(t_0)\mathcal{D} = 0$.

Equations (5.13) – (5.15) form the so-called optimality system which is seldom solved explicitly. Instead, the state equation (5.13) and the adjoint equation (5.14) are solved by integrating forwards and backwards in time, respectively. The optimality condition (5.15) provides the derivative information $\nabla_u\overline{J^r}$ of the cost functional with respect to the control u. This information can be used to improve an initial guess for the optimal control within an iterative optimization scheme as described in Algorithm 5.1 below.

In [BCB05], the condition $\boldsymbol{\lambda}(t_e) = 0$ is the only boundary condition for the adjoint equation and the condition $\boldsymbol{\lambda}^\top(t_0)\mathcal{D}\delta u(t_0) = 0$ (cf. Remark 5.1.3) is neglected. In the computations performed here, using this optimality system causes convergence issues. The reason for this is that the gradient obtained by the adjoint approach is incorrect, cf. Figure 5.8 (a) for a comparison to a finite difference approximation. (Since the state equation is approximated very accurately, the gradient can be approximated this way.) The reason for this might be that the additional condition for the adjoint equation is neglected. However, it has been reported before [Fah00, DV01] that it is favorable to include information of the adjoint state of

Algorithm 5.1 (Adjoint-based scalar optimization)

Require: Target point $\boldsymbol{T}$, initial guess $u^{(0)}$, stopping criterion ϵ

1: **for** $i = 0, \dots$ **do**
2: Compute $\boldsymbol{z}^{(i)}$ by solving the reduced state equation (5.13) with $u^{(i)}$
3: Solve the adjoint equation (5.14) in backwards time with $u^{(i)}$, $\boldsymbol{z}^{(i)}$, and $\partial\overline{J^r}/\partial\boldsymbol{z}^{(i)}$
4: Compute the gradient $\nabla_u\overline{J^r}^{(i)}$ by evaluating the optimality condition (5.15) with $u^{(i)}$, $\boldsymbol{z}^{(i)}$, $\boldsymbol{\lambda}^{(i)}$, $\partial\overline{J^r}/\partial\boldsymbol{z}^{(i)}$ and $\partial\overline{J^r}/\partial u^{(i)}$
5: **if** $\|\nabla_u\overline{J^r}^{(i)}\|_{L^2} \leq \epsilon$ **then**
6: STOP
7: **else**
8: Update the control: $u^{(i+1)} = u^{(i)} + a^{(i)}d^{(i)}$, where $a^{(i)}$ is the step length satisfying the Armijo rule, computed via backtracking [NW06], and $d^{(i)}$ is the conjugate gradient direction $d^{(i)} = -\nabla_u\overline{J^r}^{(i)} + \beta^i d^{(i-1)}$, $d^{(0)} = \nabla_u\overline{J^r}^{(0)}$ with

$$\beta^{(i)} = \frac{\nabla_u\overline{J^r}^{(i)} \cdot \nabla_u\overline{J^r}^{(i)}}{\nabla_u\overline{J^r}^{(i-1)} \cdot \nabla_u\overline{J^r}^{(i-1)}}$$

9: **end if**
10: **end for**

the PDE in the ROM. In fact, the adjoint-based gradient can often be inaccurate or even incorrect if solely based on information about the state. Consequently, it is difficult to determine whether the convergence problems stem from the missing boundary condition or are due to neglecting adjoint information of the high-fidelity system. To further investigate this, a second optimality system based on an alternative formulation of the reduced model is derived. Following [Rav00], we define an augmented state $(\boldsymbol{z}, u)^\top$ and introduce a new control $v \in L^2((t_0, t_e); \mathbb{R})$, where $v(t) = \dot{u}(t)$. It is known from optimal control theory that if a dynamical system depends linearly on the control, the control is bounded by box constraints and does not appear in the cost functional, the solution is of *bang bang* type and might include singular arcs [Ger12], see also Remark 5.1.2. This is exactly the situation for the second formulation and since the computation of such solutions is numerically challenging, a regularization term is added to the second objective in (R-MOCP), i.e. $J_2^r = \ell\|u\|_{L^2}^2 + \beta\|v\|_{L^2}^2$, where β is a small number (here: $\beta = 1e^{-5}$). The optimality system can be derived in an analogous way to the one described above in Remark 5.1.3, where the adjoint states are now denoted by $\boldsymbol{\lambda} \in H^1((t_0, t_e); \mathbb{R}^\ell)$ and

$\mu \in H^1((t_0, t_e); \mathbb{R})$:

$$\dot{\boldsymbol{z}}(t) = \mathcal{A} + \mathcal{B}\boldsymbol{z}(t) + \mathcal{C}(\boldsymbol{z}(t)) + \mathcal{D}v(t) + (\mathcal{E} + \mathcal{F}\boldsymbol{z}(t))u(t) + \mathcal{G}u^2(t), \tag{5.16}$$

$$\dot{u}(t) = v(t), \tag{5.17}$$

$$\dot{\boldsymbol{\lambda}}(t) = -\frac{\partial \overline{J^r}}{\partial \boldsymbol{z}}(t) - \left(\mathcal{B} + \frac{\partial \mathcal{C}(\boldsymbol{z}(t))}{\partial \boldsymbol{z}(t)} + \mathcal{F}u(t)\right)^\top \boldsymbol{\lambda}, \tag{5.18}$$

$$\dot{\mu}(t) = -\frac{\partial \overline{J^r}}{\partial u}(t) - (\mathcal{E} + \mathcal{F}\boldsymbol{z}(t) + 2\mathcal{G}u(t))^\top \boldsymbol{\lambda}, \tag{5.19}$$

$$0 = \frac{\partial \overline{J^r}}{\partial v}(t) + \mathcal{D}^\top \boldsymbol{\lambda}(t) + \mu(t) =: \nabla_v \overline{J^r}(t), \tag{5.20}$$

with the Fréchet derivatives:

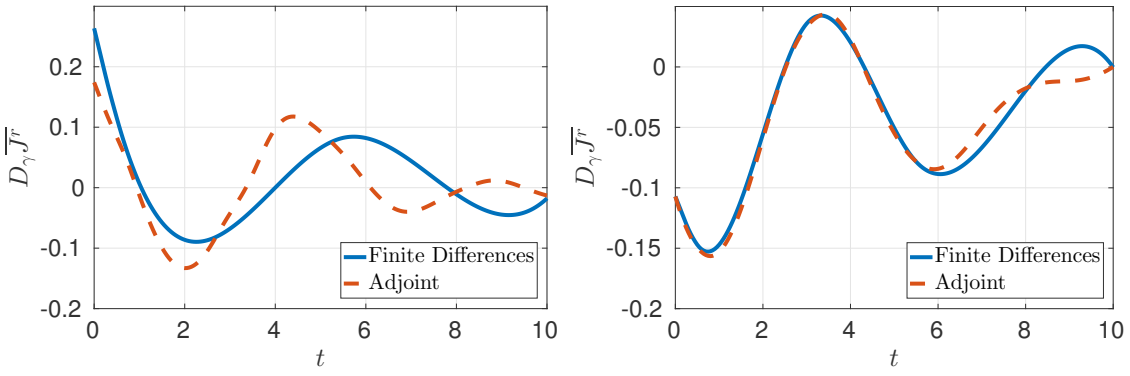
$$\frac{\partial \overline{J^r}}{\partial u}(t) = 4\ell \left(\ell \int_{t_0}^{t_e} u^2(t)\,\mathrm{d}t + \beta \int_{t_0}^{t_e} v^2(t)\,\mathrm{d}t - T_2\right) u(t),$$

$$\frac{\partial \overline{J^r}}{\partial v}(t) = 4\beta \left(\ell \int_{t_0}^{t_e} u^2(t)\,\mathrm{d}t + \beta \int_{t_0}^{t_e} v^2(t)\,\mathrm{d}t - T_2\right) v(t),$$

$\partial \overline{J^r}/\partial z_j$ as before and the respective boundary conditions

$$\boldsymbol{z}(t_0) = \boldsymbol{z}_0,\ \boldsymbol{\lambda}(t_e) = 0,\ \mu(t_0) = 0,\ \mu(t_e) = 0. \tag{5.21}$$

This is a boundary value problem for the state equations (5.16), (5.17) and the adjoint equations (5.18), (5.19) which can be solved by an iterative scheme. Applying



(a) (b)

Figure 5.8: Comparison of the gradient computed using the adjoint approaches with an approximation by finite differences (FD). (a) Optimality system (5.13) – (5.15) (FD: $\left(\nabla_u \overline{J^r}\right)_i^d = (\overline{J^r}(u^d + he_i) - \overline{J^r}(u^d))/h$, where he_i is the variation by h of the i^{th} entry of the discretization u^d of u). (b) Optimality system (5.16) – (5.20) (FD: $\left(\nabla_v \overline{J^r}\right)_i^d = (\overline{J^r}(v^d + he_i) - \overline{J^r}(v^d))/h$).

a shooting method (cf. Algorithm 5.2), the initial value $u(t_0)$ is computed such that $\mu(t_0) = 0$ is satisfied. This is computationally more expensive than the simple forward-backward integration in Algorithm 5.1. However, the system (5.16) – (5.20) yields strongly improved convergence and decreased sensitivity to the optimization parameters in comparison to the system (5.13) – (5.15), which is due to the improved gradient accuracy (cf. Figure 5.8 (b)).

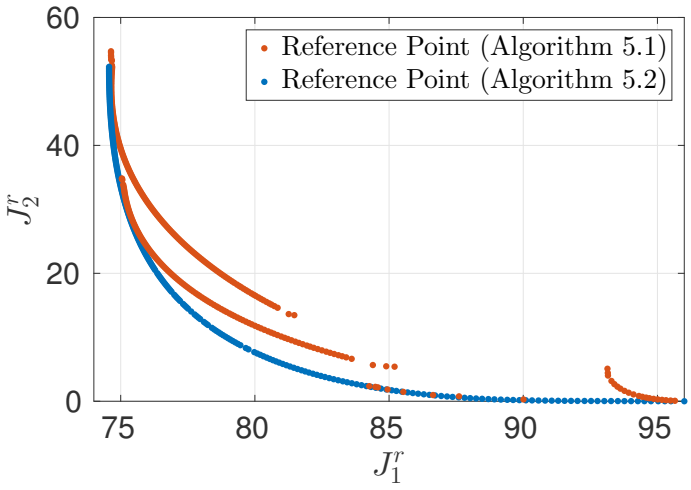


Figure 5.9: Comparison of the solutions computed with the two optimality systems (5.13) – (5.15) and (5.16) – (5.20).

The improvement is visualized in Figure 5.9, where the Pareto front based on (5.16) – (5.20) has been computed executing the reference point method (Algorithm 2.1), starting in the scalar optimum of zero control cost. In contrast to that, multiple attempts have been made with the system (5.13) – (5.15), starting at different points on the Pareto front. All of them either divert quickly from the front or stop prematurely. The final results in this section are therefore based on the system (5.16) – (5.20).

Remark 5.1.4. *As already stated, the approach based on the adjoint equation can be very sensitive. This is also the case when applying additional stabilization methods where the coefficients are manipulated to better match the state equation. Although the accuracy of the state equation can be significantly improved, this often yields severe errors in the adjoint equation and hence, the gradient. Therefore, it would be interesting to investigate direct approaches where only accuracy of the state equation is of importance.*

Algorithm 5.2 (Adjoint-based scalar optimization with shooting)

Require: Target point $\boldsymbol{T}$, initial guess $v^{(0)}$, stopping criterion ϵ

for $i = 0, \ldots$ **do**
  Shooting step: Determine $u^{(i)}(t_0)$ by solving an internal root finding problem in order to enforce the condition $\mu^{(i)}(t_0) = 0$. This step requires forward solves of (5.16), (5.17) and backward solves of (5.18), (5.19)
  Compute $\boldsymbol{z}^{(i)}$, $u^{(i)}$ by solving (5.16), (5.17) with $v^{(i)}$ and $u^{(i)}(t_0)$
  Solve the adjoint equations (5.18), (5.19) in backwards time with $v^{(i)}$, $\boldsymbol{z}^{(i)}, u^{(i)}$, $\partial\overline{J^r}/\partial\boldsymbol{z}^{(i)}$ and $\partial\overline{J^r}/\partial u^{(i)}$
  Compute the gradient $\nabla_v \overline{J^r}^{(i)}$ by evaluating the optimality condition (5.20) with $\boldsymbol{\lambda}^{(i)}$, $\mu^{(i)}$, and $\partial\overline{J^r}/\partial v^{(i)}$
  if $\|\nabla_v \overline{J^r}^{(i)}\|_{L^2} \leq \epsilon$ **then**
    STOP
  else
    Update the control: $v^{(i+1)} = v^{(i)} + a^{(i)} d^{(i)}$, with $a^{(i)}$ and $d^{(i)}$ as in Algorithm 5.1
  end if
end for

5.1.6 Results

In this section solutions of (R-MOCP) obtained with the sampling algorithm (Algorithm 2.3) and the reference point method (Algorithm 2.1) using the optimality system (5.16) – (5.20) are compared. The time interval $[t_0, t_e]$ is fixed to $[0, 10]$ which corresponds to roughly two vortex shedding cycles. In order to use the subdivision method, a sinusoidal control is introduced, i.e. $u(t) = A\sin(2\pi\omega t + \tau)$. The choice is motivated by the fact that the uncontrolled solution of the problem is also periodic. A phase shift is included to allow for controls with non-zero initial values. This way, (R-MOCP) is transformed into an MOP with $\widetilde{\boldsymbol{J}}^r : \mathbb{R}^3 \to \mathbb{R}^2$:

$$\min_{A,\omega,\tau\in\mathbb{R}} \widetilde{\boldsymbol{J}}^r(A,\omega,\tau) = \min_{A,\omega,\tau\in\mathbb{R}} \begin{pmatrix} \int_{t_0}^{t_e} \|\boldsymbol{z}\|_{L^2}^2 \,\mathrm{d}t \\ \ell \int_{t_0}^{t_e} \left(A\sin(2\pi\omega t + \tau)\right)^2 \,\mathrm{d}t \end{pmatrix}. \tag{MOP-3D}$$

Another way to reduce the dimension is to approximate u by a cubic spline with a moderate number of break points. In this case, the parameters to be optimized are the spline values at these break points. For the reference point method, u is discretized with a constant step length of $\Delta t = 0.05$ and the system (5.16) – (5.18) is solved with a fourth order Runge-Kutta scheme.

Figure 5.10 shows the box covering of the Pareto set of (MOP-3D), the respective Pareto front is shown in Figure 5.11 (a) in green. It is worth mentioning that – apart from a few spurious points at $A \approx 0$ caused by numerical errors (Figure 5.10 (a))

– the set is restricted to a small part of the frequency domain. This is comprehensible since the rotation acts against the natural dynamics of the von Kármán vortex street. The trade-off between control cost and stabilization is almost exclusively realized by changing the amplitude of the rotation. When moving towards

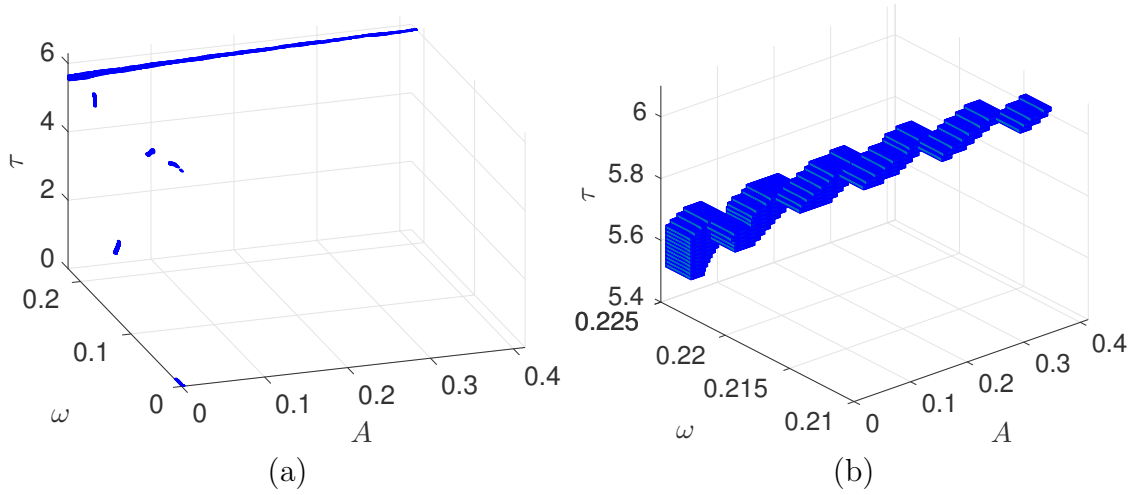
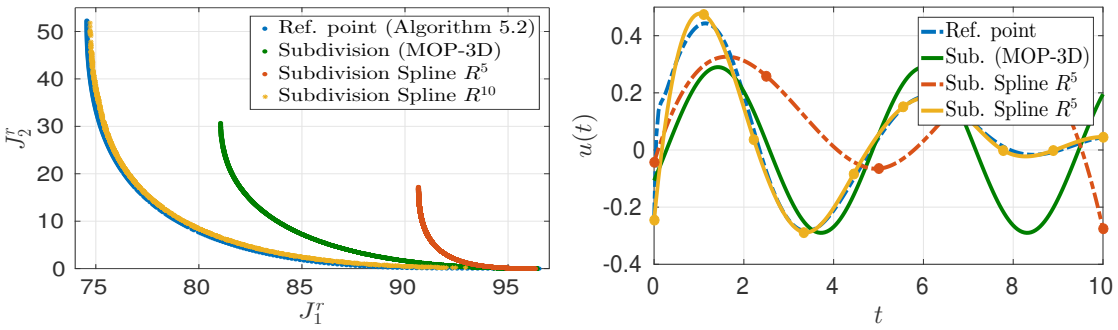


Figure 5.10: (a) Box covering of the solution of (MOP-3D), obtained with the sampling algorithm after 27 subdivision steps. (b) Only the physically relevant part is shown which is restricted to a small part of the frequency domain. The trade-off between the two objectives is mainly realized by changing the amplitude.

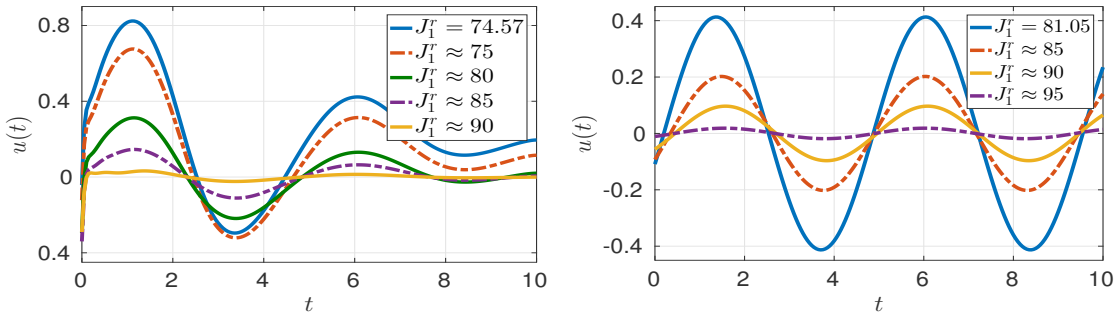
the two scalar optimal solutions, i.e. minimal fluctuations and minimal control cost, respectively, further improvements are only possible by accepting large trade-offs in the other objective which is a common phenomenon in multiobjective optimization. When designing a system, one could now accept a small increase in the main objective, i.e. flow stabilization, in order to achieve a large decrease in the control cost. This becomes more clear when looking at Figure 5.12 (b), where different optimal compromises for the control are shown. For a relatively small improvement of J_1^r from 85 to 81.05 (-5%), the control cost increases by 322%.

Figures 5.11 (a) and 5.12 also show the Pareto fronts and Pareto optimal controls computed with the sampling algorithm using two different spline approximations (five and ten breakpoints) on the one hand and with the reference point method using arbitrary controls on the other hand. We observe that a spline with five breakpoints is too restrictive to obtain control functions of acceptable quality. When using ten breakpoints on the other hand, the Pareto front clearly surpasses the solution of the sinusoidal control. However, it is numerically expensive to compute since a large number of sample points has to be evaluated to cover ten-dimensional boxes. The best solution is computed with the reference point method. The improvement over the spline-based solution is relatively small, the reason being that the controls are



(a) (b)

Figure 5.11: (a) Comparison of Pareto fronts of (R-MOCP) obtained with different solution methods. (b) Comparison of Pareto optimal controls with equal cost ($J_2^r = 15$).

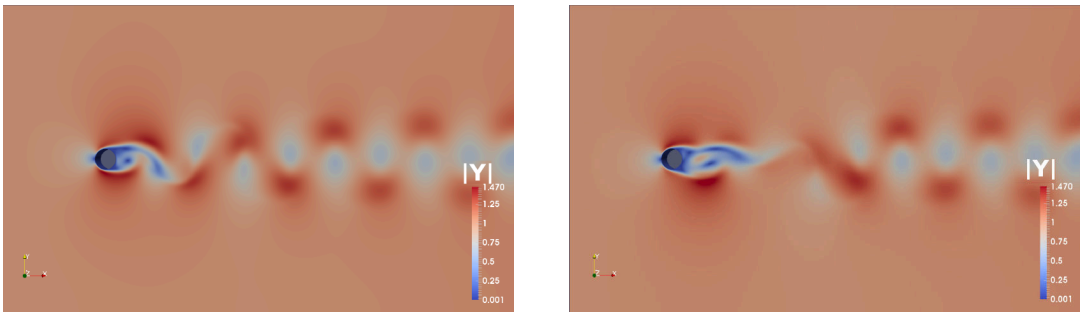


(a) (b)

Figure 5.12: Different Pareto optimal points of (R-MOCP) computed with the reference point method (a) and of (MOP-3D) computed with the sampling algorithm (b).

almost similar (see Figure 5.11 (b), where solutions of the different methods with the same control cost $J_2^r = 15$ are compared). When considering scenarios with time intervals longer than ten seconds, the spline dimension has to be further increased, leading to again much higher computational cost whereas the cost for the reference point method increases only linearly in time. This is due to the fact that only the number of time steps in the forward and backward integration (5.16) – (5.18) is affected. Finally, Figure 5.13 shows two solutions of (5.1a) – (5.1f) with controls computed with the reference point method, corresponding to different values of the control cost J_2^r and thus, different degrees of stabilization. Due to the relatively short integration time of $t_e = 10$ seconds, changes are only apparent closely past the cylinder. Nevertheless, increased stabilization can be observed when allowing higher control cost.

In order to evaluate the quality of the solution obtained with the ROM approach,



(a) (b)

Figure 5.13: High fidelity solution at $t = 10$ with two different Pareto optimal solutions computed with the reference point method. (a) Low control cost ($J_2^r = 1.84$), almost no reduction of fluctuations ($J_1^r = 85$). (b) Larger control cost ($J_2^r = 30.12$), stronger reduction of fluctuations ($J_1^r = 75$). Due to the short integration time, the effect is only visible in the vicinity of the cylinder.

the cost function is evaluated using the system state obtained from a PDE evaluation, which is shown in Figure 5.14. Since errors due to the ROM are only introduced in the first objective, the Pareto optimal points are shifted horizontally. In general, we observe a good agreement between the solutions, the error being less than 4% for all solutions that have been evaluated. However, particularly in regions with a steep gradient $\partial J_2^r / \partial J_1^r$ in the Pareto front, this small error can result in a different shape of the front. Note that the two points with the highest values for J_2^r are dominated by the next lower point. This gives a strong motivation for controlling the error of the ROM (cf. Section 5.2) and investigating the influence of the inaccuracies in the gradient obtained by the adjoint approach. Here, the sampling algorithm has the clear advantage over the reference point method since it only depends on the accuracy of the state equation.

Table 5.1: Comparison of CPU times of different methods (Subdivision algorithm in parallel: CPU time = Number of CPUs times wall clock time).

	# boxes / points	# Function evaluations	# Adjoint evaluations	CPU time [h]
Subdivision (MOP-3D)	1383	$\approx 8.3 \cdot 10^6$	0	≈ 1132
Subdivision Spline $\mathbb{R}^5$	6057	$\approx 11.1 \cdot 10^6$	0	≈ 1512
Subdivision Spline $\mathbb{R}^{10}$	1880	$\approx 42 \cdot 10^6$	0	≈ 5875
Reference point (5.16) – (5.20)	256	26781	1002	≈ 12.2

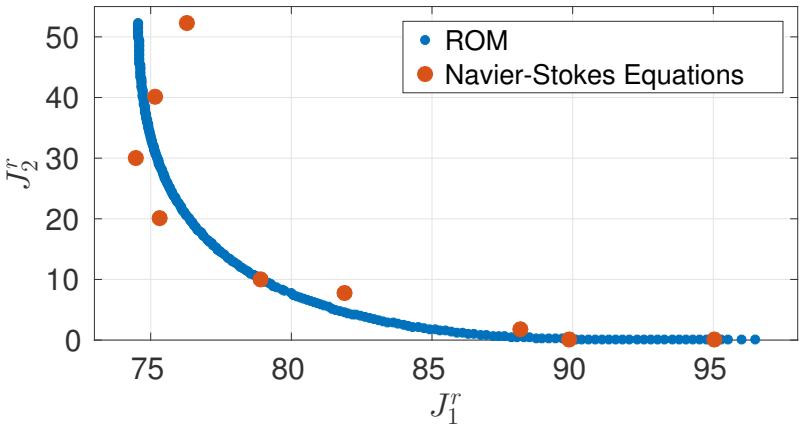


Figure 5.14: Comparison of the objective values based on the state of the ROM and the PDE solution. Since there is no error in J_2^r, the points are shifted horizontally. Although the overall agreement is acceptable, the PDE-based Pareto front appears to have a different shape, in particular for solutions with large control cost.

From a computational point of view, the reference point method is significantly faster, cf. Table 5.1.6. This is not surprising since in order to have a good representation of a box, a large number of sample points needs to be evaluated. This is already numerically expensive in 3D and more so for a spline-based approximation, i.e. we need to pay a price for global optimality and the derivative free approach. For future work, it is therefore advisable to utilize the gradient-based version of the subdivision algorithm in order to decrease the computational cost, provided that a sufficient accuracy of the reduced gradients can be achieved. Another question that needs to be addressed is how to control for error of the ROM and the respective gradient in order to guarantee optimality for the PDE-based problem. This will be the subject of the next section.

5.2 A Trust-Region Algorithm for Multiobjective Optimal Control of Nonlinear PDEs

In the previous section two fundamentally different algorithms for MOCPs have been combined with reduced order modeling in order to take PDE-constrained into account. Although the accuracy of the results is satisfactory, it is not possible to quantify the error which may cause problems when considering more complicated problem setups which have been investigated less intensively than the von Kármán vortex street. Therefore, this section is dedicated to the second approach of implementing POD-based ROMs in optimal control (cf. Section 2.3.3), namely by trust

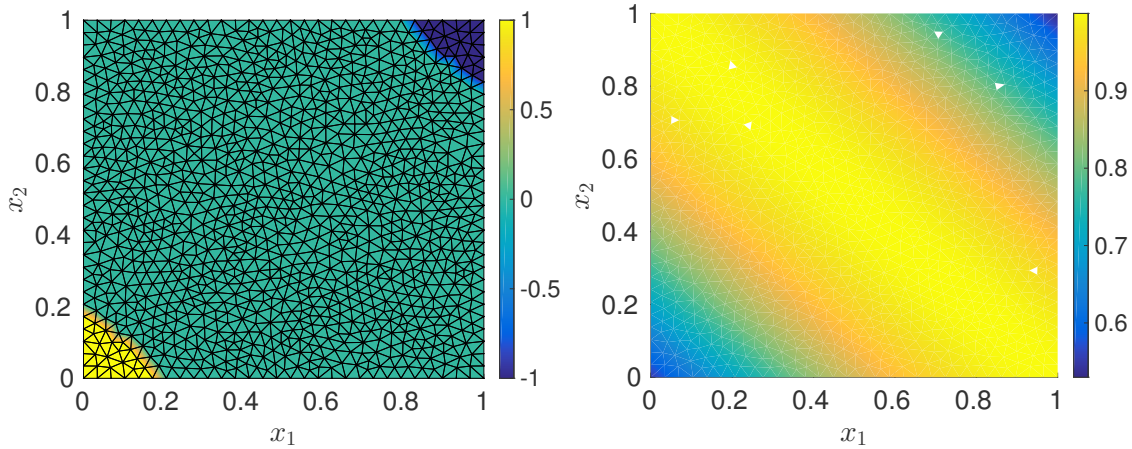
region methods [Fah00] (see also [QGVW16, YM13, RTV17]). This concept is known as *Trust Region POD (TRPOD)*. In [BC08], TRPOD has been utilized for single objective optimal control of the Navier-Stokes equations, more precisely of the von Kármán vortex street. However, the control law there is sinusoidal, which yields an optimization problem with $\boldsymbol{u} \in \mathbb{R}^2$. Moreover, the solution on the limit cycle is considered whereas here, we are interested in transient phenomena which poses more difficulties for the ROM.

5.2.1 Problem Setting

In order to compare the ROM-based to the PDE-based solution, a much simpler problem is considered here, i.e. a linear heat equation on a rectangular domain Ω (cf. Figure 5.15 (a)) with homogeneous Neumann boundary conditions and distributed control:

$$\begin{aligned} \dot{y}(\boldsymbol{x},t) - \Delta y(\boldsymbol{x},t) &= \sum_{j=1}^{2} u_j(t)\chi_j(\boldsymbol{x}), && (\boldsymbol{x},t) \in \Omega \times (t_0,t_e], \\ \frac{\partial y}{\partial \boldsymbol{n}}(\boldsymbol{x},t) &= 0, && (\boldsymbol{x},t) \in \Sigma \times (t_0,t_e], \\ y(\boldsymbol{x},t_0) &= y_0(\boldsymbol{x}) && \boldsymbol{x} \in \Omega. \end{aligned} \tag{5.22}$$

Here, $y \in H^1((t_0,t_e);L^2(\Omega)) \cap L^2((t_0,t_e);H^2(\Omega))$ is the temperature distribution and $\boldsymbol{u} \in L^2((t_0,t_e);\mathbb{R}^2)$ is the control. The two objectives are to simultaneously



(a) (b)

Figure 5.15: Domain Ω discretized by a finite element mesh. (a) In the lower left and the upper right corner, the two shape functions χ_j are shown in yellow and blue, respectively. (b) Desired state at $t = 5.7$.

minimize the distance to the *desired state* (cf. Figure 5.15 (b))

$$y_d(\boldsymbol{x}, t) = \sin(x_1 + x_2 - t), \tag{5.23}$$

and the control cost, which results in the bicriterial optimal control problem:

$$\min_{\boldsymbol{u}\in L^2} \boldsymbol{J}(\boldsymbol{u}) = \min_{\boldsymbol{u}\in L^2} \begin{pmatrix} \int_{t_0}^{t_e} \|y(\cdot,t) - y_d(\cdot,t)\|_{L^2}^2 \,\mathrm{d}t \\ \sum_{j=1}^{2} \|u_j\|_{L^2}^2 \end{pmatrix}, \tag{5.24}$$

where $\boldsymbol{J} : L^2 \to \mathbb{R}^2$ is again the reduced representation for $\boldsymbol{J}(y, \boldsymbol{u})$ as introduced in Sections 2.2.1 and 5.1.3.

Since it is necessary to measure the improvement of the objective function in the TRPOD approach, we scalarize the objective by means of the reference point method (see also Sections 2.1.4 and 5.1.5), where the distance to an infeasible target $\boldsymbol{T}$ has to be minimized:

$$\min_{\boldsymbol{u}\in L^2} \overline{J}(\boldsymbol{u}) = \min_{\boldsymbol{u}\in L^2} \|\boldsymbol{T} - \boldsymbol{J}(\boldsymbol{u})\|_2^2. \tag{5.25}$$

In contrast to the indirect method chosen in Section 5.1, a direct method is used to solve (5.25). By the FEM discretization, the PDE (5.22) is transformed into a system of coupled ODEs which is then integrated in time using an implicit Euler scheme (see [Vol15] for details). The gradient $\nabla\overline{J}(\boldsymbol{u})$ is computed using *algorithmic differentiation* which is implemented in the software package *ADiMat* [BBL+02]. Finally, problem (5.25) is solved using the SQP solver implemented in MATLAB.

5.2.2 Reduced Order Model

The solution of (5.24) via the reference point method and problem (5.25) requires many evaluations of (5.22) which is expensive. Due to this, we again replace the model by a POD-based ROM. Similar to Section 5.1, the reduced state $\boldsymbol{z} \in H^1((t_0, t_e); \mathbb{R}^\ell)$ is obtained by a Galerkin projection of y onto the POD basis $\{\psi_i\}_{i=1}^{\ell}$ computed from FEM data with the method of snapshots. Hence, y can be expressed in terms of $\boldsymbol{z}$:

$$y(\boldsymbol{x}, t) \approx \sum_{i=1}^{\ell} z_i(t)\psi_i(\boldsymbol{x}). \tag{5.26}$$

The multiobjective optimal control problem as well as the scalarized problem are formulated with respect to the reduced state accordingly:

$$\min_{\boldsymbol{u}\in\mathcal{U}_{\delta^{(i)}}} \boldsymbol{J}^r(\boldsymbol{u}) = \min_{\boldsymbol{u}\in\mathcal{U}_{\delta^{(i)}}} \begin{pmatrix} \int_{t_0}^{t_e} \left\| \left(\sum_{j=1}^{\ell} z_j(t)\psi_j\right) - y_d(\cdot,t) \right\|_{L^2}^2 \,\mathrm{d}t \\ \sum_{j=1}^{2} \|u_j\|_{L^2}^2 \end{pmatrix}, \tag{5.27}$$

and

$$\min_{\boldsymbol{u}\in\mathcal{U}_{\delta^{(i)}}} \overline{J^r}(\boldsymbol{u}) = \min_{\boldsymbol{u}\in\mathcal{U}_{\delta^{(i)}}} \|\boldsymbol{T} - \boldsymbol{J}^r(\boldsymbol{u})\|_2^2. \tag{5.28}$$

Note that a constraint set $\mathcal{U}_{\delta^{(i)}}$ has been introduced for the control. This is due to the fact that the distance between the control $\boldsymbol{u}$ and the reference control $\boldsymbol{u}^{(i)}$ at which the data for the i^{th} POD basis has been collected is bounded by the *trust region radius* $\delta^{(i)}$.

5.2.3 Trust Region Algorithm

In this section we will prove that a minimizer $\boldsymbol{u}^*$ of (5.28) is Pareto optimal with respect to the PDE-constrained multiobjective optimal control problem (5.24). This follows from the proofs for the TRPOD algorithm (Theorem 5.2.1) and the reference point method (Theorem 2.1.17). The trust region framework developed by Fahl [Fah00] is summarized below.

We begin by stating some assumptions on the PDE-based as well as the ROM-based scalar objective functions. We make standard assumptions for problem (5.25) such that a local minimizer exists:

(A1) The scalarized objective functional $\overline{J}$ is continuously differentiable,

(A2) $\overline{J}$ is bounded below,

(A3) $\nabla\overline{J}$ is Lipschitz continuous on $\mathcal{U}$.

We assume the following for the reduced scalar objective function $\overline{J^r}$:

(B1) $\overline{J^r}$ is continuously differentiable on an open convex set containing the trust region $\mathcal{U}_{\delta^{(i)}} := \{\boldsymbol{u} \in L^2((t_0,t_e);\mathbb{R}^2) \mid \|\boldsymbol{u} - \boldsymbol{u}^{(i)}\| < \delta^{(i)}\}$,

(B2) $\nabla\overline{J^r}(\boldsymbol{u}^{(i)})$ approximates $\nabla\overline{J}(\boldsymbol{u}^{(i)})$, i.e.

$$\frac{\|\nabla\overline{J}(\boldsymbol{u}^{(i)}) - \nabla\overline{J^r}(\boldsymbol{u}^{(i)})\|}{\|\nabla\overline{J^r}(\boldsymbol{u}^{(i)})\|_{L^2}} \leq \zeta$$

for a given $\zeta \in (0,1)$. (This condition is known as the *Carter condition* [Car91].)

The condition (B2) introduces a bound for the accuracy of the reduced gradient.

Both the high-fidelity and the surrogate model are evaluated in the TRPOD algorithm 5.3. While the majority of function evaluations is based on the ROM to compute a minimizer within the currently valid trust region $\mathcal{U}_{\delta^{(i)}}$, the reduction $\rho^{(i)}$ of the objective with respect to the original problem is evaluated every time a minimizer has been found. Depending on the value of $\rho^{(i)}$, the descent step is either accepted (and the trust region radius adapted) or rejected.

If the assumptions (A1) – (A3) and (B1) – (B2) are satisfied, then Algorithm 5.3

Algorithm 5.3 (TRPOD Algorithm [Fah00])

Require: $0 < \eta_1 < \eta_2 < 1$ (evaluation of objective reduction), $0 < \gamma_1 \leq \gamma_2 < 1 \leq \gamma_3$ (adaptation of trust region), initial trust region radius $\delta^{(0)} > 0$, initial iterate $\boldsymbol{u}^{(0)}$, snapshot matrix $\boldsymbol{S}^{(0)}$ corresponding to $\boldsymbol{u}^{(0)}$, $i = 0$, stopping criteria ϵ_1, ϵ_2.

1: **loop**
2: Build a POD-based ROM using the snapshot matrix $\boldsymbol{S}^{(i)}$
3: Compute a descent step $\boldsymbol{s}^{(i)}$ for the problem (5.28) with $\|\boldsymbol{s}^{(i)}\| \leq \delta^{(i)}$ such that $\boldsymbol{u}^{(i)} + \boldsymbol{s}^{(i)}$ is an approximate minimizer within the trust region radius $\delta^{(i)}$
4: Solve the full model (5.22) with $\boldsymbol{u}^{(i)} + \boldsymbol{s}^{(i)}$ and compute the gradient $\nabla J\left(\boldsymbol{u}^{(i)} + \boldsymbol{s}^{(i)}\right)$ and the snapshot matrix $\boldsymbol{S}^{(i+1)}$
5: **if** $\|\nabla J\left(\boldsymbol{u}^{(i)} + \boldsymbol{s}^{(i)}\right)\|_{L^2} \leq \epsilon_1$ or $\|\boldsymbol{s}^{(i)}\|_{L^2} \leq \epsilon_2$ **then**
6: STOP
7: **end if**
8: Compare the reduction of the objective function values using the the reduced order model (5.28) and the full solution (5.25):

$$\rho^{(i)} = \frac{\overline{J^r}\left(\boldsymbol{u}^{(i)}\right) - \overline{J^r}\left(\boldsymbol{u}^{(i)} + \boldsymbol{s}^{(i)}\right)}{\overline{J}\left(\boldsymbol{u}^{(i)}\right) - \overline{J}\left(\boldsymbol{u}^{(i)} + \boldsymbol{s}^{(i)}\right)}$$

9: **if** $\rho^{(i)} \geq \eta_2$ **then**
10: Set $\boldsymbol{u}^{(i+1)} = \boldsymbol{u}^{(i)} + \boldsymbol{s}^{(i)}$ and choose $\delta^{(i+1)} \in [\delta^{(i)}, \gamma_3\delta^{(i)}]$ (increase TR)
11: **else if** $\eta_2 > \rho^{(i)} \geq \eta_1$ **then**
12: Set $\boldsymbol{u}^{(i+1)} = \boldsymbol{u}^{(i)} + \boldsymbol{s}^{(i)}$ and choose $\delta^{(i+1)} \in [\gamma_2\delta^{(i)}, \delta^{(i)})$ (decrease TR)
13: **else if** $\rho^{(i)} < \eta_1$ **then**
14: Set $\boldsymbol{u}^{(i+1)} = \boldsymbol{u}^{(i)}$ and choose $\delta^{(i+1)} \in [\gamma_1\delta^{(i)}, \gamma_2\delta^{(i)}]$ (strongly decrease TR)
15: **end if**
16: Set $i = i + 1$
17: **end loop**

efficiently yields a sequence $\{\boldsymbol{u}^{(i)}\}$ of controls that converges to a local minimum of the PDE-constrained problem (5.25). This is stated in the following theorem:

Theorem 5.2.1 ([Fah00])**.** *Consider the scalarized problem (5.25) and let $\overline{J}$ satisfy the standard assumptions (A1) – (A3). Let $\{\boldsymbol{u}^{(i)}\}$ be a sequence computed by Algorithm 5.3, where every ROM satisfies the assumptions (B1) and (B2) for some $\zeta \in (0, 1-\eta_2)$. Define by*

$$\omega(J, \boldsymbol{u}, \boldsymbol{s}) = \frac{2}{\|\boldsymbol{s}\|_{L^2}^2} \left(\overline{J}(\boldsymbol{u}+\boldsymbol{s}) - \overline{J}(\boldsymbol{u}) - \left(\nabla \overline{J}(\boldsymbol{u}), \boldsymbol{s}\right)\right), \tag{5.29}$$

a measure for the curvature of $\overline{J}(\boldsymbol{u})$ and assume that $|\omega(\overline{J}, \boldsymbol{u}, \boldsymbol{s})| \leq c_\omega$ for some constant $c_\omega > 0$ in every iteration of Algorithm 5.3. Then

$$\lim_{i\to\infty} \|\nabla \overline{J^r}(\boldsymbol{u}^{(i)})\|_{L^2} = 0,$$

from which follows

$$\lim_{i\to\infty} \|\nabla \overline{J}(\boldsymbol{u}^{(i)})\|_{L^2} = 0.$$

Note that the curvature condition (5.29) corresponds to the *sufficient decrease condition* in the classical trust region concept, where a general non-linear model is approximated by a quadratic model [Pow75].

We can now combine the above result with the convergence result for the reference point method (Theorem 2.1.17). To this end, we first have to ensure that problem (5.25) has a unique minimizer:

Theorem 5.2.2 ([BBV16])**.** *For any $\boldsymbol{T} \in \mathbb{R}^2$ problem (5.25) has a unique solution.*

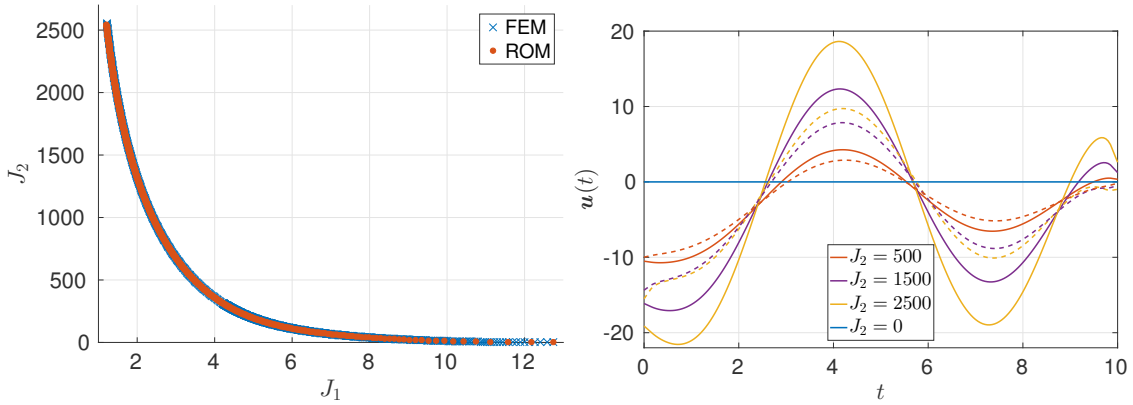
The proof of this result follows from the convexity of the objective function $\overline{J}(\boldsymbol{u})$ and Theorem 2.14 in [Trö10], see [BBV16] for details. Finally, a combination of Theorems 2.1.17 (reference point method), 5.2.1 (TRPOD) and 5.2.2 yields the desired result.

Theorem 5.2.3. *Let the target point $\boldsymbol{T}$ be less than the utopian point $\boldsymbol{J}^*$, i.e. $\boldsymbol{T} \leq_p \boldsymbol{J}^*$. Then a minimizer $\boldsymbol{u}^*$ of (5.28) is locally Pareto optimal with respect to the PDE-constrained multiobjective optimal control problem (5.24).*

Proof. By Theorem 5.2.2, problem (5.25) possesses a unique minimizer $\boldsymbol{u}^*$. According to Theorem 5.2.1, this minimizer can be computed by solving problem (5.28). Finally, $\boldsymbol{u}^*$ is Pareto optimal with respect to (5.24) by Theorem 2.1.17. □

5.2.4 Results

In order to validate the results, both (5.24) and (5.27) are solved with the reference point method. The respective scalar problems (5.25) and (5.28) are solved using an SQP method (see e.g. [NW06]), where the gradients are approximated via algorithmic differentiation. In contrast to the last section, the dynamical system is faster to solve and possesses less complicated dynamics. This allows us to compute the entire Pareto front using a FEM discretization in order to compare it to the ROM-based solution.



(a) (b)

Figure 5.16: (a) Pareto fronts computed with the reference point method, where (5.24) has been solved using a FEM discretization and (5.27) with the TRPOD approach. (b) Several Pareto optimal controls. The solid lines represent u_1 and the dashed lines u_2.

The two Pareto fronts are shown in Figure 5.16 (a) and we see that they agree very well. Due to the very long computing time for the PDE-constrained problem, the computation has been stopped and picked up again at different stages such that we do not observe point-wise agreement. Moreover, numerical divergence between the FEM and the POD-based solution is introduced by the ROM approach [Ban16]. Nevertheless, the TRPOD algorithm clearly yields Pareto optimal solutions. Figure 5.16 (b) shows several Pareto optimal controls. Similar to Section 5.1, we observe that the compromise between tracking the reference solution and control cost is realized almost exclusively by varying the amplitude.

In [BBV16, BBV17], the reference point method has been used to solve PDE-constrained MOCPs via reduced order modeling as well. Instead of adopting the TRPOD approach for general PDE-constrained problems, the authors make use of the problem structure and derive error estimates for the reduced order optimal control problem. They observe that larger values of the cost functional in (5.28),

i.e. increased distances between the targets and feasible points, lead to improved error estimates. The same behavior can be observed in the TRPOD approach. Here, Algorithm 5.3 exhibits improved convergence behavior since the values for $\rho^{(i)}$ are larger.

When comparing the computational cost, the efficiency of the TRPOD approach becomes evident, cf. Figure 5.17 (a), where the computing time for the two approaches is compared. On average, we observe a speed-up by a factor of over 60. The reason is that in the TRPOD algorithm, almost all Pareto optimal solutions can be computed with only a single evaluation of $\overline{J}(\boldsymbol{u})$ and $\nabla\overline{J}(\boldsymbol{u})$, cf. Figure 5.17 (b). The reason for this is that two consecutive Pareto optimal solutions differ only slightly. Hence, the final evaluation of the high-fidelity model in the TRPOD algorithm (which is necessary to validate the stopping criterion) is used to generate the POD basis for the solution of the consecutive scalar problem. This problem can again be solved within one trust-region and a single evaluation of the full model is required to verify the stopping criterion.

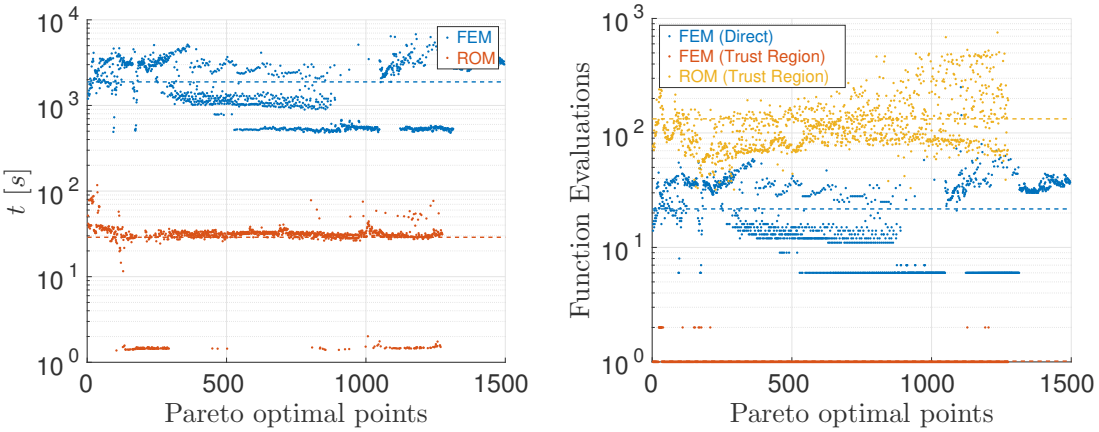


Figure 5.17: (a) Comparison of the computing times for solving (5.24) and (5.27). On average (dashed lines), the TRPOD method is over 60 times faster. (b) Comparison of the number of function evaluations per scalar problem in the respective approaches.

Since all scalarization methods have difficulties with a large number of objectives, the approach presented here is limited to a few objectives, ideally two. However, when using the results from Chapter 4, the skeleton of a Pareto set with many objectives can be computed efficiently.

5.3 Extension of the Subdivision Algorithm to Inexact Models

In the previous section the TRPOD approach has been extended to multiple objectives using scalarization. This results in efficient algorithms for MOCPs with a low number of objectives. However, in situations where globality is important or where scalarization methods may fail due to the Pareto set possessing a complicated structure, set-oriented methods are superior. Therefore, the subdivision (Algorithm 2.2) and the sampling algorithm (Algorithm 2.3) are extended to inexact function and gradient information in this section. This way, ROMs with certified error bounds can be utilized to accelerate the computation of the Pareto set.

Besides the problem of the computational effort and the necessity to apply reduced order modeling, many real world problems possess uncertainties for various reasons such as the ignorance of the exact underlying dynamical system or unknown material properties. Using surrogate models in order to decrease the computational effort also introduces errors which can often be quantified (see e.g. [TV09]). A similar concept exists in the context of evolutionary computation, where *Surrogate-Assisted Evolutionary Computation* (see e.g. [Jin11] for a survey) is often applied in situations where the exact model is too costly to solve. In this setting, the set of *almost Pareto optimal points* has to be computed (see also [Whi86], where the concept of ϵ *efficiency* has been introduced). Many researchers are investigating problems related to uncertainty quantification and several authors have addressed multiobjective optimization problems with uncertainties. In [Hug01] and in [Tei01], probabilistic approaches to MOPs with uncertainties have been derived independently. In [DG05, DMM05, BZ06, SM08], evolutionary algorithms for problems with uncertain or noisy data have been developed in order to compute robust approximations of the Pareto set. In [SCTT08], stochastic search has been used, cell mapping techniques have been applied in [HSS13] and in [EW07], the weighted sum method has been extended to uncertainties.

Large parts of this section have previously appeared in [PD17] to which the author has made substantial contributions.

5.3.1 Problem setting

The problems considered in this section are unconstrained MOPs, i.e. the control parameter $\boldsymbol{u} \in \mathbb{R}^n$ is finite-dimensional. This leads to

$$\min_{\boldsymbol{u}\in\mathbb{R}^n} \boldsymbol{J}(\boldsymbol{u}) = \min_{\boldsymbol{u}\in\mathbb{R}^n} \begin{pmatrix} J_1(\boldsymbol{u}) \\ \vdots \\ J_k(\boldsymbol{u}) \end{pmatrix}, \tag{MOPu}$$

which is equivalent to problem (MOP) introduced in Chapter 2 with $\mathcal{U} = \mathbb{R}^n$. The corresponding KKT conditions are hence

$$\begin{aligned} \sum_{i=1}^{k} \alpha_i \nabla J_i(\boldsymbol{u}^*) &= 0, \\ \boldsymbol{\alpha} &\geq_p \mathbf{0}, \\ \sum_{i=1}^{k} \alpha_i &= 1, \end{aligned} \tag{KKTu}$$

where we assume the objectives $J_i(\boldsymbol{u})$ to be continuously differentiable.

Suppose now that we only have approximations $J_i^r(\boldsymbol{u}), \nabla J_i^r(\boldsymbol{u})$ of the objectives $J_i(\boldsymbol{u})$ and the gradients $\nabla J_i(\boldsymbol{u})$, $i = 1, \ldots, k$, respectively. To be more precise, we assume that

$$J_i^r(\boldsymbol{u}) = J_i(\boldsymbol{u}) + \bar{\kappa}_i, \qquad \|J_i^r(\boldsymbol{u}) - J_i(\boldsymbol{u})\|_2 = \bar{\kappa}_i \leq \kappa_i, \tag{5.30}$$

$$\nabla J_i^r(\boldsymbol{u}) = \nabla J_i(\boldsymbol{u}) + \bar{\boldsymbol{\epsilon}}_i, \qquad \|\nabla J_i^r(\boldsymbol{u}) - \nabla J_i(\boldsymbol{u})\|_2 = \|\bar{\boldsymbol{\epsilon}}_i\|_2 \leq \epsilon_i, \tag{5.31}$$

where the upper bounds $\boldsymbol{\kappa}, \boldsymbol{\epsilon} \in \mathbb{R}^k$ are given. We will in the following assume that

$$\epsilon_i \leq \|\nabla J_i(\boldsymbol{u})\|_2, \ i = 1, \ldots, k, \quad \text{for all } \boldsymbol{u} \in \mathbb{R}^n,$$

since otherwise, we are already in the vicinity of the set of stationary points as will be shown in Lemma 5.3.2. Using (5.31) we can derive an upper bound for the angle between the exact and the inexact gradient by elementary geometrical considerations (see also Figure 5.18 (a)):

$$\sphericalangle(\nabla J_i(\boldsymbol{u}), \nabla J_i^r(\boldsymbol{u})) = \arcsin\left(\frac{\|\bar{\boldsymbol{\epsilon}}_i\|_2}{\|\nabla J_i(\boldsymbol{u})\|_2}\right) \leq \arcsin\left(\frac{\epsilon_i}{\|\nabla J_i(\boldsymbol{u})\|_2}\right) =: \varphi_i. \tag{5.32}$$

By φ_i we denote the largest possible angle, i.e. the "worst case". Based on this angle, one can define a condition for the inexact gradients such that the exact gradients could satisfy (KKTu) if the deviation was indeed φ_i for $i = 1, \ldots, k$. This is precisely the case when each inexact gradient deviates from the hyperplane defined

by (KKTu) at most by φ_i, see Figure 5.18 (b). Before proving this observation, an *inexact descent direction* is defined:

Definition 5.3.1. *A direction analog to (2.3) but based on inexact gradients, i.e.*

$$\boldsymbol{q}^r(\boldsymbol{u}) = -\sum_{i=1}^{k} \widehat{\alpha}_i \nabla J_i^r(\boldsymbol{u}) \quad \text{with } \widehat{\boldsymbol{\alpha}} \geq_p \mathbf{0} \text{ and } \textstyle\sum_{i=1}^{k} \widehat{\alpha}_i = 1, \tag{5.33}$$

is called inexact descent direction.

We can prove an upper bound for the norm of $\boldsymbol{q}^r(\boldsymbol{u}^*)$ when $\boldsymbol{u}^*$ satisfies the KKT condition of the exact problem:

Lemma 5.3.2. *Consider the unconstrained problem (MOPu) with inexact gradient information according to (5.31). Let $\boldsymbol{u}^*$ be a point satisfying the necessary conditions (KKTu) for the exact problem. Then the inexact descent direction $\boldsymbol{q}^r(\boldsymbol{u}^*)$ is bounded by*

$$\|\boldsymbol{q}^r(\boldsymbol{u}^*)\|_2 \leq \|\boldsymbol{\epsilon}\|_\infty.$$

Proof. Since $\boldsymbol{u}^*$ satisfies (KKTu), we have $\sum_{i=1}^{k} \widehat{\alpha}_i \nabla J_i(\boldsymbol{u}^*) = 0$. Consequently,

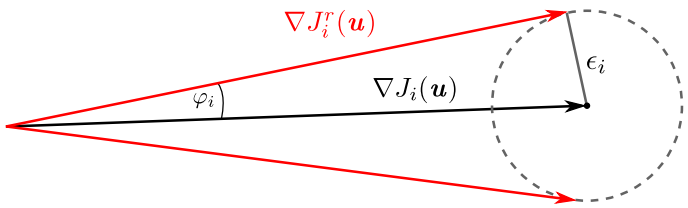
$$\sum_{i=1}^{k} \widehat{\alpha}_i \nabla J_i^r(\boldsymbol{u}^*) = \sum_{i=1}^{k} \widehat{\alpha}_i \left(\nabla J_i(\boldsymbol{u}^*) + \bar{\boldsymbol{\epsilon}}_i\right) = \sum_{i=1}^{k} \widehat{\alpha}_i \bar{\boldsymbol{\epsilon}}_i$$

and thus,

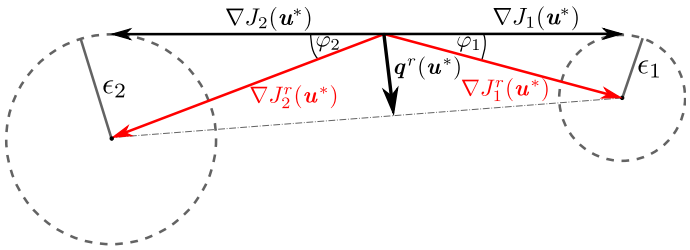
$$\begin{aligned}\left\| \sum_{i=1}^{k} \widehat{\alpha}_i \nabla J_i^r(\boldsymbol{u}^*) \right\|_2 = \left\| \sum_{i=1}^{k} \widehat{\alpha}_i \bar{\boldsymbol{\epsilon}}_i \right\|_2 &\leq \sum_{i=1}^{k} \widehat{\alpha}_i \|\bar{\boldsymbol{\epsilon}}_i\|_2 \leq \sum_{i=1}^{k} \widehat{\alpha}_i \epsilon_i \\ &\leq \sum_{i=1}^{k} \widehat{\alpha}_i \max_{i \in \{1,\dots,k\}} \epsilon_i = \|\boldsymbol{\epsilon}\|_\infty \sum_{i=1}^{k} \widehat{\alpha}_i = \|\boldsymbol{\epsilon}\|_\infty.\end{aligned}$$

□

A consequence of Lemma 5.3.2 is that in the presence of inexactness, we cannot exactly compute the set of points satisfying (KKTu). At best, we can compute the set of points determined by $\|\boldsymbol{q}^r(\boldsymbol{u})\|_2 \leq \|\boldsymbol{\epsilon}\|_\infty$. In the following section we will derive a criterion for the inexact descent direction $\boldsymbol{q}^r(\boldsymbol{u})$ which guarantees that it is also a descent direction for the exact problem.



(a)



(b)

Figure 5.18: (a) Maximal angle between the exact and the inexact gradient in dependence on the error ϵ_i. (b) Maximal angle between the inexact gradients in the situation where $\boldsymbol{u}^*$ satisfies (KKTu) (In the 2D case: $\pi - (\varphi_1 + \varphi_2)$).

5.3.2 Descent Directions in the Presence of Inexactness

The set of valid descent directions for (MOPu) is a cone defined by the intersection of all half-spaces orthogonal to the gradients $\nabla J_1(\boldsymbol{u}), \ldots, \nabla J_k(\boldsymbol{u})$ (cf. Figure 5.19 (a)), i.e. it consists of all directions $\boldsymbol{q}(\boldsymbol{u})$ satisfying

$$\sphericalangle(\boldsymbol{q}(\boldsymbol{u}), -\nabla J_i(\boldsymbol{u})) \leq \frac{\pi}{2}, \quad i = 1, \ldots, k. \tag{5.34}$$

This fact is well known from scalar optimization theory, see e.g. [NW06]. Observe that here also directions are valid for which all objectives are at least non-decreasing. In terms of the angle $\gamma_i \in [0, \pi/2]$ between the descent direction $\boldsymbol{q}(\boldsymbol{u})$ and the hyperplane orthogonal to the gradient of the i^{th} objective (cf. Figure 5.19 for an illustration), this can be expressed as

$$\gamma_i = \frac{\pi}{2} - \arccos\left(\frac{\boldsymbol{q}(\boldsymbol{u}) \cdot (-\nabla J_i(\boldsymbol{u}))}{\|\boldsymbol{q}(\boldsymbol{u})\|_2 \cdot \|\nabla J_i(\boldsymbol{u})\|_2}\right) \geq 0, \quad i = 1, \ldots, k. \tag{5.35}$$

We call the set of all descent directions satisfying (5.34) the *exact cone* $\mathcal{Q}$. If we consider inexact gradients according to (5.31) then $\mathcal{Q}$ is reduced to the *inexact cone*

$\mathcal{Q}^r$ by the respective upper bounds for the angular deviation φ_i:

$$\gamma_i \geq \varphi_i \geq 0, \quad i = 1, \ldots, k.$$

This condition can equivalently be expressed in terms of the inexact descent direction:

$$\gamma_i^r = \frac{\pi}{2} - \arccos\left(\frac{\boldsymbol{q}^r(\boldsymbol{u}) \cdot (-\nabla J_i^r(\boldsymbol{u}))}{\|\boldsymbol{q}(\boldsymbol{u})\|_2 \cdot \|\nabla J_i^r(\boldsymbol{u})\|_2}\right) \geq \varphi_i, \quad i = 1, \ldots, k. \tag{5.36}$$

This means that the angles between a descent direction and the hyperplanes defined by the inexact gradients have to be at least as large as the maximum deviation between the exact and the inexact gradients (cf. Figure 5.19 (b)). Thus, if an inexact descent direction $\boldsymbol{q}^r(\boldsymbol{u})$ satisfies (5.36), then it is also a descent direction for the exact problem.

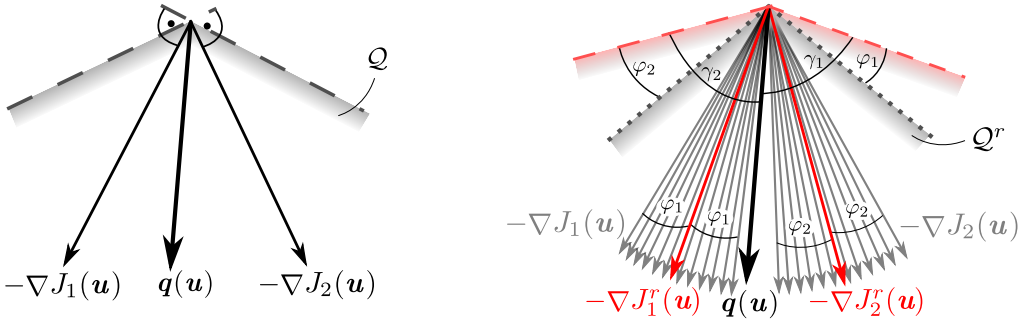


Figure 5.19: (a) Set of valid descent directions (the exact cone $\mathcal{Q}$ bounded by the dashed lines) determined by the intersection of half-spaces defined by the negative gradients. (b) Reduction of the set of valid descent directions in dependence on the error $\boldsymbol{\epsilon}$ (the inexact cone $\mathcal{Q}^r$ bounded by the dotted lines). The gray vectors represent the set of possible values for the exact gradients $\nabla J_i(\boldsymbol{u})$ and the inexact cone $\mathcal{Q}^r$ is defined by the "most aligned" (here the uppermost) realizations of $\nabla J_i(\boldsymbol{u})$.

The goal is therefore to derive an algorithm by which an inexact descent direction $\boldsymbol{q}^r(\boldsymbol{u})$ can be determined in such a way that it is also valid for the exact problem. To this end, we derive an additional criterion which is equivalent to (5.36). This is stated in the following lemma:

Lemma 5.3.3. *Consider the multiobjective optimization problem* (MOPu) *with inexact gradient information according to* (5.31). *Let* $\boldsymbol{q}^r(\boldsymbol{u})$ *be an inexact descent*

direction according to Definition 5.3.1. We assume $\|\boldsymbol{q}^r(\boldsymbol{u})\|_2 \neq 0$, $\|\nabla J_i^r(\boldsymbol{u})\|_2 \neq 0, i = 1, \ldots, k$. *Then* (5.36) *is equivalent to*

$$\widehat{\alpha}_i \geq \frac{1}{\|\nabla J_i^r(\boldsymbol{u})\|_2^2} \left(\|\boldsymbol{q}^r(\boldsymbol{u})\|_2 \epsilon_i - \sum_{\substack{j=1 \\ j \neq i}}^{k} \widehat{\alpha}_j \left(\nabla J_j^r(\boldsymbol{u}) \cdot \nabla J_i^r(\boldsymbol{u}) \right) \right), \quad i = 1, \ldots, k. \tag{5.37}$$

In particular, $\boldsymbol{q}^r(\boldsymbol{u})$ *is a descent direction for all objective functions* $J_i(\boldsymbol{u})$ *if* (5.37) *is satisfied.*

Proof. Exploiting the fact that the cosine function is strictly decreasing on the interval $[0, \pi]$, (5.36) can be transformed to

$$\frac{\boldsymbol{q}^r(\boldsymbol{u}) \cdot (-\nabla J_i^r(\boldsymbol{u}))}{\|\boldsymbol{q}^r(\boldsymbol{u})\|_2 \cdot \|\nabla J_i^r(\boldsymbol{u})\|_2} \geq \cos\left(\frac{\pi}{2} - \varphi_i\right) = \sin(\varphi_i) = \frac{\epsilon_i}{\|\nabla J_i^r(\boldsymbol{u})\|_2}.$$

Using the definition of $\boldsymbol{q}^r(\boldsymbol{u})$, this is equivalent to

$$\begin{aligned} \frac{\left(\sum_{j=1}^k \widehat{\alpha}_j \nabla J_j^r(\boldsymbol{u})\right) \cdot \nabla J_i^r(\boldsymbol{u})}{\|\boldsymbol{q}^r(\boldsymbol{u})\|_2 \, \|\nabla J_i^r(\boldsymbol{u})\|_2} &= \frac{\widehat{\alpha}_i \|\nabla J_i^r(\boldsymbol{u})\|_2^2 + \sum_{j=1, j\neq i}^k \widehat{\alpha}_j \nabla J_j^r(\boldsymbol{u}) \cdot \nabla J_i^r(\boldsymbol{u})}{\|\boldsymbol{q}^r(\boldsymbol{u})\|_2 \, \|\nabla J_i^r(\boldsymbol{u})\|_2} \\ &= \frac{\|\nabla J_i^r(\boldsymbol{u})\|_2}{\|\boldsymbol{q}^r(\boldsymbol{u})\|_2} \widehat{\alpha}_i + \frac{\sum_{j=1, j\neq i}^k \widehat{\alpha}_j \nabla J_j^r(\boldsymbol{u}) \cdot \nabla J_i^r(\boldsymbol{u})}{\|\boldsymbol{q}^r(\boldsymbol{u})\|_2 \, \|\nabla J_i^r(\boldsymbol{u})\|_2} \\ &\geq \frac{\epsilon_i}{\|\nabla J_i^r(\boldsymbol{u})\|_2} \end{aligned}$$

$$\iff \quad \widehat{\alpha}_i \geq \frac{1}{\|\nabla J_i^r(\boldsymbol{u})\|_2^2} \left(\|\boldsymbol{q}^r(\boldsymbol{u})\|_2 \epsilon_i - \sum_{\substack{j=1 \\ j \neq i}}^{k} \widehat{\alpha}_j \left(\nabla J_j^r(\boldsymbol{u}) \cdot \nabla J_i^r(\boldsymbol{u}) \right) \right).$$

□

Remark 5.3.4. *By setting* $\boldsymbol{\epsilon} = \mathbf{0}$ *(i.e.* $\boldsymbol{J}^r(\boldsymbol{u}) = \boldsymbol{J}(\boldsymbol{u})$*) in* (5.37) *and performing some elemental manipulations, we obtain the condition* (2.2) *for an exact descent*

direction:

$$\widehat{\alpha}_i \geq \frac{1}{\|\nabla J_i(\boldsymbol{u})\|_2^2} \left(- \sum_{\substack{j=1 \\ j \neq i}}^{k} \widehat{\alpha}_j \left(\nabla J_j(\boldsymbol{u}) \cdot \nabla J_i(\boldsymbol{u}) \right) \right)$$

$$\Leftrightarrow \widehat{\alpha}_i \left(\nabla J_i(\boldsymbol{u}) \cdot \nabla J_i(\boldsymbol{u}) \right) \geq \left(- \sum_{\substack{j=1 \\ j \neq i}}^{k} \widehat{\alpha}_j \left(\nabla J_j(\boldsymbol{u}) \cdot \nabla J_i(\boldsymbol{u}) \right) \right)$$

$$\Leftrightarrow -\nabla J_i(\boldsymbol{u}) \cdot \boldsymbol{q}(\boldsymbol{u}) \geq 0, \quad i = 1, \ldots, k. \tag{2.2}$$

The condition (5.37) can be interpreted as a lower bound for the "impact" of a particular gradient ∇J_i^r on the descent direction induced by the corresponding weight $\widehat{\alpha}_i$. The larger the error ϵ_i, the higher the impact of the corresponding gradient has to be in order to increase the angle between the descent direction and the hyperplane normal to the gradient. The closer a point $\boldsymbol{u}$ is to the Pareto front (i.e. for small values of $\|\boldsymbol{q}(\boldsymbol{u})\|_2$), the more confined is the region of possible descent directions. Hence, the inaccuracies gain influence until it is no longer possible to guarantee the existence of a descent direction for every objective. This is the case when the sum over the lower bounds in (5.37) exceeds one as shown in part (a) of the following result:

Theorem 5.3.5. *Consider the multiobjective optimization problem* (MOPu) *and suppose that the assumptions in Lemma 5.3.3 hold. Let*

$$\widehat{\alpha}_{\mathsf{min},i} = \frac{1}{\|\nabla J_i^r(\boldsymbol{u})\|_2^2} \left(\|\boldsymbol{q}^r(\boldsymbol{u})\|_2 \epsilon_i - \sum_{\substack{j=1 \\ j \neq i}}^{k} \widehat{\alpha}_j \left(\nabla J_j^r(\boldsymbol{u}) \cdot \nabla J_i^r(\boldsymbol{u}) \right) \right), \quad i = 1, \ldots, k. \tag{5.38}$$

Then the following statements are valid:

(a) If $\sum_{i=1}^{k} \widehat{\alpha}_{\mathsf{min},i} > 1$ *then* $\mathcal{Q}^r = \emptyset$ *and therefore it cannot be guaranteed that there exists a descent direction for all objective functions.*

(b) All points $\boldsymbol{u}$ *with* $\sum_{i=1}^{k} \widehat{\alpha}_{\mathsf{min},i} = 1$ *are contained in the set*

$$\mathcal{P}_{S,\epsilon} = \left\{ \boldsymbol{u} \in \mathbb{R}^n \;\middle|\; \left\| \sum_{i=1}^{k} \widehat{\alpha}_i \nabla J_i(\boldsymbol{u}) \right\|_2 \leq 2 \|\boldsymbol{\epsilon}\|_\infty \right\}. \tag{5.39}$$

Proof. For part (a), suppose that we have a descent direction $\boldsymbol{q}^r(\boldsymbol{u})$ for which

$\sum_{i=1}^{k} \widehat{\alpha}_{\mathsf{min},i} > 1$. Then summing up over (5.38) yields

$$
\begin{aligned}
& \sum_{i=1}^{k} \left(\frac{\|\boldsymbol{q}^r(\boldsymbol{u})\|_2}{\|\nabla J_i^r(\boldsymbol{u})\|_2^2} \epsilon_i - \frac{\sum_{j=1, j\neq i}^{k} \widehat{\alpha}_j \nabla J_j^r(\boldsymbol{u}) \cdot \nabla J_i^r(\boldsymbol{u})}{\|\nabla J_i^r(\boldsymbol{u})\|_2^2} \right) && > 1 \\
\Leftrightarrow \quad & \sum_{i=1}^{k} \left(\frac{\|\boldsymbol{q}^r(\boldsymbol{u})\|_2}{\|\nabla J_i^r(\boldsymbol{u})\|_2^2} \epsilon_i - \frac{(-\boldsymbol{q}^r(\boldsymbol{u}) - \widehat{\alpha}_i \nabla J_i^r(\boldsymbol{u})) \cdot \nabla J_i^r(\boldsymbol{u})}{\|\nabla J_i^r(\boldsymbol{u})\|_2^2} \right) && > 1 \\
\Leftrightarrow \quad & \sum_{i=1}^{k} \left(\frac{\|\boldsymbol{q}^r(\boldsymbol{u})\|_2}{\|\nabla J_i^r(\boldsymbol{u})\|_2^2} \epsilon_i - \frac{-\boldsymbol{q}^r(\boldsymbol{u}) \cdot \nabla J_i^r(\boldsymbol{u})}{\|\nabla J_i^r(\boldsymbol{u})\|_2^2} + \widehat{\alpha}_i \right) && > 1 \\
\Leftrightarrow \quad & \sum_{i=1}^{k} \left(\frac{\|\boldsymbol{q}^r(\boldsymbol{u})\|_2}{\|\nabla J_i^r(\boldsymbol{u})\|_2^2} \epsilon_i - \frac{-\boldsymbol{q}^r(\boldsymbol{u}) \cdot \nabla J_i^r(\boldsymbol{u})}{\|\nabla J_i^r(\boldsymbol{u})\|_2^2} \right) && > 1 - \sum_{i=1}^{k} \widehat{\alpha}_i = 0 \\
\Leftrightarrow \quad & \sum_{i=1}^{k} \left(\frac{\epsilon_i}{\|\nabla J_i^r(\boldsymbol{u})\|_2} - \frac{-\boldsymbol{q}^r(\boldsymbol{u}) \cdot \nabla J_i^r(\boldsymbol{u})}{\|\boldsymbol{q}^r(\boldsymbol{u})\|_2 \cdot \|\nabla J_i^r(\boldsymbol{u})\|_2} \right) && > 0 \\
\Leftrightarrow \quad & \sum_{i=1}^{k} \sin \varphi_i > \sum_{i=1}^{k} \sin \gamma_i.
\end{aligned}
$$

Since $\varphi_i, \gamma_i \in [0, \pi/2]$ for $i = 1, \ldots, k$, it follows that $\varphi_i > \gamma_i$ for at least one $i \in \{1, \ldots, k\}$. This is a contradiction to (5.36) and hence, $\mathcal{Q}^r = \emptyset$.

For part (b), we repeat the calculation from part (a) with the distinction that $\sum_{i=1}^{k} \widehat{\alpha}_{\mathsf{min},i} = 1$, and obtain $\sum_{i=1}^{k} \sin \varphi_i = \sum_{i=1}^{k} \sin \gamma_i$. This implies $\varphi_i = \gamma_i$ for $i = 1, \ldots, k$, i.e. the set of descent directions is reduced to a single valid direction. This is a situation similar to the one described in Lemma 5.3.2. Having a single valid descent direction results in the fact that there is now a possible realization of the gradients $\nabla J_i(\boldsymbol{u})$ such that each one is orthogonal to $\boldsymbol{q}^r(\boldsymbol{u})$. In this situation, $\boldsymbol{u}$ would satisfy (KKTu) and hence, $\|\boldsymbol{q}^r(\boldsymbol{u})\|_2 \leq \|\boldsymbol{\epsilon}\|_\infty$ which leads to

$$
\begin{aligned}
\left\| \sum_{i=1}^{k} \widehat{\alpha}_i \nabla J_i(\boldsymbol{u}) \right\|_2 &= \left\| \sum_{i=1}^{k} \widehat{\alpha}_i \left(\nabla J_i^r(\boldsymbol{u}) + (-\bar{\boldsymbol{\epsilon}}_i) \right) \right\|_2 = \left\| \boldsymbol{q}^r(\boldsymbol{u}) + \sum_{i=1}^{k} \widehat{\alpha}_i \left(-\bar{\boldsymbol{\epsilon}}_i \right) \right\|_2 \\
&\leq \left\| \boldsymbol{q}^r(\boldsymbol{u}) \right\|_2 + \left\| \sum_{i=1}^{k} \widehat{\alpha}_i \bar{\boldsymbol{\epsilon}}_i \right\|_2 \leq 2 \|\boldsymbol{\epsilon}\|_\infty.
\end{aligned}
$$

□

Corollary 5.3.6. *If $\sum_{i=1}^{k} \widehat{\alpha}_{\mathsf{min},i} > 1$, the minimal angle of the cone spanned by the exact gradients $\nabla J_i(\boldsymbol{u})$ is larger than for $\sum_{i=1}^{k} \widehat{\alpha}_{\mathsf{min},i} \leq 1$ (cf. Figure 5.20 (b)). Moreover, if $\|\boldsymbol{q}^r(\boldsymbol{u}^{(s)})\|_2$ is monotonically decreasing for decreasing distances of to $\mathcal{P}_{S,\mathsf{sub}}$ (i.e. for a sequence $\boldsymbol{u}^{(s)}$ with $d_h(\boldsymbol{u}^{(s)}, \mathcal{P}_{S,\mathsf{sub}}) < d_h(\boldsymbol{u}^{(s-1)}, \mathcal{P}_{S,\mathsf{sub}})$ for $s = 1, \ldots$), then part (b) of Theorem 5.3.5 also holds for $\sum_{i=1}^{k} \widehat{\alpha}_{\mathsf{min},i} \geq 1$.*

The results from Theorem 5.3.5 are visualized in Figure 5.20. In the situation where the inexactness in the gradients $\nabla J_i^r(\boldsymbol{u})$ permits the exact gradients $\nabla J_i(\boldsymbol{u})$ to satisfy (KKTu), a descent direction can no longer be computed.

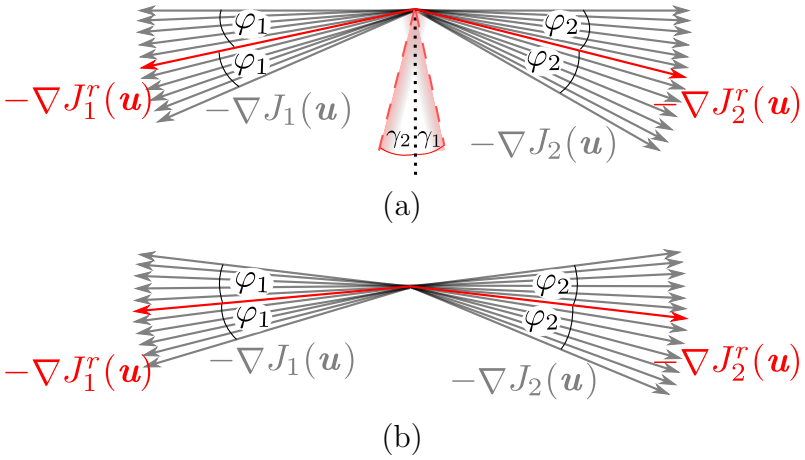


Figure 5.20: (a) Situation when a valid descent direction can no longer be computed. (b) Situation where $\sum_{i=1}^k \widehat{\alpha}_{\mathsf{min},i} > 1$.

In order to numerically compute a valid descent direction, one can compute a direction by solving the auxiliary problem (QOP) and consequently verifying that $\boldsymbol{q}^r(\boldsymbol{u})$ is indeed descending for all objectives by checking condition (5.37). If this is not the case, we can solve (QOP) again with adjusted lower bounds for $\widehat{\alpha}$ (cf. Algorithm 5.4). Alternatively, one can compute the entire set of descent directions

Algorithm 5.4 (Descent direction for inexact gradients)

Require: Inexact gradients $\nabla J_i^r(\boldsymbol{u})$, error bounds $\boldsymbol{\epsilon} \in \mathbb{R}^k$ and lower bounds $\underline{\widehat{\boldsymbol{\alpha}}} = \mathbf{0}$

loop
  Compute $\widehat{\boldsymbol{\alpha}}$ by solving (QOP) with $\widehat{\alpha}_i \in [\underline{\widehat{\alpha}}_i, 1]$, $i = 1, \dots, k$
  Compute $\widehat{\boldsymbol{\alpha}}_{\mathsf{min}}$ using (5.38)
  if $\sum_{i=1}^k \widehat{\alpha}_{\mathsf{min},i} \geq 1$ **then**
    $\boldsymbol{q}^r(\boldsymbol{u}) = 0$ ($\mathcal{Q}^r = \emptyset$)
    STOP
  else if $\widehat{\boldsymbol{\alpha}} \geq_p \widehat{\boldsymbol{\alpha}}_{\mathsf{min}}$ **then**
    $\boldsymbol{q}^r(\boldsymbol{u}) = -\sum_{i=1}^k \widehat{\alpha}_i \nabla J_i^r(\boldsymbol{u})$
    STOP
  end if
  Update lower bounds: $\underline{\widehat{\boldsymbol{\alpha}}} = \widehat{\boldsymbol{\alpha}}_{\mathsf{min}}$
end loop

[Bos12] and choose a direction from this set which satisfies (5.36) or (5.37), respectively. In all examples considered, we observe that $\boldsymbol{q}^r(\boldsymbol{u})$ is equal to $\boldsymbol{q}(\boldsymbol{u})$ for the

majority of points $\boldsymbol{u}$. This is likely due to the fact that by solving (QOP), we obtain a steepest descent like direction which very often is relatively far away from the descent cone's boundaries. Close to the set of substationary points, the lower bounds $\widehat{\boldsymbol{\alpha}}_{\mathsf{min}}$ are occasionally updated but in many cases, the line search strategy directly leads to points where $\mathcal{Q}^r = \emptyset$ without requiring adjusted bounds.

The obtained descent direction can now be utilized to efficiently compute a point which is approximately Pareto optimal. In order to compute the entire Pareto set for MOPs with inexact gradient information, the above result will be combined with the subdivision algorithm 2.2 in the next section.

5.3.3 Extension of the Subdivision Algorithm to Inexact Gradients

In this section the subdivision algorithm (see Section 2.1.5) is combined with the results from the previous section. Thereby, an algorithm for the approximation of the set of substationary points of (MOPu) with inexact function and gradient information is developed. So far, only inexactness in the descent direction has been considered where only errors in the gradients are of importance. For the computation of a descent step, a step length strategy is required where errors in the function values have to be considered as well. For this purpose, we extend the concept of *non-dominance* (2.1.5) to inexact function values[5]:

Definition 5.3.7. *Consider the multiobjective optimization problem* (MOPu) *where the objective function* $\boldsymbol{J}(\boldsymbol{u})$ *is only known approximately according to* (5.30)*. Then*

(a) a point $\boldsymbol{u}^* \in \mathbb{R}^n$ confidently dominates *a point* $\boldsymbol{u} \in \mathbb{R}^n$*, if* $\boldsymbol{J}^r(\boldsymbol{u}^*) + \boldsymbol{\kappa} \leq_p \boldsymbol{J}^r(\boldsymbol{u}) - \boldsymbol{\kappa}$ *and* $J_i^r(\boldsymbol{u}^*) + \kappa_i < J_i^r(\boldsymbol{u}) - \kappa_i$ *for at least one* $i \in 1, \ldots, k$*.*

(b) a set $\mathcal{B}^* \subset \mathbb{R}^n$ confidently dominates *a set* $\mathcal{B} \subset \mathbb{R}^n$ *if for every point* $\boldsymbol{u} \in \mathcal{B}$ *there exists at least one point* $\boldsymbol{u}^* \in \mathcal{B}^*$ *confidently dominating* $\boldsymbol{u}$*.*

(c) The set of almost non-dominated points *which is a superset of the Pareto set* $\mathcal{P}_S$ *is defined as:*

$$\mathcal{P}_{S,\kappa} = \left\{ \boldsymbol{u}^* \in \mathbb{R}^n \,\middle|\, \nexists \boldsymbol{u} \in \mathbb{R}^n \text{ which confidently dominates } \boldsymbol{u}^* \right\}. \tag{5.40}$$

Based on these considerations, we introduce an inexact dynamical system similar to (2.5) but based on inexact quantities:

$$\boldsymbol{u}^{(j+1)} = \boldsymbol{u}^{(j)} + h^{r(j)} \boldsymbol{p}^{r(j)}, \tag{5.41}$$

[5] Note that such a definition has also been introduced in [SVC09] in order to increase the number of almost Pareto optimal points and thus, the number of compromises for a decision maker.

where the direction $\boldsymbol{p}^{r(j)}$ is computed using Algorithm 5.4 (i.e. $\boldsymbol{p}^{r(j)} = \boldsymbol{q}^r(\boldsymbol{u}^{(j)})$) and the step length $h^{r(j)}$ by Algorithm 5.5 (p. 135). Therein, $h^{r(j)}$ is determined by a modified Armijo rule [NW06] such that

$$J_i^r(\boldsymbol{u}^{(j)} + h^{r(j)}\boldsymbol{p}^{r(j)}) + \kappa_i \leq J_i^r(\boldsymbol{u}^{(j-1)}) - \kappa_i + c_1 h^{r(j)}(\boldsymbol{p}^{r(j)})^\top \nabla J_i^r(\boldsymbol{u}^{(j-1)}) \tag{5.42}$$

for $i = 1, \ldots, k$. As a consequence of the inexactness in the function values and the gradients, the approximated set is a superset of the Pareto set. Depending on the errors $\boldsymbol{\kappa}$ and $\boldsymbol{\epsilon}$, each accumulation point of (5.41) is either contained in the set $\mathcal{P}_{S,\kappa}$ (5.40) or in the set $\mathcal{P}_{S,\epsilon}$ (5.39). If the errors $\boldsymbol{\kappa}$ and $\boldsymbol{\epsilon}$ are zero, the computed set is reduced to the set of substationary points. In this situation $\mathcal{P}_{S,\kappa} = \mathcal{P}_S$ and $\widehat{\boldsymbol{\alpha}}_{\mathsf{min}} = \mathbf{0}$, hence $\boldsymbol{p}^{r(j)} = \boldsymbol{q}(\boldsymbol{u}^{(j)})$. Convergence of the dynamical system (5.41) to an *approximately substationary* point is investigated in the following theorem:

Theorem 5.3.8. *Consider the multiobjective optimization problem* (MOPu) *with inexact objective functions and inexact gradients according to* (5.30) *and* (5.31)*, respectively. Suppose that* $\boldsymbol{u}^*$ *is an accumulation point of the sequence* $(\boldsymbol{u}^{(j)})_{j=0,1,\ldots}$ *created by* (5.41)*. Then*

(a) $\boldsymbol{u}^* \in \mathcal{P}_{S,\epsilon,\kappa} = \mathcal{P}_{S,\epsilon} \cup \mathcal{P}_{S,\kappa}$ *where* $\mathcal{P}_{S,\epsilon}$ *and* $\mathcal{P}_{S,\kappa}$ *are defined according to* (5.39) *and* (5.40)*, respectively,*

(b) if $\boldsymbol{\kappa} = \mathbf{0}$, $\boldsymbol{\epsilon} = \mathbf{0}$, $\boldsymbol{u}^*$ *is a substationary point of* (MOPu).

Proof. (a) For every point $\boldsymbol{u}^{(j)}$ created by the sequence (5.41), one of the following statements is true:

i)	$\boldsymbol{u}^{(j)} \in \mathcal{P}_{S,\epsilon}$	$\wedge$	$\boldsymbol{u}^{(j)} \notin \mathcal{P}_{S,\kappa}$	ii)	$\boldsymbol{u}^{(j)} \notin \mathcal{P}_{S,\epsilon}$	$\wedge$	$\boldsymbol{u}^{(j)} \in \mathcal{P}_{S,\kappa}$
iii)	$\boldsymbol{u}^{(j)} \in \mathcal{P}_{S,\epsilon}$	$\wedge$	$\boldsymbol{u}^{(j)} \in \mathcal{P}_{S,\kappa}$	iv)	$\boldsymbol{u}^{(j)} \notin \mathcal{P}_{S,\epsilon}$	$\wedge$	$\boldsymbol{u}^{(j)} \notin \mathcal{P}_{S,\kappa}$

In case i), $\boldsymbol{u}^{(j)} \in \mathcal{P}_{S,\epsilon}$ which means that the gradients $\nabla J_i^r(\boldsymbol{u}^{(j)})$, $i = 1, \ldots, k$, approximately satisfy the KKT conditions. We obtain $\sum_{i=1}^k \widehat{\alpha}_{\mathsf{min},i} = 1$, i.e. the set of valid descent directions is empty ($\mathcal{Q}^r = \emptyset$, cf. Theorem 5.3.5). Consequently, $\boldsymbol{p}^{r(j)} = 0$ and the point $\boldsymbol{u}^{(j)}$ is an accumulation point of the sequence (5.41). In case ii), the inexactness in $\boldsymbol{J}^r(\boldsymbol{u}^{(j)})$ results in $h^{r(j)} = 0$ (according to the modified Armijo rule (5.42)), such that $\boldsymbol{u}^{(j)}$ is an accumulation point. In case iii), both $\boldsymbol{p}^{r(j)} = 0$ and $h^{r(j)} = 0$. In case iv), we have $\boldsymbol{p}^{r(j)} = \boldsymbol{q}(\boldsymbol{u}^{(j)})$ and $h^{r(j)} > 0$. If for any $j \in \{0, 1, \ldots\}$, $\boldsymbol{u}^{(j)} \in \mathcal{P}_{S,\epsilon}$ or $\boldsymbol{u}^{(j)} \in \mathcal{P}_{S,\kappa}$, we are in one of the cases i) to iii) and $\boldsymbol{u}^{(j)}$ is an accumulation point. Otherwise, we obtain a descent direction such that the sequence (5.41) converges to a substationary point $\boldsymbol{u}^* \in \mathcal{P}_{S,\mathsf{sub}} \subseteq \mathcal{P}_{S,\epsilon}$ of (MOPu) according to Theorem 2.1.13, p. 16.

For part (b), we obtain $\widehat{\boldsymbol{\alpha}}_{\mathsf{min}} = \mathbf{0}$ by setting the errors $\boldsymbol{\epsilon}$ to zero and hence, the descent direction is $\boldsymbol{p}^{r(j)} = \boldsymbol{q}(\boldsymbol{u}^{(j)})$ (cf. Algorithm. 5.4). When $\boldsymbol{\kappa} = \mathbf{0}$, the modified Armijo rule becomes the standard Armijo rule for multiple objectives. Consequently, the problem is reduced to the case with exact function and gradient values (case iv) in part (a)). □

Algorithm 5.5 (Descent step under inexactness)

Require: Initial point $\boldsymbol{u}^{(j)}$, error bounds $\boldsymbol{\kappa} \in \mathbb{R}^k$ and $\boldsymbol{\epsilon} \in \mathbb{R}^k$, $c_1 \in (0,1)$

1: Compute $J_i^r(\boldsymbol{u}^{(j)})$ and $\nabla J_i^r(\boldsymbol{u}^{(j)})$, $i = 1, \ldots, k$
2: Compute a direction $\boldsymbol{p}^{r(j)}$ according to Algorithm 5.4
3: Compute a step length $h^{r(j)}$ that satisfies the modified Armijo rule (5.42), e.g. via backtracking [NW06]
4: Compute $\boldsymbol{u}^{(j+1)} = \boldsymbol{u}^{(j)} + h^{r(j)}\boldsymbol{p}^{r(j)}$

Following along the lines of [DSH05], this yields convergence of the subdivision algorithm with inexact values:

Theorem 5.3.9. *Suppose that the set $\mathcal{P}_{S,\epsilon,\kappa} = \mathcal{P}_{S,\epsilon} \cup \mathcal{P}_{S,\kappa}$ with $\mathcal{P}_{S,\epsilon}$ and $\mathcal{P}_{S,\kappa}$ according to (5.39) and (5.40) is bounded and connected. Let Q be a compact neighborhood of $\mathcal{P}_{S,\epsilon,\kappa}$. Then an application of the subdivision algorithm 2.2 to Q with respect to the iteration scheme (5.41) leads to a sequence of coverings $\mathcal{B}^{(s)}$ which is a subset of $\mathcal{P}_{S,\epsilon,\kappa}$ and a superset of the set $\mathcal{P}_{S,\mathsf{sub}}$ of substationary points $\boldsymbol{u}^* \in \mathbb{R}^n$ of (MOPu), that is,*

$$\mathcal{P}_{S,\mathsf{sub}} \subset \mathcal{B}^{(s)} \subset \mathcal{P}_{S,\epsilon,\kappa}.$$

Consequently, if the errors tend towards zero, we observe

$$\lim_{\boldsymbol{\epsilon},\boldsymbol{\kappa}\to\mathbf{0}} d_h(\mathcal{B}^{(s)}, \mathcal{P}_{S,\epsilon,\kappa}) = d_h(\mathcal{P}_{S,\mathsf{sub}}, \mathcal{B}^{(s)}) = 0.$$

Inexact Sampling Algorithm

In many problems, gradients are unknown or difficult to compute. In this case, we can extend the sampling algorithm (Algorithm 2.3) to uncertainties. The resulting algorithm 5.6 also consists of a subdivision and a selection step with the difference that the selection step is an inexact non-dominance test according to Definition 5.3.7. We thereby compute the superset $\mathcal{P}_{S,\kappa}$ of the global Pareto set $\mathcal{P}_S$ with $\lim_{\boldsymbol{\kappa}\to\mathbf{0}} h(\mathcal{P}_{S,\kappa}, \mathcal{P}_S) = 0$.

Algorithm 5.6 (Inexact sampling algorithm)

Let $\mathcal{B}^{(0)}$ be an initial collection of finitely many subsets of the compact set Q such that $\bigcup_{B\in\mathcal{B}^{(0)}} B = Q$. Then $\mathcal{B}^{(s)}$ is inductively obtained from $\mathcal{B}^{(s-1)}$ in two steps:

(i) **Subdivision.** Construct from $\mathcal{B}^{(s-1)}$ a new collection of subsets $\hat{\mathcal{B}}^{(s)}$ such that

$$\bigcup_{B\in\hat{\mathcal{B}}^{(s)}} B = \bigcup_{B\in\mathcal{B}^{(s-1)}} B,$$
$$\operatorname{diam}(\hat{\mathcal{B}}^{(s)}) = \theta_s \operatorname{diam}(\mathcal{B}^{(s-1)}), \quad 0 < \theta_{\min} \leq \theta_s \leq \theta_{\max} < 1.$$

(ii) **Selection**. Define the new collection $\mathcal{B}^{(s)}$ by

$$\mathcal{B}^{(s)} = \left\{ B \in \hat{\mathcal{B}}^{(s)} \;\middle|\; \nexists \hat{B} \in \hat{\mathcal{B}}^{(s)} \text{ such that } \hat{B} \text{ confidently dominates } B \right\}.$$

5.3.4 Examples

The results are now illustrated using three examples. To this end, we add random perturbations to the exact model such that (5.30) and (5.31) hold for fixed error bounds $\boldsymbol{\epsilon}$ and $\boldsymbol{\kappa}$. The first two examples, (5.43) and (5.44), are addressed with the gradient-based subdivision algorithm 2.2 where the exact dynamical system is replaced by the inexact version (5.41). The third example (5.45) is then treated with the inexact sampling algorithm 5.6 since the resulting Pareto set is disconnected.

We begin with a two dimensional example function $\boldsymbol{J} : \mathbb{R}^2 \to \mathbb{R}^2$ for two paraboloids:

$$\min_{\boldsymbol{u}\in\mathbb{R}^2} \boldsymbol{J}(\boldsymbol{u}) = \min_{\boldsymbol{u}\in\mathbb{R}^2} \begin{pmatrix} (u_1-1)^2 + (u_2-1)^4 \\ (u_1+1)^2 + (u_2+1)^2 \end{pmatrix}. \tag{5.43}$$

The box coverings of the Pareto sets and the corresponding Pareto fronts with $\boldsymbol{\kappa} = (0,0)^\top$, $\boldsymbol{\epsilon} = (0.1, 0.1)^\top$ and $\boldsymbol{\kappa} = (0,0)^\top$, $\boldsymbol{\epsilon} = (0.0, 0.2)^\top$ are shown in Figure 5.21 (a) and (b), respectively. The background is colored according to $\|\boldsymbol{q}(\boldsymbol{u})\|_2$ obtained by solving (QOP), and the white iso-line indicates the upper bound of the error (5.39). We see in (a) that the box covering is very close to the error bound whereas it is less sharp in (b). Consequently, the error estimate is more accurate when the errors are of comparable size in all gradients. The Pareto fronts corresponding to (a) and (b) are shown in (c) and (d). We see that the difference between the Pareto front of the exact (red) and of the inexact solution (green) is relatively small but that the front has longer tails, i.e. additional points close to the individual minima $J_1(\boldsymbol{u}) = 0$ and $J_2(\boldsymbol{u}) = 0$.

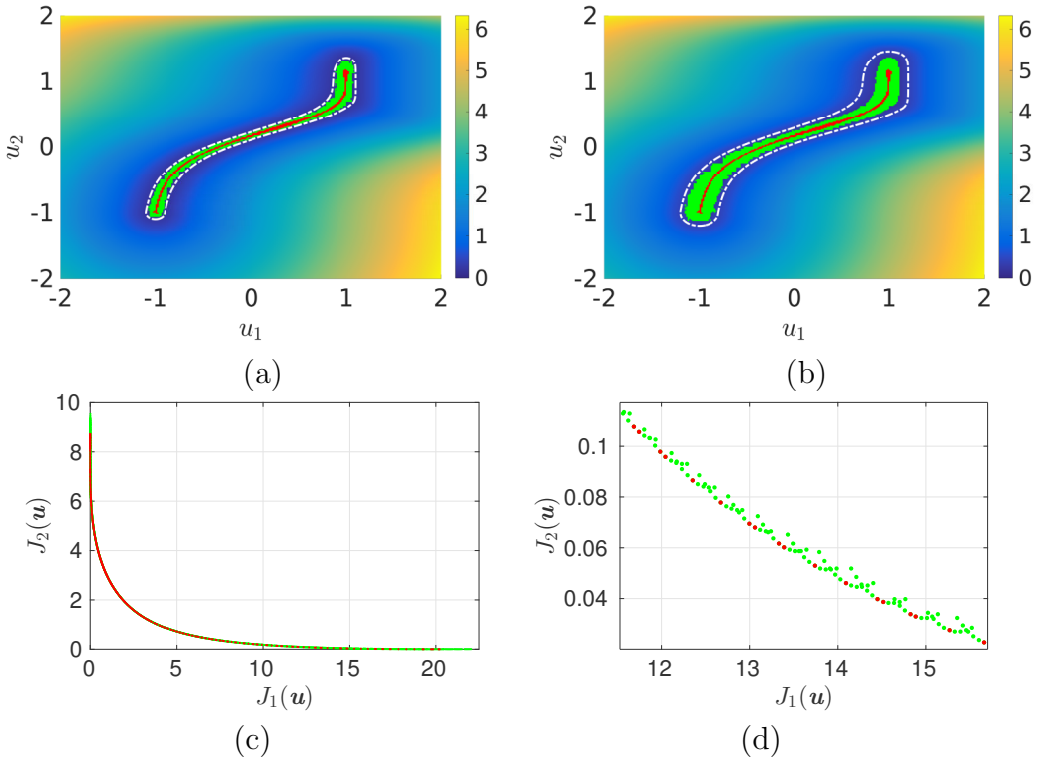


Figure 5.21: (a) Box covering of the Pareto set of problem (5.43) after 16 subdivision steps of the gradient-based inexact subdivision algorithm ($\operatorname{diam}(\mathcal{B}^{(0)}) = 4$, $\operatorname{diam}(\mathcal{B}^{(20)}) = 1/2^6$). The solution without errors ($\mathcal{P}_{S,\text{sub}}$) is shown in red, the solution with $\boldsymbol{\epsilon} = (0.1, 0.1)^\top$, $\boldsymbol{\kappa} = (0, 0)^\top$ ($\mathcal{P}_{S,\epsilon}$) is shown in green. The background color represents the norm of the optimality condition (KKTu), the upper bound of the error $\|\boldsymbol{q}(\boldsymbol{u})\|_2 = 2\|\boldsymbol{\epsilon}\|_\infty = 0.2$ is marked by the dashed white line. (b) Analog to (a) but with $\boldsymbol{\epsilon} = (0, 0.2)^\top$ and the iso-curve $\|\boldsymbol{q}(\boldsymbol{u})\|_2 = 0.4$. (c)–(d) The Pareto fronts corresponding to (a). The points are the images of the box centers (color coding as in (a) and (b)).

In the numerical realization, each box is sampled by an equidistant grid with two points in each direction, i.e. by four sample points in total. This results in a total number of approximately $50,000$ function evaluations for the exact problem. The number of boxes is much higher for the inexact solution, in this case by a factor of roughly eight. This is not surprising since the equality condition (KKTu) is replaced by the inequality condition (5.39). Hence, the approximated set is no longer a $(k-1)$-dimensional object. All sets $B \in \mathcal{B}^{(s)}$ that satisfy the inequality condition (5.39) are not discarded in the selection step. Consequently, at a certain box size, the number of boxes increases exponentially with decreasing diameter $\operatorname{diam}(B)$ (cf. Figure 5.25 (b) on p. 142). The result is an increased computational effort for higher order iterations.

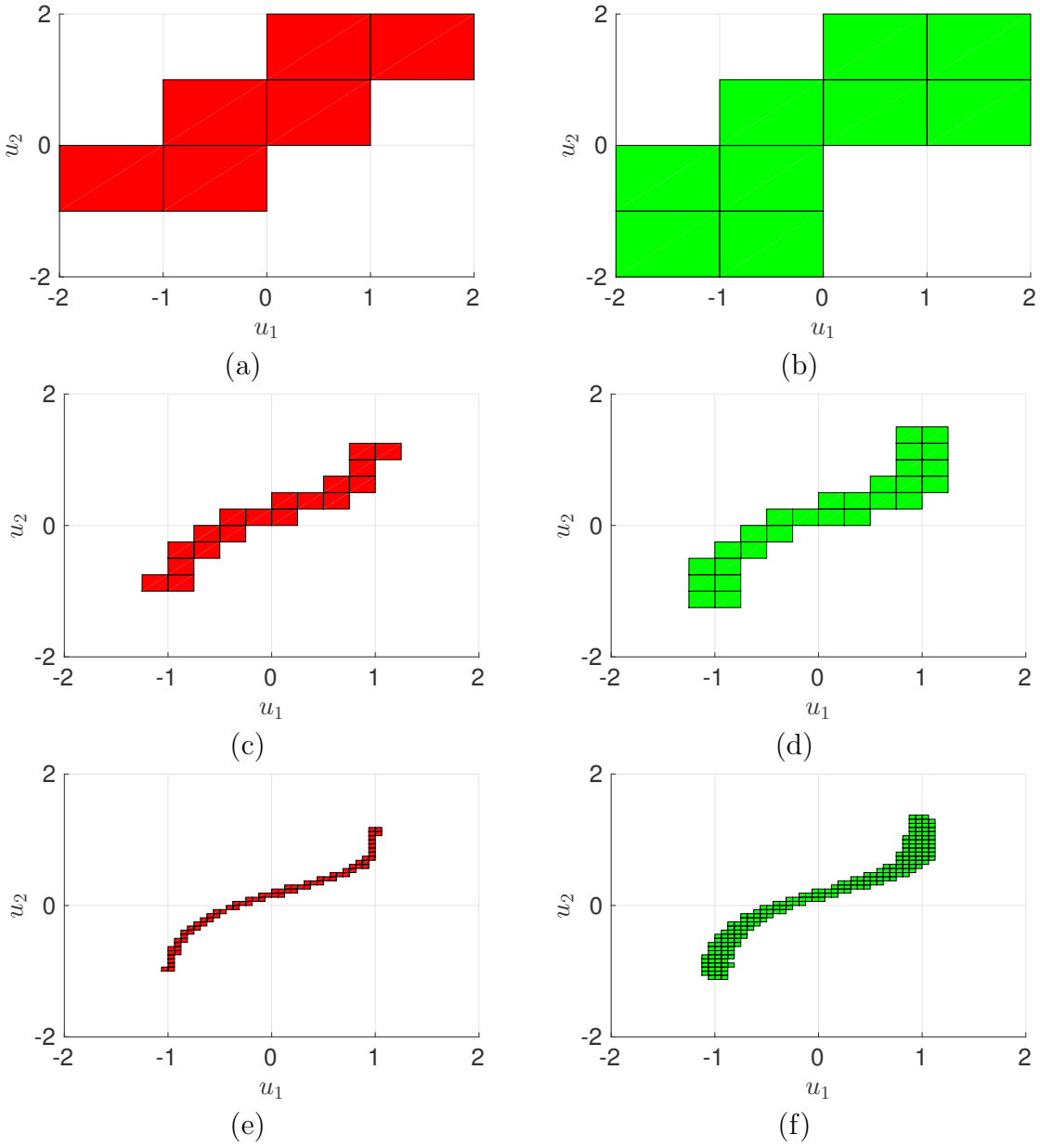


Figure 5.22: Comparison of the inexact and the exact solution of problem (5.43) after 4, 8 and 12 subdivision steps.

This is also visualized in Figure 5.22, where the solutions at different stages of the subdivision algorithm are compared. A significant difference can only be observed in later stages. For this reason, an adaptive strategy needs to be developed in the future where boxes satisfying (5.39) remain within the box collection but are no longer considered in the subdivision algorithm.

As a second example, we consider the function $\boldsymbol{J} : \mathbb{R}^3 \to \mathbb{R}^3$:

$$\min_{\boldsymbol{u}\in\mathbb{R}^3} \boldsymbol{J}(\boldsymbol{u}) = \min_{\boldsymbol{u}\in\mathbb{R}^3} \begin{pmatrix} (u_1-1)^4 + (u_2-1)^2 + (u_3-1)^2 \\ (u_1+1)^2 + (u_2+1)^4 + (u_3+1)^2 \\ (u_1-1)^2 + (u_2+1)^2 + (u_3-1)^4 \end{pmatrix}. \tag{5.44}$$

The observed behavior is very similar to the two-dimensional case, cf. Figure 5.23, where the box covering of the inexact problem as well as the iso-surface $\|\boldsymbol{q}(\boldsymbol{u})\|_2 = 2\|\boldsymbol{\epsilon}\|_\infty$ are shown in (b). One can see that the box covering lies inside this iso-surface except for very few boxes where small parts are outside. This is due to the finite box size and the fact that at least one sample point is mapped into the box itself. For smaller box radii, this artifact does no longer occur.

Finally, we consider an example where the Pareto set is disconnected. This is an example from production and was introduced in [SSW02]. We want to minimize the failure of a product which consists of n components. The probability of failing is modeled individually for each component and depends on the additional cost $\boldsymbol{u} \in \mathbb{R}^n$:

$$\begin{aligned} p_1(\boldsymbol{u}) &= 0.01 \exp\left(-(u_1/20)^{2.5}\right), \\ p_2(\boldsymbol{u}) &= 0.01 \exp\left(-(u_2/20)^{2.5}\right), \\ p_j(\boldsymbol{u}) &= 0.01 \exp\left(-u_j/15\right), \qquad j = 3, \ldots, n. \end{aligned}$$

The resulting MOP is thus to minimize the additional cost and the failure at the same time:

$$\min_{\boldsymbol{u}\in\mathbb{R}^n} \boldsymbol{J}(\boldsymbol{u}) = \min_{\boldsymbol{u}\in\mathbb{R}^n} \begin{pmatrix} \sum_{j=1}^n u_j \\ 1 - \sum_{j=1}^n (1 - p_j(\boldsymbol{u})) \end{pmatrix}. \tag{5.45}$$

We now assume that the additional cost are subject so some uncertainty, e.g. due to varying prices, and set $|\widetilde{u}_i - u_i| < 0.01$ for $i = 1, \ldots, n$. Using this, we numerically approximate the error bounds within the initial box $\mathcal{B}^{(0)} = [0, 40]^n$ and obtain $\boldsymbol{\kappa} = (0.05, 2 \cdot 10^{-5})^\top$ and $\boldsymbol{\epsilon} = (0, 8 \cdot 10^{-7})^\top$.

Since the Pareto set is disconnected, the exact and inexact sampling algorithm are now applied. The resulting Pareto sets $\mathcal{P}_S$ and $\mathcal{P}_{S,\kappa}$ are visualized in Figures 5.24 (a) and (b) for $n = 5$. Due to the small gradient of the second objective, this results in a significantly increased number of boxes by a factor of roughly 300 (cf. Figure 5.25 (b)).

The quality of the inexact solution can be measured using the Hausdorff distance $d_h(\mathcal{P}_S, \mathcal{P}_{S,\kappa})$ [RW98]. This is depicted in Figure 5.25 (a), where the distance between the exact and the inexact solution is shown for all subdivision steps for the three examples above. We see that the distance reaches an almost constant value in the later stages. This distance is influenced by the upper bounds $\boldsymbol{\epsilon}$ and $\boldsymbol{\kappa}$, respectively.

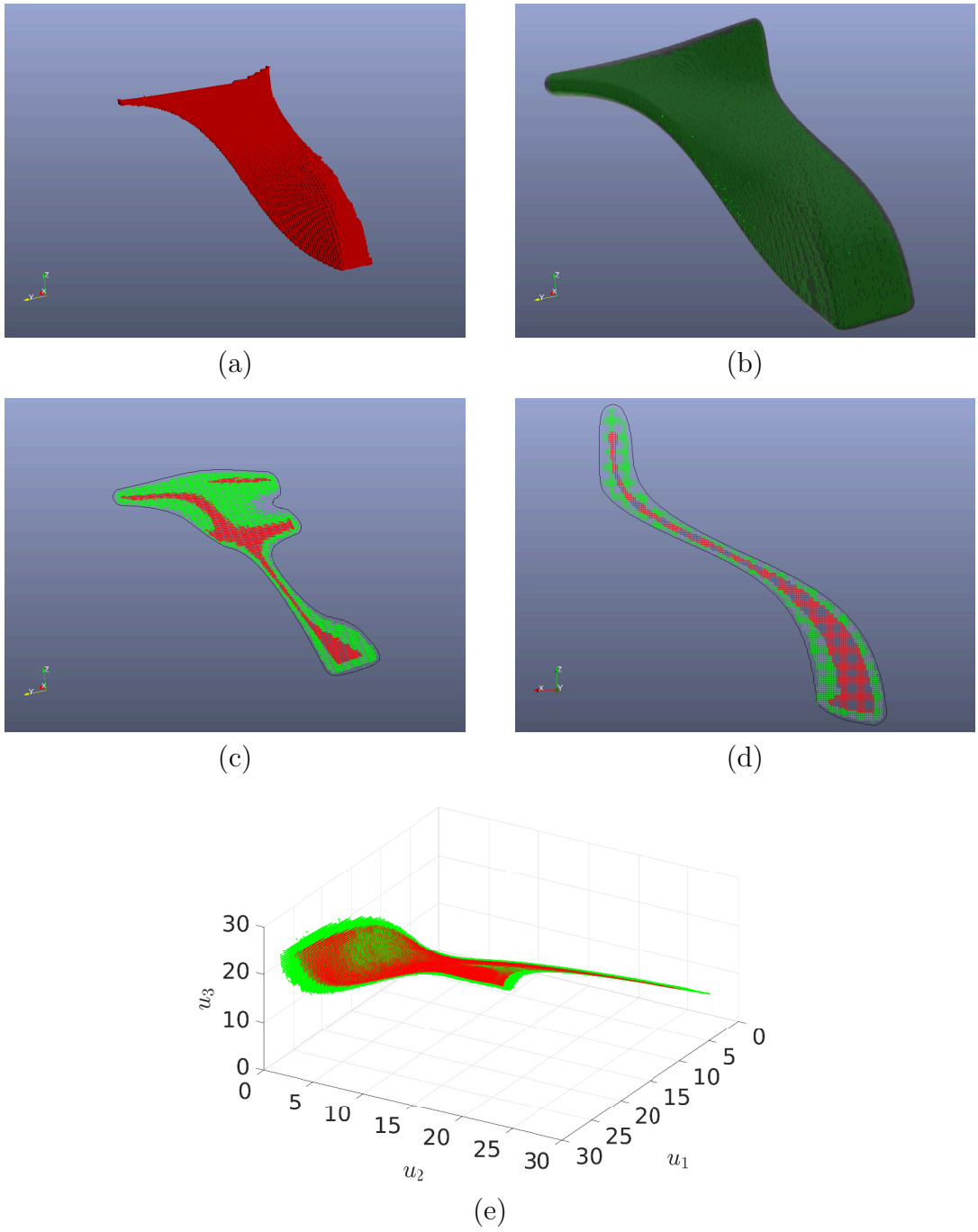


Figure 5.23: Box coverings of the Pareto set (exact solution $\mathcal{P}_{S,\mathsf{sub}}$ in red, inexact solution $\mathcal{P}_{S,\epsilon}$ in green) of problem (5.44) after 24 subdivision steps of the gradient-based inexact subdivision algorithm ($\operatorname{diam}(\mathcal{B}^{(0)}) = 4$, $\operatorname{diam}(\mathcal{B}^{(24)}) = 1/2^6$). (a) Solution without errors. (b) Solution with $\boldsymbol{\kappa} = (0,0,0)^\top$ and $\epsilon_i = (0.1, 0.1, 0.1)^\top$ and the iso-surface $\|\boldsymbol{q}(\boldsymbol{u})\|_2 = 2\|\boldsymbol{\epsilon}\|_\infty = 0.2$, i.e. the upper bound of the error. (c)–(d) Two-dimensional cut planes through the Pareto set. The point of view in (c) is as in (a) and (d) is a cut plane parallel to the u_1-u_3 plane at $u_2 = -0.9$. (e) The corresponding Pareto fronts.

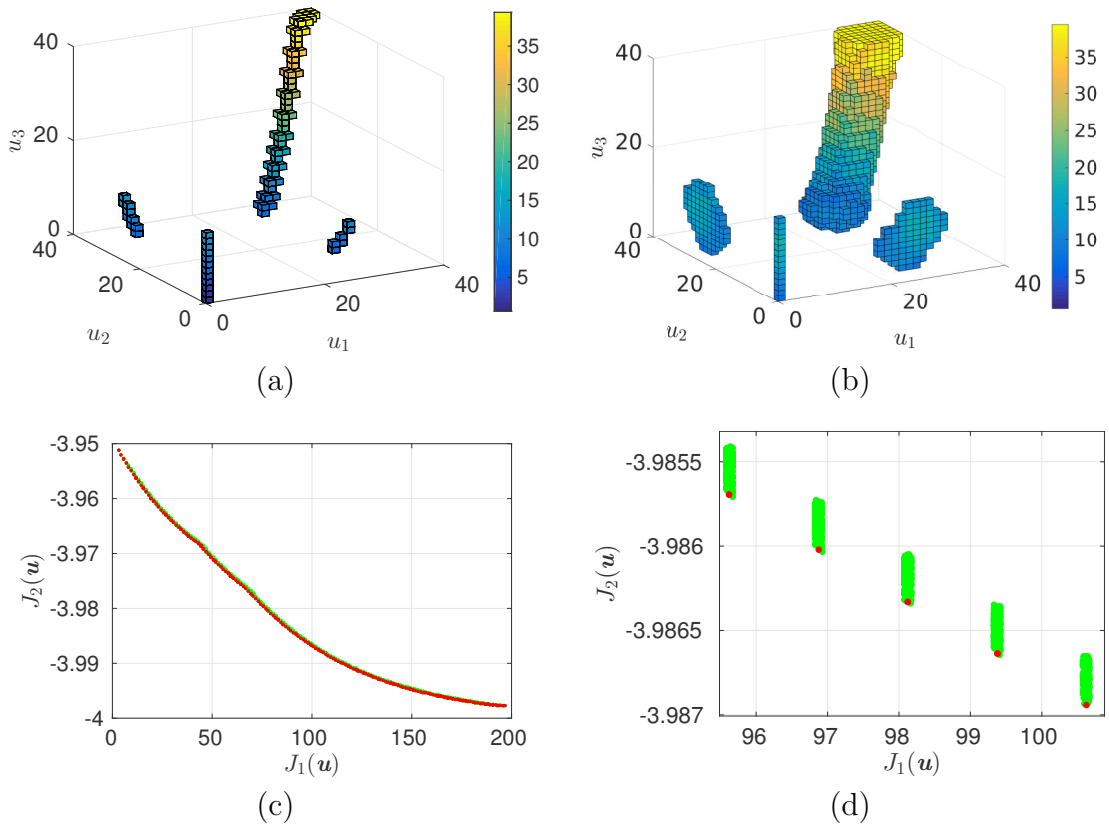
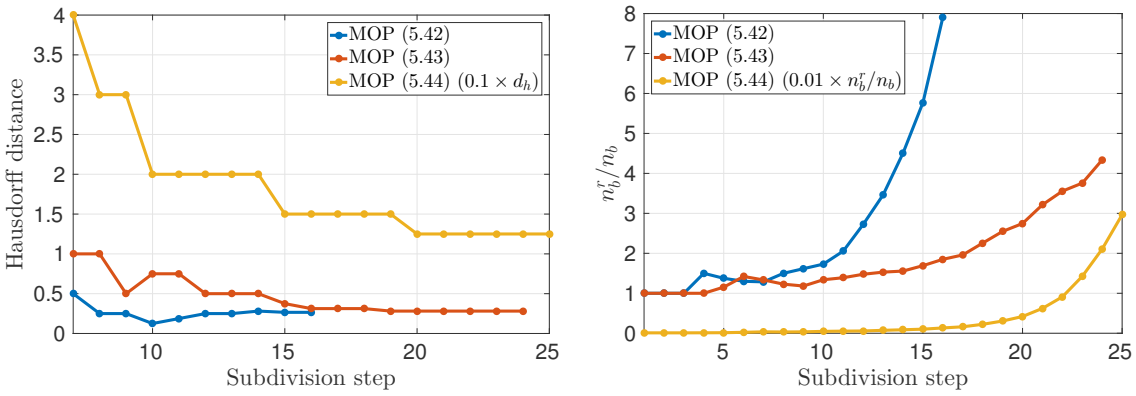


Figure 5.24: (a) Projection of the box covering of $\mathcal{P}_S$ for problem (5.45) with $n = 5$ after 25 subdivision steps of the inexact sampling algorithm ($\operatorname{diam}(\mathcal{B}^{(0)}) = 40$, $\operatorname{diam}(\mathcal{B}^{(25)}) = 1.25$). The coloring represents the fourth component u_4. (b) Box covering of $\mathcal{P}_{S,\kappa}$ with inexact data $\widetilde{\boldsymbol{u}}$ with $|\widetilde{u}_i - u_i| < 0.01$ for $i = 1, \ldots, 5$. (c)–(d) The Pareto fronts corresponding to (a) and (b) in green and red, respectively. The points are the images of the box centers.

However, it is currently not possible to prescribe an upper bound for the Hausdorff distance since depending on the algorithm, the bounds only limit inaccuracies in the optimality condition or in the dominance properties, respectively. In order to limit the error in the decision space, further assumptions on the objectives have to be made.

As observed earlier, the number of boxes is much higher for the inexact solution in all examples, cf. Figure 5.25 (b). The result is a severely increased computational effort for higher order iterations which motivates the development of an adaptive strategy in which non-dominated boxes are not further subdivided.



(a) (b)

Figure 5.25: (a) Hausdorff distance $d_h(\mathcal{P}_S, \mathcal{P}_{S,\kappa})$ for the three examples above. (b) The corresponding ratio of box numbers between the inexact solution (n_b^r) and the exact solution (n_b).

5.4 Set-Oriented Multiobjective Optimal Control of PDEs using ROMs

In this section the inexact sampling algorithm 5.6 developed in Section 5.3 is combined with model order reduction techniques in order to develop a global, derivative-free algorithm for PDE-constrained MOCPs using surrogate models. To this end, the inexact sampling algorithm is combined with a localized reduced basis approach [AHKO12] and error estimates for the objectives. The cynamical system considered here is similar to (5.22) from Section 5.2 except for an additional non-linear term which yields a semi-linear problem. An a-posteriori error estimator for the individual objectives is derived and numerical results concerning both the error estimator and the overall algorithm are presented.

The results presented in this section grew out of a cooperation with the University of Konstanz within the project *Multiobjective Optimal Control of Partial Differential Equations Using Reduced-Order Modeling* which is part of the *DFG Priority Programme 1962 - Non-smooth and Complementarity-based Distributed Parameter Systems: Simulation and Hierarchical Optimization*. They have previously appeared in [BDPV17] and the author has made substantial contributions. The results concerning error estimates for the ROMs (Section 5.4.2) are entirely due to the authors from the University of Konstanz.

5.4.1 The Multiobjective Optimal Control Problem

Throughout this section let $\Omega \subset \mathbb{R}^d$ be a bounded Lipschitz domain with boundary Γ. Further, let $(t_0, t_e) \subset \mathbb{R}$ be a time interval, $Q := (t_0, t_e) \times \Omega$ and $\Sigma := (t_0, t_e) \times \Gamma$. The domain contains subdomains $\Omega_i \subset \Omega$ and indicator functions are defined by $\chi_i(\boldsymbol{x}) = 1$ if $\boldsymbol{x} \in \Omega_i$ and $\chi_i(\boldsymbol{x}) = 0$ otherwise $(i = 1, \dots, n)$. The control space is given by $\mathcal{U} = \mathbb{R}^n$ and we consider the following MOCP:

$$\min_{\boldsymbol{u}\in\mathcal{U}} \hat{\boldsymbol{J}}(y, \boldsymbol{u}) = \min_{\boldsymbol{u}\in\mathcal{U}} \frac{1}{2} \begin{pmatrix} \int_\Omega \|y(t_e, \boldsymbol{x}) - y_{d,1}(\boldsymbol{x})\|^2 \, \mathrm{d}\boldsymbol{x} \\ \int_\Omega \|y(t_e, \boldsymbol{x}) - y_{d,2}(\boldsymbol{x})\|^2 \, \mathrm{d}\boldsymbol{x} \\ \|\boldsymbol{u}\|_2^2 \end{pmatrix} \tag{5.46a}$$

subject to the PDE constraints

$$\begin{aligned} \dot{y}(t, \boldsymbol{x}) - \Delta y(t, \boldsymbol{x}) + y^3(t, \boldsymbol{x}) &= \sum_{i=1}^{n} u_i \chi_i(\boldsymbol{x}) && \text{for } (t, \boldsymbol{x}) \in Q, \\ \frac{\partial y}{\partial \boldsymbol{n}}(t, \boldsymbol{x}) &= 0 && \text{for } (t, \boldsymbol{x}) \in \Sigma, \\ y(0, \boldsymbol{x}) &= y_0(\boldsymbol{x}) && \text{for } \boldsymbol{x} \in \Omega, \end{aligned} \tag{5.46b}$$

and the box constraints

$$\boldsymbol{u}_\mathsf{a} \leq_p \boldsymbol{u} \leq_p \boldsymbol{u}_\mathsf{b} \quad \text{in } \mathcal{U} \text{ for } t \in (t_0, t_e). \tag{5.46c}$$

In (5.46a), the functions $y_{d,1}$ and $y_{d,2} \in L^2(\Omega; \mathbb{R})$ are two conflicting desired states. The state variable y is given as the solution to the semi-linear heat equation (5.46b). One can show that such a solution always exists, is unique and in particular belongs to $C([t_0, t_e]; L^2(\Omega))$, meaning that the integrals in (5.46a) are well-defined (see [BDPV17] for details). In (5.46c), the control variable $\boldsymbol{u}$ is bounded by bilateral constraints $\boldsymbol{u}_\mathsf{a}, \boldsymbol{u}_\mathsf{b} \in \mathcal{U}$ with $\boldsymbol{u}_\mathsf{a} \leq_p \boldsymbol{u}_\mathsf{b}$. Using this, the *admissible set*

$$\mathcal{U}_\mathsf{ad} = \left\{ \boldsymbol{u} \in \mathcal{U} \,\middle|\, \boldsymbol{u}_\mathsf{a} \leq_p \boldsymbol{u} \leq_p \boldsymbol{u}_\mathsf{b} \text{ in } \mathcal{U} \right\}$$

can be defined. In order to apply a finite element method for solving (5.46), we introduce the weak formulation of the state equation (5.46b).

Definition 5.4.1. *A function $y \in \mathcal{Y} = L^2((t_0, t_e); V) \cap H^1((t_0, t_e); V')$ is called a* weak solution *to (5.46b) if for every $\varphi \in V = H^1(\Omega)$:*

$$\begin{aligned} (\dot{y}(t), \varphi) + (\nabla y(t), \nabla \varphi) + \left(y(t)^3, \varphi\right) &= \sum_{i=1}^{n} u_i \left(\chi_i, \varphi\right) \quad \text{æ in } (t_0, t_e), \\ (y(0), \varphi) &= (y_0, \varphi)\,, \end{aligned} \tag{5.47}$$

where æ stands for "almost everywhere".

In the FEM discretization, a Galerkin method is employed to replace the infinite-dimensional problem (5.47) by a finite-dimensional version. Given linearly independent spatial basis functions $\phi_1, \ldots, \phi_{n_y} \in V$, the space V is replaced by an n_y-dimensional subspace $V^d = \operatorname{span}\{\phi_1, ..., \phi_{n_y}\} \subset V$. Typically, n_y is a very large number which is why the solution of the resulting system as referred to as a *high-fidelity solution*:

Definition 5.4.2. *A function $y^d \in H^1((t_0, t_e); V^d)$ is called a* high-fidelity solution *to (5.46b) if it holds for every $\varphi^d \in V^d$:*

$$\begin{aligned} \left(\dot{y}^d(t), \varphi^d\right) + \left(\nabla y^d(t), \nabla \varphi^d\right) + \left(y^d(t)^3, \varphi^d\right) &= \sum_{i=1}^{n} u_i \left(\chi_i, \varphi^d\right) \quad \text{æ in } (t_0, t_e), \\ \left(y^d(0), \varphi^d\right) &= \left(y_0^d, \varphi^d\right), \end{aligned} \tag{5.48}$$

One can show – similar to previous problems – that for every $\boldsymbol{u} \in \mathcal{U}$, there exists a unique weak solution $y \in \mathcal{Y}$ for (5.47) as well as a corresponding finite-dimensional weak solution $y^d \in H^1((t_0, t_e); V^d)$ for (5.48). Consequently, we can define solution operators $\mathcal{S}$ and $\mathcal{S}^d$ and introduce the reduced cost functionals

$$\begin{aligned} \boldsymbol{J} &: \mathcal{U} \to \mathbb{R}^3, & \boldsymbol{J}(\boldsymbol{u}) &= \boldsymbol{J}(\mathcal{S}\boldsymbol{u}, \boldsymbol{u}), \\ \boldsymbol{J}^d &: \mathcal{U} \to \mathbb{R}^3, & \boldsymbol{J}^d(\boldsymbol{u}) &= \boldsymbol{J}^d(\mathcal{S}^d\boldsymbol{u}, \boldsymbol{u}). \end{aligned}$$

The MOCP (5.46) can now be reduced accordingly:

$$\min_{\boldsymbol{u}\in\mathcal{U}} \boldsymbol{J}(\boldsymbol{u}) = \min_{\boldsymbol{u}\in\mathcal{U}} \frac{1}{2} \begin{pmatrix} \|(\mathcal{S}\boldsymbol{u})(t_e) - y_{d,1}\|_H^2 \\ \|(\mathcal{S}\boldsymbol{u})(t_e) - y_{d,2}\|_H^2 \\ \|\boldsymbol{u}\|_2^2 \end{pmatrix} \quad \text{s.t. } \boldsymbol{u} \in \mathcal{U}_{\mathsf{ad}}$$

and the corresponding high-fidelity problem is

$$\min_{\boldsymbol{u}\in\mathcal{U}} \boldsymbol{J}^d(\boldsymbol{u}) = \min_{\boldsymbol{u}\in\mathcal{U}} \frac{1}{2} \begin{pmatrix} \|(\mathcal{S}^d\boldsymbol{u})(t_e) - y_{d,1}\|_H^2 \\ \|(\mathcal{S}^d\boldsymbol{u})(t_e) - y_{d,2}\|_H^2 \\ \|\boldsymbol{u}\|_2^2 \end{pmatrix} \quad \text{s.t. } \boldsymbol{u} \in \mathcal{U}_{\mathsf{ad}}. \tag{5.49}$$

5.4.2 Model Order Reduction

The sampling algorithm 2.3 requires evaluating the cost function $\boldsymbol{J}(\boldsymbol{u})$ for all sample points $\boldsymbol{u} \in B$ and all sets $B \in \mathcal{B}^{(s)}$. Each of these evaluations requires a solution of

system (5.48) for the current control. As has been stated before, this can quickly become infeasible and it is therefore necessary to apply model-order reduction techniques to reduce the computational effort for the optimization.

Similar to the previous sections, a POD-based ROM will be used. To this end, s snapshots from a high-fidelity solution are stored in the snapshot matrix $\boldsymbol{S} = \left[y^d(t_0), \dots, y^d(t_e)\right] \in \mathbb{R}^{n_y, s}$, where $y^d(t_i)$ are the nodal values of the snapshots $y(t_i)$. We then compute the POD basis $\{\psi_i^d\}_{i=1}^{\ell} \subset V^d$ of rank $\ell \ll d$ based on $\boldsymbol{S}$ (cf. Section 2.3).

We define the finite-dimensional subspace $V^r = \operatorname{span}\{\psi_1^d, \dots, \psi_\ell^d\} \subset V^d$. Thus, the reduced state $y^r(t) \in V^r$ solves the following POD Galerkin scheme for every $1 \leq i \leq \ell$:

$$\begin{aligned} \left(\dot{y}^r(t), \psi_i^d\right) + \left(\nabla y^r(t), \nabla \psi_i^d\right) + \left(y^r(t)^3, \psi_i^d\right) &= \sum_{i=1}^{n} u_i \left(\chi_i, \psi_i^d\right) \quad \text{æ in } (t_0, t_e), \\ y^r(0) &= \mathcal{P}^r y_0, \end{aligned}$$

where the linear projection operator $\mathcal{P}^r : H \to V^r$ is given by

$$y_0^r = \mathcal{P}^r y_0 = \operatorname*{arg\,min}_{\psi_i^d \in V^r} \|\psi_i^d - y_0\|_H.$$

Introducing the POD solution operator $\mathcal{S}^r : \mathcal{U} \to H^1((t_0, t_e); V^r)$, the reduced order cost functional $\boldsymbol{J}^r : \mathcal{U} \to \mathbb{R}^3$ can be formulated:

$$\boldsymbol{J}^r(\boldsymbol{u}) = \frac{1}{2} \begin{pmatrix} \|(\mathcal{S}^r \boldsymbol{u})(t_e) - y_{d,1}\|_H^2 \\ \|(\mathcal{S}^r \boldsymbol{u})(t_e) - y_{d,2}\|_H^2 \\ \|\boldsymbol{u}\|_2^2 \end{pmatrix}.$$

Assuming that $\mathcal{P}^r : V \to V$ is bounded, it follows that $\mathcal{S}^r$ is well-defined and that the error between the high-fidelity and the reduced order solution can be estimated [RTV17]:

$$\|\mathcal{S}^d \boldsymbol{u} - \mathcal{S}^r \boldsymbol{u}\|_{L^2((t_0,t_e);V)}^2 \leq C \sum_{i=\ell+1}^{\bar{\ell}} \lambda_i^d \|\psi_i^d - \mathcal{P}^r \psi_i^d\|_V^2 < \infty \text{ for any } \boldsymbol{u} \in \mathcal{U}_{\mathsf{ad}}. \quad (5.50)$$

Note that this error estimate is only valid for the particular control $\boldsymbol{u}_{\mathsf{ref}} \in \mathcal{U}_{\mathsf{ad}}$ which is used to generate the ROM. However, we need estimate the error for an arbitrary control $\boldsymbol{u} \in \mathcal{U}_{\mathsf{ad}}$ which is achieved by utilizing the following theorem.

Theorem 5.4.3 ([RTV17]). *Let $V^r = \operatorname{span}\{\psi_1^d, \dots, \psi_\ell^d\}$ be a finite-dimensional subspace as described above and let $\boldsymbol{u} \in \mathcal{U}_{\mathsf{ad}}$ be an arbitrary admissible control. Define the state and reduced state solutions as $y^d = \mathcal{S}^d \boldsymbol{u}$ and $y^r = \mathcal{S}^r \boldsymbol{u}$. Then the following a-posteriori estimate for the state holds:*

$$\|y^d(t) - y^r(t)\|_H^2 + \int_{t_0}^t \|y^d(\tau) - y^r(\tau)\|_V^2 \,\mathrm{d}\tau \leq \Delta^{\mathsf{pr}}(t, y^r) \quad \text{for all } t \in [t_0, t_e] \tag{5.51}$$

with the a-posteriori estimator

$$\Delta^{\mathsf{pr}}(t, y^r) = e^{2t}\Big(\|y_0 - y_0^r\|_H^2 + \int_{t_0}^t \|R^r(\tau)\|_{V'}^2 \,\mathrm{d}\tau\Big),$$

where the residual *term is defined for $t \in [t_0, t_e]$ and $\varphi^d \in V^d$ as:*

$$\left(R^r(t), \varphi^d\right) = \left(\dot{y}^r(t), \varphi^d\right) + \left(\nabla y^r(t), \nabla \varphi^d\right) + \left(y^r(t)^3, \varphi^d\right) - \sum_{i=1}^n u_i \left(\chi_i, \varphi^d\right).$$

From this, one can immediately derive an estimator for the cost function:

Lemma 5.4.4 ([BDPV17]). *Let V^r be as before and $\boldsymbol{u} \in \mathcal{U}_{\mathsf{ad}}$ an arbitrary admissible control. Then the following a-posteriori estimate for the cost function holds for $i = 1, 2$:*

$$\|J_i^d(\boldsymbol{u}) - J_i^r(\boldsymbol{u})\| \leq \sqrt{2\Delta^{\mathsf{pr}}(t_e, y^r) J^r(\boldsymbol{u})} + \frac{1}{2}\Delta^{\mathsf{pr}}(t_e, y^r). \tag{5.52}$$

Proof. We fix $i \in \{1, 2\}$ and – using basic estimations – observe:

$$\begin{aligned}
\|J_i^d(\boldsymbol{u}) - J_i^r(\boldsymbol{u})\| &= \frac{1}{2}\left|\|y^d(t_e) - y_{d,i}\|_H^2 - \|y^r(t_e) - y_{d,i}\|_H^2\right| \\
&= \frac{1}{2}\left|\|y^d(t_e) - y_{d,i}\|_H + \|y^r(t_e) - y_{d,i}\|_H\right| \\
&\quad \cdot \left|\|y^d(t_e) - y_{d,i}\|_H - \|(y^r(t_e) - y_{d,i}\|\right| \\
&\leq \frac{1}{2}\left(\|y^d(t_e) - y_{d,i}\|_H + \|y^r(t_e) - y_{d,i}\|_H\right) \cdot \|y^d(t_e) - y^r(t_e)\|_H \\
&\leq \frac{1}{2}\left(2\,\|y^r(t_e) - y_{d,i}\|_H + \|y^d(t_e) - y^r(t_e)\|_H\right) \cdot \|y^d(t_e) - y^r(t_e)\|_H \\
&= \sqrt{2J^r(\boldsymbol{u})} \cdot \|y^d(t_e) - y^r(t_e)\|_H + \frac{1}{2}\|y^d(t_e) - y^r(t_e)\|_H^2.
\end{aligned}$$

The term $\|y^d(t_e) - y^r(t_e)\|_H^2$ can be further bounded by the state estimator (5.51), and this yields (5.52). □

Remark 5.4.5. *The estimate $\Delta^{\mathsf{pr}}(\cdot, y^r(\cdot))$ is rigorous. Consequently, it has been assumed during the proof that the unknown true error always behaves according to*

the worst-case scenario. One therefore frequently observes that the true error is overestimated by an almost constant factor. Depending on the application, it may be desirable to tighten the estimator by heuristically incorporating data from an offline phase. Such a heuristic will be introduced in the next section. However, it has to be noted that the mathematical rigor will be lost by doing this.

5.4.3 A Localized Reduced Bases Algorithm

In this section we will combine the results from Section 5.3.2 with the error estimate (5.52) in order to efficiently and globally solve PDE-constrained MOCPs with set-oriented techniques. To this end, the error estimates will be tightened by a heuristic factor and then concepts from the localized reduced basis method will be adapted to the multiobjective setting. The task is to solve problem (5.46) using the inexact sampling algorithm and POD-based ROMs. In order to to compute the set of almost non-dominated points $\mathcal{P}_{S,\kappa}$, we define a maximal error $\boldsymbol{\kappa} \in \mathbb{R}^k$ that is acceptable in the objectives.

As mentioned in Remark 5.4.5, the error estimator (5.51) often constantly over-estimates the true error, a fact that can also be observed in Figure 5.26 for the given setup. In order to tighten the estimator, the following heuristic is applied. We assume an over-estimation of the true error of the form

$$\Delta^{\mathsf{pr}}(t_e, y^r) \approx C_{\mathsf{sc}} \cdot \|y^d(t_e) - y^r(t_e)\|_H^2,$$

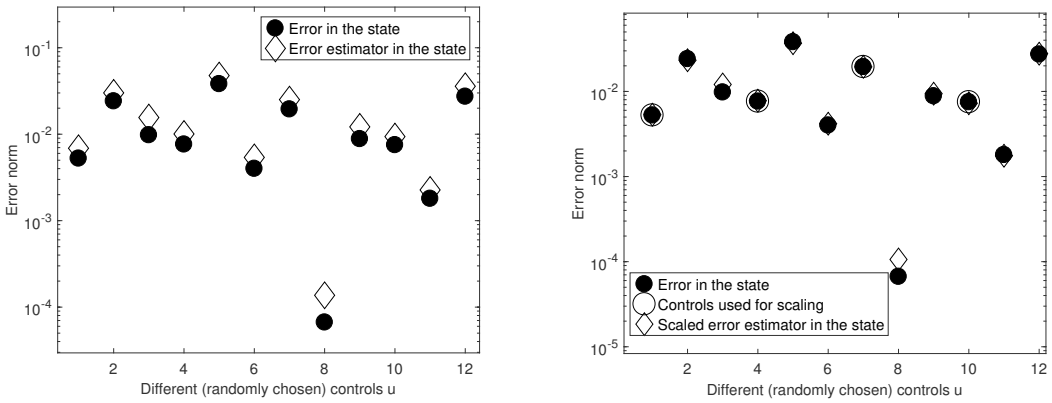
where $C_{\mathsf{sc}} > 1$ is an unknown scaling factor. Let a finite sample set $\mathcal{U}_{\mathsf{sc}} \subset \mathcal{U}_{\mathsf{ad}}$ be given prior to the optimization. For a given ROM, we compute the high-fidelity and reduced solutions $y^d = \mathcal{S}^d \boldsymbol{u}$ and $y^r = \mathcal{S}^r \boldsymbol{u}$ for all sample controls $\boldsymbol{u} \in \mathcal{U}_{\mathsf{sc}}$ and compute C_{sc} as the geometric average

$$C_{\mathsf{sc}} = \left(\prod_{\boldsymbol{u} \in \mathcal{U}_{\mathsf{sc}}} \frac{\Delta^{\mathsf{pr}}(t_e, y^r_{\boldsymbol{u}}(T))}{\|y^d_{\boldsymbol{u}}(t_e) - y^r_{\boldsymbol{u}}(t_e)\|_H^2} \right)^{\frac{1}{|\mathcal{U}_{\mathsf{sc}}|}}.$$

Using this heuristic scaling factor, the error estimator can be replaced by

$$\Delta^{\mathsf{pr}}_{\mathsf{sc}}(t, y^r(t)) = \frac{\Delta^{\mathsf{pr}}(t, y^r(t))}{C_{\mathsf{sc}}}.$$

The result of this procedure is illustrated for randomly chosen controls in Figure 5.26. It can be observed that the mathematical rigor is lost, i.e. $\Delta^{\mathsf{pr}}_{\mathsf{sc}}$ may be smaller than the true error. However, it is significantly tighter which is beneficial for the numerical efficiency in the sampling algorithm. This way, a larger number of boxes is confidently dominated.



(a) (b)

Figure 5.26: (a) Error $\|y^d(t_e) - y^r(t_e)\|_H^2$ in the state and error estimator $\Delta^{\mathsf{pr}}(t_e, y^r(t_e))$ for randomly chosen controls. (b) Error $\|y^d(t_e) - y^r(t_e)\|_H^2$ in the state and scaled error estimator $\Delta^{\mathsf{pr}}_{\mathsf{sc}}(t_e, y^r(t_e))$ for the same controls.

In Figure 5.27, the corresponding function values obtained with the high-fidelity and the surrogate model are compared. The error margin indicated by the estimator Δ^J is barely visible due to the high accuracy.

Localized Reduced Bases Approach

Using the tightened error bounds, we are now in the position to address problem (5.49) both with a finite element discretization and a POD approach. In the FEM approach, the discretized state equation is evaluated at each sample point and the value of the cost functionals J_1^d to J_3^d is determined. In the POD approach, we have to ensure that the error estimator Δ^J_{sc} is less than the prescribed bound $\boldsymbol{\kappa}$ everywhere in the parameter domain. If we want to achieve this goal with a single ROM, the model may become high-dimensional in order to satisfy the error bounds everywhere and hence, inefficient. We therefore adopt ideas from localized reduced basis methods [AHKO12, OS15] and construct a library $\mathcal{R}$ of locally valid models during the subdivision procedure.

Prior to the first computation, the factor C_{sc} is determined using several ROMs at randomly distributed controls $\boldsymbol{u}_{\mathsf{ref}} \in \mathcal{U}_{\mathsf{sc}}$ within the parameter domain. In each of these points, the factor between the true error and the error estimator is computed and C_{sc} is the mean factor over all computations. For problem (5.46), this factor is approximately 2.5 everywhere in the parameter domain such that this approach is justified. The library $\mathcal{R}$ is initialized by constructing a ROM for each of these FEM

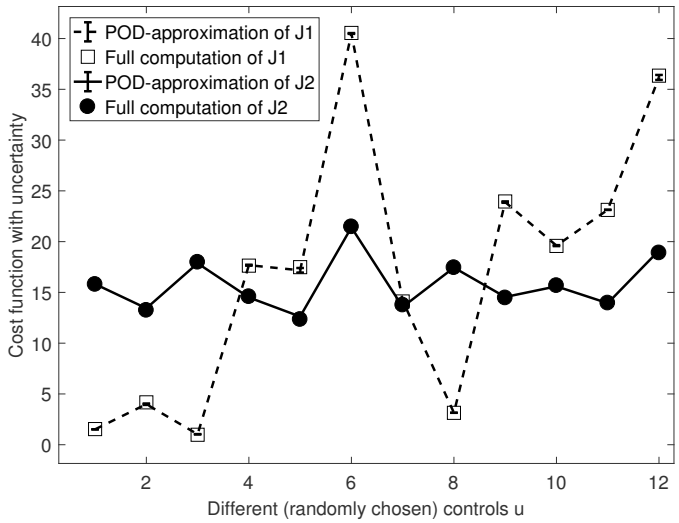


Figure 5.27: Cost function values $J_i^d(\boldsymbol{u})$ and $J_i^r(\boldsymbol{u})$, $i = 1, 2$, corresponding to the random controls in Figure 5.26.

solutions. This library can grow or shrink during the optimization, cf. Algorithm 5.7 below.

All sample points (the indices of which are contained in the set $\mathcal{N}$) are evaluated using the *closest* ROM in each subdivision step, where the distance is defined via the Euclidean distance between the control $\boldsymbol{u}$ and the reference control $\boldsymbol{u}_{\mathsf{ref}}^{(j)}$ at which the j^{th} ROM was created. In the beginning, all points are denoted as *insufficiently approximated*:

$$\mathcal{I} = \left\{ i \in \mathcal{N} \;\middle|\; \Delta_{\mathsf{sc}}^J(\boldsymbol{u}^{(i)}) \not\leq_p \boldsymbol{\kappa} \right\},$$

meaning that they have not yet been approximated well enough using a ROM. After the evaluation of $\boldsymbol{J}^r$, all points with a satisfactory error estimate according to (5.52) are eliminated from $\mathcal{I}$. Since the remaining points violate the desired error bound $\boldsymbol{\kappa} \in \mathbb{R}^k$, we evaluate the full model and add a ROM to the library $\mathcal{R}$. This is done in a *greedy* way (see also [GMNP07]), i.e. the ROM is added at the point with the maximum error. The above steps are repeated until all points are sufficiently accurately approximated and consequently, the set $\mathcal{I}$ is empty. Finally, all ROMs are removed from $\mathcal{R}$ which have not been used. This is done in order to keep the number of locally valid ROMs at an acceptable number. Moreover, ROMs belonging to regions in the parameter domain which have been identified as confidently dominated will not be required any further. The procedure is summarized in Algorithm 5.7.

Algorithm 5.7 (Greedy localized reduced basis approach)

Require: $\boldsymbol{\kappa} \in \mathbb{R}^k$, library $\mathcal{R}$, error scaling factor C_{sc}, set of sample points $\mathcal{N}$;

Consider all sample points as *insufficiently approximated*, i.e. $\mathcal{I} = \mathcal{N}$
while $\mathcal{I} \neq \emptyset$ **do**
  for $i = 1, \ldots, |\mathcal{I}|$ **do**
    Identify the *closest* ROM with respect to the 2-norm:

$$i_{\mathsf{close}} = \underset{j \in \{1,\ldots,|\mathcal{R}|\}}{\arg\min} \|\boldsymbol{u}^{(i)} - \boldsymbol{u}_{\mathsf{ref}}^{(j)}\|_2$$

    Compute $\boldsymbol{J}^r(\boldsymbol{u}^{(i)})$ using ROM i_{close}
    Evaluate the error $\Delta_{\mathsf{sc}}^J(\boldsymbol{u}^{(i)})$ for ROM i_{close} using (5.52)
    if $\Delta_{\mathsf{sc}}^J(\boldsymbol{u}^{(i)}) \leq_p \boldsymbol{\kappa}$ **then**
      Accept $\boldsymbol{J}^r(\boldsymbol{u}^{(i)})$ as sufficiently accurate
      Remove i from the set $\mathcal{I}$
    end if
  end for
  Identify the sample point with the largest error:

$$i_{\mathsf{max}} = \arg\max_{s \in \mathcal{I}} \Delta_{\mathsf{sc}}^J(\boldsymbol{u}^{(s)})$$

  Add a ROM to the library $\mathcal{R}$ with $\boldsymbol{u}_{\mathsf{ref}} = \boldsymbol{u}^{(i_{\mathsf{max}})}$
end while
Remove all ROMs from $\mathcal{R}$ that have not been used

Numerical Results

In this section we compare numerical results for the MOP (5.49) obtained by a FEM discretization (Algorithm 2.3) as well as a ROM (Algorithm 5.6), where the sample points in each subdivision step are evaluated using Algorithm 5.7.

As the domain we consider the unit square $\Omega = (0,1)^2$ and the time interval $[t_0, t_e] = [0, 1]$. The desired states are given by

$$y_{d,1}(\boldsymbol{x}) = \begin{cases} 0.5, & x_2 \leq 0.5, \\ 0.3, & x_2 > 0.5, \end{cases} \qquad y_{d,2}(\boldsymbol{x}) = \begin{cases} -0.5, & x_1 \leq 0.5, \\ 0.5, & x_1 > 0.5, \end{cases}$$

such that there are both conflicting and non-conflicting areas in the domain. The subdomains are given by

$$\Omega_1 = [0, 0.5] \times [0, 0.5], \quad \Omega_2 = [0, 0.5] \times (0.5, 1],$$
$$\Omega_3 = (0.5, 1] \times [0, 0.5], \quad \Omega_4 = (0.5, 1] \times (0.5, 1].$$

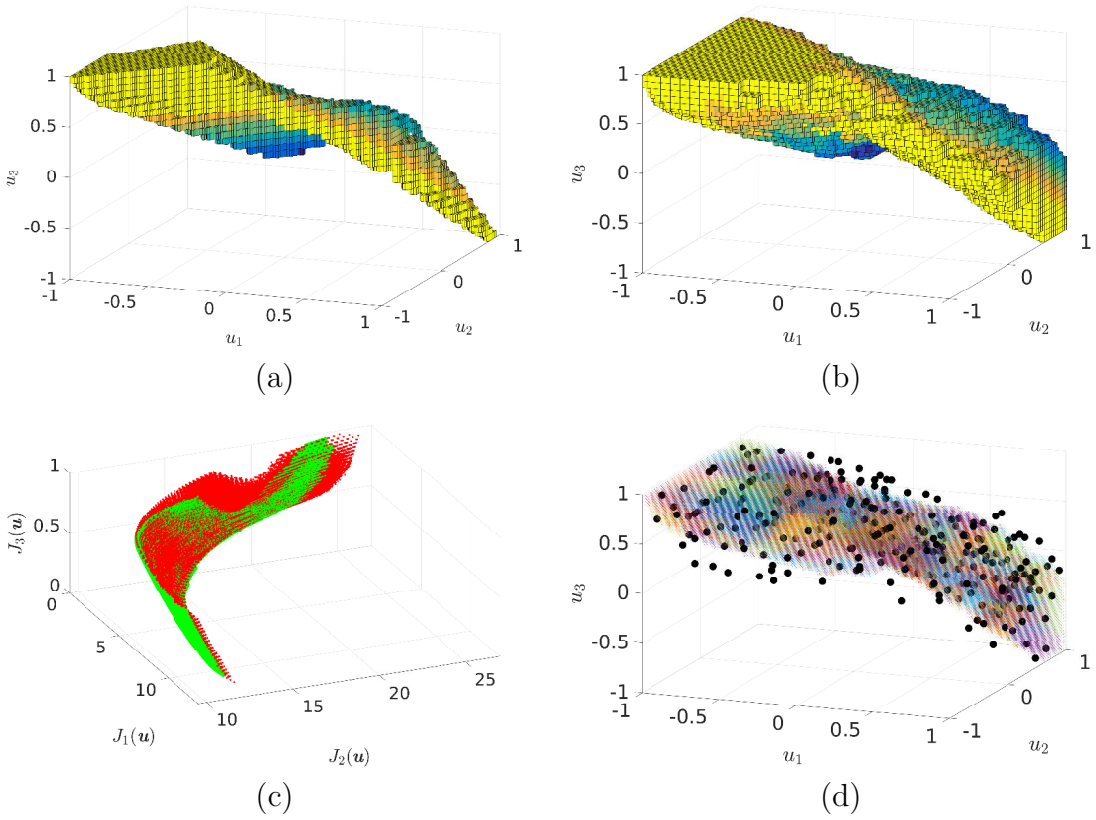


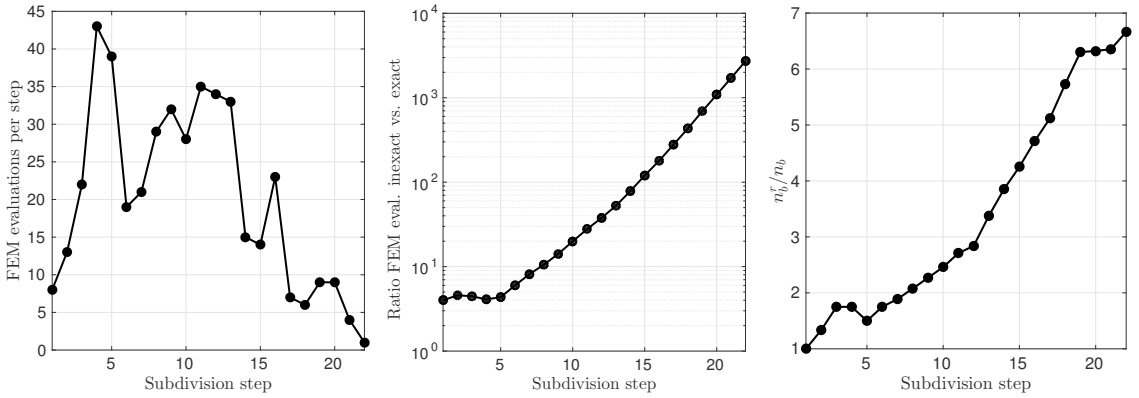
Figure 5.28: (a) Pareto set of (5.49) after 22 subdivision steps using a FEM discretization. Projection onto the first three components of $\boldsymbol{u}$, and u_4 is visualized by the box coloring. (b) The Pareto set based on local ROMs and the inexact sampling algorithm with $\boldsymbol{\kappa} = (0.025, 0.025, 0)^\top$. (c) The corresponding Pareto fronts, where the FEM-based solution is shown in green and the POD-based solution in red. The points are the images of the box centers. (d) Assignment of sample points to ROMs. Each of the colored patches has been assigned to one of the ROMs which are represented by black dots.

The initial condition is $y_0(\boldsymbol{x}) = 0$ for all $\boldsymbol{x} \in \Omega$ and box constraints $\boldsymbol{u}_\mathsf{b} = (1, 1, 1, 1)^\top$ and $\boldsymbol{u}_\mathsf{a} = -\boldsymbol{u}_\mathsf{b}$ are introduced. The maximal error is set to $\boldsymbol{\kappa} = (0.025, 0.025, 0)^\top$ where $\kappa_3 = 0$ since the ROM does not introduce an error in J_3^r. Each box B is represented by an equidistant grid of two points in each direction, i.e. by sixteen sample points in total.

The exact Pareto set of (5.49) is shown in Figure 5.28 (a), where the boxes are colored according to the fourth component u_4. The corresponding Pareto front is given in Figure 5.28 (c) in green. The Pareto set for the same problem, obtained by Algorithms 5.6 and 5.7, is shown in Figure 5.28 (b), the corresponding Pareto front

in Figure 5.28 (c) in red. A good agreement both between the Pareto sets and the Pareto fronts can be observed. The error bound $\boldsymbol{\kappa}$ for the objectives J_1^d and J_2^d is satisfied as desired. In order to also bound the error in the decision space, further assumptions on the objectives have to be made.

The locations of the reference controls $\boldsymbol{u}_{\mathsf{ref}}$ of the (remaining) ROMs are shown as black dots in Figure 5.28 (d). The colored points are the sample points evaluated in the 22^{nd} subdivision step, the coloring depends on the ROM the points were assigned to. Due to the fact that the selection is based on the Euclidean distance, the size of all the patches is comparable. Since there is no formal reason to choose this specific way of assigning sample points to ROMs, this motivates the investigation of more advanced clustering approaches in order to further reduce the number of required ROMs.



(a) (b) (c)

Figure 5.29: (a) The number of FEM evaluations in each subdivision step. (b) Ratio of the total number of high-fidelity evaluations within the FEM-based exact subdivision algorithm 2.3 and the POD-based inexact subdivision algorithm 5.6. (c) Ratio of the respective number of boxes.

For the inexact Pareto set, only 444 evaluations of the full model were required, i.e. the FEM evaluations were reduced by a factor of more than 1000. The FEM evaluations are mainly performed during the first subdivision steps (cf. Figure 5.29 (a)) whereas later, almost the entire decision space can be approximated by the existing models. This is also illustrated in Figure 5.29 (b), where an exponential increase of the ratio of high-fidelity solutions between the FEM and the POD-based approach can be observed. Therefore, it would be interesting to investigate the benefit of an offline phase similar to classical reduced basis approaches.

Due to the inexactness, the number of boxes is larger in the inexact computation, which is shown in Figure 5.29 (c). Similar to the observations in Section 5.3, this effect becomes more severe during higher order subdivision steps. All boxes in the vicinity of the exact Pareto set are not eliminated which results in a strong increase

in the number of boxes. Consequently, further research is required to address this issue.

The approach presented here is only a first step towards using local reduced bases within multiobjective optimization. It can be expected that the efficiency can further be increased by implementing more sophisticated rules for clustering the points than using the Euclidean distance. Moreover, using online enrichment (see e.g. [AHKO12, OS15]) or combining different POD bases may also be beneficial for the overall performance.

6 Conclusion and Outlook

Multiobjective optimization becomes more and more important in modern applications. During the development of a new product, for example, knowledge of the Pareto set can help the developer to judge the current design while during the operation of a system, the ability to switch between optimal compromises increases the system's flexibility. The main drawback in comparison to scalar optimization is the increased computational effort introduced by the presence of multiple criteria. This becomes even more significant when additional parameter dependencies have to be taken into account, the number of objectives is large or the underlying model is costly to solve. Hence, the question that has been addressed in this thesis is whether structural aspects, either in the multiobjective optimization or optimal control problem or in the system dynamics, can be exploited in order to develop algorithms with increased efficiency.

6.1 Continuation of Parameter Dependent Pareto Sets

An algorithm for model predictive control of nonlinear dynamical systems with respect to multiple criteria has been developed in Chapter 3. The algorithm utilizes elements from economic and explicit MPC, multiobjective optimal control and motion planning with motion primitives. To this end, a large number of MOCPs has to be solved in an offline phase. In order to reduce this number, the dynamical control system is analyzed with respect to invariances of the solution under translations in the initial conditions. This way, a known solution can be used in multiple situations. This requires a generalization of the concept of invariance in the sense that invariances of the Pareto set of an MOCP with respect to variations of a parameter have to be identified. According to a decision maker's preference the system can be controlled in real-time with respect to an optimal compromise between conflicting objectives. Using a simple heuristic for the weighting of the objectives, solutions of a quality compared to the global optimum computed by open loop dynamic programming are obtained.

In spite of exploiting invariances, the number of MOCPs that have to be solved quickly becomes very large and the computational complexity increases. Hence, only constant controls have been considered in the application of autonomously

driven electric vehicles. In order to increase the dimension of the control input, it is necessary to improve the efficiency of solving parameter dependent MOCPs. It has been shown that if the MOCP satisfies the manifold properties and both the objectives and constraints are continuously differentiable with respect to the parameter, the set of substationary points is a $(k-1+n_p)$-dimensional manifold, where n_p is the dimension of the parameter. Based on this, predictor-corrector methods can be extended to the continuation of entire Pareto sets. Despite being a heuristic based on a finite approximation of the Pareto set, the predictor step yields promising results, even in situations where the manifold properties are not globally satisfied.

Finally, another example from autonomous driving has been investigated, where a vehicle has to be steered with respect to the conflicting objectives to maximize both speed and security. The results are a first step towards multiobjective MPC of nonlinear dynamical systems.

6.2 Solving Many-Objective Optimization Problems via Subsets of Objectives

The hierarchical structure of Pareto sets has been analyzed in Chapter 4. When considering a subset of objectives, the resulting set of substationary points as well as the Pareto set are subsets of the respective solutions to the full problem. Moreover, if the problem is unconstrained and sufficiently smooth, then the set of substationary points is bounded by the union of the sets of substationary points of all subproblems where one objective is neglected. This can be exploited to effectively compute the skeleton of Pareto sets for many-objective optimization problems.

In order to compute points from the interior of the Pareto set, an extension of the well-known ϵ-constraint method has been developed where a sequence of multiobjective subproblems has to be solved. Due to the exponential increase in computational effort with the number of objectives, the interior can be approximated much faster by solving a moderate number of ϵ-constrained subproblems.

The results have been illustrated using several academic examples and an application from industrial laundries. It has been shown that – depending on the number of objectives – significant speed-up factors between 10 and more than 1000 can be achieved. Moreover, the concept is not tailored to any specific algorithm such that it can be used in combination with any method for solving MOPs.

6.3 Multiobjective Optimal Control of PDEs Using Reduced Order Modeling

Three different approaches for coupling multiobjective optimization and model order reduction have been investigated in Chapter 5.

In the first approach presented in Section 5.1, only a single ROM has been computed prior to the optimization and two different approaches for numerically solving MOCPs involving boundary control of the Navier-Stokes equations have been compared. The sampling algorithm yields a box covering of the global Pareto set which can also be disconnected. Since its applicability critically depends on both the dimension of the Pareto set and the decision space dimension, it is restricted to a moderate number of objectives as well as parameters, even in combination with reduced order modeling. Therefore the control function has to be represented by a low number of parameters, e.g. by trigonometric functions or spline coefficients. Since the reference point method converts the multiobjective optimal control problem into a sequence of scalar problems, it is also applicable to high and even infinite-dimensional problems. To incorporate gradient information, the optimality system based on the reduced state equation has been derived for the scalar problem. In this case, the accuracy of the approximated gradient is of great importance, which becomes evident when comparing two different optimality systems. Provided the gradient accuracy is sufficient, the Pareto set can be approximated very efficiently using the solution from previous scalar problems as initial guesses.

In Section 5.2, it has been shown that the trust region approach by Fahl [Fah00] can be extended to multiple objectives in a straightforward manner when scalarizing the objective function, e.g. via the reference point method. The improvement in computational efficiency is significant which is demonstrated on a heat flow problem. The advantage of this approach over computing only one model prior to the optimization is that convergence to a Pareto optimal solution can be proved. Since all scalarization methods have difficulties with a large number of objectives, the approach is limited to a small number of objectives, i.e. ideally two. However, in combination with the results from Chapter 4, the skeleton of the Pareto set of a many-objective optimization problem can be efficiently computed.

In Section 5.3, the subdivision and the sampling algorithm developed in [DSH05] have been extended to MOPs with uncertainties in the form of inexact function and gradient information. An additional condition for a descent direction has been derived in order to account for the inaccuracies in the gradients. Convergence of the inexact subdivision algorithm to a superset of the Pareto set has been proved and an upper bound for the maximal distance to the set of substationary points has been given.

In Section 5.4, the inexact version of the sampling algorithm has been applied to an MOCP with a semi-linear parabolic state equation. The inexactness is a result of applying POD-based reduced order modeling in order to accelerate cost function evaluations. Accuracy of the ROM is ensured by using a-posteriori error estimators for the state variable and the cost function values which have been coupled with a heuristic scaling factor to compensate for an almost constant over-estimation. In order to obtain ROMs of small size, a localized bases strategy has been adopted where multiple locally valid ROMs are stored in a library. This library is constructed iteratively using a worst-first Greedy approach. The numerical results confirm that the number of evaluations of the high-fidelity model can be reduced by a factor of over 1000 while the prescribed error bound for the objectives is satisfied as desired.

6.4 Future Work

The results presented in this thesis contribute to reducing the computational effort of MOPs and MOCPs with the purpose to apply multiobjective optimization in real-time applications or to consider multiple objectives in PDE-constrained problems. Nevertheless, in all three areas that have been covered, there are open questions and future directions that need to be addressed, either to further improve the efficiency or to increase the range of possible applications.

In the case of parameter dependent problems, it would be beneficial to develop a more rigorous approach to compute the tangent space of the entire Pareto set instead of using a finite-difference approximation. Moreover, adaptive strategies for the length of the predictor step (similar to the strategies presented in [AG03]) could help to further increase the efficiency due to faster convergence of the corrector step.

For the multiobjective MPC framework presented in Chapter 3, an analysis from a more theoretical point of view would be of great interest. Questions concerning feasibility, stability or optimality are crucial for the applicability of the approach to real systems. A first step would be to investigate feasibility and stability under the assumption that the solution of the required MOCP is exactly known. In the next step, this needs to be extended in the spirit of explicit MPC methods in order to account for interpolation errors in situations where the solution is only known for slightly different parameter values. Finally, it would be interesting to apply the method to the example considered in Section 3.3, where until now only one maneuver, i.e. driving through a curve, has been considered. This way efficient autonomous driving with respect to multiple objectives can be realized.

The approach to exploit the hierarchical structure of Pareto sets presented in Chapter 4 is a promising start to significantly accelerate the solution of many-objective optimization problems. Since parts of the results until now only apply to unconstrained problems, it would be of great interest to extend these to constrained problems. It can be expected that in this situation, it is significantly more difficult to compute the boundary of the set of substationary points by solving sub-problems with a reduced number of objectives. Furthermore, an algorithm which systematically constructs the Pareto set by taking more objectives into consideration in each iteration should be developed. The results can be useful for both deterministic and evolutionary approaches.

The methods presented in Chapter 5 which utilize reduced order modeling to solve PDE-constrained MOCPs have shown great potential. Depending on the dynamical system and the number of objectives, different methods for incorporating ROMs are advisable. Considering flow control problems, for example, methods based on error analysis would certainly be helpful, but until now, they are limited to special cases (e.g. steady state [VP05]). In particular, at higher Reynolds numbers the challenges for flow control and reduced order modeling increase significantly [BN15]. Therefore, it would be interesting if in this case, one could improve the results by considering another reduced order modeling approach, utilizing e.g. Koopman operator-based methods which have been briefly mentioned in Section 2.3.

The set-oriented approach to POD-based multiobjective optimal control of PDEs developed in Section 5.3 is – in the gradient-based version – until now restricted to unconstrained problems. This should certainly be addressed in the future in order to broaden the applicability. Moreover, although the error bound for the objectives is satisfied as desired, the distance in parameter space can still be significant. It would therefore be interesting to derive additional conditions under which the distance between the Pareto sets can be bounded.

Due to the inexactness in the ROM, the box covering consists of a significantly increased number of boxes, which is disadvantageous for the computational efficiency. Therefore, an adaptive strategy needs to be developed where almost Pareto optimal subsets remain within the box collection but are no longer considered in the computation. Finally, it would be interesting to investigate whether the error bound can be improved when the error satisfies a certain probability distribution.

The localized reduced basis approach presented in Section 5.4 is only a first step towards using local reduced bases within PDE-constrained multiobjective optimal control. The efficiency can probably be increased by implementing more sophisticated rules for clustering the points, by using online enrichment or by combining different ROMs. In order to use model-order reduction and in particular error estimation in the gradient-based subdivision algorithm, error estimates for the reduced

order gradients have to be developed.

Bibliography

[AB09] A. Alessio and A. Bemporad. A survey on explicit model predictive control. In L. Magni, D. M. Raimondo, and F. Allgöwer, editors, *Nonlinear Model Predictive Control: Towards New Challenging Applications*, Vol. 384, pp. 345–369. Springer Berlin Heidelberg, 2009.

[AG03] E. L. Allgower and K. Georg. *Introduction to Numerical Continuation Methods*. Society for Industrial and Applied Mathematics, 2003.

[AHKO12] F. Albrecht, B. Haasdonk, S. Kaulmann, and M. Ohlberger. The localized reduced basis multiscale method. In *Proceedings of ALGORITHMY 2012*, pp. 393–403, 2012.

[ARFL09] M. N. Albunni, V. Rischmuller, T. Fritzsche, and B. Lohmann. Multiobjective Optimization of the Design of Nonlinear Electromagnetic Systems Using Parametric Reduced Order Models. *IEEE Transactions on Magnetics*, 45(3):1474–1477, 2009.

[ASG01] A. C. Antoulas, D. C. Sorensen, and S. Gugercin. A survey of model reduction methods for large-scale systems. *Contemporary mathematics*, 280:193–220, 2001.

[AZF12] D. Amsallem, M. J. Zahr, and C. Farhat. Nonlinear model order reduction based on local reduced-order bases. *International Journal for Numerical Methods in Engineering*, 92(10):891–916, 2012.

[Ban16] S. Banholzer. *POD-Based Bicriterial Optimal Control of Convection-Diffusion Equations*. Master thesis, University of Konstanz, 2016.

[BBB+01] T. Binder, L. Blank, H. G. Bock, R. Bulirsch, W. Dahmen, M. Diehl, T. Kronseder, W. Marquardt, J. P. Schlöder, and O. von Stryk. Introduction to Model Based Optimization of Chemical Processes on Moving Horizons. In M. Grötschel, S. O. Krumke, and J. Rambau, editors, *Online Optimization of Large Scale Systems: State of the Art*, pp. 295–339. Springer Berlin Heidelberg, 2001.

[BBI09] M. Bergmann, C.-H. Bruneau, and A. Iollo. Enablers for robust POD models. *Journal of Computational Physics*, 228(2):516–538, 2009.

[BBL+02] C. H. Bischof, H. M. Bücker, B. Lang, A. Rasch, and A. Vehreschild. Combining source transformation and operator overloading techniques to compute derivatives for MATLAB programs. In *Second IEEE International Workshop on Source Code Analysis and Manipulation*, 2002.

[BBPK16] S. L. Brunton, B. W. Brunton, J. L. Proctor, and J. N. Kutz. Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control. *PLoS ONE*, 11(2):1–19, 2016.

[BBV16] S. Banholzer, D. Beermann, and S. Volkwein. POD-Based Bicriterial Optimal Control by the Reference Point Method. In *2nd IFAC Workshop on Control of Systems Governed by Partial Differential Equations*, pp. 210–215, 2016.

[BBV17] S. Banholzer, D. Beermann, and S. Volkwein. POD-Based Error Control for Reduced-Order Bicriterial PDE-Constrained Optimization. *https://kops.uni-konstanz.de/handle/123456789/37360*, 2017.

[BC08] M. Bergmann and L. Cordier. Optimal control of the cylinder wake in the laminar regime by trust-region methods and POD reduced-order models. *Journal of Computational Physics*, 227(16):7813–7840, 2008.

[BCB05] M. Bergmann, L. Cordier, and J.-P. Brancher. Optimal rotary control of the cylinder wake using proper orthogonal decomposition reduced-order model. *Physics of Fluids*, 17:1–21, 2005.

[BdJ05] P. A. N. Bosman and E. D. de Jong. Exploiting gradient information in numerical multi-objective evolutionary optimization. In *Proceedings of the 2005 conference on Genetic and evolutionary computation (GECCO 05)*, pp. 755–762, 2005.

[BDMS08] J. Branke, K. Deb, K. Miettinen, and R. Slowinski, editors. *Multiobjective Optimization*. Springer Berlin Heidelberg, 2008.

[BDPV17] D. Beermann, M. Dellnitz, S. Peitz, and S. Volkwein. Set-Oriented Multiobjective Optimal Control of PDEs using Proper Orthogonal Decomposition. *Submitted*, 2017.

[Bew01] T. R. Bewley. Flow control: New challenges for a new Renaissance. *Progress in Aerospace Sciences*, 37(1):21–58, 2001.

[BHL93] G. Berkooz, P. Holmes, and J. L. Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25:539–575, 1993.

[BM00] C. Büskens and H. Maurer. SQP-methods for solving optimal control problems with control and state constraints: adjoint variables, sensitivity analysis and real-time control. *Journal of Computational and Applied Mathematics*, 120(1-2):85–108, 2000.

[BM09a] A. Bemporad and D. Muñoz de la Peña. Multiobjective model predictive control. *Automatica*, 45(12):2823–2830, 2009.

[BM09b] A. Bemporad and D. Muñoz de la Peña. Multiobjective Model Predictive Control Based on Convex Piecewise Affine Costs. In *Proceedings of the European Control Conference*, pp. 2402–2407, 2009.

[BMS05] P. Benner, V. Mehrmann, and D. C. Sorensen, editors. *Dimension Reduction of Large-Scale Systems*. Springer Berlin Heidelberg New York, 2005.

[BMT01] T. R. Bewley, P. Moin, and R. Temam. DNS-based predictive control of turbulence: an optimal benchmark for feedback algorithms. *Journal of Fluid Mechanics*, 447:179–225, 2001.

[BN15] S. L. Brunton and B. R. Noack. Closed-Loop Turbulence Control: Progress and Challenges. *Applied Mechanics Reviews*, 67(5):1–48, 2015.

[Bos12] P. A. N. Bosman. On Gradients and Hybrid Evolutionary Algorithms. *IEEE Transactions on Evolutionary Computation*, 16(1):51–69, 2012.

[BS15] R. E. Bellmann and E. D. Stuart. *Applied dynamic programming.* Princeton University Press, 2015.

[BWC12] C. Bingham, C. Walsh, and S. Carroll. Impact of driving characteristics on electric vehicle energy consumption and range. *IET Intelligent Transport Systems*, 6(1):29–35, 2012.

[BZ06] M. Basseur and E. Zitzler. A Preliminary Study on Handling Uncertainty in Indicator-Based Multiobjective Optimization. In F. Rothlauf, J. Branke, S. Cagnoni, E. Costa, C. Cotta, R. Drechsler, E. Lutton, P. Machado, J. H. Moore, J. Romero, G. D. Smith, G. Squillero, and H. Takagi, editors, *Applications of Evolutionary Computing: EvoWorkshops 2006: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoINTERACTION, EvoMUSART, and EvoSTOC, Budapest, Hungary*, pp. 727–739, 2006.

[BZ11] J. Bader and E. Zitzler. HypE : An Algorithm for Fast Hypervolume-Based Many-Objective Optimization. *Evolutionary Computation*, 19(1):45–76, 2011.

[CAF09] L. Cordier, B. Abou El Majd, and J. Favier. Calibration of POD reduced-order models using Tikhonov regularization. *International Journal for Numerical Methods in Fluids*, 63(2):269–296, 2009.

[Car91] R. G. Carter. On the global convergence of trust region algorithms using inexact gradient information. *SIAM Journal on Numerical Analysis*, 28(1):251–265, 1991.

[CBS05] M. Couplet, C. Basdevant, and P. Sagaut. Calibrated reduced-order POD-Galerkin system for fluid flow modelling. *Journal of Computational Physics*, 207(1):192–220, 2005.

[CFCA13] K. Carlberg, C. Farhat, J. Cortial, and D. Amsallem. The GNAT method for nonlinear model reduction: Effective implementation and application to computational fluid dynamics and turbulent flows. *Journal of Computational Physics*, 242:623–647, 2013.

[CLS16] O. Cuate, A. Lara, and O. Schütze. A Local Exploration Tool for Linear Many Objective Optimization Problems. In *13th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*, 2016.

[CLV07] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen. *Evo-*

lutionary Algorithms for Solving Multi-Objective Problems, Vol. 2. Springer Science & Business Media, 2007.

[CP07] A. Chinchuluun and P. M. Pardalos. A survey of recent developments in multiobjective optimization. *Annals of Operations Research*, 154(1):29–50, 2007.

[DAR11] M. Diehl, R. Amrit, and J. B. Rawlings. A Lyapunov Function for Periodic Economic Optimizing Model Predictive Control. *IEEE Transactions on Automatic Control*, 56(3):703–707, 2011.

[DD97] I. Das and J. E. Dennis. A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems. *Structural Optimization*, 14(1):63–69, 1997.

[DD98] I. Das and J. E. Dennis. Normal-Boundary Intersection: A New Method for Generating the Pareto Surface in Nonlinear Multicriteria Optimization Problems. *SIAM Journal on Optimization*, 8(3):631–657, 1998.

[DEF+14] M. Dellnitz, J. Eckstein, K. Flaßkamp, P. Friedel, C. Horenkamp, U. Köhler, S. Ober-Blöbaum, S. Peitz, and S. Tiemeyer. Development of an Intelligent Cruise Control Using Optimal Control Methods. *Procedia Technology*, 15:285–294, 2014.

[DEF+16] M. Dellnitz, J. Eckstein, K. Flaßkamp, P. Friedel, C. Horenkamp, U. Köhler, S. Ober-Blöbaum, S. Peitz, and S. Tiemeyer. Multiobjective Optimal Control Methods for the Development of an Intelligent Cruise Control. In G. Russo, V. Capasso, G. Nicosia, and V. Romano, editors, *Progress in Industrial Mathematics at ECMI 2014 (to appear)*. Springer, 2016.

[Dés12] J.-A. Désidéri. Mutiple-Gradient Descent Algorithm for Multiobjective Optimization. In J. Eberhardsteiner, H. J. Böhm, and F. G. Rammerstorfer, editors, *European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS)*, pp. 3974–3993, 2012.

[DG05] K. Deb and H. Gupta. Searching for Robust Pareto-optimal Solutions in Multi-objective Optimization. In C. A. Coello Coello, A. Hernández Aguirre, and E. Zitzler, editors, *Evolutionary multi-criterion optimization*, pp. 150–164. Springer Berlin Heidelberg, 2005.

[DH97] M. Dellnitz and A. Hohmann. A subdivision algorithm for the computation of unstable manifolds and global attractors. *Numerische Mathematik*, 75(3):293–317, 1997.

[DKKO91] A. E. Deane, I. G. Kevrekidis, G. E. Karniadakis, and S. A. Orszag. Lowdimensional models for complex geometry flows: Application to grooved channels and circular cylinders. *Physics of Fluids A: Fluid Dynamics*, 3(10):2337–2354, 1991.

[DMM05] K. Deb, M. Mohan, and S. Mishra. Evaluating the epsilon-Domination Based Multi-Objective Evolutionary Algorithm for a Quick Com-

putation of Pareto-Optimal Solutions. *Evolutionary Computation*, 13(4):501–525, 2005.

[DMP03] Z. Denkowski, S. Migórski, and N. S. Papageorgiou. *An Introduction to Nonlinear Analysis: Theory*. Springer Science & Business Media, 2003.

[DPG] M. Dellnitz, S. Peitz, and B. Gebken. On the Hierarchical Structure of Pareto Sets. *In preparation*.

[DS83] J. E. Dennis Jr. and R. B. Schnabel. *Numerical methods for unconstrained optimization and nonlinear equations*. Prentice Hall Series in Computational Mathematics. Prentice Hall, Inc., Englewood Cliffs, NJ, 1983.

[DSH05] M. Dellnitz, O. Schütze, and T. Hestermeyer. Covering Pareto sets by Multilevel Subdivision Techniques. *Journal of Optimization Theory and Applications*, 124(1):113–136, 2005.

[DSS02] M. Dellnitz, O. Schütze, and S. Sertl. Finding zeros by multilevel subdivision techniques. *IMA Journal of Numerical Analysis*, 22(2):167–185, 2002.

[DV01] F. Diwoky and S. Volkwein. Nonlinear Boundary Control for the Heat Equation Utilizing Proper Orthogonal Decomposition. In K.-H. Hoffmann, R. H. W. Hoppe, and V. Schulz, editors, *Fast Solution of Discretized Optimization Problems: Workshop held at the Weierstrass Institute for Applied Analysis and Stochastics, Berlin, May 8-12, 2000*, pp. 73–87. Springer Basel, 2001.

[Ehr05] M. Ehrgott. *Multicriteria optimization*. Springer Berlin Heidelberg New York, 2nd edition, 2005.

[EPS+16] J. Eckstein, S. Peitz, K. Schäfer, P. Friedel, U. Köhler, M. Hessel-von Molo, S. Ober-Blöbaum, and M. Dellnitz. A Comparison of two Predictive Approaches to Control the Longitudinal Dynamics of Electric Vehicles. *Procedia Technology*, 26:465–472, 2016.

[EW07] A. Engau and M. M. Wiecek. Generating ϵ-efficient solutions in multiobjective programming. *European Journal of Operational Research*, 177(3):1566–1579, 2007.

[Fah00] M. Fahl. *Trust-region Methods for Flow Control based on Reduced Order Modelling*. Phd thesis, University of Trier, 2000.

[FDF05] E. Frazzoli, M. A. Dahleh, and E. Feron. Maneuver-based motion planning for nonlinear systems with symmetries. *IEEE Transactions on Robotics*, 21(6):1077–1091, 2005.

[FGH98] A. V. Fursikov, M. D. Gunzburger, and L. S. Hou. Boundary value problems and optimal boundary control for the Navier-Stokes system: the two-dimensional case. *SIAM Journal on Control and Optimization*, 36(3):852–894, 1998.

[FGS09] J. Fliege, L. M. Grana Drummond, and B. F. Svaiter. Newton's

method for multiobjective optimization. *SIAM Journal on Optimization*, 20(2):602–626, 2009.

[FOBK12] K. Flaßkamp, S. Ober-Blöbaum, and M. Kobilarov. Solving Optimal Control Problems by Exploiting Inherent Dynamical Systems Structures. *Journal of Nonlinear Science*, 22(4):599–629, 2012.

[FP02] J. H. Ferziger and M. Peric. *Computational Methods for Fluid Dynamics*. Springer Berlin Heidelberg, 3rd edition, 2002.

[FPL05] P. J. Fleming, R. C. Purshouse, and R. J. Lygoe. Many-Objective Optimization: An Engineering Design Perspective. In *International Conference on Evolutionary Multi-Criterion Optimization*, pp. 14–32, 2005.

[FS00] J. Fliege and B. F. Svaiter. Steepest descent methods for multicriteria optimization. *Mathematical Methods of Operations Research*, 51(3):479–494, 2000.

[GBZI04] B. Galletti, C.-H. Bruneau, L. Zannetti, and A. Iollo. Low-order modelling of laminar flow regimes past a confined square cylinder. *Journal of Fluid Mechanics*, 503:161–170, 2004.

[GeH89] M. Gad-el Hak. Flow Control. *Applied Mechanics Reviews*, 42(10):261–292, 1989.

[Ger12] M. Gerdts. *Optimal control of ODEs and DAEs*. Walter de Gruyter, 2012.

[GFDK09] J. Gausemeier, U. Frank, J. Donoth, and S. Kahl. Specification technique for the description of self-optimizing mechatronic systems. *Research in Engineering Design*, 20(4):201–223, 2009.

[GFWG10] D. Galbally, K. Fidkowski, K. Willcox, and O. Ghattas. Non-linear model reduction for uncertainty quantification in large-scale inverse problems. *International Journal for Numerical Methods in Engineering*, 81:1581–1608, 2010.

[GMNP07] M. A. Grepl, Y. Maday, N. C. Nguyen, and A. T. Patera. Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations. *ESAIM: Mathematical Modelling and Numerical Analysis*, 41(3):575–605, 2007.

[GP17] L. Grüne and J. Pannek. *Nonlinear Model Predictive Control*. Springer International Publishing, 2nd edition, 2017.

[GPT99] W. R. Graham, J. Peraire, and K. Y. Tang. Optimal Control of Vortex Shedding Using Low Order Models. Part I: Open-Loop Model Development. *International Journal for Numerical Methods in Engineering*, 44(7):945–972, 1999.

[GvL13] G. H. Golub and C. F. van Loan. *Matrix computations*. The Johns Hopkins University Press, 4th edition, 2013.

[Hil01] C. Hillermeier. *Nonlinear Multiobjective Optimization: A Generalized Homotopy Approach*. Birkhäuser, 2001.

[HLBR12] P. Holmes, J. L. Lumley, G. Berkooz, and C. W. Rowley. *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. Cambridge University Press, 2nd edition, 2012.

[HO08] B. Haasdonk and M. Ohlberger. Reduced Basis Method for Finite Volume Approximations of Parametrized Evolution Equations. *ESAIM: Mathematical Modelling and Numerical Analysis*, 42(2):277–302, 2008.

[HPUU09] M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich. *Optimization with PDE Constraints*. Springer Science+Business Media, 2009.

[HRT96] J. G. Heywood, R. Rannacher, and S. Turek. Artificial Boundaries and Flux and Pressure Conditions for the Incompressible Navier-Stokes Equations. *International Journal for Numerical Methods in Fluids*, 22(5):325–352, 1996.

[HSK06] K. Harada, J. Sakuma, and S. Kobayashi. Local Search for Multiobjective Function Optimization: Pareto Descent Method. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation (GECCO 06)*, pp. 659–666. ACM, 2006.

[HSS13] C. Hernández, J.-Q. Sun, and O. Schütze. Computing the Set of Approximate Solutions of a Multi-objective Optimization Problem by Means of Cell Mapping Techniques. In M. Emmerich, A. Deutz, O. Schütze, T. Bäck, E. Tantar, A.-A. Tantar, P. D. Moral, P. Legrand, P. Bouvry, and C. A. Coello, editors, *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation IV: International Conference held at Leiden University, July 10-13, 2013*, pp. 171–188. Springer International Publishing, 2013.

[Hug01] E. J. Hughes. Evolutionary Multi-objective Ranking with Uncertainty and Noise. In E. Zitzler, L. Thiele, K. Deb, C. A. Coello Coello, and D. Corne, editors, *Evolutionary Multi-Criterion Optimization: First International Conference, Zurich, Switzerland, March 7-9*, pp. 329–343. Springer Berlin Heidelberg, 2001.

[HV05] M. Hinze and S. Volkwein. Proper Orthogonal Decomposition Surrogate Models for Nonlinear Dynamical Systems: Error Estimates and Suboptimal Control. In P. Benner, D. C. Sorensen, and V. Mehrmann, editors, *Reduction of Large-Scale Systems*, Vol. 45, pp. 261–306. Springer Berlin Heidelberg, 2005.

[IK08] K. Ito and K. Kunisch. *Lagrange Multiplier Approach to Variational Problems and Applications*. SIAM, 2008.

[IR98] K. Ito and S. S. Ravindran. A Reduced-Order Method for Simulation and Control of Fluid Flows. *Journal of Computational Physics*, 425(2):403–425, 1998.

[IR01] K. Ito and S. S. Ravindran. Reduced Basis Method for Optimal Control of Unsteady Viscous Flows. *International Journal of Computational Fluid Dynamics*, 15(2):97–113, 2001.

[IR06] M. Ilak and C. W. Rowley. Reduced-Order Modeling of Channel Flow Using Traveling POD and Balanced POD. In *3rd AIAA Flow Control Conference*, 2006.

[IR08] M. Ilak and C. W. Rowley. Modeling of transitional channel flow using balanced proper orthogonal decomposition. *Physics of Fluids*, 20(3):034103, 2008.

[ITN08] H. Ishibuchi, N. Tsukamoto, and Y. Nojima. Evolutionary Many-Objective Optimization: A short Review. *Proceedings of the 2008 IEEE Congress on Evolutionary Computation*, pp. 2419–2426, 2008.

[ITV16] L. Iapichino, S. Trenz, and S. Volkwein. Multiobjective optimal control of semilinear parabolic problems using POD. In B. Karasözen, M. Manguoglu, M. Tezer-Sezgin, S. Goktepe, and Ö. Ugur, editors, *Numerical Mathematics and Advanced Applications (ENUMATH 2015)*, pp. 389–397. Springer, 2016.

[IUV13] L. Iapichino, S. Ulbrich, and S. Volkwein. Multiobjective PDE-Constrained Optimization Using the Reduced-Basis Method. *Advances in Computational Mathematics (to appear, preprint: http://kops.uni-konstanz.de/handle/123456789/25019)*, 2013.

[Jah06] J. Jahn. Multiobjective Search Algorithm with Subdivision Technique. *Computational Optimization and Applications*, 35(2):161–175, 2006.

[Jin11] Y. Jin. Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, 1(2):61–70, 2011.

[JJT07] H. Jasak, A. Jemcov, and Z. Tukovic. OpenFOAM : A C++ Library for Complex Physics Simulations. *International Workshop on Coupled Methods in Numerical Dynamics*, pp. 1–20, 2007.

[Kar39] W. Karush. *Minima of Functions of Several Variables with Inequalities as Side Constraints*. Master thesis, University of Chicago, Illinois, 1939.

[KB07] J. Kim and T. R. Bewley. A linear systems approach to flow control. *Annual Review of Fluid Mechanics*, 39:383–417, 2007.

[KC00] J. D. Knowles and D. W. Corne. M-PAES: A memetic algorithm for multiobjective optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 325–332, 2000.

[KGPS16] S. Klus, P. Gelß, S. Peitz, and C. Schütte. Tensor-based dynamic mode decomposition. *Submitted (preprint: arXiv:1606.06625)*, 2016.

[KHW15] T. Köthe, S. Herzog, and C. Wagner. Multi-objective shape optimization of aircraft cabin ventilation components using adjoint CFD. In *16th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, pp. 1–13. AIAA, 2015.

[Kje00] T. H. Kjeldsen. A Contextualized Historical Analysis of the

Kuhn–Tucker Theorem in Nonlinear Programming: The Impact of World War II. *Historia Mathematica*, 27(4):331–361, 2000.

[KL07] S. Kukkonen and J. Lampinen. Ranking-Dominance and Many-Objective Optimization. In *IEEE Congress on Evolutionary Computation*, pp. 3983–3990, 2007.

[Kob08] M. Kobilarov. *Discrete Geometric Motion Control of Autonomous Vehicles*. PhD thesis, University of Southern California, 2008.

[KP03] S. G. Krantz and H. R. Parks. *The Implicit Function Theorem: History, Theory, and Applications*. Birkhäuser Boston, 2003.

[KT51] H. W. Kuhn and A. W. Tucker. Nonlinear programming. In *Proceedings of the 2nd Berkeley Symposium on Mathematical and Statsitical Probability*, pp. 481–492. University of California Press, 1951.

[KV99] K. Kunisch and S. Volkwein. Control of the Burgers Equation by a Reduced-Order Approach Using Proper Orthogonal Decomposition. *Journal of Optimization Theory and Applications*, 102(2):345–371, 1999.

[KV01] K. Kunisch and S. Volkwein. Galerkin proper orthogonal decomposition methods for parabolic problems. *Numerische Mathematik*, 90(1):117–148, 2001.

[KV02] K. Kunisch and S. Volkwein. Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics. *SIAM Journal on Numerical Analysis*, 40(2):492–515, 2002.

[KV08] K. Kunisch and S. Volkwein. Proper Orthogonal Decomposition for Optimality Systems. *ESAIM*, 42(1):1–23, 2008.

[Lac61] G. V. Lachmann, editor. *Boundary layer and flow control: Its principles and application*, Vol. 2. Pergamon Press, 1961.

[Las14] O. Lass. *Reduced order modeling and parameter identification for coupled nonlinear PDE systems*. PhD thesis, University of Konstanz, 2014.

[LC97] J. H. Lee and B. Cooley. Recent advances in model predictive control. *AIChE Symposium Series*, 93(316):201–216, 1997.

[LHDvI10] F. Logist, B. Houska, M. Diehl, and J. van Impe. Fast Pareto set generation for nonlinear optimal control problems with multiple objectives. *Structural and Multidisciplinary Optimization*, 42(4):591–603, 2010.

[Lib12] D. Liberzon. *Calculus of Variations and Optimal Control Theory: A Concise Introduction*. Princeton University Press, 2012.

[LKBM05] A. V. Lotov, G. K. Kamenev, V. E. Berezkin, and K. Miettinen. Optimal control of cooling process in continuous casting of steel using a visualization-based multi-criteria approach. *Applied Mathematical Modelling*, 29(7):653–672, 2005.

[LMW12] A. Logg, K.-A. Mardal, and G. N. Wells, editors. *Automated Solutions of Differential Equations by the Finite Element Method - The FEniCS Book*. Springer Berlin Heidelberg, 2012.

[LSCS10] A. Lara, G. Sanchez, C. A. Coello Coello, and O. Schütze. HCS: A New Local Search Strategy for Memetic Multiobjective Evolutionary Algorithms. *IEEE Transactions on Evolutionary Computation*, 14(1):112–132, 2010.

[Lum67] J. L. Lumley. The structure of ingomogeneous turbulent flows. In *Atmospheric Turbulence and Radio Wave Propagation: Proceedings of the International Colloquim*, pp. 166–178, 1967.

[MDL99] N. Marco, J.-A. Désidéri, and S. Lanteri. Multi-Objective Optimization in CFD by Genetic Algorithms. *RR-3686, INRIA*, 1999.

[Mez13] I. Mezić. Analysis of Fluid Flows via Spectral Properties of the Koopman Operator. *Annual Review of Fluid Mechanics*, 45:357–378, 2013.

[Mez16] I. Mezić. On Applications of the Spectral Theory of the Koopman Operator in Dynamical Systems and Control Theory. In *IEEE 54th Annual Conference on Decision and Control (CDC)*, pp. 7034–7041, 2016.

[Mie12] K. Miettinen. *Nonlinear Multiobjective Optimization*. Springer Science & Business Media, 2012.

[MK02] X. Ma and G. E. Karniadakis. A low-dimensional model for simulating three-dimensional cylinder flow. *Journal of Fluid Mechanics*, 458:181–190, 2002.

[MKDB12] C. Masjosthusmann, U. Köhler, N. Decius, and U. Büker. A Vehicle Energy Management System for a Battery Electric Vehicle. In *2012 IEEE Vehicle Power and Propulsion Conference*, pp. 339–344. IEEE, 2012.

[MM96] K. Malanowski and H. Maurer. Sensitivity Analysis for Parametric Optimal Control Problems with Control-State Constraints. *Computational Optimization and Applications*, 5(2):253–283, 1996.

[MM02] K. Miettinen and M. M. Mäkelä. On scalarizing functions in multiobjective optimization. *OR Spectrum*, 24(2):193–213, 2002.

[MP94] H. Maurer and H. J. Pesch. Solution Differentiability for Nonlinear Parameteric Control Problems. *SIAM Journal on Control and Optimization*, 34(6):1542–1554, 1994.

[MP95] H. Maurer and H. J. Pesch. Solution differentiability for parametric nonlinear control problems with control-state constraints. *Journal of Optimization Theory and Applications*, 86(2):285–309, 1995.

[MS17] A. Martin and O. Schütze. Pareto Tracer: A Predictor Corrector Method for Multi-objective Optimization Problems. *Engineering Optimization (to appear)*, 2017.

[NAM+03] B. R. Noack, K. Afanasiev, M. Morzyński, G. Tadmor, and F. Thiele. A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. *Journal of Fluid Mechanics*, 497:335–363, 2003.

[NCM12] F. Neri, C. Cotta, and P. Moscato. *Handbook of memetic algorithms*, Vol. 379. Springer, 2012.

[NPM05] B. R. Noack, P. Papas, and P. A. Monkewitz. The need for a pressure-term representation in empirical Galerkin models of incompressible shear flows. *Journal of Fluid Mechanics*, 523:339–365, 2005.

[NW06] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer Science & Business Media, 2nd edition, 2006.

[NZP04] M. Nemec, D. W. Zingg, and T. H. Pulliam. Multipoint and Multi-Objective Aerodynamic Shape Optimization. *AIAA Journal*, 42(6):1057–1065, 2004.

[OB08] S. Ober-Blöbaum. *Discrete Mechanics and Optimal Control*. PhD thesis, University of Paderborn, 2008.

[OBRzF12] S. Ober-Blöbaum, M. Ringkamp, and G. zum Felde. Solving Multi-objective Optimal Control Problems in Space Mission Design using Discrete Mechanics and Reference Point Techniques. In *51st IEEE International Conference on Decision and Control*, pp. 5711–5716, 2012.

[OS15] M. Ohlberger and F. Schindler. Error control for the localized reduced basis multiscale method with adaptive on-line enrichment. *SIAM Journal on Scientific Computing*, 37(6):2865–2895, 2015.

[Par71] V. Pareto. *Manual of political economy*. MacMillan, 1971.

[PBK14] J. L. Proctor, S. L. Brunton, and J. N. Kutz. Dynamic mode decomposition with control. *arXiv:1409.6358v1*, 2014.

[PBK16] J. L. Proctor, S. L. Brunton, and J. N. Kutz. Generalizing Koopman Theory to allow for inputs and control. *arXiv:1602.07647v1*, 2016.

[PD15] S. Peitz and M. Dellnitz. Multiobjective Optimization of the Flow Around a Cylinder Using Model Order Reduction. In *Proceedings in Applied Mathematics and Mechanics (PAMM)*, pp. 613–614, 2015.

[PD17] S. Peitz and M. Dellnitz. Gradient-Based Multiobjective Optimization with Uncertainties. In *Proceedings of the 4th International Workshop on Numerical and Evolutionary Optimization (NEO) (to appear, preprint: arXiv:1612.03815)*. Springer, 2017.

[PF07] R. C. Purshouse and P. J. Fleming. On the Evolutionary Optimization of Many Conflicting Objectives. *IEEE Transactions on Evolutionary Computation*, 11(6):770–784, 2007.

[PGH+16] S. Peitz, M. Gräler, C. Henke, M. Hessel-von Molo, M. Dellnitz, and A. Trächtler. Multiobjective Model Predictive Control of an Industrial Laundry. *Procedia Technology*, 26:483–490, 2016.

[PLM06] R. Pepy, A. Lambert, and H. Mounier. Path Planning using a Dynamic Vehicle Model. In *2nd International Conference on Information & Communication Technologies*, pp. 781 – 786, 2006.

[POBD15] S. Peitz, S. Ober-Blöbaum, and M. Dellnitz. Multiobjective Optimal Control Methods for Fluid Flow Using Model Order Reduction. *arXiv:1510.05819*, 2015.

[Pop00] S. B. Pope. *Turbulent Flows*. Cambridge University Press, 2000.

[Pow75] M. J. D. Powell. Convergence properties of a class of minimization algorithms. In O. L. Mangasarian, R. R. Meyer, and S. M. Robinson, editors, *Nonlinear Programming 2*, pp. 1–25. Academic Press, 1975.

[PSOB+17] S. Peitz, K. Schäfer, S. Ober-Blöbaum, J. Eckstein, U. Köhler, and M. Dellnitz. A Multiobjective MPC Approach for Autonomously Driven Electric Vehicles. In *Proceedings of the 20th IFAC World Congress (to appear, preprint: arXiv:1610.08777)*, 2017.

[PW15] B. Peherstorfer and K. Willcox. Dynamic data-driven reduced-order models. *Computer Methods in Applied Mechanics and Engineering*, 291:21–41, 2015.

[PWG16] B. Peherstorfer, K. Willcox, and M. Gunzburger. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. *ACDL Technical Report TR16-1*, pp. 1–57, 2016.

[QB97] S. J. Qin and T. A. Badgwell. An overview of industrial model predictive control technology. *Control Engineering Practice*, 93(316):232–256, 1997.

[QGVW16] E. Qian, M. Grepl, K. Veroy, and K. Willcox. A Certified Trust Region Reduced Basis Approach to PDE-Constrained Optimization. *ACDL Technical Report TR16-3*, 2016.

[QHS+05] N. V. Queipo, R. T. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, and P. Kevin Tucker. Surrogate-based analysis and optimization. *Progress in Aerospace Sciences*, 41:1–28, 2005.

[RA09] J. B. Rawlings and R. Amrit. Optimizing process economic performance using model predictive control. In *Nonlinear model predictive control*, pp. 119–138. Springer Berlin Heidelberg, 2009.

[Ran00] R. Rannacher. Finite Element Methods for the Incompressible Navier-Stokes Equations. In *Fundamental directions in mathematical fluid mechanics*, pp. 191–293. Birkhäuser, 2000.

[Rav00] S. S. Ravindran. A reduced-order approach for optimal control of fluids using proper orthogonal decomposition. *International Journal for Numerical Methods in Fluids*, 34(5):425–448, 2000.

[RBW+09] C. Romaus, J. Böcker, K. Witting, A. Seifried, and O. Znamenshchykov. Optimal energy management for a hybrid energy storage system combining batteries and double layer capacitors. In *Energy Conversion Congress and Exposition, 2009. ECCE 2009. IEEE*, pp. 1640–1647. IEEE Xplore Digital Library, Los Alamitos, 2009.

[RCM04] C. W. Rowley, T. Colonius, and R. M. Murray. Model reduction for

compressible flows using POD and Galerkin projection. *Physica D: Nonlinear Phenomena*, 189:115–129, 2004.

[Rem96] D. Rempfer. Investigations of boundary layer transition via Galerkin projections on empirical eigenfunctions. *Physics of Fluids*, 8(1):175–188, 1996.

[Rem00] D. Rempfer. On Low-Dimensional Galerkin Models for Fluid Flow. *Theoretical and Computational Fluid Dynamics*, 14:75–88, 2000.

[REWH08] T. D. Robinson, M. S. Eldred, K. E. Willcox, and R. Haimes. Surrogate-Based Optimization Using Multifidelity Models with Variable Parameterization and Corrected Space Mapping. *AIAA Journal*, 46(11):2814–2822, 2008.

[RHP08] G. Rozza, D. B. P. Huynh, and A. T. Patera. Reduced Basis Approximation and a Posteriori Error Estimation for Affinely Parametrized Elliptic Coercive Partial Differential Equations. *Archives of Computational Methods in Engineering*, 15(3):229–275, 2008.

[RMB+09] C. W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, and D. S. Henningson. Spectral analysis of nonlinear flows. *Journal of Fluid Mechanics*, 641:115–127, 2009.

[Row05] C. W. Rowley. Model Reduction for Fluids, Using Balanced Proper Orthogonal Decomposition. *International Journal of Bifurcation and Chaos*, 15(3):997–1013, 2005.

[RTV17] S. Rogg, S. Trenz, and S. Volkwein. Trust-Region POD using A-Posteriori Error Estimation for Semilinear Parabolic Optimal Control Problems. *http://kops.uni-konstanz.de/handle/123456789/38240*, 2017.

[RW98] R. T. Rockafellar and R. J.-B. Wets. *Variational Analysis*, Vol. 317. Springer Berlin Heidelberg, 1998.

[Sch04] O. Schütze. *Set Oriented Methods for Global Optimization*. PhD thesis, University of Paderborn, 2004.

[Sch10] P. J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, 2010.

[SCTT08] O. Schütze, C. A. Coello Coello, E. Tantar, and E.-G. Talbi. Computing the Set of Approximate Solutions of an MOP with Stochastic Search Algorithms. In *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, pp. 713–720. ACM, 2008.

[SDD05] O. Schütze, A. Dell'Aere, and M. Dellnitz. On Continuation Methods for the Numerical Treatment of Multi-Objective Optimization Problems. In *Dagstuhl Seminar Proceedings*, 2005.

[SDT+13] D. K. Saxena, J. A. Duro, A. Tiwari, K. Deb, and Q. Zhang. Objective reduction in many-objective optimization: Linear and nonlinear algorithms. *IEEE Transactions on Evolutionary Computation*, 17(1):77–99, 2013.

[SG09] O. Sundström and L. Guzzella. A generic dynamic programming Matlab function. In *Proceedings of the 18th IEEE International Conference on Control Applications*, pp. 1625–1630, 2009.

[Sin59] H. Sinner. *Über das Waschen mit Haushaltswaschmaschinen. In welchem Umfange erleichtern Haushaltswaschmaschinen und -geräte das Wäschehaben im Haushalt?* Hamburg: Haus + Heim Verlag, 1959.

[Sir87] L. Sirovich. Turbulence and the dynamics of coherent structures part I: coherent structures. *Quarterly of Applied Mathematics*, XLV(3):561–571, 1987.

[SK04] S. Sirisup and G. E. Karniadakis. A spectral viscosity method for correcting the long-term behavior of POD models. *Journal of Computational Physics*, 194(1):92–116, 2004.

[SLC11] O. Schütze, A. Lara, and C. A. Coello Coello. On the influence of the Number of Objectives on the Hardness of a Multiobjective Optimization Problem. *IEEE Transactions on Evolutionary Computation*, 15(4):444–455, 2011.

[SLT+17] V. A. Sosa Hernández, A. Lara, H. Trautmann, G. Rudolph, and O. Schütze. The Directed Search Method for Unconstrained Parameter Dependent Multi-objective Optimization Problems. In O. Schütze, L. Trujillo, P. Legrand, and Y. Maldonado, editors, *NEO 2015: Results of the Numerical and Evolutionary Optimization Workshop NEO 2015 held at September 23-25 2015 in Tijuana, Mexico*, pp. 281–330. Springer International Publishing, 2017.

[SM08] A. Singh and B. S. Minsker. Uncertainty-based multiobjective optimization of groundwater remediation design. *Water Resources Research*, 44(2), 2008.

[SMDT03] O. Schütze, S. Mostaghim, M. Dellnitz, and J. Teich. Covering Pareto Sets by Multilevel Evolutionary Subdivision Techniques. In *International Conference on Evolutionary Multi-Criterion Optimization (EMO)*, pp. 118–132, 2003.

[SSW02] S. Schäffler, R. Schultz, and K. Weinzierl. Stochastic Method for the Solution of Unconstrained Vector Optimization Problems. *Journal of Optimization Theory and Applications*, 114(1):209–222, 2002.

[SV10] E. W. Sachs and S. Volkwein. POD-galerkin approximations in PDE-constrained optimization. *GAMM Mitteilungen*, 33(2):194–208, 2010.

[SVC09] O. Schütze, M. Vasile, and C. A. Coello Coello. Computing the Set of epsilon-efficient Solutions in Multi-Objective Space Mission Design. *Journal of Aerospace Computing, Information, and Communication*, 8(3):53–70, 2009.

[SvdVR08] W. H. A. Schilders, H. A. van der Vorst, and J. Rommes. *Model Order Reduction*. Springer Berlin Heidelberg, 2008.

[SWOBD13] O. Schütze, K. Witting, S. Ober-Blöbaum, and M. Dellnitz. Set Oriented Methods for the Numerical Treatment of Multiobjective Optimization Problems. In E. Tantar, A.-A. Tantar, P. Bouvry, P. Del Moral, P. Legrand, C. A. Coello Coello, and O. Schütze, editors, *EVOLVE - A Bridge between Probability, Set Oriented Numerics and Evolutionary Computation*, Vol. 447 of *Studies in Computational Intelligence*, pp. 187–219. Springer Berlin Heidelberg, 2013.

[TBD+17] K. Taira, S. L. Brunton, S. T. M. Dawson, C. W. Rowley, T. Colonius, B. J. McKeon, O. T. Schmidt, S. Gordeyev, V. Theofilis, and L. S. Ukeiley. Modal Analysis of Fluid Flows: An Overview. *arXiv:1702.01453v1*, 2017.

[Tei01] J. Teich. Pareto-Front Exploration with Uncertain Objectives. In E. Zitzler, L. Thiele, K. Deb, C. A. Coello Coello, and D. Corne, editors, *Evolutionary Multi-Criterion Optimization*, pp. 314–328. Springer Berlin Heidelberg, 2001.

[TL90] S. Taheri and E. H. Law. Investigation of a combined slip control braking and closed loop four wheel steering system for an automobile during combined hard braking and severe steering. *American Control Conference*, pp. 1862–1867, 1990.

[TRL+14] J. H. Tu, C. W. Rowley, D. M. Luchtenburg, S. L. Brunton, and J. N. Kutz. On Dynamic Mode Decomposition: Theory and Applications. *Journal of Computational Dynamics*, 1(2):391–421, 2014.

[Trö10] F. Tröltzsch. Optimal Control Of Partial Differential Equations. *Graduate studies in mathematics*, 112, 2010.

[TV09] F. Tröltzsch and S. Volkwein. POD a-posteriori error estimates for linear-quadratic optimal control problems. *Computational Optimization and Applications*, 44(1):83–115, 2009.

[VBB14] C. Von Lücken, B. Barán, and C. Brizuela. A survey on multi-objective evolutionary algorithms for many-objective problems. *Computational Optimization and Applications*, 58(3):707–756, 2014.

[VH83] C. Vira and Y. Y. Haimes. *Multiobjective Decision Making: Theory and Methodology*. Dover Publications Inc., 1983.

[Vol11] S. Volkwein. Model reduction using proper orthogonal decomposition. *Lecture Notes*, pp. 1–43, 2011.

[Vol15] S. Volkwein. Proper Orthogonal Decomposition zur Optimalsteuerung linearer partieller Differentialgleichungen. In D. Schröder, editor, *Elektrische Antriebe - Regelung von Antriebssystemen*, pp. 1658–1684. Springer, 2015.

[VP05] K. Veroy and A. T. Patera. Certified real-time solution of the parametrized steady incompressible Navier-Stokes equations: rigorous reduced-basis a posteriori error bounds. *International Journal for Numerical Methods in Fluids*, 47(8-9):773–788, 2005.

[Whi86] D. J. White. Epsilon efficiency. *Journal of Optimization Theory and Applications*, 49(2):319–337, 1986.

[Wie80] A. P. Wierzbicki. The Use of Reference Objectives in Multiobjective Optimization. In G. Fandel and T. Gal, editors, *Multiple criteria decision making theory and application*, pp. 468–486. Springer Berlin Heidelberg, 1980.

[Wie86] A. P. Wierzbicki. On the Completeness and Constructiveness of Parametric Characterizations to Vector Optimization Problems. *OR Spektrum*, 8:73–87, 1986.

[Wit12] K. Witting. *Numerical algorithms for the treatment of parametric multiobjective optimization problems and applications*. PhD thesis, University of Paderborn, 2012.

[WKR15] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley. A Data–Driven Approximation of the Koopman Operator: Extending Dynamic Mode Decomposition. *Journal of Nonlinear Science*, 25(6):1307–1346, 2015.

[WP02] K. Willcox and J. Peraire. Balanced Model Reduction via the Proper Orthogonal Decomposition. *AIAA Journal*, 40(11):2323–2330, 2002.

[XFB+14] D. Xiao, F. Fang, A. G. Buchan, C. C. Pain, I. M. Navon, J. Du, and G. Hu. Non-linear model reduction for the Navier–Stokes equations using residual DEIM method. *Journal of Computational Physics*, 263:1–18, 2014.

[YLLZ13] S. Yang, M. Li, X. Liu, and J. Zheng. A Grid-Based Evolutionary Algorithm for Many-Objective Optimization. *IEEE Transactions on Evolutionary Computation*, 17(5):721–736, 2013.

[YM13] Y. Yue and K. Meerbergen. Accelerating Optimization of Parametric Linear Systems by Model Order Reduction. *SIAM Journal on Optimization*, 23(2):1344–1370, 2013.

[ZFT12] V. M. Zavala and A. Flores-Tlacuahuac. Stability of multiobjective predictive control: A utopia-tracking approach. *Automatica*, 48(10):2627–2632, 2012.