INAUGURAL–DISSERTATION

zur

Erlangung der Doktorwürde

der

Naturwissenschaftlich–Mathematischen Gesamtfakultät

der

RUPRECHT–KARLS–UNIVERSITÄT
HEIDELBERG

vorgelegt von

Dipl.-Math. Kathrin Hatz

aus Baden-Baden

Tag der mündlichen Prüfung:

....

Efficient numerical methods for hierarchical dynamic optimization with application to cerebral palsy gait modeling

Gutachter: Professor Dr. Dres. h.c. Hans Georg Bock

Zusammenfassung

Das Ziel der vorliegenden Arbeit ist die Entwicklung effizienter mathematischer Methoden zum Lösen von hierarchischen dynamischen Optimierungsproblemen. Hierbei ist die zugrundeliegende Motivation die Modellierung von dynamischen Prozessen in der Natur, welche der begründeten Annahme unterliegen, dass sie optimal ablaufen. Modelle solcher Prozesse beschreiben wir durch Optimalsteuerungsprobleme (sogenannte *Optimalsteuerungsmodelle (OSM)*). Jedoch enthalten OSM häufig Parameter, die nicht vollständig auf theoretischer Ebene hergeleitet werden können, was insbesondere für die Kostenfunktion gilt. Daher werden in der vorliegenden Arbeit Parameterschätztechniken zur Bestimmung der unbekannten Größen des OSM aus Beobachtungsdaten des Prozesses entwickelt. Mathematisch führt dies zu einem hierarchischen dynamischen Optimierungsproblem, das aus einem Parameterschätzproblem auf der oberen Ebene und einem parameterabhängigen Optimalsteuerungsproblem auf der unteren Ebene besteht. Hierbei betrachten wir mehrphasige gleichungs- und ungleichungsbeschränkte Optimalsteuerungsprobleme basierend auf nichtlinearen gewöhnlichen Differentialgleichungen.

Die wesentlichen Ziele dieser Arbeit umfassen die Entwicklung numerisch effizienter mathematischer Methoden zum Lösen von hierarchischen dynamischen Optimierungsproblemen und die Anwendung dieser Methoden zur Schätzung von Parametern in hochdimensionalen OSM aus realen Messdaten. Genauer werden parameterabhängige OSM für den Gang von Zerebralparesepatienten und nicht behinderten Probanden entwickelt. Die in den OSM enthaltenen unbekannten Parameter werden dann aus realen Bewegungserfassungsdaten des Ganganalyselabors HEIDELBERG MOTIONLAB der Orthopädischen Universitätsklinik Heidelberg mit den entwickelten mathematischen Methoden geschätzt.

Die wesentlichen Neuerungen und Beiträge dieser Arbeit im Gebiet der hierarchischen dynamischen Optimierung sind im Folgenden zusammengefasst.

- Basierend auf der direkten Mehrzielmethode und Optimalitätsbedingungen erster Ordnung wird eine neuartige mathematische Methode, ein sogenannter *direkter simultaner Ansatz*, zum Lösen von hierarchischen dynamischen Optimierungsproblemen entwickelt.

- Zusätzlich wird ein effizienter numerischer Algorithmus für große hierarchische nichtlineare dynamische Optimierungsprobleme vorgestellt, der die von Diskretisierung und Mehrstufigkeit vererbten Strukturen vollständig ausnutzt.

- Pontrjagins Maximumprinzip dient als Basis für die Analyse von Eigenschaften der Lösungen von hierarchischen dynamischen Optimierungsproblemen, wie beispielsweise die Erfüllung von Optimalitätsbedingungen zweiter Ordnung auf der unteren Problemebene.

- Des Weiteren werden alternative Methoden zur hierarchischen dynamischen Optimierung vorgestellt und diskutiert, die die zweistufige Problemstellung beibehalten und das unterliegende Problem nicht durch Optimalitätsbedingungen erster Ordnung charakterisieren. Die Methoden basieren auf ableitungsfreier Optimierung und einem Subgradientenverfahren.

- Eine neuartige Liftingmethode zur Regularisierung von mathematischen Programmen mit Komplementaritätsnebenbedingungen wird entwickelt, ausführlich diskutiert und mithilfe einer anerkannten Kollektion an Testproblemen numerisch untersucht.

- Beweise für Regularitäts- und Konvergenzresultate für die sequentielle quadratische Programmierung angewendet auf geliftete mathematische Programme mit Komplementaritätsnebenbedingungen werden vorgestellt.

- Effiziente state-of-the-art Implementierungen aller im Rahmen dieser Arbeit entwickelten Algorithmen, sowie eine als Benchmark dienende Problemsammlung von hierarchischen dynamischen Optimierungsproblemen werden erstellt.

- Hochdimensionale OSM für den Gang von Zerebralparesepatienten und nicht behinderten Probanden werden entwickelt. Basierend auf den in dieser Arbeit hergeleiteten mathematischen Methoden werden unbekannte Modellparameter aus echten Bewegungserfassungsdaten des Ganganalyselabors Heidelberg MotionLab der Orthopädischen Universitätsklinik Heidelberg geschätzt.

Die in dieser Arbeit vorgestellten theoretischen und praktischen Resultate können als erster und motivierender Schritt in Richtung der Beantwortung offener aktueller Forschungsfragen der Medizin in Bereichen wie der Behandlungsplanung, der Klassifizierung von Gangarten, oder der Evaluation von Behandlungsresultaten durch die hierarchische dynamische Optimierung gesehen werden.

Abstract

This thesis aims at developing efficient mathematical methods for solving hierarchical dynamic optimization problems. The main motivation is to model processes in nature, for which there is evidence to assume that they run optimally. We describe models of such processes by optimal control problems (called *optimal control models (OCMs)*). However, an OCM typically includes unknown parameters that cannot be derived entirely on a theoretical basis, which is in particular the case for the cost function. Therefore, we develop parameter estimation techniques to estimate the unknowns in an OCM from observation data of the process. Mathematically, this leads to a hierarchical dynamic optimization problem with a parameter estimation problem on the upper level and an optimal control problem on the lower level. We focus on multi-stage equality and inequality constrained optimal control problems based on nonlinear ordinary differential equations.

The main goal of this thesis is to derive numerically efficient mathematical methods for solving hierarchical dynamic optimization problems, and to use these methods to estimate parameters in high-dimensional OCMs from real-world measurement data. We develop parameter-dependent OCMs for the gait of cerebral palsy patients and able-bodied subjects. The unknown parameters in the OCMs are then estimated from real-world motion capture data provided by the Heidelberg MotionLab of the Orthopedic University Clinic Heidelberg by using the mathematical methods developed within this work.

The main novelties and contributions of this thesis to the field of hierarchical dynamic optimization are summarized herein.

- We establish a novel mathematical method, a so-called *direct all-at-once approach*, for solving hierarchical dynamic optimization problems based on the direct multiple shooting method and first-order optimality conditions.
- Furthermore, we propose an efficient numerical algorithm for large-scale hierarchical dynamic optimization problems, which fully exploits the structures inherited from both the hierarchical setting and the discretization.
- Pontryagin's maximum principle is used to analyze solution properties of hierarchical dynamic optimization problems like second-order optimality conditions of the lower-level problem.
- In addition, we propose and discuss alternative methods for hierarchical dynamic optimization that are based on derivative-free optimization and a bundle approach. These methods keep the hierarchical problem setting and do not reformulate the lower-level problem using first-order optimality conditions.
- We establish a novel lifting method for regularizing mathematical programs with complementarity constraints, which is discussed and numerically investigated by means of a well-known collection of benchmark problems.
- Proofs of regularity and convergence results for sequential quadratic programming methods applied to lifted mathematical programs with complementarity constraints are provided.
- Efficient state-of-the-art implementations of all mathematical methods derived in this thesis, as well as a benchmark collection of hierarchical dynamic optimization problems are presented.
- High-dimensional optimal control gait models for cerebral palsy patients and able-bodied subjects are developed. The mathematical methods derived in this thesis are used to estimate the unknown model parameters from real-world motion capture data provided by the Heidelberg MotionLab of the Orthopedic University Clinic Heidelberg.

The theoretical and practical results presented in this thesis can be considered an initial motivating step towards answering open questions in current medical research in fields like treatment planning, classification of gaits, or the evaluation of surgeries by means of hierarchical dynamic optimization.

Contents

Chapter 1.

Introduction

There is evidence to assume that many processes in nature run optimally. This optimality assumption is the basis of many fields of research as, for example, of the field of bionics, where observations from nature are combined with state-of-the-art technology (cf., e.g., [Dic99]). Early works in bionics go back to Leonardo da Vinci, who constructed flying machines and ships following the example set by birds and fishes in the 16th century [RB70]. The following famous quote by Leonhard Euler goes along the same lines formulating the assumption that everything in nature has been optimized.

> *Namely, because the shape of the whole universe is the most perfect and, in fact, designed by the wisest creator, nothing in all the world will occur in which no maximum or minimum rule is somehow shining forth...*
>
> Leonhard Euler (1744)

In this thesis, we are interested in deriving mathematical models for optimal processes in nature. Therefore, we propose a very special type of model – a model, which itself is a dynamic optimization problem, namely an optimal control problem (OCP). OCPs describe the task of finding inputs or controls of a dynamic system modeled by nonlinear differential equations, which minimize a cost function and satisfy possible further equality and inequality constraints (cf., e.g., [PBGM62, BH75, Boc78, Boc81b]). In this thesis, we focus on multi-stage equality and inequality constrained OCPs based on ordinary differential equations (ODEs).

There is a growing interest in modeling real processes based on ODEs in a variety of disciplines. A mathematical model provides scientific insight into the nature of a process, and often it can be used to predict the system's behavior in a specific situation. Therefore, it is very important that a model correctly describes the behavior of the real process. In many cases, however, the model includes system parameters that cannot completely be derived on a theoretical basis. Therefore, it is necessary to estimate unknown model parameters from the observation of the real process, which describes the general idea of parameter estimation (cf., e.g., [Bar74, Boc81a, SB83, Sch88]).

An optimal control model describing a process in nature might include unknown system parameters in the underlying ODE, but most importantly, we do not know the criterion that is optimized by the process. Based on the assumption that a mixture of subcriteria is optimized (cf., e.g., [Tod04]), our goal is to estimate the weight of each subcriterion (i.e. its relative importance) from observation data. This task leads to a class of challenging hierarchical dynamic optimization problems, where the upper level consists of a parameter estimation problem and the lower level is given by an OCP with mixed control-state constraints. This class of hierarchical dynamic optimization problems combines difficulties from the field of parameter estimation, optimal control and bilevel optimization (cf., e.g., [Dem02]), and we have to deal with this combination of challenges. Works in the field of hierarchical dynamic optimization include [FLHS10, MS10, ALU12, Kna12] and [HSB12].

The main goal of this thesis is

- to derive numerically efficient mathematical methods for solving hierarchical dynamic optimization problems
- and to estimate parameters in high-dimensional optimal control models for cerebral palsy gaits and healthy gaits from real-world motion capture data provided by the Heidelberg MotionLab using the mathematical methods derived in this thesis.

We focus on the derivation of a so-called *direct all-at-once* approach for solving hierarchical dynamic optimization problems, which first parameterizes the lower-level OCP based on multiple shooting [BP84] and discretizes the controls and further constraints, and then replaces the lower level by first-order optimality conditions of the parameterized and discretized OCP. This leads to a highly structured nonlinear program (NLP), which is then solved with a tailored Gauß-Newton-type method (cf. [Boc87]). The structure of the NLP is analyzed in detail and used to derive a numerically efficient algorithm for solving hierarchical dynamic optimization problems. We furthermore derive three alternative methods for hierarchical dynamic optimization including an *indirect all-at-once approach*, a *bilevel approach* based on derivative-free optimization [KLM$^+$10] and a bilevel method built on a bundle technique [SZ92]. We provide state-of-the-art implementations of all hierarchical dynamic optimization methods derived in this thesis, and their performance is compared to each other for illustrative examples. Furthermore, we present a set of hierarchical dynamic optimization benchmark problems for which we discuss the performance of our direct all-at-once approach in detail.

The direct-all-at-once approach for hierarchical dynamic optimization problems gives rise to a mathematical program with complementarity constraint (MPCC), which requires a special treatment since it violates standard assumption in optimization like the linear independence constraint qualifications (cf., e.g., [LPR96]). Common numerical approaches for solving MPCCs are based on branch-and-bound methods [Bar88], nonsmooth optimization [OKZ98], interior-point methods [LLCN06, RB05], piecewise sequential quadratic programming (SQP) techniques [LPR97], penalization, regularization and relaxation techniques [Sch01, LF03, DFNS05, SU10, HKS13, HLSB13], or SQP methods [Ley03, FLRS06]. Most of these approaches require a computational effort that is significantly larger than the one of an SQP method applied to an MPCC. Furthermore, the numerical experience with SQP methods applied to MPCCs is quite promising [FL02b]. Hence, we follow the latter approach. In this thesis, we propose a novel lifting technique for general MPCCs and for MPCCs arising from bilevel NLPs, which allows us to prove regularization and convergence results for SQP methods applied to MPCCs. Numerical results for the lifting technique applied to a large collection of MPCCs (the MacMPEC collection [Ley00]) are provided.

The direct all-at-once approach for hierarchical dynamic optimization including the lifting technique for the complementarity constraint is implemented in the C++ software package ParaOCP, which has the capability to deal with lower-level multi-stage OCPs with discontinuous transitions expressed by nonlinear transition conditions, underlying ODEs, a Bolza-type objective function, nonlinear mixed control-state constraints, and coupled or decoupled nonlinear equality and inequality multi-point constraints.

Our motivation for the derivation of mathematical methods for solving hierarchical dynamic optimization problems is the estimation of parameters in a new and powerful type of model for processes in nature, for which there is evidence to assume that they run optimally. As main area of application, we consider different types of human locomotion. Based on the assumption that humans walk optimally minimizing a mixture of certain subcriteria (cf. [Tod04]), we aim at deriving optimal control models for human gait. In particular, we consider two types of gait: the gait of cerebral palsy (CP) patients and of able-bodied subjects. CP refers to a large group of disorders of movement and posture that lead to a so-called *crouched gait* (cf., e.g., [Gag91]). We derive high-dimensional optimal control models for the gait of cerebral palsy patients and the gait of able-bodied subjects and estimate unknown model parameters from real-world motion capture data from the Heidelberg MotionLab of the Orthopedic University Clinic Heidelberg [Wol93]. The direct all-at-once approach derived in this thesis is used for solving the respective hierarchical dynamic optimization problem. The estimation of parameters in the optimal control gait models leads to large-scale hierarchical dynamic optimization problems, which make highly efficient numerical methods indispensable. Each optimal control gait model is built on a detailed multibody-system model (cf., e.g., [Cra89, ABEvS93, vS99, Sha05]), which describes the dynamics of the respective subject. In literature, there is a variety of models to analyze human locomotion ranging from simple inverted pendulum models or mass-spring systems ([GSB06, Kuo07]) to highly complex multibody-system models including muscles [DAA$^+$07]. But they all have in common that they only describe the interaction between different parts of the body formulated as, e.g., a boundary value problem. However, in this thesis, we are interested in more powerful models: optimal control models that describe a human's gait. Such models have the potential to contribute to open questions in current research in the field of gait analysis, or more general, in the field of biomechanics and medicine including the general understanding of human motion, the development of categories/classification schemes for human locomotion, the derivation of criteria to evaluate the success of treatments, and most importantly, model-based treatment planning. Contributions of hierarchical dynamic optimization to answering open questions in biomechanics and medicine are motivating long-term goals. This thesis can be seen as a first step into that direction, which also reveals open questions for future research.

1.1. Results of this thesis

We present a novel mathematical method for solving large hierarchical dynamic optimization problems, discuss alternative approaches, develop the associated theory including necessary conditions for optimality or the treatment of the complementarity constraint, and accomplish a state-of-the-art implementation of the corresponding structure-exploiting algorithm. In addition, the proposed method is applied to estimate parameters in large-scale optimal control gait models from real-world motion capture data for a cerebral palsy patient and an able-bodied subject. The main novelties presented in this thesis are described in more detail in the following.

A mathematical method for solving hierarchical dynamic optimization problems

We derive a direct all-at-once method for hierarchical dynamic optimization, where *direct* means that we directly iterate on the controls and the parameters, and *all-at-once* refers to the fact that we aim at optimality and feasibility at once. The main focus of this thesis is on hierarchical dynamic optimization problems with a parameter estimation problem on the upper level and an OCP on the lower level, and the solution strategy can be summarized by three main steps: the parameterization of the hierarchical dynamic optimization problem based on multiple shooting and an appropriate control and constraint discretization, the reformulation of the bilevel problem as one-level problem by replacing the lower level by a suitable formulation of its first-order optimality conditions, and the solution of the resulting highly structured NLP using a tailored Gauß-Newton-type method.

A structure-exploiting numerical algorithm for hierarchical dynamic optimization

Our goal is to reliably solve large hierarchical dynamic optimization problems in order to estimate parameters in optimal control models for real-world applications, which makes an efficient and structure-exploiting numerical algorithm indispensable. A direct all-at-once approach applied to a hierarchical dynamic optimization problem leads to a highly structured NLP. The structure is inherited from the state parameterization and the control and constraint discretization, as well as from the bilevel setting. We derive a numerical algorithm corresponding to the direct all-at-once method for hierarchical dynamic optimization, which exploits the given structure in numerous parts of the algorithm as, e.g., in the numerical linear algebra we use or in the computation of derivative approximations.

Insights into hierarchical dynamic optimization based on Pontryagin's maximum principle

We describe an indirect all-at-once approach for hierarchical dynamic optimization, where in the first step, the lower-level OCP is characterized by its first-order optimality conditions based on Pontryagin's maximum principle, which transforms the hierarchical problem into a one-level problem. The indirect all-at-once approach for hierarchical dynamic optimization problems is compared to its direct alternative and is furthermore used to analyze properties of the solution as, e.g., whether second-order sufficient conditions of the lower-level OCP are satisfied.

Two bilevel methods for hierarchical dynamic optimization

We derive two bilevel methods for solving hierarchical dynamic optimization problems that keep the hierarchical structure of the problem, and completely solve the lower-level OCP in each iteration of the upper-level problem. We describe a derivative-free-optimization-based method, and an approach built on a bundle technique. Both approaches are derived for comparing them to our direct all-at-once method. The differences between the three approaches and their numerical performance are discussed in detail.

A novel lifting technique for regularizing mathematical programs with complementarity constraints

A novel lifting technique for bilevel NLPs and a generalization for MPCCs that do not necessarily arise from bilevel programs is presented. The numerical performance of the lifting approach is analyzed in detail for all problems of the MacMPEC collection. The numerical experiments reveal that lifting leads to a faster convergence of SQP methods applied to MPCCs and helps to avoid infeasible quadratic programming approximations that are due to the lack of common constraint qualifications. Furthermore, we integrate the lifting idea into the efficient direct all-at-once approach for hierarchical dynamic optimization problems developed in this thesis.

Theoretical analysis of sequential quadratic programming methods applied to lifted mathematical programs with complementarity constraints

We prove that the lifted formulation of MPCCs proposed in this thesis guarantees a constraint qualification tailored to MPCCs under mild conditions. We furthermore generalize the local convergence results for SQP methods applied to MPCCs stated in [FLRS06].

Efficient implementation of all mathematical methods proposed in this thesis

We provide an efficient state-of-the-art C++ implementation of the direct all-at-once approach for hierarchical dynamic optimization with multi-stage OCPs with discontinuous stage transitions, underlying ODEs, a Bolza-type objective function, nonlinear mixed control-state constraints and coupled or decoupled nonlinear equality and inequality multi-point constraints on the lower level. The implementation also includes the lifting approach for the complementarity constraint, and modules for solving one-level OCPs and parameter estimation problems.

The derivative-free-optimization-based bilevel approach for hierarchical dynamic optimization is implemented in Matlab based on a derivative-free optimization solver called POUNDerS [KLM+10]. We furthermore provide a Matlab implementation of the bundle-based bilevel approach following [SZ92].

A collection of hierarchical dynamic optimization problems as benchmark

A collection of illustrative hierarchical dynamic optimization problems is presented as benchmark in this thesis. For each problem of the collection, we provide a full description of the problem formulation and discuss the performance of the direct all-at-once approach.

Optimal control models for human gait based on motion capture data

The direct all-at-once method for hierarchical dynamic optimization is used to estimate parameters in two large optimal control models from real-world data. We start with the derivation of multibody-system models describing the dynamics of the human body for a cerebral palsy patient and an able-bodied subject. The two multibody-system models serve as basis for two optimal control gait models including parameters that are estimated from real-world motion capture data provided by the Heidelberg MotionLab [Wol93]. The resulting optimal control gait models reveal surprising insights as, e.g., that CP patients mainly maximize stability, whereas healthy subjects primarily minimize the total energy consumption. These results can be seen as a first and motivating step into the direction of answering open questions in current biomechanical and medical research.

1.2. Thesis overview

This thesis consists of four main parts entitled *theoretical foundations*, *mathematical methods*, *modeling human locomotion in a new way* and *numerical results*. The detailed structure of all four parts is explained in the following.

This introduction is followed by Chapter 2, which is concerned with basic definitions and methods from the field of nonlinear optimization including the SQP and the Generalized Gauß-Newton method.

The focus of Chapter 3 is on dynamic optimization, starting with an introduction to the numerics of boundary value problems, which builds a basis for the following sections on the theory and the numerical treatment of optimal control problems and parameter estimation problems.

We then enter Part II, which assembles all novel mathematical methods derived in this work. Chapter 4 can be seen as an introduction to the field of hierarchical dynamic optimization including the general problem formulation, a discussion of challenges of this problem class, a detailed survey of literature covering all contributing fields, a brief summary of all mathematical methods for hierarchical dynamic optimization derived in this thesis and an overview of possible areas of application including modeling human gaits by means of OCPs, which is done in Chapters 12 and 15.

Chapter 5 is concerned with the central algorithm derived in this thesis: a direct all-at-once approach for hierarchical dynamic optimization. We start with an introductory sketch of the mathematical method, followed by a detailed description of the three main steps: the parameterization and discretization, the formulation of necessary conditions for optimality of the lower-level problem, and the solution of the resulting NLP including a detailed analysis of the NLP's structure inherited by the parameterization/discretization and the hierarchical setting. We furthermore present a structure-exploiting algorithm that corresponds to the direct all-at-once approach derived in this chapter.

Chapter 6 presents an alternative all-at-once approach for hierarchical dynamic optimization problems with an indirect treatment of the lower-level problem based on Pontryagin's maximum principle. The indirect treatment of the lower-level OCP is compared to the direct treatment presented in Chapter 5 and used to test second-order optimality conditions of the lower-level problem in the solution of the hierarchical dynamic optimization problem for an illustrative example.

In Chapter 7, we present two bilevel approaches for hierarchical dynamic optimization, starting with a derivative-free-optimization-based approach. Furthermore, a bundle-based bilevel approach is presented. Both methods are compared to the direct all-at-once approach derived in Chapter 5.

Chapter 8 is concerned with a novel lifting technique for bilevel NLPs. We first continue the discussion on numerical methods for MPCCs and their limits from Chapter 2. Afterwards, the idea of lifting a bilevel NLP for regularity is derived in detail and its effect is demonstrated for an illustrative example. This is followed by a proof that guarantees an adapted constraint qualification for the lifted problem under mild conditions. After generalizing the lifting idea to MPCCs that do not necessarily arise from bilevel NLPs, we extend the existing convergence theory for SQP methods applied to MPCCs for the lifted formulation of the problem. The last section of Chapter 8 is concerned with lifting the complementarity constraint in the solution framework of a hierarchical dynamic optimization problem based on the direct all-at-once approach derived in Chapter 5.

Chapter 9 presents the implementation of the direct all-at-once method for hierarchical dynamic optimization introduced in Chapter 5. We discuss the problem formulation that can be treated by this software package, explain the software architecture and its modules, and demonstrate the user interface for an illustrative example.

Part III is entitled *Modeling human locomotion in a new way.* The first chapter in Part III, Chapter 10, introduces the main concepts of modeling the dynamics of human motion and locomotion as constrained multibody system, which is based on the approximation of the human skeleton as rigid bodies that are coupled via joints and forces acting on these joints.

Chapter 11 provides an overview of causes, symptoms and treatments of cerebral palsy. Furthermore, the main characteristics of the gait of cerebral palsy patients are analyzed.

Chapter 12 combines the two previous chapters and presents two optimal control gait models, one for a cerebral palsy patient and one for an able-bodied subject. We first derive two multibody-system models for the underlying dynamics of the respective gait. Both multibody-system models change during a gait cycle depending on the foot that currently touches the ground. Therefore, we introduce five model stages and derive implicit stage transition conditions. Afterwards, the subcriteria involved in the lower-level objective function are discussed and finally, we present a full formulation of two optimal control models. Both optimal control models include unknown parameters that are estimated from real-world motion capture data in Chapter 15.

The last part of this thesis contains numerical results. Chapter 13 shows the numerical performance of the lifting approach presented in Chapter 8 for all MPCCs and bilevel NLPs from the MacMPEC collection. Furthermore, the lifting idea is applied to an illustrative hierarchical dynamic optimization problem, and the results are discussed in detail.

Chapter 14 describes four hierarchical dynamic optimization benchmark problems, discusses their properties and explains why these problems have been chosen. The direct all-at-once approach derived in Chapter 5 is applied to all four problems, and the numerical performance is analyzed in detail. The last section of this chapter shows the performance of the two bilevel approaches presented in Chapter 7 for a hierarchical dynamic optimization benchmark problem. The results are compared to the performance of the direct all-at-once approach, and advantages and disadvantages of all three methods are discussed extensively.

In Chapter 15, the results of Chapters 12 and 5 are combined to estimate unknown parameters in the optimal control gait model for a cerebral palsy patient and an able-bodied subject. Therefore, we use Vicon motion capture data provided by the Heidelberg MotionLab. The measurement's properties are explained in detail in the first section of this chapter. The following two sections are concerned with the estimation of parameters in the high-dimensional optimal control model for a cerebral palsy patient and for an able-bodied subject using the direct all-at-once approach from Chapter 5 and its implementation described in Chapter 9. We furthermore present image sequences from videos demonstrating the small deviation between the optimal control gait model and the corresponding motion capture measurements. This chapter is completed with a discussion of the results and how they contribute to open questions in biomechanics and medicine.

1.3. Danksagung

Ich bedanke mich an dieser Stelle herzlich bei Prof. Dr. Dres. h.c. Hans Georg Bock und Dr. Johannes P. Schlöder für ihre aktive fortwährende Unterstützung und Förderung seit vielen Jahren, sowie für die Möglichkeit an der Fakultät für Mathematik und Informatik der Ruprecht-Karls-Universität Heidelberg zu promovieren. Zahlreiche Ratschläge, Hinweise und inspirierende sowie kritische Diskussionen haben mein Promotionsprojekt, aber auch meine persönliche Entwicklung entscheidend geprägt.

Mein besonderer Dank gilt außerdem Prof. Dr. Sven Leyffer für eine sehr lehrreiche und ergiebige Zusammenarbeit, und für die Möglichkeit eines Forschungsaufenthalts am *Argonne National Laboratory, USA*, der finanziell durch das *U.S. Department of Energy* unterstützt wurde.

Des Weiteren möchte ich mich herzlich bei Prof. Dr. Katja Mombaur und ihrer gesamten Arbeitsgruppe *Optimization in Robotics and Biomechanics* für ausgiebige Diskussionen über die Modellierung des menschlichen Gangs und regelmäßige Treffen in entspannter Atmosphäre bedanken. Hervorheben möchte ich Martin Felis, der nicht zuletzt durch die Bereitstellung hilfreicher Softwaretools zu dieser Arbeit beigetragen hat.

Prof. Dr. Sebastian Sager und seine Arbeitsgruppe *Mathematical Algorithmic Optimization* haben mir die Möglichkeit gegeben, ein sehr spannendes und produktives Jahr an der Otto-von-Guericke-Universität Magdeburg zu verbringen, wofür ich mich herzlich bedanken möchte. Für finanzielle Unterstützung bin ich der Otto-von-Guericke-Universität Magdeburg zu Dank verpflichtet.

Dr. Sebastian Wolf und den Mitgliedern des Ganganalyselabors Heidelberg MotionLab der Orthopädischen Universitätsklinik Heidelberg danke ich für die Bereitstellung ihrer Expertise im Bereich von Zerebralpareseerkrankungen und allerlei zugehöriger Materialien, sowie für die Möglichkeit mit echten Bewegungserfassungsdaten von Patienten und Probanden des Labors zu arbeiten.

Mein Dank gilt außerdem der Graduiertenschule *Heidelberg Graduate School for Mathematical and Computational Methods for the Sciences*, dem Interdisziplinären Zentrum für wissenschaftliches Rechnen und der Ruprecht-Karls-Universität Heidelberg für die wissenschaftliche und finanzielle Unterstützung in der Zeit meiner Promotion.

Des Weiteren bedanke ich mich bei allen Mitgliedern und Mitarbeitern der Arbeitsgruppe Simulation und Optimierung und der angegliederten Nachwuchsgruppen, die in vielerlei Hinsicht zur Entstehung dieser Arbeit beigetragen haben. Bei Johannes P. Schlöder, Holger Diedam, Martin Felis, Khai-Long Ho Hoang und Julia Unger möchte ich mich herzlich für Unterstützung beim Korrekturlesen bedanken.

Mein besonderer und persönlicher Dank gilt Holger Diedam, sowie meinen Eltern Maria-Luise Hatz und Bruno Hatz, die mich in allen Phasen der Entstehung dieser Arbeit mit jedmöglicher Unterstützung bedacht haben.

Part I.

Theoretical foundations

Chapter 2.

Optimization theory

This chapter briefly reviews the basics of nonlinear programming used in this thesis. After stating basic definitions, we recall the sequential quadratic programming method for solving nonlinear programs, and the Generalized Gauß-Newton approach for constrained least-squares problems. The final section of this chapter is concerned with a special class of nonlinear programs called mathematical programs with complementarity constraints, and recalls basic definitions and challenges of this problem class. In this chapter, we mainly follow the descriptions in [NW99], [Hat08] and [FLRS06].

2.1. Basic definitions

We start with the definition of an equality and inequality constrained nonlinear program (NLP). An NLP is defined as an optimization problem of the following form:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & g(x) = 0 \\ & h(x) \geq 0, \end{aligned} \tag{2.1}$$

where $f : \mathbb{R}^{n_x} \to \mathbb{R}$, $g_i : \mathbb{R}^{n_x} \to \mathbb{R}$ with $i \in \mathcal{E} := \{1, \ldots, n_g\}$, and $h_i : \mathbb{R}^{n_x} \to \mathbb{R}$ with $i \in \mathcal{I} := \{1, \ldots, n_h\}$. All functions in (2.1) are assumed to be sufficiently smooth. The feasible set of (2.1) is denoted by

$$\mathcal{F}(x) := \{x \in \mathbb{R}^{n_x} \mid g(x) = 0 \ \wedge \ h(x) \geq 0\}, \tag{2.2}$$

and the active set of the inequality constraints is given by

$$\mathcal{A}(x) := \{i \in \{1, \ldots, n_h\} \mid h_i(x) = 0\}. \tag{2.3}$$

An inequality constraint $h_i(x)$, $i \in \mathcal{I}$, is called active, if $h_i(x) = 0$. Furthermore, the constraint $h_i(x)$ is said to be satisfied if $h_i(x) \geq 0$, and strictly satisfied if $h_i(x) > 0$ with $i \in \{1, \ldots, n_h\}$.

Definition 2.1.1. (Local/Global solution) *A point $x^* \in \mathcal{F}(x^*)$ is a local solution (or a local minimizer) of* (2.1)*, if and only if there exists an open ball $B_\epsilon(x^*) \subseteq \mathbb{R}^{n_x}$ with $\epsilon > 0$, such that*

$$\forall x \in B_\epsilon(x^*) \cap \mathcal{F}(x^*) : f(x^*) \leq f(x). \tag{2.4}$$

The point x^* is said to be a strict local solution if the inequality is strictly fulfilled for all $x \neq x^*$:

$$\forall x \in B_\epsilon(x^*) \cap \mathcal{F}(x^*) : f(x^*) < f(x). \tag{2.5}$$

If solutions exist, the ones with the smallest objective function value of all local solutions are referred to as global solutions (or global minimizers) of (2.1)*.*

In order to continue with optimality conditions for NLPs, we first need to define a constraint qualification (CQ) and the Lagrangian of an NLP.

Definition 2.1.2. (Linear independence constraint qualification (LICQ)) *Given the point x and the active set $\mathcal{A}(x)$, LICQ is said to hold at x if the active constraint gradients $\nabla_x g_i(x)$, $i \in \mathcal{E}$, $\nabla_x h_i(x)$, $i \in \mathcal{A}(x)$, are linearly independent.*

The following definition explains a second CQ.

Definition 2.1.3. (Mangasarian Fromowitz CQ (MFCQ)) *Given the point x and the active set $\mathcal{A}(x)$, MFCQ is said to hold at x if the gradients $\nabla_x g_i(x)$, $i \in \mathcal{E}$ are linearly independent and there exists a vector $d \in \mathbb{R}^{n_x}$ such that*

$$\begin{aligned} \nabla_x h_i(x) d &> 0 \quad \forall i \in \mathcal{A}(x) \\ \nabla_x g_i(x) d &= 0 \quad \forall i \in \mathcal{E}. \end{aligned} \tag{2.6}$$

Note that the following relation between LICQ and MFCQ holds [Fle05]:

$$\text{LICQ} \Rightarrow \text{MFCQ}. \tag{2.7}$$

Definition 2.1.4. (Lagrangian) *The Lagrangian $\mathcal{L}$ of the NLP* (2.1) *is defined as*

$$\mathcal{L}(x, \lambda, \mu) := f(x) - g(x)^\intercal \lambda - h(x)^\intercal \mu. \tag{2.8}$$

with Lagrange multipliers $\lambda \in \mathbb{R}^{n_g}$ and $\mu \in \mathbb{R}^{n_h}$.

The gradient of the Lagrangian with respect to the unknown x is given by

$$\nabla_x \mathcal{L}(x, \lambda, \mu) = \nabla_x f(x) - \nabla_x g(x)^\intercal \lambda - \nabla_x h(x)^\intercal \mu. \tag{2.9}$$

We can now recall the following theorem (cf., e.g., [NW99]) providing the well-known Karush-Kuhn-Tucker conditions.

Theorem 2.1.1. (Karush-Kuhn-Tucker conditions) *Suppose that x^* is a local solution of* (2.1) *and that LICQ holds at x^*. Then Lagrange multipliers $\lambda^* \in \mathbb{R}^{n_g}$ and $\mu^* \in \mathbb{R}^{n_h}$ exist, such that the following conditions are satisfied*

$$\nabla_x \mathcal{L}(x^*, \lambda^*, \mu^*) = 0 \tag{2.10a}$$

$$\mu^* \geq 0 \tag{2.10b}$$

$$g(x^*) = 0 \tag{2.10c}$$

$$h(x^*) \geq 0 \tag{2.10d}$$

$$\mu^{*\intercal} h(x^*) = 0. \tag{2.10e}$$

Conditions (2.10c) and (2.10d) are called primal feasibility, (2.10a) and (2.10b) are referred to as dual feasibility and (2.10e) is called complementarity. The KKT conditions are also referred to as *first-order optimality conditions.*

Definition 2.1.5. (Strict complementarity) *Given a local solution x^* of* (2.1) *and a multiplier μ^* satisfying* (2.10)*, the strict complementarity condition holds if exactly one of μ_i^* and $h_i(x^*)$ is zero for each $i \in \mathcal{I}$. This means that we have $\mu_i^* > 0$ for each $i \in \mathcal{A}(x^*)$.*

Definition 2.1.6. (KKT point/Stationary point) *A point $x^* \in \mathcal{F}(x^*)$ is said to be a KKT point/stationary point of the problem* (2.1) *if x^* satisfies first-order necessary optimality conditions, which means that LICQ is satisfied at x^* and x^* fulfills the KKT conditions* (2.10)*.*

To conclude the brief overview of the theory of nonlinear programming, we state second-order sufficient conditions.

Theorem 2.1.2. (Second-order sufficient condition) *Suppose that for some $x^* \in \mathcal{F}(x^*)$, there are Lagrange multipliers λ^* and μ^* such that the KKT conditions* (2.10) *are satisfied. Suppose also that all nonzero directions $p \in \mathbb{R}^{n_x}$ satisfy*

$$\begin{array}{rcll} \nabla_x g(x^*)\,p & = & 0 & \\ \nabla_x h_i(x^*)\,p & = & 0 & \forall i \in \mathcal{A}(x^*) \text{ with } \mu_i^* > 0 \\ \nabla_x h_i(x^*)\,p & \geq & 0 & \forall i \in \mathcal{A}(x^*) \text{ with } \mu_i^* = 0, \end{array} \tag{2.11}$$

and

$$p^\intercal \nabla^2_{xx} \mathcal{L}(x^*, \lambda^*, \mu^*) p > 0. \tag{2.12}$$

Then x^* is a strict local solution for (2.1)*.*

We refer to [Ber03] for a detailed motivation and for the proof of Theorem 2.1.2.

2.2. Algorithms for nonlinear programming

This section briefly reviews sequential quadratic programming and the Generalized Gauß-Newton method. We start with a brief sketch of a basic sequential quadratic programming method. Details can be found in, e.g., [NW99, Ber03].

2.2.1. Sequential quadratic programming

Sequential quadratic programming (SQP) methods are commonly used for solving NLPs. The main idea of an SQP method is to solve a sequence of quadratic programs (QPs) instead of the NLP (2.1) directly. We first consider an equality constrained NLP of the following form

$$\begin{array}{ll} \underset{x}{\text{minimize}} & f(x) \\ \text{subject to} & g(x) = 0, \end{array} \tag{2.13}$$

with f and g as defined in (2.1). A KKT point of (2.13) is characterized by the following $n_x + n_g$ equations

$$\begin{bmatrix} \nabla_x \mathcal{L}(x, \lambda) \\ g(x) \end{bmatrix} = 0, \tag{2.14}$$

with $\nabla_x \mathcal{L}(x, \lambda) = \nabla_x f(x) - \nabla_x g(x)^\intercal \lambda$, and unknowns x and λ. To solve (2.14), we use Newton's method as, e.g., described in [NW99]. Therefore, we need the derivative of the left-hand side of (2.14) with respect to x and λ, which is given by

$$\begin{bmatrix} \nabla^2_{xx} \mathcal{L}(x, \lambda) & -\nabla_x g(x)^\intercal \\ \nabla_x g(x) & 0 \end{bmatrix}. \tag{2.15}$$

Newton's method is an iterative method, where in each iteration, we obtain an update (x_{k+1}, λ_{k+1}) of our current iterate (x_k, λ_k) with the relation:

$$\begin{bmatrix} x_{k+1} \\ \lambda_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ \lambda_k \end{bmatrix} + \begin{bmatrix} p_k \\ p_\lambda \end{bmatrix}, \tag{2.16}$$

where p_k and p_λ solve the following system of equations:

$$\begin{bmatrix} \nabla^2_{xx}\mathcal{L}(x_k, \lambda_k) & -\nabla_x g(x_k)^\top \\ \nabla_x g(x_k) & 0 \end{bmatrix} \begin{bmatrix} p_k \\ p_\lambda \end{bmatrix} = \begin{bmatrix} -\nabla_x f(x_k) + \nabla_x g(x_k)^\top \lambda_k \\ -g(x_k) \end{bmatrix}. \tag{2.17}$$

If the so-called *KKT matrix* (2.15) has full rank, the solution of (2.17) is well-defined.

There is another way to look at the steps (p_k, p_λ) in (2.16), namely as the solution of a QP of the following form:

$$\begin{aligned} \underset{p_k}{\text{minimize}} \quad & \tfrac{1}{2}\, p_k^\top \nabla^2_{xx}\mathcal{L}(x_k, \lambda_k) p_k + \nabla_x f(x_k)^\top p_k \\ \text{subject to} \quad & \nabla_x g(x_k) p_k + g(x_k) = 0. \end{aligned} \tag{2.18}$$

Problem (2.18) has a unique solution (p_k, ν_k) ($\nu_k \in \mathbb{R}^{n_g}$ are the Lagrange multipliers) under the assumption that $\nabla_x g(x_k)$ has full rank and the Hessian approximation of $\nabla_{xx}\mathcal{L}(x_k, \lambda_k)$ is positive definite on the null space of the constraint gradients $\nabla_x g(x_k)$. (p_k, ν_k) then satisfies the KKT conditions of (2.18):

$$\begin{aligned} \nabla^2_{xx}\mathcal{L}(x_k, \mu_k) p_k + \nabla_x f(x_k) - \nabla_x g(x_k)^\top \nu_k &= 0, \\ \nabla_x g(x_k) p_k + g(x_k) &= 0. \end{aligned} \tag{2.19}$$

Identifying ν_k with λ_{k+1} in (2.16) reveals the equivalence of the SQP method and Newton's method. For solving an equality constrained NLP (2.13) with an SQP method, the step in each iteration can either be computed by solving (2.17) or by solving (2.18) if the KKT matrix (2.15) is nonsingular. The Newton's method point of view is primarily used for analyzing convergence properties, whereas the QP point of view helps to derive practical methods. A sketch of a local SQP algorithm in a basic form is given in Algorithm 1.

Algorithm 1: Local SQP algorithm.

```
Choose an initial pair $(x_0, \lambda_0)$;
for $k = 0, 1, \ldots$ do
    Evaluate $f(x_k)$, $\nabla_x f(x_k)$,$\nabla^2_{xx}\mathcal{L}(x_k, \lambda_k)$, $g(x_k)$, and $\nabla_x g(x_k)$;
    Solve (2.18) to obtain $p_k$ and $\nu_k$;
    $x_{k+1} \leftarrow x_k + p_k$; $\lambda_{k+1} \leftarrow \nu_k$;
    if convergence test satisfied then
        STOP with approximated solution $(x_{k+1}, \lambda_{k+1})$;
    end
end
```

Sticking to the QP point view allows us to generalize the method outlined in Algorithm 1 to equality and inequality constrained NLPs by solving the following equality and inequality constrained QP instead of (2.18) in each iteration:

$$\begin{aligned} \underset{p_k}{\text{minimize}} \quad & \tfrac{1}{2}\, p_k^\top \nabla^2_{xx}\mathcal{L}(x_k, \lambda_k, \mu_k) p_k + \nabla_x f(x_k)^\top p_k \\ \text{subject to} \quad & \nabla_x g(x_k) p_k + g(x_k) = 0 \\ & \nabla_x h(x_k) p_k + h(x_k) \geq 0. \end{aligned} \tag{2.20}$$

This generalization is based on the following Theorem stated in, e.g., [NW99].

Theorem 2.2.1. *Suppose that x^* is a solution of* (2.1)*. Assume that the Jacobian of active constraints $\left[\nabla_x g(x^*)^\intercal,\ \nabla_x h_{\mathcal{A}}(x^*)^\intercal\right]$ (where $\nabla_x h_{\mathcal{A}}(x^*)$ only contains gradients of active components of $h(x^*)$) has full rank, that $p^\intercal \nabla_{xx}\mathcal{L}^*(x^*, \lambda^*, \mu^*)p > 0$ for all $p \neq 0$ such that $\nabla_x h_{\mathcal{A}}(x^*)p = 0$, and that strict complementarity holds. Then if (x_k, λ_k, μ_k) is sufficiently close to (x^*, λ^*, μ^*), there is a local solution of the subproblem* (2.20) *whose active set is the same as the active set of the NLP* (2.1) *at x^*.*

An SQP method that solves problem (2.20) in each iteration requires a QP solver that is able to treat equality and inequality constraints. We note that an active set strategy could as well be implemented on NLP level requiring the solution of an equality constrained QP in each iteration. Details about practical SQP methods and their convergence behavior can be found in, e.g., [Ber03, Fle87, NW99].

2.2.2. The Generalized Gauß-Newton method

The well-known Gauß-Newton method is used for solving nonlinear least-squares problems as described in [NW99] or [Ber03]. Bock introduces a generalization of the classical method in [Boc81a]. The so-called *Generalized Gauß-Newton (GGN) method* solves equality and inequality constrained nonlinear least-squares problems. In this section, we briefly review the GGN method described in [Boc81a]. Like the SQP method explained above, the GGN is closely related to Newton's method. This relation will be highlighted in the following.

We consider an equality constrained nonlinear least-squares problem:

$$\begin{array}{cl} \underset{x}{\text{minimize}} & \frac{1}{2}\|r(x)\|_2^2 \\ \text{subject to} & g(x) = 0 \end{array} \tag{2.21}$$

with the residual function $r : \mathbb{R}^{n_x} \to \mathbb{R}^m$ and $g_i : \mathbb{R}^{n_x} \to \mathbb{R}$ with $i = 1, \dots, n_g$. The functions r and g are assumed to be sufficiently smooth. The derivation of the GGN method is closely related to the one of the SQP method described in the last section. We formulate KKT conditions of (2.21) and apply Newton's method. The KKT conditions of (2.21) are given by

$$\begin{bmatrix} \nabla_x r(x)^\intercal r(x) - \nabla_x g(x)^\intercal \lambda \\ g(x) \end{bmatrix} = 0. \tag{2.22}$$

Applying Newton's method to (2.22) leads to the following iteration:

$$\begin{bmatrix} x_{k+1} \\ \lambda_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ \lambda_k \end{bmatrix} + \begin{bmatrix} p_k \\ p_\lambda \end{bmatrix}, \tag{2.23}$$

where p_k and p_λ solve the following system of equations:

$$\begin{bmatrix} \nabla_{xx}\mathcal{L}(x,\lambda) & -\nabla_x g(x_k)^\intercal \\ \nabla_x g(x) & 0 \end{bmatrix} \begin{bmatrix} p_k \\ p_\lambda \end{bmatrix} = \begin{bmatrix} -\nabla_x r(x_k)^\intercal r(x_k) + \nabla_x g(x_k)^\intercal \lambda_k \\ -g(x_k) \end{bmatrix}. \tag{2.24}$$

The gradient and the Hessian of the Lagrangian $\mathcal{L}(x, \lambda)$ are given by

$$\nabla_x \mathcal{L}(x,\lambda) = \nabla_x r(x)^\intercal r(x) - \nabla_x g(x)^\intercal \lambda \tag{2.25}$$

and

$$\begin{aligned}
[\nabla_{xx}\mathcal{L}(x,\lambda)]_{jl} &= \left[\sum_{i=1}^{m}\frac{\partial^2 r_i(x)}{\partial x_j \partial x_l} r_i(x) + \sum_{i=1}^{m}\frac{\partial r_i(x)}{\partial x_j}\frac{\partial r_i(x)}{\partial x_l}\right]_{jl} - \left[\sum_{i=1}^{m}\frac{\partial^2 g_i(x)}{x_j x_l}\lambda_i\right]_{jl} && (2.26)\\
&\approx \left[\nabla_x r(x)^{\intercal}\nabla_x r(x)\right]_{jl}, && (2.27)
\end{aligned}$$

where $j, l = 1, \ldots, n_x$. The following arguments are used to explain the approximation of the Hessian of the Lagrangian in (2.27) (see [Boc87]):

- The residuals $r(x)$ are expected to be small close to the solution. Hence, the first term in the right-hand side of (2.26) can be neglected.
- The gradient of the Lagrangian (2.25) is expected to become small close to the solution. Additionally, we know that the residuals $r_i(x)$ are small close to the solution. Combining this with the previous argument implies that the Lagrange multiplier λ can be expected to be small close to the solution, which is the reason for neglecting the third term in the right-hand side of (2.26).

Using approximation (2.27) in (2.24) leads to the following system of equations:

$$\begin{bmatrix} \nabla_x r(x)^{\intercal}\nabla_x r(x) & -\nabla_x g(x_k)^{\intercal} \\ \nabla_x g(x) & 0 \end{bmatrix} \begin{bmatrix} p_k \\ p_\lambda \end{bmatrix} = \begin{bmatrix} -\nabla_x r(x_k)^{\intercal} r(x_k) + \nabla_x g(x_k)^{\intercal}\,\lambda_k \\ -g(x_k) \end{bmatrix}. \quad (2.28)$$

The solution (p_k, p_λ) of (2.28) can also be seen as the solution of the following QP, which approximates the nonlinear least-squares problem (2.21):

$$\begin{aligned}
&\underset{p_k}{\text{minimize}} && \tfrac{1}{2}\|r(x) + \nabla_x r(x) p_k\|_2^2 && (2.29a)\\
&\text{subject to} && g(x) + \nabla_x g(x) p_k = 0. && (2.29b)
\end{aligned}$$

If $\nabla_x g(x_k)$ has full rank and the Hessian approximation $\nabla_x r(x)^{\intercal}\nabla_x r(x)$ is positive definite on the null space of the constraint gradients $\nabla_x g(x)$, problem (2.29) has a unique solution (p_k, ν_k), where $\nu_k \in \mathbb{R}^{n_g}$ are the Lagrange multipliers. The step (p_k, ν_k) then satisfies the KKT conditions of (2.29):

$$\begin{aligned}
\nabla_x r(x)^{\intercal} r(x) + \nabla_x r(x)^{\intercal}\nabla_x r(x) p_k - \nabla_x g(x)^{\intercal}\nu_k &= 0\\
g(x) + \nabla_x g(x) p_k &= 0.
\end{aligned}$$

Identifying ν_k with λ_{k+1} (in $p_\lambda = \lambda_{k+1} - \lambda_k$) in (2.28) shows the relationship between the GGN method and Newton's method. For solving an equality constrained nonlinear least-squares problem (2.21) with an GGN method, the step in each iteration can either be computed by solving (2.28) or by solving (2.29), if the KKT matrix in the left-hand side of (2.28) is regular. The following theorem completes the derivation of the GGN method.

Theorem 2.2.2. (Existence of a generalized inverse) *We assume that a CQ holds and that the Hessian approximation*

$$\nabla_x r(x)^{\intercal}\nabla_x r(x) \quad (2.30)$$

is positive definite on the null space of $\nabla_x g(x)$. *Then if* (x_k, λ_k) *is sufficiently close to the solution* (x^*, λ^*) *of* (2.21), *we have that*

1. *there is exactly one point* (x^*, λ^*) *satisfying the KKT conditions. Furthermore,* (x^*, λ^*) *is a strict local minimizer of* (2.29).

2. *The solution is given by*

$$x^* = -J^+(x) \begin{bmatrix} r(x) \\ g(x) \end{bmatrix} \tag{2.31}$$

or

$$\begin{bmatrix} x^* \\ \lambda^* \end{bmatrix} = -\tilde{J}^+(x) \begin{bmatrix} r(x) \\ g(x) \end{bmatrix} \tag{2.32}$$

with linear mappings

$$\begin{aligned} J^+ \quad &: \quad \mathbb{R}^{m+n_g} \to \mathbb{R}^{n_x} && (2.33) \\ \tilde{J}^+ \quad &: \quad \mathbb{R}^{m+n_g} \to \mathbb{R}^{n_x+n_g}, && (2.34) \end{aligned}$$

where

$$\tilde{J}^+(x) \quad := \quad \begin{bmatrix} \nabla_x r(x)^\intercal \nabla_x r(x) & -\nabla_x g(x_k)^\intercal \\ \nabla_x g(x) & 0 \end{bmatrix}^{-1} \begin{bmatrix} \nabla_x r(x)^\intercal & 0 \\ 0 & \mathbb{I} \end{bmatrix}, \tag{2.35}$$

and

$$J^+(x) \quad := \quad [\, \mathbb{I} \;\; 0 \,] \; \tilde{J}^+(x). \tag{2.36}$$

3. *The generalized inverse* J^+ *fulfills*

$$J^+(x) \begin{bmatrix} \nabla_x r(x) \\ \nabla_x g(x) \end{bmatrix} J^+(x) = J^+(x). \tag{2.37}$$

For the final generalization to the inequality constrained case, we consider the following equality and inequality constrained nonlinear least-squares problem:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & \frac{1}{2}\|r(x)\|_2^2 \\ \text{subject to} & g(x) = 0 \\ & h(x) \geq 0, \end{array} \tag{2.38}$$

where $r : \mathbb{R}^{n_x} \to \mathbb{R}^m$, and $g_i, h_j : \mathbb{R}^{n_x} \to \mathbb{R}$ with $i = 1, \ldots, n_g$ and $j = 1, \ldots, n_h$ are assumed to be sufficiently smooth. If we consider problem (2.38) in a small neighborhood of the solution, the active set is constant (see Theorem 2.2.1) and therewith it is sufficient to consider problem (2.21) as a local approximation. Thus, we can solve (2.38) by using the GGN method.

A proof of linear convergence as well as effective globalization strategies and further details can be found in [Boc87].

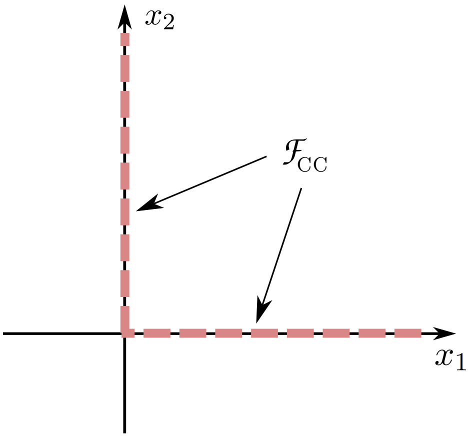


Figure 2.1.: Illustration of the set $\mathcal{F}_{\mathrm{CC}}(x_1, x_2) := \{(x_1, x_2) \in \mathbb{R}^{2q} \mid 0 \leq x_1 \perp x_2 \geq 0\}$ for $q = 1$.

2.3. Mathematical programs with complementarity constraints

In this section, we briefly describe the problem formulation and the challenges of mathematical programs with complementarity constraints (MPCCs). We furthermore discuss stationarity concepts and solution strategies. The discussion about numerical methods for solving MPCCs is continued in Chapter 8.

MPCCs are NLPs with a special and challenging structure. In this section, we consider MPCCs of the following form:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & g(x) = 0 \\ & h(x) \geq 0 \\ & 0 \leq x_1 \perp x_2 \geq 0, \end{aligned} \tag{2.39}$$

where $x = (x_0, x_1, x_2) \in \mathbb{R}^{p+q+q}$ is a decomposition of the problem variables, $f : \mathbb{R}^{p+q+q} \to \mathbb{R}$, $g : \mathbb{R}^{p+q+q} \to \mathbb{R}^{n_g}$ and $h : \mathbb{R}^{p+q+q} \to \mathbb{R}^{n_h}$, where f, g and h are assumed to be sufficiently smooth. The last constraint in (2.39) is $0 \leq x_1 \perp x_2 \geq 0$, which is the so-called *complementarity constraint (CC)*, requiring x_1 and x_2 to be component-wisely nonnegative. The notation $\perp$ requires that for each component $i = 1, \dots, q$, either x_{1i} or x_{2i} has to be equal to zero. The feasible set

$$\mathcal{F}_{\mathrm{CC}}(x_1, x_2) := \{(x_1, x_2) \in \mathbb{R}^{2q} \mid 0 \leq x_1 \perp x_2 \geq 0\} \tag{2.40}$$

of the complementarity constraint is illustrated in Figure 2.1 for $q = 1$. The specialty and the challenge of an MPCC compared to a standard NLP like problem (2.1) is the lack of certain CQ (like LICQ stated in Definition 2.1.2) at any feasible point, which is due to the combinatorial nature of the feasible set. The lack of LICQ can easily be shown for a simple

MPCC, where the only constraint is the CC:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & f(x) \\ \text{subject to} & 0 \leq x_1 \perp x_2 \geq 0. \end{array} \tag{2.41}$$

A mathematically equivalent formulation of (2.41) is given by:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & f(x) \\ \text{subject to} & x_1 \geq 0 \\ & x_2 \geq 0 \\ & x_1^\intercal x_2 = 0, \end{array} \tag{2.42}$$

which we refer to as NLP formulation of (2.41). The constraint normals of (2.42) are

$$\begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix} \text{ and } \begin{pmatrix} x_2 \\ x_1 \end{pmatrix}. \tag{2.43}$$

Clearly, the constraint normals in (2.43) are linearly dependent, independent of the value of x_1 and x_2, which means that LICQ is violated at any feasible point. In Part III and IV of this thesis, we also consider a more general type of CC given by

$$0 \leq G(x) \perp H(x) \geq 0. \tag{2.44}$$

However, the constraint (2.44) can be rewritten in the form of a standard CC as the one in (2.39) by introducing slack variables. Therefore, we stick to formulation (2.39) for the remainder of this section.

The numerical challenges that appear when solving MPCCs are manifold. The lack of LICQ might lead to an unbounded set of Lagrange multipliers, which means that the Lagrange multipliers of (2.39) are not uniquely determined anymore. This is explained and discussed by means of a well-chosen example in [FLRS06]. Another problem that might appear when solving (2.39) with an SQP algorithm is that the approximation of (2.39) by a QP might be inconsistent arbitrarily close to the solution (cf. [FLRS06]). Before we discuss how to deal with these challenges within a numerical algorithm, we define a CQ and stationarity concepts tailored to MPCCs. Note that Definition 2.1.6 of a stationary point cannot be used anymore since LICQ, as stated in Definition 2.1.2, is not satisfied for (2.39).

We now introduce a CQ that is tailored to MPCCs following [Sch01, FLRS06]. Therefore, we consider the two index sets $\mathcal{Z}_1$ and $\mathcal{Z}_2$ with $\mathcal{Z}_1, \mathcal{Z}_2 \subset \{1, \ldots, q\}$. The respective complement of the sets $\mathcal{Z}_1$ and $\mathcal{Z}_2$ in $\{1, \ldots, q\}$ is denoted by $\mathcal{Z}_1^\perp$ and $\mathcal{Z}_2^\perp$. The definition of an LICQ tailored to MPCCs centers around the following relaxed NLP:

$$\begin{array}{llll} \underset{x}{\text{minimize}} & f(x) & & \\ \text{subject to} & g(x) & = 0 & \\ & h(x) & \geq 0 & \\ & x_{1i} & = 0 & \forall i \in \mathcal{Z}_1^\perp \\ & x_{2i} & = 0 & \forall i \in \mathcal{Z}_2^\perp \\ & x_{1i} & \geq 0 & \forall i \in \mathcal{Z}_1 \\ & x_{2i} & \geq 0 & \forall i \in \mathcal{Z}_2. \end{array} \tag{2.45}$$

Note that if x^* is a solution of (2.45) and ${x_1^*}^\intercal x_2^* = 0$, then x^* is a solution of the original non-relaxed problem (2.39). The set of second-level degenerate components is defined as $\mathcal{D} := \mathcal{Z}_1 \cap \mathcal{Z}_2$.

Definition 2.3.1. (MPCC-LICQ) *Let $x_1, x_2 \geq 0$, $h(x) = 0$, $g(x) \geq 0$ and define index sets*

$$\mathcal{Z}_1 = \{i \in \{1, \ldots, q\} \mid x_{1i} = 0\} \quad \text{and} \quad \mathcal{Z}_2 = \{i \in \{1, \ldots, q\} \mid x_{2i} = 0\}. \tag{2.46}$$

The MPCC (2.39) *is said to satisfy* MPCC-LICQ *at x^* if the corresponding relaxed NLP* (2.45) *satisfies LICQ at x^*.*

In the last years, numerous specifically tailored stationarity concepts for MPCCs have been proposed. [LM07] or [Ye05] provide an overview of these stationary concepts. In this thesis, we focus on two concept: Bouligand-stationarity (or B-stationarity) and strong stationarity, which are defined in the following.

Definition 2.3.2. (B-stationarity) *A point x^* is called Bouligand stationary or B-stationary if $d = 0$ solves the following linear program with CC:*

$$\begin{array}{ll} \underset{d}{\textit{minimize}} & \nabla_x f(x^*)^\intercal d \\ \textit{subject to} & g(x^*) + \nabla_x g(x^*) d = 0 \\ & h(x^*) + \nabla_x h(x^*) d \geq 0 \\ & 0 \leq \ (x_1^* + d_{x_1}) \ \perp \ (x_2^* + d_{x_2}) \ \geq 0, \end{array} \tag{2.47}$$

where $d = (d_{x_0}, d_{x_1}, d_{x_2})$ is a partition of the step corresponding to the partition $x = (x_0, x_1, x_2)$ of variables.

For defining strong stationarity, we consider the so-called *NLP formulation* of (2.39):

$$\begin{array}{ll} \underset{x}{\text{minimize}} & f(x) \\ \text{subject to} & g(x) = 0 \\ & h(x) \geq 0 \\ & x_1 \geq 0 \\ & x_2 \geq 0 \\ & x_1^\intercal x_2 = 0. \end{array} \tag{2.48}$$

Definition 2.3.3. (Strong stationarity) *A point x^* is said to be a strongly stationary point of* (2.39) *if there exist multipliers $\lambda \in \mathbb{R}^{n_g}, \mu \in \mathbb{R}^{n_h}, \hat{\nu}_1 \in \mathbb{R}^q$ and $\hat{\nu}_2 \in \mathbb{R}^q$ such that*

$$\begin{array}{rcll} \nabla_x f(x^*) - \nabla_x g(x^*)^\intercal \ \lambda - \nabla_x h(x^*)^\intercal \ \mu - \begin{pmatrix} 0 \\ \hat{\nu}_1 \\ \hat{\nu}_2 \end{pmatrix} & = & 0, & \\ g(x^*) & = & 0, & \\ h(x^*) & \geq & 0, & \\ x_1^* & \geq & 0, & \\ x_2^* & \geq & 0, & \\ x_{1j}^* = 0 \quad \text{or} \quad x_{2j}^* & = & 0 & \forall\, j = 1, \cdots, q, \\ \mu & \geq & 0, & \\ h(x^*)^\intercal \mu & = & 0, & \\ x_{1j}^* \hat{\nu}_{1j} & = & 0 & \forall\, j = 1, \cdots, q, \\ x_{2j}^* \hat{\nu}_{2j} & = & 0 & \forall\, j = 1, \cdots, q, \\ \text{if} \quad x_{1j}^* = x_{2j}^* = 0, \quad \text{then} \quad \hat{\nu}_{1j} \geq 0 \quad \text{and} \quad \hat{\nu}_{2j} & \geq & 0 & \forall\, j = 1, \cdots, q. \end{array} \tag{2.49}$$

The equivalence of strong stationarity of an MPCC (2.39) and NLP stationarity (Definition 2.1.6) of the NLP formulation (2.48) of (2.39) is shown in [FLRS06]. Scheel and Scholtes have shown in [SS00] that strong stationarity is implied by B-stationarity and if MPCC-LICQ holds, the reverse is true.

In literature, one finds a variety of algorithms to solve problem (2.39). The main categories are mentioned in the following. Algorithms based on branch-and-bound techniques can be found in, e.g., [Bar88]. Interior-point algorithms for MPCCs are described in, e.g., [LLCN06] or [RB03]. Piecewise SQP techniques are presented in [LPR97], and [DFNS05, HKS13, LF03, HLSB13], e.g., are concerned with penalization and relaxation techniques for solving (2.39). Numerical experience with solving MPCCs as described in [FL02b, FLRS06] clearly shows the promise of applying SQP methods to MPCCs. Furthermore, the SQP approach is in general computationally cheaper than most of the techniques named above. Hence, in this thesis we concentrate on a tailored SQP technique for solving problems of type (2.39) and continue the discussion of numerical methods for MPCCs in Chapter 8.

Chapter 3.

Optimization of dynamic systems

In this chapter, we describe the basic concept of two classes of dynamic optimization problems: the class of optimal control problems and the class of parameter estimation problems. Based on the optimization methods for finite-dimensional nonlinear programs described in the last chapter, we now generalize the problem setting to dynamic systems described by ordinary differential equations. In the first section, we consider boundary value problems and discuss their properties and their numerical treatment. The two remaining sections are concerned with optimal control problems and parameter estimation problems. In each of the two sections, we start with the basic formulation of the respective problem class, continue with a discussion of the problem's properties, and close with a description of numerical methods to solve the respective problem.

3.1. Boundary value problems and their numerical solution

A two-point boundary value problem (BVP) defined as follows

$$\begin{aligned} \dot{x}(t) &= f(t, x(t)), \quad t \in I & (3.1a) \\ r(x(t_0), x(T)) &= 0, & (3.1b) \end{aligned}$$

includes a system of ordinary differential equations (ODEs) with the ODE's right-hand side f mapping to $\mathbb{R}^{n_x}$, time $t \in I = [t_0, T] \subseteq \mathbb{R}$, differential states x mapping to $\mathbb{R}^{n_x}$, and the two-point boundary conditions r mapping to $\mathbb{R}^{n_r}$. The function f is assumed to be piecewise Lipschitz continuous (details can be found in, e.g., [Har82, HNW00]). A BVP of type (3.1) is often solved with a single shooting method [HR71, SS78], which is a comparatively simple approach for solving BVPs that is relatively easy to implement. The main idea is to determine the initial value $x(t_0)$ in an iterative procedure, such that the resulting trajectory, which is obtained by numerical integration, solves (3.1). One of the main disadvantage of the single shooting method is, however, that for certain initial values, the solution of (3.1a) might not even exist on the whole time interval I. The collocation method remedies this problem (cf., e.g., [RS72]). In the collocation approach, the differential states are discretized on a fine grid and the ODE (3.1a) is replaced by a system of nonlinear equations. The system of equations plus the boundary condition (3.1b) are then solved with Newton's method (described in, e.g., [SB92]). In contrast to single shooting, collocation allows to exploit the knowledge of the process behavior by an appropriate initialization of the differential states. The nonlinear system of equations that has to be solved is typically large but very sparse. Hence, structure-exploiting Newton methods allow an efficient treatment, even of large problems of type (3.1). The main drawback of the collocation method is the difficulty of including adaptive discretization

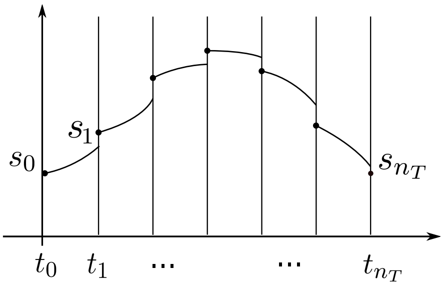


Figure 3.1.: Illustration of the parameterized solution of an ODE using multiple shooting.

methods, which is indispensable for solving highly nonlinear BVPs. To overcome both the drawback of the single shooting technique and the collocation approach, we use the multiple shooting method [Osb69, Bul71, Boc74, Boc87]. The basic idea is to divide the time horizon I into subintervals $[t_i, t_{i+1}]$ with $i = 0, \dots, n_T - 1$ and

$$t_0 < \ldots < t_i < t_{i+1} < \ldots < t_{n_T} = T, \tag{3.2}$$

where the subintervals not necessarily have to be of the same size, and to integrate the ODE (3.1a) on shorter time horizons. Therefore, we introduce new variables $s_0, \dots, s_{n_T}$, where $s_i \in \mathbb{R}^{n_x}$ describes the initial value of the solution of (3.1a) on time interval $[t_i, t_{i+1}]$. The parameterization of the solution of (3.1) is illustrated in Figure 3.1 for $n_x = 1$. On each interval $[t_i, t_{i+1}]$, the following initial value problem (IVP) is solved

$$\begin{array}{rcll} \dot{x}(t) & = & f(t, x(t)), & t \in [t_i, t_{i+1}] \\ x(t_i) & = & s_i, & \end{array} \tag{3.3}$$

with $i = 0, \dots, n_T - 1$. We note that the IVPs on the multiple shooting intervals $[t_i, t_{i+1}]$ are decoupled and can be solved independently. We denote the solution of (3.3) by $x(t; t_i, s_i)$, where the first argument states that x is evaluated at t, and the last two arguments indicate that the solution of (3.3) depends on t_i and s_i. In order to guarantee the solution of (3.1) to be continuous, we need the following n_T conditions:

$$x(t_{i+1}; t_i, s_i) - s_{i+1} = 0, \quad i = 0, \dots, n_T - 1, \tag{3.4}$$

which are called closing or matching conditions. The system of nonlinear matching conditions plus the boundary condition (3.1b) are then solved with a Newton-type method. The multiple shooting method inherits a special staircase structure in the Jacobian of the system of matching conditions (3.4), which allows us to solve large-scale problems by applying an efficient structure-exploiting Newton method (as described in, e.g., [Lei99]). The multiple shooting method combines the advantages of both the single shooting and the collocation method. Due to the shorter integration horizons, the IVP's stability improves but, in contrast to the collocation method, the integration on the intervals $[t_i, t_{i+1}]$ allows to easily incorporate adaptivity by using state-of-the-art integration methods (cf., e.g., [Alb10]).

An important issue when solving (3.1) with the multiple shooting method is the computation of derivatives. Applying Newton's method to a system including (3.4) requires derivatives of the solution $x(t_{i+1}; t_i, s_i)$ of (3.3) with respect to the initial value s_i. These derivatives are often referred to as *sensitivities*, describing the sensitivity of the trajectory with respect to the initial value, and can be obtained by solving the following variational differential equation:

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} G(t; t_i, s_i) &= \frac{\partial}{\partial x} f(t, x(t; t_i, s_i)) \cdot G(t; t_i, s_i), \quad t \in [t_i, t_{i+1}] && (3.5a) \\ G(t_i; t_i, s_i) &= \mathbb{I}_{n_x} && (3.5b) \end{aligned}$$

for $i \in \{1, \ldots, n_T - 1\}$. One can show that the solution of (3.5) is

$$G(t; t_i, s_i) = \frac{\mathrm{d}}{\mathrm{d}s_i} x(t; t_i, s_i). \tag{3.6}$$

Alternative ways for computing the sensitivities of (3.3) include a method based on perturbed trajectories (also called *external numerical differentiation* as the process of solving (3.3) is treated as black-box algorithm), which is a very intuitive approach applying the idea of finite differences to the solution trajectory of the IVP (3.3). However, doing that implicitly assumes that the procedure of computing an approximation of $x(t; t_i, s_i)$ is a sufficiently smooth mapping, which is not true in the majority of cases due to, e.g., adaptive components in the integration method or pivoting strategies in the linear algebra subroutines. In contrast to external numerical differentiation, [Boc81a, Boc83] state the concept of *internal numerical differentiation (IND)*, which remedies the lack of smoothness. The main idea of IND is to differentiate the algorithm computing the approximation of $x(t; t_i, s_i)$. In practice, this means that we have to ensure that all adaptive components in the computation of the approximation of the nominal trajectory $x(t; t_i, s_i)$ and of the perturbed trajectory $x(t; t_i, s_i + hd_i)$ with the direction $d_i \in \mathbb{R}^{n_x}$ and the perturbation $h \in \mathbb{R}$, are the same. That way, we obtain consistent derivatives of the approximation of $x(t; t_i, s_i)$ with respect to s_i. IND can also be realized when using the variational differential equation to compute the trajectory's sensitivity by, e.g., solving (3.3) and (3.5) simultaneously. For a further discussion of the principle of IND, we refer to [Boc87, Alb10].

3.2. Optimal control of dynamic systems

We consider optimal control problems (OCPs) with an objective function of Bolza-type, an underlying system of ODEs, mixed control-state constraints and multi-point boundary conditions given by

$$\begin{array}{lll} \underset{x,u,q}{\text{minimize}} & \Phi & := \phi_{\mathcal{M}}(x(T), q) + \int_{t_0}^{T} \phi_{\mathcal{L}}(t, x(t), u(t), q) \,\mathrm{d}t \\ \text{subject to} & \dot{x}(t) & = f(t, x(t), u(t), q), \qquad t \in I \\ & 0 & \leq c(t, x(t), u(t), q), \qquad t \in I \\ & 0 & = r^{\mathrm{eq}}(x(t_0), \ldots, x(T), q) \\ & 0 & \leq r^{\mathrm{ieq}}(x(t_0), \ldots, x(T), q), \end{array} \tag{3.7}$$

where $t \in I := [t_0, T] \subseteq \mathbb{R}$. The Bolza-type objective function consists of a Mayer term $\phi_{\mathcal{M}}$ and a Lagrange term $\int_{t_0}^{T} \phi_{\mathcal{L}}(\cdot) \,\mathrm{d}t$. The differential states are denoted by $x : I \to \mathbb{R}^{n_x}$,

$u : I \to \mathbb{R}^{n_u}$ is the control function, which is assumed to be measurable, and $q \in \mathbb{R}^{n_q}$ describes time-independent control values. f denotes the ODE's right-hand side and is assumed to be piecewise Lipschitz continuous. Mixed control-state constraints are denoted by c mapping to $\mathbb{R}^{n_c}$. The functions r^{eq} and r^{ieq} mapping to $\mathbb{R}^{n_{\text{eq}}}$ and $\mathbb{R}^{n_{\text{ieq}}}$, respectively, are multi-point equality and inequality constraints. In the following, we consider numerical methods for solving the OCP (3.7).

3.2.1. An indirect approach for optimal control

We consider an indirect approach for solving the OCP (3.7) based on Pontryagin's maximum principle [PBGM62]. The basic idea is to transform the OCP into a multi-point boundary value problem (MPBVP) by means of necessary conditions for optimality. The resulting MPBVP can then be solved with an appropriate numerical method. Details can be found in, e.g., [Neu76, Boc81b, HSV95]. In this thesis, we discuss Pontryagin's maximum principle for the following special case of (3.7) (see also [HSB12]):

$$\begin{array}{lrcll} \underset{x,u}{\text{minimize}} & \phi_{\mathcal{M}}(x(T)) & & & \\ \text{subject to} & \dot{x}(t) & = & f(x(t),u(t)), & t \in I \\ & 0 & = & r^{\text{eq}}(x(t_0),x(T)) & \\ & u(t) & \in & \mathcal{U} := \{u(t) \in \mathbb{R} \mid \bar{c}(u(t)) \geq 0\}, & \end{array} \tag{3.8}$$

with a scalar control and no path constraints. The function $\bar{c}(\cdot)$ describes the control constraints and maps to $\mathbb{R}^{n_{\bar{c}}}$. Furthermore, we assume

- the absence of a singular control, which means that the control $u(t)$ is uniquely determined by maximizing the Hamiltonian (defined below),
- the regularity condition $\nabla_u \bar{c}_i(u(t)) \neq 0$ for active components $i \in \{1,\dots,n_{\bar{c}}\}$ of $\bar{c}$,
- and the strict Legendre condition, which requires the second derivative of the augmented Hamiltonian (defined below) with respect to u to be negative definite on boundary arcs (where boundary arcs are subintervals of I with $\bar{c}_i(u(t)) = 0$ for at least one component $i \in \{1,\dots,n_{\bar{c}}\}$).

A discussion of Pontryagin's maximum principle for more general OCPs can be found in the references named above, or in, e.g., [Hes66]. We now formulate necessary conditions for optimality for problem (3.8) (cf. [HSB12]). Let $(x(t),u(t))$ be a solution of (3.8), then Lagrange multipliers $\alpha_0 \in \mathbb{R}$, $\alpha \in \mathbb{R}^{n_{\text{eq}}}$ and adjoint variables $\mu : I \to \mathbb{R}^{n_{\bar{c}}}$, $\lambda : I \to \mathbb{R}^{n_x}$ exist, such that with the Hamiltonian $H^0(x(t),\lambda(t),u(t)) := \lambda(t)^{\mathsf{T}} f(x(t),u(t))$, the augmented Hamiltonian

$$H(x(t),\lambda(t),u(t),\mu(t)) := H^0(x(t),\lambda(t),u(t)) - \mu(t)^{\mathsf{T}}\bar{c}(u(t)) \tag{3.9}$$

and the augmented objective function

$$\tilde{\Phi} := \alpha_0 \phi_{\mathcal{M}}(x(T)) + \alpha^{\mathsf{T}}(r^{\text{eq}}(x(t_0),x(T))) \tag{3.10}$$

the following necessary conditions are satisfied: the multiplier λ fulfills the adjoint differential equations

$$\dot{\lambda}(t) = -\nabla_x H(x(t),\lambda(t),u(t),\mu(t)) \tag{3.11}$$

on the time horizon I, the transversality conditions

$$\lambda(t_0) = \nabla_{x_0}\tilde{\Phi} \quad \text{and} \quad \lambda(T) = -\nabla_{x_T}\tilde{\Phi} \tag{3.12}$$

with $x_0 := x(t_0)$ and $x_T = x(T)$ are satisfied, and, on I, the maximum principle holds:

$$\begin{array}{lcl} \mu(t) & \geq & 0, \quad 0 = \mu(t)^{\mathsf{T}}\bar{c}(u(t)), \\ 0 & = & \nabla_u H(x(t), \lambda(t), u(t), \mu(t)). \end{array} \tag{3.13}$$

Note that we implicitly assume that the necessary optimality conditions are satisfied in normal form with $\alpha \neq 0$, choosing $\alpha = 1$. For a proof of Pontryagin's maximum principle, we refer to [BH75] or [PBGM62].

Instead of solving the OCP (3.8), it remains to solve the following BVP stating necessary conditions for optimality of (3.8):

$$\begin{array}{lcll} \dot{x}(t) & = & f(x(t), u(t)), & t \in I \\ \dot{\lambda}(t) & = & -\nabla_x H(x(t), \lambda(t), u(t), \mu(t)), & t \in I \\ 0 & = & r^{\text{eq}}(x(t_0), x(T)) & \\ \lambda(t_0) & = & \nabla_{x_0}\tilde{\Phi}, \quad \lambda(T) = -\nabla_{x_T}\tilde{\Phi} & \\ 0 & \leq & \bar{c}(u(t)) & \\ 0 & \leq & \mu(t), \quad 0 = \quad \mu(t)^{\mathsf{T}}\bar{c}(u(t)) & \\ 0 & = & \nabla_u H(x(t), \lambda(t), u(t), \mu(t)). & \end{array} \tag{3.14}$$

After preliminary work concerning the treatment of the inequality constraints, the BVP (3.14) can then be solved with, e.g., a multiple shooting method for solving multi-point BVPs based on the techniques for equality constrained BVPs described in Section 3.1.

In general, the indirect approach for solving OCPs allows to compute highly accurate solutions. This is due to the fact that, in contrast to direct approaches discussed in the next subsection, the control function $u(t)$ is not discretized and the OCP is optimized in the infinite-dimensional space. Furthermore, indirect methods are often used to investigate the structure of an OCP's solution. However, the main drawback of the indirect approach is that the derivation of necessary conditions for optimality and the corresponding BVP of an OCP of type (3.7) is highly intricate. For the general case (3.7) with multiple controls and path constraints, it is seldom possible to derive necessary conditions for optimality based on Pontryagin's maximum principle and the corresponding BVP. A further drawback is that solving an OCP with an indirect method can hardly be done in an automatic procedure. In most cases, deep insight into the structure of the OCP and its solution is needed to derive the OCP's necessary conditions for optimality in the function space. We continue this discussion in Chapter 6.

3.2.2. A direct multiple shooting method for optimal control

We now discuss a direct approach for solving OCP (3.7). The main idea is to approximate the infinite-dimensional control function $u(t)$ using finitely many variables, to parameterize the differential states based on multiple shooting and to discretize the mixed control-state constraints and the boundary conditions. The term *direct* stems from the fact that we, in contrast to the indirect approach presented in the last subsection, directly iterate on variables approximating the controls. The resulting nonlinear program (NLP) can then be solved with a tailored Newton-type method. Early works on the direct multiple shooting method for optimal control include [Pli81, BP84]. Further details and extensions can be found in, e.g., [Lei99, LBS$^+$03].

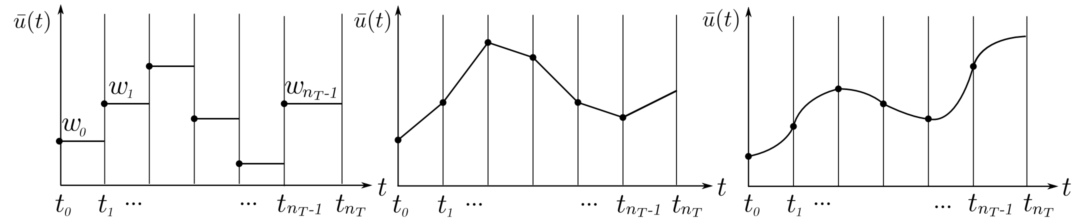


Figure 3.2.: Illustration of possible approximations of the control function $u(t)$ for $n_u = 1$: piecewise constant, piecewise linear, piecewise cubic.

Control discretization

The infinite-dimensional control function $u(t)$ is discretized using the following grid

$$t_0 < t_1 < \ldots < t_{n_T} = T. \tag{3.15}$$

The subintervals are denoted by $I_i := [t_i, t_{i+1}]$. We assume that the control discretization grid (3.15) is the same as the inequality and equality multi-point boundary condition grid used in (3.7). The control function $u(t)$ is locally approximated by basis functions $\xi_i(\cdot)$, $i \in 0, \ldots, n_T - 1$ with finite support:

$$\bar{u}(t)\big|_{I_i} = \xi_i(t, w_i), \tag{3.16}$$

where

$$w := (w_0^\intercal, \ldots, w_{n_T-1}^\intercal)^\intercal \quad \text{with} \quad w_i \in \mathbb{R}^{n_l \cdot n_u} \tag{3.17}$$

are the variables describing the basis functions. Possible choices for the basis functions include piecewise constant functions, piecewise linear functions, or piecewise cubic functions, which are illustrated in Figure 3.2. After the control discretization, the ODE in (3.7) depends on the variables w: $f(t, x(t), \xi_i(t, w_i), q)$ for $i \in I_i$.

State parameterization

As described in Section 3.1, there are several ways to numerically treat the differential states. We focus on the multiple shooting method and introduce $n_T + 1$ new variables s_i, $i = 0, \ldots, n_T$, where s_i describes the initial value of the following IVP:

$$\begin{aligned} \dot{x}(t) &= f(t, x(t), \xi_i(t, w_i), q), \quad t \in I_i && (3.18a) \\ x(t_i) &= s_i, && (3.18b) \end{aligned}$$

with the solution $x(t; t_i, s_i, w_i, q)$ and the multiple shooting grid (3.15). To obtain a continuous solution of the ODE in (3.7), we require n_T matching conditions:

$$x(t_{i+1}; t_i, s_i, w_i, q) = s_{i+1} \quad \forall i = 0, \ldots, n_T - 1. \tag{3.19}$$

For the parameterization of the differential states, we use the same grid as for the control discretization in order to obtain numerically efficient structures.

Constraint discretization

The mixed control-state constraints are enforced on the grid nodes only:

$$\begin{aligned} c(t_i, s_i, \xi_i(t_i, w_i), q) &\geq 0 \quad \forall i = 0, \dots, n_T - 1, && (3.20)\\ c(T, s_{n_T}, \xi_{n_T-1}(T, w_{n_T-1}), q) &\geq 0. && (3.21) \end{aligned}$$

If this discretization is not sufficient in practice, a more advanced approach can be used, which is described in [Pot06, PBS09].

Parameterizing the differential states and discretizing the controls and constraints leads to an NLP of the following form:

$$\begin{array}{lrcll} \underset{s,w,q}{\text{minimize}} & \multicolumn{4}{l}{\Phi_{\mathcal{M}}(s_{n_T}, q) + \sum_{i=0}^{n_T-1} \int_{t_i}^{t_{i+1}} \Phi_{\mathcal{L}}(t, x(t), \xi_i(t, w_i), q) \, \mathrm{d}t} \\ \text{subject to} & s_{i+1} & = & x(t_{i+1}; t_i, s_i, w_i, q) & \forall\, i = 0, \dots, n_T - 1 \\ & 0 & \leq & c(t_i, s_i, \xi_i(t_i, w_i), q) & \forall\, i = 0, \dots, n_T \\ & 0 & \leq & c(T, s_{n_T}, \xi_{n_T-1}(T, w_{n_T-1}), q) & \\ & 0 & = & r^{\text{eq}}(s_0, \dots, s_{n_T}, q) & \\ & 0 & \leq & r^{\text{ieq}}(s_0, \dots, s_{n_T}, q), & \end{array} \qquad (3.22)$$

which can then be solved with a tailored SQP method based on the simple version described in Section 2.2.1. Details about structure-exploiting methods for NLP (3.22) can be found in, e.g., [Lei99, LBS$^+$03].

3.3. Parameter estimation in dynamic systems

A parameter estimation problem minimizes the weighted adequately described differences between observed data and model response such that the underlying model equations based on a system of ODEs are satisfied. Following the overview given in [Hat08], we consider least-squares parameter estimation problems of the following form:

$$\begin{array}{lrcl} \underset{x,p}{\text{minimize}} & \multicolumn{3}{l}{\frac{1}{2} \sum_{i=0}^{n_m} \sum_{j=1}^{n_h} \left(\frac{h_j(t_i^m, x(t_i^m), p) - \eta_{ij}}{\sigma_{ij}} \right)^2} \\ \text{subject to} & \dot{x}(t) & = & f(t, x(t), p), \quad t \in I \\ & 0 & \leq & c(t, x(t), p), \quad t \in I \\ & 0 & = & r^{\text{eq}}(x(t_0), \dots, x(T), p) \\ & 0 & \leq & r^{\text{ieq}}(x(t_0), \dots, x(T), p), \end{array} \qquad (3.23)$$

with the system time $t \in I = [t_0, T] \subseteq \mathbb{R}$, measurements $\eta \in \mathbb{R}^{(n_m+1) \cdot n_h}$, the model response h_j mapping to $\mathbb{R}$ and variances σ_{ij}^2 of the corresponding measurement η_{ij} for $i = 0, \dots, n_m$ and $j = 1, \dots, n_h$. Variable x describes the differential states mapping to $\mathbb{R}^{n_x}$ and the function f denotes the right-hand side of the ODE, which is assumed to be piecewise Lipschitz continuous. The vector $p \in \mathbb{R}^{n_p}$ includes unknown parameters that are constant in time. Path constraints are denoted by c mapping to $\mathbb{R}^{n_c}$. The functions r^{eq} and r^{ieq} describe multi-point equality and inequality constraints mapping to $\mathbb{R}^{n_{\text{eq}}}$ and $\mathbb{R}^{n_{\text{ieq}}}$, respectively.

3.3.1. A numerical method for solving parameter estimation problems

For solving the parameter estimation problem (3.23), we in a first step parameterize the ODE in (3.23), which yields an NLP that is then solved with a tailored Generalized Gauß-Newton method based on the one described in Section 2.2.2. This procedure is explained briefly in the following (cf. [Boc87]).

As discussed in Section 3.1, there are several ways to numerically treat the ODE in (3.23). We focus on the multiple shooting method. As parameterization grid, we choose

$$t_0 < t_1 < \ldots < t_{n_T} = T, \tag{3.24}$$

where we for the ease of presentation assume that the multiple shooting grid is the same as the measurement grid

$$t_0 = t_0^m < t_1^m < \ldots < t_{n_m}^m = T, \tag{3.25}$$

where η_i is collected at time t_i^m.

We furthermore assume that the grid used in the multi-point equality and inequality constraints is the same grid as the multiple shooting grid in order to obtain numerically efficient structures. The multiple shooting parameterization described in Section 3.1 combined with the evaluation of the path constraints and the objective on grid (3.24) yields the following discretized parameter estimation problem:

$$\begin{array}{lrcll} \underset{s,p}{\text{minimize}} & \multicolumn{4}{l}{\dfrac{1}{2}\displaystyle\sum_{i=0}^{n_m}\sum_{j=1}^{n_h}\left(\frac{h_j(t_i,s_i,p)-\eta_{ij}}{\sigma_{ij}}\right)^2} \\ \text{subject to} & s_{i+1} & = & x(t_{i+1};t_i,s_i,p) & \forall\, i=0,\ldots,n_T-1 \\ & 0 & \leq & c(t_i,s_i,p) & \forall\, i=0,\ldots,n_T, \\ & 0 & = & r^{\text{eq}}(s_0,\ldots,s_{n_T},p), & \\ & 0 & \leq & r^{\text{ieq}}(s_0,\ldots,s_{n_T},p). & \end{array} \tag{3.26}$$

The NLP (3.26) is then solved with a tailored Generalized Gauß-Newton method based on the one described in Section 2.2.2.

Part II.

Mathematical methods

Chapter 4.

Hierarchical dynamic optimization

This chapter provides an introduction to a special class of hierarchical dynamic optimization problems: a parameter estimation problem constrained by an optimal control problem. The first part of this chapter is concerned with the general formulation of the problem setting, we then analyze challenges of this problem class, and provide a survey of literature and an overview of methods derived in this thesis. Finally, we discuss possible fields of application for this particular class of hierarchical dynamic optimization problems.

4.1. Problem formulation

We start with the general problem formulation of a parameter estimation problem constrained by a multi-stage optimal control problem (OCP) with discontinuous transitions expressed by nonlinear transition conditions, a system of underlying ordinary differential equations (ODEs), a Bolza-type objective function, nonlinear mixed control-state constraints, and coupled or decoupled nonlinear equality and inequality multi-point constraints, which is given by:

$$\begin{array}{lll}
\underset{\substack{x,u,q\\ \gamma,p}}{\text{minimize}} & \multicolumn{2}{l}{\displaystyle \frac{1}{2}\sum_{k=0}^{n_S-1}\sum_{i=0}^{n_{m_k}}\sum_{j=1}^{n_{h_k}} \frac{\left(h_{kj}(x_k(\tau^m_{ki}),q,p)-\eta_{kij}\right)^2}{\sigma^2_{kij}}} \\
\text{subject to} & \underset{x,u,q}{\text{minimize}} & \displaystyle \sum_{k=0}^{n_S-1}\left(\sum_{i=1}^{n_{\mathcal{M}}}\gamma_i\phi^{ki}_{\mathcal{M}}(x_k(t_{k+1}),q,p)+\sum_{i=n_{\mathcal{M}}+1}^{n_{\mathcal{M}}+n_{\mathcal{L}}}\gamma_i\int_{t_k}^{t_{k+1}}\phi^{ki}_{\mathcal{L}}(x_k(t),u_k(t),q,p)\,\mathrm{dt}\right) \\
& \text{subject to} & \begin{array}{lllr}
\dot{x}_k(t) & = f_k(x_k(t),u_k(t),q,p), & t\in[t_k,t_{k+1}], & k=0,\dots,n_S-1 \\
x_{k+1}(t_{k+1}) & = b_k(x_k(t_{k+1}),q,p), & & k=0,\dots,n_S-2 \\
0 & \le c_k(x_k(t),u_k(t),q,p), & t\in[t_k,t_{k+1}], & k=0,\dots,n_S-1 \\
0 & = r^{\text{eq}}_k(x_k(\tau_{k,0}),\dots,x_k(\tau_{k,n_k}),q,p), & & k=0,\dots,n_S-1 \\
0 & \le r^{\text{ieq}}_k(x_k(\tau_{k,0}),\dots,x_k(\tau_{k,n_k}),q,p), & & k=0,\dots,n_S-1
\end{array} \\
& \multicolumn{2}{l}{\sum_{i=1}^{n_{\mathcal{M}}+n_{\mathcal{L}}}\gamma_i=1,\ \gamma\ge 0} \\
& \multicolumn{2}{l}{b^{\text{lower}_p}\ \le\ p\ \le\ b^{\text{upper}_p}.}
\end{array} \tag{4.1}$$

We now discuss problem (4.1) in detail and note that the functions and variables in (4.1) are closely related to the OCP shown in Section 3.2 and the parameter estimation problem explained in Section 3.3. The lower-level problem of (4.1) is a multi-stage OCP with the stage index k and n_S model stages, where each model stage might have a different objective function, a different system of ODEs and different mixed control-state and multi-point constraints. The transition between model stage k and $k+1$ is described by stage transition conditions denoted as

$$x_{k+1}(t_{k+1}) = b_k(x_k(t_{k+1}),q,p). \tag{4.2}$$

In this thesis, multiple model stages are needed to describe the dynamics of human locomotion in Chapters 12 and 15, where further details are discussed.

The system time in (4.1) is denoted by $t \in [t_0, T]$, where the time horizon of model stage k is given by the interval $I_k := [t_k, t_{k+1}]$ with $k = 0, \ldots, n_S - 1$ and

$$t_0 < t_1 < \ldots < t_{n_S-2} < t_{n_S-1} < t_{n_S} = T. \tag{4.3}$$

The differential states are described by $x := (x_0, \ldots, x_{n_S-1})$, where x_k is the differential state on model stage k with $x_k : I_k \to \mathbb{R}^{n_x}$ for $k = 0, \ldots, n_S - 1$. Please note that we assume that the differential states are of the same dimension on each model stage. The control functions are given by $u := (u_0, \ldots, u_{n_S-1})$, where $u_k : I_k \to \mathbb{R}^{n_u}$ is the control function on model stage k for $k = 0, \ldots, n_S - 1$, which is assumed to be measurable. $p \in \mathbb{R}^{n_p}$ describes unknown and time-independent system parameters, and $q \in \mathbb{R}^{n_q}$ contains variables denoting time-independent control values.

The upper-level objective

$$\frac{1}{2} \sum_{k=0}^{n_S-1} \sum_{i=0}^{n_{m_k}} \sum_{j=1}^{n_{h_k}} \frac{(h_{kj}(x_k(\tau_{ki}^m), q, p) - \eta_{kij})^2}{\sigma_{kij}^2} \tag{4.4}$$

is a least-squares parameter estimation objective. A basic version of a one-level parameter estimation problem is described in Section 3.3. The objective (4.4) includes measurements η of dimension $\sum_{k=0}^{n_S-1}((n_{m_k} + 1) \cdot n_{h_k})$, where $n_{m_k} + 1$ is the number of measurement times and n_{h_k} the number of measurement functions on stage k, and the model response $h_{kj} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_q} \times \mathbb{R}^{n_p} \to \mathbb{R}$ with $j = 1, \ldots, n_{h_k}$ and $k = 0, \ldots, n_S - 1$. This means that on each model stage k, we have n_{h_k} model responses or measurement functions that are taken into account, and each model response on model stage k is evaluated at $n_{m_k} + 1$ measurement points. σ_{kij}^2 denotes the variance corresponding to measurement η_{kij} with $k = 0, \ldots, n_S - 1$, $i = 0, \ldots, n_{m_k}$ and $j = 1, \ldots, n_{h_k}$. On each model stage k, measurements are observed on the not necessarily equidistant time grid

$$t_k = \tau_{k,0}^m < \tau_{k,1}^m < \ldots < \tau_{k,n_{m_k}-1}^m < \tau_{k,n_{m_k}}^m = t_{k+1}. \tag{4.5}$$

We assume that the measurement errors are normally distributed with mean 0, that they are independent from each other and additive, and that there are no systematic errors (cf., e.g., [Kö02]).

The lower-level problem

$$\begin{array}{llll}
\underset{x,u,q}{\text{minimize}} & \displaystyle\sum_{k=0}^{n_S-1} \left(\sum_{i=1}^{n_\mathcal{M}} \gamma_i \phi_\mathcal{M}^{ki}(x_k(t_{k+1}), q, p) + \sum_{i=n_\mathcal{M}+1}^{n_\mathcal{M}+n_\mathcal{L}} \gamma_i \int_{t_k}^{t_{k+1}} \phi_\mathcal{L}^{ki}(x_k(t), u_k(t), q, p) \, \mathrm{dt} \right) & & \\
\text{subject to} & \dot{x}_k(t) \quad = f_k(x_k(t), u_k(t), q, p), & t \in [t_k, t_{k+1}], & k = 0, \ldots, n_S - 1 \\
& x_{k+1}(t_{k+1}) \quad = b_k(x_k(t_{k+1}), q, p), & & k = 0, \ldots, n_S - 2 \\
& 0 \quad \leq c_k(x_k(t), u_k(t), q, p), & t \in [t_k, t_{k+1}], & k = 0, \ldots, n_S - 1 \\
& 0 \quad = r_k^{\text{eq}}(x_k(\tau_{k,0}), \ldots, x_k(\tau_{k,n_k}), q, p), & & k = 0, \ldots, n_S - 1 \\
& 0 \quad \leq r_k^{\text{ieq}}(x_k(\tau_{k,0}), \ldots, x_k(\tau_{k,n_k}), q, p), & & k = 0, \ldots, n_S - 1
\end{array} \tag{4.6}$$

is an OCP based on the simpler one-stage problem described in Section 3.2. The objective in (4.6) is of a special type – a weighted sum of Mayer terms $\phi_\mathcal{M}^{ki}(\cdot)$ with $i = 1, \ldots, n_\mathcal{M}$ and

Lagrange terms $\phi_{\mathcal{L}}^{ki}(\cdot)$ with $i = n_{\mathcal{M}} + 1, \dots, n_{\mathcal{L}} + n_{\mathcal{M}}$ on each model stage k:

$$\underset{x,u,q}{\text{minimize}} \sum_{k=0}^{n_S-1} \left(\sum_{i=1}^{n_{\mathcal{M}}} \gamma_i \phi_{\mathcal{M}}^{ki}(x_k(t_{k+1}), q, p) + \sum_{i=n_{\mathcal{M}}+1}^{n_{\mathcal{M}}+n_{\mathcal{L}}} \gamma_i \int_{t_k}^{t_{k+1}} \phi_{\mathcal{L}}^{ki}(x_k(t), u_k(t), q, p) \, \mathrm{dt} \right) \tag{4.7}$$

with weights γ_i, $i = 1, \dots, n_{\mathcal{M}} + n_{\mathcal{L}}$, which are variables in the upper-level problem of (4.1).

The ODE in (4.1) given by

$$\dot{x}_k(t) = f_k(x_k(t), u_k(t), q, p), \qquad t \in [t_k, t_{k+1}],\ k = 0, \dots, n_S - 1 \tag{4.8}$$

combined with mixed control-state constraints

$$0 \leq c_k(x_k(t), u_k(t), q, p), \qquad t \in [t_k, t_{k+1}],\ k = 0, \dots, n_S - 1, \tag{4.9}$$

the multi-point equality and inequality constraints

$$\begin{aligned} 0 &= r_k^{\text{eq}}(x_k(\tau_{k,0}), \dots, x_k(\tau_{k,n_k}), q, p),\ k = 0, \dots, n_S - 1 \\ 0 &\leq r_k^{\text{ieq}}(x_k(\tau_{k,0}), \dots, x_k(\tau_{k,n_k}), q, p),\ k = 0, \dots, n_S - 1, \end{aligned} \tag{4.10}$$

and the stage transition conditions

$$x_{k+1}(t_{k+1}) = b_k(x_k(t_{k+1}), q, p),\ k = 0, \dots, n_S - 2 \tag{4.11}$$

describes the dynamics of the modeled process. We note that for the sake of a compact presentation, we skip the explicit dependency of f, $\phi_{\mathcal{L}}$ and c on t, which still could be included in the given formulation via a trivial ODE.

The function $f_k : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_q} \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_x}$ is the ODE's right-hand side on model stage k, which is assumed to be piecewise Lipschitz continuous. The condition (4.11) describes the transition of the differential states x_k and x_{k+1} at time t_{k+1} from stage k to stage $k+1$ for $k = 0, \dots, n_S - 2$. An example of the solution of an ODE of type (4.8) with three model stages, a discontinuous transition and a non-differentiable transition is illustrated in Figure 4.1. The mixed control-state constraints on stage k are denoted by $c_k : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_q} \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_{c_k}}$. The multi-point equality and inequality conditions r_k^{eq} and r_k^{ieq} are mapping to $\mathbb{R}^{n_{\text{eq}_k}}$ and $\mathbb{R}^{n_{\text{ieq}_k}}$ on each model stage and have the differential state x_k on stage k evaluated on the grid

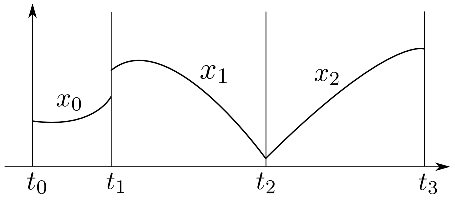


Figure 4.1.: Example of a differential state x with three model stages.

$$t_k = \tau_{k,0} < \tau_{k,1} < \ldots < \tau_{k,n_k-1} < \tau_{k,n_k} = t_{k+1}, \tag{4.12}$$

as well as q and p as arguments.

Due to the well-known fact that the objective function is invariant in terms of scaling, we need the constraint

$$\sum_{i=1}^{n_{\mathcal{M}}+n_{\mathcal{L}}} \gamma_i = 1 \tag{4.13}$$

on the upper level of (4.1) to avoid redundant solutions. We furthermore seek for solutions of the lower-level problem that minimize a positive combination of subcriteria, therefore we add the constraint

$$\gamma \geq 0, \tag{4.14}$$

which requires γ to be componentwisely nonnegative. We furthermore have bounds b^{lower_p} and b^{upper_p} on the parameters p:

$$b^{\text{lower}_p} \ \leq \ p \ \leq \ b^{\text{upper}_p} \tag{4.15}$$

in the upper-level problem of (4.1). More complicated constraints on the upper level are also possible and realized in the implementation of the algorithm presented in Chapter 5, but for the ease of notation, we skip them here.

4.2. The difference between classical parameter estimation and parameter estimation constrained by an optimal control problem

Problem (4.1) describes the task of estimating parameters in OCPs. Classical parameter estimation problems stated in Section 3.3 describe the estimation of unknown parameters p in models that are commonly given by multi-point boundary value problems (MPBVP) with additional constraints of the following type:

$$\begin{array}{rcl} \dot{x}(t) & = & f(x(t), p) \quad t \in [t_0, T] \\ 0 & \leq & c(x(t), p), \quad t \in [t_0, T] \\ 0 & = & r^{\text{eq}}(x(t_0), \ldots, x(T), p) \\ 0 & \leq & r^{\text{ieq}}(x(t_0), \ldots, x(T), p). \end{array} \tag{4.16}$$

Problem (4.16) typically describes the dynamics of a process that depends on unknown parameters p. The parameters p are then determined by fitting the model (4.16) to measurement data in, e.g., a least-squares sense, described by the objective function

$$\underset{x,p}{\text{minimize}} \, \frac{1}{2} \sum_{i=0}^{n_m} \sum_{j=1}^{n_h} \left(\frac{h_j(x(t_i), p) - \eta_{ij}}{\sigma_{ij}} \right)^2 \tag{4.17}$$

(as discussed in Section 3.3). In contrast to classical parameter estimation, in (4.1), the underlying model itself is a dynamic optimization problem – the OCP (4.6). This is the essential difference between classical parameter estimation as described in Section 3.3 and the problem class (4.1) investigated in this thesis. The underlying model in (4.1) describes a process depending on unknown parameters γ and p that optimizes a certain criterion. As in model (4.16), unknown parameters p might be included in the dynamics of the optimal control model (4.6), i.e. in the ODE's right-hand side, the mixed control-state constraints and in the multi-point boundary conditions. The unknowns γ, however, are contained in the objective of the lower-level OCP for which we assume that it is an additive weighted combination of subcriteria. However, the relative importance of each subcriterion, i.e. the weight γ_i, is unknown and cannot be measured. Hence, the weights γ_i with $i = 1, \dots, n_{\mathcal{M}} + n_{\mathcal{L}}$ are additional unknown model parameters that need to be determined from measurement data. This is a further difference between classical parameter estimation as described in Section 3.3 and problem class (4.1), as well as the fact that we have controls as variables in the underlying model (4.6).

We close this section with a remark on the variables included in (4.1), which we refer to as follows:

- Upper-level variables:
  time-independent parameters γ and p

- Lower-level variables:
  differential states x, controls u, time-independent control values q.

The reason why all of them appear as variables in the upper-level objective of (4.1) is that we want to highlight that the main approach derived in this thesis is a so-called *all-at-once approach*, which seeks for optimal upper-level and lower-level variables simultaneously in one loop of an iterative method.

4.3. Challenges

Considering problem (4.1), it is obvious that it neither belongs to the class of OCPs described in Section 3.2, nor to the class of parameter estimation problems described in Section 3.3. Problem (4.1) includes ingredients from both classes and combines the two problem types via a second optimization level. We have to deal with an infinite-dimensional hierarchical optimization problem when solving (4.1), and neither optimal control methods from Section 3.2 nor parameter estimation techniques from Section 3.3 can directly be applied to do so. This means, based on methods from Chapter 3, we have to derive a new approach that allows to solve the hierarchical problem (4.1).

In literature, for finite-dimensional bilevel nonlinear programs (NLPs), as well as for infinite-dimensional bilevel problems, one finds the following two main classes of numerical approaches:

- **All-at-once (or simultaneous) approaches:** Methods, which transform the bilevel problem into a one-level problem and solve for all variables at once.

- **Bilevel approaches:** Methods that stick to the bilevel structure and each time the upper-level objective is evaluated, the lower-level problem is solved.

The challenges that have to be faced when solving (4.1) of course depend on whether an all-at-once or a bilevel approach is used. In this thesis, we focus on the derivation of an all-at-once approach for solving (4.1). In order to be able to compare our all-at-once method to alternative approaches, we also derive two bilevel approaches (cf. Chapter 7). Challenges of the respective approaches are discussed in detail in Chapters 5, 6 and 7. In this section, we provide a short list of some of the difficulties that appear due to the combination of parameter estimation with optimal control in problem (4.1). Solving (4.1) includes the following challenges:

- Standard methods and algorithms for this problem class do not exist, a new and efficient method has to be derived.
- In an all-at-once approach, a complementarity constraint arises and needs to be treated reliably.
- Higher-order derivatives are required and need to be calculated in an efficient way.
- The number of variables becomes much larger, efficient and structure-exploiting methods are indispensable.

All of the items in the preceding list will be discussed in detail in the following chapters.

4.4. Survey of literature & methods derived in this thesis

In literature, one often refers to problem (4.1) or related problems as

- hierarchical dynamic optimization problems
- bilevel OCPs
- inverse OCPs
- parameter estimation problems constrained by OCPs.

We mainly use the first term since it expresses that the mathematical methods derived in this thesis can be applied to a very general class of problems. In the remainder of this section, we give an overview of current research in the area of hierarchical dynamic optimization. In numerous fields including mathematics, engineering, robotics or biomechanics, one can find roots of hierarchical dynamic optimization. In this thesis, we mainly focus on literature that is concerned with mathematical and algorithmic research for hierarchical dynamic optimization.

Hierarchical dynamic optimization problems are a generalization of bilevel or multi-level NLPs. In contrast to dynamic optimization problems, bilevel programs only include finite-dimensional variables. Bilevel programming has its origin in modeling economic processes and is closely related to the field of game theory. Game theory is a rather new discipline in mathematics reaching back to the early nineteenth century. Bilevel programs are closely related to mathematical programs with complementarity constraints (MPCCs). An overview of MPCCs can be found in, e.g., [LPR96]. [Ye95] is one of the first publications on generalized bilevel programs where the lower-level problem is described by an OCP. Algorithmic works on parameter estimation based on optimal control models are rather rare. Recent investigations can be found in, e.g., [APS$^+$10, HSB12]. As we have seen in the last paragraph, the field of hierarchical dynamic optimization combines several research areas. Even though some references for optimal control, parameter estimation and MPCCs have already been named in Part I of this thesis, we now want to specify and relate literature of all main roots of the field of hierarchical dynamic optimization. In the following, we provide a survey of literature for each of the following topics:

- bilevel programming

- mathematical programs with complementarity constraints
- optimal control
- parameter estimation
- hierarchical dynamic optimization.

Bilevel programming

Early works on hierarchical coupled decisions are mostly rooted in economics. Problems involving several decision variables on different levels have been investigated at the beginning of the nineteenth century by, e.g., Cournot [Cou38] or later by, e.g., von Stackelberg [vS34]. An overview of research in the field of bilevel programming can be found in, e.g., [VC94, Dem02, Dem03, BCS07, Bar10]. The field of game theory as it is known nowadays has been established in the early twentieth century with the book *Spieltheorie und wirtschaftliches Verhalten* (engl.: game theory and economic behavior) by John von Neumann and Oskar Morgenstern [vNM53]. [BM73] is one the first publications where the problem of several decision variables on different levels is formulated as an optimization problem. A generalization of the work of Bracken and McGill to MPCCs, where the solution of the lower-level problem is characterized implicitly, can be found in [HP88]. An overview of recent developments in the field of bilevel programming can be found in, e.g., [Dem02, BCS07, Bar10, AS13, DZ13] and the references therein.

Mathematical programs with complementarity constraints

Transforming a bilevel program with inequality constraints into a one-level NLP by replacing the lower-level problem by its necessary conditions for optimality leads to an MPCC. An overview of the theory and numerical methods for MPCCs can be found in, e.g., [LPR96]. Common numerical approaches for solving MPCCs are based on branch-and-bound methods [Bar88], nonsmooth optimization [OKZ98], interior-point methods [LLCN06, RB05], piecewise sequential quadratic programming (SQP) methods [LPR97], or penalization, regularization and relaxation techniques [Sch01, LF03, DFNS05, SU10, HKS13, HLSB13]. Numerical and theoretical results for SQP methods applied to MPCCs can be found in [FLRS06, Ley03]. The publications cited in this paragraph and the references therein give a broad overview of developments in the last decade in the field of MPCCs.

Optimal control

Standard textbooks for optimal control include [PBGM62, BH75, Bie10, Ger12] and provide a broad survey of both theory and numerical methods for optimal control. A summary of more recent research in the field of optimal control can be found in, e.g., [BBB$^+$01]. An efficient structure-exploiting direct multiple shooting method for optimal control is described in [BP84, Lei99].

Parameter estimation

An overview of theory and numerical methods for parameter estimation can be found in the references [Bar74, SW89, Sch91]. A structure-exploiting method for parameter estimation in boundary value problems (BVPs) including nonlinear differential algebraic equations based on multiple shooting parameterization and the Generalized Gauß-Newton (GGN) method is described in detail in [Boc87, Sch88] and [BES88]. An overview of parameter estimation methods, the multiple shooting parameterization, and the GGN method is given in [BKS07]. Tailored globalization strategies in the context of parameter estimation can be found in [BKS00], and methods for analyzing the statistical quality of the estimated parameters are described in [Boc87].

Hierarchical dynamic optimization

Before we provide a survey of literature for hierarchical dynamic optimization, we give an overview of existing types of methods and the methods derived in this thesis for solving hierarchical dynamic optimization problems. Therefore, we consider Figure 4.2, which shows the main approaches in current research, and explain Layer 1 – Layer 5 in Figure 4.2 for all-at-once approaches and bilevel methods in the following.

Layer 1: Handling of the bilevel structure. In general, there are two main classes of approaches: all-at-once approaches and bilevel approaches. The idea of all-at-once approaches is to transform the bilevel problem (4.1) into a one-level problem by means of necessary conditions for optimality. We note that for a general OCP on the lower level in (4.1), the bilevel problem and the resulting one-level problem are not necessarily mathematically equivalent. However, in this thesis we are not concerned with this issue and refer to [Mir99, Dem02, Dem03, DZ13] for a detailed discussion. As the name implies, bilevel approaches do not reformulate problem (4.1) as one-level problem, they keep the bilevel structure and each time the upper-level objective of (4.1) is evaluated, the lower-level OCP is solved completely. In the following, we discuss both classes – all-at-once and bilevel approaches for hierarchical dynamic optimization.

All-at-once approaches

Layer 2: Lower-level treatment. Using an all-at-once approach to solve (4.1), we need to formulate necessary conditions for optimality of the lower-level OCP in order to replace the lower-level problem by these conditions and to obtain a one-level problem. The formulation of necessary conditions for optimality can be done in different ways and corresponds to Layer 2 in Figure 4.2. We consider the following two variants: a direct and an indirect treatment of the lower-level problem. A direct treatment of the lower-level OCP means that in the optimization procedure, we directly iterate on variables that approximate the control function. This again implies that in a first step the control function is discretized and the differential states are parameterized/discretized (as indicated on Layer 4, cf. Chapter 5). An indirect treatment of the lower-level OCP, however, means that the controls in (4.6) are eliminated by means of Pontryagin's maximum principle (PMP) and in the optimization procedure, we do not directly iterate on variables approximating the controls – the controls are determined indirectly by applying PMP (cf. Chapter 6).

Layer 3: Upper-level treatment. In case of an all-at-once approach with a direct lower-level treatment, problem (4.1) including the upper-level objective has been discretized/parameterized already, and there is no decision left on Layer 3. If the lower-level problem is treated with an indirect approach, the lower-level OCP (including mixed control-state constraints) is replaced by a highly complex BVP with jump and switch conditions stating its necessary optimality conditions (as explained in Section 3.2). This means instead of solving (4.1), we have to solve an infinite-dimensional one-level parameter estimation problem as described in Section 3.2 with a special structure und complex jump and switch conditions. This can then be done in a direct manner, or, again in an indirect way based on PMP. However, both ways are highly complex in the presence of inequality constraints and a deep knowledge of the process is required. This means, for an all-at-once approach with an indirect lower-level treatment, one can either chose a direct or an indirect treatment of the upper-level problem of (4.1) on Layer 3.

Layer 4: Discretization method. Independent of the choices on previous layers, Layer 4 determines the discretization or parameterization of the states. On Layer 4, we consider multiple shooting and collocation, both described in Chapter 3. If multiple shooting is used, we use the term parameterization instead of discretization in the sense that the initial values on each multiple shooting interval parameterize the solution of the MPBVP. With a direct treatment of the lower-level problem on Layer 2, we on Layer 4 use multiple shooting or collocation to parameterize/discretize the lower-level OCP and obtain a finite-dimensional bilevel program. We then make use of the Karush-Kuhn Tucker (KKT) conditions described in Chapter 2 to formulate first-order necessary conditions for optimality of the discretized lower-level problem of (4.1). That way, we obtain a finite-dimensional one-level program. With an indirect treatment of the lower level on Layer 2, and a direct treatment of the upper-level problem on Layer 3, we use, e.g., multiple shooting and obtain a finite-dimensional one-level parameter estimation problem with highly complex constraints in the presence of inequality constraints on Layer 4. An indirect treatment of the lower-level problem on Layer 2 and an indirect treatment of the upper-level problem on Layer 3 leads to an infinite-dimensional BVP with jump and switch conditions that is discretized with, e.g., collocation.

Layer 5: NLP method. On Layer 5, it remains to solve an NLP of a special form depending on the choices on previous layers. The resulting NLP can be solved with a tailored sequential quadratic programming (SQP) method, a tailored GGN method (both are discussed in Chapter 2) or an interior point method as, e.g., the one described in [WB06]. For an indirect treatment on Layer 2 and Layer 3, Layer 5 describes the solution of a system of equations with, e.g., Newton's method in this case. In the following chapters, we will see how the NLP resulting from a direct treatment of the lower-level problem differs from the NLP implied by an indirect treatment on Layer 2.

Bilevel approaches

Layer 2: Lower-level treatment. We note that in Figure 4.2, we use the same order of layers for all-at-once and bilevel approaches for a better comparison. Using a bilevel approach to solve the hierarchical dynamic optimization problem (4.1), we have to solve the lower-level OCP in each upper-level iteration. For solving the OCP (4.6), we consider a direct approach, where the controls are discretized in the lower-level OCP in a first step. One could in principle also choose an indirect approach based on PMP on Layer 2. However, for a general OCP (4.6) including multiple controls and path constraints, it is highly complex to derive necessary conditions for optimality based on PMP and the corresponding BVP with jump and switch conditions. This can hardly be done in an automatic procedure (cf. Section 3.2). In addition, when using a bilevel approach, we might have to derive necessary conditions for optimality and the corresponding BVP for problem (4.6) in each iteration of the optimization procedure on the upper level of (4.1) depending on whether the structure of the OCP's solution changes. Hence, we focus on a direct treatment of the lower-level OCP on Layer 2.

Layer 3: Upper-level treatment. A common approach for the optimization of the upper-level problem in (4.1) in a bilevel method is a derivative-free optimization (DFO) technique like, e.g., the ones described in [CSV09]. There are also alternatives which require derivatives like a gradient/steepest descent method [NW99] or a bundle method [SZ92]. However, the computation of these derivatives is very expensive and needs to be done in an efficient and reliable way.

Layer 4: Discretization method. In all the bilevel approaches we consider in this thesis, the differential states in the lower-level OCP are parameterized using multiple shooting.

Layer 5: NLP method. After parameterizing and discretizing the lower-level OCP (4.6), we obtain a structured NLP, which is solved with a tailored SQP method each time the upper-level objective has to be evaluated.

Works in the field of hierarchical dynamic optimization

After explaining different types of solution approaches for hierarchical dynamic optimization problems, we now give an overview of current research in this field.

Hierarchical dynamic optimization problems are often called inverse OCPs in engineering. Early works on inverse optimal control originating from engineering can be found in, e.g., [Kal64, Mas68, Cas80] and the references therein. More recent research on inverse optimal control and human locomotion also originates from engineering and can be found in, e.g., [ALB12, PJJB12]. The idea in the last two references is to assume that the observed measurements are perfect and do not include measurement errors, but the observed values are only approximately optimal. However, the latter approach requires a continuous observation of all differential states and controls, which is not the case for many applications. In particular, this is not given for the real-world application presented in Chapter 15. Hence, this approach is not shown in Figure 4.2.

Mombaur et al. work on bilevel approaches for inverse optimal control in robotics and biomechanics [KML10, MS10, MOC13]. The lower-level problem is solved with a direct multiple shooting method for optimal control implemented in the software package MUSCOD-II [Lei96, Lei99, DLS01]. The upper-level problem is optimized with a DFO technique [Pow08, Pow09]. The bilevel approach of Mombaur et al. is used to estimate the constitution of the objective of overall human path generation, of human running, and of human yoyo playing. The upper-level problem in all mentioned applications is a parameter estimation problem. The approach of Mombaur et al. is marked in Figure 4.2.

In [FLHS10], the authors are concerned with inverse optimal control methods for increasing the fairness of air races. The lower-level problem is treated with a direct multiple shooting method, and on the upper level, a gradient method is used as marked in Figure 4.2.

Knauer et al. [KB06, Kna09, Kna12] work on mathematical methods for bilevel optimal control using an all-at-once approach. The lower-level OCP is replaced by necessary conditions for optimality derived by using PMP. The resulting OCP is then transformed into a MPBVP, again by using PMP. The MPBVP is solved with a collocation method. Knauer et al. also follow a second strategy, where the lower level is again treated with an indirect method, but the upper-level problem is treated with a direct method based on a multiple shooting parameterization of the differential states. The resulting NLP is solved with an SQP method. Inequality constraints in the lower-level OCP are not considered. Both methods are used to improve the speed and the safety of container cranes, and are included in Figure 4.2.

Albrecht et al. [APS$^+$10, KAS$^+$10, ARARU$^+$11, APBLU12, ALU12] use bilevel optimal control to estimate the constitution of the objectives of human arm motions, human-like driving styles in autonomous cars, and human navigation with crossing-interferer. The upper-level objective is always of parameter-estimation-type. Albrecht et al. use an all-at-once approach with a direct treatment of the lower-level problem and a discretization of the differential states based on collocation. The resulting NLP is solved with an interior point method. The strategy of Albrecht et al. is shown in Figure 4.2.

In this thesis, we mainly focus on the derivation of an all-at-once approach with a direct treatment of the lower-level OCP. The differential states are parameterized with multiple shooting and the resulting NLP is solved with a GGN or an SQP method (cf. Chapter 5). In the following chapters, we in detail discuss our mathematical method for hierarchical dynamic optimization problems of type (4.1). In addition, we derive a corresponding algorithm, which allows to efficiently solve large real-world problems by exploiting the problem structure. Furthermore, we in detail discuss the complexity of the treatment of inequality constraints in the lower-level OCP. In an all-at-once approach with a direct treatment of the lower-level problem, inequality constraints in the OCP result in an MPCC (cf. Section 2.3). Our direct all-at-once approach including the treatment of the complementarity constraint is described in detail in the Chapters 5 and 8, and its implementation is explained in Chapter 9. First results can be found in [Hat08, HSB12, HLSB13]. To compare our approach to other strategies, we furthermore derive an all-at-once technique with an indirect treatment of the lower-level problem. The resulting parameter estimation problem is then solved with a direct multiple shooting method and either an SQP or a GGN method on NLP level (cf. Chapter 6). Furthermore, for comparison, we derive bilevel approaches, where the lower-level problem is solved with a direct multiple shooting method and an SQP method on NLP level. We consider a DFO technique as well as a bundle method for solving the upper-level parameter estimation problem. Both strategies are described in Chapter 7, and all methods mentioned in this paragraph are marked in Figure 4.2.

4.5. Possible fields of application

We finally survey possible fields of application of hierarchical dynamic optimization with a focus on hierarchical dynamic optimization problems with a parameter estimation problem on the upper level. They all have in common that they include a process in nature, for which we have evidence to assume that it runs optimally with respect to some optimization criterion. The optimality assumption of certain processes is a common basis in many fields of research, like in, e.g., the field of bionics (cf. Chapter 1). We recall the following quote from

Leonhard Euler, which summarizes the assumption behind modeling processes by optimal control problems.

> *Namely, because the shape of the whole universe is the most perfect and, in fact, designed by the wisest creator, nothing in all the world will occur in which no maximum or minimum rule is somehow shining forth...*
>
> Leonhard Euler (1744)

In the following, we provide a non-complete list of possible fields of application for estimating parameters in optimal control models of processes occurring in nature, with a focus on human motion and locomotion. Human motion and locomotion is a large field with many open questions. Some of them can be answered by means of hierarchical dynamic optimization. A discussion of the optimality assumption in the field of human motion can be found in, e.g., [Tod04, BPS06, FMKB13]. Apart from human motion, there are also works on the optimal motion of animals, which can be found in [Ale84, Ale96] and the references therein.

New insights into human motion. If we assume that humans move optimally, the first question that arises is: What is the optimality criterion or cost function of human motion? The optimality criterion of human motion is the core of many research projects that are concerned with human motion and human locomotion, and is discussed in, e.g., [BPS06, GFK09, MS10, ALU12, FMKB13]. Having a method which allows to estimate the constitution of the objective of human motion would in general lead to new insights and a better understanding of why humans move the way they move.

Classification of gaits. The constitution of the objective function of human locomotion can be used to derive classification schemes for human gait, which then can be applied to assist in clinical decision making.

Evaluation of treatments. An important issue in current medical research is how to evaluate the success of treatments. Optimal control models for human locomotion can be used to derive criteria that imply whether a specific treatment has been successful.

Treatment planning. Being able to estimate parameters in an optimal control model of the motion or the gait of a human would for the first time allow to establish a basis for treatment planning in medicine. The main idea is discussed in the following. Currently, many surgeries are completely based on the experience of the attending physician (cf. Chapter 11). In the majority of cases, there are no models involved at all. Having a model, which describes the patient's motion would allow to plan in advance. For cerebral palsy patients, e.g., this means that one could in advance use the derived gait model to predict how a certain surgery would affect the patient's gait. These results can then be combined with the physician's experience to precisely plan the surgery. A discussion of the lack of a basis for treatment planning for cerebral palsy patients can, e.g., be found in [AD05].

For a more detailed discussion of the fields of application mentioned above, we refer to Chapters 11 and 15. Further areas where hierarchical dynamic optimization can be used include the identification of behavioral and emotional aspects in human locomotion (cf.

[FMKB13]), the control of robots in a human-like manner (cf. [ARARU^{+}11, ALU12]), or the investigation of generating paintings (cf. [RMS09, RMS11, SM13]).

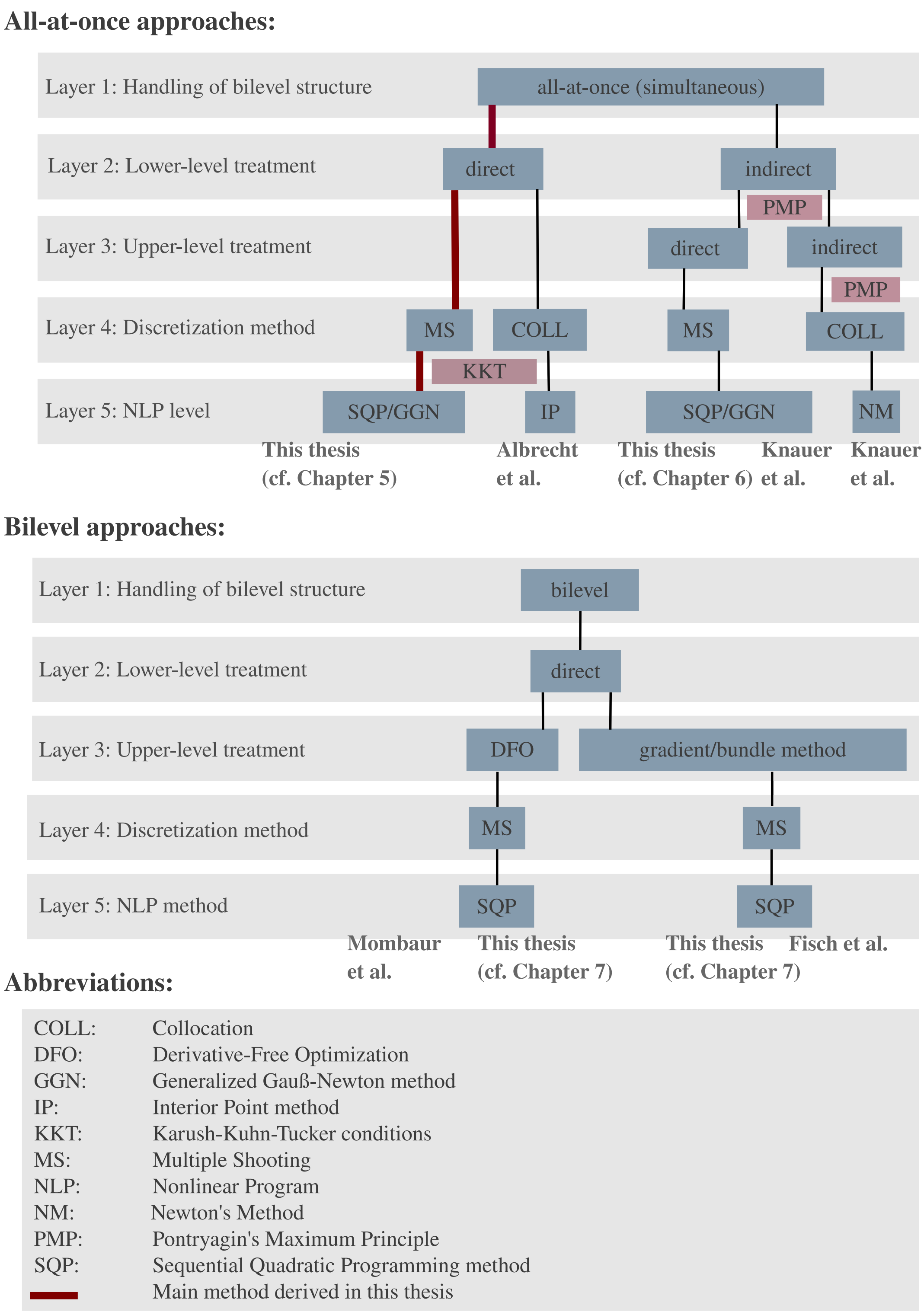

All-at-once approaches:
Layer 1: Handling of bilevel structure
all-at-once (simultaneous)
Layer 2: Lower-level treatment
direct
indirect
PMP
Layer 3: Upper-level treatment
direct
indirect
PMP
Layer 4: Discretization method
MS
COLL
MS
COLL
KKT
Layer 5: NLP level
SQP/GGN
IP
SQP/GGN
NM
This thesis (cf. Chapter 5)
Albrecht et al.
This thesis (cf. Chapter 6)
Knauer et al.
Knauer et al.
Bilevel approaches:
Layer 1: Handling of bilevel structure
bilevel
Layer 2: Lower-level treatment
direct
Layer 3: Upper-level treatment
DFO
gradient/bundle method
Layer 4: Discretization method
MS
MS
Layer 5: NLP method
SQP
SQP
Mombaur et al.
This thesis (cf. Chapter 7)
This thesis (cf. Chapter 7)
Fisch et al.
Abbreviations:
COLL: Collocation
DFO: Derivative-Free Optimization
GGN: Generalized Gauß-Newton method
IP: Interior Point method
KKT: Karush-Kuhn-Tucker conditions
MS: Multiple Shooting
NLP: Nonlinear Program
NM: Newton's Method
PMP: Pontryagin's Maximum Principle
SQP: Sequential Quadratic Programming method
🟥 Main method derived in this thesis

Figure 4.2.: Overview of existing methods for hierarchical dynamic optimization problems.

Chapter 5.

An efficient direct all-at-once approach for hierarchical dynamic optimization

In this chapter, we derive our main mathematical method for solving hierarchical dynamic optimization problems of type (4.1): an efficient direct all-at-once approach. We start with a brief sketch of the basic approach and then describe the method's main steps in detail: the parameterization and discretization of (4.1), the formulation of necessary conditions for optimality of the parameterized and discretized lower-level optimal control problem, and the Gauß-Newton-type method for solving the resulting nonlinear program. Furthermore, the structure of the nonlinear program is discussed in detail in order to present a structure-exploiting numerical algorithm for solving (4.1). The derivation of the mathematical method and the corresponding algorithm is strongly coupled and both the method itself as well as the associated algorithm are presented in this chapter by pointing out how the quantities needed in the mathematical method can be computed efficiently. A continuation of this discussion can be found in Chapter 9, where the implementation of the algorithm derived in this chapter is presented.

5.1. A sketch of the mathematical method

Our direct all-at-once approach for hierarchical dynamic optimization problems of type (4.1) is marked in the overview of approaches in Figure 4.2 with a bold line. The main steps of the mathematical method are shown in Figure 5.1. We derive an all-at-once approach, where

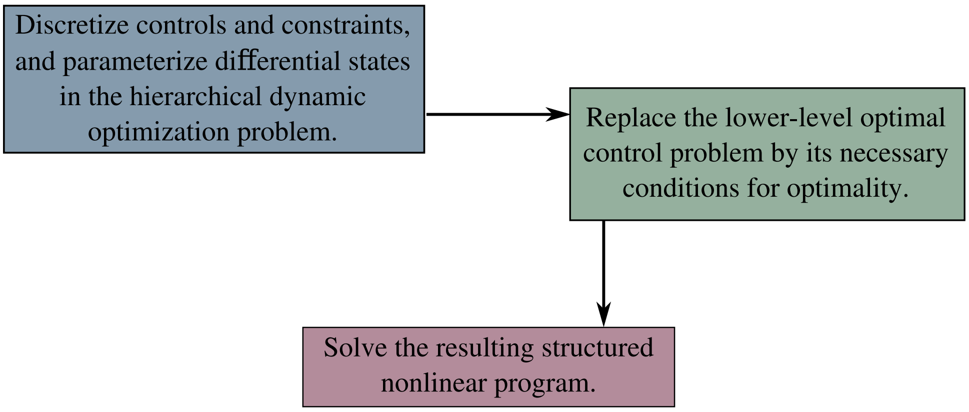

Discretize controls and constraints, and parameterize differential states in the hierarchical dynamic optimization problem.
Replace the lower-level optimal control problem by its necessary conditions for optimality.
Solve the resulting structured nonlinear program.

Figure 5.1.: The main steps of the direct all-at-once approach for hierarchical dynamic optimization problems.

the lower-level optimal control problem (OCP) is treated with a direct method. Hence, in the first step (the first box in Figure 5.1), we locally approximate the control function by basis functions with finite support and discretize the mixed control-state constraints. The differential states are parameterized based on the multiple shooting method described in Section 3.1. This leads to a finite-dimensional nonlinear bilevel program with a very special structure inherited by the control and constraint discretization and the multiple shooting parameterization.

In the second step (the second box in Figure 5.1), we formulate necessary conditions for optimality – the Karush-Kuhn-Tucker conditions described in Section 2.1 – of the parameterized and discretized lower-level OCP. The parameterized/discretized lower-level problem is then replaced by its necessary conditions for optimality, which leads to a one-level nonlinear program (NLP) with a special structure originating from two sources: the control and constraint discretization/multiple shooting parameterization and the bilevel structure of the original problem (4.1).

The resulting structured NLP is then solved with a tailored Generalized Gauß-Newton (GGN) method (the third box in Figure 5.1) based on the one presented in Section 2.2.2.

The steps mentioned in this section and the detailed structure of the resulting NLP are described in the remainder of this chapter. First approaches to the mathematical method derived in this chapter can be found in [Hat08, HSB12].

5.2. Parameterization and discretization

We start with recalling the problem setting – the hierarchical dynamic optimization problem introduced in Section 4.1:

$$
\begin{array}{lll}
\underset{\substack{x,u,q\\ \gamma,p}}{\text{minimize}} & \multicolumn{2}{l}{\displaystyle\frac{1}{2}\sum_{k=0}^{n_S-1}\sum_{i=0}^{n_{m_k}}\sum_{j=1}^{n_{h_k}}\frac{(h_{kj}(x_k(\tau_{ki}^m),q,p)-\eta_{kij})^2}{\sigma_{kij}^2}} \\
\text{subject to} & \underset{x,u,q}{\text{minimize}} & \displaystyle\sum_{k=0}^{n_S-1}\left(\sum_{i=1}^{n_{\mathcal{M}}}\gamma_i\phi_{\mathcal{M}}^{ki}(x_k(t_{k+1}),q,p)+\sum_{i=n_{\mathcal{M}}+1}^{n_{\mathcal{M}}+n_{\mathcal{L}}}\gamma_i\int_{t_k}^{t_{k+1}}\phi_{\mathcal{L}}^{ki}(x_k(t),u_k(t),q,p)\,\mathrm{dt}\right) \\
& \text{subject to} & \begin{array}{lll}
\dot{x}_k(t) & = f_k(x_k(t),u_k(t),q,p), & t\in[t_k,t_{k+1}],\ k=0,\dots,n_S-1 \\
x_{k+1}(t_{k+1}) & = b_k(x_k(t_{k+1}),q,p), & k=0,\dots,n_S-2 \\
0 & \leq c_k(x_k(t),u_k(t),q,p), & t\in[t_k,t_{k+1}],\ k=0,\dots,n_S-1 \\
0 & = r_k^{\mathrm{eq}}(x_k(\tau_{k,0}),\dots,x_k(\tau_{k,n_k}),q,p), & k=0,\dots,n_S-1 \\
0 & \leq r_k^{\mathrm{ieq}}(x_k(\tau_{k,0}),\dots,x_k(\tau_{k,n_k}),q,p), & k=0,\dots,n_S-1
\end{array} \\
& \multicolumn{2}{l}{\sum_{i=1}^{n_{\mathcal{M}}+n_{\mathcal{L}}}\gamma_i=1,\ \gamma\geq 0} \\
& \multicolumn{2}{l}{b^{\mathrm{lower}_p}\ \leq\ p\ \leq\ b^{\mathrm{upper}_p}.}
\end{array}
\tag{5.1}
$$

In the remainder of this chapter, we consider a one-stage formulation of the hierarchical dynamic optimization problem (5.1) for the sake of a clearer and simpler notation. The multi-stage formulation is extremely important in the implementation and for practical problems, but it does not significantly change the mathematical method and the problem structure described in the remainder of this chapter. However, for parts of the method derived in this chapter that are changed by multiple model stages, we discuss how the multi-stage setting affects the method and the corresponding problem structure.

In the remainder of this chapter, we consider the following hierarchical dynamic optimization problem with one model stage:

$$
\begin{array}{lll}
\underset{\substack{x,u,q\\ \gamma,p}}{\operatorname{minimize}} & \displaystyle\frac{1}{2}\sum_{i=0}^{n_m}\sum_{j=1}^{n_h}\frac{(h_j(x(t_i^m),q,p)-\eta_{ij})^2}{\sigma_{ij}^2} & \\
\text{subject to} & \underset{x,u,q}{\operatorname{minimize}} \quad \displaystyle\sum_{i=1}^{n_{\mathcal{M}}}\gamma_i\phi_{\mathcal{M}}^i(x(T),q,p) & \\
 & \begin{array}{llll}
\text{subject to} & \dot{x}(t) & = f(x(t),u(t),q,p), & t\in[t_0,T]\\
 & 0 & \leq c(x(t),u(t),q,p), & t\in[t_0,T]\\
 & 0 & = r^{\text{eq}}(x(t_0),\ldots,x(T),q,p) & \\
 & 0 & \leq r^{\text{ieq}}(x(t_0),\ldots,x(T),q,p) &
\end{array} & \\
 & \sum_{i=1}^{n_{\mathcal{M}}}\gamma_i=1,\ \gamma\geq 0 & \\
 & b^{\text{lower}_p}\ \leq\ p\ \leq\ b^{\text{upper}_p}. &
\end{array}
\tag{5.2}
$$

Apart from having just one model stage in (5.2), the lower-level objective in (5.2) does not include a Lagrange term (cf. Section 3.2) for the sake of a clearer presentation. However, nothing is lost by just considering Mayer terms since a Lagrange term can be written as a Mayer term by introducing an additional ordinary differential equation (ODE).

The variables in (5.1) and their dimensions are defined and explained at the beginning of Section 4.1. For problem (5.2), we set $n_S = 1$, skip the index k, the stage transition conditions and the Lagrange terms in the lower-level objective.

We now consider the control discretization. The discretization grid for the controls is given by

$$t_0 = t_0 < t_1 < \ldots < t_{n_T} = T \tag{5.3}$$

on $[t_0, T]$, where we denote $I_i := [t_i, t_{i+1}]$ for $i = 0, \ldots, n_T - 1$. The control function $u(t)$ is locally approximated by basis functions

$$\bar{u}(t)\big|_{I_i} = \xi_i(t, w_i) \tag{5.4}$$

with finite support, where $w_i \in \mathbb{R}^{n_l \cdot n_u}$, $i = 0, \ldots, n_T - 1$, and $w := \left(w_0^\intercal, \ldots, w_{n_T-1}^\intercal\right)^\intercal$. As described in Section 3.2, possible choices for the basis functions include, e.g., piecewise constant functions, piecewise linear functions, or piecewise cubic functions. The ODE in (5.2) now also depends on the parameter vector w: $\dot{x}(t) = f(x(t), \xi_i(t, w_i), q, p)$ for $t \in I_i$.

For the multiple shooting parameterization of the differential states in (5.2), we choose the grid

$$t_0 = t_0 < t_1 < \ldots < t_{n_T} = T, \tag{5.5}$$

where we assume that the multiple shooting grid (5.5) is the same as the control discretization grid (5.4) in order to obtain a special type of staircase structure in some blocks of the constraint Jacobian and a particular block-diagonal structure in certain blocks of the Hessian of the lower-level OCP. Another reason for choosing the same grid for the control discretization and the state parameterization is that it allows a more efficient computation of the sensitivities (cf. Section 3.1). The precise structure of the resulting NLP is further discussed in Section 5.4.

For the multiple shooting parameterization, we introduce new variables $s_0, \ldots, s_{n_T}$ with $s_i \in \mathbb{R}^{n_x}$, $i = 0, \ldots, n_T$, where s_i describes the initial value of the solution of the ODE in (5.2) on I_i. The solution of the initial value problem

$$\dot{x}(t) \;=\; f(x(t), \xi_i(t, w_i), q, p), \quad t \in I_i \tag{5.6a}$$

$$x(t_i) \;=\; s_i \tag{5.6b}$$

is denoted by $x(t; t_i, s_i, w_i, q, p)$. In order to guarantee a continuous solution of the ODE of the lower-level OCP of (5.2), we require the following n_T matching or closing conditions (cf. Section 3.2):

$$x(t_{i+1}; t_i, s_i, w_i, q, p) = s_{i+1}, \quad \forall i = 0, \dots, n_T - 1. \tag{5.7}$$

Please note that if we have a multi-stage problem of type (5.1), the matching condition at the stage transition from stage k to stage $k+1$ has to be modified using the stage transition condition:

$$b_k(x_k(t_{k+1}), q, p) = s_{k+1}, \quad \forall k = 0, \dots, n_S - 2. \tag{5.8}$$

One possibility to treat the path constraints $c(x(t), u(t), q, p) \geq 0$, $t \in [t_0, T]$, is to enforce them on the grid nodes only:

$$\begin{array}{lll} c(s_i, \xi_i(t_i, w_i), q, p) & \geq 0, & \forall i = 0, \dots, n_T - 1, \\ c(s_{n_T}, \xi_{n_T-1}(T, w_{n_T-1}), q, p) & \geq 0. & \end{array} \tag{5.9}$$

In practice, the quality of this approximation of the mixed control-state constraints often suffices. However, the approximation can be improved by choosing a finer discretization grid. A more sophisticated approach that guarantees that the path constraints are also satisfied between the grid nodes is provided in [PBS09]. We summarize the discretized inequality constraints and denote

$$\tilde{r}^{\text{ieq}}(s, w, q, p) := \begin{pmatrix} c(s_0, \xi_0(t_0, w_0), q, p) \\ \vdots \\ c(s_{n_T-1}, \xi_{n_T-1}(t_{n_T-1}, w_{n_T-1}), q, p) \\ c(s_{n_T}, \xi_{n_T-1}(T, w_{n_T-1}), q, p) \\ r^{\text{ieq}}(s_0, \dots, s_{n_T}, q, p) \end{pmatrix} \in \mathbb{R}^{n_{\tilde{r}^{\text{ieq}}}}, \tag{5.10}$$

with $n_{\tilde{r}^{\text{ieq}}} := n_c \cdot (n_T + 1) + n_{\text{ieq}}$. This leads to the following parameterized and discretized lower-level OCP of (5.2):

$$\begin{array}{llll} \underset{s,w,q}{\text{minimize}} & \tilde{\Phi} & := & \displaystyle\sum_{i=1}^{n_{\mathcal{M}}} \gamma_i \phi^i_{\mathcal{M}}(s_{n_T}, q, p) \\ \text{subject to} & 0 & = & x(t_{i+1}; t_i, s_i, w_i, q, p) - s_{i+1}, \ \forall i = 0, \dots, n_T - 1 \\ & 0 & = & r^{\text{eq}}(s_0, \dots, s_{n_T}, q, p) \\ & 0 & \leq & \tilde{r}^{\text{ieq}}(s, w, q, p). \end{array} \tag{5.11}$$

The parameterized and discretized hierarchical dynamic optimization problem (5.2) is then given by

$$\begin{array}{ll} \underset{\substack{s,w,q\\ \gamma,p}}{\text{minimize}} & \displaystyle\frac{1}{2} \sum_{i=0}^{n_m} \sum_{j=1}^{n_h} \frac{(h_j(x(t_i^m), q, p) - \eta_{ij})^2}{\sigma_{ij}^2} \\ \text{subject to} & \begin{array}{llll} \underset{s,w,q}{\text{minimize}} & \tilde{\Phi} & := & \displaystyle\sum_{i=1}^{n_{\mathcal{M}}} \gamma_i \phi^i_{\mathcal{M}}(s_{n_T}, q, p) \\ \text{subject to} & 0 & = & x(t_{i+1}; t_i, s_i, w_i, q, p) - s_{i+1}, \ \forall i = 0, \dots, n_T - 1 \\ & 0 & = & r^{\text{eq}}(s_0, \dots, s_{n_T}, q, p) \\ & 0 & \leq & \tilde{r}^{\text{ieq}}(s, w, q, p) \end{array} \\ & \sum_{i=1}^{n_{\mathcal{M}}} \gamma_i = 1, \ \gamma \geq 0 \\ & b^{\text{lower}_p} \leq p \leq b^{\text{upper}_p}, \end{array} \tag{5.12}$$

where the arguments of $\tilde{\Phi}$ are skipped. The model response $h_j(x(t_i^m), q, p)$ with $i = 0, \ldots, n_m,\ j = 1, \ldots, n_h$ depends on $x(t_i^m)$. We now briefly explain what $x(t_i^m)$ means in problem (5.12). Let t_i^m with $i = 0, \cdots, n_m$ be contained in the multiple shooting interval I_j, $j = 0, \ldots, n_T - 1$. Then $x(t_i^m)$ is the solution of the boundary value problem

$$\begin{aligned} \dot{x}(t) &= f(x(t), \xi_j(t, w_j), q, p), \quad t \in I_j && (5.13a) \\ x(t_j) &= s_j, && (5.13b) \end{aligned}$$

on I_j evaluated at t_i^m. Before we continue, we slightly reformulate problem (5.12). Therefore, we assume without loss of generality that the first n_{slacks} inequality constraints of $\tilde{r}^{\text{ieq}}(s, w, q, p) \geq 0$ are no simple bounds, and that the remaining constraints are simple bounds on s, w and q of the form

$$b^{\text{lower}} \leq \begin{pmatrix} s \\ w \\ q \end{pmatrix} \leq b^{\text{upper}}. \tag{5.14}$$

If the original constraint $\tilde{r}^{\text{ieq}}(s, w, q, p) \geq 0$ does not include bounds for each component of s, w and q, the corresponding component in b^{lower} or b^{upper} can be set to ±∞ (or a large negative and positive value in the numerical realization). For the first n_{slacks} constraints, we assume that they are transformed into equality constraints using slack variables. The reason for this assumption is related to the treatment of the complementarity constraint that arises when formulating KKT conditions of (5.11) and is discussed in Chapter 8. The new slack variables are added to the vector q:

$$\bar{q} := \left(q^{\intercal}, (q^s)^{\intercal}\right)^{\intercal} \tag{5.15}$$

with the first n_q entries being the original variables q, and the remaining n_{slacks} entries being the new slack variables q^s, for which we require $q^s \geq 0$. To summarize the equality constraints, we denote

$$\tilde{r}^{\text{eq}}(s, w, \bar{q}, p) := \begin{pmatrix} x(t_1; t_0, s_0, w_0, q, p) - s_1 \\ \vdots \\ x(t_{n_T}; t_{n_T-1}, s_{n_T-1}, w_{n_T-1}, q, p) - s_{n_T} \\ r^{\text{eq}}(s_0, \ldots, s_{n_T}, q, p) \\ \tilde{r_1}^{\text{ieq}}(s, w, q, p) - q_1^s \\ \vdots \\ \tilde{r}^{\text{ieq}}_{n_{\text{slacks}}}(s, w, q, p) - q^s_{n_{\text{slacks}}} \end{pmatrix}. \tag{5.16}$$

We furthermore denote the dimension of $\bar{q}$ by $n_{\bar{q}} := n_q + n_{\text{slacks}}$. The vector $\tilde{r}^{\text{eq}}(s, w, \bar{q}, p)$ of equality constraints is of size $n_{\tilde{r}^{\text{eq}}} := (n_x \cdot n_T) + n_{\text{eq}} + n_{\text{slacks}}$. We then obtain the following formulation of (5.12):

$$
\begin{array}{lllll}
\underset{\substack{s,w,\bar{q}\\ \gamma,p}}{\text{minimize}} & \multicolumn{4}{l}{\dfrac{1}{2}\displaystyle\sum_{i=0}^{n_m}\sum_{j=1}^{n_h}\frac{(h_j(x(t_i^m),q,p)-\eta_{ij})^2}{\sigma_{ij}^2}} \\
\text{subject to} & \underset{s,w,\bar{q}}{\text{minimize}} & \tilde{\Phi} & := & \displaystyle\sum_{i=1}^{n_{\mathcal{M}}}\gamma_i\phi_{\mathcal{M}}^i(s_{n_T},q,p) \\
 & \text{subject to} & 0 & = & \tilde{r}^{\text{eq}}(s,w,\bar{q},p) \\
 & & b^{\text{lower}} & \leq & \begin{pmatrix} s\\ w\\ q\end{pmatrix} \leq b^{\text{upper}} \\
 & & 0 & \leq & q^s \\
 & \multicolumn{4}{l}{\sum_{i=1}^{n_{\mathcal{M}}}\gamma_i = 1,\ \gamma\geq 0} \\
 & \multicolumn{4}{l}{b^{\text{lower}_p} \leq p \leq b^{\text{upper}_p}.}
\end{array}
\tag{5.17}
$$

We now continue with formulating first-order optimality conditions for the lower-level OCP of (5.17).

5.3. First-order necessary optimality conditions

Let $y := (s^\intercal, w^\intercal, \bar{q}^\intercal)^\intercal \in \mathbb{R}^{n_y}$ with the dimensions $n_y := n_s + n_w + n_{\bar{q}}$, $n_s := n_x \cdot (n_T + 1)$ and $n_w = n_u \cdot n_l \cdot n_T$ denote the vector of unknowns of the lower-level problem of (5.17) and let

$$
\mathcal{L}(y,\lambda,\mu) = \tilde{\Phi} - \left(C^{\text{eq}}\right)^\intercal \lambda - \left(C^{\text{ieq}}\right)^\intercal \mu \tag{5.18}
$$

be the Lagrangian of the lower-level problem of (5.17) in a compact form. We note that in (5.18), we skip the arguments of $\tilde{\Phi}$, C^{eq} and C^{ieq} for a compact presentation. This is also done for further variables in the following if necessary. We now explain the composition and the structure of the Lagrangian (5.18) in detail. The variables λ and μ describe the Lagrange multipliers of the equality and the inequality constraints of the lower-level problem of (5.17), respectively. λ is of dimension $n_\lambda := n_{\tilde{r}^{\text{eq}}}$ and μ is of dimension $n_\mu := n_y + n_s + n_w + n_q$. We furthermore require the Lagrange multipliers μ corresponding to the inequality constraints to be nonnegative:

$$
\mu \geq 0. \tag{5.19}
$$

The lower-level objective is denoted by $\tilde{\Phi}$, the equality constraints in (5.11) are summarized by

$$
C^{\text{eq}} := \tilde{r}^{\text{eq}}(s,w,\bar{q},p) = \begin{pmatrix} x(t_1;t_0,s_0,w_0,q,p) - s_1 \\ \vdots \\ x(t_{n_T};t_{n_T-1},s_{n_T-1},w_{n_T-1},q,p) - s_{n_T} \\ r^{\text{eq}}(s_0,\ldots,s_{n_T},q,p) \\ \tilde{r_1}^{\text{ieq}}(s,w,q,p) - q_1^s \\ \vdots \\ \tilde{r}^{\text{ieq}}_{n_{\text{slacks}}}(s,w,q,p) - q^s_{n_{\text{slacks}}} \end{pmatrix}, \tag{5.20}
$$

and the inequality constraints are denoted by

$$C^{\text{ieq}} := \begin{pmatrix} \begin{pmatrix} s \\ w \\ q \end{pmatrix} - b^{\text{lower}} \\ b^{\text{upper}} - \begin{pmatrix} s \\ w \\ q \end{pmatrix} \\ q^s_1 \\ \vdots \\ q^s_{n_{\text{slacks}}} \end{pmatrix}. \tag{5.21}$$

We now consider the derivative of the lower-level objective $\tilde{\Phi}$. Therefore, we denote

$$M_{s_{n_T}} := \sum_{j=1}^{n_{\mathcal{M}}} \gamma_j \left(\frac{\partial \phi^j_{\mathcal{M}}(s_{n_T}, q, p)}{\partial s_{n_T}} \right) \in \mathbb{R}^{n_x}, \tag{5.22}$$

which is the derivative of $\tilde{\Phi}$ with respect to s_{n_T}. The derivative of $\tilde{\Phi}$ with respect to q is given by

$$M_{q_i} \;\; := \;\; \sum_{j=1}^{n_{\mathcal{M}}} \gamma_j \left(\frac{\partial \Phi^j_{\mathcal{M}}(s_{n_T}, q, p)}{\partial q_i} \right) \in \mathbb{R}, \quad \forall i = 1, \dots, n_q, \tag{5.23}$$

and the derivative of $\tilde{\Phi}$ with respect to q^s is zero. This leads to the following gradient of $\tilde{\Phi}$

$$\nabla_y \tilde{\Phi} = \Big(0, \dots, 0, M_{s_{n_T}}^\intercal, \;\; 0, \dots, 0, \;\; M_{q_1}, \dots, M_{q_{n_q}}, \;\; 0, \dots, 0\Big)^\intercal. \tag{5.24}$$

If we just consider optimization criteria of Mayer-type in the lower-level objective of (5.2), $\nabla_y \tilde{\Phi}$ does neither depend on the first n_T variables $s_0, \dots, s_{n_T-1}$ of the multiple shooting parameterization, nor on the variables $w_0, \dots, w_{n_T-1}$ approximating the control function. Optimization criteria of Lagrange-type in the lower-level OCP in (5.2) might lead to a dense gradient of $\tilde{\Phi}$ with respect to (s, w, q).

We now consider the derivative of the discretized equality constraints C^{eq} with respect to the variables y. Therefore, we denote

$$G_{s_i} := \frac{\partial x(t_{i+1}; t_i, s_i, w_i, q, p)}{\partial s_i} \in \left(\mathbb{R}^{n_x} \times \mathbb{R}^{n_x}\right), \quad \forall i = 0, \dots, n_T - 1 \tag{5.25}$$

describing the derivative (or sensitivity) of the i-th matching condition $x(t_{i+1}; t_i, s_i, w_i, q, p) - s_{i+1}$ with respect to s_i. We furthermore have

$$\frac{\partial \left(x(t_{i+1}; t_i, s_i, w_i, q, p) - s_{i+1} \right)}{\partial s_{i+1}} = -\mathbb{I}_{n_x}, \quad \forall i = 0, \dots, n_T - 1, \tag{5.26}$$

where $\mathbb{I}_{n_x}$ denotes the identity matrix of size $n_x \times n_x$. If we have a multi-stage setting as in (5.1), we additionally need the derivative of (5.8) at each stage transition.

We furthermore define

$$G_{w_i} := \frac{\mathrm{d}x(t_{i+1}; t_i, s_i, w_i, q, p)}{\mathrm{d}w_i} \in \left(\mathbb{R}^{n_x} \times \mathbb{R}^{n_l \cdot n_u}\right), \quad \forall i = 0, \dots, n_T - 1 \tag{5.27}$$

to be the derivative of the matching conditions with respect to the control discretization variables w_i, $i = 0, \dots, n_T - 1$, and

$$G^i_{q_j} := \frac{\partial x(t_{i+1}; t_i, s_i, w_i, q, p)}{\partial q_j} \in \mathbb{R}^{n_x}, \quad \forall j = 1, \dots, n_q \text{ and } i = 0, \dots, n_T - 1 \tag{5.28}$$

to be the derivative of the matching conditions with respect to time-independent control variables q_i, $i = 1, \dots, n_q$. The derivative of the matching conditions with respect to the slack variables q^s is zero. We now consider the derivatives of the equality multi-point boundary conditions

$$R^{\text{eq}}_{s_i} := \frac{\partial r^{\text{eq}}(s_0, \dots, s_{n_T}, q, p)}{\partial s_i} \in \mathbb{R}^{n_{\text{eq}}} \times \mathbb{R}^{n_x}, \qquad \forall i = 0, \dots, n_T \tag{5.29}$$

$$R^{\text{eq}}_{w_i} := 0 \in \mathbb{R}^{n_{\text{eq}}} \times \mathbb{R}^{n_l \cdot n_u}, \qquad \forall i = 0, \dots, n_T - 1 \tag{5.30}$$

$$R^{\text{eq}}_{q_i} := \frac{\partial r^{\text{eq}}(s_0, \dots, s_{n_T}, q, p)}{\partial q_i} \in \mathbb{R}^{n_{\text{eq}}}, \qquad \forall i = 1, \dots, n_q \tag{5.31}$$

$$R^{\text{eq}}_{q^s_i} := 0 \in \mathbb{R}^{n_{\text{eq}}}, \qquad \forall i = 1, \dots, n_{\text{slacks}} \tag{5.32}$$

and of the multi-point inequality constraints which are transformed into equality constraints using q^s

$$R^{\text{ieq}}_{s_i} := \frac{\partial \tilde{r}^{\text{ieq}}(s, w, q, p)}{\partial s_i} \in \mathbb{R}^{n_{\tilde{r}^{\text{ieq}}}} \times \mathbb{R}^{n_x}, \qquad \forall i = 0, \dots, n_T \tag{5.33}$$

$$R^{\text{ieq}}_{w_i} := \frac{\partial \tilde{r}^{\text{ieq}}(s, w, q, p)}{\partial w_i} \in \mathbb{R}^{n_{\tilde{r}^{\text{ieq}}}} \times \mathbb{R}^{n_l \cdot n_u}, \qquad \forall i = 0, \dots, n_T - 1 \tag{5.34}$$

$$R^{\text{ieq}}_{q_i} := \frac{\partial \tilde{r}^{\text{ieq}}(s, w, q, p)}{\partial q_i} \in \mathbb{R}^{n_{\tilde{r}^{\text{ieq}}}}, \qquad \forall i = 1, \dots, n_q \tag{5.35}$$

$$R^{\text{ieq}}_{q^s_i} := 0 \in \mathbb{R}^{n_{\tilde{r}^{\text{ieq}}}}, \qquad \forall i = 1, \dots, n_{\text{slacks}}. \tag{5.36}$$

The derivative of the equality constraints C^{eq} of the lower-level OCP of (5.17) with respect to $y = (s^\intercal, w^\intercal, \bar{q}^\intercal)^\intercal$ is of the following structure:

$$\nabla_y C^{\text{eq}} =$$

$$\left(\begin{array}{cccc|cc|cccc}
G_{s_0} & -\mathbb{I}_{n_x} & & & G_{w_0} & & G^0_{q_1} & \dots & G^0_{q_{n_q}} & \\
& \ddots & \ddots & & & \ddots & \vdots & & \vdots & \\
& & G_{s_{n_T-1}} & -\mathbb{I}_{n_x} & & G_{w_{n_T-1}} & G^{n_T-1}_{q_1} & \dots & G^{n_T-1}_{q_{n_q}} & \\
\hline
R^{\text{eq}}_{s_0} & \dots & R^{\text{eq}}_{s_{n_T-1}} & R^{\text{eq}}_{s_{n_T}} & & & R^{\text{eq}}_{q_1} & \dots & R^{\text{eq}}_{q_{n_q}} & \\
\hline
R^{\text{ieq}}_{s_0} & \dots & R^{\text{ieq}}_{s_{n_T-1}} & R^{\text{ieq}}_{s_{n_T}} & R^{\text{ieq}}_{w_0} \dots & R^{\text{ieq}}_{w_{n_T-1}} & R^{\text{ieq}}_{q_1} & \dots & R^{\text{ieq}}_{q_{n_q}} & -\mathbb{I}_{n_{\text{slacks}}}
\end{array}\right), \tag{5.37}$$

where the rows of $\nabla_y C^{\text{eq}}$ correspond to the equality constraints in C^{eq} described in (5.20), and the columns of $\nabla_y C^{\text{eq}}$ correspond to the variables

$$y = \left(s_0^\intercal, \dots, s_{n_T}^\intercal \mid w_0^\intercal, \dots, w_{n_T-1}^\intercal \mid q_1, \dots, q_{n_q},\ q^s_1, \dots, q^s_{n_{\text{slacks}}}\right)^\intercal. \tag{5.38}$$

Some blocks of zeros in (5.37) and in the remainder of this chapter are skipped to highlight the structure of the matrix. We refer to the blocks in $\nabla_y C^{\text{eq}}$ as follows:

$$\nabla_y C^{\text{eq}} =: \left(\begin{array}{c|c|c} B1 & B2 & B3 \\ \hline B4 & B5 & B6 \\ \hline B7 & B8 & B9 \end{array}\right). \tag{5.39}$$

Block $B1$ in $\nabla_y C^{\text{eq}}$ has a staircase structure coming from the fact hat the derivatives of the matching conditions decouple over the multiple shooting intervals, i.e.

$$\frac{\partial\left(x(t_{i+1}; t_i, s_i, w_i, q, p) - s_{i+1}\right)}{\partial s_j} = 0 \quad \text{for} \;\; j \neq i \quad \text{and} \quad j \neq i+1. \tag{5.40}$$

Block $B2$ in $\nabla_y C^{\text{eq}}$ has a block-diagonal structure, which comes from the choice of the same grid for the multiple shooting parameterization and the control discretization, and from the fact that the control function is locally (on each interval I_i, $i = 0, \ldots, n_T - 1$) approximated by basis functions depending on $w_i \in \mathbb{R}^{n_l \cdot n_u}$.

The structure of Block $B3$ originates from the slack variables q^s. We know that the matching conditions do not depend on q^s, and therefore, it is

$$\frac{\partial x(t_{i+1}; t_i, s_i, w_i, q, p)}{\partial q^s_j} = 0 \quad \text{with} \quad j = 1, \ldots, n_{\text{slacks}} \quad \text{and} \quad i = 0, \ldots, n_T - 1. \tag{5.41}$$

Block $B5$ is zero since r^{eq} does not depend on the controls. The Blocks $B4$ and $B6$ seem to be almost dense on first glance, but in many practical applications, they do have a particular structure. If this particular structure is present, we intend to exploit it in the algorithm derived in this thesis for a faster and more efficient calculation of (5.37). The special structure of the equality constraints $r^{\text{eq}}(s_0, \ldots, s_{n_T}, q, p)$ is the following

$$r^{\text{eq}}(s_0, \ldots, s_{n_T}, q, p) = \begin{pmatrix} \hat{r}^{\text{eq}}_0(s_0, q, p) \\ \vdots \\ \hat{r}^{\text{eq}}_{n_T}(s_{n_T}, q, p) \\ \hat{r}^{\text{eq}}_{\text{first}}(s_0, q, p) + \hat{r}^{\text{eq}}_{\text{last}}(s_{n_T}, q, p) \end{pmatrix}, \tag{5.42}$$

where we assume that the multi-point equality constraints are either decoupled over the multiple shooting intervals, i.e. each function $\hat{r}^{\text{eq}}_i(s_i, q, p)$ just depends on one s_i, $i = 0, \ldots, n_T$, or linearly coupled, where we just consider the case of a linear coupling between the first and the last multiple shooting interval:

$$\hat{r}^{\text{eq}}_{\text{first}}(s_0, q, p) + \hat{r}^{\text{eq}}_{\text{last}}(s_{n_T}, q, p). \tag{5.43}$$

The often observed special structure of the constraints in (5.42) appears, e.g., when solving hierarchical dynamic optimization problems for identifying human gait models (cf. Chapters 12 and 15). We consider linearly coupled conditions between the first and the last multiple shooting interval as described in (5.43) since this type of constraint often appears as periodicity constraint when identifying multi-body-system-based optimal control models. In the implementation of this method, it is not required to have constraints of the special form shown in (5.42). If the structure (5.42) is given for a particular problem, the special structure is exploited, but multi-point constraints of a more general type can of course also be treated. Given the structure (5.42), Block $B4$ is of the following form:

$$\begin{pmatrix} \hat{R}^{\text{eq}}_{s_0} & & \\ & \ddots & \\ & & \hat{R}^{\text{eq}}_{s_{n_T}} \\ \hat{R}^{\text{eq}_{\text{first}}}_{s_0} & & \hat{R}^{\text{eq}_{\text{last}}}_{s_{n_T}} \end{pmatrix}, \tag{5.44}$$

with

$$\begin{array}{lcll} \hat{R}^{\text{eq}}_{s_i} & := & \frac{\partial \hat{r}^{\text{eq}}_i(s_i,q,p)}{\partial s_i} & \text{with} \quad i = 0,\ldots,n_T \\ \hat{R}^{\text{eq}_{\text{first}}}_{s_0} & := & \frac{\partial \hat{r}^{\text{eq}}_{\text{first}}(s_0,q,p)}{\partial s_0} & \\ \hat{R}^{\text{eq}_{\text{last}}}_{s_{n_T}} & := & \frac{\partial \hat{r}^{\text{eq}}_{\text{last}}(s_{n_T},q,p)}{\partial s_{n_T}} & . \end{array} \tag{5.45}$$

In Block $B6$ we can exploit the fact that we use n_{slacks} slack variables to transform inequality constraints that are no simple bounds into multi-point equality constraints. The structure of $B6$ is then as follows:

$$\begin{pmatrix} \hat{R}^{\text{eq}_0}_{q_1} & \ldots & \hat{R}^{\text{eq}_0}_{q_{n_q}} & \\ \vdots & & \vdots & \\ \hat{R}^{\text{eq}_{n_T}}_{q_1} & \ldots & \hat{R}^{\text{eq}_{n_T}}_{q_{n_q}} & \\ \hat{R}^{\text{eq}_{\text{coupled}}}_{q_1} & \ldots & \hat{R}^{\text{eq}_{\text{coupled}}}_{q_{n_q}} & \end{pmatrix}, \tag{5.46}$$

with

$$\begin{array}{lcll} \hat{R}^{\text{eq}_j}_{q_i} & := & \frac{\partial \hat{r}^{\text{eq}}_j(s_j,q,p)}{\partial q_i} & \text{with} \;\; j = 0,\ldots,n_T \;\; \text{and} \;\; i = 1,\ldots,n_q \\ \hat{R}^{\text{eq}_{\text{coupled}}}_{q_i} & := & \frac{\partial \hat{r}^{\text{eq}}_{\text{first}}(s_0,q,p)+\hat{r}^{\text{eq}}_{\text{last}}(s_{n_T},q,p)}{\partial q_i} & \text{with} \;\; i = 1,\ldots,n_q. \end{array} \tag{5.47}$$

The blocks corresponding to q^s in (5.46) are zero. A similar structure as in (5.42) often also appears in the first n_{slacks} components of $\tilde{r}^{\text{ieq}}(s,w,q,p)$:

$$\begin{pmatrix} \hat{r}^{\text{ieq}}_0(s_0,w_0,q,p) \\ \vdots \\ \hat{r}^{\text{ieq}}_{n_T-1}(s_{n_T-1},w_{n_T-1},q,p) \\ \hat{r}^{\text{ieq}}_{n_T}(s_{n_T},w_{n_T-1},q,p) \end{pmatrix}, \tag{5.48}$$

where $\hat{r}^{\text{ieq}}_i(s_i,w_i,q,p)$ with $i = 0,\ldots,n_T-1$ and $\hat{r}^{\text{ieq}}_{n_T}(s_{n_T},w_{n_T-1},q,p)$ just depend on one s_i and w_i, i.e. the constraints are decoupled over the multiple shooting nodes. Having the previously mentioned applications in mind, a constraint with a linear coupling of the first and the last multiple shooting node as in (5.42) is not necessary for the inequality constraints (5.48). Block $B7$ is of the following form if structure (5.48) is given:

$$\begin{pmatrix} \hat{R}^{\text{ieq}}_{s_0} & & \\ & \ddots & \\ & & \hat{R}^{\text{ieq}}_{s_{n_T}} \end{pmatrix}, \tag{5.49}$$

with

$$\begin{array}{lcll} \hat{R}^{\text{ieq}}_{s_i} & := & \frac{\partial \hat{r}^{\text{ieq}}_i(s_i,w_i,q,p)}{\partial s_i} & \text{with} \quad i=0,\dots,n_T-1 \\ \hat{R}^{\text{ieq}}_{s_{n_T}} & := & \frac{\partial \hat{r}^{\text{ieq}}_{n_T}(s_{n_T},w_{n_T-1},q,p)}{\partial s_{n_T}} & \end{array} \tag{5.50}$$

and Block $B8$ is of form

$$\begin{pmatrix} \hat{R}^{\text{ieq}}_{w_0} & & \\ & \ddots & \\ & & \hat{R}^{\text{ieq}}_{w_{n_T-1}} \\ & & \hat{R}^{\text{last}}_{w_{n_T-1}} \end{pmatrix}, \tag{5.51}$$

with

$$\begin{array}{lcll} \hat{R}^{\text{ieq}}_{w_i} & := & \frac{\partial \hat{r}^{\text{ieq}}_i(s_i,w_i,q,p)}{\partial w_i} & \text{with} \quad i=0,\dots,n_T-1 \\ \hat{R}^{\text{last}}_{w_{n_T-1}} & := & \frac{\partial \hat{r}^{\text{ieq}}_{n_T}(s_{n_T},w_{n_T-1},q,p)}{\partial w_{n_T-1}}. & \end{array} \tag{5.52}$$

For Block $B9$, we get

$$\begin{pmatrix} \hat{R}^{\text{ieq}_0}_{q_1} & \dots & \hat{R}^{\text{ieq}_0}_{q_{n_q}} & \\ \vdots & & \vdots & -\mathbb{I}_{n_{\text{slacks}}} \\ \hat{R}^{\text{ieq}_{n_T}}_{q_1} & \dots & \hat{R}^{\text{ieq}_{n_T}}_{q_{n_q}} & \end{pmatrix}, \tag{5.53}$$

with

$$\begin{array}{lcll} \hat{R}^{\text{ieq}_j}_{q_i} & := & \frac{\partial \hat{r}^{\text{ieq}}_j(s_j,w_j,q,p)}{\partial q_i} & \text{with} \quad j=0,\dots,n_T-1 \quad \text{and} \quad i=1,\dots,n_q \\ \hat{R}^{\text{ieq}_{n_T}}_{q_i} & := & \frac{\partial \hat{r}^{\text{ieq}}_{n_T}(s_{n_T},w_{n_T-1},q,p)}{\partial q_i} & \text{with} \quad i=1,\dots,n_q \end{array} \tag{5.54}$$

and the identity matrix $\mathbb{I}_{n_{\text{slacks}}}$ of size $n_{\text{slacks}} \times n_{\text{slacks}}$, which is due to the slack variables in (5.48). If the discretized constraints in (5.17) are of type (5.42) and (5.48), we can present the structure of (5.37) in more detail:

$$\nabla_y C^{\text{eq}} =$$

$$\left(\begin{array}{cccc|ccc|cccc} G_{s_0} & -\mathbb{I}_{n_x} & & & G_{w_0} & & & G^0_{q_1} & \dots & G^0_{n_q} & \\ & \ddots & \ddots & & & \ddots & & \vdots & & \vdots & \\ & & G_{s_{n_T-1}} & -\mathbb{I}_{n_x} & & & G_{w_{n_T-1}} & G^{n_T-1}_{q_1} & \dots & G^{n_T-1}_{n_q} & \\ \hline \hat{R}^{\text{eq}}_{s_0} & & & & & & & \hat{R}^{\text{eq}_0}_{q_1} & \dots & \hat{R}^{\text{eq}_0}_{q_{n_q}} & \\ & \ddots & & & & & & \vdots & & \vdots & \\ & & \ddots & & & & & \vdots & & \vdots & \\ & & & \hat{R}^{\text{eq}}_{s_{n_T}} & & & & \hat{R}^{\text{eq}_{n_T}}_{q_1} & \dots & \hat{R}^{\text{eq}_{n_T}}_{q_{n_q}} & \\ \hat{R}^{\text{eq}_{\text{first}}}_{s_0} & & & \hat{R}^{\text{eq}_{\text{last}}}_{s_{n_T}} & & & & \hat{R}^{\text{eq}_{\text{coupled}}}_{q_1} & \dots & \hat{R}^{\text{eq}_{\text{coupled}}}_{q_{n_q}} & \\ \hline \hat{R}^{\text{ieq}}_{s_0} & & & & \hat{R}^{\text{ieq}}_{w_0} & & & \hat{R}^{\text{ieq}_0}_{q_1} & \dots & \hat{R}^{\text{ieq}_0}_{q_{n_q}} & \\ & \ddots & & & & \ddots & & \vdots & & \vdots & -\mathbb{I}_{n_{\text{slacks}}} \\ & & \ddots & & & & \hat{R}^{\text{ieq}}_{w_{n_T-1}} & \vdots & & \vdots & \\ & & & \hat{R}^{\text{ieq}}_{s_{n_T}} & & & \hat{R}^{\text{last}}_{w_{n_T-1}} & \hat{R}^{\text{ieq}_{n_T}}_{q_1} & \dots & \hat{R}^{\text{ieq}_{n_T}}_{q_{n_q}} & \end{array}\right). \tag{5.55}$$

Further details of the structure and the size of the blocks of (5.55) for practical problems can be found in the next section.

In the following derivative of the inequality constraints (5.21) (which just includes bounds on s, w and $\bar{q}$) the rows correspond to the constraints in (5.21) and the columns correspond to the variables

$$y = \left(s^{\mathsf{T}} \mid w^{\mathsf{T}} \mid q^{\mathsf{T}} \ \left(q^s\right)^{\mathsf{T}}\right)^{\mathsf{T}}, \tag{5.56}$$

and obviously the structure is the following:

$$\nabla_y C^{\text{ieq}} = \left(\begin{array}{c|c|cc}
\mathbb{I}_{n_s} & & & \\ \hline
 & \mathbb{I}_{n_w} & & \\ \hline
 & & \mathbb{I}_{n_q} & \\ \hline
-\mathbb{I}_{n_s} & & & \\ \hline
 & -\mathbb{I}_{n_w} & & \\ \hline
 & & -\mathbb{I}_{n_q} & \\ \hline
 & & & \mathbb{I}_{n_{\text{slacks}}}
\end{array}\right) \tag{5.57}$$

with $n_s = n_x \cdot (n_T + 1)$ and $n_w = n_u \cdot n_l \cdot n_T$.

The gradient of the Lagrangian (5.18) of the lower-level problem of (5.17) is then given by

$$\nabla_y \mathcal{L}(y, \lambda, \mu) = \nabla_y \tilde{\Phi} - \left(\nabla_y C^{\text{eq}}\right)^{\mathsf{T}} \lambda - \left(\nabla_y C^{\text{ieq}}\right)^{\mathsf{T}} \mu, \tag{5.58}$$

where $\nabla_y \tilde{\Phi}$ is defined in (5.24), $\nabla_y C^{\text{ieq}}$ is defined in (5.57) and $\nabla_y C^{\text{eq}}$ is given in (5.37) for the general case, and if C^{eq} is of type (5.42) and (5.48), then the detailed structure of $\nabla_y C^{\text{eq}}$ is provided in (5.55).

Before we continue with the remaining KKT conditions of the lower-level problem of (5.17), we have a closer look at how to exploit the structure of (5.55) and (5.57), and how to compute (5.58) in an efficient way. We now just consider problems of type (5.17) with constraints that lead to the structure (5.42) and (5.48) since this is mostly the case in practice. We need the product $\nabla_y \left(C^{\text{eq}}\right)^{\mathsf{T}} \cdot \lambda$ to compute the gradient of the Lagrangian denoted in (5.58). We now exploit the structure of (5.55) when computing $\nabla_y \left(C^{\text{eq}}\right)^{\mathsf{T}} \cdot \lambda$

and just multiply elements that might be nonzero:

$$
\begin{gathered}
\left(\nabla_y C^{\text{eq}}\right)^{\intercal} \cdot \lambda = \\
\left(\begin{array}{lllll}
 & G_{s_0}^{\intercal}\lambda_{s_0}^{G} & +\hat{R}_{s_0}^{\text{eq}\,\intercal}\lambda_{s_0}^{\hat{R}^{\text{eq}}} & +\hat{R}_{s_0}^{\text{eq}_{\text{first}}\,\intercal}\lambda^{\hat{R}^{\text{eq}}_{\text{coupled}}} & +\hat{R}_{s_0}^{\text{ieq}\,\intercal}\lambda_{s_0}^{\hat{R}^{\text{ieq}}} \\
-\mathbb{I}_{n_x}\lambda_{s_0}^{G} & +G_{s_1}^{\intercal}\lambda_{s_1}^{G} & +\hat{R}_{s_1}^{\text{eq}\,\intercal}\lambda_{s_1}^{\hat{R}^{\text{eq}}} & & +\hat{R}_{s_1}^{\text{ieq}\,\intercal}\lambda_{s_1}^{\hat{R}^{\text{ieq}}} \\
\vdots & \vdots & \vdots & & \vdots \\
-\mathbb{I}_{n_x}\lambda_{s_{n_T-2}}^{G} & +G_{s_{n_T-1}}^{\intercal}\lambda_{s_{n_T-1}}^{G} & +\hat{R}_{s_{n_T-1}}^{\text{eq}}{}^{\intercal}\lambda_{s_{n_T-1}}^{\hat{R}^{\text{eq}}} & & +\hat{R}_{s_{n_T-1}}^{\text{ieq}}{}^{\intercal}\lambda_{s_{n_T-1}}^{\hat{R}^{\text{ieq}}} \\
-\mathbb{I}_{n_x}\lambda_{s_{n_T-1}}^{G} & & +\hat{R}_{s_{n_T}}^{\text{eq}}{}^{\intercal}\lambda_{s_{n_T}}^{\hat{R}^{\text{eq}}} & +\hat{R}_{s_{n_T}}^{\text{eq}_{\text{last}}\,\intercal}\lambda^{\hat{R}^{\text{eq}}_{\text{coupled}}} & +\hat{R}_{s_{n_T}}^{\text{ieq}}{}^{\intercal}\lambda_{s_{n_T}}^{\hat{R}^{\text{ieq}}} \\
\hline
G_{w_0}^{\intercal}\lambda_{s_0}^{G} & & & & +\hat{R}_{w_0}^{\text{ieq}\,\intercal}\lambda_{s_0}^{\hat{R}^{\text{ieq}}} \\
G_{w_1}^{\intercal}\lambda_{s_1}^{G} & & & & +\hat{R}_{w_1}^{\text{ieq}\,\intercal}\lambda_{s_1}^{\hat{R}^{\text{ieq}}} \\
\vdots & & & & \vdots \\
G_{w_{n_T-2}}^{\intercal}\lambda_{s_{n_T-2}}^{G} & & & & +\hat{R}_{w_{n_T-2}}^{\text{ieq}}{}^{\intercal}\lambda_{s_{n_T-2}}^{\hat{R}^{\text{ieq}}} \\
G_{w_{n_T-1}}^{\intercal}\lambda_{s_{n_T-1}}^{G} & & & +\hat{R}_{w_{n_T-1}}^{\text{last}}{}^{\intercal}\lambda_{s_{n_T}}^{\hat{R}^{\text{ieq}}} & +\hat{R}_{w_{n_T-1}}^{\text{ieq}}{}^{\intercal}\lambda_{s_{n_T-1}}^{\hat{R}^{\text{ieq}}} \\
\hline
\sum\limits_{i=0}^{n_T-1} G_{q_1}^{i}{}^{\intercal}\lambda_{s_i}^{G} & & +\sum\limits_{i=0}^{n_T}\hat{R}_{q_1}^{\text{eq}_i\,\intercal}\lambda_{s_i}^{\hat{R}^{\text{eq}}} & +\hat{R}_{q_1}^{\text{eq}_{\text{coupled}}\,\intercal}\lambda^{\hat{R}^{\text{eq}}_{\text{coupled}}} & +\sum\limits_{i=0}^{n_T}\hat{R}_{q_1}^{\text{ieq}_i\,\intercal}\lambda_{s_i}^{\hat{R}^{\text{ieq}}} \\
\sum\limits_{i=0}^{n_T-1} G_{q_2}^{i}{}^{\intercal}\lambda_{s_i}^{G} & & +\sum\limits_{i=0}^{n_T}\hat{R}_{q_2}^{\text{eq}_i\,\intercal}\lambda_{s_i}^{\hat{R}^{\text{eq}}} & +\hat{R}_{q_2}^{\text{eq}_{\text{coupled}}\,\intercal}\lambda^{\hat{R}^{\text{eq}}_{\text{coupled}}} & +\sum\limits_{i=0}^{n_T}\hat{R}_{q_2}^{\text{ieq}_i\,\intercal}\lambda_{s_i}^{\hat{R}^{\text{ieq}}} \\
\vdots & & \vdots & & \vdots \\
\sum\limits_{i=0}^{n_T-1} G_{q_{n_q-1}}^{i}{}^{\intercal}\lambda_{s_i}^{G} & & +\sum\limits_{i=0}^{n_T}\hat{R}_{q_{n_q-1}}^{\text{eq}_i}{}^{\intercal}\lambda_{s_i}^{\hat{R}^{\text{eq}}} & +\hat{R}_{q_{n_q-1}}^{\text{eq}_{\text{coupled}}\,\intercal}\lambda^{\hat{R}^{\text{eq}}_{\text{coupled}}} & +\sum\limits_{i=0}^{n_T}\hat{R}_{q_{n_q-1}}^{\text{ieq}_i}{}^{\intercal}\lambda_{s_i}^{\hat{R}^{\text{ieq}}} \\
\sum\limits_{i=0}^{n_T-1} G_{q_{n_q}}^{i}{}^{\intercal}\lambda_{s_i}^{G} & & +\sum\limits_{i=0}^{n_T}\hat{R}_{q_{n_q}}^{\text{eq}_i\,\intercal}\lambda_{s_i}^{\hat{R}^{\text{eq}}} & +\hat{R}_{q_{n_q}}^{\text{eq}_{\text{coupled}}\,\intercal}\lambda^{\hat{R}^{\text{eq}}_{\text{coupled}}} & +\sum\limits_{i=0}^{n_T}\hat{R}_{q_{n_q}}^{\text{ieq}_i\,\intercal}\lambda_{s_i}^{\hat{R}^{\text{ieq}}} \\
\hline
 & & & & -\lambda_{s_0}^{\hat{R}^{\text{ieq}}} \\
 & & & & \vdots \\
 & & & & -\lambda_{s_{n_T}}^{\hat{R}^{\text{ieq}}}
\end{array}\right)
\end{gathered}
\tag{5.59}
$$

with

$$
\lambda = \Bigg(\left(\lambda_{s_0}^{G}\right)^{\intercal}, \ldots, \left(\lambda_{s_{n_T-1}}^{G}\right)^{\intercal} \mid \left(\lambda_{s_0}^{\hat{R}^{\text{eq}}}\right)^{\intercal}, \ldots, \left(\lambda_{s_{n_T}}^{\hat{R}^{\text{eq}}}\right)^{\intercal}, \tag{5.60}
$$

$$
\left(\lambda^{\hat{R}^{\text{eq}}_{\text{coupled}}}\right)^{\intercal} \mid \left(\lambda_{s_0}^{\hat{R}^{\text{ieq}}}\right)^{\intercal}, \ldots, \left(\lambda_{s_{n_T}}^{\hat{R}^{\text{ieq}}}\right)^{\intercal} \Bigg)^{\intercal}. \tag{5.61}
$$

Furthermore, to compute (5.58), we need the product of the derivative of the inequality constraints (5.57) with the Lagrange multiplier μ:

$$
\left(\nabla_y C^{\text{ieq}}\right)^{\intercal} \cdot \mu \tag{5.62}
$$

with

$$\begin{aligned}\mu \quad = \quad & \Big(\mu_{s_0}^{\text{lower}^\intercal}, \dots, \mu_{s_{n_T}}^{\text{lower}^\intercal} \mid \mu_{w_0}^{\text{lower}^\intercal}, \dots, \mu_{w_{n_T-1}}^{\text{lower}^\intercal} \mid \mu_{q_1}^{\text{lower}}, \dots, \mu_{q_{n_q}}^{\text{lower}} \mid \\ & \mu_{s_0}^{\text{upper}^\intercal}, \dots, \mu_{s_{n_T}}^{\text{upper}^\intercal} \mid \mu_{w_0}^{\text{upper}^\intercal}, \dots, \mu_{w_{n_T-1}}^{\text{upper}^\intercal} \mid \mu_{q_1}^{\text{upper}}, \dots, \mu_{q_{n_q}}^{\text{upper}} \mid \\ & \mu_{q_1^s}^{\text{lower}}, \dots, \mu_{q_{n_{\text{slacks}}}^s}^{\text{lower}}\Big)^\intercal .\end{aligned} \tag{5.63}$$

We then obviously obtain

$$\left(\nabla_y C^{\text{ieq}}\right)^\intercal \cdot \mu = \left(\begin{array}{ccc} \mu_{s_0}^{\text{lower}} & - & \mu_{s_0}^{\text{upper}} \\ \vdots & & \vdots \\ \mu_{s_{n_T}}^{\text{lower}} & - & \mu_{s_{n_T}}^{\text{upper}} \\ \hline \mu_{w_0}^{\text{lower}} & - & \mu_{w_0}^{\text{upper}} \\ \vdots & & \vdots \\ \mu_{w_{n_T-1}}^{\text{lower}} & - & \mu_{w_{n_T-1}}^{\text{upper}} \\ \hline \mu_{q_1}^{\text{lower}} & - & \mu_{q_1}^{\text{upper}} \\ \vdots & & \vdots \\ \mu_{q_{n_q}}^{\text{lower}} & - & \mu_{q_{n_q}}^{\text{upper}} \\ \mu_{q_1^s}^{\text{lower}} & & \\ \vdots & & \\ \mu_{q_{n_{\text{slacks}}}^s}^{\text{lower}} & & \end{array}\right) . \tag{5.64}$$

Equations (5.59), (5.64) and (5.24) can then be used to efficiently compute the gradient of the Lagrangian (5.58) of the lower-level problem of (5.17).

Apart from dual feasibility, we also need primal feasibility for the KKT conditions of the lower level of (5.17), which is given by the following conditions:

$$C^{\text{eq}}(s, w, \bar{q}, p) = \begin{pmatrix} x(t_1; t_0, s_0, w_0, q, p) - s_1 \\ \vdots \\ x(t_{n_T}; t_{n_T-1}, s_{n_T-1}, w_{n_T-1}, q, p) - s_{n_T} \\ r^{\text{eq}}(s_0, \dots, s_{n_T}, q, p) \\ \tilde{r_1}^{\text{ieq}}(s, w, q, p) - q_1^s \\ \vdots \\ \tilde{r}^{\text{ieq}}_{n_{\text{slacks}}}(s, w, q, p) - q^s_{n_{\text{slacks}}} \end{pmatrix} = 0 \tag{5.65}$$

$$C^{\text{ieq}}(s, w, \bar{q}, p) = \begin{pmatrix} \begin{pmatrix} s \\ w \\ q \end{pmatrix} - b^{\text{lower}} \\ b^{\text{upper}} - \begin{pmatrix} s \\ w \\ q \end{pmatrix} \\ q_1^s \\ \vdots \\ q^s_{n_{\text{slacks}}} \end{pmatrix} \geq 0. \tag{5.66}$$

To complete the KKT conditions of the lower-level problem of (5.17), it remains to formulate complementarity, which is given by the following condition:

$$\begin{aligned} & C^{\text{ieq}}(s, w, \bar{q}, p)^{\intercal} \mu \\ = & \sum_{i=0}^{n_T} \sum_{j=0}^{n_x-1} \left(s_{ij} - b^{\text{lower}}_{s_{ij}}\right) \mu^{\text{lower}}_{s_{ij}} + \sum_{i=0}^{n_T-1} \sum_{j=0}^{(n_l \cdot n_u)-1} \left(w_{ij} - b^{\text{lower}}_{w_{ij}}\right) \mu^{\text{lower}}_{w_{ij}} \\ + & \sum_{i=1}^{n_q} \left(q_i - b^{\text{lower}}_{q_i}\right) \mu^{\text{lower}}_{q_i} + \sum_{i=0}^{n_T} \sum_{j=0}^{n_x-1} \left(b^{\text{upper}}_{s_{ij}} - s_{ij}\right) \mu^{\text{upper}}_{s_{ij}} \\ + & \sum_{i=0}^{n_T-1} \sum_{j=0}^{(n_l \cdot n_u)-1} \left(b^{\text{upper}}_{w_{ij}} - w_{ij}\right) \mu^{\text{lower}}_{w_{ij}} + \sum_{i=1}^{n_q} \left(b^{\text{upper}}_{q_i} - q_i\right) \mu^{\text{lower}}_{q_i} + \sum_{i=1}^{n_{\text{slacks}}} q_i^s \mu^{\text{lower}}_{q_i^s} \\ = & \;\; 0 \end{aligned} \tag{5.67}$$

with $b^{\text{lower}} = \left(b_s^{\text{lower}\intercal}, b_w^{\text{lower}\intercal}, b_q^{\text{lower}\intercal}\right)^{\intercal}$ and $b^{\text{upper}} = \left(b_s^{\text{upper}\intercal}, b_w^{\text{upper}\intercal}, b_q^{\text{upper}\intercal}\right)^{\intercal}$. We finally summarize the KKT conditions consisting of dual feasibility, primal feasibility and complementarity of the lower-level problem of (5.17):

$$\begin{aligned} \nabla_y \mathcal{L}(y, \lambda, \mu) \quad & = \;\; 0 \quad (\text{ dual feasibility }) \\ \mu \quad & \geq \;\; 0 \quad (\text{ dual feasibility }) \\ C^{\text{ieq}}(s, w, \bar{q}, p)^{\intercal} \mu \quad & = \;\; 0 \quad (\text{ complementarity }) \\ C^{\text{eq}}(s, w, \bar{q}, p) \quad & = \;\; 0 \quad (\text{ primal feasibility }) \\ C^{\text{ieq}}(s, w, \bar{q}, p) \quad & \geq \;\; 0 \quad (\text{ primal feasibility }). \end{aligned} \tag{5.68}$$

In order to derive and implement an efficient direct all-at-once approach for hierarchical dynamic optimization problems of type (5.2), it is indispensable to incorporate and exploit the exact structure of (5.68), which we described in this section.

5.4. Structure of the resulting nonlinear program and its derivatives

Please note that in the remainder of this chapter, we assume the structure given in (5.42) and (5.48) for the multi-point constraints in (5.17). After the parameterization and discretization of the infinite-dimensional hierarchical optimization problem (5.2), and after formulating necessary conditions for optimality for the lower-level problem of (5.17), we can now replace the lower-level problem of (5.17) by the necessary conditions for optimality (5.68) derived in the last section. We note that for a general OCP on the lower level in (5.2), the bilevel problem and the resulting one-level problem are not necessarily mathematically equivalent. However, in this thesis we are not concerned with this issue and refer to [Mir99, Dem02, Dem03, DZ13] for a detailed discussion. The replacement of the lower-level problem by its first-order optimality conditions leads to a highly structured finite-dimensional one-level problem – an NLP. A very compact form of the resulting NLP can be stated as:

$$
\begin{array}{lrcl}
\underset{\substack{s,w,\bar{q}\\ \gamma,p\\ \lambda,\mu}}{\text{minimize}} & \multicolumn{3}{l}{\dfrac{1}{2}\displaystyle\sum_{i=0}^{n_m}\sum_{j=1}^{n_h}\frac{(h_j(x(t_i^m),q,p)-\eta_{ij})^2}{\sigma_{ij}^2}} \\
\text{subject to} & 0 & = & C^{\text{eq}}(s,w,\bar{q},p) \\
 & 0 & = & \nabla_y \mathcal{L}(y,\lambda,\mu) \\
 & 0 & = & C^{\text{ieq}}(s,w,\bar{q},p)^{\mathsf{T}}\mu \\
 & 0 & \le & \mu \\
 & 0 & \le & C^{\text{ieq}}(s,w,\bar{q},p) \\
 & 1 & = & \sum_{i=1}^{n_{\mathcal{M}}}\gamma_i, \;\; \gamma\ge 0 \\
 & b^{\text{lower}_p} & \le & p \;\le\; b^{\text{upper}_p},
\end{array}
\tag{5.69}
$$

where the Lagrange multipliers λ and μ of the lower-level problem in (5.17) are now variables of the NLP (5.69). To have a closer look at the structure and the challenges of (5.69), we also provide the more detailed formulation, which is given by:

$$
\begin{array}{lrl}
\underset{\substack{s,w,\bar{q}\\ \gamma,p\\ \lambda,\mu}}{\text{minimize}} & \multicolumn{2}{l}{\dfrac{1}{2}\displaystyle\sum_{i=0}^{n_m}\sum_{j=1}^{n_h}\frac{(h_j(x(t_i^m),q,p)-\eta_{ij})^2}{\sigma_{ij}^2}} \\
\text{subject to} & 0 = & x(t_1;t_0,s_0,w_0,q,p)-s_1 \\
 & \vdots\;\vdots & \vdots \\
 & 0 = & x(t_{n_T};t_{n_T-1},s_{n_T-1},w_{n_T-1},q,p)-s_{n_T} \\
 & 0 = & r^{\text{eq}}(s_0,\dots,s_{n_T},q,p) \\
 & 0 = & \tilde{r_1}^{\text{ieq}}(s,w,q,p)-q_1^s \\
 & \vdots\;\vdots & \vdots \\
 & 0 = & \tilde{r}_{n_{\text{slacks}}}^{\text{ieq}}(s,w,q,p)-q_{n_{\text{slacks}}}^s
\end{array}
$$

$$
\begin{array}{rl}
\hline
0 = & -G_{s_0}^{\mathsf{T}} \lambda_{s_0}^{G} - \hat{R}_{s_0}^{\mathrm{eq}\,\mathsf{T}} \lambda_{s_0}^{\hat{R}^{\mathrm{eq}}} - \hat{R}_{s_0}^{\mathrm{eq}_{\mathrm{first}}\,\mathsf{T}} \lambda^{\hat{R}^{\mathrm{eq}}_{\mathrm{coupled}}} - \hat{R}_{s_0}^{\mathrm{ieq}\,\mathsf{T}} \lambda_{s_0}^{\hat{R}^{\mathrm{ieq}}} \\
 & -(\mu_{s_0}^{\mathrm{lower}} - \mu_{s_0}^{\mathrm{upper}}) \\
0 = & \mathbb{I}_{n_x} \lambda_{s_0}^{G} - G_{s_1}^{\mathsf{T}} \lambda_{s_1}^{G} - \hat{R}_{s_1}^{\mathrm{eq}\,\mathsf{T}} \lambda_{s_1}^{\hat{R}^{\mathrm{eq}}} - \hat{R}_{s_1}^{\mathrm{ieq}\,\mathsf{T}} \lambda_{s_1}^{\hat{R}^{\mathrm{ieq}}} \\
 & -(\mu_{s_1}^{\mathrm{lower}} - \mu_{s_1}^{\mathrm{upper}}) \\
\vdots\ \vdots & \vdots \\
0 = & \mathbb{I}_{n_x} \lambda_{s_{n_T-2}}^{G} - G_{s_{n_T-1}}^{\mathsf{T}} \lambda_{s_{n_T-1}}^{G} - \hat{R}_{s_{n_T-1}}^{\mathrm{eq}}{}^{\mathsf{T}} \lambda_{s_{n_T-1}}^{\hat{R}^{\mathrm{eq}}} \\
 & -\hat{R}_{s_{n_T-1}}^{\mathrm{ieq}}{}^{\mathsf{T}} \lambda_{s_{n_T-1}}^{\hat{R}^{\mathrm{ieq}}} - (\mu_{s_{n_T-1}}^{\mathrm{lower}} - \mu_{s_{n_T-1}}^{\mathrm{upper}}) \\
0 = & M_{s_{n_T}} + \mathbb{I}_{n_x} \lambda_{s_{n_T-1}}^{G} - \hat{R}_{s_{n_T}}^{\mathrm{eq}}{}^{\mathsf{T}} \lambda_{s_{n_T}}^{\hat{R}^{\mathrm{eq}}} - \hat{R}_{s_{n_T}}^{\mathrm{eq}_{\mathrm{last}}\,\mathsf{T}} \lambda^{\hat{R}^{\mathrm{eq}}_{\mathrm{coupled}}} - \hat{R}_{s_{n_T}}^{\mathrm{ieq}}{}^{\mathsf{T}} \lambda_{s_{n_T}}^{\hat{R}^{\mathrm{ieq}}} \\
 & -(\mu_{s_{n_T}}^{\mathrm{lower}} - \mu_{s_{n_T}}^{\mathrm{upper}}) \\
0 = & G_{w_0}^{\mathsf{T}} \lambda_{s_0}^{G} - \hat{R}_{w_0}^{\mathrm{ieq}\,\mathsf{T}} \lambda_{s_0}^{\hat{R}^{\mathrm{ieq}}} - (\mu_{w_0}^{\mathrm{lower}} - \mu_{w_0}^{\mathrm{upper}}) \\
0 = & G_{w_1}^{\mathsf{T}} \lambda_{s_1}^{G} - \hat{R}_{w_1}^{\mathrm{ieq}\,\mathsf{T}} \lambda_{s_1}^{\hat{R}^{\mathrm{ieq}}} - (\mu_{w_1}^{\mathrm{lower}} - \mu_{w_1}^{\mathrm{upper}}) \\
\vdots\ \vdots & \vdots \\
0 = & G_{w_{n_T-2}}^{\mathsf{T}} \lambda_{s_{n_T-2}}^{G} - \hat{R}_{w_{n_T-2}}^{\mathrm{ieq}}{}^{\mathsf{T}} \lambda_{s_{n_T-2}}^{\hat{R}^{\mathrm{ieq}}} - (\mu_{w_{n_T-2}}^{\mathrm{lower}} - \mu_{w_{n_T-2}}^{\mathrm{upper}}) \\
0 = & G_{w_{n_T-1}}^{\mathsf{T}} \lambda_{s_{n_T-1}}^{G} - \hat{R}_{w_{n_T-1}}^{\mathrm{ieq}}{}^{\mathsf{T}} \lambda_{s_{n_T-1}}^{\hat{R}^{\mathrm{ieq}}} - \hat{R}_{w_{n_T-1}}^{\mathrm{last}}{}^{\mathsf{T}} \lambda_{s_{n_T}}^{\hat{R}^{\mathrm{ieq}}} - (\mu_{w_{n_T-1}}^{\mathrm{lower}} - \mu_{w_{n_T-1}}^{\mathrm{upper}}) \\
0 = & M_{q_1} - \sum_{i=0}^{n_T-1} G_{q_1}^{i}{}^{\mathsf{T}} \lambda_{s_i}^{G} - \sum_{i=0}^{n_T} \hat{R}_{q_1}^{\mathrm{eq}_i\,\mathsf{T}} \lambda_{s_i}^{\hat{R}^{\mathrm{eq}}} - \hat{R}_{q_1}^{\mathrm{eq}_{\mathrm{coupled}}\,\mathsf{T}} \lambda^{\hat{R}^{\mathrm{eq}}_{\mathrm{coupled}}} - \\
 & \sum_{i=0}^{n_T} \hat{R}_{q_1}^{\mathrm{ieq}_i\,\mathsf{T}} \lambda_{s_i}^{\hat{R}^{\mathrm{ieq}}} - (\mu_{q_1}^{\mathrm{lower}} - \mu_{q_1}^{\mathrm{upper}}) \\
0 = & M_{q_2} - \sum_{i=0}^{n_T-1} G_{q_2}^{i}{}^{\mathsf{T}} \lambda_{s_i}^{G} - \sum_{i=0}^{n_T} \hat{R}_{q_2}^{\mathrm{eq}_i\,\mathsf{T}} \lambda_{s_i}^{\hat{R}^{\mathrm{eq}}} - \hat{R}_{q_2}^{\mathrm{eq}_{\mathrm{coupled}}\,\mathsf{T}} \lambda^{\hat{R}^{\mathrm{eq}}_{\mathrm{coupled}}} - \\
 & \sum_{i=0}^{n_T} \hat{R}_{q_2}^{\mathrm{ieq}_i\,\mathsf{T}} \lambda_{s_i}^{\hat{R}^{\mathrm{ieq}}} - (\mu_{q_2}^{\mathrm{lower}} - \mu_{q_2}^{\mathrm{upper}}) \\
\vdots\ \vdots & \vdots \\
0 = & M_{q_{n_q-1}} - \sum_{i=0}^{n_T-1} G_{q_{n_q-1}}^{i}{}^{\mathsf{T}} \lambda_{s_i}^{G} - \sum_{i=0}^{n_T} \hat{R}_{q_{n_q-1}}^{\mathrm{eq}_i}{}^{\mathsf{T}} \lambda_{s_i}^{\hat{R}^{\mathrm{eq}}} - \hat{R}_{q_{n_q-1}}^{\mathrm{eq}_{\mathrm{coupled}}\,\mathsf{T}} \lambda^{\hat{R}^{\mathrm{eq}}_{\mathrm{coupled}}} - \\
 & \sum_{i=0}^{n_T} \hat{R}_{q_{n_q-1}}^{\mathrm{ieq}_i}{}^{\mathsf{T}} \lambda_{s_i}^{\hat{R}^{\mathrm{ieq}}} - (\mu_{q_{n_q-1}}^{\mathrm{lower}} - \mu_{q_{n_q-1}}^{\mathrm{upper}}) \\
0 = & M_{q_{n_q}} - \sum_{i=0}^{n_T-1} G_{q_{n_q}}^{i}{}^{\mathsf{T}} \lambda_{s_i}^{\hat{R}^{\mathrm{eq}}} - \sum_{i=0}^{n_T} \hat{R}_{q_{n_q}}^{\mathrm{eq}_i\,\mathsf{T}} \lambda_{s_i}^{\hat{R}^{\mathrm{eq}}} - \hat{R}_{q_{n_q}}^{\mathrm{eq}_{\mathrm{coupled}}\,\mathsf{T}} \lambda^{\hat{R}^{\mathrm{eq}}_{\mathrm{coupled}}} - \\
 & \sum_{i=0}^{n_T} \hat{R}_{q_{n_q}}^{\mathrm{ieq}_i\,\mathsf{T}} \lambda_{s_i}^{\hat{R}^{\mathrm{ieq}}} - (\mu_{q_{n_q}}^{\mathrm{lower}} - \mu_{q_{n_q}}^{\mathrm{upper}}) \\
0 = & \begin{pmatrix} \lambda_{s_0}^{\hat{R}^{\mathrm{ieq}}} \\ \vdots \\ \lambda_{s_{n_T}}^{\hat{R}^{\mathrm{ieq}}} \end{pmatrix} - \begin{pmatrix} \mu_{q_1^s}^{\mathrm{lower}} \\ \vdots \\ \mu_{q_{n_{\mathrm{slacks}}}^s}^{\mathrm{lower}} \end{pmatrix} \\
\hline
0 = & \sum_{i=0}^{n_T} \sum_{j=0}^{n_x-1} \left(s_{ij} - b_{s_{ij}}^{\mathrm{lower}} \right) \mu_{s_{ij}}^{\mathrm{lower}} \\
 & -\sum_{i=0}^{n_T-1} \sum_{j=0}^{(n_l \cdot n_u)-1} \left(w_{ij} - b_{w_{ij}}^{\mathrm{lower}} \right) \mu_{w_{ij}}^{\mathrm{lower}} \\
 & -\sum_{i=1}^{n_q} \left(q_i - b_{q_i}^{\mathrm{lower}} \right) \mu_{q_i}^{\mathrm{lower}} - \sum_{i=0}^{n_T} \sum_{j=0}^{n_x-1} \left(b_{s_{ij}}^{\mathrm{upper}} - s_{ij} \right) \mu_{s_{ij}}^{\mathrm{upper}} \\
 & -\sum_{i=0}^{n_T-1} \sum_{j=0}^{(n_l \cdot n_u)-1} \left(b_{w_{ij}}^{\mathrm{upper}} - w_{ij} \right) \mu_{w_{ij}}^{\mathrm{lower}} \\
 & -\sum_{i=1}^{n_q} \left(b_{q_i}^{\mathrm{upper}} - q_i \right) \mu_{q_i}^{\mathrm{lower}} - \sum_{i=1}^{n_{\mathrm{slacks}}} q_i^s \mu_{q_i^s}^{\mathrm{lower}} \\
\hline
0 \leq & \mu \\
\hline
0 \leq & \left(s^{\mathsf{T}}, w^{\mathsf{T}}, q^{\mathsf{T}} \right)^{\mathsf{T}} - b^{\mathrm{lower}} \\
0 \leq & b^{\mathrm{upper}} - \left(s^{\mathsf{T}}, w^{\mathsf{T}}, q^{\mathsf{T}} \right)^{\mathsf{T}} \\
0 \leq & q^s \\
\hline
1 = & \sum_{i=1}^{n_{\mathcal{M}}} \gamma_i \\
0 \leq & \gamma \\
p \leq & b^{\mathrm{upper}_p} \\
p \geq & b^{\mathrm{lower}_p}.
\end{array}
\tag{5.70}
$$

We summarize all variables in NLP (5.70) by

$$\begin{aligned} z \;=\; & \left(s_0^{\mathsf{T}}, \dots, s_{n_T}^{\mathsf{T}} \mid w_0^{\mathsf{T}}, \dots, w_{n_T-1}^{\mathsf{T}} \mid q_1, \dots, q_{n_q} \mid q_1^s, \dots, q_{n_{\text{slacks}}}^s \mid \right. \\ & \left. \gamma_1, \dots, \gamma_{n_{\mathcal{M}}} \mid p_1, \dots, p_{n_p} \mid \lambda_1, \dots, \lambda_{n_\lambda} \mid \mu_1, \dots, \mu_{n_\mu}\right)^{\mathsf{T}} \\ =\; & \left(y^{\mathsf{T}} \mid \gamma_1, \dots, \gamma_{n_\gamma} \mid p_1, \dots, p_{n_p} \mid \lambda_1, \dots, \lambda_{n_\lambda} \mid \mu_1, \dots, \mu_{n_\mu}\right)^{\mathsf{T}}. \end{aligned} \tag{5.71}$$

In the next sections, we discuss the challenges of solving the NLP (5.70) in contrast to a standard NLP described in Chapter 2. Before we derive a Gauß-Newton-type method for solving (5.70), we have a closer look at the constraint Jacobian and the Hessian of the Lagrangian of (5.70). We start with the constraint Jacobian of (5.70), where we skip simple bounds on the variables. Therefore, we consider the following matrix consisting of Blocks $J1 - J9$:

$$J^{\mathsf{T}} := \begin{array}{c|c|c|c|c} {C^{\text{eq}}}^{\mathsf{T}} & \nabla_y \mathcal{L}^{\mathsf{T}} & {C^{\text{ieq}}}^{\mathsf{T}}\mu & \sum_{i=1}^{n_{\mathcal{M}}} \gamma_i\text{-}1 & \\ \hline J1 & J2 & J3 & & y \\ \hline J4 & J5 & & J9 & \gamma \\ & & & & p \\ \hline & J6 & & & \lambda \\ \hline & J7 & J8 & & \mu \end{array} \;. \tag{5.72}$$

In an algorithmic framework, it is common to provide the constraint Jacobian neglecting simple bounds on the variables, since the derivative of the simple bounds is trivial and can be added in an automatic fashion. The Jacobian (5.72) is needed to solve (5.70) and is described in the following in detail. The columns of J^{T} correspond to the constraints

$$\left({C^{\text{eq}}}^{\mathsf{T}} \mid \nabla_y \mathcal{L}^{\mathsf{T}} \mid {C^{\text{ieq}}}^{\mathsf{T}}\mu \mid \textstyle\sum_{i=1}^{n_{\mathcal{M}}} \gamma_i - 1\right) \tag{5.73}$$

from (5.69) and the rows of J^{T} correspond to the variables

$$\left(s^{\mathsf{T}},\, w^{\mathsf{T}},\, q^{\mathsf{T}},\, (q^s)^{\mathsf{T}} \mid \gamma^{\mathsf{T}},\, p^{\mathsf{T}} \mid \lambda^{\mathsf{T}} \mid \mu^{\mathsf{T}}\right)^{\mathsf{T}}. \tag{5.74}$$

The grouping in (5.73) and (5.74) corresponds to the blocks in (5.72). The blocks of J that have no label are blocks of zeros.

The Block $J1$ is the transpose of the derivative of the equality constraints C^{eq} with respect to y shown in (5.55):

$$J1 := \nabla_y {C^{\text{eq}}}^{\mathsf{T}}. \tag{5.75}$$

Block $J1$ appears a second time in J in the derivative of $\nabla_y \mathcal{L}(y, \lambda, \mu) = \nabla_y \tilde{\Phi} - \left(\nabla_y C^{\text{eq}}\right)^{\mathsf{T}} \lambda - \left(\nabla_y C^{\text{ieq}}\right)^{\mathsf{T}} \mu$ with respect to λ in a negated form:

$$J6 := \nabla_{y\lambda} \mathcal{L}(y, \lambda, \mu)^{\mathsf{T}} = -\nabla_y C^{\text{eq}}, \tag{5.76}$$

which means that $\nabla_y C^{\text{eq}}$ has to be computed once and can then be used in Blocks $J1$ and $J6$, and to compute $\left(\nabla_y C^{\text{eq}}\right)^{\mathsf{T}} \lambda$ in $\nabla_y \mathcal{L}(y, \lambda, \mu)$.

Block $J2$ contains the Hessian of the Lagrangian of the lower-level problem of (5.17):

$$J2 := \nabla_y^2 \mathcal{L}(y, \lambda, \mu) =$$

$$\left(\begin{array}{ccc|ccc|c|c}
\nabla^2_{s_0}\mathcal{L} & & & \nabla^2_{w_0,s_0}\mathcal{L} & & & \nabla^2_{q,s_0}\mathcal{L} & \\
 & \ddots & & & \ddots & & \vdots & \\
 & & & & & \nabla^2_{w_{n_T-1},s_{n_T-1}}\mathcal{L} & & \\
 & & \nabla^2_{s_{n_T}}\mathcal{L} & & & \nabla^2_{w_{n_T-1},s_{n_T}}\mathcal{L} & \nabla^2_{q,s_{n_T}}\mathcal{L} & \\
\hline
\nabla^2_{s_0,w_0}\mathcal{L} & & & \nabla^2_{w_0}\mathcal{L} & & & \nabla^2_{q,w_0}\mathcal{L} & \\
 & \ddots & & & \ddots & & \vdots & \\
 & \nabla^2_{s_{n_T-1},w_{n_T-1}}\mathcal{L} & \nabla^2_{s_{n_T},w_{n_T-1}}\mathcal{L} & & & \nabla^2_{w_{n_T-1}}\mathcal{L} & \nabla^2_{q,w_{n_T-1}}\mathcal{L} & \\
\hline
\nabla^2_{s_0,q}\mathcal{L} & \dots & \nabla^2_{s_{n_T},q}\mathcal{L} & \nabla^2_{w_0,q}\mathcal{L} & \dots & \nabla^2_{w_{n_T-1},q}\mathcal{L} & \nabla^2_{q}\mathcal{L} & \\
\hline
 & & & & & & &
\end{array}\right), \tag{5.77}$$

where arguments of $\mathcal{L}$ are skipped and the rows and columns within $J2$ correspond to

$$\left(s_0^{\mathsf{T}}, \dots, s_{n_T}^{\mathsf{T}} \mid w_0^{\mathsf{T}}, \dots, w_{n_T-1}^{\mathsf{T}} \mid q^{\mathsf{T}} \quad (q^s)^{\mathsf{T}}\right). \tag{5.78}$$

Based on the very special structure of the Hessian of the Lagrangian of the lower-level problem of (5.17), we derive an algorithm for efficiently computing $J2$, which is discussed at the end of this section. In many practical applications we can make sure that

- the control discretization grid and the multiple shooting grid are the same
- the grid of the multi-point equality and inequality constraints is the same as the multiple shooting grid
- the multi-point equality and inequality constraints are decoupled over the multiple shooting nodes or the constraints on the first and last node are linearly coupled (structures (5.42) and (5.48) are given),

which forms the basis for the special structure of $J2$. The structure and the dimensions of $J2$ for a specific hierarchical dynamic optimization problem of type (5.1) with $n_x = 10$, $n_T = 20$, $n_u = 3$, $n_{\bar{q}} = 18$ and $n_{\tilde{r}^{\text{eq}}} = 21$ is shown in Figure 5.2 up to the different order of variables $\left(s^{\mathsf{T}} \mid \bar{q}^{\mathsf{T}} \mid w^{\mathsf{T}}\right)$.

Blocks $J4$ and $J5$ are almost dense blocks without a special structure because of the global parameter p:

$$J4 := \left(\nabla_{(\gamma,p)} C^{\text{eq}}\right)^{\mathsf{T}} \quad \text{and} \quad J5 := \left(\nabla_{y,(\gamma,p)} \mathcal{L}(y, \lambda, \mu)\right)^{\mathsf{T}}, \tag{5.79}$$

where the notation $\nabla_{(\gamma,p)} C^{\text{eq}}$ means $(\nabla_\gamma C^{\text{eq}}, \nabla_p C^{\text{eq}}) \in \mathbb{R}^{n_{\tilde{r}^{\text{eq}}}} \times \mathbb{R}^{n_{\mathcal{M}}+n_p}$. In Block $J4$, we can exploit that the rows corresponding to γ are zero. Block $J7$ contains the derivative of

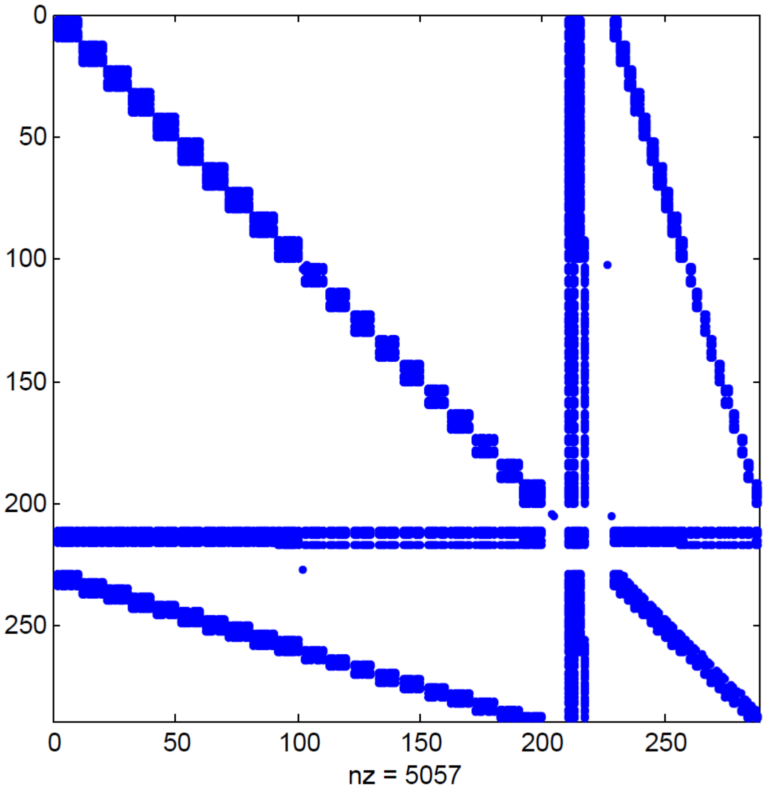


Figure 5.2.: Structure of $J2$ for a hierarchical dynamic optimization problem of type (5.1) with $n_x = 10$, $n_T = 20$, $n_u = 3$, $n_{\bar{q}} = 18$ and $n_{\tilde{r}^{\text{eq}}} = 21$ where rows and columns correspond to the variables $\left(s^\intercal \mid \bar{q}^\intercal \mid w^\intercal\right)$. The number of nonzeros is 5057.

$\nabla_y \mathcal{L}^\intercal$ with respect to the Lagrange multiplier μ for the inequality constraints with

$$
\begin{aligned}
\mu \quad &= \quad \Big({\mu_{s_0}^{\text{lower}}}^\intercal, \dots, {\mu_{s_{n_T}}^{\text{lower}}}^\intercal \mid {\mu_{w_0}^{\text{lower}}}^\intercal, \dots, {\mu_{w_{n_T-1}}^{\text{lower}}}^\intercal \mid \mu_{q_1}^{\text{lower}}, \dots, \mu_{q_{n_q}}^{\text{lower}} \mid \qquad (5.80)\\
&\qquad {\mu_{s_0}^{\text{upper}}}^\intercal, \dots, {\mu_{s_{n_T}}^{\text{upper}}}^\intercal \mid {\mu_{w_0}^{\text{upper}}}^\intercal, \dots, {\mu_{w_{n_T-1}}^{\text{upper}}}^\intercal \mid \mu_{q_1}^{\text{upper}}, \dots, \mu_{q_{n_q}}^{\text{upper}} \mid \qquad (5.81)\\
&\qquad {\mu_{q_1^s}^{\text{lower}}}^\intercal, \dots, {\mu_{q^s_{n_{\text{slacks}}}}^{\text{lower}}}^\intercal \Big)^\intercal \qquad (5.82)\\
&= \quad \left({\mu^{\text{lower}}}^\intercal \mid {\mu^{\text{upper}}}^\intercal \mid {\mu_{q^s}^{\text{lower}}}^\intercal \right)^\intercal, \qquad (5.83)
\end{aligned}
$$

where μ is separated into the multipliers corresponding to lower bounds on s, w, q, the multipliers corresponding to upper bounds on s, w, q, and the multipliers corresponding to the lower bound on the slacks q^s. For Block $J7$, we then obtain:

$$J7 := \left(\nabla^2_{y\mu}\mathcal{L}(y,\lambda,\mu)\right)^{\mathsf{T}} = \left(\begin{array}{ccc|ccc} -1 & & & & & \\ & \ddots & & & & \\ & & -1 & & & \\ \hline 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ \hline & & & -1 & & \\ & & & & \ddots & \\ & & & & & -1 \end{array}\right) \tag{5.84}$$

with the rows of $J7$ corresponding to $\left(\mu^{\text{lower}^{\mathsf{T}}} \mid \mu^{\text{upper}^{\mathsf{T}}} \mid \mu_{q^s}^{\text{lower}^{\mathsf{T}}}\right)^{\mathsf{T}}$, and the columns corresponding to $\left((\nabla_{(s,w,q)}\mathcal{L})^{\mathsf{T}} \mid (\nabla_{q^s}\mathcal{L})^{\mathsf{T}}\right)$.

The constraint ${C^{\text{ieq}}}^{\mathsf{T}}\mu$ is one-dimensional, which leads to tall Blocks $J3$ and $J8$. For $J3$ we have

$$J3 := \nabla_y\left({C^{\text{ieq}}}^{\mathsf{T}}\mu\right) = \left(\begin{array}{c} \mu^{\text{lower}}_{s_0} - \mu^{\text{upper}}_{s_0} \\ \vdots \\ \mu^{\text{lower}}_{s_{n_T}} - \mu^{\text{upper}}_{s_{n_T}} \\ \hline \mu^{\text{lower}}_{w_0} - \mu^{\text{upper}}_{w_0} \\ \vdots \\ \mu^{\text{lower}}_{w_{n_T-1}} - \mu^{\text{upper}}_{w_{n_T-1}} \\ \hline \mu^{\text{lower}}_{q_1} - \mu^{\text{upper}}_{q_1} \\ \vdots \\ \mu^{\text{lower}}_{q_{n_q}} - \mu^{\text{upper}}_{q_{n_q}} \\ \hline \mu^{\text{lower}}_{q^s_1} \\ \vdots \\ \mu^{\text{lower}}_{q^s_{n_{\text{slacks}}}} \end{array}\right), \tag{5.85}$$

and $J8$ is given by

$$J8 := \nabla_\mu\left({C^{\text{ieq}}}^{\mathsf{T}}\mu\right) = C^{\text{ieq}} = \left(\begin{array}{c} \begin{pmatrix} s \\ w \\ q \end{pmatrix} - b^{\text{lower}} \\ b^{\text{upper}} - \begin{pmatrix} s \\ w \\ q \end{pmatrix} \\ q^s_1 \\ \vdots \\ q^s_{n_{\text{slacks}}} \end{array}\right). \tag{5.86}$$

The Block $J9$ is given by

$$(1,\dots,1,\ 0,\dots,0)^{\intercal}\,, \tag{5.87}$$

where the first $n_{\mathcal{M}}$ entries are ones and the remaining entries are zeros.

Figure 5.3 shows the structure of the constraint Jacobian (5.72) of the NLP (5.70) for a relatively small hierarchical dynamic optimization problem of type (5.1) with $n_x = 10$, $n_T = 20$, $n_u = 3$, $n_{\bar{q}} = 18$, $n_p = 1$ and $n_{\tilde{r}^{\mathrm{eq}}} = 21$. The illustration in Figure 5.3 is based on the implementation of the algorithm derived later in this chapter, where the order of constraints and variables is slightly changed. The columns in Figure 5.3 correspond to the constraints

$$\left(C_1^{\mathrm{eq}^{\intercal}} \mid \nabla_y \mathcal{L}^{\intercal} \mid C^{\mathrm{ieq}^{\intercal}}\mu \mid C_2^{\mathrm{eq}^{\intercal}}\right) \tag{5.88}$$

with $C^{\mathrm{eq}^{\intercal}} = \left(C_1^{\mathrm{eq}^{\intercal}}, C_2^{\mathrm{eq}^{\intercal}}\right)^{\intercal}$, where C_1^{eq} contains the matching conditions and C_2^{eq} contains the remaining constraints of C^{eq}. In Figure 5.3, Block $J11$ refers to $\nabla_y C_1^{\mathrm{eq}^{\intercal}}$ and Block $J12$ refers to $\nabla_y C_2^{\mathrm{eq}^{\intercal}}$. The rows of Figure 5.3 correspond to the variables

$$\left(s^{\intercal} \mid q^{\intercal}, (q^s)^{\intercal} \mid w^{\intercal} \mid \gamma^{\intercal} \mid p^{\intercal} \mid \lambda^{\intercal} \mid \mu^{\intercal}\right)^{\intercal}. \tag{5.89}$$

Furthermore, the slack variable q^s have lower and upper bounds in Figure 5.2.

5.5. A structure-exploiting Gauß-Newton-type method

In this section, we start with explaining the basic steps of a Gauß-Newton-type method for solving (5.70). Afterwards, we in detail discuss how to compute the quantities needed in the Gauß-Newton approach in a structure-exploiting way based on the analysis of the structure of NLP (5.70) and its derivatives presented in the last section.

As mentioned above, we choose a Gauß-Newton-type method for solving (5.70). The advantage of a GGN approach compared to a sequential quadratic programming (SQP) technique (with, e.g., an exact Hessian of the Lagrangian or a BFGS approximation [NW99]) is the fact that full step GGN methods converge locally to stable minima (cf. [Boc87]). In this section, we describe a GGN method based on the one stated in Section 2.2.2 for equality constrained NLPs of type

$$\begin{aligned} \underset{z}{\text{minimize}} \quad & \tfrac{1}{2}\|\mathfrak{F}(z)\|_2^2 \\ \text{subject to} \quad & \mathfrak{G}(z) = 0, \end{aligned} \tag{5.90}$$

where we summarize the dimension of $\mathfrak{F}(z)$ by $n_{\mathfrak{F}}$ and of $\mathfrak{G}(z)$ by $n_{\mathfrak{G}}$, and the size of z is given by n_z. For a compact presentation of the algorithm, we for now consider the NLP (5.70) in the form of (5.90), where $\mathfrak{F}(z)$ describes the least-squares residual and $\mathfrak{G}(z)$ includes the equality constraints and all active inequality constraints at z. In this section, we do not focus on the active-set strategy used for solving (5.70). In the implementation of the algorithm presented in this chapter (cf. Chapter 9), a null-space active set method is used, which is implemented in the NLP solver filterSQP [FL02a]. The main focus of this section is the influence of the structure of (5.70) inherited from the bilevel setting, the multiple shooting parameterization, and the control and constraint discretization of (5.2) on the quantities needed in the GGN method. As already stated in Section 2.2.2, we make the following two assumption:

Assumption 1: The constraint normals of (5.90) are linearly independent:

$$\operatorname{rank}(\nabla_z \mathfrak{G}(z)) = n_{\mathfrak{G}}. \tag{5.91}$$

Assumption 2: All free parameters are identifiable:

$$\operatorname{rank}\begin{pmatrix} \nabla_z \mathfrak{F}(z) \\ \nabla_z \mathfrak{G}(z) \end{pmatrix} = n_z. \tag{5.92}$$

Assumptions 1 and 2 ensure that the Hessian approximation $\nabla_z \mathfrak{F}(z)^{\mathsf{T}} \nabla_z \mathfrak{F}(z)$ is positive definite on the null space of $\nabla_z \mathfrak{G}(z)$, and that the KKT matrix

$$\begin{pmatrix} \nabla_z \mathfrak{F}(z)^{\mathsf{T}} \nabla_z \mathfrak{F}(z) & -\nabla_z \mathfrak{G}(z)^{\mathsf{T}} \\ \nabla_z \mathfrak{G}(z) & 0 \end{pmatrix} \tag{5.93}$$

is regular [Boc87, NW99, Kö02]. We provide a brief sketch of a basic GGN method for solving (5.90) in Algorithm 2.

Algorithm 2: A basic Generalized Gauß-Newton method.

```
Choose an initial value $z_0$;
for $k = 0, 1, \cdots$ do
    Solve the linearized problem
        $\underset{\Delta z}{\text{minimize}} \quad \frac{1}{2}\|\nabla_z \mathfrak{F}(z)\Delta z + \mathfrak{F}(z)\|_2^2$
        $\text{subject to} \quad \nabla_z \mathfrak{G}(z)\Delta z + \mathfrak{G}(z) = 0;$        (5.94)
    Evaluate the iterate
        $z_{k+1} = z_k + \alpha \cdot \Delta z,$        (5.95)
    where $\alpha \in (0, 1]$ is computed by means of a globalization strategy;
    if convergence test satisfied then
        STOP with approximated solution $z_{k+1}$;
    end
end
```

We now take a closer look at the objective of the linearized subproblem (5.94), which is given by:

$$\begin{aligned} &\tfrac{1}{2}\|\nabla_z \mathfrak{F}(z)\Delta z + \mathfrak{F}(z)\|_2^2 = \\ &\tfrac{1}{2}\left(\Delta z^{\mathsf{T}} \nabla_z \mathfrak{F}(z)^{\mathsf{T}} \nabla_z \mathfrak{F}(z) \Delta z\right) + \mathfrak{F}(z)^{\mathsf{T}} \nabla_z \mathfrak{F}(z)\Delta z + \tfrac{1}{2}\mathfrak{F}(z)^{\mathsf{T}} \mathfrak{F}(z). \end{aligned} \tag{5.96}$$

The last term in (5.96) can be neglected in the optimization problem (5.94) since it does not depend on Δz. In Section 2.2.1, we state the quadratic program (QP) that is solved within an SQP framework. In Section 2.2.2, we explain the relation between an SQP and a GGN method and furthermore discuss the approximation of the Hessian of the Lagrangian by $\nabla_z \mathfrak{F}(z)^{\mathsf{T}} \nabla_z \mathfrak{F}(z)$ for an NLP of type (5.90). With the latter approximation, the QP (5.94) corresponds to the one of an SQP method, which means that in practice, a GGN method can be realized in an SQP framework by approximating the Hessian of the Lagrangian of

(5.90) by $\nabla_z \mathfrak{F}(z)^\intercal \nabla_z \mathfrak{F}(z)$, which we do in the implementation of the algorithm derived in this chapter based on the NLP solver filterSQP [FL02a], which is further discussed in Chapter 9.

We now discuss how to exploit the structure of the NLP (5.70) when computing quantities needed in Algorithm 2. We analyze the computation of the following main quantities:

- the constraints: $\mathfrak{G}(z)$
- the constraint Jacobian: $\nabla_z \mathfrak{G}(z)$
- the derivative of the objective: $\nabla_z \left(\frac{1}{2}\|\mathfrak{F}(z)\|_2^2\right)$
- the Hessian approximation of the Lagrangian: $\nabla_z \mathfrak{F}(z)^\intercal \nabla_z \mathfrak{F}(z)$
- the Hessian approximation times a vector d: $\left(\nabla_z \mathfrak{F}(z)^\intercal \nabla_z \mathfrak{F}(z)\right) \cdot d$

and start with the constraints $\mathfrak{G}(z) = 0$ containing all active constraints of

$$\begin{aligned}
&0 = C^{\text{eq}}(s, w, \bar{q}, p)\\
&0 = \nabla_y \mathcal{L}(y, \lambda, \mu)\\
&0 = C^{\text{ieq}}(s, w, \bar{q}, p)^\intercal \mu\\
&0 \leq \mu\\
&0 \leq C^{\text{ieq}}(s, w, \bar{q}, p)\\
&1 = \textstyle\sum_{i=1}^{n_\mathcal{M}} \gamma,\ \gamma \geq 0\\
&b^{\text{lower}_p} \leq p \ \leq \ b^{\text{upper}_p},
\end{aligned} \tag{5.97}$$

where C^{eq} is given in (5.20), C^{ieq} is stated in (5.21), and complementarity $C^{\text{ieq}}(s, w, \bar{q}, p)^\intercal \mu$ is provided in (5.67). The cost of computing the complementarity constraint is low, but the numerical treatment of this constraint is complex since it violates standard constraint qualifications (cf. Section 2.3). The treatment of the complementarity constraint is discussed in detail in Chapter 8. The gradient of the Lagrangian $\nabla_y \mathcal{L}(y, \lambda, \mu)$ includes derivatives of the matching conditions and the equality constraints (5.20) with respect to y. The derivative of C^{eq} with respect to y is given in (5.55) (for multi-point constraints of the form (5.42) and (5.48)). The quantities G_{s_i}, G_{w_i} and $G_{q_j}^i$ with $i = 0, \ldots, n_T - 1$ and $j = 1, \ldots, n_q$ are derivatives of the ODE's solution on interval I_i with respect to s, w and q, which we call sensitivities. The sensitivities are computed based on internal numerical differentiation and the solution of the variational differential equation (cf. Section 3.1). To ensure the principle of internal numerical differentiation (also explained in Section 3.1) the variational differential equation and the ODE in (5.2) are integrated simultaneously. The variational differential equations are based on the derivative of the ODE's right-hand side, which is obtained by using symbolic expressions (which are available for many applications and can be provided by the user in the implementation of this algorithm described in Chapter 9) or automatic differentiation. The quantities $\hat{R}_{s_i}^{\text{eq}}, \hat{R}_{q_j}^{\text{eq}_i}, \hat{R}_{s_0}^{\text{eq}_{\text{first}}}, \hat{R}_{s_{n_T}}^{\text{eq}_{\text{last}}}, \hat{R}_{q_j}^{\text{eq}_{\text{coupled}}}, \hat{R}_{s_i}^{\text{ieq}}, \hat{R}_{w_k}^{\text{ieq}}, \hat{R}_{w_{n_T-1}}^{\text{last}}$, and $\hat{R}_{q_j}^{\text{ieq}_i}$ with $i = 0, \ldots, n_T$, $j = 1, \ldots, n_q$ and $k = 0, \ldots, n_T - 1$ in (5.55) contain derivatives of the mixed control-state constraints and the multi-point constraints, and are also obtained by using symbolic expressions (which are available for most of the applications and can also be provided by the user in the implementation of this algorithm described in Chapter 9) or automatic differentiation. The same is true for the derivative of the lower-level objective of (5.17), which is stated in (5.24). For the computation of the products $\left(\nabla_y C^{\text{eq}}\right)^{\mathsf{T}} \lambda$ and $\left(\nabla_y C^{\text{ieq}}\right)^{\mathsf{T}} \mu$, we use formula (5.59) and (5.64). We note that the products of sensitivities with vectors in (5.59) can also directly be computed using automatic differentiation.

We now consider the computation of the constraint Jacobian $\nabla_z \mathfrak{G}(z)$, where we neglect the simple bounds $C^{\text{ieq}}(s, w, \bar{q}, p) \geq 0$, $\mu \geq 0$ and $b^{\text{lower}_p} \leq p \leq b^{\text{upper}_p}$ since their derivative is trivial and can be added in an automatic fashion. The computation of the Blocks $J1 - J9$ in (5.72) is discussed in detail in the following.

Computation of Blocks $J1$ and $J6$

Blocks $J1$ and $J6$ contain the derivatives of the matching conditions and the equality constraints (5.20) with respect to y. These derivatives are also computed for evaluating $\nabla_y \mathcal{L}(y, \lambda, \mu)$ as described in the last paragraph and can be reused here or need to be recomputed if the variables z have changed.

Computation of Block $J2$

The most expensive part of J in (5.72) is the Hessian of the Lagrangian of the lower-level problem in (5.17) in Block $J2$, which is shown in more detail in (5.77) for multi-point constraints of the form (5.42) and (5.48). The gradient of the Lagrangian $\nabla_y \mathcal{L}(y, \lambda, \mu)$ can be computed based on symbolic/automatic differentiation derivatives of the ODE's right-hand side, symbolic/automatic differentiation derivatives of C^{eq}, C^{ieq} and $\tilde{\Phi}$ with respect to y, and based on the solution of a system of variational differential equations. This means that $\nabla_y \mathcal{L}(y, \lambda, \mu)$ does not contain derivatives computed using finite differences. The derivative of $\nabla_y \mathcal{L}(y, \lambda, \mu)$ with respect to q is now built based on finite differences. The rows and columns in (5.77) corresponding to q have no special structure. The derivative of $\nabla_y \mathcal{L}(y, \lambda, \mu)$ with respect to q^s is zero and is not computed. The remaining entries of (5.77) are also computed using finite differences, but in a very efficient and structure-exploiting way since these entries are the computationally most expensive ones in J. The computation is based on the assumptions that the control discretization grid, the multiple shooting grid and the multi-point equality and inequality constraint grid are the same, and that the multi-point equality and inequality constraints are decoupled over the multiple shooting nodes or the constraints on the first and last node are linearly coupled (structures (5.42) and (5.48) are given). We need to compute the derivatives of

$$\begin{aligned} \nabla_{(s,w)} \mathcal{L}(y, \lambda, \mu) = \; & \Big(\left(\nabla_{s_0} \mathcal{L}(y, \lambda, \mu)\right)^{\mathsf{T}}, \ldots, \left(\nabla_{s_{n_T}} \mathcal{L}(y, \lambda, \mu)\right)^{\mathsf{T}}, \\ & \left(\nabla_{w_0} \mathcal{L}(y, \lambda, \mu)\right)^{\mathsf{T}}, \ldots, \left(\nabla_{w_{n_T-1}} \mathcal{L}(y, \lambda, \mu)\right)^{\mathsf{T}} \Big)^{\mathsf{T}} \end{aligned} \tag{5.98}$$

with respect to

$$\left(s^{\mathsf{T}}, w^{\mathsf{T}}\right)^{\mathsf{T}} = \left({s_0}^{\mathsf{T}}, \ldots, {s_{n_T}}^{\mathsf{T}}\,,\; {w_0}^{\mathsf{T}}, \ldots, {w_{n_T-1}}^{\mathsf{T}}\right)^{\mathsf{T}}, \tag{5.99}$$

which is done based on simultaneously perturbed variables on each multiple shooting interval. To explain this in more detail, we denote

$$\begin{aligned} \tilde{\mathcal{L}}_{s,i} := \Big(& \nabla_{s_0} \mathcal{L}(\tilde{y}_s^i, \lambda, \mu)^{\mathsf{T}}, \ldots, \nabla_{s_{n_T}} \mathcal{L}(\tilde{y}_s^i, \lambda, \mu)^{\mathsf{T}}, \\ & \nabla_{w_0} \mathcal{L}(\tilde{y}_s^i, \lambda, \mu)^{\mathsf{T}}, \ldots, \nabla_{w_{n_T-1}} \mathcal{L}(\tilde{y}_s^i, \lambda, \mu)^{\mathsf{T}} \Big)^{\mathsf{T}}, \end{aligned} \tag{5.100}$$

with

$$\tilde{y}_s^i = \left((s_0 + e_i \cdot \Delta)^\intercal, \dots, (s_{n_T} + e_i \cdot \Delta)^\intercal, w^\intercal, \bar{q}^\intercal \right)^\intercal \tag{5.101}$$

and $i = 0, \dots, n_x - 1$, where e_i denotes a unit vector of appropriate size with the i-th entry being one and $\Delta \in \mathbb{R}$ describes the constant perturbation. We furthermore denote

$$\begin{aligned} \tilde{\mathcal{L}}_{w,j} := \Big(& \nabla_{s_0} \mathcal{L}(\tilde{y}_w^j, \lambda, \mu)^\intercal, \dots, \nabla_{s_{n_T}} \mathcal{L}(\tilde{y}_w^j, \lambda, \mu)^\intercal \\ & \nabla_{w_0} \mathcal{L}(\tilde{y}_w^j, \lambda, \mu)^\intercal, \dots, \nabla_{w_{n_T-1}} \mathcal{L}(\tilde{y}_w^j, \lambda, \mu)^\intercal \Big)^\intercal, \end{aligned} \tag{5.102}$$

with

$$\tilde{y}_w^j = \left(s^\intercal, (w_0 + e_j \cdot \ \Delta)^\intercal, \dots, (w_{n_T-1} + e_j \cdot \ \Delta)^\intercal, \bar{q}^\intercal \right)^\intercal \tag{5.103}$$

and $j = 0, \dots, (n_u \cdot n_l) - 1$. We then compute

$$\left[\bar{\mathcal{L}}_1\right]_{ki} := \frac{\left[\tilde{\mathcal{L}}_{s,i}\right]_k - \left[\nabla_{(s,w)} \mathcal{L}\right]_k}{\Delta}, \quad \forall i = 0, \dots, n_x - 1, \ k = 0, \dots, n_s - 1 \tag{5.104}$$

and

$$\left[\bar{\mathcal{L}}_2\right]_{kj} := \frac{\left[\tilde{\mathcal{L}}_{w,j}\right]_k - \left[\nabla_{(s,w)} \mathcal{L}\right]_k}{\Delta}, \quad \forall j = 0, \dots, (n_u \cdot n_l) - 1, \ k = 0, \dots, n_s + n_w - 1, \tag{5.105}$$

which leads to the following finite difference approximations of the requires derivatives:

$$\bar{\mathcal{L}}_1 \approx \begin{pmatrix} \nabla^2_{s_0} \mathcal{L} \\ \vdots \\ \nabla^2_{s_{n_T}} \mathcal{L} \end{pmatrix} \in \mathbb{R}^{n_s} \times \mathbb{R}^{n_x} \tag{5.106}$$

and

$$\bar{\mathcal{L}}_2 \approx \left(\begin{array}{c} \nabla^2_{s_0, w_0} \mathcal{L} \\ \vdots \\ \nabla^2_{s_{n_T-1}, w_{n_T-1}} \mathcal{L} \\ \nabla^2_{s_{n_T}, w_{n_T-1}} \mathcal{L} \\ \hline \nabla^2_{w_0} \mathcal{L} \\ \vdots \\ \nabla^2_{w_{n_T-1}} \mathcal{L} \end{array} \right) \in \mathbb{R}^{(n_s + n_w)} \times \mathbb{R}^{(n_u \cdot n_l)}. \tag{5.107}$$

This means that with $\bar{\mathcal{L}}_1$ and $\bar{\mathcal{L}}_2$, we have efficiently computed approximations of the remaining blocks in $J2$, which have to be stored in $J2$ as shown in (5.77).

Computation of Block $J3$

Block $J3$ contains the derivative of the complementarity constraint with respect to y. For the computation of Block $J3$, we use the symbolic expression provided in formula (5.85).

Computation of Blocks $J4$ and $J5$

Blocks $J4$ and $J5$ are dense blocks without a special structure because of the global parameter p with

$$J4 := \left(\nabla_{(\gamma,p)} C^{\text{eq}}\right)^{\intercal} \quad \text{and} \quad J5 := \left(\nabla_{y(\gamma,p)} \mathcal{L}(y,\lambda,\mu)\right)^{\intercal}. \tag{5.108}$$

Block $J4$ is computed by solving a system of variational differential equations based on the derivatives of the ODE's right-hand side with respect to x and p, which are obtained by using symbolic expressions (which are available for many applications and can be provided by the user) or automatic differentiation. C^{eq} does not depend on γ and the corresponding block is set to zero. The derivative of $\nabla_y \mathcal{L}(y,\lambda,\mu)$ with respect to p and γ is built based on finite differences. As already mentioned above, we note that for most applications, $\nabla_y \mathcal{L}(y,\lambda,\mu)$ does not contain derivatives computed using finite differences.

Computation of Blocks $J8$ and $J9$

Block $J8$ contains derivatives of $\nabla_y \mathcal{L}(y,\lambda,\mu)$ with respect to μ. The symbolic expression is given in formula (5.86), which is used to fill Block $J8$ in J. For Block $J9$, we also use the trivial symbolic derivative provided in (5.87). Block $J9$ contains the derivative of $\sum_{i=1}^{n_\mathcal{M}} \gamma_i - 1$ with respect to γ.

Sparse constraint Jacobian

In the implementation of the algorithm derived in this Chapter, the constraint Jacobian J is stored in a sparse format based on the precise structure of each of the Blocks $J1 - J9$ analyzed in this Chapter. All zero blocks and entries in J are neglected (cf. Figure 5.3).

The previous paragraph explains the computation of the constraints and the constraint Jacobian of (5.90), and it remains to discuss the computation of the derivative of the objective $\nabla_z \left(\frac{1}{2}\|\mathfrak{F}(z)\|_2^2\right)$, the Hessian approximation $\nabla_z \mathfrak{F}(z)^{\intercal} \nabla_z \mathfrak{F}(z)$, and the computation of $\left(\nabla_z \mathfrak{F}(z)^{\intercal} \nabla_z \mathfrak{F}(z)\right) \cdot d$. Therefore, we consider the following assumptions, which are often satisfied for practical applications:

> **Assumption I:** The measurement grid is the same grid as the multiple shooting grid, denoted by $t_0 < t_1 < \ldots < t_{n_T} = T$.
>
> **Assumption II:** Differential states are measured directly, which means that for all $i = 0, \ldots, n_T$ and $j = 0, \ldots, n_x - 1$ in (5.70), we have:
>
> $$h_j(x(t_i), q, p) = s_{ij}. \tag{5.109}$$

We note that Assumption II can be weakened such that it requires that just some of the differential states are measured. This still leads to the desired structure in the Hessian approximation of (5.90). However, for a simpler presentation, we assume that all states are measured. The gradient of the objective of (5.90) is then given by

$$\nabla_z \left(\tfrac{1}{2}\|\mathfrak{F}(z)\|_2^2\right) = \mathfrak{F}(z). \tag{5.110}$$

Furthermore, Assumptions I-II lead to a diagonal Hessian approximation $\nabla_z \mathfrak{F}(z)^\intercal \nabla_z \mathfrak{F}(z)$ with n_s entries on the diagonal and the remaining entries on the diagonal are zero:

$$\begin{pmatrix} 1/\sigma_{0,0}^2 & & & & & \\ & \ddots & & & & \\ & & 1/\sigma_{n_T,n_x-1}^2 & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{pmatrix}. \tag{5.111}$$

The Hessian approximation (5.111) times a vector $d \in \mathbb{R}^{n_z}$ is then given by

$$\left(\nabla_z \mathfrak{F}(z)^\intercal \nabla_z \mathfrak{F}(z)\right) \cdot d = \begin{pmatrix} \frac{d_0}{\sigma_{0,0}^2} \\ \vdots \\ \frac{d_{n_s}}{\sigma_{n_T,n_x-1}^2} \\ 0 \\ \vdots \\ 0 \end{pmatrix}. \tag{5.112}$$

This means that almost no computational cost is needed to compute the derivative of the objective $\nabla_z \left(\frac{1}{2}\|\mathfrak{F}(z)\|_2^2\right)$, the Hessian approximation $\nabla_z \mathfrak{F}(z)^\intercal \nabla_z \mathfrak{F}(z)$, and $\left(\nabla_z \mathfrak{F}(z)^\intercal \nabla_z \mathfrak{F}(z)\right) \cdot d$ if Assumptions I-II are satisfied. In the implementation of the algorithm described in this chapter (cf. Chapter 9), we distinguish between a general least-squares objective, and an objective satisfying a weakened form of the case described above (Assumptions I-II).

We close this section with a final note. In this chapter, the numerical treatment of the complementarity constraint in the NLP (5.70) is not discussed. As already mentioned before, complementarity constraints need a special treatment since they violate many common constraint qualifications. The complementarity constraint poses a significant challenge when solving dynamic hierarchical optimization problems with a direct approach and its treatment within this framework is investigated in detail in Chapter 8, where we propose a tailored lifting technique.

5.6. Summary

In this chapter, we described a direct efficient all-at-once approach for hierarchical dynamic optimization problems of the form (5.2) based on three main steps:

- The parameterization of the differential states, and the discretization of the controls and the mixed control-state constraints.
- The formulation of necessary conditions for optimality of the parameterized and discretized lower-level OCP followed by replacing the lower-level problem by the optimality conditions.
- The solution of the resulting highly structured NLP with a structure-exploiting GGN method.

We discussed several challenges of the method derived in this chapter including:

- The efficient and reliable computation of higher-order derivatives/sensitivities.
- The derivation of a structure-exploiting algorithm for solving (5.70) based on the analysis of the NLP's structure inherited from the discretization/parameterization and the bilevel setting of (5.2).

The algorithm proposed in this chapter is implemented in the software package ParaOCP, which is described in detail in Chapter 9. We furthermore provide benchmark tests for our algorithm using ParaOCP for illustrative examples in Chapter 14. The algorithm and its implementation are also used to solve large real-world problems identifying optimal control gait models for human locomotion in Chapter 15.

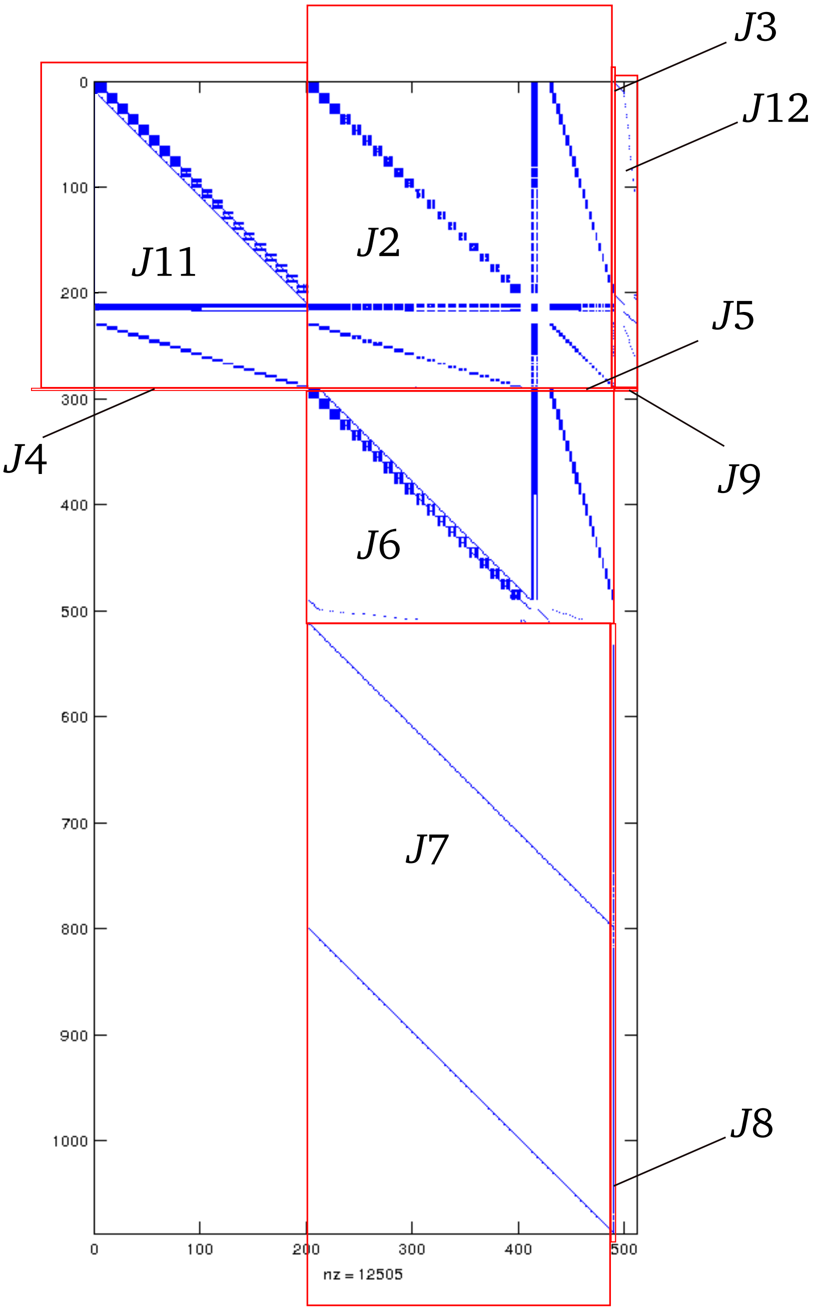


Figure 5.3.: Structure of $J^{\intercal}$ for a hierarchical dynamic optimization problem of type (5.1) with $n_x = 10$, $n_T = 20$, $n_u = 3$, $n_{\bar{q}} = 18$ $n_p = 1$ and $n_{\tilde{r}^{\text{eq}}} = 21$. The number of nonzeros is 12505.

Chapter 6.

An indirect all-at-once approach for hierarchical dynamic optimization

In this chapter, we consider an all-at-once approach for hierarchical dynamic optimization problems with an indirect treatment of the lower-level problem based on Pontryagin's maximum principle. We furthermore discuss numerical methods for testing second-order sufficient conditions for the solution of a specific class of optimal control problems based on the Riccati approach [MO02]. Numerical results and a discussion of the methods described in this chapter are provided for an illustrative example. This chapter is based on the work in [Hat08] and [HSB12].

6.1. Pontryagin's maximum principle to derive necessary optimality conditions

In this chapter, we describe an all-at-once approach for hierarchical dynamic optimization problems with an indirect treatment of the lower-level problem based on Pontryagin's maximum principle. The main structure of the method is illustrated in Figure 6.1.

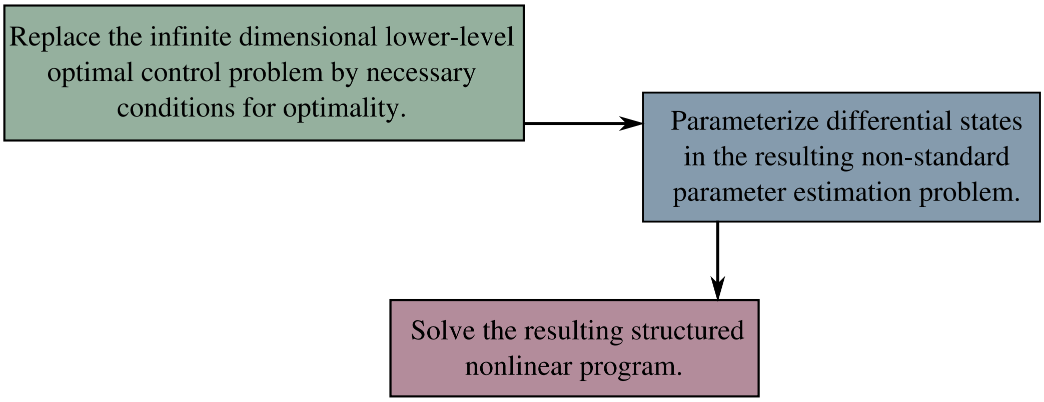

Replace the infinite dimensional lower-level optimal control problem by necessary conditions for optimality.
Parameterize differential states in the resulting non-standard parameter estimation problem.
Solve the resulting structured nonlinear program.

Figure 6.1.: The main steps of the all-at-once approach for hierarchical dynamic optimization problems with an indirect treatment of the lower-level problem.

Comparing Figure 6.1 and Figure 5.1, which illustrates the main steps of an all-at-once approach for hierarchical dynamic optimization problems with a direct treatment of the lower-level problem, shows that the approach in this chapter permutes parameterization/discretization and optimization. A direct treatment of the lower-level problem means that the lower-level optimal control problem (OCP) is first parameterized/discretized, and then we formulate necessary conditions for optimality. In this chapter, the two steps are interchanged – we first formulate necessary conditions for optimality of the infinite-dimensional lower-level OCP and then replace the lower-level problem by these conditions, and in the second step, the resulting parameter estimation problem is parameterized/discretized. We now describe the approach illustrated in Figure 6.1.

In Section 3.2, we state Pontryagin's maximum principle for deriving necessary conditions for optimality for OCPs of the following type:

$$\begin{array}{lrcll}
\underset{x,u}{\text{minimize}} & \multicolumn{3}{l}{\displaystyle\sum_{i=1}^{n_{\mathcal{M}}} \gamma_i \phi_{\mathcal{M}}^i(x(T),p)} & \\
\text{subject to} & \dot{x}(t) & = & f(x(t),u(t),p), & t \in [t_0,T] \\
 & 0 & = & r^{\mathrm{eq}}(x(t_0),x(T),p) & \\
 & u(t) & \in & \mathcal{U} := \{u(t) \in \mathbb{R} \mid \bar{c}(u(t),p) \geq 0\} &
\end{array} \tag{6.1}$$

with the system time $t \in [t_0, T]$, differential states x mapping to $\mathbb{R}^{n_x}$, a scalar control u, the Mayer terms $\phi_{\mathcal{M}}^i(\cdot) \in \mathbb{R}$ with $i = 1, \ldots, n_{\mathcal{M}}$, the ordinary differential equation's (ODE) right-hand side $f(\cdot) \in \mathbb{R}^{n_x}$ and boundary conditions $r^{\mathrm{eq}}(\cdot) \in \mathbb{R}^{n_{\mathrm{eq}}}$. f is assumed to be piecewise Lipschitz continuous. The parameters $p \in \mathbb{R}^{n_p}$ and $\gamma \in \mathbb{R}^{n_{\mathcal{M}}}$ are given quantities and no variables in (6.1). Problem (6.1) does not include path constraints. The function $\bar{c}(\cdot)$ describes the control constraints and maps to $\mathbb{R}^{n_{\bar{c}}}$. We focus on OCP (6.1) instead of the general setting (3.7) since deriving necessary optimality conditions and the corresponding boundary value problem (BVP) based on Pontryagin's maximum principle for general OCPs including path constraints and several controls is highly complex and can hardly be done in an automatic procedure. We briefly repeat the assumptions on the OCP (6.1) (cf. Section 3.2):

- We assume the absence of a singular control, which means that the control $u(t)$ is uniquely determined by maximizing the Hamiltonian (defined below).
- The regularity condition $\nabla_u \bar{c}_i(u(t)) \neq 0$ for active components $i \in \{1, \ldots, n_{\bar{c}}\}$ of $\bar{c}$ is required.
- We assume the strict Legendre condition, which requires the second derivative of the augmented Hamiltonian (defined below) with respect to u to be negative definite on boundary arcs (where boundary arcs are subintervals of I with $\bar{c}_i(u(t)) = 0$ for at least one component $i \in \{1, \ldots, n_{\bar{c}}\}$).

Please note that the latter conditions are only assumed in this chapter – in Chapter 5, we consider a very general class of OCPs. Necessary optimality conditions for (6.1) based on the ones derived in Section 3.2 are given by the following equations:

$$\begin{array}{lcll}
\dot{x}(t) & = & f(x(t),u(t),p), & t \in [t_0,T] \\
\dot{\lambda}(t) & = & -\nabla_x H(x(t),\lambda(t),u(t),\mu(t),p), & t \in [t_0,T] \\
0 & = & r^{\mathrm{eq}}(x(t_0),x(T),p) & \\
\lambda(t_0) & = & \nabla_{x_0}\tilde{\Phi}, \quad \lambda(T) = -\nabla_{x_T}\tilde{\Phi} & \\
0 & \leq & \bar{c}(u(t),p) & \\
0 & \leq & \mu(t), \quad 0 = \quad \mu(t)^{\mathsf{T}}\bar{c}(u(t),p) & \\
0 & = & \nabla_u H(x(t),\lambda(t),u(t),\mu(t),p), &
\end{array} \tag{6.2}$$

where the function $\tilde{\Phi} := \alpha_0 \phi_{\mathcal{M}}(x(T), p) + \alpha^{\intercal}(r^{\text{eq}}(x(t_0), x(T), p))$ denotes the augmented Lagrangian and $H^0(x(t), \lambda(t), u(t), p) := \lambda(t)^{\intercal} f(x(t), u(t), p)$ describes the Hamiltonian. The augmented Hamiltonian is given by

$$H(x(t), \lambda(t), u(t), \mu(t), p) := H^0(x(t), \lambda(t), u(t), p) - \mu(t)^{\intercal} \bar{c}(u(t), p). \tag{6.3}$$

The variables $\alpha_0 \in \mathbb{R}$ and $\alpha \in \mathbb{R}^{n_{eq}}$ are Lagrange multipliers, and $\mu : [t_0, T] \to \mathbb{R}^{n_{\bar{c}}}$, $\lambda : [t_0, T] \to \mathbb{R}^{n_x}$ are adjoint infinite-dimensional variables.

We can now realize the first step of the method illustrated in Figure 6.1 and replace the OCP (6.1) by the infinite-dimensional necessary conditions for optimality (6.2) in the following hierarchical dynamic optimization problem:

$$\begin{array}{lllll}
\underset{\substack{x,u\\ \gamma,p}}{\text{minimize}} & \multicolumn{4}{l}{\displaystyle\frac{1}{2}\sum_{i=0}^{n_m}\sum_{j=1}^{n_h}\frac{(h_j(x(t_i^m),p)-\eta_i^j)^2}{\sigma_{ij}^2}} \\
\text{subject to} & \underset{x,u}{\text{minimize}} & \multicolumn{3}{l}{\displaystyle\sum_{i=1}^{n_{\mathcal{M}}}\gamma_i\phi_{\mathcal{M}}^i(x(T),p)} \\
 & \text{subject to} & \dot{x}(t) & = f(x(t),u(t),p), & t\in[t_0,T] \\
 & & 0 & = r^{\text{eq}}(x(t_0),x(T),p) & \\
 & & u(t) & \in \mathcal{U} := \{u(t)\in\mathbb{R} \mid \bar{c}(u(t),p)\geq 0\} & \\
 & \multicolumn{4}{l}{\sum_{i=1}^{n_{\mathcal{M}}}\gamma_i = 1,\ \gamma_i\geq 0\ \forall i,\ i=1,\dots,n_{\mathcal{M}}} \\
 & \multicolumn{4}{l}{b^{\text{lower}_p}\ \leq\ p\ \leq\ b^{\text{upper}_p},}
\end{array} \tag{6.4}$$

which leads to the non-standard parameter estimation problem:

$$\begin{array}{llll}
\underset{\substack{x,\lambda,\alpha,\mu\\ \gamma,p}}{\text{minimize}} & \multicolumn{3}{l}{\displaystyle\frac{1}{2}\sum_{i=0}^{n_m}\sum_{j=1}^{n_h}\frac{(h_j(x(t_i^m),p)-\eta_i^j)^2}{\sigma_{ij}^2}} \\
\text{subject to} & \dot{x}(t) & = f(x(t),u^*(t),p), & t\in[t_0,T] \\
 & \dot{\lambda}(t) & = -\nabla_x H(x(t),\lambda(t),u^*(t),\mu(t),p), & t\in[t_0,T] \\
 & 0 & = r^{\text{eq}}(x(t_0),x(T),p) & \\
 & \lambda(t_0) & = \nabla_{x_0}\tilde{\Phi},\quad \lambda(T) = -\nabla_{x_T}\tilde{\Phi} & \\
 & \sum_{i=1}^{n_{\mathcal{M}}}\gamma_i & = 1,\ \gamma_i\geq 0\ \forall i,\ i=1,\dots,n_{\mathcal{M}} & \\
 & b^{\text{lower}_p} & \leq\ p\ \leq\ b^{\text{upper}_p}, &
\end{array} \tag{6.5}$$

where $u^*(t)$ satisfies

$$\begin{array}{lll}
0 & \leq & \bar{c}(u^*(t),p) \\
0 & \leq & \mu(t),\quad 0 = \quad \mu(t)^{\intercal}\bar{c}(u^*(t),p) \\
0 & = & \nabla_u H(x(t),\lambda(t),u^*(t),\mu(t),p).
\end{array} \tag{6.6}$$

The size of the system of ODEs in (6.5) is doubled. Compared to (6.1), we have n_x additional ODEs in the infinite-dimensional dual states $\lambda(\cdot) \in \mathbb{R}^{n_x}$. We furthermore have additional boundary conditions $\lambda(t_0) = \nabla_{x_0}\tilde{\Phi}$ and $\lambda(T) = -\nabla_{x_T}\tilde{\Phi}$, so-called *transversality conditions* for the dual states λ. The control function u is eliminated in (6.5) and no variable anymore. In the resulting NLP (5.70) of the direct all-at-once approach for hierarchical dynamic optimization problems with a direct treatment of the lower-level problem described in Chapter 5, we have a complementarity constraint arising from the KKT conditions. The infinite-dimensional problem (6.5) does also include a similar constraint – the infinite-dimensional complementarity constraint

$$0 \leq \bar{c}(u(t), p), \ \ 0 \leq \mu(t), \ \ 0 = \mu(t)^{\intercal} \bar{c}(u(t), p). \tag{6.7}$$

Furthermore, as in the all-at-once approach for hierarchical dynamic optimization problems with a direct treatment of the lower-level problem described in Chapter 5, the one-level problem (6.5) includes first-order derivatives, and second-order sensitivities are needed to solve (6.5) based on multiple shooting and a Generalized Gauß-Newton method. This means that the main difficulties of the method proposed in Chapter 5 persist. One difference of the indirect treatment of the lower-level problem compared to the direct treatment of the lower-level problem is the size of the resulting NLP. With the indirect treatment of the lower-level problem, we have less variables in the NLP since the control is eliminated and we do not have the variable w, which describes the control discretization in Chapter 5.

For a specific practical problem, one usually manually derives an expression for the control function $u(t)$ with $t \in [t_0, T]$ from the conditions (6.6) and then replaces $u(t)$ by this expression in (6.5). After eliminating the control u in (6.5), the primal and dual states x and λ are parameterized using multiple shooting (cf. Section 3.1). This leads to a nonlinear program (NLP), which is then solved with a Generalized Gauß-Newton method based on the one described in Section 2.2.2.

In general, we observe that using Pontryagin's maximum principle to derive necessary optimality conditions of the lower-level OCP may lead to a highly complex and numerically challenging parameter estimation problem with possibly jumps or discontinuities in the ODE's right-hand side (due to the control function, which may be non-differentiable or even discontinuous). The OCP (6.1) we consider in this chapter only includes a scalar control with pure control constraints. For the practical hierarchical dynamic optimization problem we intend to solve in this thesis (cf. Chapter 15), it is indispensable to allow higher-dimensional control functions and mixed control-path constraints. However, for general hierarchical dynamic optimization problems like (4.1), the derivation of necessary conditions for optimality and the corresponding BVP for the lower-level problem based on Pontryagin's maximum principle is highly complex and can hardly be done (cf. [Boc81b]). This is why we mainly focus on a direct treatment of the lower-level problem in this thesis.

The advantage of the all-at-once approach for hierarchical dynamic optimization problems with an indirect treatment of the lower-level problem is that we obtain high-accuracy solutions since we derive an analytic expression for the control function. Furthermore, the indirect treatment of the lower-level problem often allows interesting insights. Based on conditions (6.2), we can, e.g., test second-order sufficient conditions (SOSCs) of the lower-level problem in the solution of (6.5). In the hierarchical dynamic optimization method described in this chapter, but also in the method described in Chapter 5, we replace the lower-level problem by first-order optimality conditions. For general nonlinear OCPs, this is not an equivalent transformation. It might happen that the solution of (6.5) leads to a stationary point of the lower-level problem in (6.4), which is not a local minimum, or we could even end up in a maximum of the lower-level problem. Even though this is unlikely for good initial data, based on the method described in this section, we can test SOSCs of the lower-level problem of (6.4) in the solution of (6.5), which is described in the next section for an illustrative example including an OCP of type (6.1).

	γ_1	γ_2
Setting 1	1/2	1/2
Setting 2	6/7	1/7
Setting 3	16/17	1/17

Table 6.1.: Different settings for γ_1 and γ_2.

6.2. A Riccati approach for testing second-order sufficient conditions for the rocket car example and the solution of the corresponding hierarchical dynamic optimization problem

In this section, we start with deriving necessary conditions for optimality based on Pontryagin's maximum principle for an OCP of type (6.1), which describes the optimal movement of a rocket car with non-constant mass and including friction. We then solve the hierarchical dynamic optimization problem (6.4) for the rocket car example for simulated measurements. Based on the computed solution, we use a Riccati approach [MO02] for testing second-order sufficient conditions of the lower-level OCP. The following investigation can be found in [HSB12].

The objective function of the OCP is a combination (a weighted sum) of energy consumption and travel time. Our goal is to identify the weighting factors in the objective function, and to estimate unknown system parameters that are contained in the differential equations by fitting the model to observation data. We use generated pseudo measurements since they allow a better investigation of the proposed methods. We start by describing the OCP in detail. The equations of motion are given by the following ODE:

$$\dot{x}(t) = \begin{pmatrix} \dot{z}(t) \\ \dot{v}(t) \\ \dot{m}(t) \end{pmatrix} = \begin{pmatrix} q_0 v(t) \\ q_0 \left(u(t) - p_1 v(t)^2\right)/m(t) \\ -q_0 p_2 u(t)^2 \end{pmatrix} =: f(x(t), u(t), q_0, p), \tag{6.8}$$

where $q_0 \in \mathbb{R}$ is a control value representing the free end time after a time transformation on the fix time horizon $[0,1]$ (cf. [HSB12]). In (6.1), the control value q_0 would correspond to a differential state with initial value q_0 and a right-hand side equal to zero. $t \in [0,1]$ is the system time, $x := (x_1, x_2, x_3)^\intercal := (z, v, m)^\intercal$ are the differential states, where $z(t) \in \mathbb{R}$ describes the position, $v(t) \in \mathbb{R}$ the velocity and $m(t) \in \mathbb{R}$ the mass of the car. Note that the mass is time dependent. $p = (p_1, p_2)^\intercal$ are model parameters with the true value $p_1 = p_2 = 0.1$. The term $(p_1 v(t)^2)/m(t)$ describes Newton friction. The objective function $\Phi := -\gamma_1\ m(1) + \gamma_2\ q_0$ is a linear combination of Mayer terms. The first term describes energy consumption (the mass at the end of the time horizon) and is weighted with γ_1 and the second term denotes travel time and is weighted with γ_2. We investigate three different settings for the weights $\gamma = (\gamma_1, \gamma_2)^\intercal$ shown in Table 6.1.

The scalar control function $u(t)$ has to lie in the interval $[-1, 1]$. Furthermore, we have

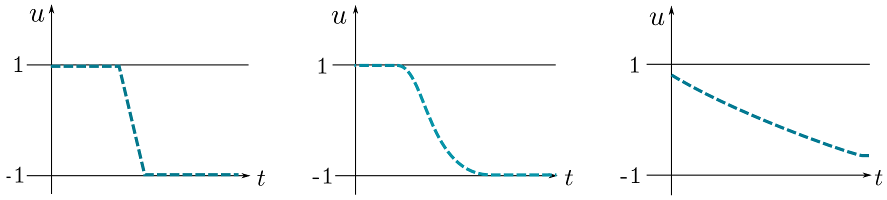


Figure 6.2.: Illustration of the optimal control $u(t)$ for Setting 1, 2, and 3.

start and end values for the differential states. This leads to the following OCP:

$$\begin{array}{lrcll}\underset{x,u,q_0}{\text{minimize}} & \multicolumn{3}{l}{-\gamma_1\, m(1) + \gamma_2\, q_0 =: \Phi} & \\ \text{subject to} & 0 & = & \dot z(t) - q_0 v(t), & t \in [0,1] \\ & 0 & = & \dot v(t) - q_0(u(t) - p_1 v(t)^2)/m(t), & t \in [0,1] \\ & 0 & = & \dot m(t) + q_0 p_2 u(t)^2, & t \in [0,1] \\ & 0 & = & z(0), \quad 0 = v(0), \quad 0 = m(0) - 1 & \\ & 0 & = & z(1) - 10, \quad 0 = v(1) - 1 & \\ & 0 & \le & 1 - u(t) =: \bar c_1(u(t)), & t \in [0,1] \\ & 0 & \le & 1 + u(t) =: \bar c_2(u(t)), & t \in [0,1].\end{array} \tag{6.9}$$

As illustrated in Figure 6.2, the different settings for the weights γ lead to different solutions and in particular, they lead to different types of control functions. For Setting 1, we obtain a control function that is close to a bang-bang control. Setting 3 implies a control function which lies completely in the interior of the control region. For Setting 2, we get a control function which is partly on the boundary, and partly in the interior of the control region.

We generate measurements for the position z, the velocity v, and the mass m on a time grid $0 = t_0 < \cdots < t_{n_T} = 1$. Therefore, we add normally distributed random numbers with mean 0 and standard deviation 0.04 to the solution $x^*(t)$ of the OCP (6.9). This is done for all three settings. For each setting, we then obtain the simulated measurements

$$\eta = \left[\left(\eta_0^1,\ \eta_0^2,\ \eta_0^3\right)^\intercal,\ \ldots,\ \left(\eta_{n_T}^1,\ \eta_{n_T}^2,\ \eta_{n_T}^3\right)^\intercal\right]^\intercal \in \mathbb{R}^{(n_T+1)} \times \mathbb{R}^3. \tag{6.10}$$

The hierarchical nonlinear constrained parameter estimation problem is then:

$$\begin{array}{lll}\underset{\substack{x,u,q_0,\\ p,\gamma}}{\text{minimize}} & \multicolumn{2}{l}{\dfrac{1}{2}\displaystyle\sum_{i=0}^{n_T}\sum_{j=1}^{3}\frac{(x_j(t_i) - \eta_i^j)^2}{\sigma_{ij}^2}} \\ \text{subject to} & \underset{x,u,q}{\text{minimize}} & -\gamma_1\, m(1) + \gamma_2\, q_0 \\ & \text{subject to} & 0 = \dot z(t) - q_0 v(t), \qquad t \in [0,1] \\ & & 0 = \dot v(t) - q_0(u(t) - p_1 v(t)^2)/m(t), \qquad t \in [0,1] \\ & & 0 = \dot m(t) + q_0 p_2 u(t)^2, \qquad t \in [0,1] \\ & & 0 = z(0), \quad 0 = v(0), \quad 0 = m(0) - 1 \\ & & 0 = z(1) - 10, \quad 0 = v(1) - 1 \\ & & 0 \le 1 - u \qquad t \in [0,1] \\ & & 0 \le 1 + u \qquad t \in [0,1] \\ & 0 \le \gamma_1, \gamma_2, & \\ & 0 = \gamma_1 + \gamma_2 - 1, & \end{array} \tag{6.11}$$

with $\sigma_{ij} = 0.04$ for $i = 0, \ldots, n_T$ and $j = 1, 2, 3$. Note that the last constraint is for avoiding redundancy. We start by using Pontryagin's maximum principle to formulate necessary optimality conditions of (6.9). The Hamiltonian of (6.9) is given by

$$\begin{aligned} H(x(t), \lambda(t), u(t), \mu(t), q_0, p) &= H^0(x(t), \lambda(t), u(t), q_0, p) - \mu(t)^\intercal \bar{c}(u(t), p) \\ &= \lambda_1(t) q_0 v(t) + (\lambda_2(t) q_0 / m(t))(u(t) - p_1 v(t)^2) \\ &\quad -\lambda_3(t) q_0 p_2 u(t)^2 - \mu_1(t)(1 - u(t)) - \mu_2(t)(1 + u(t)) \end{aligned} \tag{6.12}$$

and the adjoint differential equations are

$$\dot{\lambda}_1(t) = 0 \tag{6.13}$$

$$\dot{\lambda}_2(t) = -\lambda_1(t) q_0 + \frac{2\lambda_2(t) q_0 p_1 v(t)}{m(t)} \tag{6.14}$$

$$\dot{\lambda}_3(t) = \lambda_2 q_0 \frac{u(t) - p_1 v(t)^2}{m(t)^2}. \tag{6.15}$$

It is well known that the solution of (6.9) has two boundary arcs – one with $\bar{c}_1(u(t)) = 0$ for $0 \le t \le \tau_1$, and one with $\bar{c}_2(u(t)) = 0$ and $\tau_2 \le t \le 1$ (where $0 \le \tau_1 < \tau_2 \le 1$). Between τ_1 and τ_2, there is an interior arc, where the control is computed from $\nabla_u H^0(x(t), \lambda(t), u(t), q_0, p) = \frac{\lambda_2(t) q_0}{m(t)} - 2\lambda_3(t) q_0 p_2 u(t) \equiv 0$:

$$u_{\text{int}}(x(t), \lambda(t), q_0, p) = \frac{\lambda_2(t)}{2m(t)\lambda_3(t) p_2} \quad \text{for} \quad \tau_1 \le t \le \tau_2, \tag{6.16}$$

and we have $\mu(t) \equiv 0$ from complementarity. The boundary control on the first boundary arc is computed from $\bar{c}_1(u(t)) = 0$ and the control on the second boundary arc is computed from $\bar{c}_2(u(t)) = 0$, which leads to the following control on [0,1]:

$$\tilde{u}^*(t) := u^*(x(t), \lambda(t), q_0, p) = \begin{cases} u_{\text{int}}(x(t), \lambda(t), q_0, p) & \text{if} \quad \tau_1 \le t \le \tau_2 \\ \ 1 & \text{if} \quad 0 \le t \le \tau_1 \\ -1 & \text{if} \quad \tau_2 \le t \le 1. \end{cases} \tag{6.17}$$

The multiplier μ on the first boundary arc with $0 \le t \le \tau_1$ can be computed from

$$H_u(x(t), \lambda(t), u(t), \mu(t), q_0, p) \equiv 0, \ \mu_2(t) \equiv 0, \ \text{ and } \ u(t) \equiv 1 \tag{6.18}$$

(and similarly for the second boundary arc), which leads to the following formula for μ:

$$\mu^*(x(t), \lambda(t), q_0, p) = \begin{cases} 0 & \text{if} \quad \tau_1 \le t \le \tau_2 \\ -\frac{\lambda_2(t) q_0}{m(t)} + 2\lambda_3(t) q_0 p_2 & \text{if} \quad 0 \le t \le \tau_1 \\ \frac{\lambda_2(t) q_0}{m(t)} + 2\lambda_3(t) q_0 p_2 & \text{if} \quad \tau_2 \le t \le 1. \end{cases} \tag{6.19}$$

The augmented objective function is given by

$$\tilde{\Phi} = \Phi + \alpha_2^\intercal r^{\text{eq}}(x(t_0), x(T), p) \tag{6.20}$$

$$= -\gamma_1 m(1) + \gamma_2 q_0 + \alpha_2^\intercal \begin{pmatrix} z(0) \\ v(0) \\ m(0) - 1 \\ z(1) - 10 \\ v(1) - 1 \end{pmatrix} \tag{6.21}$$

$$\begin{aligned} &= -\gamma_1 m(1) + \gamma_2 q_0 \\ &\quad + \alpha_{2,1} z(0) + \alpha_{2,2} v(0) + \alpha_{2,3}(m(0) - 1) + \alpha_{2,4}(z(1) - 10) + \alpha_{2,5}(v(1) - 1), \end{aligned} \tag{6.22}$$

Setting 1	q_0	$= 5.268999\mathrm{E}+00$,	Φ	$= 2.388591\mathrm{E}+00$
	$\lambda_1(0)$	$= 3.450745\mathrm{E}-01$,	$\lambda_1(1)$	$= 3.450745\mathrm{E}-01$
	$\lambda_2(0)$	$= 8.673559\mathrm{E}-01$,	$\lambda_2(1)$	$= -3.375312\mathrm{E}-01$
	$\lambda_3(0)$	$= -1.326436\mathrm{E}+00$,	$\lambda_3(1)$	$= 1.000000\mathrm{E}+00$
	$m(1)$	$= 4.918159\mathrm{E}-01$,	τ_1	$= 8.264790\mathrm{E}-01$
	τ_2	$= 8.802960\mathrm{E}-01$		
Setting 2	q_0	$= 5.956249\mathrm{E}+00$,	Φ	$= 2.589167\mathrm{E}-01$
	$\lambda_1(0)$	$= 7.140323\mathrm{E}-02$,	$\lambda_1(1)$	$= 7.140323\mathrm{E}-02$
	$\lambda_2(0)$	$= 1.883686\mathrm{E}-01$,	$\lambda_2(1)$	$= -9.690561\mathrm{E}-02$
	$\lambda_3(0)$	$= 4.551155\mathrm{E}-01$,	$\lambda_3(1)$	$= 8.571428\mathrm{E}-01$
	$m(1)$	$= 6.906386\mathrm{E}-01$,	τ_1	$= 2.421000\mathrm{E}-01$
Setting 3	q_0	$= 7.536531\mathrm{E}+00$,	Φ	$= -3.569777\mathrm{E}-01$
	$\lambda_1(0)$	$= 3.867384\mathrm{E}-02$,	$\lambda_1(1)$	$= 3.867384\mathrm{E}-02$
	$\lambda_2(0)$	$= 1.322845\mathrm{E}-01$,	$\lambda_2(1)$	$= -5.976373\mathrm{E}-02$
	$\lambda_3(0)$	$= 7.437162\mathrm{E}-01$,	$\lambda_3(1)$	$= 9.411764\mathrm{E}-01$
	$m(1)$	$= 8.503220\mathrm{E}-01$,		

Table 6.2.: Solution of the BVP resulting from Pontryagin's maximum principle applied to (6.9).

with $\alpha_2 \in \mathbb{R}^5$, where $\alpha_{2,i}$ denotes the ith component of α_2. The gradient of $\tilde{\Phi}$ with respect to $x(0), x(1)$ and q_0 is then

$$\begin{aligned} \nabla_{x(0)}\tilde{\Phi} &= (\alpha_{2,1} \;\; \alpha_{2,2} \;\; \alpha_{2,3})^\intercal, && (6.23)\\ \nabla_{x(1)}\tilde{\Phi} &= (\alpha_{2,4} \;\; \alpha_{2,5} \;\; -\gamma_1)^\intercal, && (6.24)\\ \nabla_{q_0}\tilde{\Phi} &= \gamma_2. && (6.25) \end{aligned}$$

The transversality conditions are given by

$$\begin{aligned} \lambda(t_0) &= (\alpha_{2,1} \;\; \alpha_{2,2} \;\; \alpha_{2,3})^\intercal, && (6.26a)\\ \lambda(1) &= -(\alpha_{2,4} \;\; \alpha_{2,5} \;\; -\gamma_1)^\intercal, && (6.26b)\\ \tilde{H}(1) &= \gamma_2, && (6.26c) \end{aligned}$$

where $\tilde{H}(1) := H(x(1), \lambda(1), u(1), \mu(1), q_0, p)/q_0$.

We solve the resulting BVP with the software package MUSCOD-II (described below) and $n_T = 20$. The junction times τ_1 and τ_2 are given by implicit conditions. A technique based on switching functions is used to numerically solve the discontinuous initial value problem problem (cf. [Kir06], [Boc81a]). Conditions and explanations describing when this approach is feasible can be found in, e.g., [Boc78]. The solutions for Setting 1-3 are shown in Table 6.2.

The solution for Setting 1,2 and 3 is plotted in Figure 6.3, 6.4 and 6.5 (the dashed line). Before we start solving the parameter estimation problem (6.11), we want to make sure that the solution of the lower-level OCP for all three settings, which are used to simulate measurements, does satisfy SOSCs. Therewith, we can guarantee that the solution of the OCP for each setting is a local minimum. To check SOSCs for (6.9), we follow the work of Maurer and Oberle [MO02]. We clearly have linearly independent gradients for active constraints in (6.9), and the modified strict Legendre-Clebsch condition holds

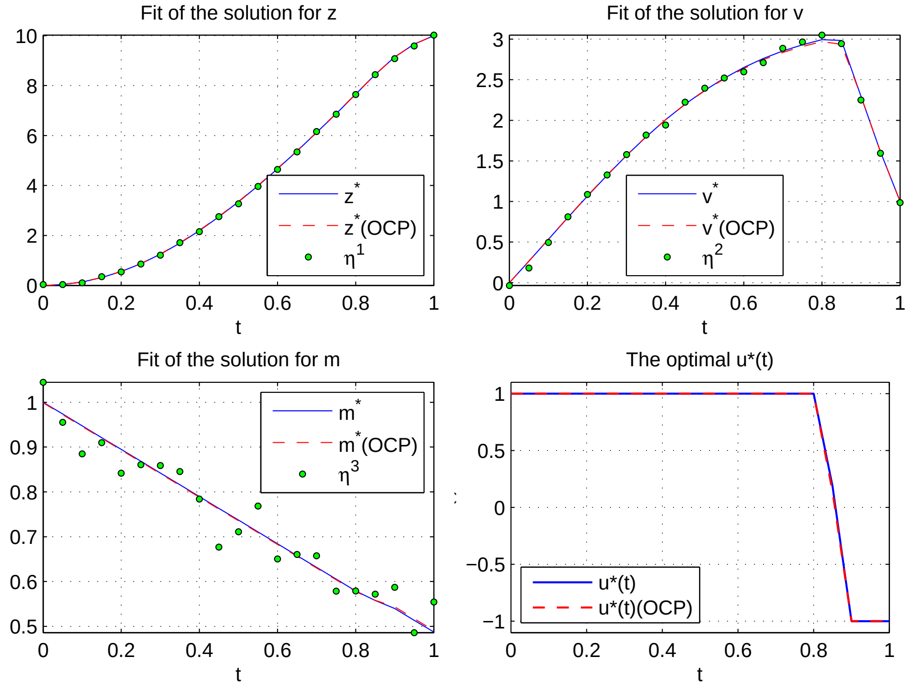


Figure 6.3.: The solution of (6.31) and of the corresponding OCP (as reference trajectory), and measurements η for Setting 1.

because $\nabla_{uu}H(x(t), \lambda(t), u(t), \mu(t), q_0, p) = -2\lambda_3(t)q_0p_2 < 0$ $(\lambda_3(t) > 0)$ on the interior arc where $\tau_1 \leq t \leq \tau_2$. It remains to show that the Riccati equations introduced in [MO02] have a bounded solution on $[0, 1]$, such that certain sign conditions hold. For the rocket car problem (6.9), the symmetric Riccati matrix $\tilde{Q} \in \mathbb{R}^{(n_x+1)\times(n_x+1)}$ is given by

$$\tilde{Q} = \begin{pmatrix} Q & R \\ R^\intercal & q \end{pmatrix} = \left(\begin{array}{ccc|c} q_{11} & q_{12} & q_{13} & r_1 \\ q_{12} & q_{22} & q_{23} & r_2 \\ q_{13} & q_{23} & q_{33} & r_3 \\ \hline r_1 & r_2 & r_3 & q_T \end{array}\right). \tag{6.27}$$

The strong Riccati equations are denoted as

$$\begin{aligned} \dot{Q} &= q_0[-Qf_x - f_x^\intercal Q - H_{xx} + (H_{xu} + Qf_u)(H_{uu})^{-1}(H_{xu} + Qf_u)^\intercal] \\ \dot{R} &= -Qf - q_0 f_x^\intercal R - (H_x^0)^\intercal + (H_{xu} + Qf_u)(H_{uu})^{-1}(H_u^0 + q_0 f_u^\intercal R)^\intercal \\ \dot{q}_T &= -2R^\intercal f + \tfrac{1}{q_0}(H_u^0 + q_0 f_u^\intercal R)(H_{uu})^{-1}(H_u^0 + q_0 f_u^\intercal R)^\intercal, \end{aligned} \tag{6.28}$$

where in this paragraph, we use f_x to denote the derivative of f with respect to x, i.e. $\nabla_x f$, for a compact presentation (and similarly for the other quantities). Furthermore, the dependency on t in (6.28) is skipped. The sign conditions in case of the OCP (6.9) are given by

$$q_T(0) > 0, \quad q_T(1) < 0, \quad \text{and} \quad q_{33}(1) < 0. \tag{6.29}$$

In some cases, the strong Riccati equations (6.28) fail to have a bounded solution satisfying the sign conditions, even though SOSCs are fulfilled. Therefore, modified or reduced Riccati

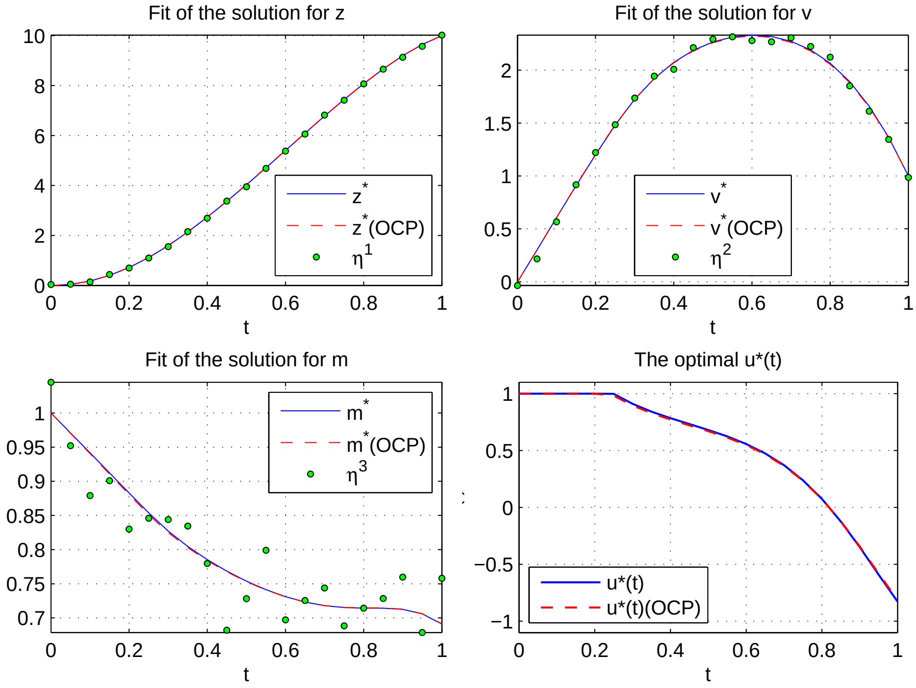


Figure 6.4.: The solution of (6.31) and of the corresponding OCP (as reference trajectory), and measurements η for Setting 2.

equations (see [MO02]) have been derived and are stated in the following for the case of pure control constraints and a boundary arc with as many controls as active constraints:

$$\begin{aligned} \dot{Q} &= q_0[-Qf_x - f_x^\intercal Q - H_{xx}] \\ \dot{R} &= -Qf - q_0 f_x^\intercal R - (H_x^0)^\intercal \\ \dot{q}_T &= -2R^\intercal f. \end{aligned} \tag{6.30}$$

For all three settings, we need to find a bounded solution of (6.30) on the boundary arcs $[0,\tau_1]$ and $[\tau_2,1]$, and of (6.28) on the interior arc $[\tau_1,\tau_2]$. With $q_{11}(1) = q_{12}(1) = q_{13}(1) = q_{22}(1) = q_{23}(1) = r_1(1) = r_2(1) = r_3(1) = q_T(1) = 0.1$ and $q_{33}(1) = q_T(1) = -0.1 < 0$ for all three settings, we were successful using the initial values shown in Table 6.3, which satisfy the sign conditions (6.29).

For Setting 1 we have $\|\tilde{Q}(t)\|_\infty \leq 1.075843\text{E}+00$, $\|\tilde{Q}(t)\|_\infty \leq 1.925011\text{E}+00$ for Setting 2, and $\|\tilde{Q}(t)\|_\infty \leq 1.474774\text{E}+00$ for Setting 3. In fact, it is even possible to find a bounded solution of the strong conditions (6.28) for interior and boundary arcs for all settings except from the boundary arc $[\tau_2,1]$ in Setting 3. Since SOSC have been verified for Setting 1-3 of (6.9), we now continue with solving the hierarchical problem (6.11). The explicit form of

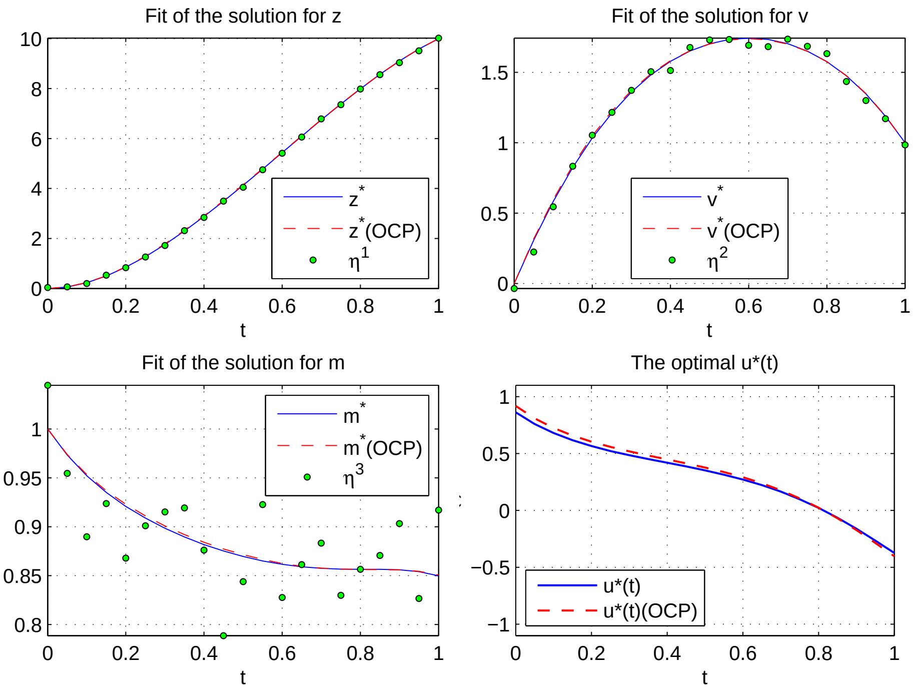


Figure 6.5.: The solution of (6.31) and of the corresponding OCP (as reference trajectory), and measurements η for Setting 3.

(6.5) for the rocket car OCP is given by:

$$\begin{array}{llll}
\underset{\substack{x,q,\\ \lambda,\gamma,p}}{\text{minimize}} & \frac{1}{2}\sum_{i=0}^{n_T}\sum_{j=1}^{3}\frac{(x_j(t_i)-\eta_i^j)^2}{\sigma_{ij}^2} & & \\
\text{subject to} & \dot{z}(t) = q_0 v(t), & t\in[0,1] & \\
& \dot{v}(t) = \frac{q_0(\tilde{u}^*(t)-p_1 v(t)^2)}{m(t)}, & t\in[0,1] & \\
& \dot{m}(t) = -q_0 p_2 \tilde{u}^*(t)^2, & t\in[0,1] & \\
& \dot{\lambda}_1(t) = 0, & t\in[0,1] & (6.31)\\
& \dot{\lambda}_2(t) = -\lambda_1(t)q_0 - 2\frac{(\lambda_2(t)q_0p_1v(t))}{m(t)}, & t\in[0,1] & \\
& \dot{\lambda}_3(t) = \lambda_2(t)q_0\frac{(\tilde{u}^*(t)-p_1v(t)^2)}{m(t)^2}, & t\in[0,1] & \\
& z(0) = 0, \quad z(1)=10 & & \\
& v(0) = 1, \quad v(1)=0, \quad m(0)=1 & & \\
& \gamma_1 = \lambda_3(1),\ \gamma_2 = \tilde{H}(1),\ \gamma_1,\gamma_2\geq 0,\ \gamma_1+\gamma_2=1, & &
\end{array}$$

where the control function $\tilde{u}^*(t)$ is defined in (6.17).

Problem (6.31) is solved with the software package MUSCOD-II ([Lei96]), which is based on the direct multiple shooting method ([BP84], [Pli81]). The software solves highly nonlinear problems with complex equality or inequality constraints, such as coupled and decoupled multi-point constraints on states, control functions and control values. The optimization problem with ODEs is reformulated as a large-scale NLP with a block sparse structure. The NLP problem is solved with a tailored sequential quadratic programming method with an active set strategy, which fully exploits the inherent structure. Furthermore, a Gauss-

Setting 1	$q_{11}(0)$	$= 9.787677\mathrm{E}-03$,	$q_{12}(0)$	$= 1.373429\mathrm{E}-02$	
	$q_{13}(0)$	$= 8.586854\mathrm{E}-03$,	$q_{22}(0)$	$= 1.411248\mathrm{E}-02$	
	$q_{23}(0)$	$= -1.423103\mathrm{E}-01$,	$q_{33}(0)$	$= -1.076012\mathrm{E}+00$	
	$r_1(0)$	$= 1.167103\mathrm{E}-01$,	$r_2(0)$	$= 1.124075\mathrm{E}-01$	
	$r_3(0)$	$= 5.550459\mathrm{E}-01$,	$q_T(0)$	$= 8.207909\mathrm{E}-01$	> 0
Setting 2	$q_{11}(0)$	$= 9.794009\mathrm{E}-03$,	$q_{12}(0)$	$= 1.383907\mathrm{E}-02$	
	$q_{13}(0)$	$= -2.310602\mathrm{E}-03$,	$q_{22}(0)$	$= 1.139493\mathrm{E}-01$	
	$q_{23}(0)$	$= -2.777279\mathrm{E}-01$,	$q_{33}(0)$	$= -1.925010\mathrm{E}+00$	
	$r_1(0)$	$= 1.240196\mathrm{E}-01$,	$r_2(0)$	$= 9.906369\mathrm{E}-02$	
	$r_3(0)$	$= 4.798097\mathrm{E}-01$,	$q_T(0)$	$= 9.547290\mathrm{E}-01$	> 0
Setting 3	$q_{11}(0)$	$= 9.935399\mathrm{E}-03$,	$q_{12}(0)$	$= 1.578702\mathrm{E}-02$	
	$q_{13}(0)$	$= 2.644530\mathrm{E}-03$,	$q_{22}(0)$	$= 1.976495\mathrm{E}-01$	
	$q_{23}(0)$	$= -2.947827\mathrm{E}-01$,	$q_{33}(0)$	$= -1.474774\mathrm{E}+00$	
	$r_1(0)$	$= 1.265753\mathrm{E}-01$,	$r_2(0)$	$= 6.112673\mathrm{E}-02$	
	$r_3(0)$	$= 3.308242\mathrm{E}-01$,	$q_T(0)$	$= 9.442405\mathrm{E}-01$	> 0

Table 6.3.: Solution of the Riccati problem.

Newton approach allows to efficiently treat least-squares objectives. Implicit switches in the ODE's right-hand side can also be handled by this software package (cf. [Kir06], [Boc81a]). For a discussion of a parameter estimation problem with a variable time horizon, we refer to Chapter 14. The measurements η are generated as described above. We have six differential equations, six boundary constraints, two additional constraints, a non-differentiable function u and therewith, a non-differentiable right-hand side of the ODE. The solution of (6.31) delivers estimates of γ_1, p_1 and p_2 that are shown for all three settings in Table 6.4, where the absolute error is the squared and summed up deviation between true values and estimates for γ_1, p_1 and p_2. The computations are done on a Intel(R) Core(TM) i7 920 (2.67GHz). The solution $x^* = (z^*, v^*, m^*)^{\mathsf{T}}$ of (6.31), the corresponding solution of the BVP and the generated measurements are shown in Figures 6.3, 6.4 and 6.5. For the adjoint states λ, we get initial values from the Lagrange multipliers of the solution of the OCP obtained by using the direct approach.

The optimal movement of a rocket car discussed in this section is revisited in Chapter 14, where we discuss the performance of our algorithm for hierarchical dynamic optimization problems with a direct treatment of the lower-level problem (cf. Chapter 5) for a set of illustrative benchmark examples, and compare it to alternative approaches. In this chapter, we have seen that our all-at-once approach for hierarchical dynamic optimization problems with an indirect treatment of the lower-level problem works for an OCP describing the optimal movements of a rocket car with simulated measurement data. Furthermore, we have used the necessary optimality conditions we derived for (6.9) to check SOSCs in the solution of (6.31) (cf. [Hat08]). However, the approach proposed in this chapter is highly complex and hardly possible for more general OCPs including multiple controls and path constraints. Hence, in the remainder of this thesis, we mainly focus on an all-at-once approach for hierarchical dynamic optimization problems with a direct treatment of the lower-level problem.

γ_1	**estimated** γ_1	**abs. error**	γ_2	**estimated** γ_2	**abs. error**
1/2	4.678379E − 01	3.216220E − 02	1/2	5.322162E − 01	3.221620E − 02
6/7	8.561771E − 01	9.657522E − 04	1/7	1.443822E − 01	9.657522E − 04
16/7	9.411264E − 01	5.004058E − 05	1/16	5.898735E − 02	5.004058E − 05

(a) Estimated γ_1 and γ_2 for all settings (row 1-3 show Setting 1-3 from Table 6.1).

	p_1	**estimated** p_1	**abs. error**	**obj. function**
Setting 1	0.1	9.772291E − 02	2.276091E − 03	8.760749E − 02
Setting 2	0.1	1.013627E − 01	1.363765E − 03	9.036917E − 02
Setting 3	0.1	8.900934E − 01	1.099065E − 02	8.743350E − 02

(b) Estimated p_1 and the objective function values for all settings.

	p_2	**estimated** p_2	**abs. error**	**CPU times (s)**
Setting 1	0.1	1.002243E − 01	2.243294E − 04	0.043
Setting 2	0.1	9.841098E − 02	1.589012E − 03	0.033
Setting 3	0.1	1.079109E − 01	7.910981E − 03	0.037

(c) Estimated p_2 and computational times for all settings.

Table 6.4.: The solution of (6.31) using Pontryagin's maximum principle.

Chapter 7.

Bilevel approaches for hierarchical dynamic optimization

In Section 4.4, we describe two main classes of approaches for hierarchical dynamic optimization problems: all-at-once/simultaneous approaches and bilevel approaches. In Chapters 5 and 6, we derive two all-at-once approaches for hierarchical dynamic optimization problems – one with a direct and one with an indirect treatment of the lower-level problem. In this chapter, we develop two alternative methods that belong to the class of bilevel approaches: a bundle method and a derivative-free optimization technique. The algorithms derived in this chapter allow us to compare the direct all-at-once approach for hierarchical dynamic optimization problems proposed in Chapter 5 to bilevel techniques. The performance of the all-at-once method and the bilevel approaches derived in this thesis is analyzed in Chapter 14 for a set of illustrative benchmark problems.

7.1. Introduction

The main difference between the all-at-once approach described in Chapter 5 and the bilevel approaches discussed in this chapter is that in the latter approaches, the bilevel structure remains and in each iteration of the upper-level problem, the lower-level problem is solved (i.e. feasibility is established). The main strategy is illustrated in Figure 7.1. The following

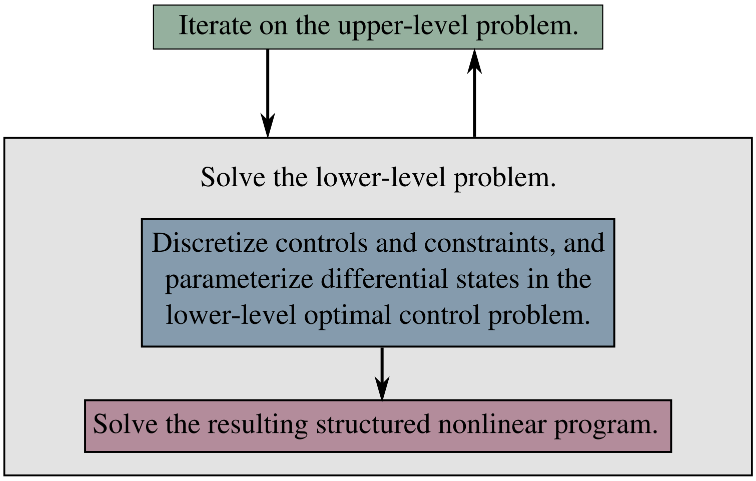

Iterate on the upper-level problem.
Solve the lower-level problem.
Discretize controls and constraints, and parameterize differential states in the lower-level optimal control problem.
Solve the resulting structured nonlinear program.

Figure 7.1.: The main steps of a bilevel approach for hierarchical dynamic optimization problems.

two sections describe bilevel techniques for hierarchical dynamic optimization problems. The first approach is based on a derivative-free optimization (DFO) technique used for the optimization of the upper-level problem of 4.1. The upper-level problem, however, depends on the solution (x, u, q) of the lower-level optimal control problem (OCP), which has to be computed each time the upper-level objective is evaluated. In the latter case, the first box in Figure 7.1 illustrates the DFO method. The second approach we consider in this chapter is based on a technique using first-order derivatives on the upper level – a bundle method. Again, each time the upper-level objective is evaluated, the lower-level OCP has to be solved. Both bilevel approaches are described in detail in the following two sections.

7.2. A derivative-free optimization approach

The DFO method relies on the approach described in [KLM$^+$10], which is tailored to bound-constrained optimization problems with a least-squares objective. As a general motivation for DFO, one can say that there are optimization problems, where the computation of derivatives is impractical and the need for derivative-free algorithms arises.

Popular DFO methods include genetic algorithms [Mit98] or direct search algorithms like the Nelder-Mead method [NM65]. In this thesis, we focus on the work described in [KLM$^+$10] for two reasons:

- The DFO method described in [KLM$^+$10] is tailored to least-squares problems and exploits the objective's structure.
- Wild and Moré have shown in a recent benchmarking study [MW09] that methods that form a smooth approximation model of the objective may be able to obtain better solutions in fewer evaluations.

We now briefly review the basic idea of the algorithm described in [KLM$^+$10]. Therefore, we consider the least-squares problem

$$\bar{F}(y) = \frac{1}{2}\sum_{i=0}^{n_m} F_i(y)^2 = \frac{1}{2}\|F(y)\|_2^2, \tag{7.1}$$

with $\bar{F}, F_i : \mathbb{R}^{n_y} \to \mathbb{R}$ and $F : \mathbb{R}^{n_y} \to \mathbb{R}^{(n_m+1)}$. In the first step, a quadratic model for each component F_i of the residual is formed:

$$q_i(y + \delta) = F_i(y) + \delta^\intercal g_i + \frac{1}{2}\delta^\intercal H_i \delta, \tag{7.2}$$

where g_i and $H_i = H_i^\intercal$ replace the unknown Jacobian and Hessian of F_i with respect to y, respectively, and $\delta \in \mathbb{R}^{n_y}$. Let $\mathcal{Y}$ be the interpolation set of points y where $F_i(y)$ is known, containing between $n_y + 1$ and $(n_y + 1)(n_y + 2)/2$ points (details can be found in [CSV09, Wil08]). Then g_i and H_i are obtained by solving the following convex problem

$$\min_{g_i, H_i} \left\{\|H_i\|_F : q_i(y) = F_i(y) \quad \forall y \in \mathcal{Y}\right\}. \tag{7.3}$$

This means, the approximations g_i and H_i of the first-order and second-order derivatives of the i-th component of the residual F are obtained by requiring that the model q_i agrees with the true function F_i for all $y \in \mathcal{Y}$. The authors of [KLM$^+$10] choose a trust region framework for their DFO algorithm since the quadratic model q_i defined in (7.2) can only be expected to approximate the i-th component of the residual with $i = 0, \dots, n_m$ within a certain neighborhood. Therefore, we denote the base-point by $\hat{y}$ and its neighborhood by $\mathcal{B} = \{y \in \mathbb{R}^{n_y} : \|y - \hat{y}\| \leq \Delta\}$ with the trust region radius $\Delta \in \mathbb{R}$, $\Delta > 0$. The interpolation points contained in $\mathcal{Y}$ should not be too far from $\mathcal{B}$. Knowing all function evaluations $F(y)$ with $y \in \mathcal{Y}$, the approximated derivatives g_i, H_i with $i = 0, \dots, n_m$ can be computed. The derivative-free model of (7.1) centered around $\hat{y}$ is then denoted by

$$m(\hat{y} + \delta) = \bar{F}(\hat{y}) + \delta^{\mathsf{T}} \sum_{i=0}^{n_m} F_i(\hat{y}) g_i + \frac{1}{2} \delta^{\mathsf{T}} \sum_{i=0}^{n_m} \left[g_i g_i^{\mathsf{T}} + F_i(\hat{y}) H_i \right] \delta, \tag{7.4}$$

which can be trusted within the neighborhood $\mathcal{B}$. In the next step, the following trust region subproblem is solved:

$$\begin{aligned} \underset{\delta}{\text{minimize}} \quad & m(\hat{y} + \delta) \\ \text{subject to} \quad & \hat{y} + \delta \in \mathcal{B}, \end{aligned} \tag{7.5}$$

which is easier than the original problem (7.1) since it is a quadratic program with known derivatives over a convex compact set. The residual F is then evaluated at the solution of (7.5) and added to the interpolation set $\mathcal{Y}$. The procedure of solving the subproblem (7.5) and updating the interpolation set $\mathcal{Y}$ is performed in an iterative way. The trust region Δ grows and shrinks in each iteration depending on the ratio of predicted and actual decrease. Furthermore, the current estimate of the solution of (7.1) is only updated by the solution of the subproblem (7.5) if the objective $\bar{F}$ decreases adequately. Otherwise, the interpolation set $\mathcal{Y}$ has to be improved by further evaluations of the objective $\bar{F}$. Details of the algorithm can be found in [Wil08, KLM$^+$10].

In the remainder of this section, we describe the embedding of the DFO method described in the last paragraph into a solution framework for hierarchical dynamic optimization problems (5.2) (where we choose the one-stage formulation of the hierarchical dynamic optimization problem for the sake of a simpler notation). The DFO technique described in [Wil08, KLM$^+$10] is in its current version only able to handle bound constraints, therefore, we skip the lower-level constraint

$$\sum_{i=1}^{n_{\mathcal{M}}} \gamma_i = 1 \tag{7.6}$$

in (5.2) and keep one γ_i of $\gamma_1, \dots, \gamma_{n_{\mathcal{M}}}$ fixed. The upper-level objective with upper-level bound constraints of (5.2) is given by

$$\begin{aligned} \underset{\gamma, p}{\text{minimize}} \quad & \frac{1}{2} \sum_{i=0}^{n_m} \sum_{j=1}^{n_h} \frac{(h_j(x(t_i^m), q, p) - \eta_{ij})^2}{\sigma_{ij}^2} \\ \text{subject to} \quad & \gamma_i \geq 0 \ \forall i = 1, \dots, n_{\mathcal{M}} \\ & b^{\text{lower}_p} \ \leq \ p \ \leq \ b^{\text{upper}_p}, \end{aligned} \tag{7.7}$$

where (x, u, q) is the solution of the lower-level OCP. We solve problem (7.7) using the DFO technique described in the last paragraph with

$$\bar{F}(y) = \quad \frac{1}{2} \sum_{i=0}^{n_m} \sum_{j=1}^{n_h} \frac{(h_j(x(t_i^m), q, p) - \eta_{ij})^2}{\sigma_{ij}^2} \tag{7.8}$$

and $y = \left(\gamma^\intercal, p^\intercal\right)^\intercal$. The solution (x, u, q) of the lower-level OCP implicitly depends on the upper-level variables y. The residual $F(y)$ is given by

$$F(y) = \begin{pmatrix} \frac{(h_1(x(t_0^m),q,p)-\eta_{0,1})}{\sigma_{0,1}} \\ \vdots \\ \frac{(h_{n_h}(x(t_0^m),q,p)-\eta_{0,n_h})}{\sigma_{0,n_h}} \\ \vdots \\ \frac{(h_1(x(t_{n_m}^m),q,p)-\eta_{n_m,1})}{\sigma_{n_m,1}} \\ \vdots \\ \frac{(h_{n_h}(x(t_{n_m}^m),q,p)-\eta_{n_m,n_h})}{\sigma_{n_m,n_h}} \end{pmatrix}. \tag{7.9}$$

The procedure in then the following:

- Solve the least-squares problem (7.7) with the DFO technique described in the last paragraph.
- Each time the residual F defined in (7.9) has to be evaluated at a point y, the lower-level OCP

$$\begin{array}{lll} \underset{x,u,q}{\text{minimize}} & \sum_{i=1}^{n_{\mathcal{M}}} \gamma_i^k \phi_{\mathcal{M}}^i(x(T), q, p^k) & \\ \text{subject to} & \dot{x}(t) \quad = f(x(t), u(t), q, p^k), & t \in [t_0, T] \\ & 0 \qquad \leq c(x(t), u(t), q, p^k), & t \in [t_0, T] \\ & 0 \qquad = r^{\text{eq}}(x(t_0), \dots, x(T), q, p^k) & \\ & 0 \qquad \leq r^{\text{ieq}}(x(t_0), \dots, x(T), q, p^k), & \end{array} \tag{7.10}$$

  has to be solved in order to obtain its solution $(x^*(y_k), u^*(y_k), q^*(y_k))$ for the given iterate y_k with $y_k = \left({\gamma^k}^\intercal, {p^k}^\intercal\right)^\intercal$ of the upper-level optimization problem (7.7).

Please note that this procedure requires two solvers: a solver that implements the DFO technique described at the beginning of this section, and most importantly, a reliable solver that is able to handle the lower-level OCP of (4.1), where the quality of the latter solver is essential for the success of this bilevel approach. The DFO method of Wild et al. [Wil08, KLM+10] is implemented in the MATLAB software package POUNDERS (Practical Optimization Using No DERivatives with Squares). For solving the lower-level OCP, we use our own C/C++ package PARAOCP, which is developed for solving hierarchical dynamic optimization problems but also includes an efficient method for solving classical one-level OCPs. PARAOCP is described in detail in Chapter 9.

7.3. A bundle method

In this section, we derive a bilevel approach for hierarchical dynamic optimization problems based on a trust region bundle method for the upper-level problem. We therefore briefly review a trust region bundle method for optimizing a nonsmooth, nonconvex and nonlinear cost function. It mainly follows the ideas of Zowe and Schramm [SZ92], Lemarechal [Lem81], and Kiwiel [Kiw90]. We consider the following problem

$$\underset{y}{\text{minimize}}\, \bar{F}(y), \tag{7.11}$$

where $\bar{F} : \mathbb{R}^{n_y} \to \mathbb{R}$. The subdifferential of $\bar{F}$ at y is defined as

$$\begin{aligned} &\partial \bar{F}(y) := \\ &\text{conv}\{g \in \mathbb{R}^{n_y} | g = \lim_{i\to\infty} \nabla \bar{F}(y_i),\ y_i \to y,\ \nabla \bar{F}(y_i) \text{ exists},\ \nabla \bar{F}(y_i) \text{ converges}\}, \end{aligned} \tag{7.12}$$

and for $\bar{F}$ as in (7.11), the subdifferential is a well-defined, convex and compact subset of $\mathbb{R}^{n_y}$ (see, e.g., [Roc97, Cla90]). In nonsmooth optimization, subgradients (elements of the subdifferential) naturally serve as substitutes for the gradient. The functional $\bar{F}$ is further required to be locally Lipschitzian and weakly semi-smooth, which means that the directional derivative of $\bar{F}$ in the direction d denoted by

$$\bar{F}'(y; d) = \lim_{t\downarrow 0}[\bar{F}(y + td) - \bar{F}(y)] \tag{7.13}$$

exists for all $y, d \in \mathbb{R}^{n_y}$, and

$$\bar{F}'(y; d) = \lim_{t\downarrow 0} g(y + td)^{\mathsf{T}} d, \tag{7.14}$$

where $g(y+td) \in \partial \bar{F}(y+d)$. In the remainder of this section, we assume to have a subroutine that evaluates the cost function $\bar{F}(y)$ and calculates a subgradient $g \in \partial \bar{F}(y)$ for given y.

At an iterate y_k, we have the sequence of former iterates $y_1, \dots, y_k$ and a collection of auxiliary points $\bar{y}_i$ together with subgradients $g_i \in \partial \bar{F}(\bar{y}_i)$ for $i \in J_k$, where J_k is a nonempty set of indices. For convex $\bar{F}$, this bundle of information leads to the following cutting plane (CP) model of $\bar{F}$ at y_k

$$\max_{i\in J_k}\{g_i^{\mathsf{T}}(y - \bar{y}_i) + \bar{F}(\bar{y}_i)\}. \tag{7.15}$$

With the linearization errors

$$\alpha_{k,i} := \alpha(y_k, \bar{y}_i) := \bar{F}(y_k) - (\bar{F}(\bar{y}_i) + g_i^{\mathsf{T}}(y_k - \bar{y}_i)) \tag{7.16}$$

and with $d := y - y_k$, we can rewrite (7.15) as

$$\bar{F}_{\text{CP}}(y_k; d) := \max_{1\le i\le k}\{g_i^{\mathsf{T}} d - \alpha_{k,i}\} \text{ for } d \in \mathbb{R}^{n_y}, \tag{7.17}$$

where the constant term $\bar{F}(y_k)$ is skipped. Since we cannot trust this model far away from y_k, we add a trust region stabilization term $\frac{1}{2t_k} d^{\mathsf{T}} d$ with positive t_k to the CP model and write it as a quadratic program in $\mathbb{R}^1 \times \mathbb{R}^{n_y}$:

$$\begin{aligned} &\underset{v(t),d(t)}{\text{minimize}} && v + \tfrac{1}{2t_k}\|d\|_2^2 \\ &\text{subject to} && v \ge g_i^{\mathsf{T}} d - \alpha_{k,i} \text{ for } i \in J_k, \end{aligned} \tag{7.18}$$

where t_k is updated in a trust region sense. This means t_k is increased and decreased in a systematic way: the CP model is improved by Null Steps until we can trust the CP model and make a Serious Step.

However, we need to deal with a nonconvex function $\bar{F}$, for which the cutting plane model (7.17) is no longer an approximation from below. To cope with this difficulty, we replace $\alpha_{k,i}$ by

$$\beta_{k,i} := \beta(y_k, \bar{y}_i) := \max\left\{\alpha_{k,i}, c_0\|y_k - \bar{y}_i\|_2^2, \max_{i \in J_k}\left\{g_i^{\mathsf{T}}(y_{k+1} - y_k) - \bar{F}(y_{k+1})\right\}\right\} \tag{7.19}$$

with c_0 being a fixed small positive number. It is $\beta_{i,k} \geq 0$ by construction and in (7.17), we replace α by β:

$$\bar{F}_{\text{CP}}(y_k; d) := \max_{1 \leq i \leq k}\{g_i^{\mathsf{T}} d - \beta_{k,i}\} \ \text{ for } \ d \in \mathbb{R}^{n_y}. \tag{7.20}$$

The last term in (7.19) given by $\max_{i \in J_k}\left\{g_i^{\mathsf{T}}(y_{k+1} - y_k) - \bar{F}(y_{k+1})\right\}$ ensures that the CP model approximates $\bar{F}$ at $\bar{y}_i$ from below for all $i \in J_k$. As stopping criterion we use

$$\left\|\sum_{i \in J_k} \lambda_i g_i\right\|_2 \leq \epsilon \ \text{ and } \ \sum_{i \in J_k} \lambda_i \beta_{k,i} \leq \epsilon, \tag{7.21}$$

where $\lambda \in \Lambda(|J_k|) := \left\{\lambda \in \mathbb{R}^{|J_k|} | \lambda_i \geq 0, 1 \leq i \leq |J_k|, \text{and} \sum_{i=1}^{|J_k|} \lambda_i = 1\right\}$. For convex $\bar{F}$, (7.21) ensures the ϵ-optimality of y_k, which means that $\bar{F}(y_k) \leq \bar{F}(y) - \epsilon\|y - y_k\|_2 + \epsilon$ for all $y \in \mathbb{R}^{n_y}$. However, for nonconvex $\bar{F}$, this is no longer true and (7.21) corresponds to *almost stationary* in smooth optimization. The criteria to determine whether a Null Step (NS) or a Serious Step (SS) is taken are the following:

$$\begin{array}{ll} \textbf{SS} & \bar{F}(\bar{y}^j) - \bar{F}(y_k) < m_1 v^j, \\ \textbf{NS(i)} & \bar{F}(\bar{y}^j) - \bar{F}(y_k) \geq m_1 v^j, \\ \textbf{NS(ii)} & \textbf{(i)}\ \alpha(y_k, \bar{y}^j) \leq m_3 \sigma_{k-1} \ \textit{or} \ \textbf{(ii)}\ |\bar{F}(y_k) - \bar{F}(\bar{y}^j)| \leq \|z_{k-1}\|_2 + \sigma_{k-1}, \\ \textbf{NS(iii)} & {g^j}^{\mathsf{T}} d^j - \beta_{k,j} \geq m_2 v^j, \end{array} \tag{7.22}$$

with $z_k := \sum_{i \in J_k} \lambda_{k,i} g_i$ and $\sigma_k := \sum_{i \in J_k} \lambda_{k,i} \beta_{k,i}$ ($z_0 := g_1$, $\sigma_0 := 0$). Algorithm 3 is very similar to the one proposed in [SZ92], except for increasing/decreasing t_k after a Serious Step/Null Step, a reset strategy, and the gradient sampling approach. The line search in Algorithm 3 is considered as an *emergency exit* (as discussed in [SZ92]) and is described in detail in Algorithm 4 (compare [Lem81]). As reported in [SZ92], we also noticed numerical troubles after switching to line search. Therefore, we implemented and tested a simple gradient sampling approach instead as described in Algorithm 5.

In the remainder of this section, we describe the embedding of the bundle method described in the last paragraph into a solution procedure for hierarchical dynamic optimization problems of type (5.2). The approach described in the last paragraph is strongly based on the work of Schramm and Zowe [SZ92]. However, the algorithm in [SZ92] is not able to handle any type of constraints. Hence, we assume for the hierarchical dynamic optimization problem of type (5.2) that there are no bounds on the lower-level variables γ and p, which means that the bounds

$$\begin{array}{c} \gamma_i \geq 0 \ \ \forall i,\ i = 1, \ldots, n_{\mathcal{M}} \\ b^{\text{lower}_p} \ \leq \ p \ \leq \ b^{\text{upper}_p} \end{array} \tag{7.23}$$

Algorithm 3: Bundle trust region algorithm for (7.11).

```
INPUT: Initial value $y_1 \in \mathbb{R}^{n_y}$, $t_0 \in \mathbb{R}$, parameters $T > 0$, $0 < m_1 < m_2 < 1$,
$0 < m_3 < 1$, $\nu > 0$, $0 < \rho < 1$, $\epsilon \geq 0$,
$J_{\max} \geq 3$, maximum number of iterations $M, M_{\text{inner}}$.
OUTPUT: Local solution $y$ of (7.11).
Compute $\bar{F}(y_1)$, $g_1 \in \partial \bar{F}(y_1)$;
Initialize $\bar{y}_1 := y_1$, $J_1 := \{1\}$;
for $k = 1, \ldots, M$ do
    INNER ITERATION: Set $t^1 := t_{k-1}$;
    for $j = 1, \ldots, M_{\text{inner}}$ do
        $(v^j, d^j) \leftarrow$ solve (7.20).
        if converged then
            stop.
        end
        if SS then
            Serious Step: $y_{k+1} = y_k + d^j$, $\bar{y}_{k+1} = \bar{y}^j$, $g_{k+1} = g^j$, $t_k = \min(\frac{1}{\rho} t^j, T)$
            and stop.
        end
        if NS(i), NS(ii) and NS(iii) then
            Null Step: $y_{k+1} = y_k$, $\bar{y}_{k+1} = \bar{y}^j$, $g_{k+1} = g^j$, $t_k = \rho t^j$ and stop.
        end
        if NS(i), NS(ii) and $\neg$ NS(iii) then
            if NS(ii)(ii) then
                line search (Algorithm 4) or gradient sampling (Algorithm 5) until
                Null Step or Serious Step is taken, then stop.
            else
                $t^{j+1} = \rho t^j$.
            end
        end
        if NS(i) and $\neg$ NS(ii) then
            $t^{j+1} = \rho t^j$.
        end
    end
    RESET: if $|J| = J_{\max}$ then
        choose $J \subset J_k$ with $|J| \leq J_{\max} - 2$ and smallest $\beta_{k,i}$.
    end
    UPDATE: $J_{k+1} := J \cup k + 1$;
    $\beta_{k+1,i} := \max\left\{\alpha_{k+1,i}, c_0 \|y_{k+1} - \bar{y}_i\|_2^2, \max_{i \in J_k}\left\{g_i^\intercal (y_{k+1} - y_k) - \bar{F}(y_{k+1})\right\}\right\}$ for
    $i \in J_{k+1}$;
end
```

Algorithm 4: Line search algorithm.

```
INPUT: $y_k$, $d^j$, initial step size $s^1$, $\bar{m}$, $\bar{m}_1$, $\rho_{\text{ls}}$, maximum number of iterations $M_{\text{ls}}$.
OUTPUT: $y_{k+1}, \bar{y}_{k+1}, g_{k+1}$.
Initialize $u^1 := 0$, $l^1 := 0$;
for $i = 1, \dots, M_{ls}$ do
    Compute $\bar{y}^i = y_k + s^i \cdot d^j$, $\bar{F}(\bar{y}^i)$, $g(\bar{y}^i)$, $\bar{F}(y_k)$, $g(y_k)$;
              (suffdecr) $\bar{F}(\bar{y}^i) \leq \bar{F}(y_k) + \bar{m} \cdot s^i \cdot g(y_k)^\intercal d^j$;
              (curv)       $g(\bar{y}^i)^\intercal d^j \leq \bar{m}_1 \cdot g(y_k)^\intercal d^j$;
    if (suffdecr) then
        if (curv) then
            Serious Step: $y_{k+1} = y_k + d^j,\ \bar{y}_{k+1} = \bar{y}^i,\ g_{k+1} = g^i$, stop;
        else
            $l^{i+1} = s^i$;
            if $u^i = 0$ then
                Increase $s^i$: $u^{i+1} = u^i$, $s^{i+1} = \frac{1}{\rho_{\text{ls}}} l^{i+1}$;
            else
                Decrease $s^i$: $u^{i+1} = u^i$, $s^{i+1} = l^{i+1} + \rho_{\text{ls}} (u^{i+1} - l^{i+1})$;
            end
        end
    else
        $u^{i+1} = s^i$;
    end
    if $l^i = 0$ then
        if curv then
            Null Step: $y_{k+1} = y_k,\ y_{k+1} = y^i,\ g_{k+1} = g^i$, stop;
        else
            Decrease $s^i$: $l^{i+1} = l^i$, $s^{i+1} = \rho_{\text{ls}} l^{i+1}$;
        end
        Decrease $s^i$: $l^{i+1} = l^i$, $s^{i+1} = l^{i+1} + \rho_{\text{ls}} (u^{i+1} - l^{i+1})$;
    end
end
```

Algorithm 5: Gradient sampling algorithm.

```
INPUT: $y_k$, max_gs, maximum number of iterations $M_{\text{gs}}$, parameter $\epsilon$.
OUTPUT: $y_{k+1}$.
Choose independently and uniformly distributed
$\hat{d}_1, \dots, \hat{d}_{\text{max\_gs}} \in \mathbb{B}_\epsilon := \{y \| \|y\|_2 \leq \epsilon\}$;
Choose $i = \arg\min_{1 \leq j \leq \text{max\_gs}} \|g(y_k + \hat{d}_j)\|_2$;
Serious Step: $y_{k+1} = y_k + \hat{d}_i$ and stop.
```

are skipped, and we furthermore fix one γ_i of $\gamma_1, \ldots, \gamma_{n_\mathcal{M}}$ instead of requiring $\sum_{i=1}^{n_\mathcal{M}} \gamma_i = 1$. This means, the lower-level problem of (5.2) is given by

$$\underset{\gamma,p}{\text{minimize}}\ \frac{1}{2}\sum_{i=0}^{n_m}\sum_{j=1}^{n_h}\frac{(h_j(x(t_i^m),q,p)-\eta_{ij})^2}{\sigma_{ij}^2}. \tag{7.24}$$

All variables in (7.24) are described in detail at the beginning of Chapter 5. We now apply the bundle method described in the last paragraph to problem (7.24) with

$$\bar{F}(y) = \frac{1}{2}\sum_{i=0}^{n_m}\sum_{j=1}^{n_h}\frac{(h_j(x(t_i^m),q,p)-\eta_{ij})^2}{\sigma_{ij}^2}, \tag{7.25}$$

and the upper-level variables $y = (\gamma^\intercal, p^\intercal)^\intercal$. The variables (x,u,p) are the solution of the lower-level OCP (7.10). The bundle method described in Algorithm 3 requires the computation of subgradients $g \in \partial\bar{F}(y)$. For many applications, the model response h_j and the differential states x are smooth functions for $j = 1, \ldots, n_h$. We nevertheless choose a bundle technique instead of, e.g., a gradient descent method since the bundle approach can be interpreted as an improved gradient descent method because of the additional derivative information contained in the cutting plane model. We finally note that there are examples of hierarchical dynamic optimization problems in practice, where the upper-level objective is not continuously differentiable but locally Lipschitzian and weakly semi-smooth, as, e.g., when identifying an optimal control model of human gaits with measurements of the local joint velocities. However, this case is not considered in this section. Assuming that h_j and x are sufficiently smooth for $j = 1, \ldots, n_h$, we compute derivatives of $\bar{F}$ with respect to y based on finite differences. The solution procedure for hierarchical dynamic optimization problems for a bilevel approach with the bundle method described in the first part of this section for the solution of the lower-level problem is the following:

- Solve the unconstrained least-squares problem (7.11) with the bundle technique described in the last paragraph.
- Each time the objective $\bar{F}$ and its derivative $\nabla_y\bar{F}$ have to be evaluated, the lower-level OCP

$$\begin{array}{lll} \underset{x,u,q}{\text{minimize}} & \displaystyle\sum_{i=1}^{n_\mathcal{M}}\gamma_i^k\phi_\mathcal{M}^i(x(T),q,p^k) & \\ \text{subject to} & \dot{x}(t) \ = f(x(t),u(t),q,p^k), & t\in[t_0,T] \\ & 0 \quad\ \ \le c(x(t),u(t),q,p^k), & t\in[t_0,T] \\ & 0 \quad\ \ = r^{\text{eq}}(x(t_0),\ldots,x(T),q,p^k) & \\ & 0 \quad\ \ \le r^{\text{ieq}}(x(t_0),\ldots,x(T),q,p^k) & \end{array} \tag{7.26}$$

has to be solved in order to obtain its solution $(x^*(y_k), u^*(y_k), q^*(y_k))$ for the given iterate y_k with $y_k = \left(\gamma^{k^\intercal}, p^{k^\intercal}\right)^\intercal$ of the upper-level optimization problem (7.11), and the derivative $\nabla_y\bar{F}(y_k)$ is computed as follows:

$$\nabla_{y_i}\bar{F}(y_k) \approx \frac{\bar{F}(y_k + h\cdot e_i) - \bar{F}(y_k)}{h}, \tag{7.27}$$

with $e_i \in \mathbb{R}^{n_y}$ being a unit vector with 1 as the i-th entry and the constant h.

We use our own implementation of the bundle method described at the beginning of this section in Matlab and for the solution of the lower-level OCP, we use our C/C++ package ParaOCP, which is described in detail in Chapter 9. The numerical performance and the efficiency of the bundle approach for hierarchical dynamic optimization is discussed in Chapter 14 for an illustrative example.

7.4. Discussion

This chapter shows two bilevel approaches for hierarchical dynamic optimization problems with a DFO approach and a bundle technique for the upper-level problem. Both approaches have in common that they require a reliable and efficient solver for the lower-level OCP, which is able to handle general nonlinear OCPs with mixed control-path constraints (cf. the lower-level problem in (4.1)). The efficiency of the OCP solver is crucial for the performance of the bilevel approach – solving the lower-level OCP is by far the most expensive part of both bilevel approaches. Hence, the implementation of the DFO technique and bundle method in MATLAB instead of C/C++ does not significantly influence the overall performance. An advantage of the algorithms presented in this Chapter over the all-at-once approaches from Chapters 5 and 6 is that the treatment of a complementarity constraint can be avoided.

Chapter 8.

Regularizing bilevel nonlinear programs by lifting

In this chapter, we continue the discussion of numerical methods for mathematical programs with complementarity constraints from Section 2.3. We start with a brief review of a sequential quadratic programming method for mathematical programs with complementarity constraints and the associated convergence results stated in [FLRS06]. Afterwards, we propose a lifting technique for bilevel nonlinear programs and a generalization for mathematical programs with complementarity constraints that do not arise from bilevel programs. The lifting approach allows us to guarantee a constraint qualification tailored to mathematical programs with complementarity constraints that is violated in many examples, and a requirement of the convergence results for sequential quadratic programming methods applied to mathematical programs with complementarity constraints. At the end of this chapter, we discuss the lifting idea in the framework of the all-at-once approach for hierarchical dynamic optimization problems described in Chapter 5.

8.1. Numerical methods for solving mathematical programs with complementarity constraints and their limits

Numerical experiments in [FL02b] demonstrate that sequential quadratic programming (SQP) methods reliably and efficiently solve a large class of mathematical programs with complementarity constraints (MPCCs) reformulated as nonlinear programs (NLPs). Motivated by the numerical success in [FL02b], Fletcher et al. [FLRS06] provide local convergence results for SQP methods applied to MPCCs. In the following, we discuss the results of [FLRS06] in more detail, to highlight the strengths and weaknesses of this approach. Therefore, we consider the MPCC

$$\begin{array}{rl}\underset{x}{\text{minimize}} & f(x)\\ \text{subject to} & g(x)=0\\ & h(x)\geq 0\\ & 0\leq x_1\perp x_2\geq 0,\end{array}\tag{8.1}$$

where $x = (x_0, x_1, x_2) \in \mathbb{R}^{p+q+q}$ is a decomposition of the problem variables, $f(\cdot) \in \mathbb{R}$, $g(\cdot) \in \mathbb{R}^{n_g}$ and $h(\cdot) \in \mathbb{R}^{n_h}$. The functions f, g and h are assumed to be sufficiently smooth.

The corresponding NLP formulation (cf. Chapter 2.3) is given by

$$\begin{array}{rl}\underset{x}{\text{minimize}} & f(x)\\ \text{subject to} & g(x)=0\\ & h(x)\geq 0\\ & x_1,x_2\geq 0\\ & x_1^\intercal x_2=0.\end{array}\tag{8.2}$$

We now briefly review the definition of strong stationarity of (8.1) and NLP stationarity of (8.2) to highlight the relationship between the respective multipliers.

A point x is said to be a strongly stationary point of (8.1) if there exist multipliers $\lambda\in\mathbb{R}^{n_g},\mu\in\mathbb{R}^{n_h},\hat{\nu}_1\in\mathbb{R}^q$ and $\hat{\nu}_2\in\mathbb{R}^q$ such that

$$\begin{array}{rcll}\nabla_x f(x)-\nabla_x g(x)^\intercal\,\lambda-\nabla_x h(x)^\intercal\,\mu-\begin{pmatrix}0\\ \hat{\nu}_1\\ \hat{\nu}_2\end{pmatrix} & = & 0, & \\ g(x) & = & 0, & \\ h(x) & \geq & 0, & \\ x_1 & \geq & 0, & \\ x_2 & \geq & 0, & \\ x_{1j}=0\quad\text{or}\quad x_{2j} & = & 0 & \forall\, j=1,\cdots,q,\\ \mu & \geq & 0, & \\ h(x)^\intercal\mu & = & 0, & \\ x_{1j}\hat{\nu}_{1j} & = & 0 & \forall\, j=1,\cdots,q,\\ x_{2j}\hat{\nu}_{2j} & = & 0 & \forall\, j=1,\cdots,q,\\ \text{if}\quad x_{1j}=x_{2j}=0,\quad\text{then}\quad\hat{\nu}_{1j}\geq 0\quad\text{and}\quad\hat{\nu}_{2j} & \geq & 0 & \forall\, j=1,\cdots,q.\end{array}\tag{8.3}$$

The KKT conditions of (8.2) are as follows: There are multipliers $\lambda\in\mathbb{R}^{n_g},\mu\in\mathbb{R}^{n_h},\nu_1\in\mathbb{R}^q,\nu_2\in\mathbb{R}^q$ and $\xi\in\mathbb{R}$ such that

$$\begin{array}{rcl}\nabla_x f(x)-\nabla_x g(x)^\intercal\,\lambda-\nabla_x h(x)^\intercal\,\mu-\begin{pmatrix}0\\ \nu_1\\ \nu_2\end{pmatrix}+\xi\begin{pmatrix}0\\ x_2\\ x_1\end{pmatrix} & = & 0\\ g(x) & = & 0\\ h(x) & \geq & 0\\ x_1 & \geq & 0\\ x_2 & \geq & 0\\ x_1^\intercal x_2 & = & 0\\ \mu & \geq & 0\\ \nu_1 & \geq & 0\\ \nu_2 & \geq & 0\\ \xi & \geq & 0\\ h(x)^\intercal\mu & = & 0\\ x_{1j}\nu_{1j} & = & 0\\ x_{2j}\nu_{2j} & = & 0\end{array}\tag{8.4}$$

for $j=1,\cdots,q$. The condition $\xi x_1^\intercal x_2=0$ has been skipped since it is implied by feasibility. Fletcher et al. [FLRS06] show that a point x is a strongly stationary point of MPCC (8.1) if and only if it is a stationary point of the NLP (8.2). We briefly recall the idea of the proof (cf. [FLRS06, Hat08]) in order to explain the success of SQP methods applied to MPCCs.

We note that the gradient of the Lagrangian of (8.1) and (8.2) are equivalent if

$$\hat{\nu}_1 = \nu_1 - \xi x_2 \quad \text{and} \quad \hat{\nu}_2 = \nu_2 - \xi x_1. \tag{8.5}$$

We now start by showing (8.4)⇒(8.3). We distinguish three cases:

1. If $x_{1j} > 0$, then $x_{2j} = 0 = \nu_{1j}$ from complementarity ($x_1^\intercal x_2 = 0$) and slackness ($x_{1j}\nu_{1j} = 0$). From the first equation of (8.5) it follows that $\hat{\nu}_{1j} = 0$, and further $\hat{\nu}_{2j} = \nu_{2j} - \xi x_{1j}$ satisfies (8.3).

2. If $x_{2j} > 0$, then transpose the above argument.

3. If $x_{1j} = x_{2j} = 0$, then (8.5) implies that $\hat{\nu}_1 = \nu_1 \geq 0$ and $\hat{\nu}_2 = \nu_2 \geq 0$.

Combining 1.- 3., one sees that (8.4) implies (8.3). Next we show that (8.3)⇒(8.4). We also distinguish three cases:

4. If $x_{1j} > 0$, then $\hat{\nu}_{1j} = 0$ and $x_{2j} = 0$. This implies $\nu_{1j} = \xi x_{2j} + \hat{\nu}_{1j} = 0 \geq 0$ for any ξ. To ensure that $\nu_{2j} = \xi x_{1j} + \hat{\nu}_{2j}$ is nonnegative, we need to choose ξ such that $\xi x_{1j} + \hat{\nu}_{2j} \geq 0$ for all j, or equivalently that $\xi \geq -\hat{\nu}_{2j}/x_{1j}$ for all j.

5. If $x_{2j} > 0$, then transpose the above argument.

6. $x_{1j} = x_{2j} = 0$, then $\nu_{1j} = \hat{\nu}_{1j} \geq 0$, for any ξ. From parts 4. and 5. it follows that choosing ξ to be at least

$$\xi = \max_i \left\{ 0, \max_{i \in \mathcal{Z}_2^\perp} \frac{-\widehat{\nu}_{1i}}{x_{2i}}, \max_{i \in \mathcal{Z}_1^\perp} \frac{-\widehat{\nu}_{2i}}{x_{1i}} \right\}. \tag{8.6}$$

   will ensure that $\nu_1, \nu_2 \geq 0$ ($\mathcal{Z}_2^\perp$ and $\mathcal{Z}_1^\perp$ are defined in Section 2.3). Examining the right-hand side of (8.6), one can see that ξ is bounded (it is uniquely determined).

Combining 4.- 6., it follows that (8.4) implies (8.3). We have mentioned already that the multipliers of MPCCs are not unique anymore and the set of multipliers is unbounded, which is due to the fact that any value $\hat{\xi} > \xi$ would also satisfy the stationarity conditions (8.4). But if we make sure to choose ξ as defined in (8.6) (called the basic multiplier in [FLRS06]), it can be shown that the constraint normals corresponding to nonzero multipliers are linearly independent. This result implies that if MPCC-LICQ (defined in Section 2.3) holds at a local minimizer of (8.2), then it is a strongly stationary point and the multipliers in (8.3) and the basic multiplier (8.6) are unique.

As main result, we summarize the following: Solving the MPCC (8.1) and solving its NLP formulation (8.2) with an SQP method that guarantees that ξ is chosen as defined in (8.6) is equivalent as long as MPCC-LICQ holds, and we can guarantee that the constraint normals corresponding to nonzero multipliers are linearly independent.

It remains to discuss whether SQP solvers choose the multiplier ξ as defined in (8.6). It can be shown that active-set based SQP methods that work with a nonsingular basis (as, e.g., the one described in [NW99]) guarantee to choose ξ as in (8.6). Details can be found in [FLRS06, Hat08]. The key idea is to use an SQP method iterating on a working set, which is a subset of the active set including all equality constraints and the inequality constraints with nonzero multipliers. Starting with an initial working set that only includes constraints with linearly independent constraint normals, a constraint with a linearly dependent constraint normal cannot enter the working set since it never becomes a blocking constraint (cf. [NW99, Hat08]). An active-set based SQP method that works with a nonsingular basis ensures that the constraint normals corresponding to nonzero multipliers are linearly independent, which then allows to guarantee that ξ is chosen as defined in (8.6) [FLRS06].

We now briefly review the convergence results for SQP methods applied to MPCCs. We make the following assumptions:

[A1] f, g and h are twice Lipschitz continuously differentiable.

[A2] The MPCC (8.1) satisfies the MPCC-LICQ (Definition 8.2.2).

[A3] x^* is a strongly stationary point of (8.1) with multipliers $\lambda^*, \mu^*, \nu_1^*, {\nu_2}^*$ (as defined in (8.3)), and x^* satisfies the MPCC second-order sufficient condition MPCC-SOSC (see [FLRS06]).

[A4] It holds $\lambda^* \neq 0$, $\mu_i^* > 0 \; \forall i \in \mathcal{A}^*$ (where $\mathcal{A}^*$ is the active set of (8.1) at x^* as defined in Section 2.1) and both $\nu_{1j}^* > 0$ and $\nu_{2j}^* > 0 \;\; \forall j \in \mathcal{D}^* := \{i \mid x_{1i}^* = x_{2i}^* = 0\}$.

[A5] The QP solver chooses a linearly independent basis.

In [FLRS06], two cases are considered: the case where exact complementarity is satisfied at a point sufficiently close to a stationary point, and the case where exact complementarity is not satisfied.

For the first case, we have the following additional assumption:

[A6] For some $\bar{k}$ we have that $x_1^{(\bar{k})\intercal} x_2^{(\bar{k})} = 0$ and $(x^{(\bar{k})}, \mu^{(\bar{k})})$ is sufficiently close to a strongly stationary point.

The key results of [FLRS06] can then be summarized in the following theorem.

Theorem 8.1.1. *If Assumption [A1]-[A6] hold, SQP applied to* (8.2) *generates a sequence*

$$\{(x^{(\bar{l})}, \lambda^{(\bar{l})}, \mu^{(\bar{l})}, \nu_1^{(\bar{l})}, {\nu_2}^{(\bar{l})}, \xi^{(\bar{k})})\}_{\bar{l}>\bar{k}} \tag{8.7}$$

that converges Q-quadratically to a solution $\{(x^*, \lambda^*, \mu^*, \nu_1^*, \; \nu_2^*, \xi^*)\}$ *of* (8.1)*, satisfying strong stationarity. Moreover, the sequence* $\{x^{(\bar{k})}\}_{\bar{l}>\bar{k}}$ *converges Q-superlinearly to* x^* *and* $x_1^{(\bar{k})\intercal} x_2^{(\bar{k})} = 0 \; \forall \, \bar{l} \geq \bar{k}$.

For the latter case, strict complementarity does not hold in the starting point, i.e., $x_1^{(\bar{k})T} x_2^{(\bar{k})} > 0$, which might lead to inconsistent QP approximations arbitrarily close to a stationary point. To tackle this problem, an additionally assumption is required.

[A7] All QP approximations remain consistent.

For the latter case, the following results are obtained (cf. [FLRS06]).

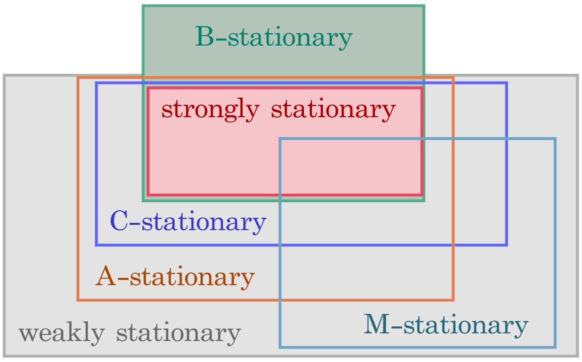


Figure 8.1.: Illustration of the relationship between stationarity concepts for MPCCs.

Theorem 8.1.2. *Let Assumption [A1]-[A5] and [A7] hold. Then it follows that SQP applied to the NLP formulation* (8.2) *of the MPCC* (8.1) *converges quadratically near a solution* $(x^*, \lambda^*, \mu^*, \nu_1^*, \nu_2{}^*, \xi^*)$.

Proofs for Theorem 8.1.1 and 8.1.2 can be found in [FLRS06]. [A7] is an undesirable assumption. In, e.g., FILTERSQP, a feasibility restoration method is implemented to handle inconsistent QPs.

To sum up, under conditions [A1]-[A5] and [A6] or [A7], it can be shown that SQP methods applied to MPCCs converge to strongly stationary points. There is a wide range of algorithms for MPCCs as, e.g., algorithms based on branch-and-bound techniques (as, e.g., described in [Bar88]), interior-point methods (cf. [LLCN06] or [RB03]), or penalization and relaxation techniques [DFNS05, HKS13, LF03]. However, for most algorithms for MPCCs, convergence to strongly or even B-stationary points cannot be shown, and therefore, weaker stationarity concepts are used. The main difference of the stationarity concepts is the condition on the sign of multipliers $\hat{\nu}_1$ and $\hat{\nu}_2$ in (8.3). We now briefly review the most common ones. Therefore, let x be the solution of (8.1) and let $\mathcal{D}$ denote the set of degenerate indices (cf. Section 2.3). Considering the set of conditions (8.3) and replacing the last condition by one of the following ones leads to the subsequent definitions:

1. No further conditions on $\hat{\nu}_1$ and $\hat{\nu}_2$ is called weak stationarity.

2. $\hat{\nu}_{1i} \geq 0$ or $\hat{\nu}_{2i} \geq 0 \;\; \forall i \in \mathcal{D}$ is called A-stationarity.

3. $\hat{\nu}_{1i}\hat{\nu}_{2i} \geq 0 \;\; \forall i \in \mathcal{D}$ is called C-stationarity.

4. $(\hat{\nu}_{1i} > 0 \;\text{ and }\; \hat{\nu}_{2i} > 0)$ or $\hat{\nu}_{1i}\hat{\nu}_{2i} = 0 \;\; \forall i \in \mathcal{D}$ is called M-stationarity.

5. $\hat{\nu}_{1i}, \hat{\nu}_{2i} \geq 0 \;\; \forall i \in \mathcal{D}$ is called strong stationarity.

The relation between the stationarity concepts is illustrated in Figure 8.1 (cf. [LM07]). We decide to not rely on algorithms for MPCCs with convergence results to weakly, A-, C-, or M-stationary points, since these points might still have first-order descent directions as shown in [LM07]. B- and strongly stationary points, however, guarantee the absence of first-order descent directions. Hence, we focus on SQP methods applied to MPCCs since they ensure local convergence to strongly stationary points under relatively mild conditions.

And if MPCC-LICQ holds, B-stationarity (defined in Section 2.3) and strong stationarity are equivalent (cf. [SS00]).

Based on the observations described in this section, we summarize:

- SQP methods applied to MPCCs are efficient and guarantee local convergence to strongly stationary points under certain conditions (as, e.g., MPCC-LICQ).
- Many other popular algorithms for MPCCs only guarantee local convergence to weakly, A-, C-, or M-stationary points, and these points might still have first-order descent directions.
- If MPCC-LICQ holds, B-stationarity and strong stationarity are equivalent.
- MPCC-LICQ plays an important role as prerequisite for the local convergence results of SQP methods applied to MPCCs, and it furthermore allows to guarantee the convergence to B-stationary points.

In this section, we have discussed the fundamental role of MPCC-LICQ. Therefore, in the remainder of this chapter, we propose a new lifting technique for bilevel programs but also for general MPCCs, which allows to guarantee MPCC-LICQ.

8.2. Lifting bilevel programs for regularity

In this section, we describe the basic idea of lifting bilevel nonlinear programs in order to be able to guarantee MPCC-LICQ at the solution. The following three sections mainly follow the description in [HLSB13]. An idea related to the method described in this section can be found in [Ani05, ATW07, Ste12], and the references therein. In [Ani05, ATW07], the authors consider general MPCCs, and introduce an additional scalar variable to relax the MPCC's standard equality and inequality constraints on quadratic programming level (within an SQP framework) and the complementarity condition is treated with an exact penalty term. Global convergence properties based on this formulation are discussed. However, this approach requires strong modifications of the SQP method. Our goal is to use standard SQP solvers and to just modify the problem formulation. In [Ste12], a smoothing approach for mathematical programs with complementarity constraints is presented, based on the projection of a suitable set in $\mathbb{R}^3$. The author introduces a regularization approach involving a new concept of tilting stability, discusses the regularity of the feasible set and presents preliminary numerical results.

In this section, we consider a bilevel nonlinear program of the following form:

$$\begin{array}{lll} \underset{x}{\text{minimize}} & F(x,y) & \\ \text{subject to} & \underset{y}{\text{minimize}} & f(x,y) \\ & \text{subject to} & h(x,y) \geq 0, \end{array} \tag{8.8}$$

where $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$. The inequality constraints are denoted as $h(x,y) \in \mathbb{R}^{n_h}$, the upper-level objective is described by $F(x,y)$, the lower-level objective by $f(x,y)$. All functions in (8.8) are assumed to be sufficiently smooth. For a compact presentation, we skip upper-level constraints and lower-level equality constraints. However, the results in this chapter can be generalized to the following bilevel problem:

$$\begin{array}{lll} \underset{x}{\text{minimize}} & F(x,y) & \\ \text{subject to} & \underset{y}{\text{minimize}} & f(x,y) \\ & \text{subject to} & h(x,y) \geq 0 \\ & & g(x,y) = 0 \\ & H(x) \geq 0 & \\ & G(x) = 0, & \end{array} \tag{8.9}$$

with $g(x,y) \in \mathbb{R}^{n_g}, H(x) \in \mathbb{R}^{n_H}$ and $G(x) \in \mathbb{R}^{n_G}$ being twice continuously differentiable and under mild conditions on the regularity of the additional upper-level constraints: the composition of the Jacobian of the equality constraints $G(x)$ and the Jacobian of the active inequality constraints of $H(x)$ has to have full rank.

The bilevel NLP (8.8) can be written as an MPCC by replacing the lower-level problem by its first-order optimality conditions:

$$\begin{array}{lll} \underset{x,y,z}{\text{minimize}} & F(x,y) & \\ \text{subject to} & 0 & = \nabla_y \mathcal{L}(x,y,z) \\ & 0 & \leq z \perp h(x,y) \geq \ 0, \end{array} \tag{8.10}$$

with the gradient of the Lagrangian given by $\nabla_y \mathcal{L}(x,y,z) := \nabla_y f(x,y) - \nabla_y h(x,y)^{\intercal} z$ and the Lagrange multipliers $z \in \mathbb{R}^{n_h}$. If the lower-level problem of (8.8) is nonconvex, the MPCC (8.10) is in general not equivalent to (8.8). We are not concerned with this issue here. Our focus is on developing practical algorithms for solving (8.10). In the following, we refer to

$$\begin{array}{ll} \underset{y}{\text{minimize}} & f(x,y) \\ \text{subject to} & h(x,y) \geq 0 \end{array} \tag{8.11}$$

as the lower-level problem of (8.8). We note that x is not a variable in (8.11). We denote the active set of (8.11) by

$$\mathcal{A}(x,y^*) := \{i \in \{1,\cdots,n_h\} \mid h_i(x,y^*) = 0\} \tag{8.12}$$

for a given point $y^* \in \mathbb{R}^m$. We furthermore say that (8.11) satisfies LICQ at y^* if the set of active constraint gradients $\{\nabla_y h_i(x,y^*),\ i \in \mathcal{A}(x,y^*)\}$ is linearly independent (cf. Definition 2.1.2). The feasible set of (8.11) is given by the set $\mathcal{F}(x,y) := \{y \in \mathbb{R}^m \mid h(x,y) \geq 0\}$ and the Lagrangian $\mathcal{L}(x,y,z) := f(x,y) - h(x,y)^{\intercal} z$ is the Lagrangian of the lower-level NLP (8.11) with multipliers $z \in \mathbb{R}^{n_h}$ and the tangent cone is denoted as $\mathcal{T}(x,y) := \{p \in \mathbb{R}^m \mid \nabla_y h_i(x,y)^{\intercal} p = 0\ \forall i \in \mathcal{A}(x,y) \text{ with } z_i > 0\ \}$. We furthermore define a strong second-order condition for (8.11).

Definition 8.2.1. (Strong Second-Order Condition) *If*

$$p^{\intercal} \nabla_y^2 \mathcal{L}(x,y^*,z^*) p > 0,\ \forall p \in \mathcal{T}(y^*) \backslash \{0\}, \tag{8.13}$$

then (8.11) *is said to satisfy the strong second-order condition (SSOC) at* (x,y^*,z^*)*.*

In Section 2.3, we define the relaxed NLP formulation of an MPCC, which is the basis for the definition of MPCC-LICQ. We briefly recall these definitions here is a slightly different version, which is tailored to the MPCC (8.10). Therefore, we consider two index sets $\mathcal{Z}_1$ and $\mathcal{Z}_2$ with $\mathcal{Z}_1, \mathcal{Z}_2 \subset \{1, \cdots, n_h\}$ and denote their respective complement in $\{1, \cdots, n_h\}$ by $\mathcal{Z}_1^\perp$ and $\mathcal{Z}_2^\perp$. For any pair $\mathcal{Z}_1, \mathcal{Z}_2$, the relaxed NLP corresponding to the MPCC (8.10) is given by

$$\begin{array}{lll} \underset{x,y,z}{\text{minimize}} & F(x,y) & \\ \text{subject to} & \nabla_y \mathcal{L}(x,y,z) & = 0 \\ & h_i(x,y) & = 0 \quad \forall i \in \mathcal{Z}_1^\perp \\ & z_i & = 0 \quad \forall i \in \mathcal{Z}_2^\perp \\ & h_i(x,y) & \geq 0 \quad \forall i \in \mathcal{Z}_1 \\ & z_i & \geq 0 \quad \forall i \in \mathcal{Z}_2. \end{array} \tag{8.14}$$

The set of biactive components is denoted by $\mathcal{D} := \mathcal{Z}_1 \cap \mathcal{Z}_2$. We furthermore define LICQ tailored to MPCC (8.10).

Definition 8.2.2. (MPCC-LICQ) *Let $z \geq 0$ and x, y be such that $h(x,y) \geq 0$, and define index sets*

$$\mathcal{Z}_1 = \{i \in \{1, \cdots, n_h\} \mid z_i = 0\} \quad \text{and} \quad \mathcal{Z}_2 = \{i \in \{1, \cdots, n_h\} \mid h_i(x,y) = 0\}. \tag{8.15}$$

The MPCC (8.10) *is said to satisfy* MPCC-LICQ *at* (x^*, y^*) *if the corresponding relaxed NLP* (8.14) *satisfies LICQ at* (x^*, y^*).

In the following, we refer to problem

$$\begin{array}{lll} \underset{x,y,z}{\text{minimize}} & F(x,y) & \\ \text{subject to} & \nabla_y \mathcal{L}(x,y,z) & = 0 \\ & h(x,y) & \geq 0 \\ & z & \geq 0 \\ & h(x,y)^\intercal z & \leq 0, \end{array} \tag{8.16}$$

as the NLP formulation corresponding to MPCC (8.10). We note that problem (8.16) is formulated without slacks for a compact presentation. A reformulation of (8.16) using slacks does not change the results derived in this chapter. However, when solving an MPCC of type (8.16), slacks should be used to achieve a better convergence behavior and to maintain linear feasibility as shown in [FLRS06] (cf. Chapter 5). Furthermore, we follow [FLRS06] and write $h(x,y)^\intercal z \leq 0$ instead of $h(x,y)^\intercal z = 0$ in (8.16).

8.2.1. The lifting approach

We have observed that the bilevel NLP (8.8) does not inherit MPCC-LICQ even if its lower-level satisfies LICQ and SSOC. This fact is demonstrated by means of an example in the next section. We now propose a lifting technique, which allows us to guarantee MPCC-LICQ for the MPCC (8.10) resulting from the bilevel problem (8.8). The importance of MPCC-LICQ when solving MPCCs has been discussed in detail at the end of Section 8.1.

We now apply a componentwise lifting of the inequality constraints to the lower-level problem of (8.8) following [HLSB13]. Therefore, we introduce new variables $v \in \mathbb{R}^{n_h}$ and define the lifting of (8.8) as follows:

$$\begin{array}{lll} \underset{x,y,v}{\text{minimize}} & F(x,y)+\pi\,P(v) & \\ \text{subject to} & \underset{y}{\text{minimize}} & f(x,y) \\ & \text{subject to} & h(x,y) \geq v, \end{array} \tag{8.17}$$

where $\pi \in \mathbb{R}$ is a constant penalty parameter and $P : \mathbb{R}^{n_h} \to \mathbb{R}$ is a penalty function that drives v to zero and ensures convergence to a solution of the original unlifted problem. The function $P(v)$ is discussed in detail in Subsection 8.2.2. Two possible choices for the penalty term $P(v)$ are $P(v) = \|v\|_1$ or $P(v) = \|v\|_2^2$. Reformulating (8.17) as MPCC leads to the following problem:

$$\begin{array}{ll} \underset{x,y,z,v}{\text{minimize}} & F(x,y)+\pi\,P(v) \\ \text{subject to} & 0 \;\; = \nabla_y \mathcal{L}(x,y,z) \\ & 0 \;\; \leq z \;\perp (h(x,y) - v) \;\geq\; 0. \end{array} \tag{8.18}$$

In general, SSOC and LICQ of the lower-level problem are not sufficient to ensure that the MPCC (8.10) satisfies MPCC-LICQ in the solution (which is demonstrated by means of an example in the next section). If there are degenerate lower-level components, then we need $|\mathcal{D}|$ (which is at most equal to n_h) variables to regularize the Jacobian of the active constraint normals. This observation forms the basis of the following theorem, which shows that the lifted problem (8.18) satisfies an MPCC-LICQ (cf. [HLSB13]).

Theorem 8.2.1. *If the lower-level problem of* (8.17) *satisfies LICQ and SSOC at the point* (x^*, y^*, z^*, v^*)*, then the MPCC* (8.18) *satisfies MPCC-LICQ at* (x^*, y^*, z^*, v^*)*.*

Proof. We need to show that the relaxed NLP formulation of (8.18) given by

$$\begin{array}{llll} \underset{x,y,z,v}{\text{minimize}} & F(x,y)+\pi\,P(v) & & \\ \text{subject to} & \nabla_y \mathcal{L}(x,y,z) & = 0 & \\ & h_i(x,y) - v_i & = 0 & \forall i \in \mathcal{Z}_1^\perp \\ & z_i & = 0 & \forall i \in \mathcal{Z}_2^\perp \\ & h_i(x,y) - v_i & \geq 0 & \forall i \in \mathcal{Z}_1 \\ & z_i & \geq 0 & \forall i \in \mathcal{Z}_2, \end{array} \tag{8.19}$$

satisfies LICQ (according to Definition 8.2.2 with $\mathcal{Z}_1 = \{i \mid z_i = 0\}$ and $\mathcal{Z}_2 = \{i \mid h_i(x,y) - v_i = 0\}$). We now consider the gradients of the active constraints of (8.19) with respect to all optimization variables, i.e. with respect to y, z, v and x (columns are constraint normals):

$$
\begin{aligned}
J \;&:= \left(\begin{array}{cc|ccc}
\nabla_{yy}\mathcal{L}(x,y,z) & \nabla_y h_{\mathcal{Z}_1^\perp} & & & \nabla_y h_{\mathcal{D}} \\
-\nabla_y h_{\mathcal{Z}_1^\perp}^\top & & & & \\
\hline
-\nabla_y h_{\mathcal{Z}_2^\perp}^\top & & \mathbb{I} & & \\
-\nabla_y h_{\mathcal{D}}^\top & & & \mathbb{I} & \\
 & & & & -\mathbb{I} \\
\hline
 & -\mathbb{I} & & & \\
 & & & & \\
\nabla_{yx}\mathcal{L}(x,y,z) & \nabla_x h_{\mathcal{Z}_1^\perp} & & & \nabla_x h_{\mathcal{D}}
\end{array}\right) && (8.20) \\
&=: \left(\begin{array}{c|c}
A & B \\
\hline
C & D \\
\hline
E & F
\end{array}\right), && (8.21)
\end{aligned}
$$

where we have skipped zero entries and assumed that the identity matrices $\mathbb{I}$ are of appropriate size. The columns of (8.20) correspond to the constraints

$$
\left(\nabla_y \mathcal{L}(x,y,z)^\top, \quad \tilde{h}_{\mathcal{Z}_1^\perp}^\top, \quad z_{\mathcal{Z}_2^\perp}^\top, \quad z_{\mathcal{D}}^\top, \quad \tilde{h}_{\mathcal{D}}^\top\right) \tag{8.22}
$$

and the rows of (8.20) correspond to the variables

$$
\left(y^\top, \quad z_{\mathcal{Z}_1^\perp}^\top, \quad z_{\mathcal{Z}_2^\perp}^\top, \quad z_{\mathcal{D}}^\top, \quad v_{\mathcal{D}}^\top, \quad v_{\mathcal{Z}_1^\perp}^\top, \quad v_{\mathcal{Z}_2^\perp}^\top, \quad x^\top\right)^\top. \tag{8.23}
$$

Let $\bar{J}$ be the submatrix of J that consists of blocks A, B, C and D. We want to show that the constraint normals of active constraints of (8.19) (columns of (8.20)) are linearly independent. This statement is equivalent to showing that $\bar{J}$ has full rank, which means that $\bar{J}$ is of rank $(m+n_h+|\mathcal{D}|)$ (J is of size $(m+n_h+|\mathcal{D}|+|\mathcal{Z}_1^\perp|+|\mathcal{Z}_2^\perp|+n) \times (m+n_h+|\mathcal{D}|)$). As shown in, e.g., [NW99], Matrix A has full rank since SSOC and LICQ are assumed to hold for the lower-level problem of (8.17). Block D has full rank since it is a diagonal matrix with nonzeros on the diagonal. The submatrix $\bar{J}$ of J has full rank if

$$
\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \left[\begin{array}{c} q \\ r \end{array}\right] = \left[\begin{array}{c} 0 \\ 0 \end{array}\right] \Longleftrightarrow \left[\begin{array}{c} q \\ r \end{array}\right] = \left[\begin{array}{c} 0 \\ 0 \end{array}\right]. \tag{8.24}
$$

We note that we assume that 0 is a scalar, vector or matrix of zeros of appropriate size. Using the Schur complement to perform a block Gaussian elimination by multiplying $\bar{J}$ from the right with the block matrix

$$
\left[\begin{array}{c|c} \mathbb{I} & 0 \\ \hline -D^{-1}C & D^{-1} \end{array}\right] \tag{8.25}
$$

leads to

$$
\left[\begin{array}{c|c} S_D & BD^{-1} \\ \hline 0 & \mathbb{I} \end{array}\right] \left[\begin{array}{c} q \\ r \end{array}\right] = \left[\begin{array}{c} 0 \\ 0 \end{array}\right], \tag{8.26}
$$

where $S_D = A - BD^{-1}C$ is the Schur complement of D. Equation (8.26) implies that $r = 0$. It remains to show that $S_D q = 0 \Leftrightarrow q = 0$ which is true if $BD^{-1}C = 0$. The structure of $BD^{-1}C$ is as follows:

$$\begin{aligned}
BD^{-1}C &= \left[\begin{array}{c|c|c} & & \nabla_y h_{\mathcal{D}} \\ \hline & & \end{array}\right] \left[\begin{array}{c|c|c} \mathbb{I} & & \\ \hline & \mathbb{I} & \\ \hline & & -\mathbb{I} \end{array}\right] \left[\begin{array}{c|c} -\nabla_y h^{\mathsf{T}}_{\mathcal{Z}_2^{\perp}} & \\ \hline -\nabla_y h^{\mathsf{T}}_{\mathcal{D}} & \\ \hline & \end{array}\right] && (8.27) \\
&= \left[\begin{array}{c|c|c} & & -\nabla_y h_{\mathcal{D}} \\ \hline & & \end{array}\right] \left[\begin{array}{c|c} -\nabla_y h^{\mathsf{T}}_{\mathcal{Z}_2^{\perp}} & \\ \hline -\nabla_y h^{\mathsf{T}}_{\mathcal{D}} & \\ \hline & \end{array}\right] && (8.28) \\
&= 0, && (8.29)
\end{aligned}$$

which means that the Schur complement has full rank, because A has full rank. Thus, (8.24) holds and problem (8.18) satisfies MPCC-LICQ. □

The proof of Theorem 8.2.1 points out that not even SSOC and LICQ on the lower level of (8.8) are sufficient to show that (8.10) satisfies MPCC-LICQ. The matrix of the active constraint normals (8.20) of the relaxed NLP of (8.10) is of size $(n_y + n_g + n_h + n_x) \times (n_y + n_g + n_h + |\mathcal{D}|)$, and for MPCC-LICQ we need this matrix to have rank $(n_y + n_g + n_h + |\mathcal{D}|)$. This means MPCC-LICQ cannot be achieved without either having assumptions on the number of upper-level variables n_x (which is rather restrictive), or introducing at least $|\mathcal{D}|$ new variables (which lifts the MPCC to a higher dimension).

Remark 8.2.1. *MPCC-LICQ is inherited without any modifications if the lower level of* (8.8) *satisfies LICQ and SSOC, and if $|\mathcal{D}| = \emptyset$, which means that there are no degenerate indices. This follows directly from the proof of Theorem 8.2.1, or can be found in, e.g., [RW04].*

Figure 8.2 illustrates how the lifting changes the feasible set of the complementarity constraint (CC). The feasible set of $0 \leq z \perp h(x,y) \geq 0$ for $n_h = 1$ is the nonnegative part of both axes. The feasible set of the lifted CC, $0 \leq z \perp (h(x,y) - v) \geq 0$, for $n_h = 1$ is shown on the right of Figure 8.2. The original feasible set is now extended to a third dimension and includes the nonnegative part of the planes with $h(x,y) = v$ $(z \geq 0)$ and $z = 0$ $(h(x,y) - v \geq 0)$.

8.2.2. Driving v to Zero

In order to solve the original unlifted problem (8.10), we have to ensure that $v_i = 0$ $\forall i$ at the solution. There are several ways to do that. In the following, we discuss an exact and an inexact penalty function for driving v to zero, namely

$$\text{(a)} \quad P(v) = \|v\|_1, \quad \text{and} \tag{8.30a}$$

$$\text{(b)} \quad P(v) = \|v\|_2^2. \tag{8.30b}$$

The ℓ_1-norm penalty (8.30a) is an exact penalty function. It ensures that we solve the original unlifted problem when choosing π larger than a threshold $\bar{\pi}$ (the exact convergence behavior is discussed in the next section). However, the exact ℓ_1-norm penalty is

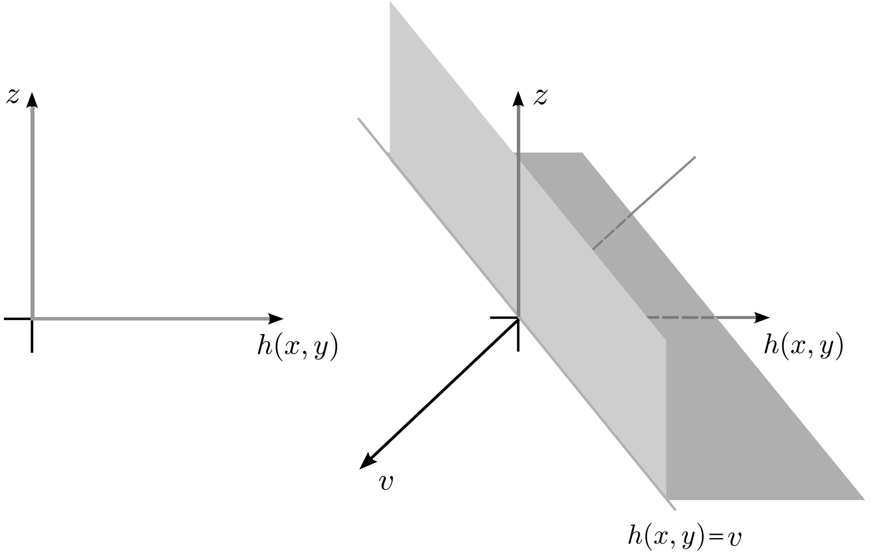


Figure 8.2.: The feasible set (in gray) of a complementarity constraint before (left) and after (right) lifting.

a nonsmooth function. Instead of using a numerical method which is able to handle the nonsmoothness of the function (as proposed in, e.g., [IPS11]), we add the additional constraint $v \geq 0$ which ensures the componentwise nonnegativity of v and the smoothness of the penalty $\|v\|_1$. For the lifting of the complementarity constraint this means that the we cut the two planes in Figure 8.2 at $v = 0$ and neglect the part with $v < 0$. The proof of Theorem 8.2.1 and the geometry of the feasible set directly imply that we keep MPCC-LICQ as long as the additional constraint $v_i \geq 0$, $i-1, \dots, n_h$ is not active at a point where $h_i(x, y)$ and z_i are zero for $i \in \{1, \cdots, n_h\}$. Our numerical experiments clearly indicate that having MPCC-LICQ everywhere but points where $h_i(x, y) = z_i = v_i = 0$ still stabilizes the MPCC and leads to a faster convergence. Furthermore, this approach allows us to use standard NLP solvers.

The squared ℓ_2-norm penalty (8.30b) is an analytic function, which is a nice property since most numerical NLP algorithms assume smoothness of the problem. However, with an inexact penalty function we have to solve a sequence of problems where $\pi \to \infty$ in order to ensure to converge to a solution of the original unlifted problem (8.10). This is an undesirable property. However, we decided to also investigate the inexact penalty because our numerical experience clearly shows, that having MPCC-LICQ everywhere decreases the number of iterations. Furthermore, to avoid an ill-conditioned Hessian, reformulations such as the one described in [NW99, Chapter 17] could be used. Both penalty functions are tested, compared and discussed in Chapter 13, and the convergence properties of penalty (8.30a) are analyzed in the next section.

8.3. A motivating example

As already mentioned in the last section, we now demonstrate that an MPCC as defined in (8.10), which arises from a bilevel NLP like (8.8), does in general not satisfy MPCC-LICQ in the solution, even if the lower-level NLP of (8.8) satisfies LICQ and SOSC (both defined in Section 8.2). We furthermore show that the lifted formulation of our example indeed does satisfy MPCC-LICQ in the solution. Consider the following bilevel NLP:

$$\begin{array}{lll} \underset{x}{\text{minimize}} & -x + 2y_1 + y_2 & \\ \text{subject to} & \underset{y:=(y_1,y_2)}{\text{minimize}} & (x - y_1)^2 + y_2^2 \\ & \text{subject to} & y_1, y_2 \geq 0, \end{array} \tag{8.31}$$

with the solution $(x^*, y_1^*, y_2^*) = (0, 0, 0)$. LICQ and SSOC are satisfied for the lower-level problem. The gradient of the Lagrangian of the lower-level problem is given by

$$\nabla_y \mathcal{L}(x, y, z) = \begin{pmatrix} -2x + 2y_1 - z_1 \\ 2y_2 - z_2 \end{pmatrix}, \tag{8.32}$$

and the MPCC corresponding to (8.31) is given by

$$\begin{array}{lll} \underset{x,y,z}{\text{minimize}} & -x + 2y_1 + y_2 & \\ \text{subject to} & 0 & = \begin{pmatrix} -2x + 2y_1 - z_1 \\ 2y_2 - z_2 \end{pmatrix} \\ & 0 & \leq z \perp y \geq \ 0. \end{array} \tag{8.33}$$

In the solution, we have $z_1 = z_2 = 0$. This means that all constraints are weakly active at the solution and the relaxed NLP of (8.33) is

$$\begin{array}{lll} \underset{x,y,z}{\text{minimize}} & -x + 2y_1 + y_2 & \\ \text{subject to} & 0 & = \begin{pmatrix} -2x + 2y_1 - z_1 \\ 2y_2 - z_2 \end{pmatrix} \\ & 0 & = y \\ & 0 & = z. \end{array} \tag{8.34}$$

Clearly, (8.34) does not satisfy LICQ because the Jacobian of the constraints (the columns are the constraints normals) given by

$$\begin{pmatrix} 2 & 0 & 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 \\ -2 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{8.35}$$

has rank 5. This fact implies that the MPCC (8.10) does not satisfy MPCC-LICQ. The degeneracy problem is caused by the weakly active constraints. However, if the lower-level NLP satisfies strict complementarity, LICQ and SSOC in the solution, then it can be shown that the resulting MPCC satisfies MPCC-LICQ (see [RW04], or Remark 8.2.1).

Let us now consider lifting proposed in the last section for (8.31):

$$\begin{array}{lll} \underset{x,v:=(v_1,v_2)}{\text{minimize}} & -x+2y_1+y_2+\pi P(v) & \\ \text{subject to} & \underset{y}{\text{minimize}} & (x-y_1)^2+y_2^2 \\ & \text{subject to} & y_1 \geq v_1 \\ & & y_2 \geq v_2, \end{array} \tag{8.36}$$

where $v := (v_1, v_2) \in \mathbb{R}^2$ are the additional lifting variables. The corresponding MPCC is then given by

$$\begin{array}{ll} \underset{x,y,v,z}{\text{minimize}} & -x+2y_1+y_2+\pi P(v) \\ \text{subject to} & 0 \ \ = \nabla_y \mathcal{L}(x,y,z) \\ & 0 \ \ \leq z \perp y \geq \ \ v. \end{array} \tag{8.37}$$

The Jacobian of active constraints of the NLP formulation of (8.37) has now two additional rows for variables v_1 and v_2:

$$\left(\begin{array}{cccccc} 2 & 0 & 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 \\ -2 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \end{array}\right), \tag{8.38}$$

which extends (8.35) by two rows that ensure the Jacobian has full rank. After lifting problem (8.31), the active constraint gradients of the NLP formulation of the resulting MPCC are linearly independent and MPCC-LICQ is satisfied for (8.37) everywhere. In the next section, we discuss the convergence behavior of SQP methods applied to lifted MPCCs.

8.4. Convergence results

Fletcher et al. [FLRS06] provide convergence results for SQP methods applied to MPCCs under relatively mild conditions (described in the first section of this chapter). One important assumption for the convergence to strongly stationary points is that the MPCC satisfies MPCC-LICQ. This condition keeps the multipliers bounded and ensures a fast local convergence. We now derive an assumption which is weaker than MPCC-LICQ for problems of type (8.10) and for penalty (8.30a). The analysis in [FLRS06] centers around the relaxed NLP

$$\begin{array}{llll} \underset{x,y,z}{\text{minimize}} & F(x,y) & & \\ \text{subject to} & \nabla_y \mathcal{L}(x,y,z) & = 0 & \\ & z_i & = 0 & \forall i \in \mathcal{Z}_2^{\perp} \\ & h_i(x,y) & \geq 0 & \forall i \in \mathcal{Z}_1 \\ & z_i & \geq 0 & \forall i \in \mathcal{Z}_2. \end{array} \tag{8.39}$$

Without loss of generality and to keep the notation simple, we assume $\mathcal{Z}_1^{\perp} = \emptyset$. The Lagrangian of (8.39) is given by

$$L(x,y,z,\lambda,\mu,\nu) := F(x,y) - \nabla_y \mathcal{L}(x,y,z)^{\mathsf{T}} \lambda - z^{\mathsf{T}} \mu - h(x,y)^{\mathsf{T}} \nu, \tag{8.40}$$

with Lagrange multipliers $\lambda \in \mathbb{R}^m, \mu =: (\mu_1, \mu_2) \in \mathbb{R}^{|\mathcal{Z}_2^\perp|+|\mathcal{Z}_2|}$ with $\mu_2 \geq 0$ and $\nu \in \mathbb{R}^{|\mathcal{Z}_1|}$ with $\nu \geq 0$. Dual feasibility of (8.39) is given as

$$\begin{aligned}
\nabla_x L(x,y,z,\lambda,\mu,\nu) &= \nabla_x F(x,y) & -\nabla_{yx}\mathcal{L}(x,y,z)^\intercal \lambda & \; -\nabla_x h(x,y)^\intercal \nu & & = 0 \\
\nabla_y L(x,y,z,\lambda,\mu,\nu) &= \nabla_y F(x,y) & -\nabla_{yy}\mathcal{L}(x,y,z)^\intercal \lambda & \; -\nabla_y h(x,y)^\intercal \nu & & = 0 \\
\nabla_z L(x,y,z,\lambda,\mu,\nu) &= & \nabla_y h(x,y)^\intercal \lambda & & -\mu & = 0.
\end{aligned} \tag{8.41}$$

Combining conditions (8.41), feasibility of (8.39) and complementary slackness

$$h_i(x,y)\nu_i = 0 \;\forall\, i \in \mathcal{Z}_1 \quad \text{and} \quad \mu_{2i} z_i = 0 \;\forall\, i \in \mathcal{Z}_2 \tag{8.42}$$

gives us the KKT conditions of (8.39). Now consider the relaxed lifted NLP

$$\begin{array}{lll}
\underset{x,y,z,v}{\text{minimize}} & F(x,y) + \pi\|v\|_1 & \\
\text{subject to} & \nabla_y \mathcal{L}(x,y,z) = 0 & \\
& z_i = 0 & \forall i \in \mathcal{Z}_2^\perp \\
& h_i(x,y) - v_i \geq 0 & \forall i \in \mathcal{Z}_1 \\
& z_i \geq 0 & \forall i \in \mathcal{Z}_2.
\end{array} \tag{8.43}$$

To simplify the analysis, we will replace v by $v = v^+ - v^-$ with $v^+, v^- \geq 0$ and $\|v\|_1 = v^+ + v^-$, giving rise to the following conditions for dual feasibility of (8.43):

$$\begin{aligned}
\nabla_x L(\cdot) &= \nabla_x F(x,y) & -\nabla_{yx}\mathcal{L}(x,y,z)^\intercal \lambda & \; -\nabla_x h(x,y)^\intercal \nu & & & = 0 \\
\nabla_y L(\cdot) &= \nabla_y F(x,y) & -\nabla_{yy}\mathcal{L}(x,y,z)^\intercal \lambda & \; -\nabla_y h(x,y)^\intercal \nu & & & = 0 \\
\nabla_z L(\cdot) &= & \nabla_y h(x,y)^\intercal \lambda & & -\mu & & = 0 \\
\nabla_{v^+} L(\cdot) &= \pi\, e & & +\nu & & -\xi^+ & = 0 \\
\nabla_{v^-} L(\cdot) &= \pi\, e & & -\nu & & -\xi^- & = 0,
\end{aligned} \tag{8.44}$$

where the dependencies of L are skipped for a compact presentation, e denotes a vector of ones of appropriate size, and ξ^+, ξ^- are the Lagrange multipliers for $v^+, v^- \geq 0$. We note, that in order to keep the notation intuitive, L in (8.41) denotes the Lagrangian of (8.39), and L in (8.44) denotes the Lagrangian of (8.43). In the following, (x,y,z) is called a KKT point of (8.39) if the KKT conditions (as described in, e.g., [FLRS06]) are satisfied at (x,y,z). We can now state the following two propositions.

Proposition 8.4.1. *Let (x^*, y^*, z^*) be a KKT point of (8.39) and assume that $\pi > \|\nu^*\|_\infty$. Then it follows that (x^*, y^*, z^*, v^*) with $v^* = 0$ is a KKT point of (8.43).*

Proof. Clearly, the first three equations of (8.44) are satisfied at (x^*, y^*, z^*). We also see that $\xi^+ = \pi e + \nu^* \geq 0$ since $\pi, \nu^* \geq 0$. With $v^* = 0$ we get complementarity for the constraints $v^+, v^- \geq 0$. It remains for us to show that $\xi^- = \pi e - \nu^* \geq 0$, but we choose $\pi > \|\nu^*\|_\infty = \max_i \nu_i^*$, hence $\xi^- \geq 0$. □

Proposition 8.4.2. *Let (x^*, y^*, z^*, v^*) be a KKT point of (8.43). If $v^* = 0$, then it follows that (x^*, y^*, z^*) is a KKT point of (8.39).*

Proof. Dual feasibility of (8.39) (equation (8.41)) directly follows from (8.44). The same holds true for the nonnegativity of μ_2^* and ν^* in the solution of (8.39). Given that $v^* = 0$, we have primal feasibility and complementarity for the unlifted problem (8.39). □

This result means that if we solve (8.43) with the solution (x^*, y^*, z^*, v^*) and $v^* = 0$, the point (x^*, y^*, z^*) is also a KKT point of (8.39), and we can now apply the convergence analysis from [FLRS06]. Furthermore, this means that instead of requiring MPCC-LICQ in the convergence proof of [FLRS06], it suffices for MPCCs arising from bilevel NLPs like problem (8.10) to require LICQ and SSOC for the lower-level NLP. LICQ and SSOC are reasonable assumptions which are satisfied for most well-posed problems. Hence, in order to have bounded multipliers and a faster convergence in practice for bilevel NLPs with LICQ and SOSC on the lower level, is suffices to lift the complementarity constraint as described in (8.18) and to choose a π that is sufficiently large. Without lifting the complementarity constraint, we would have to require strict complementarity of the lower-level problem ($|\mathcal{D}| = 0$) in order to guarantee MPCC-LICQ. This is a rather restrictive condition which is already violated in simple examples like the one described in (8.31).

As discussed in the last section, the ℓ_1-norm is a nonsmooth function, but we wish to use standard SQP methods which usually require the objective and the constraints to be at least twice continuously differentiable. If we now assume that F, g and h are twice continuously differentiable functions, and if we add the additional constraint $v \geq 0$ to (8.43), we obtain the smooth problem

$$\begin{array}{lllll} \underset{x,y,z}{\text{minimize}} & F(x,y) + \pi \sum_{i=1}^{n_h} v_i & & & \\ \text{subject to} & \nabla_y \mathcal{L}(x,y,z) & = 0 & & \\ & z_i & = 0 & \forall i \in \mathcal{Z}_2^{\perp} & \\ & h_i(x,y) - v_i & \geq 0 & \forall i \in \mathcal{Z}_1 & \\ & z_i & \geq 0 & \forall i \in \mathcal{Z}_2 & \\ & v & \geq 0, & & \end{array} \tag{8.45}$$

which is sufficiently smooth for standard SQP methods. The disadvantage of the additional constraint $v \geq 0$ is that we cannot guarantee MPCC-LICQ anymore at points (x, y, z, v) with $h_i(x, y) = z_i = v_i = 0$ for at least one $i \in \{1, \cdots, n_h\}$. However, in the next section we show that our lifting technique still stabilizes the MPCC and leads to a faster convergence. We now briefly investigate how the convergence results from Propositions 8.4.1 and 8.4.2 change for (8.45). The only difference in the proof will be in the dual feasibility conditions (8.44), where the last two equations have to be replaced by

$$\nabla_v L(\cdot) = \pi \, e + \nu - \xi = 0, \tag{8.46}$$

where ξ is the Lagrange multiplier for $v \geq 0$. We have to ensure that $\xi = \pi e + \nu \geq 0$, which is true since $\pi, \nu \geq 0$. This implies that Proposition 8.4.1 and Proposition 8.4.2 also hold for problem (8.45). Moreover, the additional assumption on π is not needed anymore.

Corollary 8.4.1. *Let (x^*, y^*, z^*) be a KKT point of (8.39). Then (x^*, y^*, z^*, v^*) is also a KKT point of (8.45) with $v^* = 0$. And vice versa, if (x^*, y^*, z^*, v^*) is a KKT point of (8.45) and if $v^* = 0$, then (x^*, y^*, z^*) is also a KKT point of (8.39).*

8.5. Lifting general mathematical programs with complementarity constraints

In this section, we discuss the generalization of our lifting approach to general MPCCs (which do not originate from a bilevel program). In particular, we consider

$$\begin{array}{lll} \underset{x:=(x_1,x_2),y}{\text{minimize}} & F(x,y) & \\ \text{subject to} & 0 & \leq h(x,y) \\ & 0 & \leq x_1 \perp x_2 \geq \ \ 0, \end{array} \tag{8.47}$$

with $x_1, x_2 \in \mathbb{R}^n$, $y \in \mathbb{R}^m$ and $h(x,y) \in \mathbb{R}^{n_h}$. The relaxed NLP of (8.47) is given by

$$\begin{array}{lll} \underset{x,y}{\text{minimize}} & F(x,y) & \\ \text{subject to} & 0 & \leq h(x,y) \\ & 0 & = x_1 \quad \forall i \in \mathcal{Z}_2^{\perp} \\ & 0 & = x_2 \quad \forall i \in \mathcal{Z}_1^{\perp} \\ & 0 & \leq x_1 \quad \forall i \in \mathcal{Z}_2 \\ & 0 & \leq x_2 \quad \forall i \in \mathcal{Z}_1, \end{array} \tag{8.48}$$

with $\mathcal{Z}_1 = \{i \in \{1, \cdots, n\} \mid x_{1i} = 0\}$ and $\mathcal{Z}_2 = \{i \in \{1, \cdots, n\} \mid x_{2i} = 0\}$. Lifting general MPCCs of the form (8.48) leads to the following problem:

$$\begin{array}{lll} \underset{v,x:=(x_1,x_2),y}{\text{minimize}} & F(x,y) + \pi P(v) & \\ \text{subject to} & 0 & \leq h(x,y) - v \\ & 0 & \leq x_1 \perp x_2 \geq \ \ 0. \end{array} \tag{8.49}$$

The relaxed NLP for this MPCC is given by:

$$\begin{array}{lll} \underset{x,y,v}{\text{minimize}} & F(x,y) + \pi P(v) & \\ \text{subject to} & 0 & \leq h(x,y) - v \\ & 0 & = x_1 \quad \forall i \in \mathcal{Z}_2^{\perp} \\ & 0 & = x_2 \quad \forall i \in \mathcal{Z}_1^{\perp} \\ & 0 & \leq x_1 \quad \forall i \in \mathcal{Z}_2 \\ & 0 & \leq x_2 \quad \forall i \in \mathcal{Z}_1, \end{array} \tag{8.50}$$

where the constraints $h(x,y)$ are lifted with variables $v \in \mathbb{R}^{n_h}$. The lifting of general MPCCs is not as elegant as in the bilevel case because of the lack of information about the structure of the constraints $h(x,y) \geq 0$. However, lifting still guarantees linearly independent constraint normals. To see this, we consider the Jacobian of active constraints of the lifted relaxed NLP (8.50):

$$J := \left(\begin{array}{c|c|c|c|c} -\mathbb{I} & & & & \\ \hline \nabla_{x_{1\mathcal{Z}_2^{\perp}}} h & \mathbb{I} & & & \\ \hline \nabla_{x_{1\mathcal{D}}} h & & \mathbb{I} & & \\ \hline \nabla_{x_{2\mathcal{Z}_1^{\perp}}} h & & & \mathbb{I} & \\ \hline \nabla_{x_{2\mathcal{D}}} h & & & & \mathbb{I} \\ \hline \nabla_{x_{1\mathcal{Z}_1^{\perp}}} h & & & & \\ \hline \nabla_{x_{2\mathcal{Z}_2^{\perp}}} h & & & & \\ \hline \nabla_y h & & & & \end{array} \right)$$

where the notation $x_{j\mathcal{I}}$ denotes the subvector of x_j whose indices belong to $\mathcal{I}$ for $j = 1, 2$. The column of J correspond to the constraints

$$\Big((h - v)^\top, x_{1\mathcal{Z}_2^\perp}^\top, x_{1\mathcal{D}}^\top, x_{2\mathcal{Z}_1^\perp}^\top, x_{2\mathcal{D}}^\top\Big) \tag{8.51}$$

and the rows of J correspond to the variables

$$(v^\top, x_{1\mathcal{Z}_2^\perp}^\top, x_{1\mathcal{D}}^\top, x_{2\mathcal{Z}_1^\perp}^\top, x_{2\mathcal{D}}^\top, x_{1\mathcal{Z}_1^\perp}^\top, {x_{2\mathcal{Z}_2^\perp}}^\top, y^\top)^\top. \tag{8.52}$$

Zero entries are skipped. Clearly, the Jacobian J has full column rank without any restrictions on the structure of the constraint function $h(x, y)$. More general complementarity constraints of the form

$$0 \leq \; G(x) \;\perp\; H(x) \;\geq 0 \tag{8.53}$$

can easily be treated by introducing slack variables. No further restrictions on the functions G, H are needed, since G and H are then lifted in the same way as the inequalities $h(x, y) \geq 0$.

It is straightforward to show that the convergence results for the exact penalty from the end of last section remain true for the lifted general MPCC (8.50) with $P(v) = \|v\|_1$. Numerical test are promising and show that lifted MPCCs behave in the same way as lifted bilevel NLPs. The computational results are discussed in detail in Chapter 13.

8.6. Lifting hierarchical dynamic optimization problems

In the previous sections of this chapter, we have seen that solving a lifted formulation of an MPCC allows us to guarantee MPCC-LICQ in the solution, which plays an important role in the convergence behavior of SQP methods applied to MPCCs. Furthermore, MPCC-LICQ guarantees the convergence to B-stationary points (under the assumptions mentioned in the first section of this chapter), since we know that strong stationarity and B-stationarity are equivalent if MPCC-LICQ holds. In this section, we incorporate the lifting idea proposed in Section 8.2 into our direct all-at-once approach for hierarchical dynamic optimization problems described in Chapter 5, where we also end up with a very special MPCC that is solved with a tailored Newton-based method. As discussed in detail in Chapter 5, it is indispensable to exploit the structure inherited from the bilevel setting and the discretization/parameterization in order obtain an efficient method that is able to solve large real-word problems. In this section, we discuss how the problem structure of the NLP in Section 5.4 resulting from the hierarchical dynamic optimization problem (5.2) changes if we lift its complementarity constraint.

Lifting NLP (5.69) leads to the following problem:

$$\begin{array}{lrcl}
\underset{\substack{s,w,\bar{q}\\ \gamma,p\\ \lambda,\mu,\\ v}}{\text{minimize}} & \multicolumn{3}{l}{\dfrac{1}{2}\displaystyle\sum_{i=0}^{n_m}\sum_{j=1}^{n_h}\frac{(h_j(x(t_i^m),q,p)-\eta_{ij})^2}{\sigma_{ij}^2} \quad + \pi P(v)} \\
\text{subject to} & 0 & = & \nabla_y \mathcal{L}(y,\lambda,\mu) \\
 & 0 & \le & \mu \\
 & 0 & = & \left(C^{\text{ieq}}(s,w,\bar{q},p)-v\right)^{\mathsf{T}} \mu \\
 & 0 & = & C^{\text{eq}}(s,w,\bar{q},p) \\
 & 0 & \le & C^{\text{ieq}}(s,w,\bar{q},p)-v \\
 & 1 & = & \sum_{i=1}^{n_{\mathcal{M}}} \gamma_i, \quad \gamma \ge 0 \\
 & b^{\text{lower}_p} & \le & p \ \le \ b^{\text{upper}_p},
\end{array} \tag{8.54}$$

with $v \in \mathbb{R}^{n_\mu}$ (where n_μ is defined in Chapter 5) and the remaining variables of (8.54) as defined in Chapter 5. For the penalty term $P(v)$ we consider two choices:

a) $P(v) = \|v\|_1, \ v \ge 0$ or

b) $P(v) = \|v\|_2^2$.

As discussed in the previous sections, penalty a) is an exact penalty, and with the additional constraint $v \ge 0$, a smooth function. Penalty term b) is also smooth, but an inexact penalty function. We now discuss the constraint Jacobian of (8.54) and compare it to the constraint Jacobian (5.72) of the unlifted NLP (5.69). Therefore, we consider the following matrix:

$$J_{\text{lift}}^{\mathsf{T}} := \left(\begin{array}{c|c|c|c|c}
C^{\text{eq}\,\mathsf{T}} & \nabla_y \mathcal{L}^{\mathsf{T}} & (C^{\text{ieq}}\text{-}v)^{\mathsf{T}}\mu & \sum_{i=1}^{n_{\mathcal{M}}}\gamma_i\text{-}1 & (C^{\text{ieq}}\text{-}v)^{\mathsf{T}} \\
\hline
J1 & J2 & J3 & & J11 \\
\hline
J4 & J5 & & J9 & \\
\hline
 & J6 & & & \\
\hline
 & J7 & J\overline{8} & & \\
\hline
 & & J10 & & J12
\end{array}\right)
\begin{array}{c}
\\ y \\ \gamma \\ p \\ \lambda \\ \mu \\ v
\end{array} . \tag{8.55}$$

The matrix J_{lift} in (8.55) shows the main blocks of the constraints Jacobian of the lifted NLP (8.54) without simple bounds. In (5.72), the inequality constraints C^{ieq} just contain simple bounds and are not included. In (8.55), we have a new block corresponding to the constraints $\left(C^{\text{ieq}} - v\right)^{\mathsf{T}}$. The simple bounds in (5.69) are now replaced by the linear constraints $\left(C^{\text{ieq}} - v\right) \ge 0$. We furthermore add n_μ rows in (8.55), which correspond to the lifting variables v. The complementarity constraint now also depends on v. Comparing J

in (5.72) with J_{lift} in (8.55), we have new blocks $J10, J11$ and $J12$, which are described in detail in the following. Furthermore block $J8$ is replaced by $\bar{J}8$ due to dependency of the complementarity constraint on v. Block $\bar{J}8$ is defined as follows.

$$\bar{J}8 := \nabla_\mu \left(\left(C^{\text{ieq}} - v \right)^\mathsf{T} \mu \right) = C^{\text{ieq}} - v = \begin{pmatrix} \begin{pmatrix} s \\ w \\ q \end{pmatrix} - b^{\text{lower}} \\ b^{\text{upper}} - \begin{pmatrix} s \\ w \\ q \end{pmatrix} \\ q_1^s \\ \vdots \\ q_{n_{\text{slacks}}}^s \end{pmatrix} - v. \tag{8.56}$$

Block $J10$ is defined as

$$J10 := \nabla_v \left(\left(C^{\text{ieq}} - v \right)^\mathsf{T} \mu \right) = - \begin{pmatrix} \mu_1 \\ \vdots \\ \mu_{n_\mu} \end{pmatrix}, \tag{8.57}$$

and block $J11$ is given by

$$J11 := \nabla_y \left(C^{\text{ieq}} - v \right)^\mathsf{T} = \left(\begin{array}{ccc|ccc|ccc} 1 & & & -1 & & & & & \\ & \ddots & & & \ddots & & & & \\ & & 1 & & & -1 & & & \\ \hline & & & & & & 1 & & \\ & & & & & & & \ddots & \\ & & & & & & & & 1 \end{array} \right), \tag{8.58}$$

where the rows correspond to the variables

$$\left(s^\mathsf{T}, w^\mathsf{T}, q^\mathsf{T} \mid {q^s}^\mathsf{T} \right)^\mathsf{T} \tag{8.59}$$

and the columns correspond to

$$\left(\left(\begin{array}{c} \begin{pmatrix} s \\ w \\ q \end{pmatrix} - b^{\text{lower}} \\ \hline b^{\text{upper}} - \begin{pmatrix} s \\ w \\ q \end{pmatrix} \\ \hline q_1^s \\ \vdots \\ q_{n_{\text{slacks}}}^s \end{array} \right) - v \right)^\mathsf{T}. \tag{8.60}$$

It remains to define block $J12$:

$$J12 := \nabla_v \left(C^{\text{ieq}} - v \right)^\mathsf{T} = \begin{pmatrix} -1 & & \\ & \ddots & \\ & & -1 \end{pmatrix}. \tag{8.61}$$

The nonnegativity constraint on v for penalty a) is imposed as simple bound and not considered in the constraint Jacobian (8.55). In the last paragraph, we have discussed how the lifting approach introduced in Section 8.2 changes the NLP (5.69) resulting from our direct all-at-once approach (cf. Chapter 5) applied to a hierarchical dynamic optimization problem of the form (5.2) and the corresponding constraint Jacobian. It remains to discuss the Hessian of the Lagrangian of (8.54). The modified objective of NLP (5.69) in (8.54) is given by

$$\frac{1}{2}\sum_{i=0}^{n_m}\sum_{j=1}^{n_h}\frac{(h_j(x(t_i^m),q,p)-\eta_{ij})^2}{\sigma_{ij}^2} \quad +\pi P(v). \tag{8.62}$$

The unlifted NLP (5.69) is solved with a Generalized Gauß-Newton method as described in Chapter 5. We now discuss whether the assumptions for a Gauß-Newton approximation of the Hessian described in Section 2.2.2 of (8.54) are still satisfied for the lifted NLP, in spite of the additional term $\pi P(v)$. The objective (8.62) can be written in the general form

$$\Gamma(z,v) := \tfrac{1}{2}\|\mathfrak{F}(z)\|_2^2 + \pi P(v), \tag{8.63}$$

with its derivatives

$$\nabla_z\Gamma(z,v) = \nabla_z\mathfrak{F}(z)^{\mathsf{T}}\mathfrak{F}(z) \tag{8.64}$$

$$\nabla_v\Gamma(z,v) = \pi\nabla_v P(v). \tag{8.65}$$

Penalty a) ($P(v) = \|v\|_1, v \geq 0$) can be written as $P(v) = \sum_{i=1}^{n_\mu} v_i,\ v \geq 0$, and we have $\nabla_v P(v) = (1,\dots,1)^{\mathsf{T}}$. For penalty b) ($P(v) = \|v\|_2^2$) it is $\nabla_v P(v) = 2v$. We consider the following NLP

$$\begin{aligned}&\underset{z,v}{\text{minimize}} && \tfrac{1}{2}\|\mathfrak{F}(z)\|_2^2 + \pi P(v)\\ &\text{subject to} && \mathfrak{G}(z,v) = 0,\end{aligned} \tag{8.66}$$

where we summarize the dimension of $\mathfrak{F}(z)$ by $n_{\mathfrak{F}}$, $\mathfrak{G}(z,v)$ by $n_{\mathfrak{G}}$, z is of size n_z and v of size n_v. In particular, NLP (8.66) represents (8.54), where $\mathfrak{G}(z,v)$ includes the equality constraints and all active inequality constraints at (z,v). The gradient and the Hessian of the Lagrangian $L(z,v,\nu)$ of (8.66) with Lagrange multiplier ν are given by

$$\nabla_z L(z,v,\nu) = \nabla_z\mathfrak{F}(z)^{\mathsf{T}}\mathfrak{F}(z) - \nabla_z\mathfrak{G}(z,v)^{\mathsf{T}}\nu \tag{8.67}$$

$$\nabla_v L(z,v,\nu) = \pi\nabla_v P(v) - \nabla_v\mathfrak{G}(z,v)^{\mathsf{T}}\nu \tag{8.68}$$

and

$$[\nabla_{zz}L(z,v,\nu)]_{jl} = \left[\sum_i \frac{\partial^2\mathfrak{F}_i(z)}{\partial z_j\partial z_l}\mathfrak{F}_i(z) + \sum_i \frac{\partial\mathfrak{F}_i(z)}{\partial z_j}\frac{\partial\mathfrak{F}_i(z)}{\partial z_l}\right]_{jl} \tag{8.69}$$

$$- \left[\sum_i \frac{\partial^2\mathfrak{G}_i(z,v)}{z_j z_l}\nu_i\right]_{jl}, \quad j,l = 1,\cdots,n_z \tag{8.70}$$

$$[\nabla_{vv}L(z,v,\nu)]_{jl} = \pi\left[\sum_i \frac{\partial^2 P_i(v)}{\partial v_j\partial v_l}\right]_{jl}, \quad j,l = 1,\cdots,n_v \tag{8.71}$$

$$[\nabla_{zv}L(z,v,\nu)]_{jl} = -\left[\sum_i \frac{\partial^2\mathfrak{G}_i(z,v)}{z_j v_l}\nu_i\right]_{jl}, \quad j = 1,\cdots,n_z,\ l = 1,\cdots,n_v. \tag{8.72}$$

For penalty a), we have

$$[\nabla_{vv} L(z,v,\nu)]_{jl} = \pi \left[\sum_i \frac{\partial^2 P_i(v)}{\partial v_j \partial v_l} \right]_{jl} = 0 \tag{8.73}$$

for $j, l = 1, \cdots, n_v$ and penalty b) leads to

$$[\nabla_{vv} L(z,v,\nu)]_{jl} = \pi \left[\sum_i \frac{\partial^2 P_i(v)}{\partial v_j \partial v_l} \right]_{jl} = 2\pi \ \mathbb{I}_{n_v} \tag{8.74}$$

for $j, l = 1, \cdots, n_v$. The mixed derivative $\nabla_{zv} L(z,v,\nu)$ just includes the term

$$\nabla_{zv} \left(\left(\left(C^{\text{ieq}}(s,w,\bar{q},p) - v \right)^\intercal \mu \right) \nu_k \right) = -\nabla_{zv} \left(v^\intercal \mu \ \ \nu_k \right) \tag{8.75}$$

which corresponds to the complementarity constraint, and ν_k describes the associated Lagrange multiplier. All remaining constraints only depend linearly on the lifting variable v and their second derivative with respect to v and z is zero. The only nonzero part in (8.72) is given by

$$\nabla_{\mu v} \left(\left(\left(C^{\text{ieq}}(s,w,\bar{q},p) - v \right)^\intercal \mu \right) \nu_k \right) = \begin{pmatrix} \nu_k & & \\ & \ddots & \\ & & \nu_k \end{pmatrix}. \tag{8.76}$$

We now summarize the observations of the last paragraph for penalty term a) and b) separately. We start with penalty a). In Section 2.2.2, the Gauß-Newton method is introduced and the approximation of the Hessian of the Lagrangian by (2.28) is based on the following analysis (cf. [Boc87, Die01]):

- The residuals $\mathfrak{F}_i(z)$ are expected to be small close to the solution. Hence, the first term in the right-hand side of (8.69) can be neglected.
- The gradient of the Lagrangian (8.67) with respect to z is expected to become small close to the solution. Additionally, we know that the residuals $\mathfrak{F}_i(z)$ are small close to the solution. Both arguments imply that the Lagrange multiplier ν can be expected to be small close to the solution, which is the reason for neglecting the term in (8.70).

This implies the approximation

$$\begin{aligned} [\nabla_{zz} L(z,v,\nu)]_{jl} &= \left[\sum_i \frac{\partial^2 \mathfrak{F}_i(z)}{\partial z_j \partial z_l} \mathfrak{F}_i(z) + \sum_i \frac{\partial \mathfrak{F}_i(z)}{\partial z_j} \frac{\partial \mathfrak{F}_i(z)}{\partial z_l} \right]_{jl} \\ & \quad - \left[\sum_i \frac{\partial^2 \mathfrak{G}_i(z,v)}{z_j z_l} \nu_i \right]_{jl} \\ &\approx \sum_i \frac{\partial \mathfrak{F}_i(z)}{\partial z_j} \frac{\partial \mathfrak{F}_i(z)}{\partial z_l}. \end{aligned} \tag{8.77}$$

Furthermore, for penalty term a), we know that $[\nabla_{vv} L(z,v,\nu)]_{jl} = 0$ for $j, l = 1, \cdots, n_v$ and that $[\nabla_{zv} L(z,v,\nu)]_{jl}$ just includes entries ν_k, for which we know that they become small close to the solution. Hence,

$$\nabla_{(z,v)(z,v)} L(z,v,\nu) \ \approx \ \left(\begin{array}{c|c} \left(\nabla_z \mathfrak{F}(z) \right)^\intercal \nabla_z \mathfrak{F}(z) & \\ \hline & \end{array} \right), \tag{8.78}$$

where the upper left block corresponds to the variable z, and the remaining zero blocks correspond to v. This means the Generalized Gauß-Newton method for hierarchical dynamic optimization problems described in Section 5.5 is still valid if we solve the lifted NLP (8.54) with penalty term a).

For penalty b), the arguments remain the same, but we need additional entries in the lower right block:

$$\nabla_{(z,v)(z,v)} L(z,v,\nu) \quad \approx \quad \left(\begin{array}{c|c} \left(\nabla_z \mathfrak{F}(z)\right)^{\mathsf{T}} \nabla_z \mathfrak{F}(z) & \\ \hline & 2\pi \ \mathbb{I}_{n_z} \end{array} \right), \tag{8.79}$$

which is due to (8.74).

In Chapter 13, we present and discuss the numerical performance of the lifting approach derived in this chapter for the MacMPEC collection of MPCCs [Ley00] and for an illustrative hierarchical dynamic optimization problem.

Chapter 9.

The software package ParaOCP

In this chapter, we discuss the implementation of the direct-all-at-once approach described in Chapter 5 in the C++ software package ParaOCP. We start with a brief introduction and a formulation of the hierarchical dynamic optimization problem that can be solved with ParaOCP. The following section describes the structure of the software package with a detailed explanation of its modules. Finally, we illustrate the implementation of a hierarchical dynamic optimization problem by means of an example.

9.1. Introduction and problem formulation

The C++ software package ParaOCP (standing for PARAmeter estimation in Optimal Control Problems) implements the direct-all-at once approach proposed in Chapter 5 for hierarchical dynamic optimization problems of type (4.1) including:

- a nonlinear least-squares-type objective function
- nonlinear constraints on upper-level variables
- a nonlinear multi-stage lower-level optimal control problem (OCP) with
    - a nonlinear lower-level objective function of Bolza-type (Mayer and Lagrange term)
    - a nonlinear system of ordinary differential equations (ODEs) as constraints
    - multiple model stages with discontinuous transitions expressed by nonlinear transition conditions
    - nonlinear mixed control-state constraints
    - coupled and decoupled nonlinear equality and inequality multi-point constraints.

Furthermore, the software package includes a module for solving nonlinear one-level multi-stage parameter estimation problems constrained by a system of ODEs, nonlinear transition conditions, state constraints and nonlinear coupled/decoupled equality/inequality multi-point constraints. A second module allows to solve nonlinear one-level multi-stage OCPs, again, constrained by a system of ODEs, nonlinear transition conditions, mixed control-path constraints and nonlinear coupled/decoupled equality/inequality multi-point constraints.

The implementation of the data structure is based on C++ classes, whereas the numerical algorithms are written in C using BLAS/LAPACK routines [WP05] for a fast linear algebra.

9.2. The software structure, its modules and features

This section starts with a description of the main software architecture and how the problem variables and functions are arranged in public C++ classes. Afterwards, the structure of the algorithm presented in Chapter 5 is discussed and illustrated, followed by a description of the user interface, which coincides with the definition of two constructors. Finally, features and dependencies of ParaOCP are discussed.

9.2.1. Software architecture

Figure 9.1 illustrates the main structure with three levels. On the top level, we have two classes called `Para_OCP` and `Para_OCP_num`. Class `Para_OCP` includes attributes and methods that describe the continuous problem, which is either a parameter estimation problem constrained by an OCP, a one-level OCP, or a one-level parameter estimation problem. Attributes of `Para_OCP` include, e.g., the number of differential states and controls. Typical methods are functions computing the objective (lower and upper-level objective if we intend to solve a bilevel problem), or the ODE's right-hand side.

Numerical tolerances are summarized in the class `Para_OCP_num`, which includes, e.g., the integration tolerance, the initial stepsize for the integrator, or the stopping tolerance.

On the second level of the software package, we have the class `NLP_base`, which inherits from `Para_OCP` and `Para_OCP_num`. The class `NLP_base` summarizes attributes and methods describing the discretized problem resulting from a parameter estimation problem constrained by an OCP, a one-level OCP, or a one-level parameter estimation problem. All attributes that are not similar for the three problem types, are not included in `NLP_base`. Typical attributes are the number of multiple shooting intervals for the parameterization of the differential states or all NLP variables and their bounds. Virtual methods of `NLP_base` include all functions required by the NLP solver, as, e.g., a function computing the NLP's objective and constraints, the constraint Jacobian, the Hessian of the Lagrangian, and the product of the Hessian times a vector. These methods are then implemented in either `NLP_ocp` if a one-level optimal control or parameter estimation problem is solved, or `NLP_par` if a parameter estimation problem constrained by an OCP is solved.

The classes `NLP_ocp` and `NLP_par` are on the third level and inherit from `NLP_base`. Class `NLP_ocp` includes attributes and methods that are concerned with solving discretized optimal control or parameter estimation problems, whereas `NLP_par` contains attributes and methods for the solution of parameter estimation problems that are constrained by an OCP. Both classes also include methods writing information about the solution and for plotting to text files. Furthermore, different variants of the solution algorithm are implemented on this level, as, e.g., the approximation of the Hessian of the Lagrangian or the treatment of the complementarity constraint in class `NLP_par`.

9.2.2. Modules

Figure 9.2 illustrates the main function call for solving a one-level parameter estimation problem or OCP, whereas Figure 9.3 shows this procedure for solving a parameter estimation problem, which is constrained by an OCP. Both figures highlight the modular structure of ParaOCP. The common structure of the procedure of solving a one-level and bilevel dynamic optimization problem with a direct (all-at-once in case of the bilevel problem) approach is that it results in a structured NLP, which is then solved with a tailored Newton-type method. For solving the resulting NLP, we in both cases need routines for the computation of

- the (lower/upper-level) objective
- the NLP's equality and inequality constraints
- the constraint Jacobian
- the Hessian of the Lagrangian and/or the product of the Hessian with a vector,

which of course differ for one-level and bilevel problems. These four routines can be considered as modules in the solution process of dynamic optimization problems in ParaOCP and are chosen depending on the problem class we are currently considering (cf. Figures 9.2 and 9.3). All algorithmic parts that are concerned with the continuous problem like

- the multiple shooting parameterization
- the solution of the ODE
- the solution of the variational differential equation (VDE)
- the structure of the NLP inherited by the discretization

are called within the NLP routines.

9.2.3. User interface

We now briefly discuss the arguments of the constructor of NLP_ocp and NLP_par, which coincides with the user interface. The prototype of the constructor of NLP_ocp is given by

```
NLP_ocp( // problem dimension
         int NMOS, int NX, int NQ, int NU, int NP,
         int* NSHOOT, int NLSQ, int NCONuser,
         // ODE's right hand side and its derivatives
         FFCN_type* ffcn_ptr, FFCN_type* dffcn_ptr,
         FFCN_type* dffcn_qup_ptr,
         // least-squares objective and its derivatives
         LSQFCN_type lsqfcn_ptr, LSQFCN_type dlsqfcn_ptr,
         LSQFCN_type dlsqfcn_qwp_ptr,
         // Mayer objective and its derivatives
         MFCN_type* mfcn_ptr, MFCN_type* dmfcn_ptr, MFCN_type* dmfcn_q_ptr,
         // Lagrange objective and its derivatives
         FFCN_type* lfcn_ptr, FFCN_type* dlfcn_ptr, FFCN_type* dlfcn_qu_ptr,
         // control discretization
         U_DISC_type u_disc_ptr,
         // discretized coupled/decoupled equality/inequality constraints
         CONFCN_type confcn_ptr, DCONFCN_type dconfcn_ptr,
         DCONFCN_type dconfcn_qwp_ptr,
// initialization of NLP variables
INITnlp_ocp_type init_nlp_ptr,
// user output
OUTocp_type out_ptr,
// numerical constants                                          (9.1)
double TOL, int MAXITER,
double STEP_INT_INI, double STEP_INT_MIN,
// output/algorithmic flags
int PLOTLEVEL, int HESS, int HESSlower, int LSQsimple);
```

The constructor of class NLP_par for solving bilevel problems differs only slightly from (9.1). The differences are highlighted in the following prototype:

```
NLP_par( // problem dimension
         ...
         int NCONuserUPPER,
         ...
         // initialization of NLP variables
         INITnlp_par_type init_nlp_ptr,                         (9.2)
         ...
         // output/algorithmic flags
         ...
         int LIFTING, int NQSLACKS, int P_IN_RHS );
```

We now briefly explain the constants and variables used in (9.1) and (9.2). Table 9.1 contains the constants in (9.1) and (9.2) that describe the problem dimensions. To pass functions like, e.g., the ODE's right-hand side to the constructor, we use function pointers. However, if we have multiple model stages, we need a function for the ODE's right-hand side for each model stage (details about the implementation of multiple model stages with discontinuous transitions can be found in Section 12.1). Therefore, the constructors (9.1) and (9.2) have arguments like

Name	Type	Description
NMOS	int	number of model stages
NX	int	number of differential states
NQ	int	number of time-independent control values
NU	int	number of controls
NP	int	number of parameters
NSHOOT	int*	array of numb. of mult. shoot. intervals for each model stage
NLSQ	int	number of residuals in the least-squares objective
NCONuser	int	number of discretized equality/inequality constraints
NCONuserUPPER	int	number of upper-level discretized eq./ineq. constraints

Table 9.1.: Overview of constants describing the problem dimension used in constructors (9.1) and (9.2).

```
FFCN_type* ffcn_ptr,   using the type definition                  (9.3)
typedef void (*FFCN_type)(double* t, double* x,                    (9.4)
                          double* q, double* u, double* p, double* res);
```

Type definitions of a similar form are used for the remaining functions in (9.1) and (9.2). The argument (9.3) of the constructors (9.1) and (9.2) is an array of function pointers – one function pointer for each model stage. If the function name, e.g. `ffcn_ptr` includes an additional leading `d`, as e.g. `dffcn_ptr`, this describes the derivative of the respective function with respect to x in case of `dffcn_ptr`, and with respect to q, u and p in case of `dffcn_qup_ptr`. The user can either provide symbolic derivatives, derivatives computed using automatic differentiation or finite differences. Table 9.2 explains the basic function names used in the constructors (9.1) and (9.2). All further details can be found in the documentation of ParaOCP.

Name	**Type**	**Description**
`ffcn_ptr`	`FFCN_type*`	array of right-hand sides of the ODEs
`lsqfcn_ptr`	`LSQFCN_type`	residual of least-squares objective function
`mfcn_ptr`	`MFCN_type*`	array of Mayer objective functions
`lfcn_ptr`	`FFCN_type*`	array of Lagrange objective functions
`u_disc_ptr`	`U_DISC_type*`	array of control discretization functions
`confcn_ptr`	`CONFCN_type`	all discretized equality/inequality constraints
`init_nlp_ptr`	`INITnlp_ocp_type`	initialization of all NLP variables (one-level prob.)
`init_nlp_ptr`	`INITnlp_par_type`	initialization of all NLP variables (bilevel prob.)
`out_ptr`	`OUTocp_type`	user output function

Table 9.2.: Description of function pointers used in constructors (9.1) and (9.2).

Table 9.3 lists numerical constants and flags of ParaOCP that can directly be set by the user. The first column of Table 9.3 explains whether this is a feature for one-level problems (denoted by O) or bilevel problems (denoted by B), and the third column shows possible values if the variable is a simple flag. The fourth column explains the variable and its settings.

The current dependencies of ParaOCP and input/output possibilities are listed in Table 9.4. The number of dependencies of ParaOCP is kept as low as possible to allow a simple maintenance and to ensure that the code runs on different machines with Unix-like operating systems (as, e.g., Linux or Mac OS). Furthermore, the integrator interface and the NLP solver interface are written in a standard form, which allows to replace these modules by alternatives.

We finally note that the description of ParaOCP in this chapter does not include all features and does not serve as a user manual. This chapter presents a brief overview of the main structure of the software package, it demonstrated how one-level and bilevel problems are formulated, and describes some of ParaOCP's features. For more details, we refer to the user manual and the source code of ParaOCP.

9.3. An example

In this section, we consider an example to demonstrate its implementation first as one-level OCP and then as bilevel parameter estimation problem with the OCP as constraint in ParaOCP. The OCP describes the optimal movement of a rocket car with non-constant

Prob.	**Var.**	**Value**	**Description**
O/B	`SPARSE`	0	a dense constraint Jacobian is stored
		1	only nonzero entries of the constraint Jacobian are stored
B	`CC_AS_PENALTY`	0	no penalty formulation of the compl. constraint
		1	the compl. constraint is formulated as ℓ_1-norm penalty
B	`LIFTING`	0	no lifting of the complementarity constraint
		1	lifting (cf. Sec. 8.2) is applied
O/B	`HESS`		upper-level Hessian approximation
		1	constant diagonal Hessian
		2	BFGS updates
		3	finite differences
		4	Gauß-Newton Hessian
B	`HESSlower`		lower-level Hessian approximation
		1	constant diagonal Hessian
		2	finite differences
		3	efficient Hessian calc. (poss. for a certain probl. struct.)
O/B	`NQSLACKS`	n_{slacks}	number of slacks in q used in `confcn_ptr`
B	`P_IN_RHS`	0	the ODE's right-hand side does not depend on p
		1	the ODE's right-hand side depends on p
O/B	`DPND`		array used in `confcn_ptr` to store which constraint depends on which multiple shooting interval
B	`LSQsimple`	n_{xmeas}	number of diff. states that are measured directly

Table 9.3.: Some features of ParaOCP.

mass and including friction (cf. Section 6.2, [HSB12]). We briefly recall the formulation of the OCP:

$$
\begin{array}{rrcll}
\underset{x,u,q_0}{\text{minimize}} & \multicolumn{3}{l}{-\gamma_1\, m(1) + \gamma_2\, q_0 =: \Phi} & \\
\text{subject to} & 0 & = & \dot{z}(t) - q_0 v(t), & t \in [0,1] \\
& 0 & = & \dot{v}(t) - q_0(u(t) - p_1 v(t)^2)/m(t), & t \in [0,1] \\
& 0 & = & \dot{m}(t) + q_0 p_2 u(t)^2, & t \in [0,1] \\
& 0 & = & z(0), \quad 0 = v(0), \quad 0 = m(0) - 1 & \\
& 0 & = & z(1) - 10, \quad 0 = v(1) - 1 & \\
& 0 & \le & 1 - u(t), & t \in [0,1] \\
& 0 & \le & 1 + u(t), & t \in [0,1].
\end{array}
\tag{9.5}
$$

The variables in (9.5) are described in Section 6.2. The OCP (9.5) has three differential states z, v and m, one control function u, and one time-independent control value q_0 in (9.5). We have one model stage, no mixed control-state constraints, just simple bounds and boundary conditions.

Integrator:	explicit Runge-Kutta-Fehlberg 4th/5th order ([BPL$^+$06])
NLP solver:	filterSQP ([FL02a])
Linear algebra:	BLAS/LAPACK ([BDD$^+$02, ABB$^+$99])
Input:	C++/C-file
Output:	command line/iteration log, text file, Gnuplot ([GNU]), Matlab ([Mat13])

Table 9.4.: Dependencies, input and output of ParaOCP.

The optimal control problem

The quantities γ_1 and γ_2 are constant since we first consider the implementation of the OCP (9.5) in ParaOCP. Therefore, we define the following constants:

```
const int NMOS      = 1;     const double TOL           = 1e-6;
int NSHOOT[NMOS]    = {20};  const int MAXITER          = 100;
const int NX        = 3;     const double STEP_INT_INI  = 1e-8;
const int NU        = 1;     const double STEP_INT_MIN  = 1e-14;
const int NQ        = 1;     const int PLOTLEVEL        = 1;
const int NCONuser  = 0;     const int HESS             = 1;
```

We furthermore define helper variables:

```
int NS[NMOS]  = (NSHOOT[0]+1)*NX;  int sumNSHOOT  = sum_int(NSHOOT,NMOS);
int sumNS     = sum_int(NS,NMOS);  int NW         = sumNSHOOT;
```

The function `sum_int(int* x, int n)` calculates the sum of the first n entries of the integer array x. We furthermore choose a piecewise constant control discretization, and the multiple shooting and control discretization grid are the same. The ODE's right hand side function is defined as

```
//right-hand side ODE
void ffcn(double* t, double* x, double* q, double* u, double* p, double* res){
      double p1  = .1;
      double p2  = .1;
      res[0]     = q[0]*x[1];
      res[1]     = q[0]*(u[0] - p1*x[1]*x[1])/x[2];
      res[2]     = -q[0]*p2*u[0]*u[0];
};
```

and the Mayer objective function is implemented by the following function.

```
//mayer objective function
void mfcn(double* x, double* q, double* p, double* res){
     double a = 0.5;
     double b = 0.5;
     res[0] = -a*x[2]+b*q[0];
};
```

As mentioned above, the user also passes functions for the derivative of the ODE's right-hand side with respect to x, q, u and p, which can be either implemented using the symbolic derivatives, automatic differentiation or finite differences. The same types of derivatives have to be provided for the Mayer objective function. For a compact presentation, we just state the prototypes of these functions:

```
void dffcn(double* t, double* x, double* q, double* u, double* p, double* res);
void dffcn_qup(double* t, double* x, double* q, double* u, double* p, double* res);
void dmfcn(double* x, double* q, double* p, double* res);
void dmfcn_q(double* x, double* q, double* p, double* res);
```

It remains to define the function `init_nlp`, which allows the user to initialize all NLP variables like, e.g., the multiple shooting parameterization variables s, the control discretization variables w, controls values q, parameters p and the corresponding upper and lower bounds. Boundary conditions are implemented by setting upper and lower bound to the same value. Furthermore, upper and lower bound for the user constraints (no simple bounds) are set in `init_nlp`. The prototype is given by

```
void init_nlp(double* ms_grid, double* s, double* lb_s, double* ub_s,
              double* u_grid, double* w, double* lb_w, double* ub_w,
              double* q, double* lb_q, double* ub_q,
              double* lb_confcn, double* ub_confcn,
              int* s_fix, int* w_fix, int* q_fix, double* x_sca);
```

The type of control discretization (e.g. piecewise constant, piecewise linear, etc.) can be implemented in the function `u_disc`, whose prototype is denoted as

```
void u_disc(double* t, double* w, double* res, double* u_grid);
```

The `main`-function written by the user then calls the constructor of `NLP_ocp` passing the constants and functions defined above. Before calling the constructor, we need to define the arrays of function pointers for the ODE's right hand side and the Mayer objective. Both have just one entry in this example since we have only one model stage. After calling the constructor, the method `solve` of `NLP_ocp` is called. The user's `main`-function is given by:

```
int main(){


       FFCN_type ffcn_ptrs[NMOS];
       FFCN_type dffcn_ptrs[NMOS];
       FFCN_type dffcn_qup_ptrs[NMOS];
       MFCN_type mfcn_ptrs[NMOS];
       MFCN_type dmfcn_ptrs[NMOS];
       MFCN_type dmfcn_q_ptrs[NMOS];

       ffcn_ptrs[0] = ffcn;
       dffcn_ptrs[0] = dffcn;
       dffcn_qup_ptrs[0] = dffcn_qup;
       mfcn_ptrs[0] = mfcn;
       dmfcn_ptrs[0] = dmfcn;
       dmfcn_q_ptrs[0] = dmfcn_q;

       // call contructor of NLP_ocp
       NLP_ocp rocketcar( NMOS, NX, NQ, NU, 0, NSHOOT, 0, NCONuser,
            ffcn_ptrs, dffcn_ptrs, dffcn_qup_ptrs,
            NULL, NULL, NULL,
            mfcn_ptrs, dmfcn_ptrs, dmfcn_q_ptrs, NULL,NULL,NULL,
            u_disc, NULL,NULL,NULL,
            init_nlp,NULL, TOL, MAXITER, STEP_INT_INI,
            STEP_INT_MIN, PLOTLEVEL, HESS, 0, 0);

       // solve OCP
       rocketcar.solve();

       return 0;
};
```

The hierarchical dynamic optimization problem

Next, we consider the implementation of the following bilevel problem:

$$\begin{array}{lll}
\underset{\substack{x,u,q_0,\\ p,\gamma_1}}{\text{minimize}} & \displaystyle\frac{1}{2}\sum_{i=0}^{n_T}\sum_{j=1}^{3}\frac{(x_j(t_i)-\eta_{ij})^2}{\sigma_{ij}^2} & \\
\text{subject to} & \underset{x,u,q_0}{\text{minimize}} \quad -\gamma_1\, m(1) + 0.5\, q_0 & \\
& \text{subject to} \quad 0 = \dot{z}(t) - q_0 v(t), & t\in[0,1] \\
& \qquad\qquad\quad 0 = \dot{v}(t) - q_0(u(t) - p_1 v(t)^2)/m(t), & t\in[0,1] \\
& \qquad\qquad\quad 0 = \dot{m}(t) + q_0 p_2 u(t)^2, & t\in[0,1] \\
& \qquad\qquad\quad 0 = z(0), \quad 0 = v(0), \quad 0 = m(0) - 1 & \\
& \qquad\qquad\quad 0 = z(1) - 10, \quad 0 = v(1) - 1 & \\
& \qquad\qquad\quad 0 \le 1 - u & t\in[0,1] \\
& \qquad\qquad\quad 0 \le 1 + u & t\in[0,1] \\
& 0 \le \gamma_1 &
\end{array} \tag{9.6}$$

with standard deviations $\sigma_{ij} = 0.04$ and measurements η_{ij} of all differential states for $i = 0, \cdots, n_T$ and $j = 1, 2, 3$. In (9.6), we have upper variables γ_1, p_1, p_2, which are in

ParaOCP summarized in variable p. We keep all functions and constants implemented for solving the OCP, and add additional functions and constants needed for the bilevel setting. For the implementation of (9.6), we define the following additional helper variables:

```
const int NLSQ          = NX*(sumNSHOOT+1);  const int NP       = 3;
const int NCONuserUPPER = 0;                 const int HESS     = 5;
const int HESSlower     = 3;                 const int LIFTING  = 0;
const int LSQsimple     = 3;                 const int P_IN_RHS = 1;
```

We choose a Gauß-Newton Hessian approximation for the upper-level problem (`HESS=5`) and an efficient Hessian calculation (cf. Chapter 5) for the lower-level problem (`HESSlower=3`). The constant `LSQsimple` is set to `3`, which means that all `3` differential states are measured directly. The flag `P_IN_RHS` is set to `1`, which means that components of `p` appear in the ODE's right-hand side. If the flag `P_IN_RHS` is `0`, the ODE's right-hand side does not depend on p and we skip the computation of first and second-order sensitivities with respect to p in order to obtain a fast and efficient algorithm.

The function `lsqfcn` implements the residual of the upper-level objective. For problem (9.6), `lsqfcn` is defined as follows:

```
void lsqfcn(double* s, double* q, double* w, double* p, double* res){
      //compute least-squares residual for all nodes
      res[0] = s[0] - 2.800000e-02;
      res[1] = s[1] - 3.700000e-02;
      res[2] = s[2] - 1.006000e+00;
      res[3] = s[3] - 8.193413e-02;
            ⋮
}
```

We also need to pass derivatives `lsqfcn` of to the constructor of `NLP_par`. For a compact presentation, we just state the prototypes for `dlsqfcn` and `dlsqfcn_qwp`:

```
void dlsqfcn(double* s, double* q, double* w, double* p, double* res):
void dlsqfcn_qwp(double* s, double* q, double* w, double* p, double* res);
```

The lower-level objective `mfcn` has to be adapted to the bilevel setting, which means that constant `a` is replaced by variable `p[0]`.

```
void mfcn(double* x, double* q, double* p, double* res){
      res[0] = -p[0]*x[2]+0.5*q[0];
};
```

The derivatives `dmfcn` and `dmfcn_q` also have to be adapted. For the bilevel setting, the function `init_nlp` for initialization of the NLP variables passed to the constructor `NLP_par` has a different prototype than the `init_nlp` in the one-level setting. We have additional variables

```
double* p, double* lb_p, double* ub_p,
double* lambda, double* lb_lambda, double* ub_lambda,
double* mu, double* lb_mu, double* ub_mu,
double* lb_confcn_upperlevel, double* ub_confcn_upperlevel,
double* p_fix,
```

where `p` are the upper-level variables with lower and upper bounds `double* lb_p` and `double* ub_p`. In the bilevel problem, the multipliers of the lower-level problem are variables, and are denoted by `lambda` and `mu`. The arguments `lb_confcn_upperlevel` and ub_confcn_upperlevel are used for upper and lower bounds of upper-level constraints if NCONuserUPPER is greater than zero. The additional variable p_fix can be used to fix components of p. The prototype of the NLP initialization function for the bilevel problem is given by

```
void init_nlp(double* ms_grid, double* s, double* lb_s, double* ub_s,
              double* u_grid, double* w, double* lb_w, double* ub_w,
              double* q, double* lb_q, double* ub_q,
              double* p, double* lb_p, double* ub_p,
              double* lambda, double* lb_lambda, double* ub_lambda,
              double* mu, double* lb_mu, double* ub_mu,
              double* lb_confcn_upperlevel, double* ub_confcn_upperlevel,
              int* s_fix, int* w_fix, int* q_fix, int* p_fix, double* x_sca);
```

In the user's main-function, we just have to adapt the call of the constructor for the bilevel problem. Instead of calling the constructor of NLP_ocp

```
NLP_ocp rocketcar( NMOS, NX, NQ, NU, 0, NSHOOT, 0, NCONuser,
     ffcn_ptrs, dffcn_ptrs, dffcn_qup_ptrs,
     NULL, NULL, NULL,
     mfcn_ptrs, dmfcn_ptrs, dmfcn_q_ptrs, NULL,NULL,NULL,
     u_disc, NULL,NULL,NULL,
     init_nlp,NULL, TOL, MAXITER, STEP_INT_INI,
     STEP_INT_MIN, PLOTLEVEL, HESS, 0, 0);
```

we call the constructor of NLP_par

```
NLP_par raketenwagen( NMOS, NX, NQ, NU, NP, NSHOOT, NLSQ, NCONuser,
                      NCONuserUPPER, ffcn_ptrs, dffcn_ptrs, dffcn_qup_ptrs,
                      lsqfcn, dlsqfcn, lsqfcn_qwp,
                      mfcn_ptrs, dmfcn_ptrs, dmfcn_q_ptrs,
                      NULL, NULL, NULL,u_disc,
                      NULL,NULL,NULL, init_nlp,NULL,
                      TOL, MAXITER, STEP_INT_INI, STEP_INT_MIN,
                      PLOTLEVEL, HESS, HESSlower,
                      LSQsimple, LIFTING,0,P_IN_RHS);
```

with the additional functions, variables and constants. The user's main-function for the implementation of the bilevel problem (9.6) is given by

```
int main(){


      //set functions for each stage
      FFCN_type ffcn_ptrs[NMOS];
      FFCN_type dffcn_ptrs[NMOS];
      FFCN_type dffcn_qup_ptrs[NMOS];
      MFCN_type mfcn_ptrs[NMOS];
      MFCN_type dmfcn_ptrs[NMOS];
      MFCN_type dmfcn_q_ptrs[NMOS];

      ffcn_ptrs[0] = ffcn;
      dffcn_ptrs[0] = dffcn;
      dffcn_qup_ptrs[0] = dffcn_qup;
      mfcn_ptrs[0] = mfcn;
      dmfcn_ptrs[0] = dmfcn;
      dmfcn_q_ptrs[0] = dmfcn_q;

      // call contructor of NLP_par
      NLP_par raketenwagen( NMOS, NX, NQ, NU, NP, NSHOOT, NLSQ, NCONuser,
                            NCONuserUPPER, ffcn_ptrs, dffcn_ptrs, dffcn_qup_ptrs,
                            lsqfcn, dlsqfcn, dlsqfcn_qwp,
                            mfcn_ptrs, dmfcn_ptrs, dmfcn_q_ptrs,
                            NULL, NULL, NULL,u_disc,
                            NULL,NULL,NULL, init_nlp,NULL,
                            TOL, MAXITER, STEP_INT_INI, STEP_INT_MIN,
                            PLOTLEVEL, HESS, HESSlower,
                            LSQsimple, LIFTING,0,P_IN_RHS);

      // solve Para_OCP
      raketenwagen.solve();

      return 0;
};
```

For further details about the implementation of the direct all-at-once approach proposed in Chapter 9, we refer to the user manual or the source code of ParaOCP.

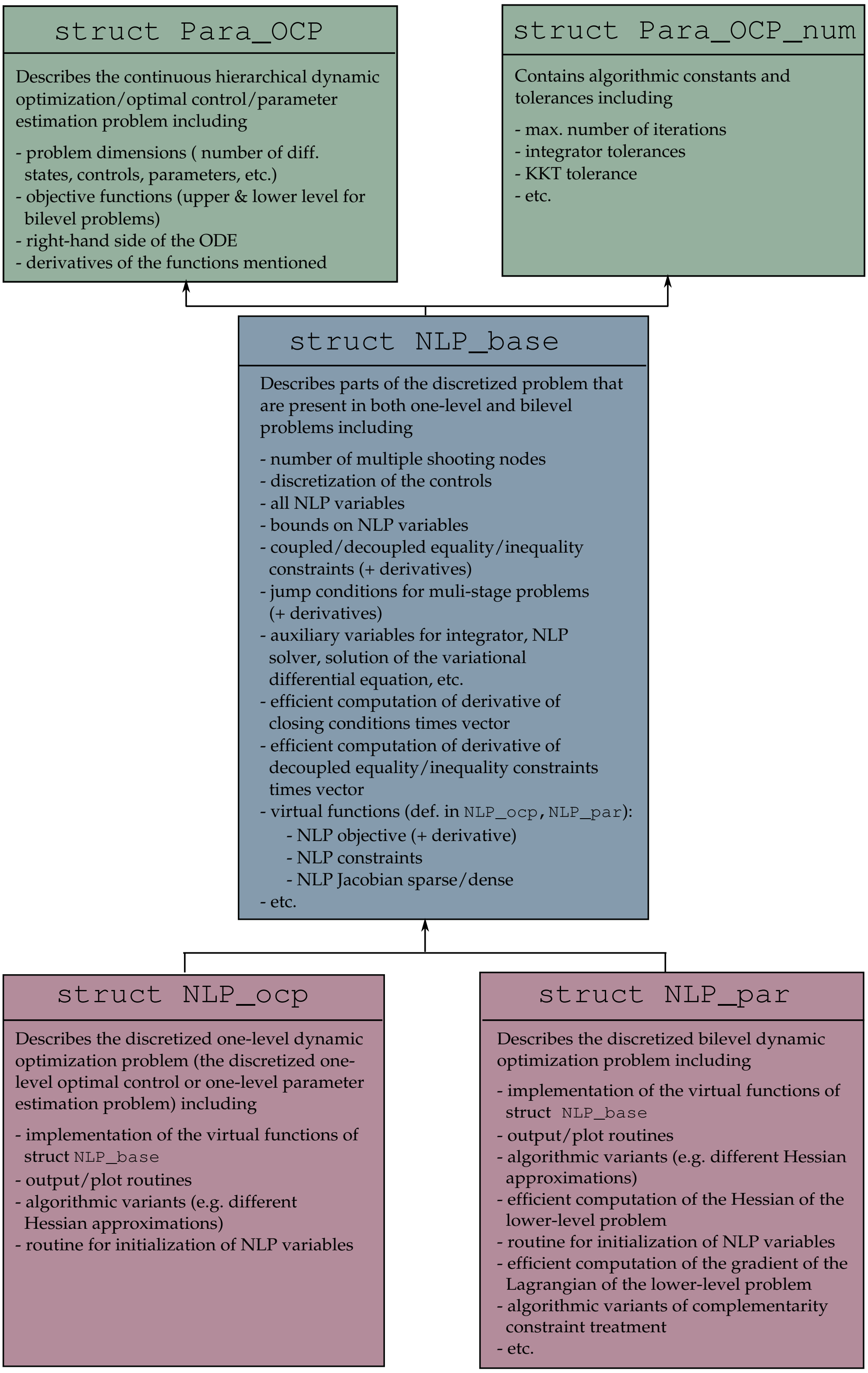

struct Para_OCP
Describes the continuous hierarchical dynamic optimization/optimal control/parameter estimation problem including
- problem dimensions (number of diff. states, controls, parameters, etc.)
- objective functions (upper & lower level for bilevel problems)
- right-hand side of the ODE
- derivatives of the functions mentioned
struct Para_OCP_num
Contains algorithmic constants and tolerances including
- max. number of iterations
- integrator tolerances
- KKT tolerance
- etc.
struct NLP_base
Describes parts of the discretized problem that are present in both one-level and bilevel problems including
- number of multiple shooting nodes
- discretization of the controls
- all NLP variables
- bounds on NLP variables
- coupled/decoupled equality/inequality constraints (+ derivatives)
- jump conditions for muli-stage problems (+ derivatives)
- auxiliary variables for integrator, NLP solver, solution of the variational differential equation, etc.
- efficient computation of derivative of closing conditions times vector
- efficient computation of derivative of decoupled equality/inequality constraints times vector
- virtual functions (def. in NLP_ocp, NLP_par):
- NLP objective (+ derivative)
- NLP constraints
- NLP Jacobian sparse/dense
- etc.
struct NLP_ocp
Describes the discretized one-level dynamic optimization problem (the discretized one-level optimal control or one-level parameter estimation problem) including
- implementation of the virtual functions of struct NLP_base
- output/plot routines
- algorithmic variants (e.g. different Hessian approximations)
- routine for initialization of NLP variables
struct NLP_par
Describes the discretized bilevel dynamic optimization problem including
- implementation of the virtual functions of struct NLP_base
- output/plot routines
- algorithmic variants (e.g. different Hessian approximations)
- efficient computation of the Hessian of the lower-level problem
- routine for initialization of NLP variables
- efficient computation of the gradient of the Lagrangian of the lower-level problem
- algorithmic variants of complementarity constraint treatment
- etc.

Figure 9.1.: Illustration of the design of ParaOCP.

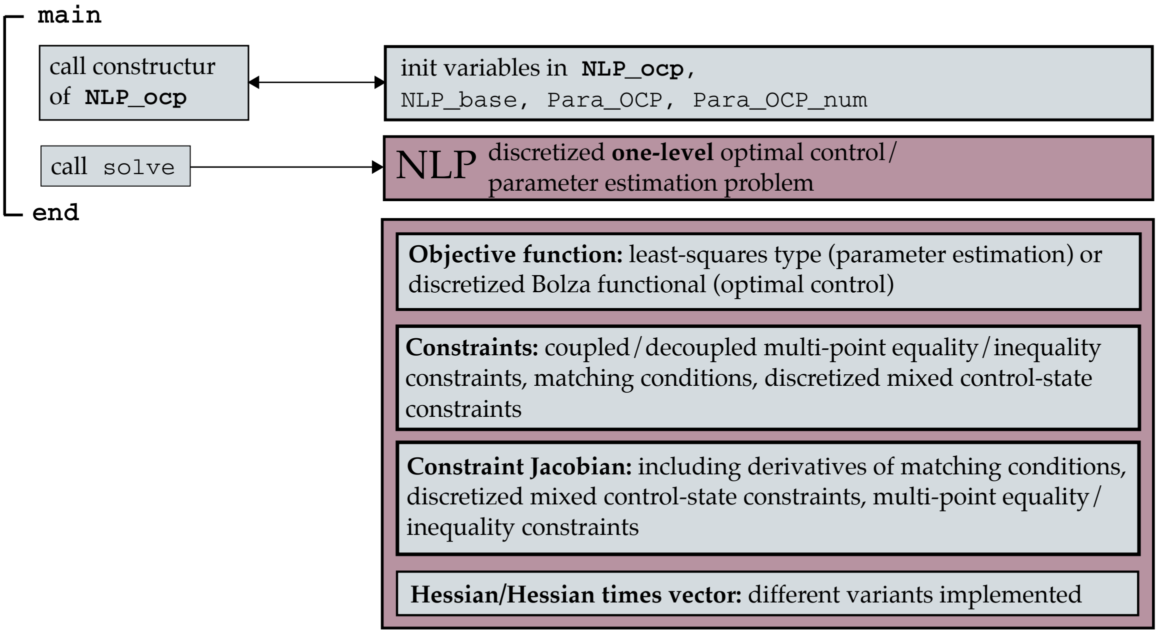

main
call constructur of NLP_ocp
init variables in NLP_ocp, NLP_base, Para_OCP, Para_OCP_num
call solve
NLP discretized **one-level** optimal control / parameter estimation problem
end
Objective function: least-squares type (parameter estimation) or discretized Bolza functional (optimal control)
Constraints: coupled / decoupled multi-point equality / inequality constraints, matching conditions, discretized mixed control-state constraints
Constraint Jacobian: including derivatives of matching conditions, discretized mixed control-state constraints, multi-point equality / inequality constraints
Hessian/Hessian times vector: different variants implemented

Figure 9.2.: Illustration of the modules of ParaOCP for an optimal control or parameter estimation problem.

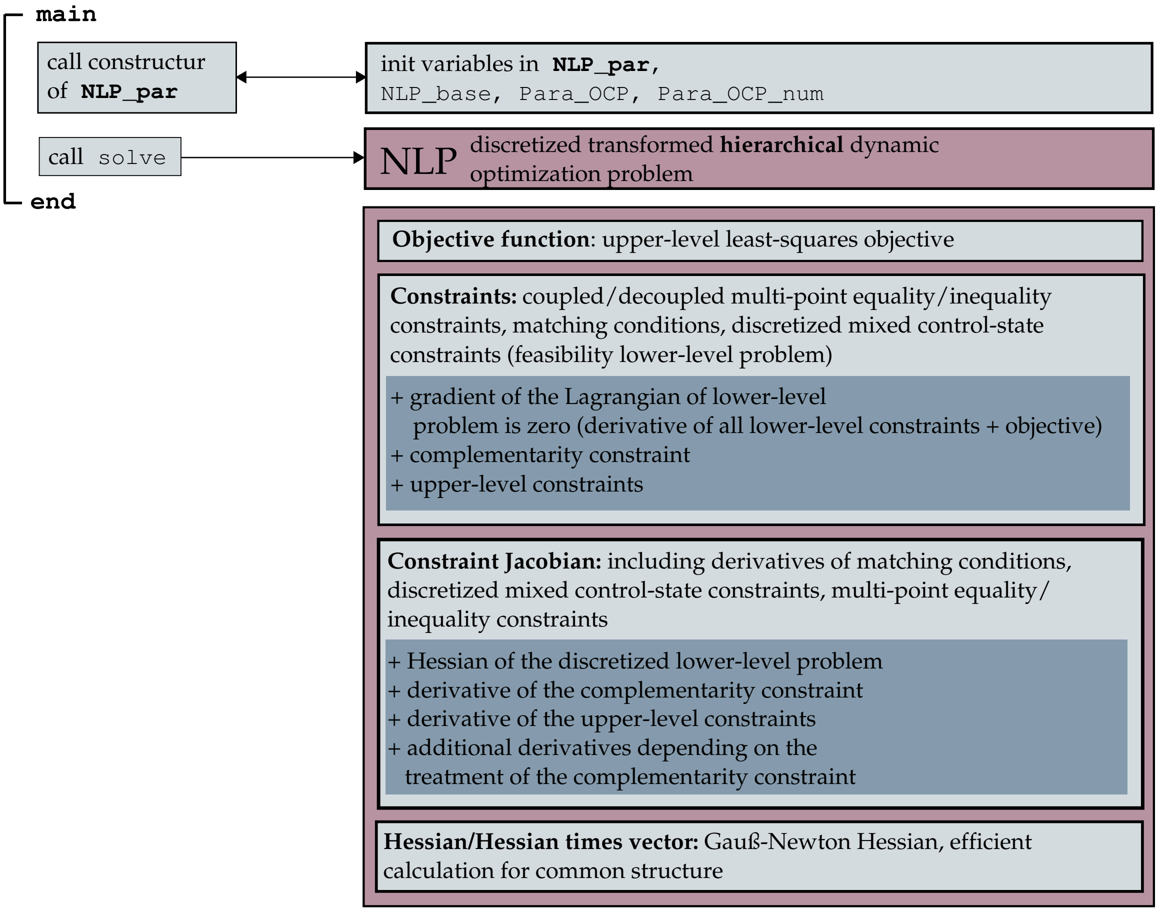

main
call constructur of **NLP_par**
init variables in **NLP_par**, NLP_base, Para_OCP, Para_OCP_num
call solve
NLP discretized transformed **hierarchical** dynamic optimization problem
end
Objective function: upper-level least-squares objective
Constraints: coupled/decoupled multi-point equality/inequality constraints, matching conditions, discretized mixed control-state constraints (feasibility lower-level problem)
+ gradient of the Lagrangian of lower-level problem is zero (derivative of all lower-level constraints + objective)
+ complementarity constraint
+ upper-level constraints
Constraint Jacobian: including derivatives of matching conditions, discretized mixed control-state constraints, multi-point equality/inequality constraints
+ Hessian of the discretized lower-level problem
+ derivative of the complementarity constraint
+ derivative of the upper-level constraints
+ additional derivatives depending on the treatment of the complementarity constraint
Hessian/Hessian times vector: Gauß-Newton Hessian, efficient calculation for common structure

Figure 9.3.: Illustration of the modules of ParaOCP for a hierarchical dynamic optimization problem.

Part III.

Modeling human locomotion in a new way

Chapter 10.

Modeling the dynamics of human locomotion based on multibody systems

This chapter is concerned with modeling the dynamics of human locomotion as constrained multibody systems. We start with concepts and definitions that build the basis of the multibody-system models derived in the remainder of this Chapter. In the second section, we explain the derivation of the equations of motion describing the dynamics of a human's skeleton. Furthermore, modeling gait dynamics also requires a model of the skeleton's ground contact, which is discussed in the third section of this chapter. Finally, possible extensions of constrained multibody-system models for the dynamics of human gaits are highlighted.

10.1. Basic concepts and definitions

In this chapter, we model the dynamics of human walking. Standard references and review articles in the field of human locomotion include [Sha05, Cra89, ABEvS93, vS99] and [GL18].

We now start with a definition of walking.

Definition 10.1.1. (Walking) *The term walking defines a form of locomotion, where at any point in time, at least one leg touches the ground.*

Human walking is a very complex process, which consists of repetitive cycles. One cycle includes two steps and the posture at the beginning and at the end of the cycle are approximately the same. The typical phases of a human's gait cycle are illustrated in Figure 10.1 (cf. [Ayy97]). In Figure 10.1, the gait cycle starts with the touchdown of the right foot. In the first phase of the cycle, both feet touch the ground. This phase is called *double support phase.* After the lift off of the left foot, the second phase starts, which is called *single support phase* of the right foot. At the same time, this is the *swing phase* of the left leg, whereas this part also belongs to the *stance phase* of the right leg. With the touchdown of the left foot, the second step and the second double support phase starts, followed by the single support phase of the left foot, which is also the swing phase of the right leg.

The human gait cycle is often assumed to be symmetric, which implies that is suffices to model just one step instead of the full gait cycle. However, this is not always the case, which we will see in Chapter 11. The gait cycle is sometimes divided in more than four phases, where the additional phases describe the time between heel strike and toe off. However, in this thesis we focus on the cycle illustrated in Figure 10.1 and adapt it to the specialties of certain types of gait (cf. Chapter 11).

For the description of the motion of a human's body, we now introduce the three anatomical planes and axes of the body: the coronal (or frontal) plane, the sagittal plane, and the

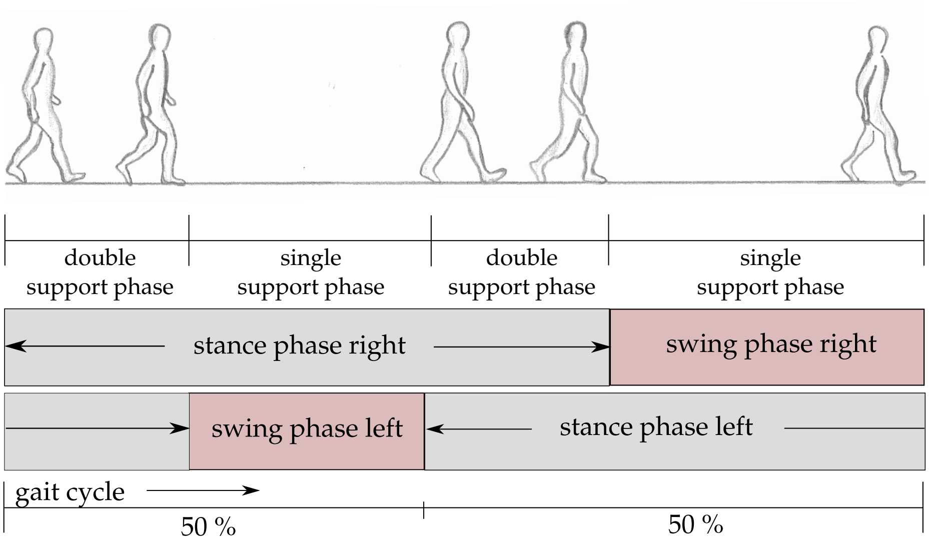


Figure 10.1.: Illustration of a human's gait cycle.

transverse plane. The axes include the longitudinal, the sagittal and the transverse axis. The three planes and axes are illustrated in Figure 10.2. Each of the three main planes split the human body into two parts: front (or anterior) and back (or posterior) part (coronal plane), left and right part (sagittal plane), and upper (or superior) and lower (or inferior) part (transverse plane). The longitudinal axis is the intersection of coronal and sagittal plane, the sagittal axis is the intersection of sagittal and transverse plane, and the transverse axis described the intersection of the coronal and the transverse plane.

We use the coordinate system shown in Figure 10.2 to describe the initial position of the body. The following posture of the human body is called *neutral zero joint position* and is used as reference to describe the motion of parts of the body:

- standing erect
- face forward
- arms at side, close to the body
- palms directed forward
- toes directed forward.

The neutral zero joint position is used in the next chapters as reference position for modeling human gait. The rotation around the main anatomical axes are briefly described in the following. Note that the terms *left* and *right* are always meant from the patient's/subject's perspective.

Flexion/Extension Flexion and extension are motions in the sagittal plane around the transverse axis. The bending of a joint around the transverse axis with an increasing angle to the coronal plane is called flexion, whereas the straightening motion that decreases the angle to the coronal plane is called extension. A typical exercise based on flexion and extension is a squat.

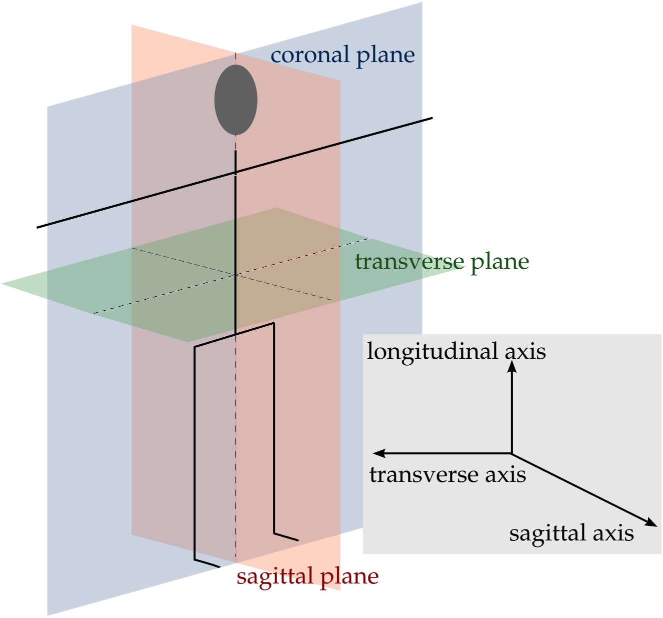


Figure 10.2.: Illustration of the coronal, transverse and sagittal plane, and the longitudinal, transverse and sagittal axis.

Abduction/Adduction Abduction and adduction happen in the coronal plane around the sagittal axis. Abduction is a motion that pulls a part of the body away from its midline in the coronal plane. Adduction is the inverse motion, which pulls a part of the body towards its midline. An example of abduction and adduction is lifting the arms sideways away from the body, and dropping them back to the sides.

Internal rotation/External rotation Internal and external rotation are motions in the transverse plane around the longitudinal axis. The internal rotation describes a motion towards the body, which means clockwise rotation of the right arm and a counterclockwise rotation of the left arm in a right-handed trihedron.

A human's musculoskeletal system consists of a passive part including the skeleton, the joints and the ligaments, and an active part, which includes the skeletal muscles (for details, see [GL18]). In this thesis, we focus on models of the human's musculoskeletal system that are based on a human's skeleton including joints (details can be found in the next section).

The skeleton consists of more than 200 bones (children have more than 270 bones) and is illustrated in Figure 10.5. The skeleton's axial part (along the longitudinal axis of the body) maintains the upright posture. The main function of the skeleton include the support of the body, the movement of the body via joints, the protection of vital organs, the blood-cell production in the bone marrow, the storage of calcium and the regulation of endocrine. The joints serve as contact point between two bones and allow the skeleton to move (cf. [GL18]).

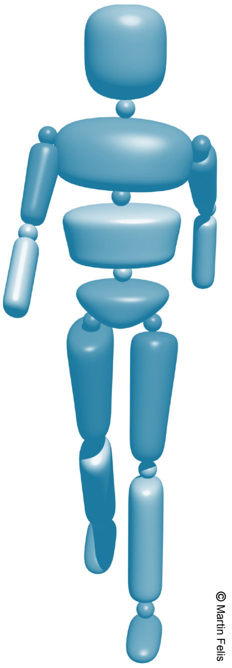


Figure 10.3.: Illustration of a typical segmentation of the human body for multibody-system models.

10.2. Multibody-system models

The body of an adult consists of more than 200 bones and 400 muscles, which makes it indispensable for a model to approximate a human's locomotor system, where the approximation of course depends on the specific application. In literature, numerous models of the locomotor system of a human can be founds reaching from very basic inverted pendulum models [Kuo07, GCRC98] to models that include the majority of the skeletal muscles [DAA$^+$07].

Following the work of [Fel09, Kos08, Sch07, Sim98, Mom01], we do not model the skeletal muscles and focus on the skeleton assuming that during walking, torques directly act at the joints. We start with a definition of a multibody system following [vS99].

Definition 10.2.1. (Multibody system) *A multibody system describes a mechanical system via single bodies/segments, whose relative interaction is constrained by joints, where a joint always connects two segments.*

We focus on multibody-system models for the whole body, which means that we model the human body as a composition of rigid segments that interact via joints. The analysis of muscle forces during walking is a possible extension that is beyond the scope of thesis. Two types of joints are illustrated in Figure 10.4: the revolute joint and the spherical/ball joint. The motion of, e.g., the elbow joint can be approximated by a revolute joint, and the motion of the hip joint can be approximating by a spherical joint.

A typical segmentation of the body is shown in Figure 10.3 with 14 rigid bodies connected via joints. In the following, we use the term *degrees of freedom*, which we define as follows.

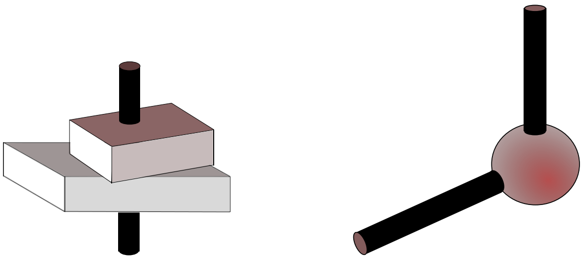


Figure 10.4.: Illustration of two joint types: a revolute joint (left) and a spherical joint (right).

Definition 10.2.2. (Degrees of freedom (DOFs)) *The degrees of freedom of a mechanical system are defined as the minimum number of coordinates required to completely describe the system's motion.*

A system of n segments that are not connected has $6n$ DOFs. Each segment has three DOFs for the position in the three-dimensional space, and three DOFs for the segment's orientation. This is not true anymore for a multibody system with segments that are connected via joints. In general, multibody systems can be separated into two classes: systems with kinematic loops and systems without kinematic loops (cf., e.g., [vS99]). Systems with kinematic loops include at least one circle, whereas systems without kinematic loops have a tree structure. In the latter case, for a fixed root segment, the relative positions of the segments are independent from each other and the relative description of the segments is unique. Both is not the case for systems with kinematic loops. The joints illustrated in Figure 10.4 have rotatory DOFs. A system of two segments coupled via a revolute joint with a fixed base has 1 DOFs instead of 6 DOFs without any coupling. If the two segments are coupled via a spherical joint and the base is fixed, the system has 3 DOFs.

In the derivation of a multibody-system model for the human body with the segmentation shown in Figure 10.3, the segment representing the pelvis is usually chosen as root segment with six DOFs, where the six DOFs describe the segment's position and orientation in the three-dimensional space. The global frame of reference of the root segment serves as basis for the position and the orientation of additional segments and can be described by a translation

$$\mathfrak{T}_{\text{root}} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{10.1}$$

and an orientation

$$\mathfrak{R}_{\text{root}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{10.2}$$

where the columns of $\mathfrak{R}_{\text{root}}$ are the coordinate axes of the global inertial system/frame. We refer to the global inertial frame by $(\mathfrak{T}_{\text{root}}, \mathfrak{R}_{\text{root}})$.

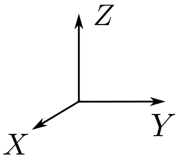
We introduce a frame $(\mathfrak{T}_{\text{i}}, \mathfrak{R}_{\text{i}})$ of reference for each additional segment i, which is defined relative to its parent frame. Like the root segment, each additional segment has its own frame of reference with three translational DOFs and three rotational DOFs. The child frame $(\mathfrak{T}_{\mathrm{j}}, \mathfrak{R}_{\mathrm{j}})$ can be expressed in the coordinate system of the parent frame $(\mathfrak{T}_{\mathrm{i}}, \mathfrak{R}_{\mathrm{i}})$ described by the matrix

$$\bar{\mathfrak{R}}_i := \left(\begin{array}{ccc|c} & \mathfrak{R}_{\mathrm{i}} & & \mathfrak{T}_{\mathrm{i}} \\ \hline 0 & 0 & 0 & 1 \end{array}\right) \tag{10.3}$$

by multiplying (10.3) with the following rotation and translation matrices (cf. [Cra89]):

$$\begin{aligned}
\bar{\mathfrak{R}}_x(\theta_1) &:= \left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & \cos\theta_1 & -\sin\theta_1 & 0 \\ 0 & \sin\theta_1 & \cos\theta_1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right), \\
\bar{\mathfrak{R}}_y(\theta_2) &:= \left(\begin{array}{ccc|c} \cos\theta_2 & 0 & \sin\theta_2 & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta_2 & 0 & \cos\theta_2 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right), \\
\bar{\mathfrak{R}}_z(\theta_3) &:= \left(\begin{array}{ccc|c} \cos\theta_3 & -\sin\theta_3 & 0 & 0 \\ \sin\theta_3 & \cos\theta_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right), \\
\bar{\mathfrak{T}}(r_x, r_y, r_z) &:= \left(\begin{array}{ccc|c} 1 & 0 & 0 & r_x \\ 0 & 1 & 0 & r_y \\ 0 & 0 & 1 & r_z \\ \hline 0 & 0 & 0 & 1 \end{array}\right),
\end{aligned} \tag{10.4}$$

where the first matrix in (10.4) is a rotation matrix of θ_1 around the x-axis, the second matrix performs a rotation θ_2 around the y-axis, and the third matrix describes a rotation θ_3 around the z-axis. The last matrix in (10.4) performs a translation of (r_x, r_y, r_z). The labels of the axes are as follows:



The angles $(\theta_1, \theta_2, \theta_3)$ can be referred to as Euler angles (cf. [Cra89]). Expressing the child frame in the coordinate system of the parent frame is done by a multiplication of $\bar{\mathfrak{R}}_i$ with $\bar{\mathfrak{R}}_x$, $\bar{\mathfrak{R}}_y$, $\bar{\mathfrak{R}}_z$ and $\bar{\mathfrak{T}}$, where the order of the multiplication is important.

There are 24 possible Euler angle conventions for mapping a frame of reference to another frame of reference: twelve possible mappings where after a rotation $\bar{\mathfrak{R}}_i$, the axes of the rotated system are used (denoted by $\bar{\mathfrak{R}}'_i$) for the consecutive rotation, and twelve possible mappings, where the original axes are used (denoted by $\bar{\mathfrak{R}}_i$). In this thesis, we use the $Y'X'Z'$-Euler angles with a subsequent translation:

$$\bar{\mathfrak{R}}'_y \bar{\mathfrak{R}}'_x \bar{\mathfrak{R}}'_z \bar{\mathfrak{T}}. \tag{10.5}$$

We furthermore have to determine coordinates to describe a certain point in a frame of reference. We choose generalized coordinates, which are the minimum number of independent coordinates that completely describe the configuration of the mechanical system.

In general, mechanical multibody-system models can be separated into two main classes: kinematic multibody-system models and multibody-system models that also include dynamic quantities. Kinematic models are concerned with geometric properties like position, orientation, or velocities, and do not consider any forces. Dynamic models also include forces like torques acting at the joints, masses, or inertia. In this thesis, we are interested in dynamic multibody-system models. The basis for a dynamic multibody-system model of the dynamics of human locomotion is Newton's second law [TM04]:

$$F = m \cdot a, \tag{10.6}$$

stating that the acceleration of a body is directly proportional the overall force acting on the body, where m is the mass of the body, F is the force, and a the body's acceleration.

We describe the dynamics of a constrained multibody system by the following system of ordinary differential equations (ODEs) (cf. [Sha05, Mom01]):

$$\mathfrak{M}(\mathfrak{q})\ddot{\mathfrak{q}} = \tau - \mathfrak{N}(\mathfrak{q}, \dot{\mathfrak{q}}) + \mathfrak{G}(\mathfrak{q})^{\intercal} \varkappa \tag{10.7}$$

$$g(\mathfrak{q}) = 0, \tag{10.8}$$

which is described in detail in the following. Let n_{DOF} denote the number of degrees of freedom of the multibody system (e.g. the system illustrated in Figure 10.3), which leads to n_{DOF} generalized coordinates

$$\begin{pmatrix} \mathfrak{q}_1(t) \\ \vdots \\ \mathfrak{q}_{n_{\mathrm{DOF}}}(t) \end{pmatrix}, \tag{10.9}$$

where the argument of $\mathfrak{q}$ is skipped in (10.7)-(10.8). We note that the coupling of the bodies or segments via joints is implicitly modeled by choosing generalized coordinates. No further constraints are necessary. The mass matrix is in (10.7)-(10.8) denoted by $\mathfrak{M}$ mapping from $\mathbb{R}^{n_{\mathrm{DOF}}}$ to $\mathbb{R}^{n_{\mathrm{DOF}}} \times \mathbb{R}^{n_{\mathrm{DOF}}}$. $\mathfrak{N}$ describes the generalized nonlinear effects like the Coriolis, centrifugal or gravitational force mapping from $\mathbb{R}^{2n_{\mathrm{DOF}}}$ to $\mathbb{R}^{n_{\mathrm{DOF}}}$. The generalized forces are denoted by τ. The contact of a segment to the ground can be formulated as constraint

$$g(\mathfrak{q}) = \begin{pmatrix} g_x(\mathfrak{q}) \\ g_y(\mathfrak{q}) \\ g_z(\mathfrak{q}) \end{pmatrix} = 0 \tag{10.10}$$

with its derivative $\frac{\partial}{\partial \mathfrak{q}} g(\mathfrak{q})$ denoted by $\mathfrak{G}(\mathfrak{q})$. The vector $\varkappa = (\varkappa_x, \varkappa_y, \varkappa_z)^{\intercal}$ describes the Cartesian forces acting on the contact point, where these forces are mapped to generalized coordinates by the matrix $\mathfrak{G}(\mathfrak{q})$. System (10.7)-(10.8) is a differential algebraic equation (DAE) of index 3, which is now transformed into a DAE with index 1:

$$\mathfrak{M}(\mathfrak{q})\ddot{\mathfrak{q}} = \tau - \mathfrak{N}(\mathfrak{q}, \dot{\mathfrak{q}}) + \mathfrak{G}(\mathfrak{q})^{\intercal} \varkappa \tag{10.11}$$

$$\mathfrak{G}(\mathfrak{q})\ddot{\mathfrak{q}} = -\zeta(\mathfrak{q}, \dot{\mathfrak{q}}) \tag{10.12}$$

or written as linear system of equations

$$\begin{pmatrix} \mathfrak{M} & \mathfrak{G}^{\intercal} \\ \mathfrak{G} & 0 \end{pmatrix} \begin{pmatrix} \ddot{\mathfrak{q}} \\ -\varkappa \end{pmatrix} = \begin{pmatrix} \tau - \mathfrak{N} \\ -\zeta \end{pmatrix}, \tag{10.13}$$

where equation (10.8) has been replaced by

$$\frac{d^2}{dt^2} g(\mathfrak{q}) = \mathfrak{G}(\mathfrak{q})\ddot{\mathfrak{q}} + \left(\left(\frac{\partial}{\partial \mathfrak{q}} \mathfrak{G}(\mathfrak{q}) \right) \dot{\mathfrak{q}} \right) \dot{\mathfrak{q}} = \mathfrak{G}(\mathfrak{q})\ddot{\mathfrak{q}} + \zeta(\mathfrak{q}, \dot{\mathfrak{q}}) = 0, \tag{10.14}$$

where the arguments of several functions are skipped for a compact presentation. Equation (10.14) requires the second derivative of the contact constraints $g(\mathfrak{q})$ with respect to t to be zero. If we additionally ensure that (10.10) and

$$\frac{d}{dt} g(\mathfrak{q}) = \mathfrak{G}(\mathfrak{q})\dot{\mathfrak{q}} = 0 \tag{10.15}$$

are satisfied at the beginning of the contact phase, the ODE (10.13) combined with the latter conditions describes the dynamics of the constrained multibody system. We note that the linear system of equations (10.13) has a unique solution if $\mathfrak{G}$ has full rank and $\mathfrak{M}$ is positive definite on the null space of the constraints $g(\mathfrak{q}) = 0$.

The previous paragraph derives an ODE describing the dynamics of a constrained multibody system. It remains to explain how the quantities $\mathfrak{M}, \mathfrak{N}, \mathfrak{G}$ and ζ can be computed. We therefore use the HuMAnS toolbox [INR05], which relies on the recursive Newton-Euler algorithm. The exact algorithm, alternatives like, e.g., the composite rigid body algorithm and the articulated body algorithm and their implementation can be found in [FO00, Fel12]. For more details about multibody-system dynamics we refer to [Fel09, MBLS01, Kos08, Sch07, Sim98, vS99, Cra89].

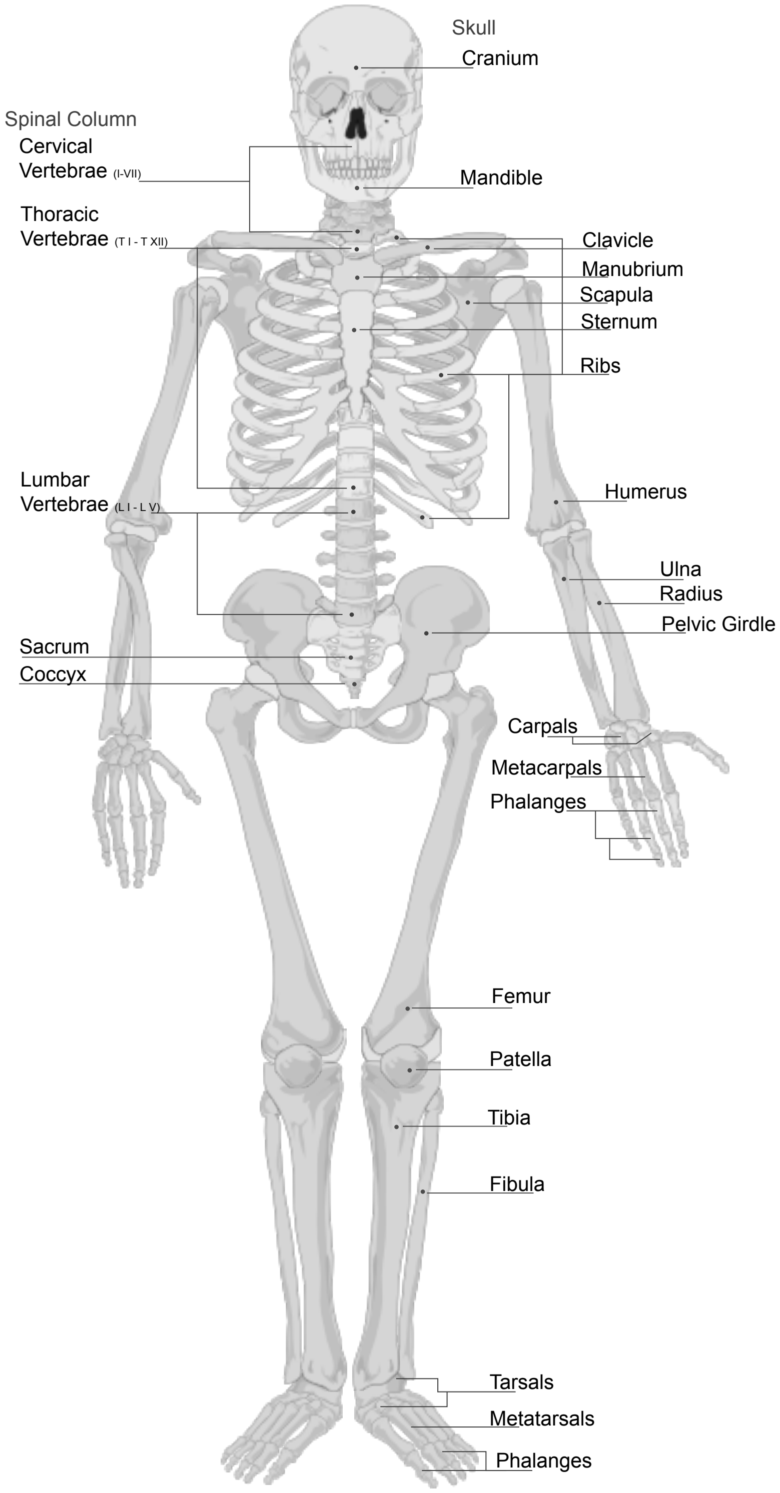


Figure 10.5.: Illustration of the human skeleton.

Chapter 11.

Cerebral palsy

Cerebral palsy describes a large group of disorders of movement and posture that are due to a damage or an abnormal development of the brain. The first section of this chapter analyzes the causes, the symptoms and the treatment of cerebral palsy in detail. In the second section, we discuss the gait of cerebral palsy patients and highlight special properties that are important for identifying a model for the gait of a cerebral palsy patient.

11.1. Introduction

An overview of causes, symptoms and the treatment of cerebral palsy (CP) can be found in, e.g., [Gag91, PS04, DMBG07, vdKDH09, Wik13] or [Död07]. This chapter is based on the latter references. CP stands for a wide range of disorders that are caused during birth or in early childhood in the majority of cases. CP can result from a variety of different mechanisms, which mostly lead to a damage or an abnormal development of the brain affecting muscle tone, posture, skeleton, movement and the sensory system. Often, such defects in the brain's development imply spastic muscles that are permanently contracting (e.g. the calf muscle), which leads to deformed joints and a deformed skeleton. The abnormal muscle tones combined with a deformed skeleton then cause the typical so-called *crouched* or *scissored* gait. In industrialized countries, the prevalence of CP is about two to three per 1000 live births [Död07]. Figure 11.1 shows pictures of a CP patient with Vicon motion capture marker attached to the patient's skin [VIC13] provided by the Heidelberg MotionLab [Wol93]. In Figure 11.1, the spasticity mainly affects the patient's left foot, which is in a typical club foot posture (*pes equinus*).

We now briefly introduce the Heidelberg MotionLab. The Heidelberg MotionLab has been founded in 1993 in the Department of Orthopedic Surgery of the Heidelberg University Clinic in Heidelberg, Germany, with the goal of improving treatment planning and treatment evaluation. The Heidelberg MotionLab mainly concentrates on gait analysis, the development of new gait analysis techniques and of new biomechanical models in order to improve the understanding of human motions/gaits. Furthermore, the laboratory is used for clinical trials for a better evaluation of the effect of surgeries and of conservative therapies, which also include the supply of ortheses and protheses. The development of an optimal control gait model for CP patients in this thesis is supported by the Heidelberg MotionLab [Wol93] by providing the pictures in Figure 11.1 and 11.3, motion capture measurement data of CP patients and of subjects with a healthy gait described in Section 15.1, as well as expertise in CP gait analysis. We now continue with the introduction to CP.

Typical causes of CP include prenatal, perinatal and postnatal mechanisms. Some examples are

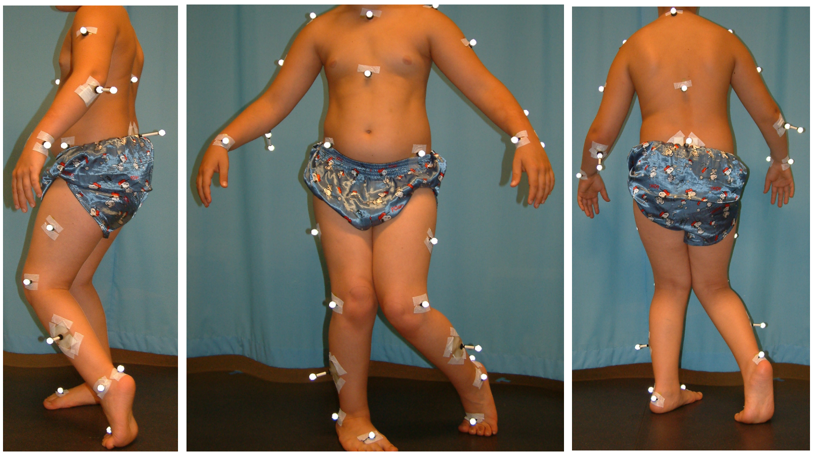


Figure 11.1.: Picture of the posture of a CP patient with Vicon motion capture marker, provided by the Heidelberg MotionLab [Wol93].

- oxygen deficiency during birth
- brain bleeding
- infectious diseases of the mother
- umbilical cord complications
- genetic causes (only around two percent of all CP patients)
- etc.

For an extensive discussion of mechanisms causing CP, we refer to [Död07, PS04]. CP can be divided into five main classes reflecting the areas of the brain that are damaged:

Spastic CP: Spastic CP is the most common type of CP (more than 70%) and the major impairment is due to tense and contracted muscles.

Ataxic CP: Ataxic CP is less common than spastic CP (5-10%) and can be caused by damage to the cerebellum. Typical symptoms include a poor sense of balancing, poor motor skills and difficulties with visual and auditory processing.

Athetoid CP: Athetoid CP is approximately as common as ataxic CP (around 10%) and leads to involuntary, uncontrolled motions of the limbs, the head and the eyes. Furthermore, patients with an athetoid CP often have trouble to hold themselves in a steady position.

Rigidity: Typical signs of CP leading to rigidity are very tight muscles that resist to move.

Tremor: Tremors describe involuntary muscle contractions and relaxations, and uncontrollable shaking of one or more parts of the body. The symptoms of CP patients with tremors are often worsen by stress.

Symptoms of all five classes of CP can also appear simultaneously, which makes the treatment even harder because of the extremely heterogeneous causes. Additional symptoms of CP include epilepsy (for around 50% of all patients), mental health problems or amyotrophia. Another way of classifying CP depends on the part of the body that is affected. We consider hemiplegia, paraplegia and quadriplegia, which are briefly explained in the following and illustrated in Figure 11.2.

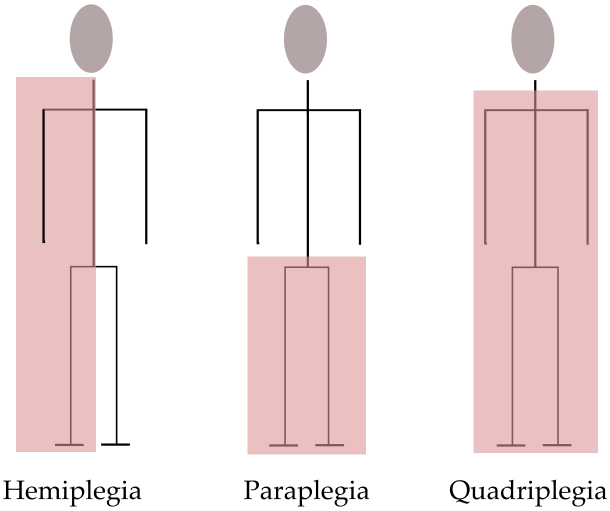


Figure 11.2.: Illustration of the parts of the body affected in case of spastic hemiplegia, paraplegia and quadriplegia.

Hemiplegia: Spastic hemiplegia describes a type of spasticity that only affects one part of the body being constantly under contraction (around 35% of all CP patients).

Paraplegia: Spastic paraplegia mainly affects the lower limbs of the body, whereas the upper part of the body is nearly unaffected or less affected than the legs (around 40% of all CP patient).

Quadriplegia or tetraplegia: Spastic quadriplegia describes a generalized spasticity that affects all parts of the body and is often combined with the loss of motor and sensory functions. Spastic quadriplegia is one of the strongest forms of CP.

In general, one can say that CP cannot be cured. The main treatments of CP patients can be separated into the class of conservative or interpersonal treatments, medication, surgeries and orthotics. These four main classes are briefly described in the following.

Conservative treatments: Conservative treatments for CP patients include physiotherapy, occupational therapy, speech therapy, conductive education (unified therapy based on the work of A. Pető), etc.

Medication: The muscle hypertonus of CP patients is in some cases reduced by antispasmodics like botulinum toxin, which has to be repeated after approximately three month. Furthermore, in case of mental health problems, psychotropic drugs are used.

Surgeries: Surgeries mostly intend to correct contractures and deformed parts of the skeleton or the joints in order to prevent pathological gaits. Typical examples are loosening tight muscles in the hip, the knees or the ankle, the transfer of muscles, derotation osteotomy (bones are broken and set in the correct alignment) to reduce abnormal twists and forces on the bones (e.g. femur and tibia), or rhizotomy, which means that nerves on the limbs that are most effected are cut.

Orthotics: Orthotic devices also play an important role in the treatment of CP patients to stabilize the joints and to prevent contractures.

The posture of CP patient before and after multiple surgeries (as, e.g., the rectus femoris transfer described in [DBW+12]) is shown in Figure 11.3.

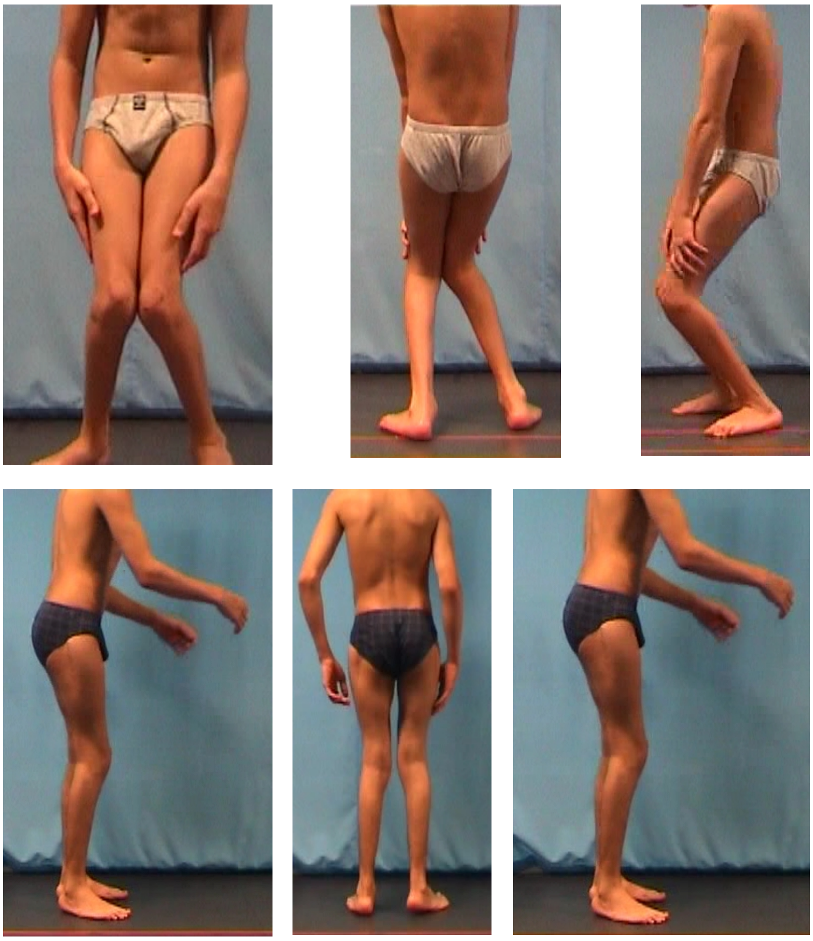


Figure 11.3.: Picture of the posture of a CP patient before (first row) and after (second row) multiple surgeries, provided by the Heidelberg MotionLab [Wol93].

In the remainder of this section, we state open questions in the current focus of research in the field of CP (cf. [AD05, ALS+06, vdKDH09, DMBG07, DGW+12, DBV+13]). In this thesis, our goal is to show how numerical techniques for solving hierarchical dynamic optimization problems can be used as a first step in answering these open questions. In the following summary of open questions in the field of CP gait analysis, we also briefly discuss the contribution of the techniques derived in this thesis to the process of answering these questions.

General understanding of CP gaits The general understanding of the motion of CP patients is a topic in current research. The main goal is to understand the exact cause of certain motion patterns of patients. Furthermore, researchers are concerned with the analysis of how, e.g., specific spastic muscles affect a patient's gait. Moreover, one part of CP research is to understand healthy and pathological gaits in general, which means to get an idea of why humans move the way they move with limited or unlimited mobility. Many works analyze very specific parts of a patient's body as, e.g., the hamstring length (cf. [vdKDH09]). However, there are only very few works following a holistic approach, which considers the whole body at once (cf. [AD05]). Our goal in this thesis is to improve the general, holistic understanding of human motions and of healthy and pathological gaits.

Categories/classification of CP gaits Gait classification of healthy gaits and especially of CP gaits is an important topic in current research (cf. [DMBG07]). The task is to develop classification schemes of CP gaits that can be used in clinical decision-making. However, it is very difficult to find one clinically significant classification scheme for most of the CP gaits because of the extremely high diversity of CP gaits. Our goal is to assist in the development of classification schemes by providing a new type of gait models presented in Chapters 12 and 15.

Criteria to evaluate the success of treatments One problem in the treatment of CP patients is the high diversity of symptoms and types of CP. As described above, there is a large variety of possible treatments including physiotherapy, occupational therapy, speech therapy, conductive education, medication (e.g. based on antispasmodics), surgeries (loosening tight muscles, muscle transfer, derotation osteotomy, etc.) and orthotics. For clinical decision-making, it is very important to evaluate the success of previous treatments. Therefore, one needs to derive criteria that imply whether, e.g., the gait of a CP patient has improved after a specific treatment. In literature, there are mainly trials investigating one specific aspect and one specific treatment of CP (cf. [FRGS00]). Therefore, our goal is to use the gait models derived in this thesis as basis for the derivation of holistic criteria for the evaluation of CP treatments.

Model-based treatment planning The following quote about treatment planning for CP patients in taken from Arnold and Delp in [AD05]:

> *The treatment of gait abnormalities in persons with CP is challenging. Theoretically, gait abnormalities can be diminished by decreasing the muscle forces that disrupt normal movement ... and/or increasing the muscle and ground reaction forces that have the potential to improve movement ... . However, different patients exhibit varying degrees of neurologic impairment, spasticity, weakness, and bone deformity, suggesting that gait deviations arise from a variety of sources, each of which requires a different treatment. Treatment planning is further complicated because there is cur-*

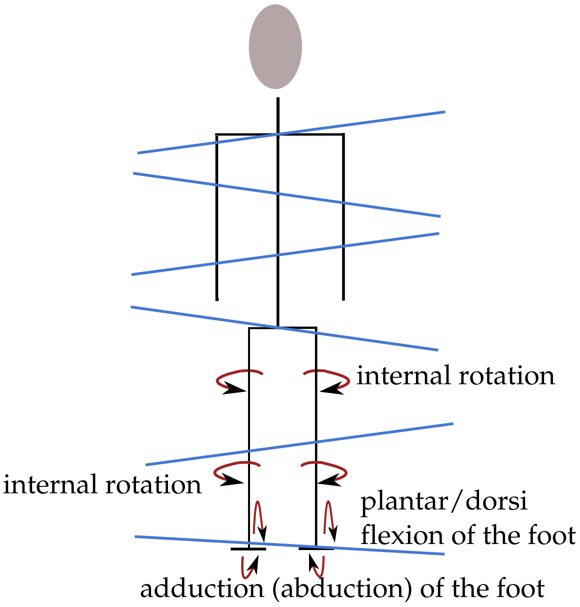


Figure 11.4.: Illustration of possible differences of the skeleton of a CP patients.

> *rently no scientific basis for determining how patients' neuromusculoskeletal impairments contribute to abnormal movement.*

As written in [AD05], the main problem in treatment planning for CP patients is to derive a theoretical basis that is able to capture the main characteristics of CP gaits and of possible treatments. Our goal in this thesis is to provide a gait model as main first step in the derivation of such a theoretical basis. The idea is to use our gait model (formulated as an optimal control problem) as non-invasive test environment for treatment planning. Having a model that does not just describe the kinematics (as in [AD05]) or the dynamics of a patient's skeleton but the gait itself would allow to get an idea of how one specific surgery, e.g., would affect the patient's gait. This idea is discussed in detail in Chapter 15.

To sum up, the basis of our contribution to the process of answering the four questions described above is a new type of gait model in the form of an optimal control problem. This novel type of model might lead to new insights into the characteristic of healthy and pathological gaits by allowing to access information that cannot be accessed otherwise.

In the next Chapter, we derive gait models for CP patients and for able-bodied humans. Unknown parameters in these models are identified in Chapter 15, followed by a discussion and an interpretation of the results. The last section of this chapter in concerned with the characteristics of CP gaits that have to be taken into account when modeling the gait's dynamics. Furthermore, the next section highlights the differences between healthy and pathological gaits in the corresponding multibody-system model.

11.2. Characteristics of cerebral palsy gaits

In this section, we briefly discuss the characteristics of CP gaits and compare it to healthy gaits, especially in consideration of the multibody-system models that are derived in the next chapter. We start with a discussion of differences in the skeleton with a focus on the lower limbs, followed by an analysis of the foot contact.

Figure 11.4 illustrates some possible differences in the skeleton of a CP patient. In this section, we always have in mind that we intend to derive a multibody-system model of the patients skeleton. Hence, we do not consider all possible differences of the body of a CP patient, we focus on some simplified characteristics of the patient's skeleton that can be captured by a multibody-system model. As shown in Figure 11.4, the transverse axis of a patient's body is rotated around the sagittal axis at various points along the midline of the body (which is illustrated by the rotated horizontal lines in Figure 11.4). This, e.g., reflects a deformed spine.

In general, one can say that the neutral zero joint position explained in Chapter 10 is different for CP patients. Apart from the rotation of the transverse axis, CP patients often have internally rotated thighs and shanks. The gait of able-bodied humans is often modeled in only two dimensions – the transverse axis is neglected since the motion is mainly performed in the sagittal plane. This is not true anymore for the gait of CP patients. In this case, we have a truly three-dimensional gait with the main rotation in the hip joint. The motion of the legs are primarily executed in a plane corresponding to the rotated sagittal plane, where the rotation is around the longitudinal axis. A typical posture of the lower body of a CP patient is illustrated in the first row of Figure 11.5 and compared to a human with a healthy gait (second row of 11.5). CP patients often have so-called *pes equinus*, with an adduction of both feet (a rotation around the sagittal axis such the soles point towards the midline of the body), where often, only the forefoot is in touch with the ground. The patients often also have a dorsi flexion (towards the shank) or a plantar flexion (away from the shank) of the feet. The rotation of several parts of the skeleton, as well as the posture and the ground contact of the feet are taken into account in the derivation of a gait model for CP patients in the next Chapter.

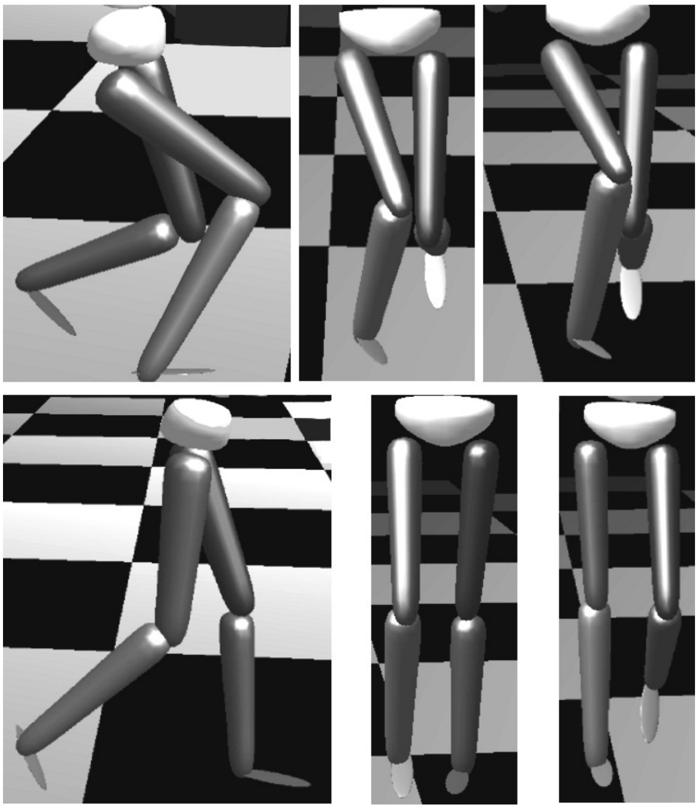


Figure 11.5.: Illustration of the posture of the lower limbs of a CP patient (first row) and a human with a healthy gait (second row) based on a multibody-system model.

Chapter 12.

Optimal control models for human locomotion

This chapter is concerned with the derivation of a new type of model for human locomotion– a model, which itself is an optimal control problem. In this chapter, we derive two gait models: a cerebral palsy gait model and a healthy gait model. For both models, the underlying dynamics are described by the equations of motions for a multibody system that approximates the respective skeleton. We start with a general formulation of a gait model in the first section. In the two remaining sections, a cerebral palsy gait model and a healthy gait model are derived based on the first section's general formulation.

12.1. General formulation

The general idea of modeling processes in nature that are assumed to be optimal in a certain sense by optimal control problems is introduced in Chapter 4. In this chapter, we transfer this approach to human gait. This means, we follow the basic idea of the field of bionics and assume that human motions have been optimized in the course of evolution with respect to certain criteria (cf. [BPS06, FMKB13, MS10]).

We consider optimal control gait models of the following general form:

$$\begin{array}{llll} \underset{x,u,q}{\text{minimize}} & \multicolumn{3}{l}{\displaystyle\sum_{k=0}^{n_M-1}\left(\sum_{i=1}^{n_{\mathcal{M}}}\gamma_i\phi_{\mathcal{M}}^{ki}(x_k(t_{k+1}),q,p)+\sum_{i=n_{\mathcal{M}}+1}^{n_{\mathcal{M}}+n_{\mathcal{L}}}\gamma_i\int_{t_k}^{t_{k+1}}\phi_{\mathcal{L}}^{ki}(x_k(t),u_k(t),q,p)\,\mathrm{dt}\right)} \\ \text{subject to} & \dot{x}_k(t) & = f_k(x_k(t),u_k(t),q,p), & t\in[t_k,t_{k+1}], \quad k\in S_{n_S} \\ & x_{k+1}(t_{k+1}) & = b(x_k(t_{k+1}),q,p), & \qquad\qquad\qquad\quad k\in S_{n_S-1} \\ & 0 & \le c_k(x_k(t),u_k(t),q,p), & t\in[t_k,t_{k+1}], \quad k\in S_{n_S} \\ & 0 & = r_k^{\mathrm{eq}}(x_k(\tau_{k,0}),\dots,x_k(\tau_{k,n_k}),q,p), & \qquad\qquad\qquad\quad k\in S_{n_S} \\ & 0 & \le r_k^{\mathrm{ieq}}(x_k(\tau_{k,0}),\dots,x_k(\tau_{k,n_k}),q,p), & \qquad\qquad\qquad\quad k\in S_{n_S} \end{array} \tag{12.1}$$

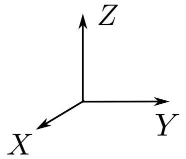
for given $p \in \mathbb{R}^{n_p}$ and $\gamma \in \mathbb{R}^{n_{\mathcal{M}}+n_{\mathcal{L}}}$, with $S_{n_S} := \{0,\dots,n_S-1\}$ and $S_{n_S-1} := \{0,\dots,n_S-2\}$, $x := (x_0,\dots,x_{n_S-1})$ and $u := (u_0,\dots,u_{n_S-1})$. The variables p and γ are unknown and are identified from real-world measurements in Chapter 15. The remaining functions and variables in (12.1) are defined at the beginning of Chapter 4. In the following, the functions in problem (12.1) are explained in detail for optimal control gait models, where the underlying dynamics are described by the equations of motions for a multibody system approximating a human's skeleton. Furthermore, the control functions in (12.1) model torques acting at the skeleton's joints.

12.1.1. Differential states, control functions and control values

In Section 10.1, we describe the dynamics of multibody-system models using generalized coordinates. In problem (12.1), the differential states are denoted by x_k with $k \in S_{n_S}$, which we define as $x_k := (\mathfrak{q}_k^\mathsf{T}, \dot{\mathfrak{q}}_k^\mathsf{T})^\mathsf{T} \in \mathbb{R}^{2n_{\mathrm{DOF}}}$, where $\mathfrak{q}_k$ describes the generalized coordinates on model stage k with the first three entries $\mathfrak{q}_{k,0}, \mathfrak{q}_{k,1}, \mathfrak{q}_{k,2}$ denoting the global position of the basis segment, and $\mathfrak{q}_{k,3}, \mathfrak{q}_{k,4}, \mathfrak{q}_{k,5}$ denoting the orientation of the basis segment. Furthermore, the summarized degrees of freedom (DOFs) of all model stages are given by $\mathfrak{q} := (\mathfrak{q}_0, \ldots, \mathfrak{q}_{n_S-1})$. We note that in this thesis, we solely use

$$Y'X'Z' \quad \text{Euler angles (cf. [Cra89])} \tag{12.2}$$

with the following axes:



The remaining entries in $\mathfrak{q}_k$ describe the rotational DOFs of each joint in the local frame as described in Section 10.1. The variable $\dot{\mathfrak{q}}_k$ is of the same dimension as $\mathfrak{q}_k$ containing the corresponding velocities: three global translational velocities, three global rotational velocities, and the remaining rotational velocities in each segment's frame. The controls in (12.1) describe the torques acting at the joints. We assume to have one control for each local DOF in a joint, which means that we have $n_{\mathrm{DOF}} - 6$ controls. Furthermore, we have n_S model stage in (12.1). Some of the model stages have a free end time, which is modeled using the control values q (cf., e.g., Chapter 6).

12.1.2. Model stages

In Section 10.1, we describe the human gait cycle and its division into phases: two double support phases, a single support phase right and a single support phase left. On average, the double support phase is approximately 15% of the whole gait cycle and for some gaits, it is even less. For many cerebral palsy (CP) gaits and also for some healthy gaits, we observe that the double support phase is extremely short and thus, we only consider three single support phases and their transition in this thesis: a single support phase of the right foot, a single support phase of the left foot, and again a single support phase of the right foot. At the end of a single support phase, the respective other foot touches the ground. This transition between two single support phases has to be modeled carefully since we have to deal with a jump in the differential states. In particular, we have a jump in the local joint velocities, which is due to the forces of the ground acting on the foot that just entered the contact phase and is modeled as a momentum acting on this foot. In (12.1), this jump in the differential states can be formulated using the transition conditions

$$x_{k+1}(t_{k+1}) \; = \; b(x_k(t_{k+1}), q, p), \quad k \in S_{n_S-1}. \tag{12.3}$$

In detail, the transition conditions between two single support stages with, e.g., model stage index k and $k+1$ is given by

$$\begin{aligned} \mathfrak{q}_{k+1}(t_{k+1}) \; &= \; \mathfrak{q}_k(t_{k+1}) \\ \dot{\mathfrak{q}}_{k+1}(t_{k+1}) \; &= \; \dot{\mathfrak{q}}_{k,+}(t_{k+1}), \end{aligned} \tag{12.4}$$

where $\dot{\mathfrak{q}}_{k,+}$ describes the velocities after the discontinuity and solves the following system of equations:

$$\begin{pmatrix} \mathfrak{M}(\mathfrak{q}_k(t_{k+1})) & \mathfrak{G}^{\mathsf{T}}_{\text{onefoot}}(\mathfrak{q}_k(t_{k+1})) \\ \mathfrak{G}_{\text{onefoot}}(\mathfrak{q}_k(t_{k+1})) & 0 \end{pmatrix} \begin{pmatrix} \dot{\mathfrak{q}}_{k,+} \\ \Lambda_{\text{onefoot}} \end{pmatrix} = \begin{pmatrix} \mathfrak{M}(\mathfrak{q}_k(t_{k+1}))\dot{\mathfrak{q}}_k(t_{k+1}) \\ 0 \end{pmatrix} \tag{12.5}$$

with $\mathfrak{M}$ being the mass matrix and $\mathfrak{G}_{\text{onefoot}}$ the constraint Jacobian of the foot that enters the contact phase defined in Section 10.1. The momentum acting on the foot entering the contact phase is denoted by Λ_{onefoot}. This means, the momentum Λ_{onefoot} in (12.5) is determined such that the Cartesian velocities in the contact point of the foot entering the contact phase are zero: $\mathfrak{G}_{\text{onefoot}}\dot{\mathfrak{q}}_{k,+} = 0$ (where Cartesian refers to the global frame). In order to have a consistent framework in the implementation ParaOCP of the direct all-at-once approach for hierarchical dynamic optimization problems described in Chapter 9, we implement a discontinuous transition between two model stages as an extra model stage with length zero, where the corresponding right-hand side executes the jump in the differential states. This means in total, we have three single support phases (right, left, right) and two transition stages as illustrated in Figure 12.1, with the time grid

$$t_0 < t_1 = t_2 < t_3 = t_4 < t_5. \tag{12.6}$$

This means the first single support stage with model stage index $k = 0$ is on the interval $[t_0, t_1]$, at $t_1 = t_2$ we have a transition stage with index $k = 1$, the second single support stage is on $[t_2, t_3]$ with index $k = 2$ followed by a transition stage at $t_3 = t_4$ with index $k = 3$, and the last single support stage is on $[t_4, t_5]$ with model stage index $k = 4$. In (12.1), this leads to $n_S = 5$, $S_{n_S} = \{0, 1, 2, 3, 4\}$ and $S_{n_S-1} = \{0, 1, 2, 3\}$. The duration of the single support phases is not fixed – each of the three single support phases has a free end time, which means that $t_1 = t_2, t_3 = t_4$ and t_5 are variable. In practice, however, we do a time transformation of $[t_0, t_1]$ on the fix horizon $[0, 1]$, of $[t_1, t_2]$ on the fix horizon $[1, 2]$ and of $[t_2, t_3]$ on the fix horizon $[2, 3]$, and the duration of the three single support phases is described by the control values q_0, q_1 and q_2, respectively. The end of a single support phase is reached when, e.g., the respective other foot touches the ground (cf. Subsection 12.1.4). The two transition stages have the fixed length zero. As discussed in, e.g., [HSB12], this leads to a linear transformation of the ordinary differential equation's (ODE's) right-hand side as stated below, and the solution of the new problem formulation on the intervals $[0, 3]$ coincides with the solution of the old formulation on $[t_0, t_5]$. We finally note that the control functions in the transition stage are set to the value of the control functions of the subsequent model stage at the transition time: $u_1(1) = u_2(1)$ and $u_3(2) = u_4(2)$.

12.1.3. The dynamics

For each model stage, we have a different right-hand side of the ODE in (12.1). We note that for a compact presentation, problem (12.1) only includes ODEs and no differential algebraic equations (DAEs), which are used to model the dynamics of multibody systems as explained in Chapter 10. In an algorithmic context, one could either directly treat the DAE or resolve the DAE (10.13) with index 1 into an ODE by explicitly solving the linear system of equations. We now define the functions $f_k(\cdot)$ in (12.1) for $k = 0, 1, 2, 3, 4$.

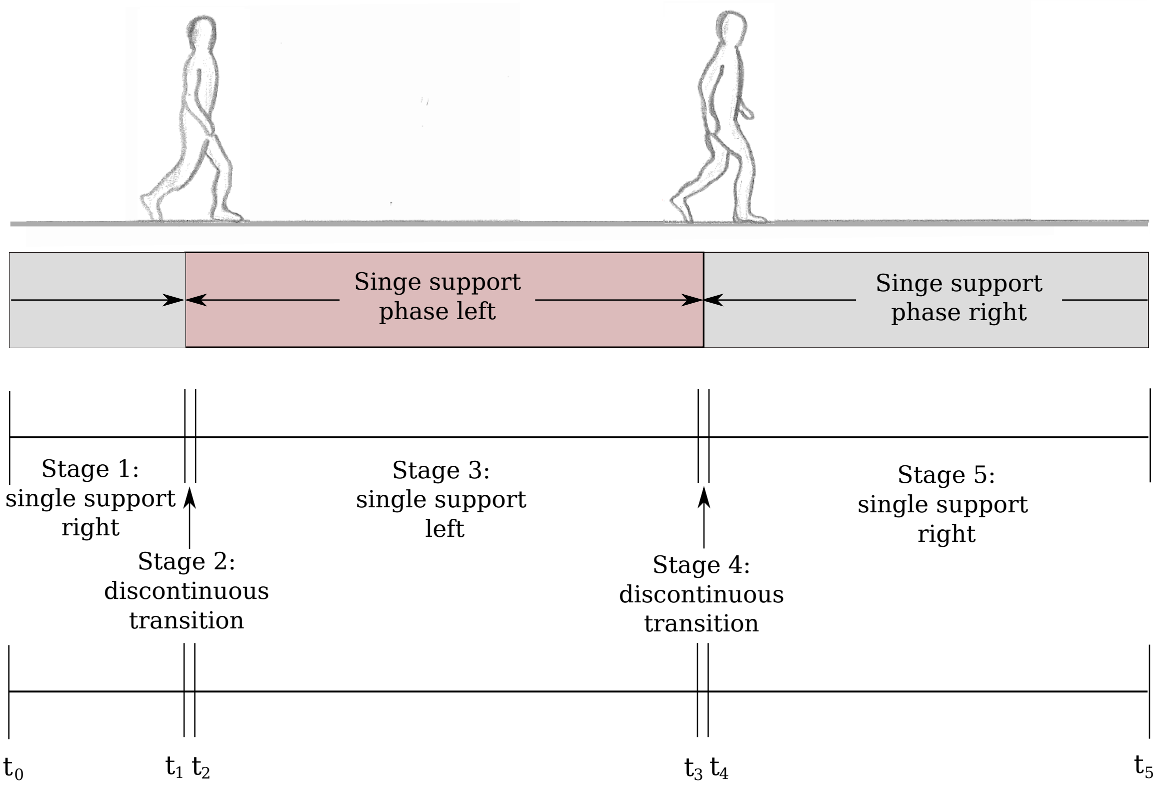


Figure 12.1.: Illustration of the model stages in optimal control gait models for human locomotion.

First single support phase right

Based on the multibody-system dynamics explained in Chapter 10, we have the following ODE in the first model stage:

$$\dot{x}_0(t) = f_0(x_0(t), u_0(t), q, p) = q_0 \overset{\cdot}{\begin{pmatrix} \mathfrak{q}_0(t) \\ \dot{\mathfrak{q}}_0(t) \end{pmatrix}} = q_0 \begin{pmatrix} \dot{\mathfrak{q}}_0(t) \\ \ddot{\mathfrak{q}}_0(t) \end{pmatrix}, \quad t \in [0,1], \tag{12.7}$$

where $\ddot{\mathfrak{q}}_0(t)$ is part of the solution of

$$\begin{pmatrix} \mathfrak{M}(\mathfrak{q}_0(t)) & \mathfrak{G}^{\mathsf{T}}_{\text{right}}(\mathfrak{q}_0(t)) \\ \mathfrak{G}_{\text{right}}(\mathfrak{q}_0(t)) & 0 \end{pmatrix} \begin{pmatrix} \ddot{\mathfrak{q}}_0(t) \\ -\varkappa \end{pmatrix} = \begin{pmatrix} u_0 - \mathfrak{N}(\mathfrak{q}_0(t), \dot{\mathfrak{q}}_0(t)) \\ -\zeta_{\text{right}}(\mathfrak{q}_0(t), \dot{\mathfrak{q}}_0(t)) \end{pmatrix}. \tag{12.8}$$

In (12.8), $\mathfrak{M}$ denotes the mass matrix, the contact Jacobian for the right foot is given by $\mathfrak{G}_{\text{right}}$, the Cartesian forces at the contact point are described by $\varkappa$, $\mathfrak{N}$ are the nonlinear effects, and ζ_{right} denotes the contact Hessian for the respective contact point (cf. Section 10.1). The torques in (12.8) are described by the controls from (12.1). The quantities $\mathfrak{M}, \mathfrak{G}_{\text{left}}$ (the contact Jacobian of the left foot), $\mathfrak{G}_{\text{right}}, \mathfrak{N}$, ζ_{left} (the contact Hessian of the left foot) and ζ_{right} within this chapter are computed using the HuMAnS toolbox [INR05] based on kinematic data like the definition of the exact frame of each segment, and dynamic data including the human's body weight and hight, gravitation or the inertia of each segment.

Furthermore, the dynamic data consists of the following quantities for each segment: the radius of gyration, the center of gravity and the mass. These quantities are then used to compute the inertia based on the Huygens theorem [INR05, TM04]. The exact parameters that enter the computation are provided in Section 12.2 and 12.3 for the respective model. We finally note that the multiplication in the right-hand side of (12.8) with q_0 is due to the time transformation on the fix horizon $[0, 1]$.

Transition phase left

For the second model stage, which is the transition phase with length 0, we denote

$$x_1(1) = \begin{pmatrix} \mathfrak{q}_1(1) \\ \dot{\mathfrak{q}}_1(1) \end{pmatrix}, \; x_2(t) = q_1 \begin{pmatrix} \mathfrak{q}_2(t) \\ \dot{\mathfrak{q}}_2(t) \end{pmatrix}, \; t \in [1, 2], \tag{12.9}$$

and execute the following jump conditions

$$\begin{array}{rclcrcl} \mathfrak{q}_1(1) & = & \mathfrak{q}_0(1), & \qquad & \mathfrak{q}_2(1) & = & \mathfrak{q}_1(1), \\ \dot{\mathfrak{q}}_1(1) & = & \dot{\mathfrak{q}}_{0,+}(1), & & \dot{\mathfrak{q}}_2(1) & = & \dot{\mathfrak{q}}_1(1), \end{array} \tag{12.10}$$

where $\dot{\mathfrak{q}}_{0,+}(1)$ describes the velocities after the discontinuity and solves the following system of equations:

$$\begin{pmatrix} \mathfrak{M}(\mathfrak{q}_0(1)) & \mathfrak{G}_{\text{left}}^{\mathsf{T}}(\mathfrak{q}_0(1)) \\ \mathfrak{G}_{\text{left}}(\mathfrak{q}_0(1)) & 0 \end{pmatrix} \begin{pmatrix} \dot{\mathfrak{q}}_{0,+}(1) \\ \Lambda_{\text{left}} \end{pmatrix} = \begin{pmatrix} \mathfrak{M}(\mathfrak{q}_0(1))\dot{\mathfrak{q}}_0(1) \\ 0 \end{pmatrix} \tag{12.11}$$

with the constraint Jacobian $\mathfrak{G}_{\text{left}}$ and the Cartesian momentum Λ_{left} for the left foot, which enters the contact phase.

Single support phase left

In the third stage with single support of the left foot we have

$$\dot{x}_2(t) = f_2(x_2(t), u_2(t), q, p) = q_1 \dot{\begin{pmatrix} \mathfrak{q}_2(t) \\ \dot{\mathfrak{q}}_2(t) \end{pmatrix}} = q_1 \begin{pmatrix} \dot{\mathfrak{q}}_2(t) \\ \ddot{\mathfrak{q}}_2(t) \end{pmatrix}, \quad t \in [1, 2], \tag{12.12}$$

where $\ddot{\mathfrak{q}}_2(t)$ is determined by

$$\begin{pmatrix} \mathfrak{M}(\mathfrak{q}_2(t)) & \mathfrak{G}_{\text{left}}^{\mathsf{T}}(\mathfrak{q}_2(t)) \\ \mathfrak{G}_{\text{left}}(\mathfrak{q}_2(t)) & 0 \end{pmatrix} \begin{pmatrix} \ddot{\mathfrak{q}}_2(t) \\ -\varkappa \end{pmatrix} = \begin{pmatrix} u_2 - \mathfrak{N}(\mathfrak{q}_2(t), \dot{\mathfrak{q}}_2(t)) \\ -\zeta_{\text{left}}(\mathfrak{q}_2(t), \dot{\mathfrak{q}}_2(t)) \end{pmatrix}, \quad t \in [1, 2]. \tag{12.13}$$

Transition phase right

After the single support phase of the left foot, we again have a transition stage, now of the following form:

$$x_3(2) = \begin{pmatrix} \mathfrak{q}_3(2) \\ \dot{\mathfrak{q}}_3(2) \end{pmatrix}, \; x_4(t) = q_2 \begin{pmatrix} \mathfrak{q}_4(t) \\ \dot{\mathfrak{q}}_4(t) \end{pmatrix}, \; t \in [2, 3], \tag{12.14}$$

and we execute the following jump conditions

$$\begin{array}{rclcrcl} \mathfrak{q}_3(2) & = & \mathfrak{q}_2(2), & \qquad & \mathfrak{q}_4(2) & = & \mathfrak{q}_3(2), \\ \dot{\mathfrak{q}}_3(2) & = & \dot{\mathfrak{q}}_{2,+}(2), & & \dot{\mathfrak{q}}_4(2) & = & \dot{\mathfrak{q}}_3(2), \end{array} \tag{12.15}$$

where $\dot{\mathfrak{q}}_{2,+}(2)$ describes the velocities after the discontinuity and solves the following system of equations:

$$\begin{pmatrix} \mathfrak{M}(\mathfrak{q}_2(2)) & \mathfrak{G}^{\mathsf{T}}_{\text{right}}(\mathfrak{q}_2(2)) \\ \mathfrak{G}_{\text{right}}(\mathfrak{q}_2(2)) & 0 \end{pmatrix} \begin{pmatrix} \dot{\mathfrak{q}}_{2,+}(2) \\ \Lambda_{\text{right}} \end{pmatrix} = \begin{pmatrix} \mathfrak{M}(\mathfrak{q}_2(2))\dot{\mathfrak{q}}_2(2) \\ 0 \end{pmatrix}. \tag{12.16}$$

Second single support phase right

The equations of motion for the last single support phase of the right foot are given by

$$\dot{x}_4(t) = f_4(x_4(t), u_4(t), q, p) = q_2 \dot{\begin{pmatrix} \mathfrak{q}_4(t) \\ \dot{\mathfrak{q}}_4(t) \end{pmatrix}} = q_2 \begin{pmatrix} \dot{\mathfrak{q}}_4(t) \\ \ddot{\mathfrak{q}}_4(t) \end{pmatrix}, \quad t \in [2, 3], \tag{12.17}$$

where $\ddot{\mathfrak{q}}_4(t)$ is part of the solution of

$$\begin{pmatrix} \mathfrak{M}(\mathfrak{q}_4(t)) & \mathfrak{G}^{\mathsf{T}}_{\text{right}}(\mathfrak{q}_4(t)) \\ \mathfrak{G}_{\text{right}}(\mathfrak{q}_4(t)) & 0 \end{pmatrix} \begin{pmatrix} \ddot{\mathfrak{q}}_4(t) \\ -\varkappa \end{pmatrix} = \begin{pmatrix} u_4 - \mathfrak{N}(\mathfrak{q}_4(t), \dot{\mathfrak{q}}_4(t)) \\ -\zeta_{\text{right}}(\mathfrak{q}_4(t), \dot{\mathfrak{q}}_4(t)) \end{pmatrix}, \ t \in [2, 3]. \tag{12.18}$$

12.1.4. Constraints and stage transition conditions

We now explicitly formulate mixed control-state constraints and multi-point boundary constraints

$$\begin{array}{lll} 0 & \leq c_k(x_k(t), u_k(t), q, p), & t \in [t_k, t_{k+1}], \quad k \in S_{n_S} \\ 0 & = r^{\text{eq}}_k(x_k(\tau_{k,0}), \ldots, x_k(\tau_{k,n_k}), q, p), & \qquad\qquad\qquad\ \ k \in S_{n_S} \\ 0 & \leq r^{\text{ieq}}_k(x_k(\tau_{k,0}), \ldots, x_k(\tau_{k,n_k}), q, p), & \qquad\qquad\qquad\ \ k \in S_{n_S} \end{array} \tag{12.19}$$

from (12.1) for our optimal control gait model.

First single support phase right

As discussed in Section 10.2, we need to require that the right foot touches the ground at $t = 0$ and that the Cartesian velocities at the contact point are zero at $t = 0$ if the equations of motion for the multibody-system dynamics are formulated as differential algebraic equation with index 1. Therefore, we require

$$\mathfrak{G}_{\text{right}}(\mathfrak{q}_0(0))\dot{\mathfrak{q}}_0(0) = 0. \tag{12.20}$$

In the first stage, the single support phase of the right foot, we furthermore require that the left foot does not penetrate the ground:

$$[T_{\text{left}}(\mathfrak{q}_0(t))]_z \geq 0, \quad \forall t \in [0, 1), \tag{12.21}$$

where $[T_{\text{left}}(\mathfrak{q}_0(t))]_z$ describes the $z-$component of the position of the contact point of the left foot a time t in Cartesian coordinates of the global frame. In practice, this is realized via so-called *tags* in the HuMAnS toolbox [INR05]. Tags are characteristic points in the model for which HuMAnS provides the position in Cartesian coordinates in the global frame.

Transition phase left

The conditions for the first stage transition are required at the end of the single support phase of the right foot. As explained above, the single support phase of the right foot is of variable length and ends as soon as the transition conditions (or switching conditions) are satisfied. The first condition is given by

$$[T_{\text{left}}(\mathfrak{q}_0(1))]_z = 0, \tag{12.22}$$

where $[T_{\text{left}}(\mathfrak{q}_0(t)]_z$ describes the trajectory of the $z-$component of the contact point of the right foot in Cartesian coordinates of the global frame. Condition (12.22) requires the contact point of the right foot to touch the ground. The second condition wants the z-component of the Cartesian velocities at the contact point of the left foot to point into the direction of the ground, which is denoted by

$$\left[\mathfrak{G}_{\text{left}}(\mathfrak{q}_0(1))\dot{\mathfrak{q}}_0(1)\right]_z \leq 0. \tag{12.23}$$

Single support phase left

During the single support of the left foot, we require that the contact point of the right foot does not penetrate the ground:

$$\left[T_{\text{right}}(\mathfrak{q}_2(t))\right]_z \geq 0, \quad \forall t \in [1, 2). \tag{12.24}$$

Transition phase right

The switching condition between the single support phase of the left and the right foot are given by

$$[T_{\text{right}}(\mathfrak{q}_2(2))]_z = 0 \tag{12.25}$$

and

$$\left[\mathfrak{G}_{\text{right}}(\mathfrak{q}_2(2))\dot{\mathfrak{q}}_2(2)\right]_z \leq 0, \tag{12.26}$$

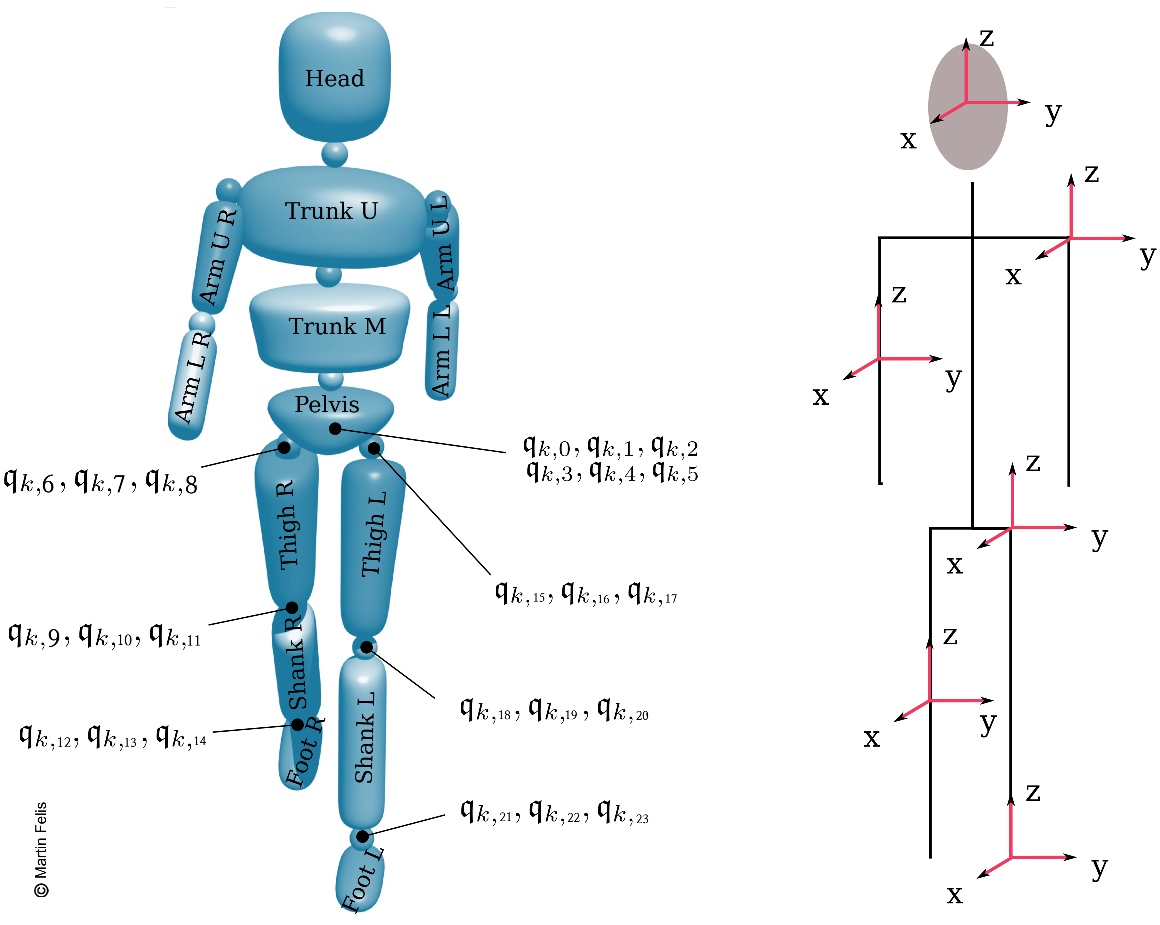
analogously to the switching condition of the transition of the left foot requiring that the right foot enters the contact phase and the z-component of the Cartesian velocities at the contact point of the right foot points into the direction of the ground.

Second single support phase right

In the second single support phase of the right foot, we simply require that the left foot does not penetrate the ground:

$$\left[T_{\text{left}}(\mathfrak{q}_4(t))\right]_z \geq 0, \quad \forall t \in [2, 3]. \tag{12.27}$$

All remaining constraints needed in the optimal control gait model (like, e.g., simple bounds on the variables) are stated for the respective model in the next two sections. The objective function for (12.1) is also specified in the following two sections, which explain the details of the CP gait model and the healthy gait model based on the general model formulation derived in this section.


Head
Trunk U
Arm U R
Arm U L
Trunk M
Arm L R
Arm L L
Pelvis
Thigh R
Thigh L
Shank R
Shank L
Foot R
Foot L
$\mathfrak{q}_{k,0}, \mathfrak{q}_{k,1}, \mathfrak{q}_{k,2}$
$\mathfrak{q}_{k,3}, \mathfrak{q}_{k,4}, \mathfrak{q}_{k,5}$
$\mathfrak{q}_{k,6}, \mathfrak{q}_{k,7}, \mathfrak{q}_{k,8}$
$\mathfrak{q}_{k,9}, \mathfrak{q}_{k,10}, \mathfrak{q}_{k,11}$
$\mathfrak{q}_{k,12}, \mathfrak{q}_{k,13}, \mathfrak{q}_{k,14}$
$\mathfrak{q}_{k,15}, \mathfrak{q}_{k,16}, \mathfrak{q}_{k,17}$
$\mathfrak{q}_{k,18}, \mathfrak{q}_{k,19}, \mathfrak{q}_{k,20}$
$\mathfrak{q}_{k,21}, \mathfrak{q}_{k,22}, \mathfrak{q}_{k,23}$
© Martin Felis
z
y
x

(a)

(b)

Figure 12.2.: Illustration of the segments of the gait model's underlying multibody system in (a) and a human's posture, where the global and all local frames have the same orientation in (b).

12.2. The cerebral palsy gait model

We now describe the details of our optimal control CP gait model based on the model stages, the dynamics and the constraints described in the last section.

The multibody system

We start with the segmentation of the CP gait model's underlying multibody system, which is illustrated in Figure 12.2. The basis segment is the pelvis, where we have six global DOFs– three variables $\mathfrak{q}_{k,0}, \mathfrak{q}_{k,1}, \mathfrak{q}_{k,2}$ for the position of the segment in the global frame, and three variables $\mathfrak{q}_{k,3}, \mathfrak{q}_{k,4}, \mathfrak{q}_{k,5}$ for the orientation of the basis segment, again in the global frame. We furthermore have three DOFs $\mathfrak{q}_{k,6}, \mathfrak{q}_{k,7}, \mathfrak{q}_{k,8}$ in the right hip joint and three DOFs $\mathfrak{q}_{k,15}, \mathfrak{q}_{k,16}, \mathfrak{q}_{k,17}$ in the left hip joint. Due to the truly three-dimensional gait of CP patients (cf. Section 11.2) and the analysis of motion capture data of CP patients, it is

Frame	**Parent**	**Rot Y**	**Rot X**	**Rot Z**	**Trans X**	**Trans Y**	**Trans Z**
Pelvis	–	$\mathfrak{q}_{k,0}$	$\mathfrak{q}_{k,1}$	$\mathfrak{q}_{k,2}$	$\mathfrak{q}_{k,3}$	$\mathfrak{q}_{k,4}$	$\mathfrak{q}_{k,5}$
Thigh R	Pelvis	$\mathfrak{q}_{k,6}$	$\mathfrak{q}_{k,7}$	$\mathfrak{q}_{k,8}$	0.	-0.0680	0.
Shank R	Thigh R	$\mathfrak{q}_{k,9}$	$\mathfrak{q}_{k,10}$	$\mathfrak{q}_{k,11}$	0.	0.	-0.3545
Foot R	Shank R	$\mathfrak{q}_{k,12} + o_1$	$\mathfrak{q}_{k,13} + o_2$	$\mathfrak{q}_{k,14} + o_3$	0.	0.	-0.3990
Thigh L	Pelvis	$\mathfrak{q}_{k,15}$	$\mathfrak{q}_{k,16}$	$\mathfrak{q}_{k,17}$	0.	0.0680	0.
Shank L	Thigh L	$\mathfrak{q}_{k,18}$	$\mathfrak{q}_{k,19}$	$\mathfrak{q}_{k,20}$	0.	0.	-0.3616
Foot L	Shank L	$\mathfrak{q}_{k,21} + o_4$	$\mathfrak{q}_{k,22} + o_5$	$\mathfrak{q}_{k,23} + o_6$	0.	0.	-0.3792
Trunk M	Pelvis	0.1002	-0.0500	0.	0.	0.	0.1350
Trunk U	Trunk M	0.1700	0.	0.	0.	0.	0.
Arm U R	Trunk U	0.	0.	0.	0.	-0.0680	0.1700
Arm L R	Arm U R	0.	0.	0.	0.	0.	-0.1700
Arm U L	Trunk U	0.	0.	0.	0.	0.0680	0.1700
Arm L L	Arm U L	0.	0.	0.	0.	0.	-0.1700
Head	Trunk U	0.1500	0.	0.	0.	0.	0.

Table 12.1.: Overview of the multibody-system frames and their relation in rad (column three, four and five) and in m (column six, seven and eight) with 24 DOFs and 14 segments based on measurement data of a CP patient from the Heidelberg MotionLab, see Section 15.1), and the the Plug-In-Gait model in the Vicon system [VIC13] rounded to four decimals.

important to also have three DOFs in the knee joint: $\mathfrak{q}_{k,9}, \mathfrak{q}_{k,10}, \mathfrak{q}_{k,11}$ in the right knee joint and $\mathfrak{q}_{k,18}, \mathfrak{q}_{k,19}, \mathfrak{q}_{k,20}$ in the left knee joint. The same is true for the ankle joint, the three DOFs in the right ankle joint are denoted by $\mathfrak{q}_{k,12}, \mathfrak{q}_{k,13}, \mathfrak{q}_{k,14}$ and in the left ankle joint by $\mathfrak{q}_{k,21}, \mathfrak{q}_{k,22}, \mathfrak{q}_{k,23}$.

We note that our multibody-system model has point feet but an ankle joint, which is rather unusual in literature. We choose this model in order to be able to capture the typical club foot (pes equinus) gait of CP patients. For alternative foot models, we refer to, e.g., [MFS12]. In this thesis, we concentrate on the DOFs in the lower body and fix the joints in the upper body in an average position. One reason for focusing on the DOFs in the lower body is related to the motion capture measurement data of CP patients and subjects obtained from the Heidelberg MotionLab (described in detail in Chapter 15), which does not include data of the arm and head motion. However, we note that we do not use a point mass for the upper body, we indeed model the upper body with the segments illustrated in Figure 12.2 and the inertia of each segment is taken into account, we just keep the upper body joints fixed.

An overview of the kinematic data of the CP gait model's underlying multibody system is provided in Table 12.1. The first column of Table 12.1 describes the segment whose frame we are considering and the second column names the corresponding parent frame, where the frame of the pelvis is the global root frame. The labels of the frames are explained in Figure 12.2(a). Column three, four and five in Table 12.1 state the rotation of the respective parent frame to obtain the child frame in $Y'X'Z'$-Euler angles. The translation in X, Y and Z direction for the transformation of the parent frame to the child frame is given in

Offsets	Value	Description
o_1	0.1011	right thigh rotation + offset1
o_2	-1.5133	-1.57 rad foot rotation + right static plantar flexion - offset2
o_3	-0.5932	right tibial torsion + right shank rotation + right static rotation
o_4	-0.1843	left thigh rotation + offset3
o_5	-1.5463	-1.57 rad foot rotation + left static plantar flexion + offset4
o_6	0.5666	- left tibial torsion - left shank rotation - left static rotation

Table 12.2.: Description of the offset angles in Table 12.1.

column six, seven and eight. The data in Table 12.1 is taken from measurements of the Heidelberg MotionLab [Wol93] of a CP patient using a Vicon motion capture system and the corresponding Plug-In-Gait model [VIC13]. Details about the measurements can be found in Section 15.1.

Table 12.1 also contains the 24 DOFs $\mathfrak{q}_{k,0}, \ldots, \mathfrak{q}_{k,23}$ of the multibody system illustrated in Figure 12.2(a) and offset angles $o_1, \ldots, o_6$, which describe the patient's neutral zero joint position explained in Section 10.1 based on a human's posture, where the global and all local frames have the same orientation as illustrated in Figure 12.2(b). The values and description for the offsets $o_1, \ldots, o_6$ are given in Table 12.2, where the angles in the description are provided by the Plug-In-Gait model of the Vicon system from the Heidelberg MotionLab based on measurements of the patient, and are given in Table 12.3. Table 12.2 furthermore contains the quantities offset1, offset2, offset3 and offset4. These offsets are necessary for the multibody-system model to be able to reproduce the motion capture data obtained from the Heidelberg MotionLab. However, these offsets are not explained by the Vicon Plug-In-Gait model. Table 12.3 also contains the patient's body mass and hight, which are used for the derivation of the multibody-system's equation of motion in HuMAnS [INR05].

We note that all data provided by the Plug-In-Gait model of the Vicon system including the angles shown in Table 12.3 follow different sign conventions based on whether the rotation is flexion or extension, an abduction or an adduction, or an internal or external rotation (defined in Section 10.1). Figure 12.3 shows some of the sign conventions for rotations in the lower body, where, e.g., the signs of a rotation depend on the foot. Another example are flexions, which always have a positive sign, even though a flexion in the hip joint is in the opposite direction of a flexion in the knee joint. External rotations have a negative sign, and internal rotations have a positive sign. For the global orientation of the pelvis and the local rotation of the feet, there are additional, more complex rules. The values in Table 12.3 are based on the Plug-In-Gait model conventions shown in Figure 12.3(a), and the rotations in the remaining tables in this section are based on standard conventions in physics explained in Section 10.1. For more details about sign conventions in Vicon motion capture data, we refer to the Plug-In-Gait user manual, [VIC13] or [Ret12].

We are interested in a multibody-system model that includes dynamic quantities like forces. Hence, we need additional knowledge like the inertia of each segment, which is estimated based on the radius of gyration, the length, the mass and the location of the center of gravity of each segment. The radius of gyration is taken from [dL96] and the HUMAN36

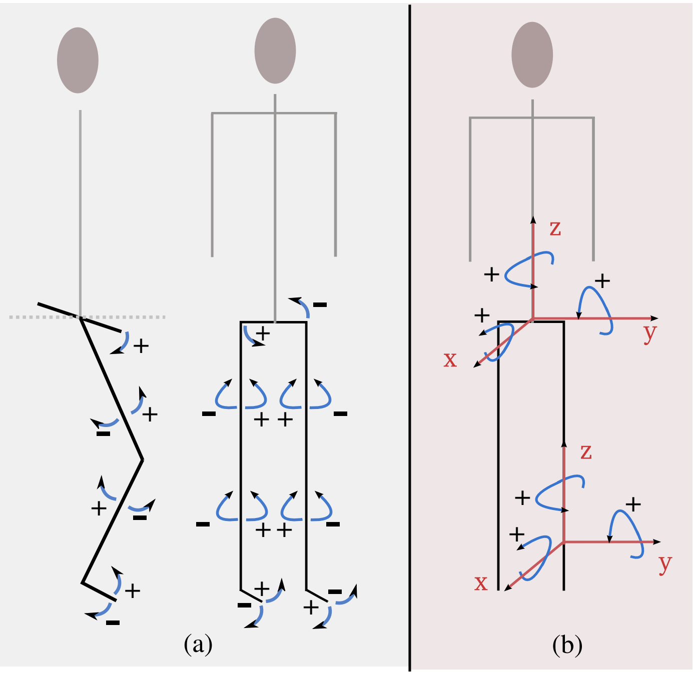


Figure 12.3.: Illustration of some of the sign conventions of rotations of frames used in the Plug-In-Gait model of Vicon [VIC13] in (a) and sign conventions in a multibody-system model as described in Section 10.2 in (b).

model in HuMAnS, and the exact values are shown in Table 12.4. For the location of the center of gravity (COG), we use data from [dL96] and adapt it if necessary. The exact values are given in Table 12.5 in % of the segment length. The values we use to approximate the length and the mass of each segment are shown in Table 12.6. The segment lengths are computed from motion capture data of a CP patient from the Heidelberg MotionLab and are given in m. The motion capture data we use is described in more detail in Section 15.1. Values for the segment mass are mainly taken from [dL96] and adapted if necessary, and are provided in % of the total body mass.

Bounds and additional constraints

We now state simple bounds on the variables $x_k = (\mathfrak{q}_k, \dot{\mathfrak{q}}_k), u_k$ and q with $k \in S_{n_S}$ used in (12.1) starting with the bound on $x_k \; \forall k \in S_{n_S}$. For all differential states, we initially choose very loose bounds, since in the parameter identification described in Chapter 15, we intend to track measurement data of the patient's gait, which typically avoids that the rotation in the joints and the corresponding velocities exceed natural bounds. The same is true for the simple bounds on the torques/controls u. However, after identifying the unknowns p and γ in (12.1) as described in Chapter 15, these bounds can be tightened based on the joint limits in the measurement data in order to use the optimal control gait model for treatment planning. We now state the simple bounds used in (12.1) for the parameter estimation in

Quantity	**Value**	
body mass	39.2	kg
body height	1.46	m
right thigh rotation	0.0861	rad
offset1	0.015	rad
right static plantar flexion	0.1775	rad
offset2	0.12	rad
right tibial torsion	-0.5236	rad
right shank rotation	-0.0953	rad
right static rotation offset	0.0257	rad
left thigh rotation	0.1257	rad
offset3	-0.31	rad
left static plantar flexion	0.0845	rad
offset4	-0.06	rad
left tibial torsion	-0.3490	rad
left shank rotation	-0.2844	rad
left static rotation offset	0.0669	rad

Table 12.3.: Measurements and calculated values based on data of a CP patient from the Heidelberg MotionLab, see Section 15.1, and the Plug-In-Gait model in the Vicon system [VIC13] rounded to four decimals.

Chapter 15:

$$\begin{array}{lllllll} -1000 & \leq & \mathfrak{q}_{k,i}(t) & \leq & 1000 & i=\{0,1,2\}, & k \in S_{n_S}, \quad \forall t \in \mathfrak{I}_k \\ -1.5 & \leq & \mathfrak{q}_{k,i}(t) & \leq & 1.5 & i=\{3,\dots,23\}, & k \in S_{n_S}, \quad \forall t \in \mathfrak{I}_k \\ -1000 & \leq & \dot{\mathfrak{q}}_{k,i}(t) & \leq & 1000 & i=\{0,\dots,23\}, & k \in S_{n_S}, \quad \forall t \in \mathfrak{I}_k \\ -500 & \leq & u_i(t) & \leq & 500 & i=\{0,\dots,17\}, & k \in S_{n_S}, \quad \forall t \in \mathfrak{I}_k, \end{array} \tag{12.28}$$

with $\mathfrak{I}_0 = [0,1]$, $\mathfrak{I}_1 = [1,1]$, $\mathfrak{I}_2 = [1,2]\,, \mathfrak{I}_3 = [2,2]$, and $\mathfrak{I}_4 = [2,3]$. For the three control values describing the lengths of the single support phases in seconds, we choose

$$\begin{array}{lllll} 0 & \leq & q_0 & \leq & 10 \\ 0 & \leq & q_1 & \leq & 10 \\ 0 & \leq & q_2 & \leq & 10. \end{array} \tag{12.29}$$

In the parameter estimation process in Chapter 15, we furthermore require that the joints in the initial position $\mathfrak{q}_{k,i}(0)$ for $i = \{0,\dots,23\}, k \in S_{n_S}$ are fixed to the values obtained from the motion capture measurements. At the end of the gait cycle ($t = 3$), we require the orientation and the position of the pelvis, and the contact point of the left foot $T_{\text{left}}(\mathfrak{q}_4(3))$ is equal to the measured values. This initial and terminal constraint are removed after the parameter estimation process and replaced by constraints, which, e.g., require the origin of the root segment to be in a box centered around a certain value at the beginning ($t = 0$) and at the end ($t = 3$) of the gait cycle.

Radius of gyration **(% of the segment length)**	**X**	**Y**	**Z**
Pelvis	0.615	0.551	0.587
Thigh R	0.329	0.329	0.149
Shank R	0.251	0.246	0.102
Foot R	0.124	0.245	0.257
Thigh L	0.329	0.329	0.149
Shank L	0.251	0.246	0.102
Foot L	0.124	0.245	0.257
Trunk M	0.482	0.383	0.468
Trunk U	0.505	0.320	0.465
Arm U R	0.285	0.269	0.158
Arm L R	0.276	0.265	0.121
Arm U L	0.285	0.269	0.158
Arm L L	0.276	0.265	0.121
Head	0.303	0.261	0.315

Table 12.4.: Radius of gyration in % of the segment length based on [INR05] (HUMAN36 model) and [dL96].

The objective function

We finally describe the components $\phi_{\mathcal{M}}^{ki}$ and $\phi_{\mathcal{L}}^{kj}$ for $k \in S_{n_S}, i \in \{1, \dots, n_{\mathcal{M}}\}$ and $j \in \{n_{\mathcal{M}} + 1, \dots, n_{\mathcal{M}} + n_{\mathcal{L}}\}$ of the objective function of (12.1):

$$\sum_{k=0}^{n_M - 1} \left(\sum_{i=1}^{n_{\mathcal{M}}} \gamma_i \phi_{\mathcal{M}}^{ki}(x_k(t_{k+1}), q, p) + \sum_{i=n_{\mathcal{M}}+1}^{n_{\mathcal{M}}+n_{\mathcal{L}}} \gamma_i \int_{t_k}^{t_{k+1}} \phi_{\mathcal{L}}^{ki}(x_k(t), u_k(t), q, p) \, \mathrm{dt} \right). \tag{12.30}$$

Discussions with experts from the Heidelberg Motionlab led to four types of criteria resulting in $n_{\mathcal{M}} = 0$ and $n_{\mathcal{L}} = 6$, which are defined in the following.

Stability: The first criterion is a stability-maximization-type criterion, which we define for the first model stage (single support right) as

$$\phi_{\mathcal{L}}^{0,1}(x_0(t), u_0(t), q, p) := \int_0^1 \Big([T_{\text{hip\_right}}(\mathfrak{q}_0(t))]_y - [T_{\text{right}}(\mathfrak{q}_0(t))]_y \Big)^2 \mathrm{dt}, \tag{12.31}$$

where $T_{\text{hip\_right}}$ describes the origin of the frame located in the right hip joint with the label Thigh R in Figure 12.2 in global Cartesian coordinates of the root frame labeled Pelvis. In practice, $T_{\text{hip\_right}}$ is obtained via a tag in HuMAnS like T_{left}, which is explained above. Criterion (12.31) minimizes the distance between the y-component of the right hip joint and the contact point of the right foot in the global coordinates of the root frame. Common stability criteria are based on the COG or the so-called *Zero Moment Point (ZMP)*, as described in, e.g., [Wie02, PH05] and used in, e.g., [DDW$^+$08]. For more details about stability optimization of open-loop controlled locomotion, we refer to [Mom01]. Criterion (12.31) is a basic variant of the common criterion requiring that the ZMP/COG has to

Location center of gravity (% of the segment length)	**X**	**Y**	**Z**
Pelvis	0.	0.	0.6115
Thigh R	0.	0.	-0.4095
Shank R	0.	0.	-0.4365
Foot R	0.	0.	-0.4
Thigh L	0.	0.	-0.4095
Shank L	0.	0.	-0.4395
Foot L	0.	0.	-0.4
Trunk M	0.	0.	0.4502
Trunk U	0.	0.	0.5066
Arm U R	0.	0.	-0.5772
Arm L R	0.	0.	-0.4574
Arm U L	0.	0.	-0.5772
Arm L L	0.	0.	-0.4574
Head	0.	0.	0.5002

Table 12.5.: Location of the center of gravity (COG) of each segment in % of the segment length (taken from [dL96] and partly adapted).

strictly lie in the convex hull of the contact points, adapted to point feet. In the single support phase of the left foot, we minimizes the distance between the y-component of the left hip joint and the contact point of the left foot in the global coordinates of the root frame:

$$\phi_{\mathcal{L}}^{2,2}(x_2(t),u_2(t),q,p) := \int_1^2 \Big([T_{\text{hip\_left}}(\mathfrak{q}_2(t))]_y - [T_{\text{left}}(\mathfrak{q}_2(t))]_y\Big)^2 \mathrm{dt}, \tag{12.32}$$

where $T_{\text{hip\_left}}$ describes the origin of the frame located in the left hip joint with the label Thigh L in Figure 12.2 in global Cartesian coordinates of the root frame labeled Pelvis. In the last single support phase, the following criterion is minimized:

$$\phi_{\mathcal{L}}^{4,3}(x_4(t),u_4(t),q,p) := \int_2^3 \Big([T_{\text{hip\_right}}(\mathfrak{q}_4(t))]_y - [T_{\text{right}}(\mathfrak{q}_4(t))]_y\Big)^2 \mathrm{dt}. \tag{12.33}$$

Energy consumption: The second type of criterion minimizes the overall energy consumption by minimizing the squared and summed up integrals over each torque on each single support stage:

$$\phi_{\mathcal{L}}^{k,4}(x_k(t),u_k(t),q,p) := \sum_{j=0}^{n_u-1} \int_{t_k}^{t_{k+1}} (u_{k,j}(t))^2 \, \mathrm{dt}, \quad k = \{0,2,4\}, \tag{12.34}$$

which is an interesting criterion since in many areas of research, one assumes that humans minimize the total energy consumption when they are moving (see, e.g., [Ale97, BC95]).

Abduction/adduction in the hip joints: The third type of criterion can be interpreted as a convenience-maximization criterion, which minimizes the squared and summed

	Segment length (m)	**Segment mass (% of the body mass)**
Pelvis	0.135	0.1117
Thigh R	0.3545	0.1279
Shank R	0.399	0.0433
Foot R	0.1405	0.0137
Thigh L	0.3616	0.1279
Shank L	0.3792	0.0433
Foot L	0.1175	0.0137
Trunk M	0.17	0.1633
Trunk U	0.17	0.1596
Arm U R	0.17	0.0271
Arm L R	0.17	0.0162
Arm U L	0.17	0.0271
Arm L L	0.17	0.0162
Head	0.15	0.0694

Table 12.6.: Length and mass of each segment in m and % of the total body mass, respectively, based on measurement data of a CP patient from the Heidelberg MotionLab, see Section 15.1, own estimates and [dL96].

up integrals over the torques corresponding to abduction/adduction in both hip joints for each single support stage:

$$\phi_{\mathcal{L}}^{k,5}(x_k(t), u_k(t), q, p) := \sum_{j=1,10} \int_{t_k}^{t_{k+1}} (u_{k,j}(t))^2 \, \mathrm{dt}, \quad k = \{0, 2, 4\}. \tag{12.35}$$

As discussed in, e.g., [SAH75], CP patients often have an adduction deformity, which makes a rotation in the hip joint around the sagittal axis impossible or very painful. Hence, the minimization of the squared and summed up integrals over the torques corresponding to abduction/adduction in both hip joints can be interpreted as a convenience criterion.

Rotation in the hip joints: The fourth type of criterion is related to the latter one, and minimizes the squared and summed up integrals over the torques corresponding to internal and external rotation in both hip joints for each single support stage:

$$\phi_{\mathcal{L}}^{k,6}(x_k(t), u_k(t), q, p) := \sum_{j=2,11} \int_{t_k}^{t_{k+1}} (u_{k,j}(t))^2 \, \mathrm{dt}, \quad k = \{0, 2, 4\}. \tag{12.36}$$

Arnold et al. discuss the excessive internal rotation of the hip of CP patients in [ALS$^+$06]. Therefore, criterion (12.36) can also be interpreted as a convenience criterion, which avoids torques implying additional hip rotations.

To complete the subject-specific CP gait model of type (12.1) derived in this Section, it remains to identify the unknowns p and γ, which is done by solving a hierarchical dynamic optimization problem based on motion capture measurement data for a CP patient obtained from the Heidelberg MotionLab in Section 15.2.

Frame	**Parent**	**Rot Y**	**Rot X**	**Rot Z**	**Trans X**	**Trans Y**	**Trans Z**
Pelvis	–	$\mathfrak{q}_{k,0}$	$\mathfrak{q}_{k,1}$	$\mathfrak{q}_{k,2}$	$\mathfrak{q}_{k,3}$	$\mathfrak{q}_{k,4}$	$\mathfrak{q}_{k,5}$
Thigh R	Pelvis	$\mathfrak{q}_{k,6}$	$\mathfrak{q}_{k,7}$	$\mathfrak{q}_{k,8}$	0.	-0.0720	0.
Shank R	Thigh R	$\mathfrak{q}_{k,9}$	$\mathfrak{q}_{k,10}$	$\mathfrak{q}_{k,11}$	0.	0.	-0.3877
Foot R	Shank R	$\mathfrak{q}_{k,12} + \bar{o}_1$	$\mathfrak{q}_{k,13} + \bar{o}_2$	$\mathfrak{q}_{k,14} + \bar{o}_3$	0.	0.	-0.38680
Thigh L	Pelvis	$\mathfrak{q}_{k,15}$	$\mathfrak{q}_{k,16}$	$\mathfrak{q}_{k,17}$	0.	0.0720	0.
Shank L	Thigh L	$\mathfrak{q}_{k,18}$	$\mathfrak{q}_{k,19}$	$\mathfrak{q}_{k,20}$	0.	0.	-0.3972
Foot L	Shank L	$\mathfrak{q}_{k,21} + \bar{o}_4$	$\mathfrak{q}_{k,22} + \bar{o}_5$	$\mathfrak{q}_{k,23} + \bar{o}_6$	0.	0.	-0.3764
Trunk M	Pelvis	0.1200	-0.0500	0.	0.	0.	0.1350
Trunk U	Trunk M	0.1700	0.	0.	0.	0.	0.
Arm U R	Trunk U	0.	0.	0.	0.	-0.0680	0.1700
Arm L R	Arm U R	0.	0.	0.	0.	0.	-0.1700
Arm U L	Trunk U	0.	0.	0.	0.	0.0680	0.1700
Arm L L	Arm U L	0.	0.	0.	0.	0.	-0.1700
Head	Trunk U	0.1500	0.	0.	0.	0.	0.

Table 12.7.: Overview of multibody-system frames for an able-bodied subject and their relation in rad (column three, four and five) and in m (column six, seven and eight) with 24 DOFs and 14 segments based on measurement data from the Heidelberg MotionLab, see Section 15.1), and the Plug-In-Gait model in the Vicon system [VIC13] rounded to four decimals.

12.3. The healthy gait model

In this section, we derive an optimal control gait model of the form (12.1) for the gait of able-bodied subjects, which is based on the general formulation in Section 12.1 and closely related to the CP gait model derived in the last section. We choose the segmentation illustrated in Figure 12.2(a) with the same degrees of freedom as in the CP gait model: $(\mathfrak{q}_k, \dot{\mathfrak{q}}_k) \in \mathbb{R}^{48}$ and $k \in S_{n_S}$, and the torques/controls $u_k \in \mathbb{R}^{18}$ with $k \in S_{n_S}$. The model stages are chosen to be the same as for the CP gait model, we also have five model stages as illustrated in Figure 12.1. The transformation of the frames of the multibody system are adapted to the body of an able-bodied subject, which is shown in Table 12.7. The offsets $\bar{o}_1, \dots, \bar{o}_6$ describe the patient's neutral zero joint position explained in Section 10.1 on the basis of the posture, where the global and all local frames have the same orientation as illustrated in Figure 12.2(b). The values of $\bar{o}_1, \dots, \bar{o}_6$ are listed in Table 12.7 and depend on measurements and calculated values based on motion capture data for an able-bodied subject from the Heidelberg MotionLab using the Vicon system, and are stated in Table 12.9. The values for the radius of the gyration and the location of the COG of each segment are the same as for the CP gait model (based on [dL96]) shown in Table 12.4 and 12.5, respectively. The length of each segment has been adapted as stated in Table 12.10. The simple bounds, additional constraints and the objective function of (12.1) are chosen as described in the last section for the CP gait model, where the criteria $\phi_{\mathcal{L}}^{k,5}$ and $\phi_{\mathcal{L}}^{k,6}$ for $k \in S_{n_S}$ can also be interpreted as convenience-maximization criteria, which imply an almost two-dimensional gait in the sagittal plane for able-bodied subjects.

Offsets	Value	Description
$\bar{o}_1$	0.0395	Right thigh rotation + offset1
$\bar{o}_2$	-1.4629	-1.57 rad foot rotation + right static plantar flexion - offset2
$\bar{o}_3$	-0.4523	Right tibial torsion + right shank rotation + right static rotation
$\bar{o}_4$	0.0285	Left thigh rotation + offset3
$\bar{o}_5$	-1.4178	-1.57 rad foot rotation + left static plantar flexion + offset4
$\bar{o}_6$	0.3904	- Left tibial torsion - left shank rotation - left static rotation

Table 12.8.: Description of the offset angles in Table 12.7.

Finally, it remains to identify the unknowns p and γ in the subject-specific healthy gait model described in this section, which is done by solving a hierarchical dynamic optimization problem based on motion capture measurement data for an able-bodied subject obtained from the Heidelberg MotionLab in Section 15.3.

12.4. The difference to existing models

In literature, there is a variety of models to analyze human locomotion ranging from simple inverted pendulum models or mass-spring systems ([Kuo07, GSB06]) to highly complex multibody-system models including muscles [DAA+07]. A large class of these models just involve kinematic quantities (like position or velocity of certain parts of the model) and does not include dynamic quantities like forces. However, there is also a large class of models, mostly based on multibody systems, which include dynamic quantities like gravity, forces (torques), or inertia as, e.g., the ones described in [FM12, Mom01, SM10]. Dynamic multibody-system models are often used to analyze the motion of the system for given forces as, e.g. for given torques acting on the multibody-system's joints, or for the generation of motions with a certain objective (cf. [FM12]).

In this thesis, we propose a new type of gait model (12.1), which not just describes the behavior of different parts of the model (or multibody-system segments) based on certain inputs like forces or torques, it also allows to describe a human's gait. This is based on the assumption that humans move optimally minimizing a weighted sum of certain subcriteria, and that the weights γ in the objective of (12.1) (and maybe additional parameters p) have been identified from measurement data (cf. Chapter 15). However, this leads to a dynamic multibody-system-based gait model, which allows to access new information that cannot be accessed by existing models as, e.g., the constitution of the motion's objective. Related approaches can be found in [KML10, KAS+10, ALU12, APS+10, ARARU+11, PJJB12].

In the remainder of this section, we discuss how optimal control gait models contribute to open questions in current research in the field of gait analysis, or more general, in the field of biomechanics (which are already discussed for CP research in a more general framework in Chapter 11).

General understanding of human motion: Optimal control gait models for healthy or pathological gaits increase the general understanding of human motion due to the new information provided by the models based on the optimality assumption.

Quantity	**Value**	
body mass	51	kg
body height	1.560	m
right thigh rotation	0.0245	rad
offset1	0.015	rad
right static plantar flexion	0.2278	rad
offset2	0.12	rad
right tibial torsion	-0.3491	rad
right shank rotation	-0.1122	rad
right static rotation offset	0.090	rad
left thigh rotation	0.0285	rad
offset3	0.	rad
left static plantar flexion	0.2130	rad
offset4	-0.06	rad
left tibial torsion	-0.3490	rad
left shank rotation	-0.0519	rad
left static rotation offset	0.0106	rad

Table 12.9.: Measurements and calculated values based on motion capture data of an able-bodied subject [Wol93, VIC13] rounded to four decimals.

Knowing the constitution of the objective of a human motion and comparing these results for different types of motions (as, e.g., simple arm motions, running or walking) and different groups of people (as, e.g., children, adults, disabled or able-bodied people) provides new insights into human motion in general.

Categories/classification of gaits: The constitution of the optimality criterion of human motion, which is part of an optimal control provides the possibility to use this constitution for developing classes or categories of gaits, e.g., based on the leading subcriterion in the objective of the gait model.

Criteria to evaluate the success of treatments: The evaluation of the success of treatments is a very important issue, which is also in the focus of the research in the Heidelberg MotionLab. Optimal control models could be used to, e.g., evaluate the deviation of a CP gait and a healthy gait before and after CP treatments, which have been successful if the deviation of the patient's gait and a healthy gait has decreased.

Model-based treatment planning: Our primal long-term goal is to use optimal control gait models for model-based treatment planning. The idea is to identify a subject-specific gait model for a patient, which then can be used as a non-invasive test environment for possible treatments providing an idea of how the patient's gait changes after a specific treatment/surgery. For the use in treatment planning, it is indispensable to derive an optimal control gait model. Treatment planning in this form can not be done with, e.g., a dynamic multibody-system model since it would require various types of input like, e.g., torques acting on the joints.

	Segment length (m)	**Segment mass (% of the body mass)**
Pelvis	0.135	0.1117
Thigh R	0.3877	0.1279
Shank R	0.3868	0.0433
Foot R	0.1344	0.0137
Thigh L	0.3972	0.1279
Shank L	0.3792	0.0433
Foot L	0.1314	0.0137
Trunk M	0.17	0.1633
Trunk U	0.17	0.1596
Arm U R	0.17	0.0271
Arm L R	0.17	0.0162
Arm U L	0.17	0.0271
Arm L L	0.17	0.0162
Head	0.15	0.0694

Table 12.10.: Length and mass of each segment in m and % of the total body mass, respectively, based on measurement data of an able-bodied subject from the Heidelberg MotionLab and [dL96].

The topics discussed above can be considered as motivating long-term goals and the contribution of this thesis can be seen as a first step into that direction, which also reveals open questions for future research. The discussion in this section is continued at the end of Chapter 15.

Part IV.

Numerical results

Chapter 13.

Numerical results for regularizing bilevel nonlinear programs by lifting

In this chapter, we present numerical results for the lifting technique derived in Chapter 8. Our lifting approach is tested for nonlinear bilevel programs, as well as general mathematical programs with complementarity constraints from the MacMPEC collection [Ley00], and the numerical results are discussed in detail (cf. [HLSB13]). The last section of this chapter provides numerical results for the lifting technique applied to hierarchical dynamic optimization problems for an illustrative benchmark problem (cf. Section 8.6). We start this chapter with an introduction to the MacMPEC collection and performance profiles.

13.1. The MacMPEC collection and performance profiles

The MacMPEC collection contains mathematical programs with complementarity constraints (MPCCs) formulated in AMPL [FGK02], which is an algebraic modeling language for linear and nonlinear optimization problems including discrete or continuous variables with interfaces to common solvers like CONOPT, CPLEX, LANCELOT, LOQO, MINOS, SNOPT, or filterSQP. A student version of AMPL is freely available and can handle problems with up to 300 variables or constraints. The MacMPEC testset includes 193 problem, which are mainly academic examples and some real-world application problems. All problems are available on

http://wiki.mcs.anl.gov/leyffer/index.php/MacMPEC

for individual download or as gzip'ed tar archive, which includes all problem files.

We consider 165 of the 193 examples and skip problems for which the general structure is not changed by lifting, like in, e.g., `scholtes3`:

$$\begin{aligned} \underset{x_1,x_2}{\text{minimize}} \quad & 0.5\left((x_1-1)^2+(x_2-1)^2\right) \\ & 0 \leq x_1 \perp x_2 \geq 0, \end{aligned} \tag{13.1}$$

which is a general MPCC not arising from a bilevel nonlinear program (NLP), where the complementarity constraint (CC) is the only constraint. Problem (13.1) satisfies MPCC-LICQ (defined in 2.3) and there are no additional constraints to which lifting as described in Chapter 8 could be applied.

We separate the 165 problems into two classes: 47 problems arising from bilevel NLPs and 118 general MPCCs. In the next section, we present numerical results for the lifting technique proposed in Section 8.2 applied to 47 bilevel NLPs from the MacMPEC collection. The third section of this chapter presents numerical results for the lifting approach stated in Section 8.5 applied to 118 general MPCCs from the MacMPEC collection.

We now briefly describe the concept of performance profiles, which are used to illustrate the numerical results in the next two sections following [DM02]. Performance profiles can be interpreted as a probability distribution that a solver outperforms all other solvers. Let $\mathcal{S}$ denote the set of solvers and let $\mathcal{P}$ describe the set of test problems. For each problem $p \in \mathcal{P}$ and solver $s \in \mathcal{S}$, we define

$$t_{p,s} := \text{performance measure for problem } p \text{ and solver } s. \tag{13.2}$$

Examples for the performance measure are run time or number of iterations. The performance ratio is defined by

$$r_{p,s} := \frac{t_{p,s}}{\min\{t_{p,s'} : s' \in \mathcal{S}\}}, \tag{13.3}$$

and the probability for solver $s \in \mathcal{S}$ that a performance ratio $r_{p,s}$ is within a factor $\tau \in \mathbb{R}$ of the best possible ratio is denoted as

$$\rho_s(\tau) := \frac{1}{|\mathcal{P}|}|\{p \in \mathcal{P} : r_{p,s} \leq \tau\}|.$$

The performance profiles then illustrate τ versus $\rho_s(\tau)$.

13.2. Numerical results for lifting bilevel programs

We now analyze the performance of the lifting approach presented in Section 8.2 for 47 problems of the MacMPEC collection [Ley00] arising from bilevel problems (excluding problems that are unchanged by the lifting) and for example (8.31) in Chapter 8. More information about the computing system can be found in Section 14.1. We consider the performance of filterSQP without lifting (formulation (8.16) using slacks), with the lifted ℓ_1-penalty approach (8.30a) including the additional constraint $v \geq 0$ denoted by the solver filterSQP_lift_l1 , and with the lifted ℓ_2-penalty (8.30b) denoted by the solver filterSQP_lift_l2. We furthermore define the set of problems as $\mathcal{P} := \{1, \cdots, 48\}$ and the set of solvers as $\mathcal{S} :=$ {filterSQP, filterSQP_lift_l1, filterSQP_lift_l2}.

The performance profile of filterSQP for the 48 bilevel test problems without lifting, and with ℓ_1-lifting using the penalty function (8.30a) and ℓ_2-lifting (8.30b) is shown in Figure 13.1.

Table 13.1 shows the number of iterations, the objective value in the solution and the penalty parameter π for each problem and each approach. The theoretical results from Section 8.2 are confirmed for example (8.31) from Chapter 8. Without lifting, 12 iterations are needed to solve the problem. Lifting with penalty (8.30b) ensures MPCC-LICQ for all $x \in \mathbb{R}, y \in \mathbb{R}$ and $v \in \mathbb{R}^2$ and the number of iterations decreases from 12 to 7 iteration. Using penalty (8.30a) and the additional constraints $v \geq 0$ leads to 8 iterations. The decrease is even larger for `ex9.2.2` from 22 to 9 iterations for penalty (8.30b).

Using the exact penalty (8.30a), for 28 of 48 problems we use $\pi = 1$. The largest π in this setting is $\pi = 20$. For both settings (penalty (8.30a) and penalty (8.30b)), the lifting is not sensitive with respect to the initial value for v. It some cases it helps to choose the initial for v such that $h(x, y) \geq v$ or even $h(x, y) = v$ is satisfied at the initial point. For penalty (8.30b), the largest value of π for driving v to zero is $1\text{E} + 06$. However, this is only the case for 2 of 48 problems. For 13 problems, $\pi = 1$ is sufficient. For most of the remaining problems we use $\pi = 1\text{E} + 04$ or $\pi = 1\text{E} + 05$. Practical experience shows that

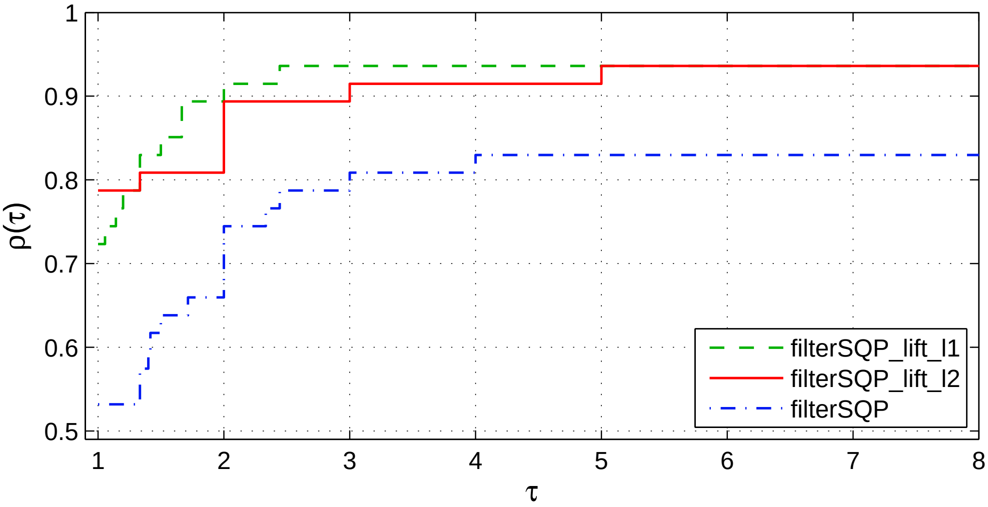


Figure 13.1.: Performance profile of FILTERSQP for bilevel problems from MACMPEC.

for lifting with the inexact penalty, it is reasonable to choose π to be in the same order of magnitude as $\max(F(x_0, y_0), \mathrm{NVAR}, 1\mathrm{E}+04)$, where x_0, y_0 are the initial values for x, y and NVAR is the total number of variables of the problem. For lifting with the exact penalty, $\pi = 1$ is a reasonable choice. If the initial penalty parameter π is not sufficiently large to drive v to zero in the solution, π has to be increased. In Table 13.1, (I) means that the problem is locally infeasible and (ERR) stands for an IEEE error in the AMPL function evaluations. For problem `design-cent2, design-cent3` and `design-cent21`, infeasibility has been detected or an error occurred with and without lifting. For six other problems without lifting the problem is locally infeasible, but after lifting the problem (independent of the penalty we use), FILTERSQP converges to a solution. There are three problems that could not be solved, neither the unlifted nor the lifted problem.

In total, for all bilevel test problems, the number of iterations with lifting has decreased from 273 to 222 iterations for penalty (8.30a), and to 205 for penalty (8.30b) as shown in Figure 13.1 and Table 13.1. Even though we loose MPCC-LICQ when using the exact penalty with $v \geq 0$ for points where $h_i(x, y) = z_i = v_i = 0$ for at least one $i \in \{1, \cdots, n_h\}$, Figure 13.1 shows that the performance of filterSQP_lift_l1 is the best. The performance of filterSQP_lift_l2 is worse than the ℓ_1-lifting, but still better than FILTERSQP without lifting.

	filterSQP_lift_l1			filterSQP_lift_l2			filterSQP	
	iter.	pen.	objf.	iter.	pen.	objf.	iter.	objf.
bard1m	3	3	17	3	1.50E+04	17	3	17
bard2m	3	2	-6598	3	1.00E+04	-6598	3	-6598
bard3m	4	2	-12.68	4	1.00E+04	-12.68	4	-12.68
design-cent-31	94	2	0	89	1.00E+00	0	126	0
desilva	2	1	-1	2	1.00E+00	-1	2	-1
hs044-i	4	1	17.09	4	1.00E+05	17.09	4	17.09
portfl1	6	1	1.50E-005	3	1.00E+04	1.50E-005	6	1.50E-005

portfl2	5	1	1.46E-005	3	1.00E+04	1.46E-005	6	1.46E-005
portfl3	4	1	6.27E-006	4	1.00E+00	6.27E-006	4	6.27E-006
portfl4	4	1	2.18E-006	4	1.00E+02	2.18E-006	4	2.18E-006
portfl6	4	1	2.36E-006	3	4.00E+00	2.36E-006	4	2.36E-006
hs044-i	4	1	17.09	4	1.00E+05	17.09	4	17.09
liswet1-050	1	1	0.16	2	1.00E+05	0.14	1	0.16
liswet1-100	1	1	0.16	3	1.00E+04	0.14	1	0.16
liswet1-200	1	1	0.17	5	1.00E+04	0.15	1	0.17
sl1	1	1	0	1	1.00E+00	0	1	0
bard1	3	3	17	3	1.00E+05	17	3	17
bard2	1	1	6598	1	1.00E+06	6598	1	6598
bard3	2	1	-12.68	2	1.00E+04	-12.68	4	-12.68
bilevel1	3	3	0	3	1.00E+05	0	4	-60
bilevel2	2	1	-6600	2	1.00E+06	-6600	6	-6600
bilevel3	6	5	-20	5	1.00E+05	-20	7	-15.82
ex9.1.1	1	1	-13	1	1.00E+00	-13	1	-13
ex9.1.3	2	20	-29.2	2	1.00E+05	-29.2	2	-29.2
ex9.1.4	1	1	-37	2	1.00E+04	-37	4	-37
ex9.1.5	3	1	-1	2	1.00E+00	-1	3	-1
ex9.1.6	2	3	-21	2	1.00E+05	-21	2	-21
ex9.1.7	5	15	-23	6	1.00E+05	-23	3	-23
ex9.1.8	2	1	-3.25	2	1.00E+04	-3.25	2	-3.25
ex9.1.9	2	2	3.11	2	1.00E+04	3.11	2	3.11
ex9.1.10	1	1	-3.25	2	1.00E+04	-3.25	2	-3.25
ex9.2.1	1	1	2	1	1.00E+05	2	1	2
ex9.2.2	22	7	100	9	1.00E+05	100	22	100
ex9.2.4	3	1	0.5	3	1.00E+00	0.5	3	0.5
ex9.2.5	3	10	9	3	1.00E+00	9	7	9
ex9.2.6	3	1	-1	3	1.00E+00	-1	3	-1
ex9.2.7	2	10	17	2	1.00E+05	17	2	17
ex9.2.8	3	1	-1.5	3	1.00E+00	-1.5	3	-1.5
Example (8.31)	8	1	0	7	1.00E+03	0	12	0
ex9.2.3	3	1	5	3	1.00E+05	5	(I)	
ex9.2.9	1	1	2	1	1.00E+00	2	(I)	
bilin	3	8	5.6	4	1.00E+04	5.6	(I)	
design-cent-1	5	2	1.86	5	1.00E+04	1.86	(I)	
design-cent-4	1	1	0	1	1.00E+00	0	(I)	
design-cent-2	(I)			(I)			(I)	
design-cent-3	(ERR)			(ERR)			(ERR)	
design-cent-21	(I)			(I)			(I)	

Table 13.1.: Numerical results for bilevel problems from MacMPEC.

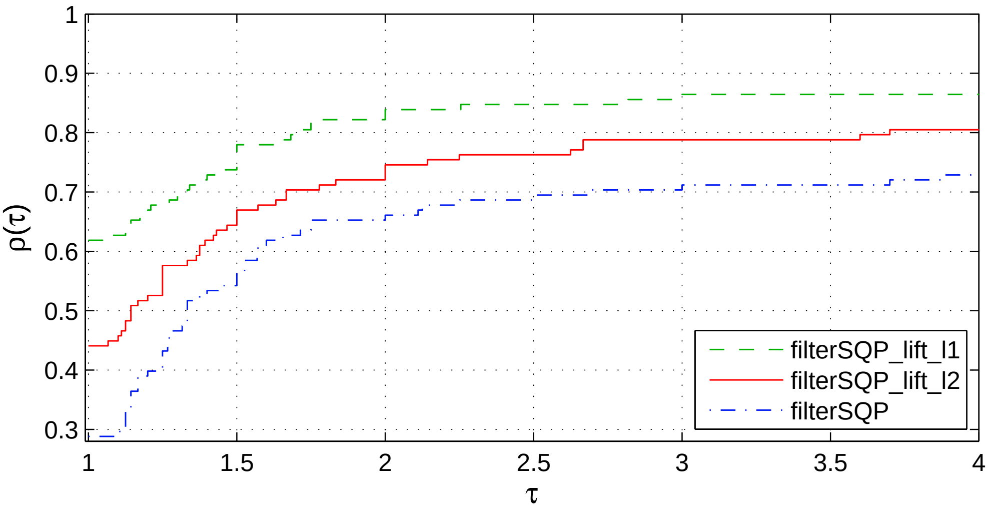


Figure 13.2.: Performance profile of FILTERSQP for general MPCCs from MACMPEC.

13.3. Numerical results for lifting general mathematical programs with complementarity constraints

We now consider general MPCCs from the `MacMPEC` collection which do not arise from bilevel problem – 118 problems in total. Again, note that if lifting does not change the problem formulation, the problem is skipped. Table 13.2 shows the number of iterations, the objective value in the solution and the penalty parameter π for each problem and each approach, and the performance profile is illustrated in Figure 13.2. The total number of iterations decreases from 1288 for the unlifted problem to 871 for the ℓ1-lifted problem, and to 1090 for the ℓ2-lifted problem. The performance profile in Figure 13.2 clearly shows that filterSQP_lift_l1 performs best. Ten problems could not be solved (neither lifted nor unlifted) because of infeasibility. In Table 13.2, (no conv) means that the trust region becomes too small, ($v \neq 0$) means that we were not able to drive v to zero, and (fail QP) means that the QP solver exits with an error. For 14 problems, FILTERSQP detected infeasibility in the unlifted problem, but after lifting (independent of the penalty), FILTERSQP converges to a solution. There is one problem of special interest, which is problem `scholtes4`, known for a solution which is a B-stationary point that is not strongly stationary. Without lifting, FILTERSQP detects infeasibility. After lifting, the problem can be solved and FILTERSQP converges to the reported solution. There are two problems, where FILTERSQP without lifting converges, but we cannot get a solution for the lifted problem (independent of the penalty we use).

For a better understanding of the convergence behavior of FILTERSQP applied to MPCCs with and without lifting, we take a closer look at problem `incidset1-8`. Figure 13.3 shows the convergence behavior without lifting and with ℓ_1-lifting, where the constraint violation on the y-axis is plotted on a logarithmic scale. Except for the last iteration, Figure 13.3 indicates a linear rate of convergence for the unlifted problem, whereas with lifting, we obtain a superlinear rate of convergence.

In summary, lifting the complementarity constraint (independent of the penalty function)

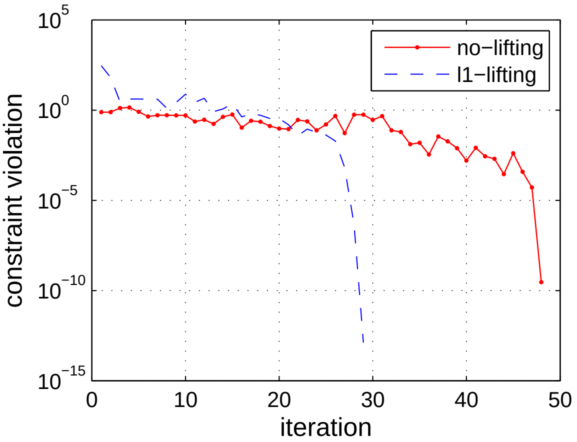


Figure 13.3.: Convergence behavior of `incidset1-8`.

leads to a more stable method with a better convergence behavior, for bilevel programs, but also for general MPCCs. Practical experience and the performance profiles in Figures 13.1 and 13.2 show, that lifting with the exact penalty performs best, and if we converge to a KKT-point (cf. Chapter 2) of the ℓ_1-lifted problem with $v^* = 0$, we can guarantee that this is also a KKT-point of the original unlifted problem. Furthermore, the lifting proposed in this thesis does not require a tailored solver, standard SQP solvers can be used after adapting the problem formulation.

	filterSQP_lift_l1			filterSQP_lift_l2			filterSQP	
	iter.	pen.	objf.	iter.	pen.	objf.	iter.	objf.
bar-truss	13	1000	10166.6	37	1.00E+06	10166.6	10	10166.6
dempe	39	1	28.25	8	1.00E+05	49	58	28.25
df-1	2	1	0	2	1.00E+00	0	2	0
flp2	1	1	0	1	1.00E+00	0	1	0
flp4-1	3	1	0	3	1.00E+00	0	3	0
flp4-2	3	1	0	3	1.00E+00	0	3	0
flp4-3	3	1	0	3	1.00E+00	0	3	0
flp4-4	3	1	0	3	1.00E+00	0	3	0
gnash10	7	1	-230.82	7	1.00E+05	-230.82	8	-230.82
gnash11	7	1	-129.91	7	1.00E+05	-129.91	7	-129.91
gnash12	8	1	-36.93	7	1.00E+05	-36.93	8	-36.93
gnash13	8	1	-7.06	8	1.00E+05	-7.06	12	-7.06
gnash14	14	1	-0.18	8	1.00E+04	-0.18	24	-0.18
gnash15	9	100	-354.7	8	1.00E+05	-354.7	17	-354.7
gnash16	12	1	-241.44	7	1.00E+05	-241.44	14	-241.44
gnash17	8	1	-90.75	9	1.00E+05	-90.75	12	-90.75
gnash18	14	100	-25.7	8	1.00E+05	-25.7	14	-25.7
gnash19	7	1	-6.12	8	1.00E+05	-6.12	9	-6.12
gnash10m	7	1	-230.82	8	1.00E+05	-230.82	11	-230.82
gnash11m	7	1	-129.91	7	1.00E+05	-129.91	12	-129.91

gnash12m	8	1	-36.93	9	1.00E+05	-36.93	10	-36.93
gnash13m	8	1	-7.06	8	1.00E+04	-7.06	11	-7.06
gnash14m	8	1	-0.18	8	1.00E+04	-0.18	31	-0.18
hakonsen	10	1	24.37	10	1.00E+04	24.37	10	24.37
incid-set1-8	19	1	0.23	27	1.00E+00	0	29	0.23
incid-set1-16	40	1	0.17	39	1.00E+00	0	28	0.17
incid-set1c-8	23	1	0.23	19	9.00E+00	0	29	0.23
incid-set1c-16	37	1	0.17	22	1.00E+00	0	28	0.17
incid-set2-32	1	1	0	1	1.00E+00	0	169	0.02
monteiro	8	1000	37.53	10	1.00E+06	38.25	9	37.53
monteiroB	8	1000	827.86	10	1.00E+06	828.04	9	827.86
nash1	1	1	0	1	1.00E+00	0	5	0
outrata31	7	3	3.21	7	1.00E+04	3.21	8	3.21
outrata32	7	2	3.45	7	1.00E+04	3.45	8	3.45
outrata33	6	2	4.6	6	1.00E+04	4.6	7	4.6
outrata34	6	1	6.59	6	1.00E+04	6.59	6	6.59
pack-comb1-8	5	1	0.6	6	1.00E+03	0.6	8	0.6
pack-comb1-16	9	110	0.62	10	1.00E+03	0.6	19	0.62
pack-comb1c-8	6	1	0.6	8	1.00E+03	0.6	8	0.6
pack-comb1c-16	7	27	0.62	10	1.00E+03	0.82	5	0.62
pack-comb2-8	8	14	0.67	10	1.00E+05	0.67	8	0.67
pack-comb2-16	10	30	0.73	36	1.00E+05	0.72	37	0.73
pack-comb2c-8	9	15	0.66	9	1.00E+04	0.67	6	0.67
pack-comb2c-16	10	30	0.73	10	1.00E+04	0.7	15	0.73
pack-rig1-4	8	2	0.72	10	1.00E+05	0.72	8	0.72
pack-rig1-8	14	3	0.79	11	1.00E+05	0.79	14	0.79
pack-rig1-16	27	3	0.83	23	1.00E+05	0.82	62	0.83
pack-rig1c-4	5	10	0.72	5	1.00E+05	0.72	6	0.72
pack-rig1c-8	8	10	0.79	8	1.00E+05	0.79	9	0.79
pack-rig1c-16	7	1	0.83	11	1.00E+05	0.82	11	0.83
pack-rig1p-4	6	1	0.6	7	1.00E+04	0.6	7	0.6
pack-rig1p-8	16	3	35.94	22	1.00E+06	0.73	12	35.94
pack-rig1p-16	27	7	264.48	27	1.00E+06	0.72	18	264.5
pack-rig2-4	8	2	0.69	11	1.00E+05	0.69	9	0.69
pack-rig2-8	9	6	0.77	16	1.00E+05	0.78	9	0.78
pack-rig2c-4	4	2	0.71	8	1.00E+05	0.71	7	0.71
pack-rig2c-8	7	8	0.8	8	5.00E+05		7	0.8
pack-rig2p-4	6	1	0.6	6	1.00E+04	0.6	6	0.6
pack-rig2p-8	15	6	46.68	22	1.00E+07	0.76	19	46.68
pack-rig2p-16	19	400	625.94	31	1.00E+06	-14.44	25	625.93
qpec-100-1	6	2	0.1	5	1.00E+04	0.1	7	0.1
qpec-100-2	6	3	-6.59	9	1.00E+05	-3.84	7	-6.26
qpec-100-3	4	2	-5.48	5	1.00E+05	-5.38	5	-5.48
qpec-100-4	4	5	-4.06	4	1.00E+05	-1.44	5	-3.6

qpec-200-1	4	5	-1.93	9	1.00E+05	-1.93	10	-1.94
qpec-200-2	11	3	-23.88	15	1.00E+05	-22.69	11	-24.04
qpec-200-3	10	3	-1.92	11	1.00E+06	-1.93	11	-1.95
qpec-200-4	4	3	-6.04	5	1.00E+06	-6.04	5	-6.22
ralphmod	59	1	-683.03	44	1.00E+01	-683.03	64	-683.03
scholtes1	3	1	2	3	1.00E+04	2	4	2
scholtes2	2	4	15	2	1.00E+07	15	2	15
stackelberg1	4	1	-3266.67	4	1.00E+06	-3266.67	4	-3266.67
tap-09	8	1	109.15	11	1.00E+04	109.15	8	109.13
TraficSignalCycle-1	3	1	56.73	8	1.00E+05	54.96	4	56.73
TraficSignalCycle-2	3	1	54.34	8	1.00E+04	52.57	4	54.34
TraficSignalCycle-3	2	1	88.84	16	1.00E+04	87.07	1	88.84
TraficSignalCycle-4	4	1	80.81	31	1.00E+05	79.04	9	80.84
TraficSignalCycle-5	3	1	103.24	25	1.00E+05	101.47	1	103.24
TraficSignalCycle-6	3	1	103.3	26	1.00E+04	101.53	3	103.3
TraficSignalCycle-9	3	1	54.98	6	1.00E+04	53.2	3	54.98
TraficSignalCycle-10	3	1	56.57	15	1.00E+04	54.8	3	56.57
TraficSignalCycle-11	2	1	103.34	20	1.00E+04	101.57	1	103.34
TraficSignalCycle-13	3	1	88.17	5	1.00E+04	86.4	4	88.17
water-net	95	5	974.39	136	1.00E+07	918.43	149	918.36
b-pn2	33	1	0.09	55	1.00E+06	1020.93	(I)	
gauvin	3	7	20	3	1.00E+04	20	(I)	
gnash15m	12	1	-354.7	8	1.00E+04	-354.7	(I)	
gnash16m	9	1	-241.44	8	1.00E+05	-241.44	(I)	
gnash17m	12	1	-90.75	8	1.00E+05	-90.75	(I)	
gnash18m	13	1	-25.7	8	1.00E+04	-25.7	(I)	
gnash19m	11	1	-6.12	8	1.00E+04	-6.12	(I)	
scholtes4	19	1	0	18	1.00E+04	0	(I)	
taxmcp	15	11	0.82	16	1.00E+04	0.82	(I)	
incid-set1-32	115	1	0.15	51	1.00E+00	0	(I)	
incid-set1c-32	56	1	0.15	147	1.00E+00	0	(I)	
incid-set2-8	1	1	0	1	1.00E+00	0	(I)	
incid-set2-16	1	1	0	1	1.00E+00	0	(I)	
incid-set2c-8	21	1	0.02	45	1.00E+04	0.03	(I)	
incid-set2c-16	131	100	0	(I)			(I)	
incid-set2c-32	(I)			(no conv)			(I)	
TraficSignalCycle-7	(I)			(v $\neq$ 0)			(I)	
TraficSignalCycle-8	(I)			(v $\neq$ 0)			(I)	
TraficSignalCycle-12	(I)			(v $\neq$ 0)			(I)	
pack-comb1-32	(no conv)			(I)			(I)	
pack-comb2-32	(fail QP)			(fail QP)			(I)	
pack-comb2c-32	(fail QP)			(I)			(I)	
pack-comb1c-32	(no conv)			(no conv)			(I)	
pack-rig2-16	(I)			28	1.00E+05	0.83	(I)	

pack-rig2-32	(I)			145	1.00E+04	0.74	(I)	
pack-rig2c-16	(I)			13	5.00E+05	0.92	(I)	
pack-rig2c-32	(I)			(I)			(I)	
tap-15	(I)			(I)			(I)	
pack-rig1-32	(no conv)			(no conv)			40	0.85
pack-rig1c-32	18	10	0.85	(I)			12	0.85
pack-rig1p-32	61	100	2242.07	(I)			101	2242.06
pack-rig2p-32	(I)			(I)			23	871.75
tollmpec	31	100	208.26	(v $\neq$ 0)			11	208.26
tollmpec1	19	100	979.39	(v $\neq$ 0)			(I)	

Table 13.2.: Numerical results for general MPCCs from MacMPEC.

13.4. Lifting an illustrative hierarchical dynamic optimization problem: the polar robot example

In this section, we present numerical results for an illustrative lifted hierarchical dynamic optimization problem. We consider optimal point-to-point trajectories of a robot arm illustrated in Figure 13.4. The dynamics of the robot are modeled based on a multibody system (cf. Chapter 10) with three degrees of freedom (DOFs) as illustrated in Figure 13.4:

A rotatory DOF (revolute joint):
the angle φ_1 between the two segments of the robot.

A rotatory DOF (revolute joint):
the angle φ_2 describing the rotation of the robot's base.

A translatory DOF:
the length of the robot arm r (the robot arm can be retracted).

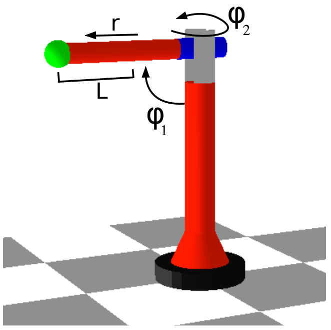


Figure 13.4.: Illustration of a robot arm with three degrees of freedom.

The optimal control problem

Optimal point-to-point trajectories of the robot can be described as the solution of the following optimal control problem (OCP):

$$\begin{array}{lrcll}
\underset{x,u,q}{\text{minimize}} & \multicolumn{3}{l}{\gamma_1 \int_0^1 u_2(t)^2 \,\mathrm{dt} + \gamma_2\, q_0 =: \Phi} & \\
\text{subject to} & \ddot{x}(t) & = & f(x(t), u(t), q), & t \in [0,1] \\
& x_0(0) & = & 0, \quad x_1(0) = 0, \quad x_2(0) = 0.7 & \\
& x_3(0) & = & 0, \quad x_4(0) = 0, \quad x_5(0) = 0 & \\
& x_0(1) & = & 0, \quad x_1(1) = 0, \quad x_2(1) = -0.1 & \\
& x_3(1) & = & 0, \quad x_4(1) = 0, \quad x_5(1) = 0 & \\
& -1 & \le & x_0(t) \le 1, \quad -1 \le x_1(t) \le 1, \quad -0.5 \le x_2(t) \le 0.7, & t \in [0,1] \\
& -5 & \le & x_3(t) \le 5, \quad -5 \le x_4(t) \le 5, \quad -6 \le x_5(t) \le 6, & t \in [0,1] \\
& -1000 & \le & u_0(t) \le 1000, & t \in [0,1] \\
& -1500 & \le & u_1(t) \le 1500, & t \in [0,1] \\
& -1000 & \le & u_2(t) \le 1000, & t \in [0,1]
\end{array} \tag{13.4}$$

with differential states and controls

$$x(t) := \begin{pmatrix} x_0(t) \\ x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \\ x_5(t) \end{pmatrix} = \begin{pmatrix} \varphi_1(t) \\ \varphi_2(t) \\ r(t) \\ \dot{\varphi}_1(t) \\ \dot{\varphi}_2(t) \\ \dot{r}(t) \end{pmatrix} \quad \text{and} \quad u(t) := \begin{pmatrix} u_0(t) \\ u_1(t) \\ u_2(t) \end{pmatrix}, \tag{13.5}$$

where the controls $(u_0(t), u_1(t), u_2(t))$ describe the torques acting on $(\varphi_1, \varphi_2, r)$, respectively. The differential equation $\ddot{x}(t) = f(x(t), u(t), q)$ is in detail given by

$$\begin{aligned}
\ddot{\varphi}_1 &= q_0 \frac{(u_0+((18.5+40r^2+10(r+L)^2-0.12)\sin(2\varphi_2)\dot{\varphi}_2-2(40r+10(r+L)\dot{r}\cos(\varphi_2)^4))\dot{\varphi}_1}{10+(18.5+40r^2 10(r+L)^2\cos(\varphi_2)^2+0.12\cos(\varphi_2)^2)} \\
\ddot{\varphi}_2 &= q_0 \frac{u_1+(0.12-18.5-40r^2-10.(r+L)^2)0.5\sin(2\varphi_2)\dot{\varphi}_1^2-(40r+10(r+L))(g\cos(\varphi_2)+2\dot{r}\dot{\varphi}_2}{18.5+40r^2+10(r+L)^2} \\
\ddot{r} &= q_0 \frac{(u_2+(40r+10(r+L))(\cos(\varphi_2)^2\dot{\varphi}_1^2+\dot{\varphi}_2^2)-50g\sin(\varphi_2))}{50},
\end{aligned} \tag{13.6}$$

where the argument t of $\varphi_1, \varphi_2, \dot{\varphi}_1, \dot{\varphi}_2, \ddot{\varphi}_1, \ddot{\varphi}_2, r, \dot{r}, \ddot{r}$ is skipped for a compact presentation. The parameter L describes the half length of the robot arm (cf. Figure 13.4) and $g \approx 9.81\frac{\mathrm{m}}{\mathrm{s}^2}$ is the gravitational acceleration. The control value q_0 in (13.6) describes the length of the time horizon of the motion (the OCP (13.4) is transformed on the fix time horizon [0,1], cf., e.g., Chapter 12 or [Hat08]). The task of the robot is to find the optimal path between a fixed initial rest position and a fixed final rest position, as illustrated in Figure 13.5. Bounds on the differential states and on the controls are given in (13.4). The objective of (13.4) includes two subcriteria. The first subcriterion is $\int_0^1 u_2(t)^2$ dt, which can be interpreted as the retraction energy, and the second one is denoted by q_0, which is the duration of the motion.

We now solve (13.4) for $\gamma_1 = 0.5$, $\gamma_2 = 1\text{E}{+}04{\cdot}0.5$ and $L = 0.75$ with ParaOCP described in Chapter 9 using six equally sized multiple shooting intervals, a piecewise constant control discretization on the multiple shooting grid and a KKT tolerance of 1E-06 for filterSQP in ParaOCP (see Chapter 9). As initial values, we use $q_0 = 1$, all differential states are set to 0 and all controls are set to 10 on the multiple shooting nodes. The differential states x and the controls u in solution of (13.4) are shown in Figure 13.7. For the control values q_0 representing the time horizon of the process, we have $q_0 = 1.1526\text{E}{+}00$.

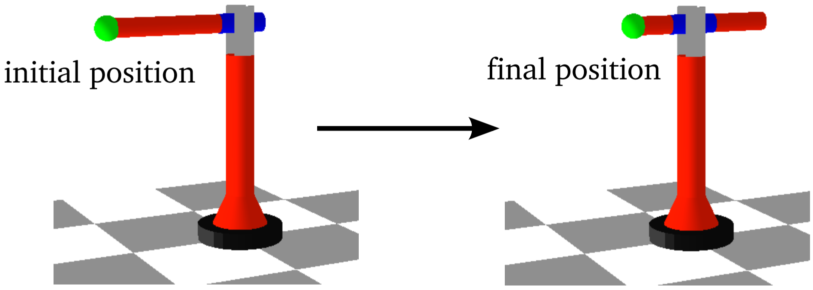


Figure 13.5.: Illustration of the inital and final position of the desired point-to-point trajectory.

The hierarchical dynamic optimization problem

We now simulate measurements using the solution of (13.4) for $\gamma_1 = 0.5$, $\gamma_2 = 1\text{E}{+}04 \cdot 0.5$ and $L = 0.75$. Therefore, we assume that the measurement grid is equal to the multiple shooting grid and we add normally distributed random numbers with mean 0 and the standard deviations $\sigma_{ij} = 0.01 \cdot 0.5 \ \forall i \in 0, \ldots, n_T, \ j \in 0, \ldots, 5$ and $\sigma_{ij} = 0.01 \ \forall i \in 0, \ldots, n_T, \ j \in 0, \ldots, 5$ (which corresponds to an error of approximately 1%) to the solution $x^*(t)$ of the OCP (13.4) and obtain

$$\begin{aligned} \eta = [\ &(\eta_{0,1},\ \eta_{0,2},\ \eta_{0,3}, \eta_{0,4},\ \eta_{0,5},\ \eta_{0,6})\,, \\ &\ldots,\ \ (\eta_{n_T,1},\ \eta_{n_T,2},\ \eta_{n_T,3}, \eta_{n_T,4},\ \eta_{n_T,5},\ \eta_{n_T,6})\ ]^{\mathsf{T}} \end{aligned} \tag{13.7}$$

with $\eta \in \mathbb{R}^{(n_T+1)\times 6}$, $n_T = 6$, and measurement times $\tau_i = \frac{i}{n_T} \ \forall i = 0, \ldots, n_T$. We use simulated measurements in this section since they allow a better investigation of the proposed methods: the direct all-at-once approach for hierarchical dynamic optimization problems proposed in Chapter 5 and the lifting technique introduced in Chapter 8. We now consider the following hierarchical dynamic optimization problem in order to identify the parameters γ_1, γ_2 in the objective and the parameter L in the ODE's right-hand side describing the half length of the robot arm from the simulated measurements (13.7):

$$
\begin{array}{lll}
\underset{\substack{x,u,q,\\ L,\gamma}}{\text{minimize}} & \frac{1}{2}\sum\limits_{i=0}^{n_T}\sum\limits_{j=1}^{6}\frac{(x_{j-1}(\tau_i)-\eta_{ij})^2}{\sigma_{ij}^2} & \\
\text{subject to} & \underset{x,u,q}{\text{minimize}} & \gamma_1 \int_0^1 u_2(t)^2\,\mathrm{dt} + \gamma_2\; q_0 =: \Phi \\
 & \text{subject to} & \begin{array}{lcll}
\ddot{x}(t) & = & f(x(t),u(t),q), & t \in [0,1] \\
x_0(0) & = & 0, \quad x_1(0) = 0, \quad x_2(0) = 0.7 & \\
x_3(0) & = & 0, \quad x_4(0) = 0, \quad x_5(0) = 0 & \\
x_0(1) & = & 0, \quad x_1(1) = 0, \quad x_2(1) = -0.17 & \\
x_3(1) & = & 0, \quad x_4(1) = 0, \quad x_5(1) = 0 & \\
-1 & \le & x_0(t) \le 1, \quad -1 \le x_1(t) \le 1, & \forall t \in (0,1) \\
-0.5 & \le & x_2(t) \le 0.7, \quad -5 \le x_3(t) \le 5, & \forall t \in (0,1) \\
-5 & \le & x_4(t) \le 5, \quad -6 \le x_5(t) \le 6, & \forall t \in (0,1) \\
-1000 & \le & u_0(t) \le 1000, & \forall t \in [0,1] \\
-1500 & \le & u_1(t) \le 1500, & \forall t \in [0,1] \\
-1000 & \le & u_2(t) \le 1000, & \forall t \in [0,1]
\end{array} \\
 & 0 \le \gamma_1, \gamma_2 & \\
 & \gamma_2 = 1\text{E+}04 \cdot 0.5, &
\end{array}
\tag{13.8}
$$

where the detailed ODE in (13.8) is given in (13.6). Note that γ_2 has been eliminated in (13.8) by setting it to a fix value, which can be done if we know that the true value of γ_2 is not zero. We use six multiple shooting intervals, a Gauß-Newton Hessian approximation, efficient Hessian calculations for the lower-level OCP (explained in Chapter 5 and 9), a piecewise constant control discretization on the multiple shooting grid and a KKT tolerance of 1E-06 for FILTERSQP. We furthermore need to specify initial data for the differential states, the controls, the control values, the upper-level parameters and the Lagrange multipliers of the lower-level OCP (cf. Chapter 5). Therefore, we solve the OCP (13.4) using an initial guess for the true values of γ_1, γ_2 and L. We choose $\gamma_1 = 0.2, \gamma_2 = 1\text{E+}04 \cdot 0.5$ and $L = 0.6$, which significantly changes the structure of the solution, and solve (13.4) using these guesses. The resulting differential states, the controls, the control value and the Lagrange multipliers of the equality and inequality constraints are then used as initial values for problem (13.8), as well as $\gamma_1 = 0.2$ and $L = 0.6$. We solve the one-level formulation of (13.8) (explained in Chapter 5) with PARAOCP and after 671 iterations, PARAOCP terminates because of an infeasible quadratic program (QP) approximation with $\gamma_1 = 1.9472\text{E-}02$ and $L = 5.9287\text{E-}01$. The differential states, the measurements used in the objective and the controls in the termination point are illustrated in Figure 13.6. The value of the objective function in the termination point is 1.5611E+04. As discussed in Chapter 8, the one-level formulation of (13.8) includes a complementarity constraint, which leads to a lack of the linear independence constraint qualification (LICQ) at any feasible point. In the termination point, the one-level formulation of (13.8) does not even satisfy the adapted constraint qualification MPCC-LICQ (defined in Section 2.3), which might be the reason for the infeasible QP. To remedy the lack of MPCC-LICQ, we lift problem (13.8) as proposed in Section 8.6. Therefore, we define the parameterized differential states as

$$
s := \left[(s_{0,0}, s_{0,1}, s_{0,2}, s_{0,3}, s_{0,4}, s_{0,5}), \ldots, (s_{6,0}, s_{6,1}, s_{6,2}, s_{6,3}, s_{6,4}, s_{6,5})\right]^{\mathsf{T}}, \tag{13.9}
$$

the discretized controls as

$$
w := \left[(w_{0,0}, w_{0,1}, w_{0,2}), \ldots, (w_{5,0}, w_{5,1}, w_{5,2})\right]^{\mathsf{T}}, \tag{13.10}
$$

and the lifting variables are given by

$$\begin{aligned} v := \Big[&(v^{s,l}_{0,0},\dots,v^{s,l}_{6,0}),\dots,(v^{s,l}_{0,5},\dots,v^{s,l}_{6,5}),(v^{s,u}_{0,0},\dots,v^{s,u}_{6,0}),\dots,(v^{s,u}_{0,5},\dots,v^{s,u}_{6,5}),\\ &(v^{w,l}_{0,0},\dots,v^{w,l}_{5,0}),\dots,(v^{w,l}_{0,2},\dots,v^{w,l}_{5,2}),(v^{w,u}_{0,0},\dots,v^{w,u}_{5,0}),\dots,(v^{w,u}_{0,2},\dots,v^{w,u}_{5,2})\,\Big]^{\mathsf{T}}. \end{aligned} \tag{13.11}$$

The inequality constraints in the discretized one-level formulation of (13.8) are replaced by

$$\begin{array}{rcccl} -1+v^{s,l}_{i,0} & \le & s_{i,0} & \le & 1-v^{s,u}_{i,0}, \quad \text{for } i=0,\dots,6\\ -1+v^{s,l}_{i,1} & \le & s_{i,1} & \le & 1-v^{s,u}_{i,1}, \quad \text{for } i=0,\dots,6\\ -0.5+v^{s,l}_{i,2} & \le & s_{i,2} & \le & 0.7-v^{s,u}_{i,2}, \quad \text{for } i=0,\dots,6\\ -5+v^{s,l}_{i,3} & \le & s_{i,3} & \le & 5-v^{s,u}_{i,3}, \quad \text{for } i=0,\dots,6\\ -5+v^{s,l}_{i,4} & \le & s_{i,4} & \le & 5-v^{s,u}_{i,4}, \quad \text{for } i=0,\dots,6\\ -6+v^{s,l}_{i,5} & \le & s_{i,5} & \le & 6-v^{s,u}_{i,5}, \quad \text{for } i=0,\dots,6 \end{array} \tag{13.12}$$

and

$$\begin{array}{rcl} -1000+v^{w,l}_{i,0} & \le & w_{i,0} \le 1000-v^{w,u}_{i,0}, \quad \text{for } i=0,\dots,5\\ -1500+v^{w,l}_{i,1} & \le & w_{i,1} \le 1500-v^{w,u}_{i,1}, \quad \text{for } i=0,\dots,5\\ -1000+v^{w,l}_{i,2} & \le & w_{i,2} \le 1000-v^{w,u}_{i,2}, \quad \text{for } i=0,\dots,5. \end{array} \tag{13.13}$$

We use the exact penalty function (8.30a) explained in Chapter 8 with the nonnegativity constraint $v \ge 0$ on the lifting variables. The penalty term is added to the objective function of the discretized one-level formulation of (13.8) leading to the new objective

$$\underset{\substack{x,u,q,\\ L,\gamma,v}}{\text{minimize}}\ \frac{1}{2}\sum_{i=0}^{n_T}\sum_{j=1}^{6}\frac{(x_{j-1}(\tau_i)-\eta_{ij})^2}{\sigma_{ij}^2}+\pi\|v\|_1. \tag{13.14}$$

We solve the lifted discretized formulation of (13.8) of the form (8.54) with ParaOCP and the penalty parameter $\pi = 30$. After 229 iterations, ParaOCP converges with an objective value of 6.7863E-01, $v = 0$, $\gamma_1 = 4.8453\text{E-}01$ and $L = 7.4821\text{E-}01$. The differential states x and the controls u in the solution of the lifted discretized one-level formulation of (13.8) (cf. Section 8.6) are illustrated in Figure 13.7. In the remainder of this thesis, we call the solution of a discretized one-level formulation of a hierarchical dynamic optimization problem just *solution of a hierarchical dynamic optimization problem* without explicitly mentioning that we solve the discretized one-level formulation when using a direct all-at-once approach.

The numerical results for lifting a hierarchical dynamic optimization problem follow the experience reported in the last section: Lifting helps to deal with the infeasible QP arising from the lack of MPCC-LICQ.

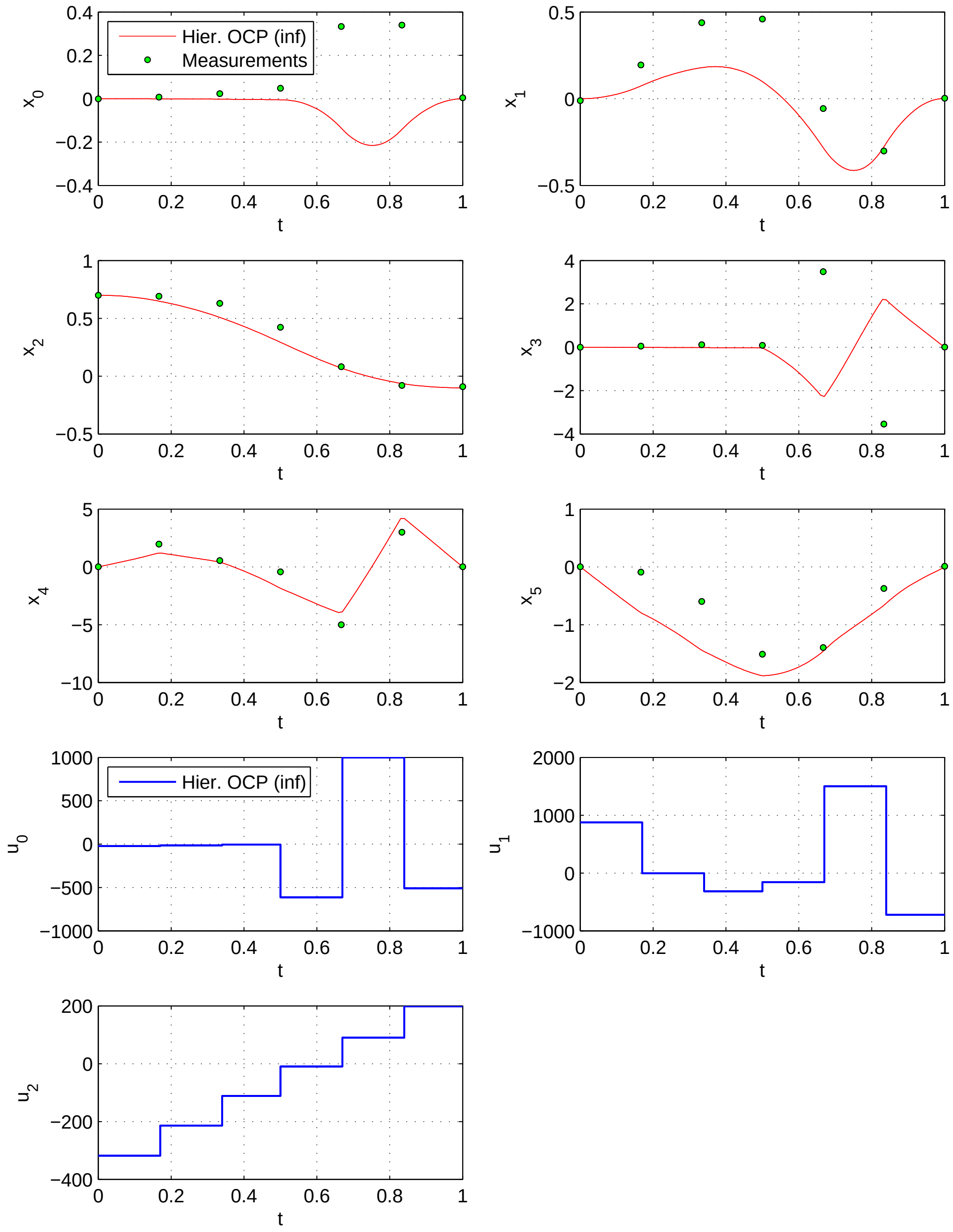


Figure 13.6.: Simulated measurements of the solution x of (13.4) (circles), differential states x and controls u of (13.8), when ParaOCP exits because of an infeasible QP approximation (solid line).

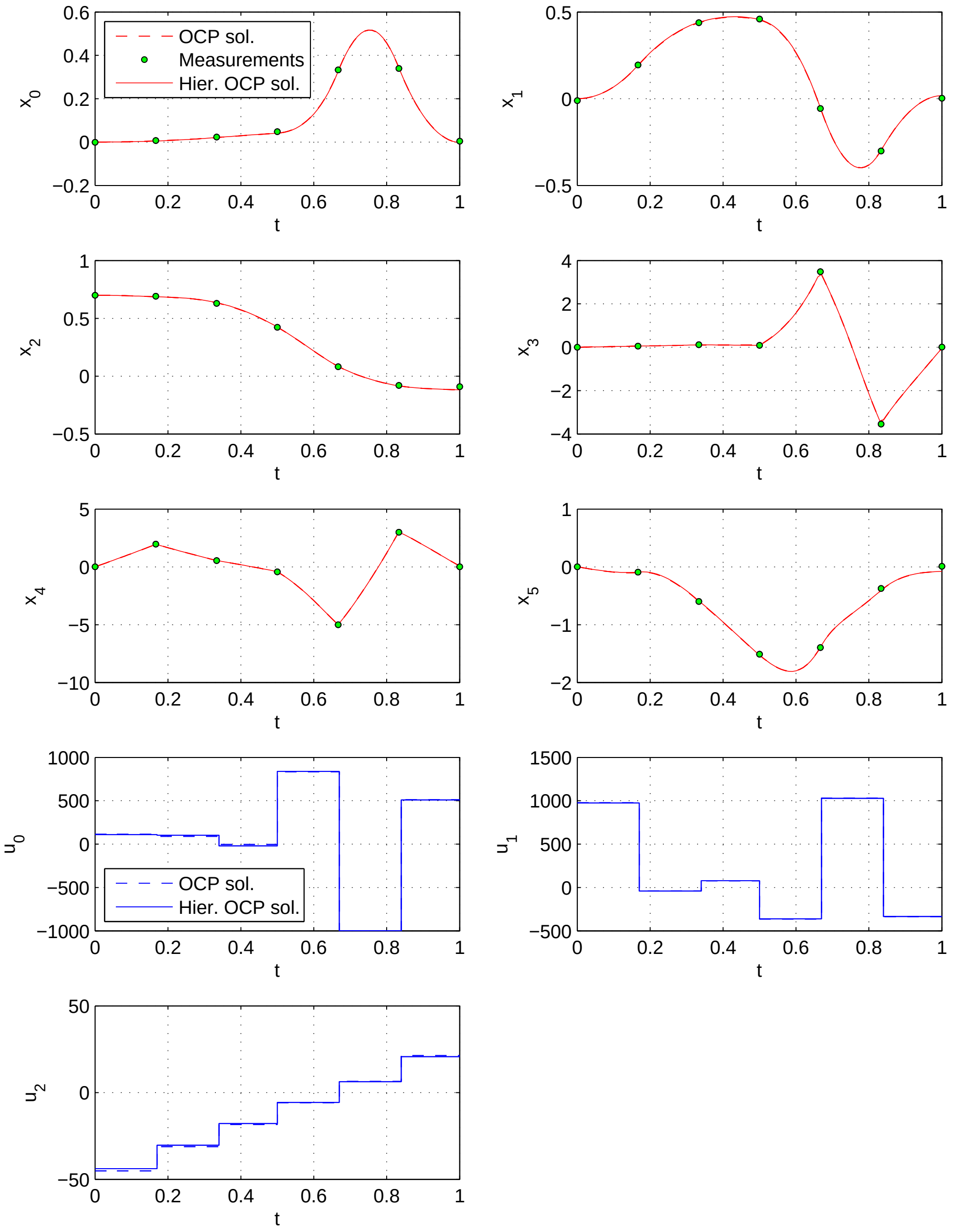


Figure 13.7.: Simulated measurements of the solution x of (13.4) (circles), differential states x and controls u in the solution of the OCP (13.4) (dashed line) and in the solution of the lifted hierarchical formulation (solid line).

Chapter 14.

Hierarchical dynamic optimization for a new collection of benchmark problems

In this chapter, we introduce a benchmark test set for hierarchical dynamic optimization problems consisting of four illustrative examples. Each example is solved with the direct all-at-once approach presented in Chapter 5 and the numerical results are discussed in detail. We furthermore show the performance of the two bilevel approaches for hierarchical dynamic optimization problems described in Chapter 7, where one approach is based on a derivative-free optimization technique, and the second approach is built on a bundle method.

14.1. Preliminaries

In the first section of this chapter, we briefly state some preliminaries like the exact computing system we use. We furthermore discuss how we numerically deal with parameter estimation problems with a variable time horizon.

Computing system

All computations in this chapter are performed on a PC with an Intel Xeon(R) W3565 processor with 3.2 GHz and 11.7 GB RAM with Ubuntu version 12.04. We furthermore use the C++ compiler version 4.6 and the Fortran compiler gfortran version 4.6.3 to compile ParaOCP and filterSQP.

Parameter estimation with variable time horizon

In this thesis, we consider parameter estimation problems constrained by an optimal control problem (OCP) of the form (4.1). The total time horizon or the duration of several model stages might be unknown and an optimization variable (if, e.g., one of the subcriteria in the lower-level objective function is the duration of the process). However, in parameter estimation, measurement times are often assumed to be fix since time can be measured very accurately. Furthermore, a variable time horizon might lead to discontinuities in the parameter estimation problem. We deal with these issues as follows. In all the computations in this thesis where we have a variable time horizon or model stage duration, we apply a time transformation and formulate the parameter estimation problem with an OCP as constraint on a fix time horizon using a control value to describe the free final time (see, e.g., below in (14.3)). That way, we avoid the discontinuity that would appear without a time transformation on a fix time horizon when the variable end time passes a measurement node (i.e. when the time horizon of the parameter estimation problem is less than the measured end time). Furthermore, the duration of the process is not expected to change significantly during the optimization since we have measured the end time of the process. However, if the measured end time is significantly different from the end time (or model stage duration) in the solution of the parameter estimation problem with an OCP as constraint, this is indication for initial values far away from the solution, an incorrect optimal control model or an incorrect solution method.

Miscellaneous

We finally note that in this section, our goal is to show the performance of the pure direct all-at-once approach described in Chapter 5. Hence, we do not lift the complementarity constraint as described in Section 13.4.

Furthermore, we call the solution of a discretized one-level formulation of a hierarchical dynamic optimization problem where the discretized lower-level problem is replaced by first-order optimality conditions just *solution of a hierarchical dynamic optimization problem* without explicitly mentioning that we solve the discretized one-level formulation when using a direct all-at-once approach.

14.2. The rocket car example

The first process we consider in this chapter is the optimal movement of a rocket car, which is already described in Section 6.2 (cf. [HSB12]). For the sake of completeness, we briefly recall the formulation of the OCP and the resulting hierarchical dynamic optimization problem. The OCP is given by

$$\begin{array}{lrcll}
\underset{x,u,q}{\text{minimize}} & \multicolumn{3}{l}{-\gamma_1\, m(1) + \gamma_2\, q_0} & \\
\text{subject to} & 0 & = & \dot{z}(t) - q_0 v(t), & t \in [0,1] \\
 & 0 & = & \dot{v}(t) - q_0(u(t) - p_1 v(t)^2)/m(t), & t \in [0,1] \\
 & 0 & = & \dot{m}(t) + q_0 p_2 u(t)^2, & t \in [0,1] \\
 & 0 & = & z(0), \quad 0 = v(0), \quad 0 = m(0) - 1 & \\
 & 0 & = & z(1) - 10, \quad 0 = v(1) - 1 & \\
 & 0 & \leq & 1 - u(t), & t \in [0,1] \\
 & 0 & \leq & 1 + u(t), & t \in [0,1],
\end{array} \tag{14.1}$$

where (14.1) is formulated based on a fix time horizon $[0,1]$ with $t \in [0,1]$, and q_0 is a control value representing the variable length of the true time horizon of the motion (cf., e.g., [Ger12, HSB12, Hat08]). The variables $x := (z,v,m)^\intercal$ are the differential states, where $z(t) \in \mathbb{R}$ describes the position, $v(t) \in \mathbb{R}$ the velocity and $m(t) \in \mathbb{R}$ the mass of the car. $p = (p_1,p_2)^\intercal$ are model parameters with the true value $p_1 = p_2 = 0.1$, and $u(t) \in \mathbb{R}$ is the control function. The objective function $-\gamma_1\, m(1) + \gamma_2\, q_0$ is a linear combination of Mayer terms, where the first term describes energy consumption (the mass at the end of the time horizon) and is weighted with γ_1 and the second term denotes travel time and is weighted with γ_2.

Problem (14.1) is solved with ParaOCP (cf. Chapter 9) using 20 multiple shooting intervals (on an equidistant time grid) and a termination tolerance for filterSQP of 1E-4. The Hessian of the Lagrangian of the discretized OCP (cf. Chapter 9) is computed using finite differences and we use a dense representation of the constraint Jacobian (due to the relatively small problem). We furthermore use a piecewise constant control approximation

	OCP (14.1)	**Hier. problem** (14.3)
# eq. sized multiple shooting intervals	20	20
# eq. sized control discr. intervals	20	20
control discretization type	piecew. const.	piecew. const.
# equid. measurement times	–	21 (= $n_T + 1$)
constraint Jacobian representation	dense	dense
termination tolerance filterSQP	1E-04	1E-04
upper-level Hessian type	fin. diff.	Gauß-Newton
lower-level Hessian type	–	efficient Hessian calculations
integration tolerance	1E-05	1E-05

Table 14.1.: Overview of the exact setting used in ParaOCP to solve the OCP (14.1) and the hierarchical problem (14.3).

on the multiple shooting grid. The exact setting used in ParaOCP for solving the OCP (14.1) is summarized in Table 14.1. The differential states and the control in the solution of (14.1) for $\gamma_1 = 0.5$, $\gamma_2 = 0.5$, and $p_1 = p_2 = 0.1$ are shown in Figure 14.1, the control value in the solution of (14.1) is $q_0 = 5.2663\text{E+}00$, and the objective value is 2.3906E+00.

We now generate measurements for the position z, the velocity v, and the mass m. Therefore, we add normally distributed random numbers with mean 0 and standard deviations $\sigma_{i1} = 0.05 \cdot 5,\ \sigma_{i2} = 0.05 \cdot 1.5,\ \sigma_{i3} = 0.05 \cdot 0.7\ \ \forall i \in 0, \dots, n_T$ (which corresponds to an error of approximately 5%) to the solution $x^*(t)$ of the OCP (14.1) for $\gamma_1 = 0.5$, $\gamma_2 = 0.5$, $p_1 = p_2 = 0.1$. We use simulated measurements throughout this chapter since they allow a better investigation of the proposed direct all-at-once approach for hierarchical dynamic optimization problems (cf. Chapter 5). The simulated measurements are denoted by

$$\eta = [(\eta_{0,1},\ \eta_{0,2},\ \eta_{0,3}),\ \dots\ ,(\eta_{n_T,1},\ \eta_{n_T,2},\ \eta_{n_T,3})]^\intercal \in \mathbb{R}^{(n_T+1)\times 3}, \tag{14.2}$$

with $n_T = 20$ and measurement times $\tau_i := \frac{i}{n_T}\ \forall i = 0, \dots, n_T$. The measurement values are illustrated in Figure 14.1. The parameter estimation problem with (14.1) and conditions on γ as constraints is given by:

$$\begin{array}{lllr}
\underset{\substack{x,u,q,\\ p,\gamma}}{\text{minimize}} & \multicolumn{2}{l}{\displaystyle\frac{1}{2}\sum_{i=0}^{n_T}\sum_{j=1}^{3}\frac{(x_{j-1}(\tau_i)-\eta_{ij})^2}{\sigma_{ij}^2}} & \\
\text{subject to} & \underset{x,u,q}{\text{minimize}} & -\gamma_1\, m(1) + \gamma_2\, q_0 & \\
& \text{subject to} & 0 = \dot{z}(t) - q_0 v(t), \qquad t \in [0,1] & \\
& & 0 = \dot{v}(t) - q_0(u(t) - p_1 v(t)^2)/m(t), \qquad t \in [0,1] & \\
& & 0 = \dot{m}(t) + q_0 p_2 u(t)^2, \qquad t \in [0,1] & (14.3)\\
& & 0 = z(0), \quad 0 = v(0), \quad 0 = m(0) - 1 & \\
& & 0 = z(1) - 10, \quad 0 = v(1) - 1 & \\
& & 0 \le 1 - u(t) \qquad t \in [0,1] & \\
& & 0 \le 1 + u(t) \qquad t \in [0,1] & \\
& 0 \le \gamma_1, \gamma_2 & & \\
& \gamma_2 = 0.5, & &
\end{array}$$

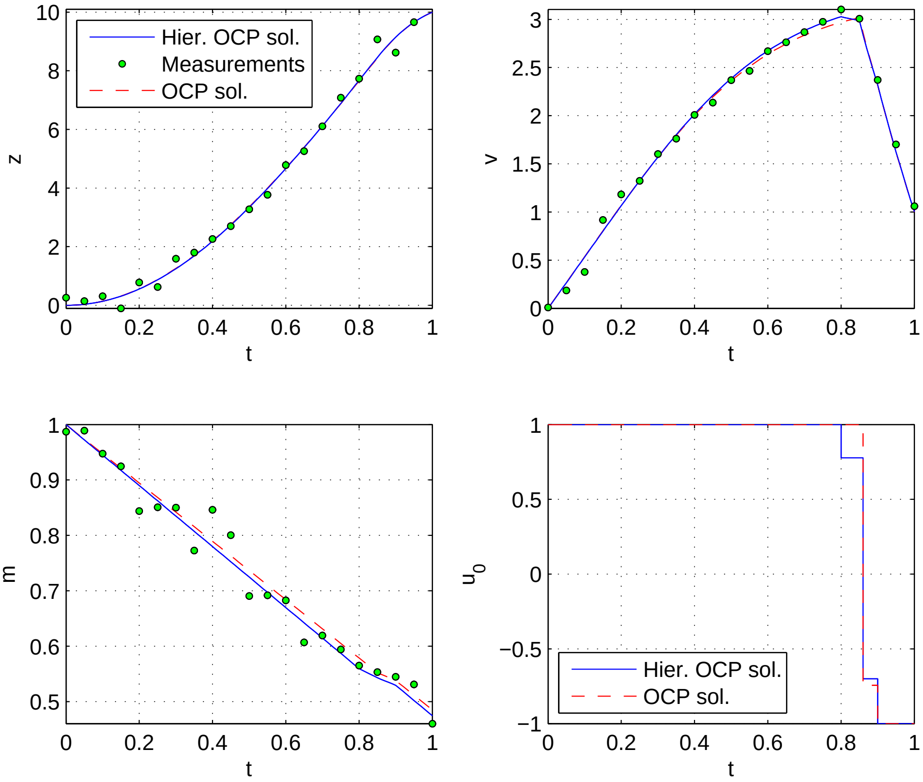


Figure 14.1.: Simulated measurements of the solution x of (14.1) (circles), differential states x and control u in the solution of the OCP (14.1) (dashed line) and in the solution of the hierarchical formulation (14.3) (solid line).

where γ_2 is set to a fix value (instead of requiring $\gamma_1 + \gamma_2 = 1$), which can be done if we know that the true value of γ_2 is not zero. We now solve the parameter estimation problem (14.3), which is constrained by (14.1) and additional constraints on γ_1 and γ_2 with the direct all-at-once approach described in Chapter 5 and implemented in ParaOCP (cf. Chapter 9). As initial values, we use the solution of the OCP (14.1) based on a guess of the unknowns γ_1, p_1 and p_2. Our guess is $\gamma_1 = 0.7$ and $p_1 = p_2 = 0.15$. The resulting values for the differential states, the control, the control value and the Lagrange multipliers are used as initial values for (14.3). The termination tolerance for filterSQP is 1E-04, we use a Gauß-Newton Hessian approximation and a dense representation of the constraint Jacobian (due to the relatively small problem). The exact setting in ParaOCP is given in Table 14.1. The differential states and the control in the solution of (14.3) are illustrated in Figure 14.1. The direct all-at-once approach described in Chapter 5 converges after 145 iterations with an objective value of 2.4714E+01 and $q_0 = 5.2265E{+}00$. The values of the upper-level variables of (14.3) in the solution are $\gamma_1 = 5.7131E\text{-}01$, $p_1 = 9.5561E\text{-}02$ and $p_2 = 1.0532E\text{-}01$, and the computation time is 8.8791E+00 s. The results of (14.3) solved with ParaOCP are summarized in Table 14.2.

	Solution of (14.3)
objective value	2.4714E+01
γ_1 (true value: 0.5)	5.7131E-01
p_1 (true value: 0.1)	9.5561E-02
p_2 (true value: 0.1)	1.0532E-01
q_0 (OCP: q_0 = 5.2663E+00)	5.2265E+00
comp. time in s	8.8791E+00
# iterations	145

Table 14.2.: Summary of the solution of (14.3) obtained with ParaOCP.

14.3. The Rayleigh example

In this section, we consider the performance of a Rayleigh problem formulated as an OCP from [MO02]. We consider a tunnel-diode oscillator as illustrated in Figure 14.2 with the induction L, the capacity C, the resistance R, the electric current I and the diode D. The differential state $x_0(t)$ denotes the electric current and the second differential state $x_1(t)$ denotes the derivative of the first differential state with respect to t. We furthermore have a scalar control function $u(t)$, which describes the transformation of the voltage V_0 at the generator (see [MO02] and references therein).

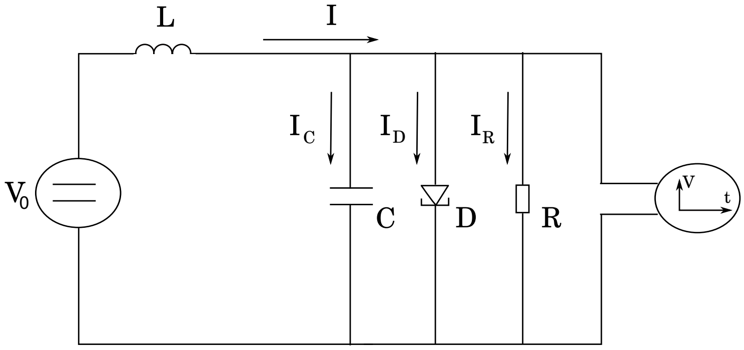


Figure 14.2.: Illustration of a tunnel-diode oscillator after [MO02].

We consider the following OCP based on [MO02]:

$$\begin{array}{rrcll}
\underset{x,u,q}{\text{minimize}} & \multicolumn{3}{l}{\gamma_1 q_0 + \gamma_2 q_0 \int_0^1 (u(t)^2 + x_0(t)^2)\,\mathrm{dt}} & \\
\text{subject to} & 0 & = & \dot{x}_0(t) - q_0 x_1(t), & t \in [0,1] \\
& 0 & = & \dot{x}_1(t) + q_0 \left(x_0(t) - x_1(t)(1.4 - 0.14 x_1(t)^2) - 4u(t)\right), & t \in [0,1] \\
& -5 & = & x_0(0), \quad -5 = x_1(0), & \\
& 0 & = & x_0(1), \quad 0 = x_1(1) & \\
& 0 & \leq & 1 - u(t), & t \in [0,1] \\
& 0 & \leq & 1 + u(t), & t \in [0,1].
\end{array} \tag{14.4}$$

As in the last section, (14.4) is formulated based on a fix time horizon $[0,1]$ with $t \in [0,1]$, and q_0 is a control value representing the variable length of the true time horizon of the

	OCP (14.4)	**Hier. problem** (14.6)
# eq. sized multiple shooting intervals	100	100
# eq. sized control discr. intervals	100	100
control discretization type	piecew. const.	piecew. const.
# equid. measurement times	–	101 (= $n_T + 1$)
constraint Jacobian representation	sparse	sparse
termination tolerance FILTERSQP	1E-04	1E-04
upper-level Hessian type	fin. diff.	Gauß-Newton
lower-level Hessian type	–	efficient Hessian calculations
integration tolerance	1E-05	1E-05

Table 14.3.: Overview of the exact setting used in PARAOCP to solve the OCP (14.4) and the hierarchical problem (14.6).

process. Problem (14.4) is of special interest since Maurer et al. analyze the solution of the OCP (14.4) in detail for $\gamma_1 = 0, \gamma_2 = 1$ and $\gamma_1 = 1/16, \gamma_2 = 1$ in [MO02] using the code BNDSCO (cf. [OG89]), and for the first one of these settings, the authors cannot verify second-order sufficient conditions, which indicates that this is just a stationary point and no local minimum. We are now interested in whether our direct-all-at-once approach, which identifies γ_1 and γ_2 in (14.4) based on simulated measurements using $\gamma_1 = 1/16, \gamma_2 = 1$, is attracted by $\gamma_1 = 0, \gamma_2 = 1$.

The algorithmic setting used in PARAOCP for solving (14.4) with $\gamma_1 = 1/16, \gamma_2 = 1$ is summarized in Table 14.3. The differential states x and the control u in the solution of (14.4) are shown in Figure 14.3. In the solution of (14.4), we have an objective value of 4.5004E-01 and $q_0 = 4.5361\text{E+}00$. We now simulate measurements for the differential states x adding normally distributed random numbers with mean 0 and standard deviations $\sigma_{i1} = 0.05 \cdot 3,\ \sigma_{i2} = 0.05 \cdot 2\ \ \forall i \in 0, \dots, n_T$ (which corresponds to an error of approximately 5%) with $n_T = 100$ to the solution $x^*(t)$ of the OCP (14.4) for $\gamma_1 = 1/16, \gamma_2 = 1$. The simulated measurements are denoted by

$$\eta = \left[(\eta_{0,1},\ \eta_{0,2})\,,\ \ \dots\ \ ,\ (\eta_{n_T,1},\ \eta_{n_T,2})\right]^\mathsf{T} \in \mathbb{R}^{(n_T+1)\times 2} \tag{14.5}$$

with measurement times $\tau_i := \frac{i}{n_T}\ \forall i = 0, \dots, n_T$, and are illustrated in Figure 14.3. The parameter estimation problem with (14.4) and conditions on γ as constraints is then given by:

$$\begin{array}{lll}
\underset{\substack{x,u,q,\\ \gamma}}{\text{minimize}} & \multicolumn{2}{l}{\displaystyle\frac{1}{2}\sum_{i=0}^{n_T}\sum_{j=1}^{2}\frac{(x_{j-1}(\tau_i)-\eta_{ij})^2}{\sigma_{ij}^2}} \\
\text{subject to} & \underset{x,u,q}{\text{minimize}} & \gamma_1 q_0 + \gamma_2 q_0 \int_0^1 (u(t)^2 + x_0(t)^2)\,\mathrm{dt} \\
& \text{subject to} & 0 = \dot{x}_0(t) - q_0 x_1(t), \hfill t \in [0,1] \\
& & 0 = \dot{x}_1(t) + q_0\left(x_0(t) - x_2(t)(1.4 - 0.14 x_2(t)^2) - 4u(t)\right), \ t \in [0,1] \\
& & 0 = x_0(0), \quad -5 = x_1(0), \\
& & 0 = x_0(1), \quad 0 = x_1(1) \\
& & 0 \le 1 - u \hfill t \in [0,1] \\
& & 0 \le 1 + u \hfill t \in [0,1] \\
& 0 \le \gamma_1, \gamma_2, & \\
& \gamma_2 = 1, &
\end{array} \tag{14.6}$$

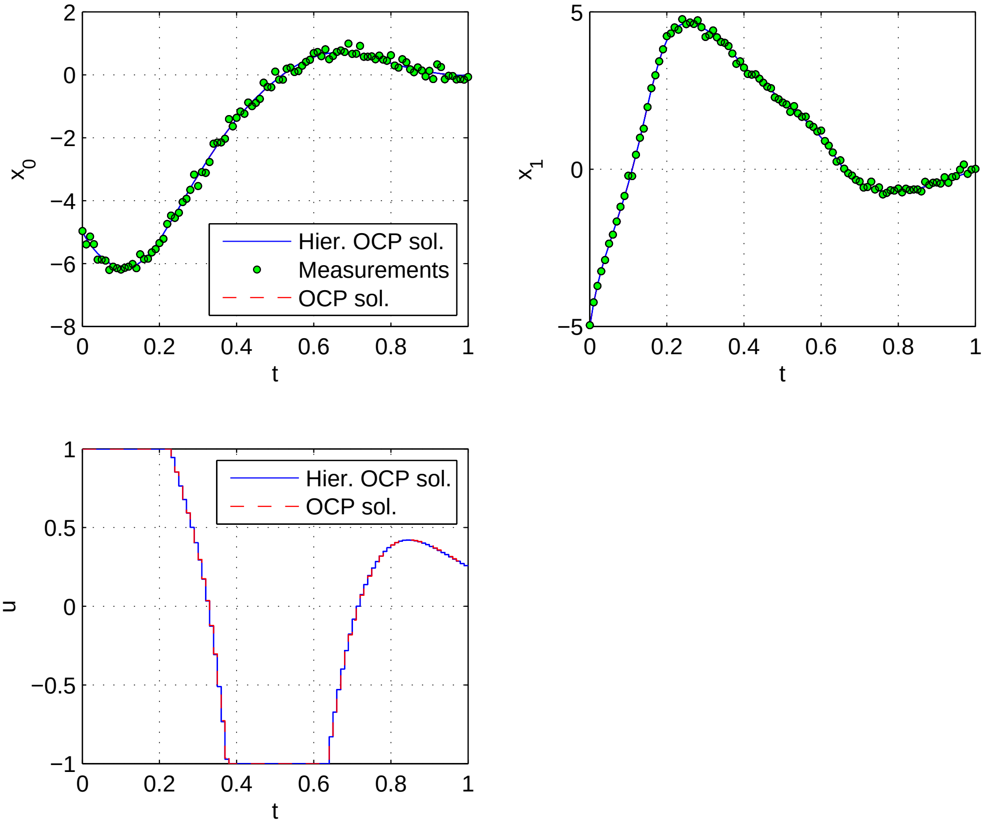


Figure 14.3.: Simulated measurements of the solution x of (14.4) (circles), differential states x and control u in the solution of the OCP (14.4) (dashed line) and in the solution of the hierarchical formulation (14.6) (solid line).

where γ_2 is set to a fix value (instead of requiring $\gamma_1 + \gamma_2 = 1$) since we know that the true value of γ_2 is not zero. We now solve the hierarchical problem (14.6) with ParaOCP. As initial values, we use the solution of the OCP (14.4) based on a guess of the unknown γ_1. As mentioned above, Maurer et al. report in [MO02] a solution of (14.4) found by BNDSCO for $\gamma_1 = 0, \gamma_2 = 1$, which might be no local minimum. The measurements used in (14.6) are generated based on $\gamma_1 = 1/16, \gamma_2 = 1$, and we now choose our initial guess close to 0 using $\gamma_1 = 0.01$ to check if our method described in Chapter 5 is attracted by the solution of (14.6) corresponding to $\gamma_1 = 0, \gamma_2 = 1$, which might be no local minimum of the lower-level problem. The OCP (14.4) is solved for $\gamma_1 = 0.01, \gamma_2 = 1$ and the resulting values for the differential states, the control, the control value and the Lagrange multipliers are used as initial values for (14.6). The exact setting in ParaOCP for solving (14.6) is summarized in Table 14.3. The differential states and the control in the solution of (14.6) are illustrated in Figure 14.3. The direct all-at-once approach described in Chapter 5 converges after 45 iterations. Further details of the solution of (14.6) are provided in Table 14.4.

We furthermore observe that the direct-all-at-once approach implemented in ParaOCP

	Solution of (14.3)
objective value	9.2207E+01
γ_1 (true value: 6.25E-02)	6.0661E-02
q_0 (OCP: $q_0 = 4.5361\text{E+}00$)	4.5392E+00
comp. time in s	6.0839E+00
# iterations	45

Table 14.4.: Summary of the solution of (14.6) obtained with ParaOCP.

applied to (14.6) is not attracted to the solution of (14.6) corresponding to $\gamma_1 = 0$, even if we use initial values for the parameters, the differential states, the control, the control value and the Lagrange multipliers corresponding to a solution of (14.4) for $\gamma_1 = 0.01, \gamma_2 = 1$.

14.4. The Reeds-Shepp car example

In this section, we consider the well-known Reeds-Shepp model, which is discussed in many articles as, e.g, in [DS13] and originates from the article *Optimal paths for a car that goes both forwards and backwards* by J. A. Reeds and L. A. Shepp [RS90]. Reeds et al. describe the motion of a car in a plane with three degrees of freedom (DOFs): the position at time t in a plane is described by $(x_0(t), x_1(t))$, and the orientation of the car at time t is given by the angle $x_2(t)$ as illustrated in Figure 14.4.

The Reeds-Shepp car is of particular interest since it is the basis of the model of the dynamics of human overall locomotion path generation in [KML10], which is used to identify the objectives of human path generation via inverse optimal control with a bilevel approach (cf. Chapter 4).

We consider the following OCP:

$$\begin{array}{lllll}
\underset{x,u,q}{\text{minimize}} & \multicolumn{3}{l}{\gamma_1\, q_0(t) + \gamma_2 q_0 \int_0^1 u_0(t)^2 \,\mathrm{dt} + \gamma_3 q_0 \int_0^1 u_1(t)^2 \,\mathrm{dt}} & \\
\text{subject to} & 0 & = & \dot{x}_0(t) - q_0 \cos(x_2(t)) u_0(t), & t \in [0,1] \\
 & 0 & = & \dot{x}_1(t) - q_0 \sin(x_2(t)) u_0(t), & t \in [0,1] \\
 & 0 & = & \dot{x}_2(t) - q_0 u_1(t), & t \in [0,1] \\
 & 0 & = & x_0(0), \quad 0 = x_1(0), \quad 0 = x_2(0) - \frac{\pi}{2} & \\
 & 0 & = & x_0(1) - 1, \quad 0 = x_1(1) - 1, \quad 0 = x_2(1) - 1 & \\
 & 0 & \leq & 1 - u_0(t), \quad 0 \leq 1 - u_1(t), & t \in [0,1] \\
 & 0 & \leq & 1 + u_0(t), \quad 0 \leq 1 + u_1(t), & t \in [0,1],
\end{array} \tag{14.7}$$

with the controls $u_0(t)$, $u_1(t)$ and the control value q_0 describing the duration of the process. The objective of (14.7) is a weighted sum of three subcriteria describing the duration of the process, the integral over the squared control u_0 and the integral over the squared control u_1.

Problem (14.7) is solved with ParaOCP using 20 equally sized multiple shooting intervals and a termination tolerance for filterSQP of 1E-4. Further details of the algorithmic setting in ParaOCP are given in Table 14.5.

The differential states and the controls in the solution of (14.7) for $\gamma_1 = 1/3$, $\gamma_2 = 1/3$,

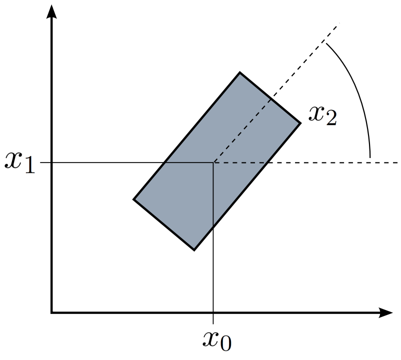


Figure 14.4.: Illustration of the Reeds-Shepp car in a plane and it's degrees of freedom (cf. [DS13, RS90]).

and $\gamma_3 = 1/3$ are shown in Figure 14.5, the control value in the solution of (14.7) is $q_0 = 2.1140\text{E+}00$, and the objective value is 1.4247E+00.

We generate measurements for the differential states x adding normally distributed random numbers with mean 0 and standard deviations $\sigma_{i1} = 0.05 \cdot 0.6$, $\sigma_{i2} = 0.05 \cdot 0.6$, $\sigma_{i3} = 0.05 \cdot 1 \;\; \forall i \in 0, \dots, n_T$ (which corresponds to an error of approximately 5%) to the solution $x^*(t)$ of the OCP (14.7) for $\gamma_1 = 1/3$, $\gamma_2 = 1/3$, and $\gamma_3 = 1/3$ and denote the measurement values by

$$\eta = \left[(\eta_{0,1},\ \eta_{0,2},\ \eta_{0,3})^\intercal,\ \dots\ ,(\eta_{n_T,1},\ \eta_{n_T,2},\ \eta_{n_T,3})^\intercal \right]^\intercal \in \mathbb{R}^{(n_T+1)\times 3}, \tag{14.8}$$

with $n_T = 20$ and measurement times $\tau_i := \frac{i}{n_T} \; \forall i = 0, \dots, n_T$. The measurement values are illustrated in Figure 14.5. The parameter estimation problem with (14.7) and conditions on γ as constraints is given by:

$$\begin{array}{lll}
\underset{\substack{x,u,q,\\ \gamma}}{\text{minimize}} & \multicolumn{2}{l}{\frac{1}{2}\sum_{i=0}^{n_T}\sum_{j=1}^{3}\frac{(x_{j-1}(\tau_i)-\eta_{ij})^2}{\sigma_{ij}^2}} \\
\text{subject to} & \underset{x,u,q}{\text{minimize}} & \gamma_1\, q_0(t) + \gamma_2 q_0 \int_0^1 u_0(t)^2 \,\mathrm{dt} + \gamma_3 q_0 \int_0^1 u_1(t)^2 \,\mathrm{dt} \\
& \text{subject to} & 0 = \dot{x}_0(t) - q_0 \cos(x_2(t)) u_0(t), \qquad t \in [0,1] \\
& & 0 = \dot{x}_1(t) - q_0 \sin(x_2(t)) u_0(t), \qquad t \in [0,1] \\
& & 0 = \dot{x}_2(t) - q_0 u_1(t), \qquad t \in [0,1] \\
& & 0 = x_0(0), \quad 0 = x_1(0), \quad 0 = x_2(0) - \frac{\pi}{2} \\
& & 0 = x_0(1) - 1, \quad 0 = x_1(1) - 1, \quad 0 = x_2(1) - 1 \\
& & 0 \le 1 - u_0(t), \quad 0 \le 1 - u_1(t), \qquad t \in [0,1] \\
& & 0 \le 1 + u_0(t), \quad 0 \le 1 + u_1(t), \qquad t \in [0,1] \\
& 0 \le \gamma_1, \gamma_2, \gamma_3 & \\
& \gamma_3 = 1/3, &
\end{array} \tag{14.9}$$

where γ_3 is set to a fix value (instead of requiring $\gamma_1 + \gamma_2 + \gamma_3 = 1$) since we know that $\gamma_3 \neq 0$.

We now solve the parameter estimation problem (14.9) with ParaOCP. As in the previous sections, we use the solution of the OCP (14.7) based on a guess of the unknowns γ_1, γ_2 and γ_3 as initial values for solving (14.9). Our guess is $\gamma_1 = 1.$ and $\gamma_2 = 0.1$. The resulting values for the differential states, the controls, the control value and the Lagrange

	OCP (14.7)	**Hier. problem** (14.9)
# eq. sized multiple shooting intervals	20	20
# eq. sized control discr. intervals	20	20
control discretization type	piecew. const.	piecew. const.
# equid. measurement times	–	21 ($= n_T + 1$)
constraint Jacobian representation	sparse	sparse
termination tolerance FILTERSQP	1E-04	1E-04
upper-level Hessian type	fin. diff.	Gauß-Newton
lower-level Hessian type	–	efficient Hessian calculations
integration tolerance	1E-05	1E-05

Table 14.5.: Overview of the exact setting used in ParaOCP to solve the OCP (14.7) and the hierarchical problem (14.9).

multipliers are used as initial values for (14.9). Further details about the algorithmic setting in ParaOCP are provided in Table 14.5. The differential states and the controls in the solution of (14.9) are illustrated in Figure 14.5. The direct all-at-once approach described in Chapter 5 converges after 102 iterations with an objective value of 2.6106E+01 and $q_0 = 2.2101\text{E}{+}00$. The values of the upper-level variables of (14.9) in the solution and further details are given in Table 14.6.

	Solution of (14.9)
objective value	2.6106E+01
γ_1 (true value: 1/3)	3.0932E-01
γ_1 (true value: 1/3)	3.4398E-01
q_0 (OCP: $q_0 = 2.1140\text{E}{+}00$)	2.2101E+00
comp. time in s	2.3945E+00
# iterations	102

Table 14.6.: Summary of the solution of (14.9) obtained with ParaOCP.

14.5. The polar robot example

The polar robot we consider in this section is modeled as a simple multibody system and has already been described in Section 13.4. The multibody-system model of the polar robot is based on the models analyzed in [Ste87, Mom01]. For the sake of completeness, we briefly recall the problem formulation. The robot we consider has three DOFs, $(\varphi_1, \varphi_2, r)$, which are illustrated in Figure 13.4 and performs optimal point-to-point trajectories. In total, we have six differential states

$$\begin{aligned} x(t) &:= (x_0(t), x_1(t), x_2(t), x_3(t), x_4(t), x_5(t))^\intercal \\ &= (\varphi_1(t), \varphi_2(t), r(t), \dot{\varphi}_1(t), \dot{\varphi}_2(t), \dot{r}(t))^\intercal \end{aligned} \tag{14.10}$$

	OCP (14.11)	**Hier. problem** (14.14)
# eq. sized multiple shooting intervals	6	6
# eq. sized control discr. intervals	6	6
control discretization type	piecew. const.	piecew. const.
# equid. measurement times	–	7 $(= n_T + 1)$
constraint Jacobian representation	dense	dense
termination tolerance FILTERSQP	1E-04	1E-04
upper-level Hessian type	fin. diff.	Gauß-Newton
lower-level Hessian type	–	efficient Hessian calculations
integration tolerance	1E-05	1E-05

Table 14.7.: Overview of the exact setting used in PARAOCP to solve the OCP (14.11) and the hierarchical problem (14.14).

and three controls $u(t) := (u_0(t), u_1(t), u_2(t))^\intercal$ describing the torques acting on $(\varphi_1, \varphi_2, r)$, respectively. Optimal point-to-point trajectories of the robot can be described as the solution of the following OCP:

$$\begin{array}{rrcll}
\underset{x,u,q}{\text{minimize}} & \multicolumn{3}{l}{\gamma_1\, q_0 \int_0^1 u_2(t)^2\,\mathrm{dt} + \gamma_2\, q_0} & \\
\text{subject to} & \ddot{x}(t) & = & f(x(t), u(t), q), & t \in [0,1] \\
& x_0(0) & = & 0, \quad x_1(0) = 0, \quad x_2(0) = 0.7 & \\
& x_3(0) & = & 0, \quad x_4(0) = 0, \quad x_5(0) = 0 & \\
& x_0(1) & = & 0, \quad x_1(1) = 0, \quad x_2(1) = -0.1 & \\
& x_3(1) & = & 0, \quad x_4(1) = 0, \quad x_5(1) = 0 & \\
& -1 & \leq & x_0(t) \leq 1, \quad -1 \leq x_1(t) \leq 1, \quad -0.5 \leq x_2(t) \leq 0.7, & t \in [0,1] \\
& -5 & \leq & x_3(t) \leq 5, \quad -5 \leq x_4(t) \leq 5, \quad -6 \leq x_5(t) \leq 6, & t \in [0,1] \\
& -1000 & \leq & u_0(t) \leq 1000, & t \in [0,1] \\
& -1500 & \leq & u_1(t) \leq 1500, & t \in [0,1] \\
& -1000 & \leq & u_2(t) \leq 1000, & t \in [0,1].
\end{array} \tag{14.11}$$

As in the last sections, (14.11) is formulated based on a fix time horizon $[0,1]$, and q_0 is a control value representing the variable length of the true time horizon of the motion. The differential equation $\ddot{x}(t) = f(x(t), u(t), q)$ in (14.11) is in detail given by

$$\begin{aligned}
\ddot{\varphi}_1 &= q_0 \tfrac{(u_0+((18.5+40r^2+10(r+L)^2-0.12)\sin(2\varphi_2)\dot{\varphi}_2-2(40r+10(r+L)\dot{r}\cos(\varphi_2)^4))\dot{\varphi}_1}{10+(18.5+40r^2 10(r+L)^2\cos(\varphi_2)^2+0.12\cos(\varphi_2)^2)} \\
\ddot{\varphi}_2 &= q_0 \tfrac{u_1+(0.12-18.5-40r^2-10.(r+L)^2)0.5\sin(2\varphi_2)\dot{\varphi}_1^2-(40r+10(r+L))(g\cos(\varphi_2)+2\dot{r}\dot{\varphi}_2}{18.5+40r^2+10(r+L)^2} \\
\ddot{r} &= q_0 \tfrac{(u_2+(40r+10(r+L))(\cos(\varphi_2)^2\dot{\varphi}_1^2+\dot{\varphi}_2^2)-50g\sin(\varphi_2))}{50}
\end{aligned} \tag{14.12}$$

where the argument t is skipped for a compact presentation. Problem (14.11) is solved with PARAOCP using 6 equally sized multiple shooting intervals, with $\gamma_1 = 2.$, $\gamma_2 = 1\text{E}{+}4 \cdot 1.$, $L = 0.75$ and a termination tolerance for FILTERSQP of 1E-04. Further details of the algorithmic setting in PARAOCP are given in Table 14.7. The differential states and the controls in the solution of (14.11) for $\gamma_1 = 2.$, $\gamma_2 = \text{E} + 04 \cdot 1.$, and $L = 0.75$ are shown in Figure 13.7, the control value in the solution of (14.11) is $q_0 = 1.2595\text{E}{+}00$, and the objective value is 1.2964E+04. The motion of the polar robot in the solution of (14.11) is shown in Figure 14.6.

We generate measurements for the differential states x adding normally distributed random numbers with mean 0 and standard deviations $\sigma_{i1} = 0.05 \cdot 0.2$, $\sigma_{i2} = 0.05 \cdot 0.2$, $\sigma_{i3} = 0.05 \cdot 0.2 \;\; \forall i \in 0, \dots, n_T$ and $\sigma_{i4} = 0.05 \cdot 1.$, $\sigma_{i5} = 0.05 \cdot 1.$, $\sigma_{i6} = 0.05 \cdot 1.$ $\forall i \in 0, \dots, n_T$ to the solution $x^*(t)$ of the OCP (14.11) for $\gamma_1 = 2.$, $\gamma_2 = 1\text{E}{+}4$, and $L = 0.75$ and denote the measurement values by

$$\begin{aligned} \eta = [(&\eta_{0,1},\ \eta_{0,2},\ \eta_{0,3}, \eta_{0,4},\ \eta_{0,5},\ \eta_{0,6})\,,\ \dots\ , \\ &(\eta_{n_T,1},\ \eta_{n_T,2},\ \eta_{n_T,3}, \eta_{n_T,4},\ \eta_{n_T,5},\ \eta_{n_T,6})]^\intercal, \end{aligned} \tag{14.13}$$

with $\eta \in \mathbb{R}^{(n_T+1)\times 3}$, $n_T = 6$ and measurement times $\tau_i := \frac{i}{n_T}$ $\forall i = 0, \dots, n_T$. The measurement values are illustrated in Figure 13.7. The parameter estimation problem with (14.11) and conditions on γ as constraints is given by:

$$\begin{array}{lllll}
\underset{\substack{x,u,q,\\ L,\gamma}}{\text{minimize}} & \multicolumn{4}{l}{\displaystyle\frac{1}{2}\sum_{i=0}^{n_T}\sum_{j=1}^{6}\frac{(x_{j-1}(\tau_i)-\eta_{ij})^2}{\sigma_{ij}^2}} \\
\text{subject to} & \underset{x,u,q}{\text{minimize}} & \multicolumn{3}{l}{\gamma_1 \int_0^1 u_2(t)^2 \,\mathrm{dt} + \gamma_2\, q_0} \\
& \text{subject to} & \ddot{x}(t) = & f(x(t),u(t),q), & t \in [0,1] \\
& & x_0(0) = & 0, \quad x_1(0) = 0, \quad x_2(0) = 0.7 & \\
& & x_3(0) = & 0, \quad x_4(0) = 0, \quad x_5(0) = 0 & \\
& & x_0(1) = & 0, \quad x_1(1) = 0, \quad x_2(1) = -0.17 & \\
& & x_3(1) = & 0, \quad x_4(1) = 0, \quad x_5(1) = 0 & \\
& & -1 \leq & x_0(t) \leq 1, \quad -1 \leq x_1(t) \leq 1, & \forall t \in [0,1] \\
& & -0.5 \leq & x_2(t) \leq 0.7, \quad -5 \leq x_3(t) \leq 5, & \forall t \in [0,1] \\
& & -5 \leq & x_4(t) \leq 5, \quad -6 \leq x_5(t) \leq 6, & \forall t \in [0,1] \\
& & -1000 \leq & u_0(t) \leq 1000, & \forall t \in [0,1] \\
& & -1500 \leq & u_1(t) \leq 1500, & \forall t \in [0,1] \\
& & -1000 \leq & u_2(t) \leq 1000, & \forall t \in [0,1] \\
& 0 \leq \gamma_1, \gamma_2 & & & \\
& \gamma_2 = 1\text{E}{+}04, & & &
\end{array} \tag{14.14}$$

where γ_2 is set to a fix value since we know that the true value of γ_2 is not zero.

We now solve the hierarchical problem (14.14) with ParaOCP. As initial values, we use the solution of the OCP (14.11) based on a guess of the unknowns γ_1 and L. The OCP (14.11) is solved for $\gamma_1 = 2.3$, $\gamma_2 = 1\text{E}{+}04$, $L = 1.5$, and the resulting values for the differential states, the controls, the control value and the Lagrange multipliers are used as initial values for (14.14). The exact setting in ParaOCP for solving (14.14) is summarized in Table 14.7. The differential states and the controls in the solution of (14.14) are illustrated in Figure 13.7. The direct all-at-once approach described in Chapter 5 converges after 649 iterations. Further details of the solution of (14.14) are provided in Table 14.8.

14.6. Bilevel approaches applied to the Reeds-Shepp car example

In the last section of this chapter, we discuss the numerical performance of the bilevel approaches described in Chapter 7 for identifying the objective of an optimal motion of the

	Solution of (14.14)
objective value	1.0522E+01
γ_1 (true value: 2.)	2.2815E+00
L (true value: 0.75)	7.3888E-01
q_0 (OCP: $q_0 = 1.2595\text{E}{+}00$)	1.2698E+00
comp. time in s	4.3294E+01
# iterations	649

Table 14.8.: Summary of the solution of (14.14) obtained with ParaOCP.

Reeds-Shepp car analyzed in Section 14.4. We first show the performance of the derivative-free optimization (DFO) technique explained in Section 7.2 and then discuss the results of the trust region bundle approach described in Section 7.3. The performance of both bilevel approaches is compared to the results of the direct all-at-once approach from Chapter 5.

We consider the hierarchical dynamic optimization problem given by

$$\begin{array}{lll}
\underset{\substack{x,u,q,\\ \gamma}}{\text{minimize}} & \multicolumn{2}{l}{\displaystyle\sum_{i=0}^{n_T}\sum_{j=1}^{3}\frac{\left(x_{j-1}(\tau_i)-\eta_{ij}^x\right)^2}{\sigma_{ij}^{x\,2}}+\sum_{i=0}^{n_T-1}\sum_{j=1}^{2}\frac{\left(u_{j-1}(\tau_i)-\eta_{ij}^u\right)^2}{\sigma_{ij}^{u\,2}}} \\
\text{subject to} & \underset{x,u,q}{\text{minimize}} & \gamma_1\, q_0(t)+\gamma_2 q_0\int_0^1 u_0(t)^2\,\mathrm{dt}+\gamma_3 q_0\int_0^1 u_1(t)^2\,\mathrm{dt} \\
& \text{subject to} & 0=\dot{x}_0(t)-q_0\cos(x_2(t))u_0(t), \qquad t\in[0,1] \\
& & 0=\dot{x}_1(t)-q_0\sin(x_2(t))u_0(t), \qquad t\in[0,1] \\
& & 0=\dot{x}_2(t)-q_0u_1(t), \qquad t\in[0,1] \\
& & 0=x_0(0),\quad 0=x_1(0),\quad 0=x_2(0)-\frac{\pi}{2} \\
& & 0=x_0(1)-1,\quad 0=x_1(1)-1,\quad 0=x_2(1)-1 \\
& & 0\le 1-u_0(t),\quad 0\le 1-u_1(t), \qquad t\in[0,1] \\
& & 0\le 1+u_0(t),\quad 0\le 1+u_1(t), \qquad t\in[0,1] \\
& 0\le\gamma_1,\gamma_2,\gamma_3 & \\
& \gamma_3=1/3, &
\end{array} \tag{14.15}$$

which is closely related to the problem described in Section 14.4. In this section, we simulate measurements for the differential states x and the controls u by adding normally distributed random numbers with mean 0 and standard deviations $\sigma_{i1}^x = 0.05\cdot 0.6$, $\sigma_{i2}^x = 0.05\cdot 0.6$, $\sigma_{i3}^x = 0.05\cdot 1\ \ \forall i \in 0,\dots,n_T$ to the solution $x^*(t)$ of the OCP (14.7) for $\gamma_1 = 1/3$, $\gamma_2 = 1/3$, and $\gamma_3 = 1/3$ and by adding normally distributed random numbers with mean 0 and standard deviations $\sigma_{i1}^u = 0.05\cdot 0.5$, $\sigma_{i2}^u = 0.05\cdot 0.2\ \ \forall i \in 0,\dots,n_T-1$ to the solution $u^*(t)$ of the OCP (14.7) for $\gamma_1 = 1/3$, $\gamma_2 = 1/3$, and $\gamma_3 = 1/3$. The measurement values are denoted by

$$\eta^x = \left[\left(\eta_{0,1}^x,\ \eta_{0,2}^x,\ \eta_{0,3}^x\right),\ \dots\ ,\ \left(\eta_{n_T,1}^x,\ \eta_{n_T,2}^x,\ \eta_{n_T,3}^x\right)\right]^\intercal \in \mathbb{R}^{(n_T+1)\times 3}, \tag{14.16}$$

and

$$\eta^u = \left[\left(\eta_{0,1}^u,\ \eta_{0,2}^u\right),\ \dots\ ,\ \left(\eta_{n_T,1}^u,\ \eta_{n_T,2}^u\right)\right]^\intercal \in \mathbb{R}^{n_T\times 2} \tag{14.17}$$

with $n_T = 20$ and measurement times $\tau_i := \frac{i}{n_T}\ \forall i = 0,\dots,n_T$. The reason for including measurements of the control u is to show that in principle, both bilevel approaches work. If we do not add measurements of the control, both bilevel approaches have serious trouble with finding a solution, which is discussed at the end of this section.

Constants in POUNDerS **for solving the upper-level problem of** (14.15)	**Value**
# equid. measurement times	21
maximum number of function evaluations	1000
tolerance for the ℓ_2-norm of the model gradient	1E-04
positive trust region radius	10
number of objective function values known in advance	0

Table 14.9.: Overview of the algorithmic setting used in POUNDerS for solving the upper-level problem of (14.15).

The derivative-free optimization approach

The DFO technique by Wild et al. [Wil08, KLM+10, MW09] explained in Section 7.2 is implemented in the software package POUNDerS (Practical Optimization Using No DERivatives with Squares) and used to solve the upper-level problem of (14.15). In each iteration of the DFO method, the lower-level OCP of (14.15) is solved with ParaOCP described in Chapter 9. The algorithmic setting used in POUNDerS and in ParaOCP is summarized in Table 14.9 and 14.10, respectively. We choose $(1., 0.1)$ as initial values for the upper-level variables (γ_1,γ_2). As initial values for the differential states x, the controls u and the control value q_0 in ParaOCP for solving the lower-level OCP of (14.15) in each iteration of the DFO method, we use the solution of the lower-level OCP of (14.15) for $\gamma_1 = 1$ and $\gamma_2 = 0.1$. The differential states and the controls in the solution of (14.15) obtained using the DFO technique for solving hierarchical dynamic optimization problems are shown in Figure 14.8.

Constants in ParaOCP **for solving the lower-level OCP of** (14.15)	**Value**
# eq. sized multiple shooting intervals	20
# eq. sized control discr. intervals	20
control discretization type	piecew. const.
constraint Jacobian representation	sparse
termination tolerance filterSQP	1E-05
Hessian type	finite differences
integration tolerance	1E-06

Table 14.10.: Overview of the algorithmic setting used in ParaOCP for solving the lower-level OCP in (14.15).

Furthermore, Figure 14.8 compares the result of the DFO technique applied to (14.15) to the result of (14.15) obtained with the direct all-at-once approach from Chapter 5 using ParaOCP with the settings summarized in Table 14.5 with one change: we use the same integration tolerance as in the DFO approach, which is 1E-06 (DAAO in the legend of

	DFO method	**Direct all-at-once approach**
objective value	4.3929E-01	4.3929E-01
γ_1 (true value: 1/3)	3.3503E-01	3.3404E-01
γ_2 (true value: 1/3)	3.3265E-01	3.3570E-01
q_0 (OCP: $q_0 = 2.1140\text{E}{+}00$)	2.1158E-00	2.1157E+00
# iterations	53	123
comp. time in s	6.9023E+01	3.2799E+00

Table 14.11.: Overview of the solution of (14.15) obtained with the DFO method from Section 7.2 and the direct all-at-once approach from Chapter 5.

Figure 14.8 stands for the direct all-at-once approach). As initial values for the differential states x, the controls u, the control value q_0 and the Lagrange multipliers of the discretized lower-level OCP we us the solution of (14.7) for $\gamma_1 = 1.$ and $\gamma_2 = 0.1$. As reference, Figure 14.8 also shows the solution of the OCP (14.7) for $\gamma_1 = 1/3$, $\gamma_2 = 1/3$ and $\gamma_3 = 1/3$.

Table 14.11 contains the resulting upper-level variables, the control value, the upper-level objective value and the computation time of the DFO method and the direct all-at-once approach in the solution of (14.15). The results presented above serve as proof of concept that the DFO approach for solving hierarchical dynamic optimization problems stated in Section 7.2 works in general. However, we observe the following:

- The performance of the DFO method is very sensitive with respect to the choice of the initial trust region radius.

- If we do not use measurements for the controls in (14.15) and choose, e.g., $(0.1, 0.1)$, $(1, 1)$ or $(10, 10)$ as initial values for (γ_1, γ_2), the DFO method converges to a point close to the initial value after often more than 100s. If we use, e.g., $(10, 10)$ as initial value for (γ_1, γ_2) (and $(1/3, 1/3)$ is the true value) and just measurements of the differential states with the standard deviations stated above, the DFO method converges to $(9.3950\text{E}{+}00, 9.1035\text{E}{+}00)$ after 1.4108E+02s with an upper-level objective value of 5.2201E+01. With the direct all-at-once approach from Chapter 5, we obtain $\gamma_1 = 3.1982\text{E-}01$ and $\gamma_2 = 3.3463\text{E-}01$, also without using measurements of the controls and with the same initial value $(10, 10)$ for (γ_1, γ_2) in 3.8186E+00s (and the remaining setting as above).

- For (14.15), which is a relatively small hierarchical dynamic optimization problem with only two upper-level variables, the computation time of the DFO method is already around 20 times larger than the one of direct all-at-once approach. We note that the majority of computation time is spent to solve the lower-level OCP, which is done with the C/C++ code ParaOCP. Hence, we do not expect significant changes in the computation times if POUNDerS would have been implemented in C/C++ instead of Matlab.

- The DFO method seems to be less robust with respect to the initial values of the upper-level variables and with respect to the amount/quality of measurement data, which might be due to the lack of derivative information.

Constants in the trust region bundle method for solving the upper-level problem of (14.15)	**Value**
# equid. measurement times	21
maximum number of subgradients (bundle size, $J_{\max}$ in Section 7.3)	5
termination tolerance (ϵ in Section 7.3)	1E-04
initial trust region radius (t_0 in Section 7.3)	0.2
parameter to update trust region (ρ in Section 7.3)	0.3
m_1, m_2, m_3, ν from Section 7.3	$0.1, 0.2, 0.9, 0.1$

Table 14.12.: Overview of constants used in the trust region bundle method for solving the upper-level problem of (14.15).

- However, the general advantage of bilevel approaches is that they avoid the complementarity constraints that has to be treated in the framework of the direct all-at-once approach if inequality constraints are present.
- Another advantage of the DFO approach is that it allows to use a black box optimal control model, i.e. a black box OCP solver can be used to solve the lower-level OCP. However, a reliable OCP solver is crucial for the success of this bilevel approach.

To sum up, the disadvantages of using a DFO-based bilevel approach outweigh the advantages by far.

The trust region bundle approach

We now solve the hierarchical dynamic optimization problem (14.15) using the simulated measurements (14.16) and (14.17) with the trust region bundle technique stated in Section 7.3. We have implemented the bundle idea of Zowe and Schramm [SZ92] with some extensions described in Section 7.3 in Matlab. The main constants used within the bundle method are shown in Table 14.12. In each iteration of the bundle method applied to (14.15), the lower-level OCP is solved with ParaOCP using the settings summarized in Table 14.10. For the upper-level variables γ_1 and γ_2, we use $\gamma_1 = 1.$ and $\gamma_2 = 0.1$ as initial values. As initial values for the differential states x, the controls u and the control value q_0 in ParaOCP for solving the lower-level OCP of (14.15) in each iteration of the bundle method, we use the solution of the lower-level OCP of (14.15) for $\gamma_1 = 1.$ and $\gamma_2 = 0.1$. The differential states and the controls in the solution of (14.15) obtained using the bundle approach are shown in Figure 14.8.

Furthermore, Figure 14.8 compares the result of the bundle approach applied to (14.15) to the solution of (14.15) obtained with the direct all-at-once approach from Chapter 5 using ParaOCP with the settings summarized in Table 14.5. As already described above, we use the differential states x, the controls u, the control value q_0 and the Lagrange multipliers of the discretized lower-level OCP in the solution of (14.7) for $\gamma_1 = 1.$ and $\gamma_2 = 0.1$ as initial values for the direct all-at-once approach. Table 14.13 shows the upper-level variables, the control value, the upper-level objective value and the computation time of the bundle method and the direct all-at-once approach in the solution of (14.15). We note that the majority of computation time is spent to solve the lower-level OCP, which is done with the

	Bundle method	**Direct all-at-once approach**
objective value	4.3929E-01	4.3929E-01
γ_1 (true value: 1/3)	3.3509E-01	3.3404E-01
γ_1 (true value: 1/3)	3.3265E-01	3.3570E-01
q_0 (OCP: q_0 = 2.1140E+00)	2.1158E-00	2.1157E+00
# iterations	30	123
comp. time in s	5.0612E+02	3.2799E+00

Table 14.13.: Overview of the solution of (14.15) obtained with the bundle method from Section 7.3 and the direct all-at-once approach from Chapter 5.

C/C++ code ParaOCP. Hence, we do not expect significant changes in the computation times if the bundle method would have been implemented in C/C++ instead of Matlab. The results for the bundle method applied to (14.15) presented above serve as proof of concept that the bundle approach for solving hierarchical dynamic optimization problems stated in Section 7.3 works in general. We observe the following:

- Our implementation of the bundle method from [SZ92] is very sensitive with respect to the algorithmic setting (like, e.g., the choice of the initial trust region).

- The bundle idea presented in [SZ92] does not include bounds on the variables. This is not a problem for the results presented in this section, since we start relatively close to the solution and γ_1 and γ_2 stayed positive throughout the iterations, but for a more detailed study with initial values far away from the solution, one would have to include bounds in the algorithm presented in Section 7.3.

- The bundle method seems to be very sensitive with respect to the measurement data. If we do not include measurements for the controls, the bundle method converges to a point very close to the initial value, whereas this is not a problem for the direct all-at-once approach (see Section 14.4).

- For (14.15), the computation time of the bundle method is already around 150 times larger than the one of the direct all-at-once approach. One reason for that is certainly the computation of the gradient of the upper-level objective, which is currently done with finite differences. However, the difference between the computation times of the direct all-at-once approach and the bundle method is that large that we do not expect a computation time of a few second of the bundle method for (14.15), even if the derivatives are computed in a more efficient way.

- As for the DFO approach, the general advantage of bilevel methods is that they avoid the complementarity constraints that has to be treated in the framework of the direct all-at-once approach if inequality constraints are present.

- A further advantage of the bundle approach with its current gradient implementation is that it allows to use a black box OCP model, i.e. a black box OCP solver can be used to solve the lower-level OCP.

As for the DFO approach, the disadvantages of using a bundle-based bilevel approach outweigh the advantages by far.

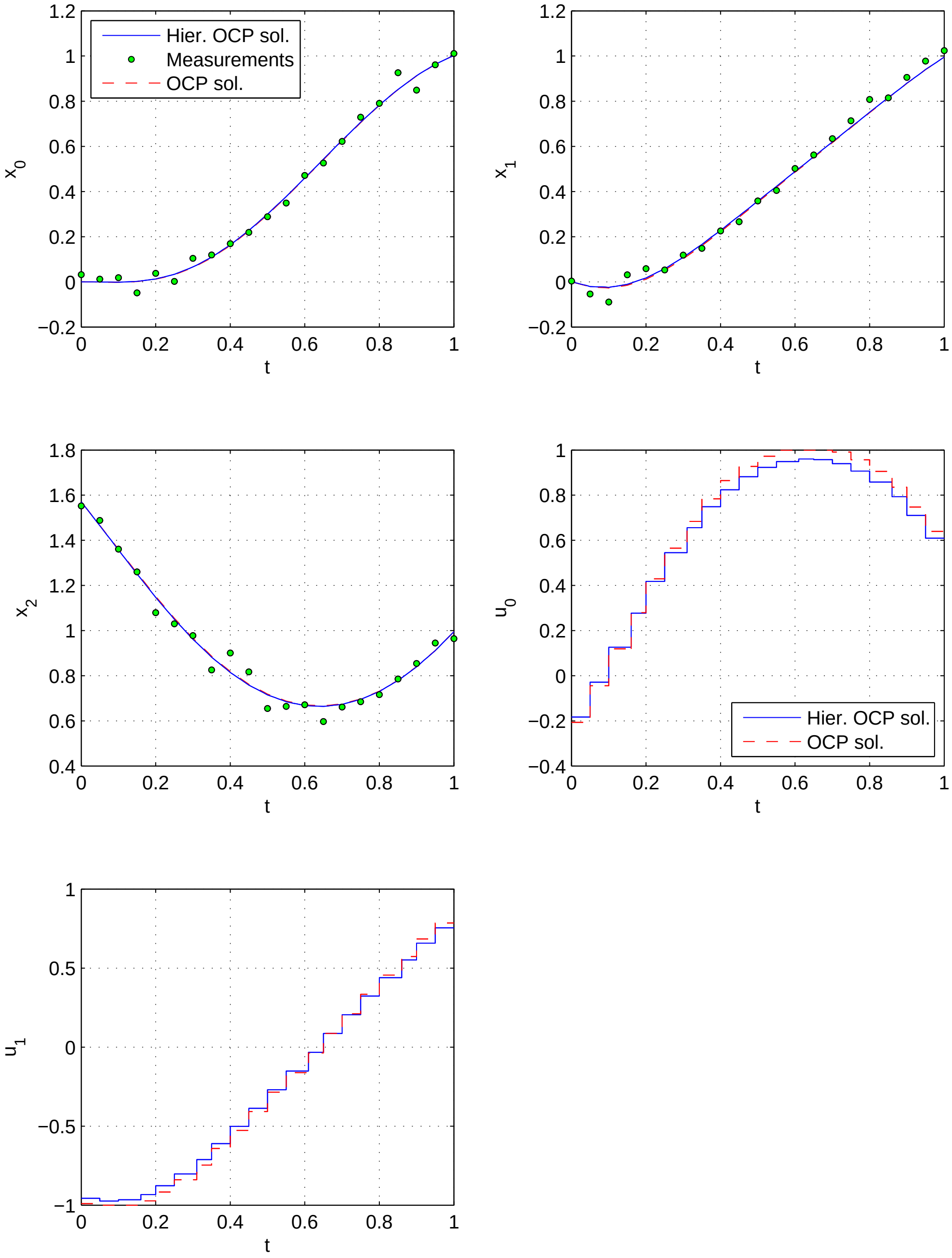


Figure 14.5.: Simulated measurements of the solution x of (14.7) (circles), differential states x and controls u in the solution of the OCP (14.7) (dashed line) and in the solution of the hierarchical formulation (14.9) (solid line).

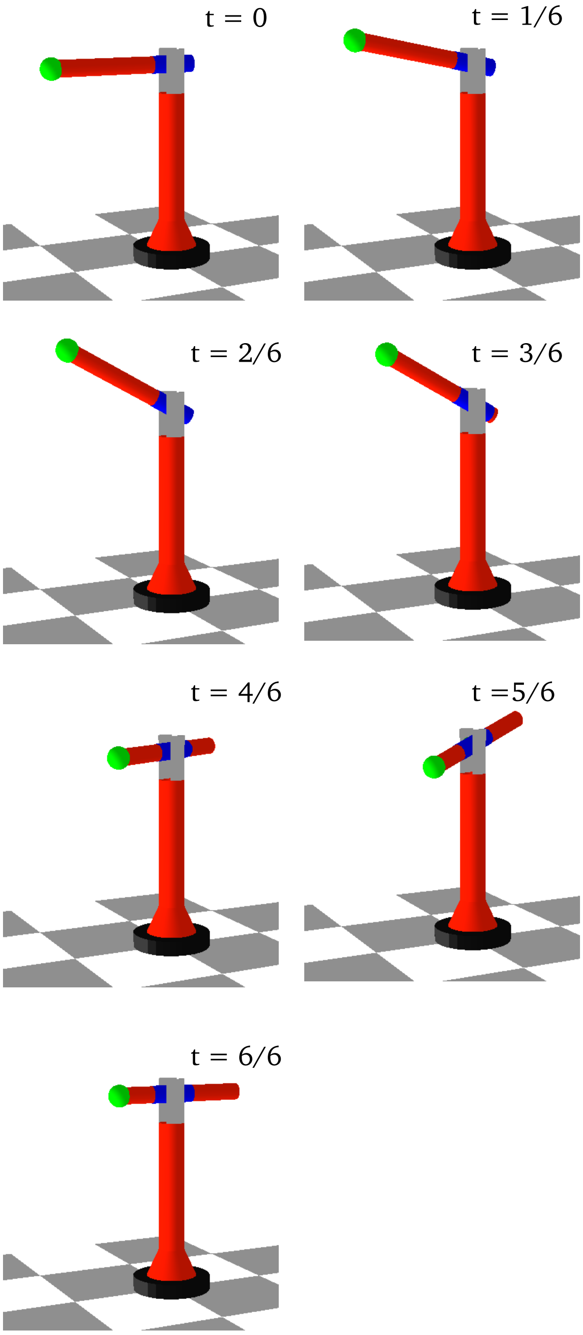


Figure 14.6.: The motion of the polar robot in the solution of (14.11) (using JAVF for the visualization [Win99]).

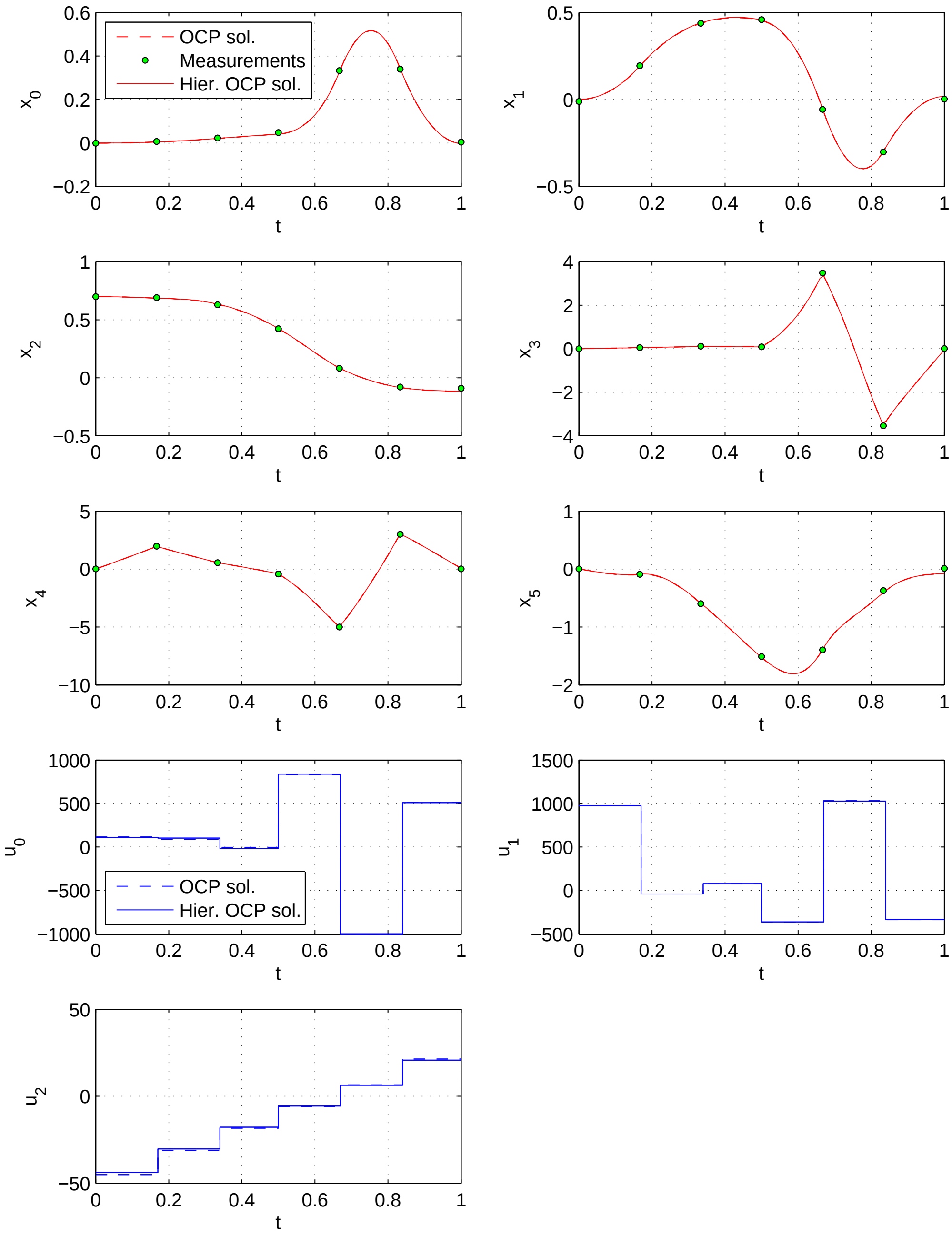


Figure 14.7.: Simulated measurements of the solution x of (14.11) (circles), differential states x and controls u in the solution of the OCP (14.11) (dashed line) and in the solution of the hierarchical formulation (14.14) (solid line).

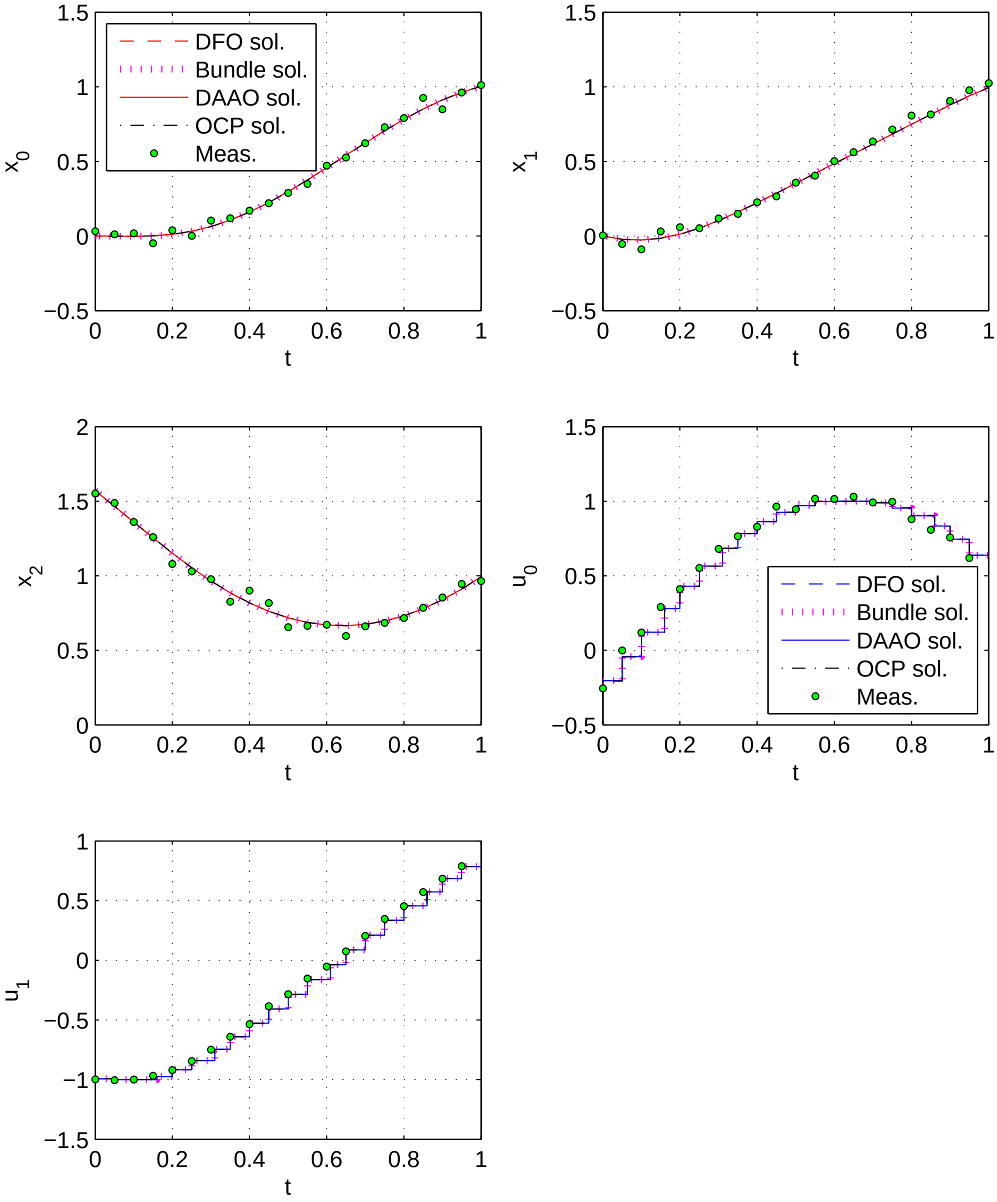


Figure 14.8.: Illustration of the differential states and the controls in the solution of (14.15) obtained by the DFO approach from Section 7.2, the bundle method from Section 7.3 and the direct all-at-once approach (DAAO) from Chapter 5. Furthermore, the solution of (14.7) for $\gamma_1 = \gamma_2 = \gamma_3 = 1/3$ is shown as reference (dash-dot line), and measurements (14.16) and (14.17) are plotted (circles).

Chapter 15.

Estimating parameters in cerebral palsy and healthy gait models from real-world data

In this chapter, we solve a large-scale hierarchical dynamic optimization problem to estimate parameters in an optimal control model for a cerebral palsy patient from real-word motion capture data from the HEIDELBERG MOTIONLAB in the Department of Orthopedic Surgery of the Heidelberg University Clinic in Heidelberg, Germany, based on the optimal control model derived in Section 12.2. We furthermore estimate parameters in an optimal control model for an able-bodied subject based on the optimal control model derived in Section 12.3, also from motion capture data from the HEIDELBERG MOTIONLAB. The results are discussed in detail in the last section of this chapter. We start with a description of the motion capture data provided by the HEIDELBERG MOTIONLAB.

15.1. Motion capture data from the Heidelberg MotionLab

In general, motion capture describes the process of recording a movement and has its origin in the field of biomechanics. Nowadays, tracking systems are not just used in biomechanics, but also in fields like gaming, virtual reality, or animation. In this thesis, we use data recorded with the Vicon motion capture system, which is one of the leading state-of-the-art marker tracking systems produced by VICON MOTION SYSTEMS [VIC13]. The data is collected at the HEIDELBERG MOTIONLAB (more details can be found in Section 11.1 or in [Wol93]). The setup of the Vicon motion capture system in the HEIDELBERG MOTIONLAB is shown in the left picture of Figure 15.1 with high-resolution cameras installed all over the room, which capture the position of the reflecting markers on the subject's skin over time. The second picture in Figure 15.1 shows the placement of the reflecting markers on the subject's skin. The correct positioning of the markers is highly important for achieving accurate data. The markers are placed at specific locations, which are easily identifiable and close to joints/bones. The exact placement is described in the Vicon manual. After recording the motion, the data can be processed and labeled in VICON WORKSTATION, which is a software package supplied by Vicon with the hardware setup. Figure 15.2 shows the recorded markers in Vicon Workstation as stick figure. Furthermore, Figure 15.2 shows joint centers, which are computed based on the collected information (static and dynamic measurements) using the Plug-In-Gait model provided by Vicon. The raw and processed motion capture data is stored in acquisition files in, e.g., c3d format.

We need to access the acquisition files in order to transform and process the collected data such that the measurements can be used to estimate parameters in an optimal control

Figure 15.1.: The Vicon motion capture system in the Heidelberg MotionLab (left picture) and a subject with motion capture markers attached to the skin (right picture) provided by [Wol93].

model with the software package ParaOCP described in Chapter 9. For accessing the c3d files from Vicon, we use the biomechanical toolkit BTK [BTK13], which is an open-source cross-platform library for biomechanical analysis. BTK allows to read, write and modify acquisition files in, e.g., c3d format using the C++, Matlab or Scilab interface.

In this thesis, we use Vicon motion capture data of one gait cycle of a cerebral palsy (CP) patient and of an able-bodied subject provided by the Heidelberg MotionLab including

- the position of lower-body and torso markers in the global coordinate system (cf. Chapter 10)
- static measurements used in Chapter 12 including quantities like body mass, body height and local angle offsets determining the neutral zero joint position explained in Chapter 10 of the respective subject
- joint centers computed based on the Vicon Plug-In-Gait model
- the global marker data expressed as local joint angles (cf. Chapter 10) based on the Vicon Plug-In-Gait model,

which are extracted from the acquisition files in Matlab using BTK. We then use our own implementation of an interface between BTK in Matlab and ParaOCP that transforms the motion capture data into a format that can be accessed by ParaOCP.

15.2. Estimating parameters in an optimal control gait model for a cerebral palsy patient

The goal of this section is to estimate the unknowns γ in the optimal control model for a CP patient derived in Section 12.2 from motion capture measurements of the gait of a CP patient. Therefore, we use local joint angle information and global position and orientation information of the patient's pelvis provided by the motion capture system from the Heidelberg MotionLab as measurements and denote these measurements by

$$
\begin{aligned}
\eta_0 &= \left[(\eta_{0,0,1},\dots,\ \eta_{0,0,24}),\ \ \dots\ \ ,\ (\eta_{0,23,1},\dots,\ \eta_{0,23,24})\right]^{\mathsf{T}} \in \mathbb{R}^{24\times 24},\\
\eta_2 &= \left[(\eta_{2,0,1},\dots,\ \eta_{2,0,24}),\ \ \dots\ \ ,\ (\eta_{2,27,1},\dots,\ \eta_{2,27,24})\right]^{\mathsf{T}} \in \mathbb{R}^{28\times 24},\\
\eta_4 &= \left[(\eta_{4,0,1},\dots,\ \eta_{4,0,24}),\ \ \dots\ \ ,\ (\eta_{4,10,1},\dots,\ \eta_{4,10,24})\right]^{\mathsf{T}} \in \mathbb{R}^{11\times 24},
\end{aligned}
\tag{15.1}
$$

where the first index of η refers to the model stage, the second index to the measurement time on this model stage, and the third index to the differential state that has been measured.

We consider the following hierarchical dynamic optimization problem with $x = (\mathfrak{q}^{\mathsf{T}}, \dot{\mathfrak{q}}^{\mathsf{T}})^{\mathsf{T}}$ (cf. Chapter 12):

$$
\begin{aligned}
\underset{\substack{x,u,q\\ \gamma}}{\text{minimize}}\quad & \frac{1}{2}\Bigg(\sum_{i=0}^{23}\sum_{j=1}^{24}\frac{(\mathfrak{q}_{0,j-1}(\tau^m_{0,i})-\eta_{0,i,j})^2}{\sigma^2}+\sum_{i=0}^{27}\sum_{j=1}^{24}\frac{(\mathfrak{q}_{2,j-1}(\tau^m_{2,i})-\eta_{2,i,j})^2}{\sigma^2}\\
&+\sum_{i=0}^{10}\sum_{j=1}^{24}\frac{(\mathfrak{q}_{4,j-1}(\tau^m_{4,i})-\eta_{4,i,j})^2}{\sigma^2}\Bigg)
\end{aligned}
\tag{15.2}
$$

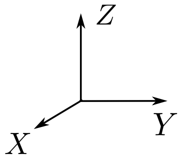
$$
\begin{aligned}
\text{subject to}\quad \underset{x,u,q}{\text{minimize}}\quad & \gamma_1\gamma_1^{\text{sca}}\textstyle\int_0^1\Big([T_{\text{hip\_right}}(\mathfrak{q}_0(t))]_y-[T_{\text{right}}(\mathfrak{q}_0(t))]_y\Big)^2\text{dt}\\
&+\gamma_1\gamma_1^{\text{sca}}\textstyle\int_1^2\Big([T_{\text{hip\_left}}(\mathfrak{q}_2(t))]_y-[T_{\text{left}}(\mathfrak{q}_2(t))]_y\Big)^2\text{dt}\\
&+\gamma_1\gamma_1^{\text{sca}}\textstyle\int_2^3\Big([T_{\text{hip\_right}}(\mathfrak{q}_4(t))]_y-[T_{\text{right}}(\mathfrak{q}_4(t))]_y\Big)^2\text{dt}\\
&+\gamma_2\gamma_2^{\text{sca}}\textstyle\sum_{j=0}^{n_u-1}\Big(\int_0^1 u_{0,j}(t)^2\ \text{dt}\Big)+\gamma_2\gamma_2^{\text{sca}}\ \sum_{j=0}^{n_u-1}\Big(\int_1^2 u_{2,j}(t)^2\ \text{dt}\Big)\\
&+\gamma_2\gamma_2^{\text{sca}}\textstyle\sum_{j=0}^{n_u-1}\Big(\int_2^3 u_{4,j}(t)^2\ \text{dt}\Big)\\
&+\gamma_3\gamma_3^{\text{sca}}\textstyle\sum_{j=1,10}\Big(\int_0^1 u_{0,j}(t)^2\ \text{dt}\Big)+\gamma_3\gamma_3^{\text{sca}}\ \sum_{j=1,10}\Big(\int_1^2 u_{2,j}(t)^2\ \text{dt}\Big)\\
&+\gamma_3\gamma_3^{\text{sca}}\textstyle\sum_{j=1,10}\Big(\int_2^3 u_{4,j}(t)^2\ \text{dt}\Big)\\
&+\gamma_4\gamma_4^{\text{sca}}\textstyle\sum_{j=2,11}\Big(\int_0^1 u_{0,j}(t)^2\ \text{dt}\Big)+\ \gamma_4\gamma_4^{\text{sca}}\ \sum_{j=2,11}\Big(\int_1^2 u_{2,j}(t)^2\ \text{dt}\Big)\\
&+\gamma_4\gamma_4^{\text{sca}}\textstyle\sum_{j=2,11}\Big(\int_2^3 u_{4,j}(t)^2\ \text{dt}\Big)
\end{aligned}
\tag{15.3}
$$

$$
\text{C1}\left\{
\begin{aligned}
\text{subject to}\quad & \dot{x}_0(t)=q_0\begin{pmatrix}\dot{\mathfrak{q}}_0(t)\\ \ddot{\mathfrak{q}}_0(t)\end{pmatrix},\quad t\in[0,1],\ \ x_1(1)=\begin{pmatrix}\mathfrak{q}_2(1)\\ \dot{\mathfrak{q}}_2(1)\end{pmatrix}\\
& \dot{x}_2(t)=q_1\begin{pmatrix}\dot{\mathfrak{q}}_2(t)\\ \ddot{\mathfrak{q}}_2(t)\end{pmatrix},\quad t\in[1,2],\ \ x_3(2)=\begin{pmatrix}\mathfrak{q}_4(2)\\ \dot{\mathfrak{q}}_4(2)\end{pmatrix}\\
& \dot{x}_4(t)=q_2\begin{pmatrix}\dot{\mathfrak{q}}_4(t)\\ \ddot{\mathfrak{q}}_4(t)\end{pmatrix},\quad t\in[2,3]\\
& \mathfrak{q}_1(1)\ =\ \mathfrak{q}_0(1),\ \dot{\mathfrak{q}}_1(1)\ =\ \dot{\mathfrak{q}}_{0,+}(1),\ \mathfrak{q}_2(1)\ =\ \mathfrak{q}_1(1),\dot{\mathfrak{q}}_2(1)\ =\ \dot{\mathfrak{q}}_1(1),\\
& \mathfrak{q}_3(2)\ =\ \mathfrak{q}_2(2),\ \dot{\mathfrak{q}}_3(2)\ =\ \dot{\mathfrak{q}}_{2,+}(2),\ \mathfrak{q}_3(2)\ =\ \mathfrak{q}_2(2),\ \dot{\mathfrak{q}}_3(2)\ =\ \dot{\mathfrak{q}}_2(2),\\[1ex]
& \begin{pmatrix}\mathfrak{M}(\mathfrak{q}_0(t)) & \mathfrak{G}^{\mathsf{T}}_{\text{right}}(\mathfrak{q}_0(t))\\ \mathfrak{G}_{\text{right}}(\mathfrak{q}_0(t)) & 0\end{pmatrix}\begin{pmatrix}\ddot{\mathfrak{q}}_0(t)\\ -\varkappa\end{pmatrix}=\begin{pmatrix}u_0-\mathfrak{N}(\mathfrak{q}_0(t),\dot{\mathfrak{q}}_0(t))\\ -\zeta_{\text{right}}(\mathfrak{q}_0(t),\dot{\mathfrak{q}}_0(t))\end{pmatrix},\ t\in[0,1]\\
& \begin{pmatrix}\mathfrak{M}(\mathfrak{q}_2(t)) & \mathfrak{G}^{\mathsf{T}}_{\text{left}}(\mathfrak{q}_2(t))\\ \mathfrak{G}_{\text{left}}(\mathfrak{q}_2(t)) & 0\end{pmatrix}\begin{pmatrix}\ddot{\mathfrak{q}}_2(t)\\ -\varkappa\end{pmatrix}=\begin{pmatrix}u_2-\mathfrak{N}(\mathfrak{q}_2(t),\dot{\mathfrak{q}}_2(t))\\ -\zeta_{\text{left}}(\mathfrak{q}_2(t),\dot{\mathfrak{q}}_2(t))\end{pmatrix},\ t\in[1,2]\\
& \begin{pmatrix}\mathfrak{M}(\mathfrak{q}_4(t)) & \mathfrak{G}^{\mathsf{T}}_{\text{right}}(\mathfrak{q}_4(t))\\ \mathfrak{G}_{\text{right}}(\mathfrak{q}_4(t)) & 0\end{pmatrix}\begin{pmatrix}\ddot{\mathfrak{q}}_4(t)\\ -\varkappa\end{pmatrix}=\begin{pmatrix}u_4-\mathfrak{N}(\mathfrak{q}_4(t),\dot{\mathfrak{q}}_4(t))\\ -\zeta_{\text{right}}(\mathfrak{q}_4(t),\dot{\mathfrak{q}}_4(t))\end{pmatrix},\ t\in[2,3]\\[1ex]
& \begin{pmatrix}\mathfrak{M}(\mathfrak{q}_0(1)) & \mathfrak{G}^{\mathsf{T}}_{\text{left}}(\mathfrak{q}_0(1))\\ \mathfrak{G}_{\text{left}}(\mathfrak{q}_0(1)) & 0\end{pmatrix}\begin{pmatrix}\dot{\mathfrak{q}}_{0,+}(1)\\ \Lambda_{\text{left}}\end{pmatrix}=\begin{pmatrix}\mathfrak{M}(\mathfrak{q}_0(1))\dot{\mathfrak{q}}_0(1)\\ 0\end{pmatrix}\\
& \begin{pmatrix}\mathfrak{M}(\mathfrak{q}_2(2)) & \mathfrak{G}^{\mathsf{T}}_{\text{right}}(\mathfrak{q}_2(2))\\ \mathfrak{G}_{\text{right}}(\mathfrak{q}_2(2)) & 0\end{pmatrix}\begin{pmatrix}\dot{\mathfrak{q}}_{2,+}(2)\\ \Lambda_{\text{right}}\end{pmatrix}=\begin{pmatrix}\mathfrak{M}(\mathfrak{q}_2(2))\dot{\mathfrak{q}}_2(2)\\ 0\end{pmatrix}
\end{aligned}
\right.
\tag{15.4}
$$

$$
\begin{aligned}
&\text{C2}\begin{cases}
0 = \mathfrak{G}_{\text{right}}(\mathfrak{q}_0(0))\dot{\mathfrak{q}}_0(0) & \\
0 = [T_{\text{left}}(\mathfrak{q}_0(1))]_z & \\
0 \geq \left[\mathfrak{G}_{\text{right}}(\mathfrak{q}_0(1))\dot{\mathfrak{q}}_0(1)\right]_z & \\
0 \leq \left[T_{\text{right}}(\mathfrak{q}_2(t))\right]_z \,, & \forall t \in [1,2) \\
0 = [T_{\text{right}}(\mathfrak{q}_2(2))]_z & \\
0 \geq [\mathfrak{G}_{\text{left}}(\mathfrak{q}_2(2))\dot{\mathfrak{q}}_2(2)]_z & \\
0 \leq [T_{\text{left}}(\mathfrak{q}_4(t))]_z \,, & \forall t \in [2,3]
\end{cases}\\
&\text{C3}\begin{cases}
-1000 \leq \mathfrak{q}_{k,i}(t) \leq 1000 & i = \{0,1,2\}, \quad k \in \{0,\dots,4\}, \forall t \in \mathfrak{I}_k \\
-1.5 \leq \mathfrak{q}_{k,i}(t) \leq 1.5 & i = \{3,\dots,23\}, k \in \{0,\dots,4\}, \forall t \in \mathfrak{I}_k \\
-1000 \leq \dot{\mathfrak{q}}_{k,i}(t) \leq 1000 & i = \{0,\dots,23\}, k \in \{0,\dots,4\}, \forall t \in \mathfrak{I}_k \\
-500 \leq u_i(t) \leq 500 & i = \{0,\dots,17\}, k \in \{0,\dots,4\}, \forall t \in \mathfrak{I}_k \\
0 \leq q_0 \leq 10 & \\
0 \leq q_1 \leq 10 & \\
0 \leq q_2 \leq 10 &
\end{cases}\\
&\text{C4}\begin{cases}
\mathfrak{q}_0(0) = \mathfrak{q}_{\text{initial}} \\
\mathfrak{q}_{4,i}(3) = \mathfrak{q}_{\text{end},i}, i = \{0,\dots,5\} \\
T_{\text{left}}(\mathfrak{q}_4(3)) = P_{\text{foot\_left\_end}}
\end{cases}
\end{aligned}
\tag{15.5}
$$

$$\textstyle\sum_{i=1}^{4} \gamma_i = 1, \;\; \gamma \geq 0, \tag{15.6}$$

which contains the optimal control model for CP gaits derived and explained in Section 12.2. and scaling factors $(\gamma_1^{\text{sca}}, \gamma_2^{\text{sca}}, \gamma_3^{\text{sca}}, \gamma_4^{\text{sca}}) = (1\text{E}{+}03, 1\text{E-}04, 1, 1)$ to compensate for the different orders of magnitude of the subcriteria. We briefly recall the main definitions used in (15.2)-(15.6). Problem (15.2)-(15.6) includes five model stages (i.e. $n_S = 5$): the first single support phase of the right foot, the first transition stage, the single support phase of the left foot, the second transition stage and the second single support phase of the right foot. We have 48 differential states $x_k(t) := (\mathfrak{q}_k(t)^\intercal, \dot{\mathfrak{q}}_k(t)^\intercal)^\intercal \in \mathbb{R}^{48}$ on each model stage $k \in \{0,\dots,4\}$, where the first 24 differential states $\mathfrak{q}_k(t)$ are the DOFs of the multibody system modeling the patient's skeleton illustrated in Figure 12.2. We use $Y'X'Z'$ Euler angles and the following coordinate system:



The meaning of the first 24 differential states is explained in Table 15.1, where $k \in \{0,\dots 4\}$.

Diff. state		**Meaning**
$\mathfrak{q}_{k,0}$	–	x-position of the pelvis in the global coordinate system
$\mathfrak{q}_{k,1}$	–	y-position of the pelvis in the global coordinate system
$\mathfrak{q}_{k,2}$	–	z-position of the pelvis in the global coordinate system
$\mathfrak{q}_{k,3}$	–	y-angle of the orientation of the pelvis in the global coordinate system
$\mathfrak{q}_{k,4}$	–	x-angle of the orientation of the pelvis in the global coordinate system
$\mathfrak{q}_{k,5}$	–	z-angle of the orientation of the pelvis in the global coordinate system
$\mathfrak{q}_{k,6}$	–	y-angle of the orientation of the right hip joint in the local frame
$\mathfrak{q}_{k,7}$	–	x-angle of the orientation of the right hip joint in the local frame
$\mathfrak{q}_{k,8}$	–	z-angle of the orientation of the right hip joint in the local frame

$\mathfrak{q}_{k,9}$	–	y-angle of the orientation of the right knee joint in the local frame
$\mathfrak{q}_{k,10}$	–	x-angle of the orientation of the right knee joint in the local frame
$\mathfrak{q}_{k,11}$	–	z-angle of the orientation of the right knee joint in the local frame
$\mathfrak{q}_{k,12}$	–	y-angle of the orientation of the right ankle joint in the local frame
$\mathfrak{q}_{k,13}$	–	x-angle of the orientation of the right ankle joint in the local frame
$\mathfrak{q}_{k,14}$	–	z-angle of the orientation of the right ankle joint in the local frame
$\mathfrak{q}_{k,15}$	–	y-angle of the orientation of the left hip joint in the local frame
$\mathfrak{q}_{k,16}$	–	x-angle of the orientation of the left hip joint in the local frame
$\mathfrak{q}_{k,17}$	–	z-angle of the orientation of the left hip joint in the local frame
$\mathfrak{q}_{k,18}$	–	y-angle of the orientation of the left knee joint in the local frame
$\mathfrak{q}_{k,19}$	–	x-angle of the orientation of the left knee joint in the local frame
$\mathfrak{q}_{k,20}$	–	z-angle of the orientation of the left knee joint in the local frame
$\mathfrak{q}_{k,21}$	–	y-angle of the orientation of the left ankle joint in the local frame
$\mathfrak{q}_{k,22}$	–	x-angle of the orientation of the left ankle joint in the local frame
$\mathfrak{q}_{k,23}$	–	z-angle of the orientation of the left ankle joint in the local frame

Table 15.1.: Explanation of the differential states describing the DOFs of the underlying multibody system in (15.2)-(15.6).

The remaining 24 differential states $\dot{\mathfrak{q}}_{k,0}, \ldots, \dot{\mathfrak{q}}_{k,23}$ are the velocities corresponding to the angles $\mathfrak{q}_{k,0}, \ldots, \mathfrak{q}_{k,23}$. In (15.2)-(15.6), we use $x := (x_0, \ldots, x_4)$, which summarizes the differential states on all model stages. The argument t of $x, \mathfrak{q}, \dot{\mathfrak{q}}$ or u is sometimes skipped for the sake of a compact presentation. We furthermore have 18 controls $u_{k,0}, \ldots, u_{k,17}$, which are the torques acting on the angles $\mathfrak{q}_{k,6}, \ldots, \mathfrak{q}_{k,23}$. Problem (15.2)-(15.6) includes three control values q_0, q_1 and q_2, which describe the length of the model stages with the index $0, 2$ and 4, respectively. We have 24 equidistant measurement times on the first model stage ($k = 0$):

$$\tau_{0,i}^m = i \cdot \frac{1}{24}, \quad i \in \{0, \ldots, 23\}, \tag{15.7}$$

28 equidistant measurement times on the third model stage ($k = 2$):

$$\tau_{2,i}^m = 1 + i \cdot \frac{1}{28}, \quad i \in \{0, \ldots, 27\}, \tag{15.8}$$

and 11 equidistant measurement times on the last model stage ($k = 4$):

$$\tau_{4,i}^m = 2 + i \cdot \frac{1}{10}, \quad i \in \{0, \ldots, 10\}. \tag{15.9}$$

The upper-level parameter estimation objective of (15.2)-(15.6) contains the standard deviation σ. The standard deviations are the same for all measurements in (15.1) and are therefore chosen to be 1 in the implementation of (15.2)-(15.6) in ParaOCP since the objective function is scale invariant. The lower-level objective function of (15.2)-(15.6) contains four subcriteria on each model stage weighted by the upper-level variables $\gamma_1, \ldots, \gamma_4$, which are to be determined from measurements in (15.2)-(15.6). T_{right} and T_{left} describe the potential contact points of the right and the left foot in the global coordinate system,

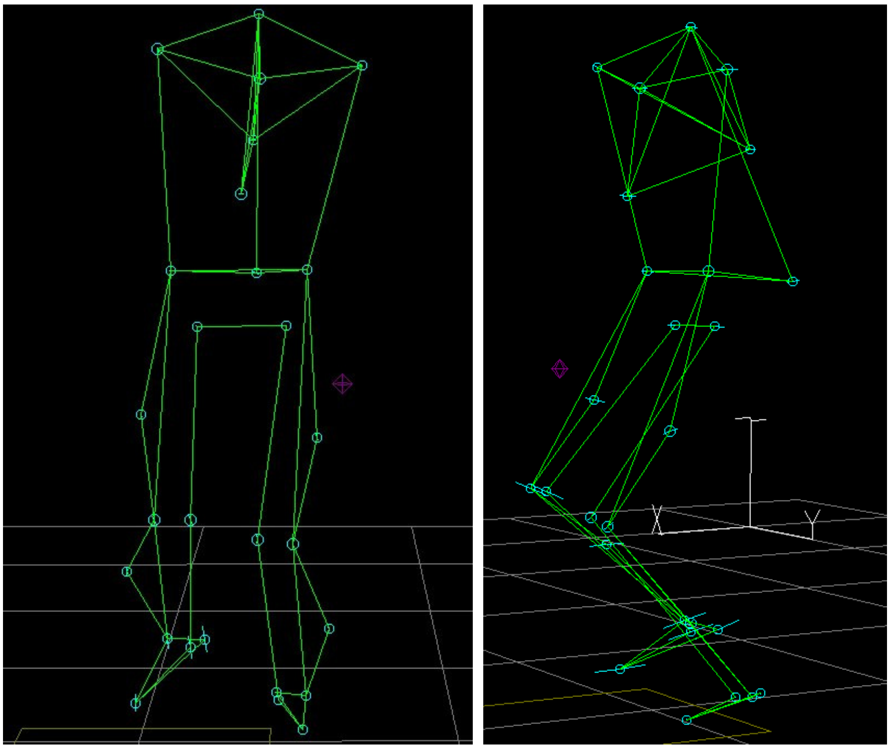


Figure 15.2.: A visual representation of the collected marker data in the Vicon software.

and $T_{\text{hip\_right}}$ and $T_{\text{hip\_left}}$ denote the right and left hip joint in the global coordinate system (details are explained in Section 12.2).

Block C1 of the constraints in (15.2)-(15.6) contains the differential algebraic equation for each model stage derived in Section 12.2, and the transition conditions with two jumps between the five model stages, where in (15.2)-(15.6), $\mathfrak{M}(\cdot)$ describes the mass matrix of the underlying multibody system model, $\mathfrak{G}_{\text{right}}(\cdot)$ and $\mathfrak{G}_{\text{left}}(\cdot)$ are the contact Jacobians of the right and left foot, $\mathfrak{N}(\cdot)$ denotes the nonlinear forces, and $\zeta_{\text{right}}(\cdot)$ and $\zeta_{\text{left}}(\cdot)$ the contact Hessians.

The constraints in block C2 in (15.2)-(15.6) are explained in detail in Section 12.2 and, e.g., implicitly determine the length of the single support phases. Block C3 contains bounds on the differential states, the controls and the control values. Block C4 includes initial and end values for the differential states. The first 24 differential states are fixed to $\mathfrak{q}_{\text{initial}}$ at $t = 0$, where $\mathfrak{q}_{\text{initial}}$ contains motion capture data for the respective quantities at $t = 0$. At $t = 3$, the location and the y- and x-orientation of the patient's pelvis is fixed to motion capture data for the respective quantities denoted by $\mathfrak{q}_{\text{end}}$. Furthermore, the potential contact point of the left foot is fixed to the corresponding measurement value denoted by $P_{\text{foot\_left\_end}}$.

All computations in this chapter are performed on a compute server with 4 Intel Quad Core Xeon E7330 CPUs with 2.4 GHz and 128 GB RAM in total. We note that only one core is used for our computations.

Problem (15.2)-(15.6) is now solved with ParaOCP. The exact setting used in ParaOCP is summarized in Table 15.2. In total, we use 62 multiple shooting and control discretization intervals. The multiple shooting grid and the control discretization grid are equal to the measurement grid (15.7)-(15.9). Table 15.2 states the number of multiple shooting intervals per model stage. As initial guess for $\gamma_1, \gamma_2, \gamma_3, \gamma_4$, we use $(9.7195\text{E-}01, 2.7770\text{E-}02, 1.3885\text{E-}04, 1.3885\text{E-}04)$. The lower-level OCP of (15.2)-(15.6) is a highly nonlinear multistage problem, which requires very good initial values in order to converge to a solution. Our experience is that just using the motion capture data and the force data provided by the Vicon system as initial values is not sufficient. We have carefully generated initial data for the lower-level problem of (15.2)-(15.6) by building homotopies between the motion capture data and a set of initial data that leads to a solution of (15.2)-(15.6). The resulting values are used as initial data for problem (15.2)-(15.6). The measurements (15.1), the differential states and the controls in the solution of (15.2)-(15.6) obtained with ParaOCP are shown in Figures 15.3, 15.4, 15.5 and 15.6. The upper-level least-squares objective value, the upper-level variables γ, the control values, the computation time and the number of iterations is given in Table 15.3. An important observation is that for problem (15.2)-(15.6), the direct all-at-once approach derived in Chapter 5 leads to infeasible quadratic programs (QPs) as discussed in Chapter 8, and not even the lifting approach presented in Chapter 8 helps to tackle this problem. In order to solve problem (15.2)-(15.6), the complementarity constraint (CC) (5.67) is added as exact ℓ_1-norm penalty to the upper-level objective function as described in, e.g., [BSSV06]. The complementarity constraint is an inner product of two vectors of size 8406, where in the solution, the largest summand is 1.5771E-03 and 6530 of 8406 summands are zero. As penalty parameter, we use 1000. The penalty term is expected to become zero is the solution for a penalty parameter that is sufficiently large. However, our experience is that it is not possible to completely drive the CC to zero in the solution (even for much larger penalty parameters), which might be due to scaling issues related to the high-dimensional problem. The treatment of infeasible QP approximations due to the lack of a constraint qualification in large-scale hierarchical dynamic optimization problems, where lifting the problem formulation does not help is investigated in continuing research. The upper-level least-squares objective value in the solution of (15.2)-(15.6) is 2.4723E+00, which corresponds to a mean squared error of 1.6351E-03. The one-level NLP resulting from (15.2)-(15.6) includes 15654 variables and 7243 nonlinear constraints. Table 15.2 shows the values of γ in the solution of (15.2)-(15.6).

For the visualization of the patient's gait described by the optimal control model identified in (15.2)-(15.6), we us the C++ package MeshUp [Fel13]. MeshUp is a visualization tool for multi-body systems rendering motions directly to videos or image sequences. Figures 15.7 and 15.8 show such image sequences in two perspectives. Each of the figures shows the local joint angle information and global position and orientation information of the patient's pelvis provided by the motion capture system, and the derived optimal control gait model (the solution of (15.2)-(15.6)) as overlay.

15.3. Estimating parameters in an optimal control gait model for an able-bodied subject

In this section, we estimate parameters in an optimal control model for an able-bodied subject, also from motion capture data provided by the Heidelberg MotionLab, based on the optimal control model derived in Section 12.3. As in the last section, we use local joint angle information and global position and orientation information of the subject's pelvis

	Hier. problem (15.2)-(15.6)
# model stages	5
# multiple shooting intervals per m. stage	24,0,28,0,10
# sized control discr. intervals per m. stage	24,0,28,0,10
control discretization type	piecew. const.
# measurement times	63
# measurement values	$63 \cdot 24$
constraint Jacobian representation	sparse
termination tolerance filterSQP	1E-04
upper-level Hessian type	Gauß-Newton
lower-level Hessian type	efficient Hessian calculations (cf. Chapter 5)
integration tolerance	1E-05

Table 15.2.: Overview of the exact setting used in ParaOCP to solve the hierarchical problem (15.2)-(15.6).

provided by the motion capture system from the Heidelberg MotionLab as measurements and denote these measurements by

$$\eta_0 = \left[(\eta_{0,0,1}, \dots,\ \eta_{0,0,24})\,,\ \dots\ ,\ (\eta_{0,34,1}, \dots,\ \eta_{0,34,24})\right]^\intercal \in \mathbb{R}^{35\times 24}, \tag{15.10}$$

$$\eta_2 = \left[(\eta_{2,0,1}, \dots,\ \eta_{2,0,24})\,,\ \dots\ ,\ (\eta_{2,59,1}, \dots,\ \eta_{2,59,24})\right]^\intercal \in \mathbb{R}^{60\times 24}, \tag{15.11}$$

$$\eta_4 = \left[(\eta_{4,0,1}, \dots,\ \eta_{4,0,24})\,,\ \dots\ ,\ (\eta_{4,20,1}, \dots,\ \eta_{4,20,24})\right]^\intercal \in \mathbb{R}^{21\times 24}, \tag{15.12}$$

where the first index of η refers to the model stage, the second index to the measurement time on this model stage, and the third index to the differential state that has been measured.

We again consider the hierarchical dynamic optimization problem (15.2)-(15.5) and replace the parameter estimation function objective (15.2) by

$$\frac{1}{2}\left(\sum_{i=0}^{35}\sum_{j=1}^{24}\frac{(\mathfrak{q}_{0,j-1}(\tau_{0,i}^m)-\eta_{0,i,j})^2}{\sigma^2}+\sum_{i=0}^{60}\sum_{j=1}^{24}\frac{(\mathfrak{q}_{2,j-1}(\tau_{2,i}^m)-\eta_{2,i,j})^2}{\sigma^2}+\sum_{i=0}^{20}\sum_{j=1}^{24}\frac{(\mathfrak{q}_{4,j-1}(\tau_{4,i}^m)-\eta_{4,i,j})^2}{\sigma^2}\right), \tag{15.13}$$

which is then referred to as problem ((15.13), (15.3)-(15.6)) with 35 equidistant measurement times on the first model stage ($k = 0$):

$$\tau_{0,i}^m = i \cdot \frac{1}{35}, \quad i \in \{0, \dots, 34\}, \tag{15.14}$$

60 equidistant measurement times on the third model stage ($k = 2$):

$$\tau_{2,i}^m = i \cdot \frac{1}{60}, \quad i \in \{0, \dots, 59\}, \tag{15.15}$$

and 20 equidistant measurement times on the last model stage ($k = 4$):

$$\tau_{4,i}^m = i \cdot \frac{1}{20}, \quad i \in \{0, \dots, 20\}. \tag{15.16}$$

Solution of (15.2)-(15.6)	
least-squares obj. val.	2.4724E+00
γ_1	9.9969E-01
γ_2	3.1267E-04
γ_3	0
γ_4	0
q_0	3.4100E-01
q_1	3.5125E-01
q_2	7.0494E-02
comp. time in s	6.45074E+05
# iterations	41

Table 15.3.: Summary of the solution of (15.2)-(15.6) obtained with ParaOCP.

All remaining adaptions of the underlying OCP of ((15.13), (15.3)-(15.6)) (like, e.g., the adaption of $P_{\text{foot\_left\_end}}$) are described in Section 12.3. The constraints in Block C4 of ((15.13), (15.3)-(15.6)) now fixes the first 24 differential states to $\mathfrak{q}_{\text{initial}}$ at $t = 0$, where $\mathfrak{q}_{\text{initial}}$ contains motion capture data of the able-bodied subject for the respective quantities at $t = 0$. And at $t = 3$, the location and the y- and x-orientation of the subject's pelvis is fixed to motion capture data of the able-bodied subject for the respective quantities denoted by $\mathfrak{q}_{\text{end}}$. We also fix the potential contact point of the left foot to the corresponding measurement value denoted by $P_{\text{foot\_left\_end}}$.

We now solve the problem ((15.13), (15.3)-(15.6)) with ParaOCP. The exact setting used in ParaOCP is summarized in Table 15.4. In total, we use 115 multiple shooting and control discretization intervals. The multiple shooting grid and the control discretization grid are equal to the measurement grid (15.14)-(15.16), and Table 15.4 states the number of multiple shooting intervals per model stage. We use $\sigma = 1.$ in (15.13) since the standard deviation is the same for all measurements and the objective function is scale invariant. As initial guess for $\gamma_1, \gamma_2, \gamma_3, \gamma_4$, we use $(6.0606\text{E-}09, 9.0909\text{E-}01, 6.0606\text{E-}02, 3.0303\text{E-}02)$. As scaling variables, we use $(\gamma_1^{\text{sca}}, \gamma_2^{\text{sca}}, \gamma_3^{\text{sca}}, \gamma_4^{\text{sca}}) = (1\text{E+}05, 1\text{E-}03, 1\text{E-}02, 1\text{E-}02)$. We furthermore generate initial values for the differential states, the controls, the control values, and the Lagrange multipliers by building homotopies between the motion capture data and a set of initial data that leads to a solution of ((15.13), (15.3)-(15.6)). The upper-level least-squares objective value in the solution of ((15.13), (15.3)-(15.6)) is 1.0599E+00, which corresponds to a mean squared error of 3.8071E-04. The one-level nonlinear program resulting from problem ((15.13), (15.3)-(15.6)) includes 15654 variables and 7242 nonlinear constraints. The complementarity constraint is, as described in the last section, added as ℓ_1-norm penalty with a penalty parameter of 1000. In the solution, the largest summand of the inner product of two vectors of size 15488 in the complementarity constraint is 1.5649E-01, and 11503 of 15488 summands are zero. The measurements (15.10), the differential states and the controls in the solution of ((15.13), (15.3)-(15.6)) obtained with ParaOCP are shown in Figures 15.9, 15.10, 15.11 and 15.12. Further details of the solution are given in Table 15.5. Figures 15.13 and 15.14 show an image sequence in two perspectives, where each of the pictures shows the local joint angle information and global position and orientation

	Hier. problem ((15.13), (15.3)-(15.6))
# model stages	5
# mult. shoot. int. per m. stage	35,0,60,0,20
# contr. discr. int. per m. stage	35,0,60,0,20
control discretization type	piecew. const.
# measurement times	116
# measurement values	$116 \cdot 24$
constraint Jacobian representation	sparse
termination tolerance FILTERSQP	1E-04
upper-level Hessian type	Gauß-Newton
lower-level Hessian type	efficient Hessian calculations
integration tolerance	1E-05

Table 15.4.: Overview of the exact setting used in ParaOCP to solve the hierarchical problem ((15.13), (15.3)-(15.6)).

information of the subject's pelvis provided by the motion capture system, and the derived optimal control gait model as overlay using MeshUp.

15.4. Discussion and outlook

In this chapter, we have estimated parameters in an optimal control model for a CP patient and for an able-bodied subject from motion capture data provided by the Heidelberg MotionLab by solving two large-scale hierarchical dynamic optimization problems. We provide visualizations of the measured gait data for the patient and the subject and compare it to the respective derived gait model in Figures 15.7, 15.8, 15.13 and 15.14, which shows that the gait of both derived gait models is very close to the motion capture data. This means that both optimal control models presented in this Chapter are able to capture the main characteristics of the measured gait.

Furthermore, the constitutions of the lower-level objectives of the CP gait model and the able-bodied subject's gait model provide new insights. In the CP gait model, the stability criterion on the lower-level plays an important role and is mainly responsible for the typical S-shape of the y-component of the pelvis location, which is observed in the CP patient's gaits. The stability criterion is much less important in the gait model for the able-bodied subject, where the total energy consumption plays a much more important role. We note that these insights cannot be obtained with pure multibody system models from literature.

In Section 12.4, we describe our long term goals concerning the use of optimal control gait models, which include the improvement of the general understanding of human motion, the development of categories of gaits and of criteria for evaluating gaits, and model-based treatment planning. The results presented in this chapter are a first step into that direction. The estimated constitution of the objectives of both gait models provides new insights and a basis for the classification of gaits. Furthermore, comparing the CP gait model to the able-bodied gait model and the estimation of parameters in the CP gait model from

Solution of ((15.13), (15.3)-(15.6))	
least-squares obj. val.	1.0599E+00
γ_1	1.5997E-07
γ_2	7.3423E-01
γ_3	2.1784E-01
γ_4	4.7929E-02
q_0	2.4079E-01
q_1	3.7071E-01
q_2	1.2882E-01
comp. time in s	1.10279E+06
# iterations	29

Table 15.5.: Summary of the solution of problem ((15.13), (15.3)-(15.6)) obtained with ParaOCP.

measurements recorded after a treatment allows to establish gait evaluation criteria. The CP gait model derived in this thesis can now be used as a first test environment for possible treatments that change the patient's skeleton. The gait model then allows to get idea of how the patient's gait changes after the treatment.

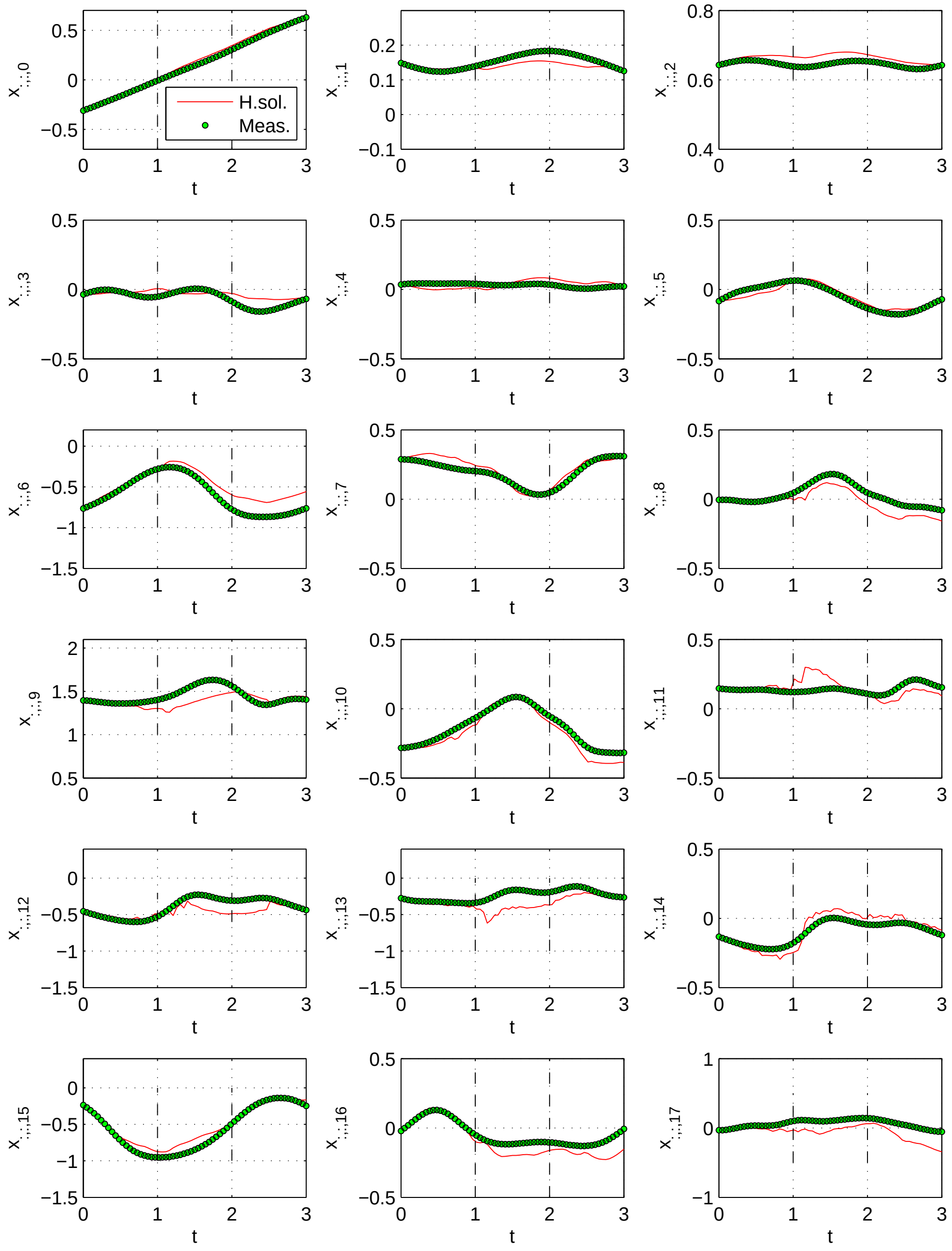


Figure 15.3.: Motion capture data (called Meas.) and the differential states (called H. sol.) in the solution of (15.2)-(15.6).

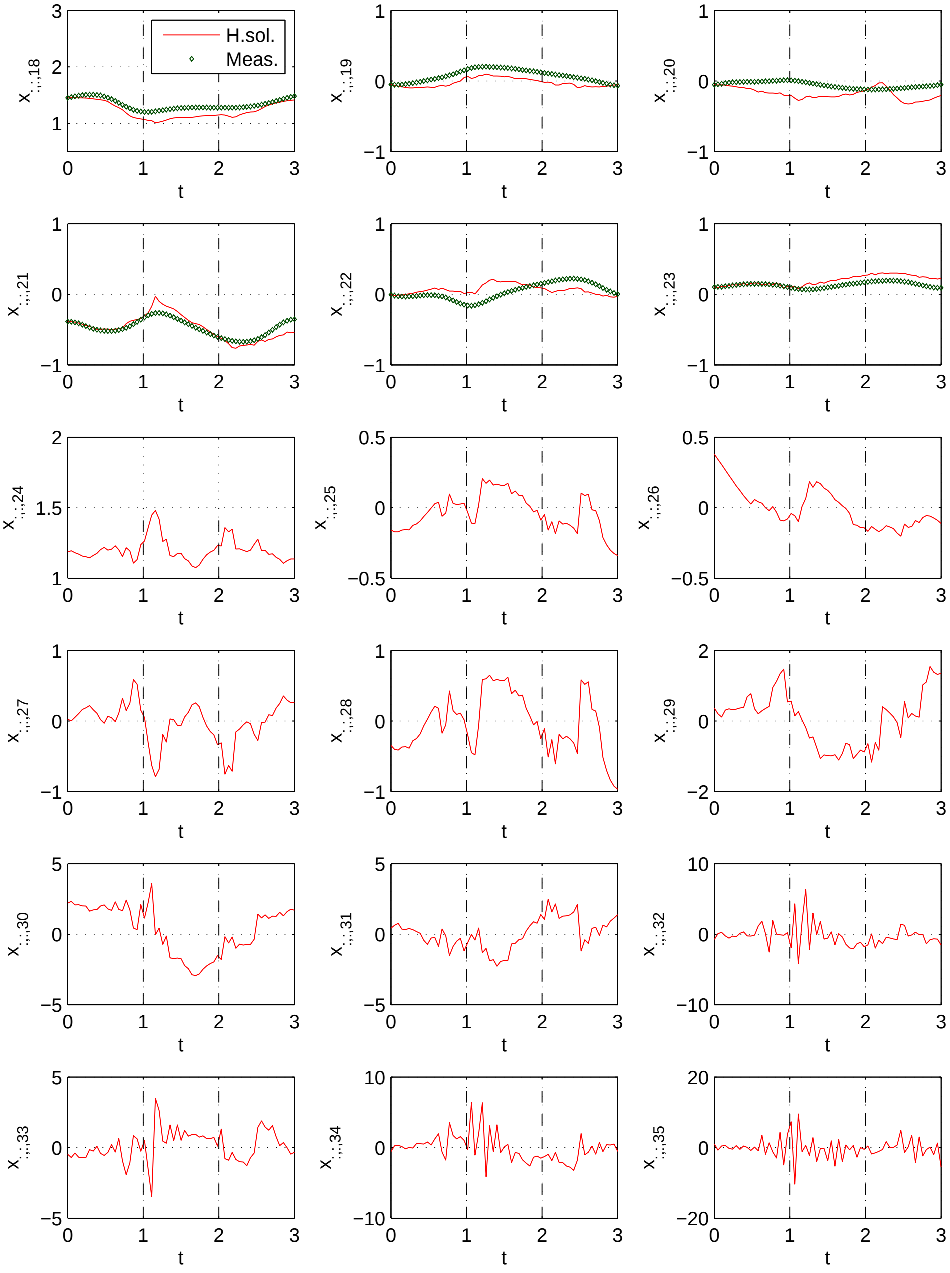


Figure 15.4.: Motion capture data (called Meas.) and the differential states (called H. sol.) in the solution of (15.2)-(15.6).

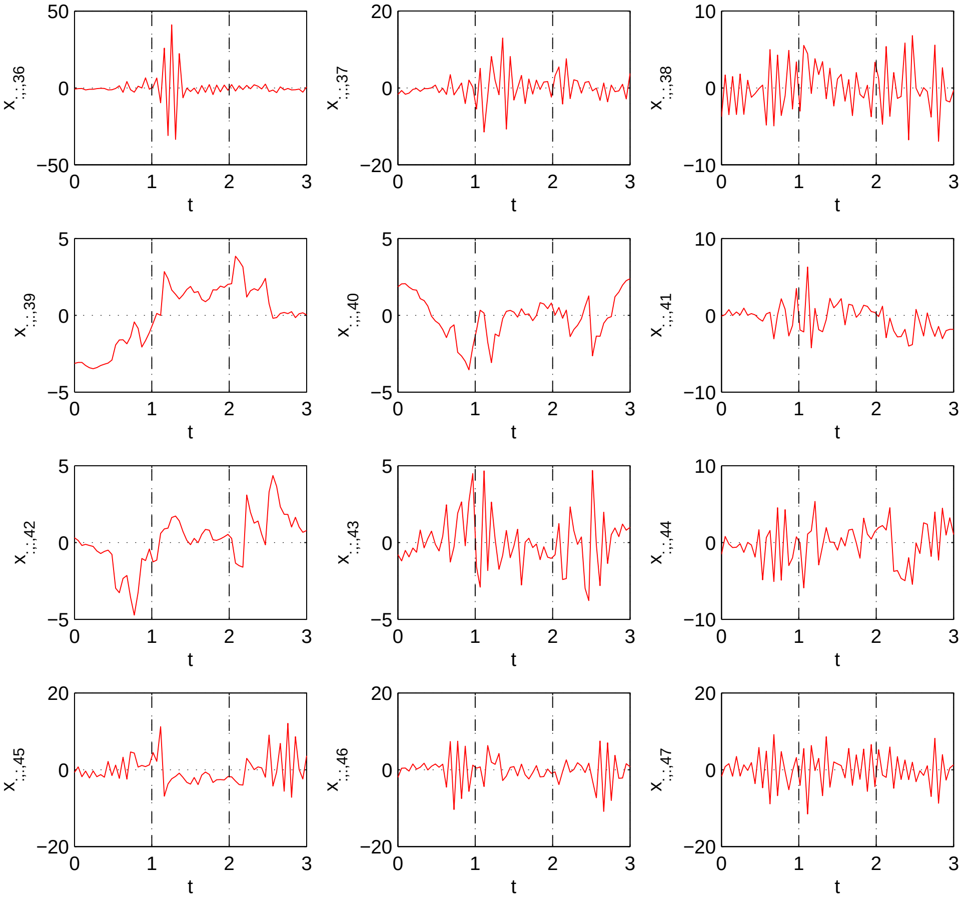


Figure 15.5.: Differential states in the solution of (15.2)-(15.6).

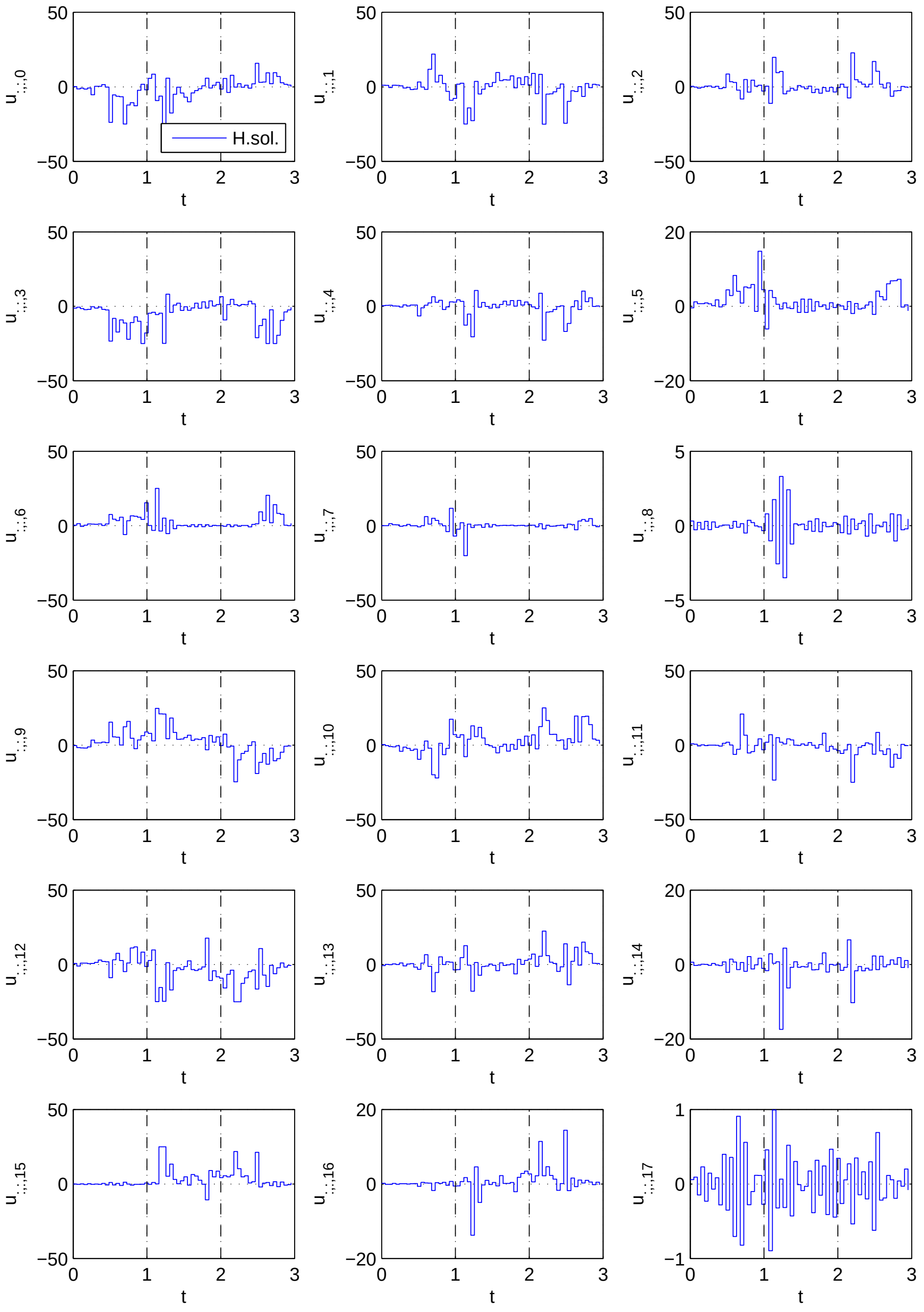


Figure 15.6.: Controls in the solution of (15.2)-(15.6).

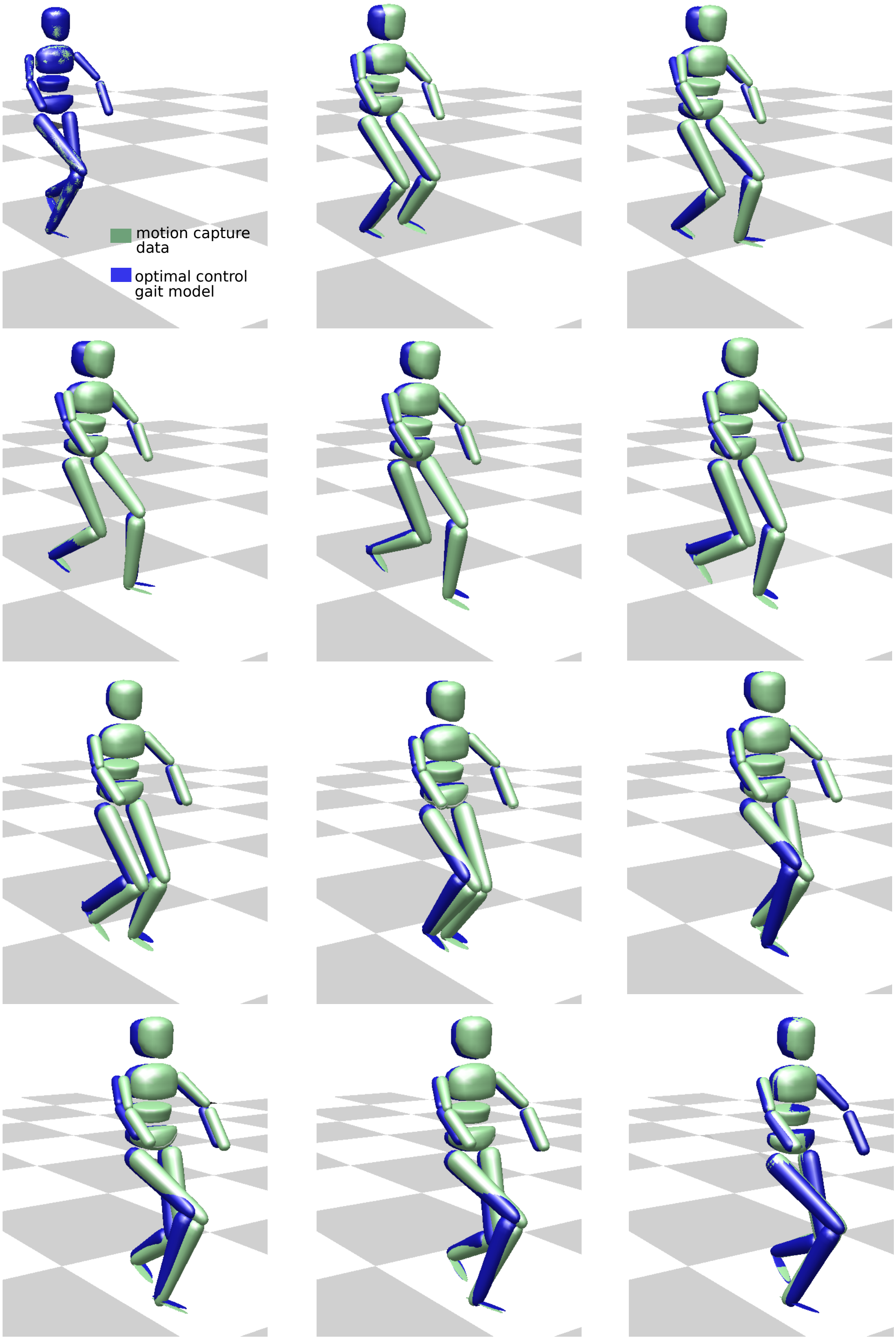


Figure 15.7.: Visualization of the measured gait compared to the solution of (15.2)-(15.6) (the optimal control CP gait model).

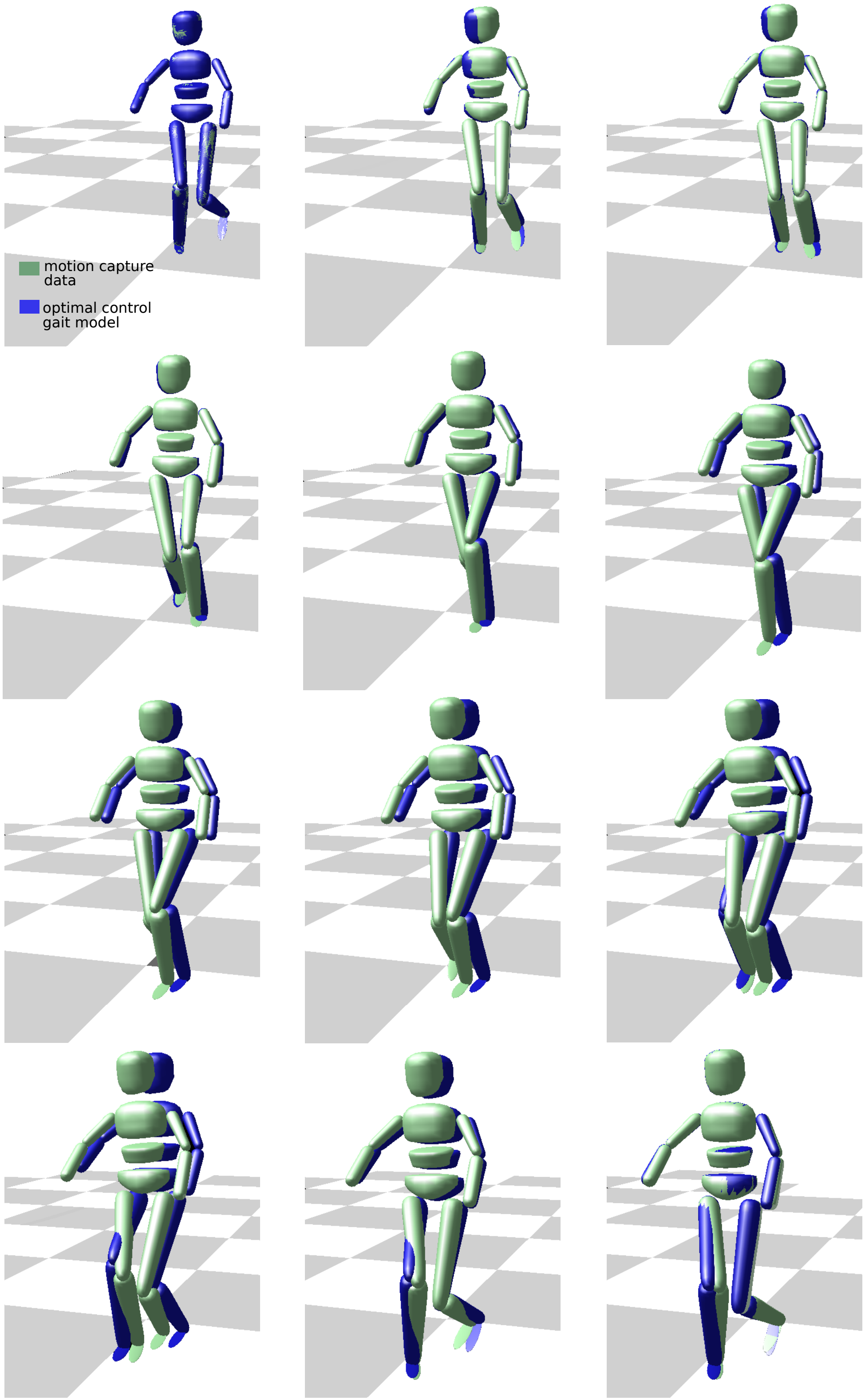


Figure 15.8.: Visualization of the measured gait compared to the solution of (15.2)-(15.6) (the optimal control CP gait model) from a second perspective.

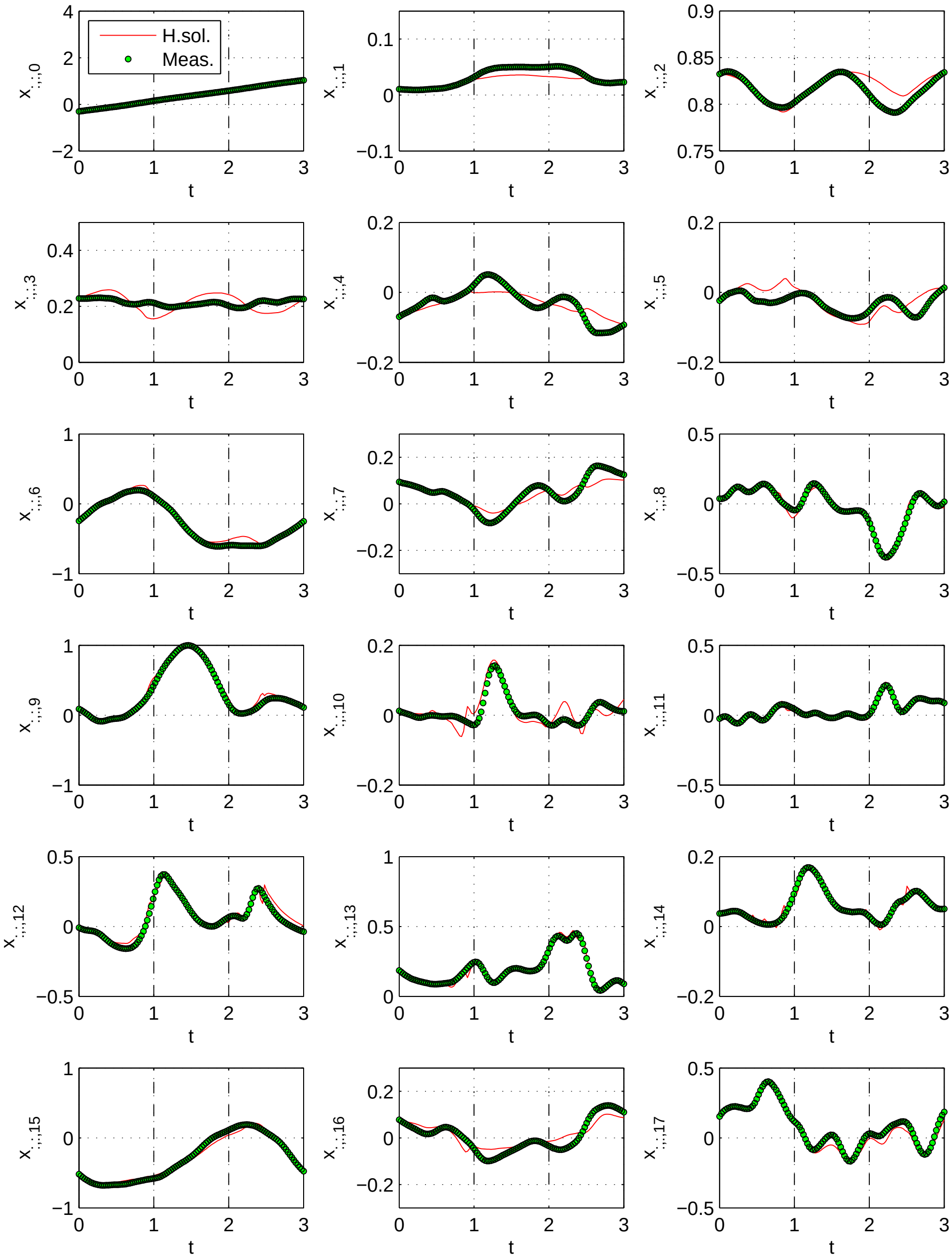


Figure 15.9.: Motion capture data (called Meas.) and the differential states (called H. sol.) in the solution of ((15.13), (15.3)-(15.6)).

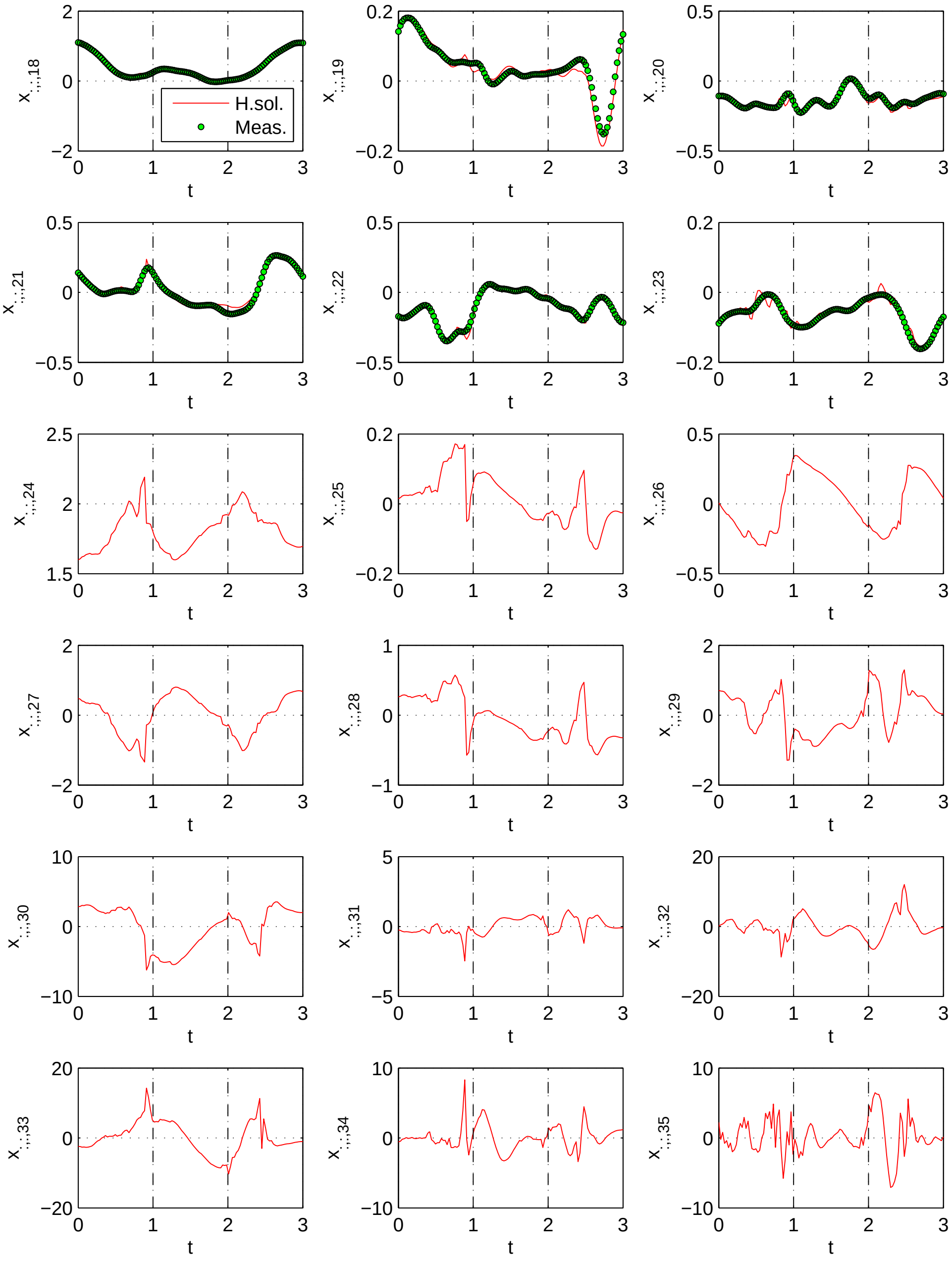


Figure 15.10.: Motion capture data (called Meas.) and the differential states (called H. sol.) in the solution of ((15.13), (15.3)-(15.6)).

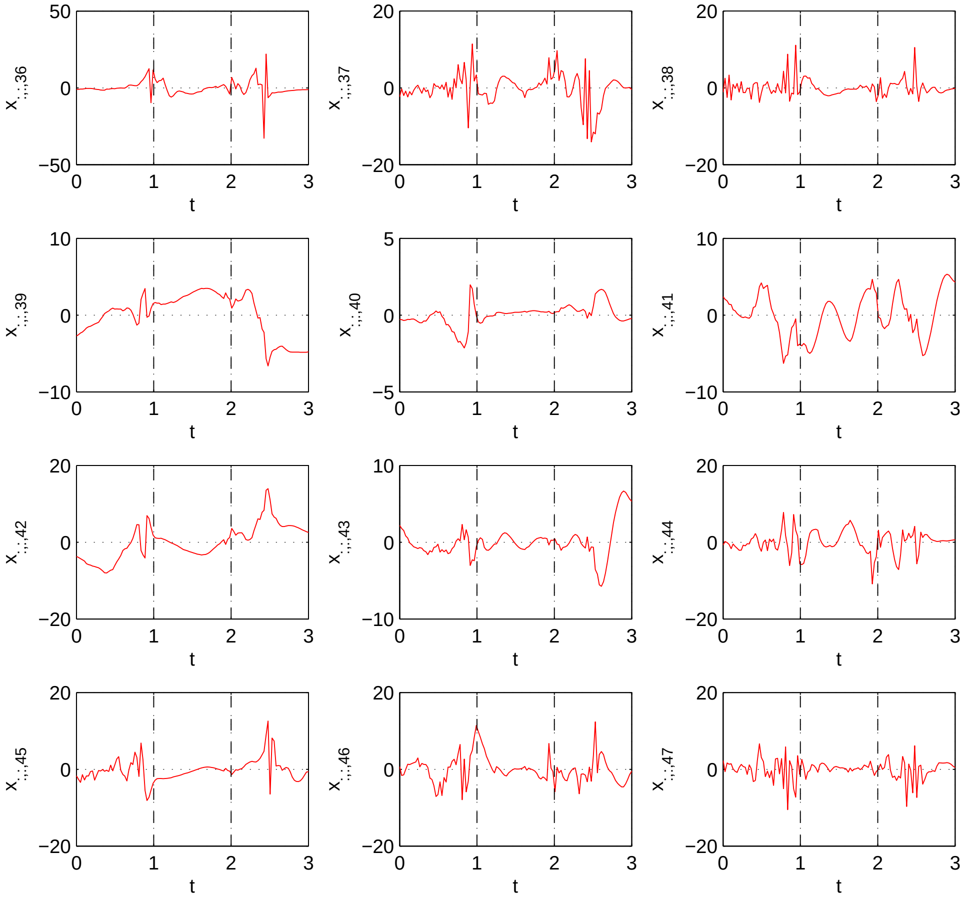


Figure 15.11.: Differential states in the solution of ((15.13), (15.3)-(15.6)).

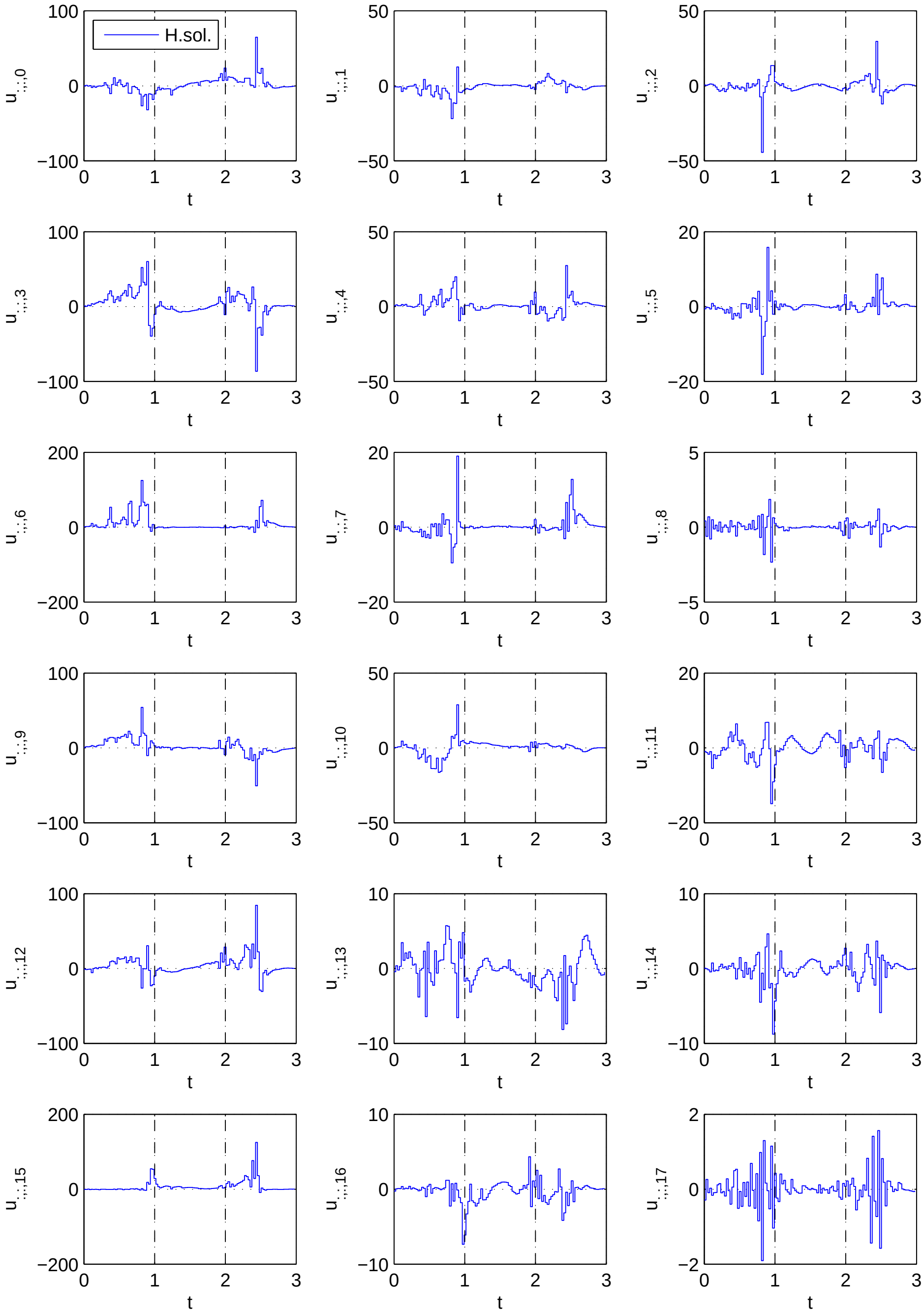


Figure 15.12.: Controls in the solution of ((15.13), (15.3)-(15.6)).

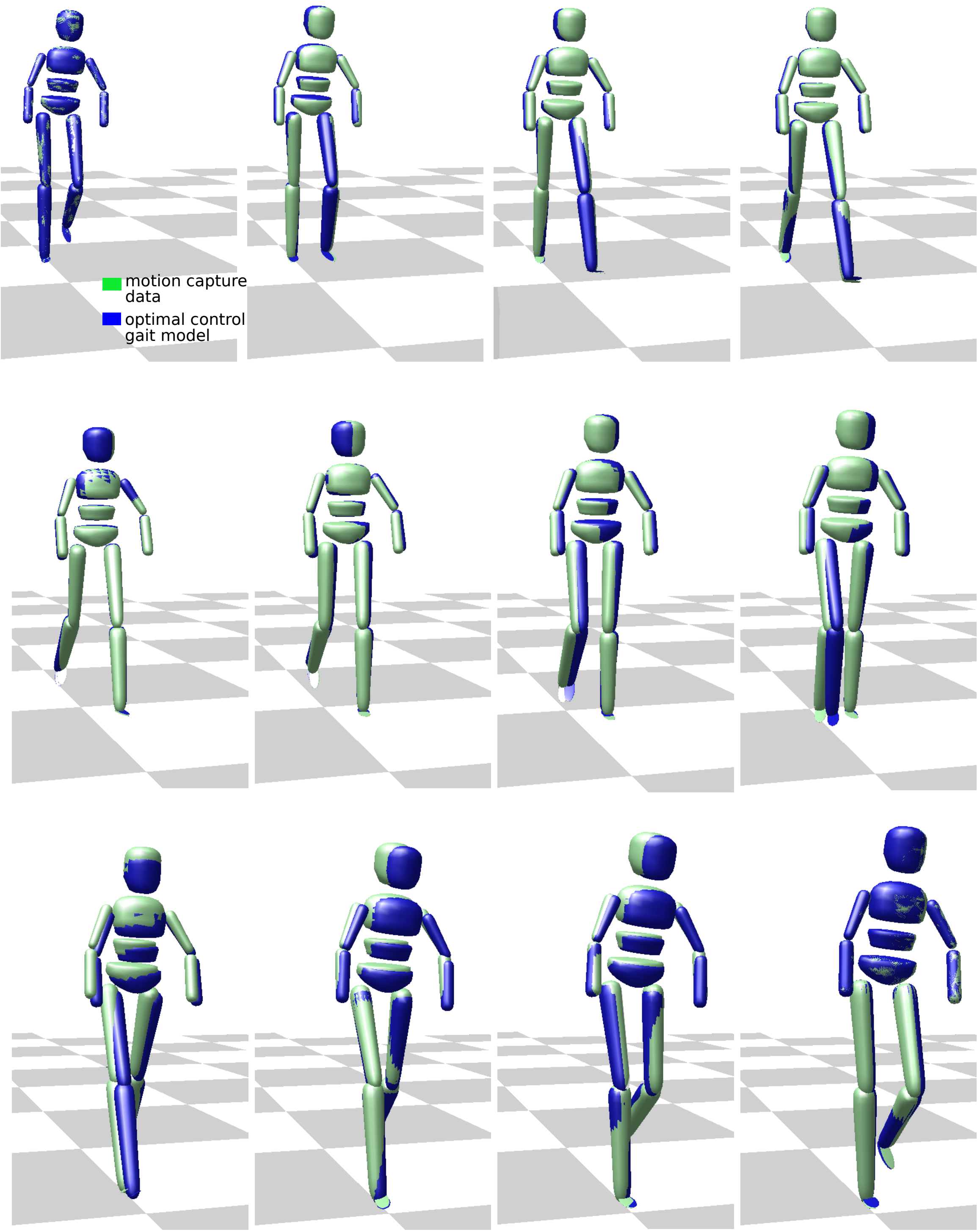


Figure 15.13.: Visualization of the measured gait compared to the solution of ((15.13), (15.3)-(15.6)) (the optimal control healthy gait model).

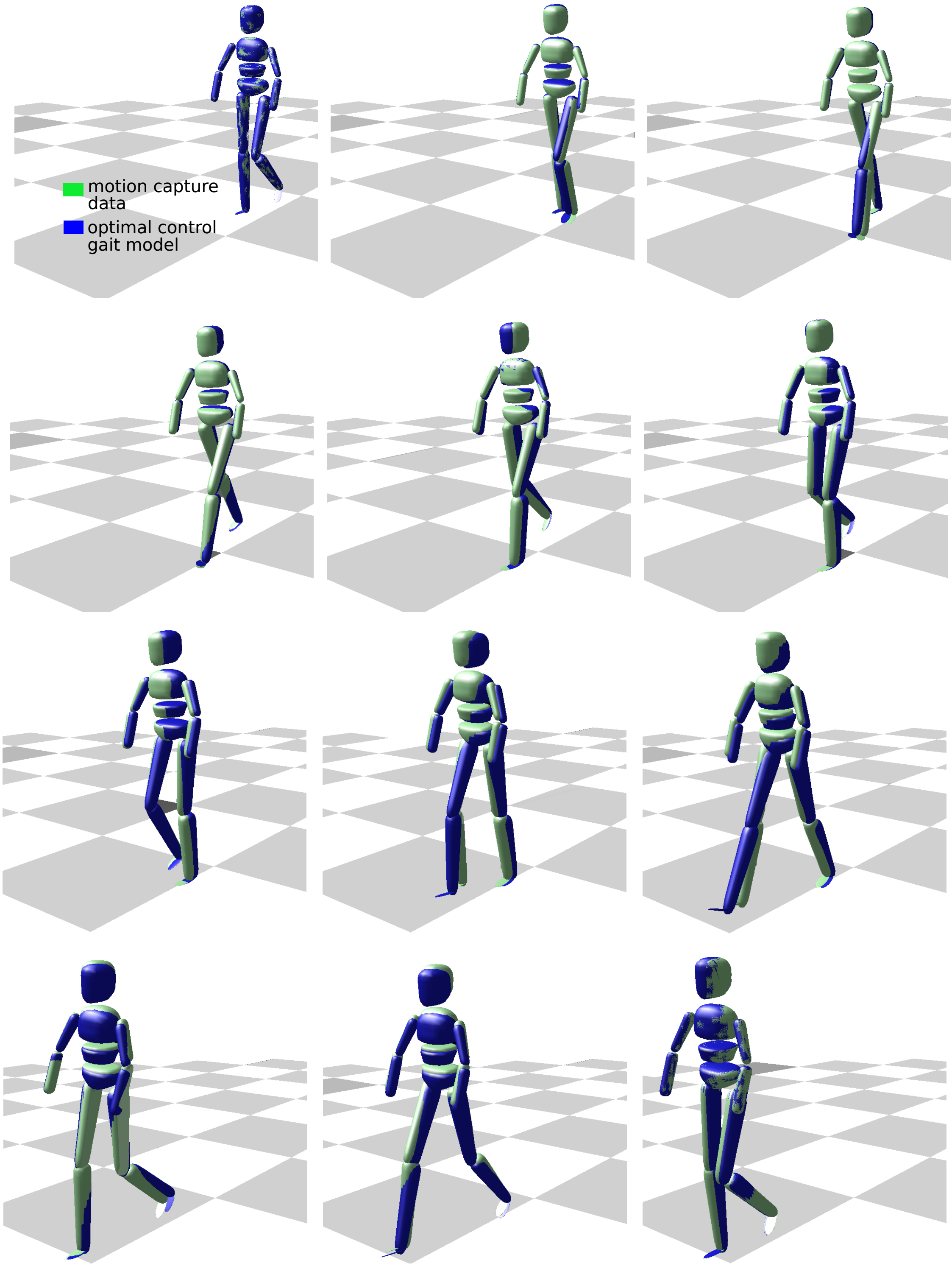


Figure 15.14.: Visualization of the measured gait compared to the solution of ((15.13), (15.3)-(15.6)) (the optimal control healthy gait model) from a second perspective.

Acronyms

BVP	Boundary Value Problem
CC	Complementarity Constraint
CP	Cerebral Palsy
CQ	Constraint Qualification
DFO	Derivative-Free Optimization
DOF	Degree Of Freedom
GGN	Generalized Gauß-Newton
IND	Internal Numerical Differentiation
IVP	Initial Value Problem
KKT	Karush-Kuhn-Tucker
LICQ	Linear Independence Constraint Qualification
MFCQ	Mangasarian-Fromowitz Constraint Qualification
MPBVP	Multi-Point Boundary Value Problem
MPCC	Mathematical Program with Complementarity Constraint
NLP	Nonlinear Program
OCM	Optimal Control Model
OCP	Optimal Control Problem
ODE	Ordinary Differential Equation
PMP	Pontryagin's Maximum Principle
SSOC	Strong Second-Order Condition
SQP	Sequential Quadratic Programming
QP	Quadratic Program

Nomenclature

General mathematical symbols

$\|\cdot\|_2$	Euclidean ℓ_2-norm.
$\|\cdot\|_1$	Euclidean ℓ_1-norm.
x	A column vector x.
$\nabla_x f(x)$	If f is a scalar function, $\nabla_x f(x)$ is the gradient (a column vector). If f is a vector function, $\nabla_x f(x)$ is the Jacobian (the rows are the gradients $\nabla_x f_i(x)$).
$\nabla^2_{xy} f(x,y)$	The Hessian of the scalar function f with respect to x and y.
$\nabla^2_x f(x)$	Short form for the Hessian of the scalar function f with respect to x: $\nabla^2_{xx} f(x)$.
$[a,b]$	The interval from a to b (including a and b).
(a,b)	The interval from a to b (excluding a and b).
$\dot{x}(t)$	The derivative of x with respect to the time t.
A^{-1}	The inverse of a matrix A.
$A^\intercal$, $a^\intercal$	The transpose of a matrix A or a vector a.
$x \geq 0$	All components of vector x are greater than or equal to zero.
$x = 0$	All components of vector x are equal to zero.
$x \perp y$	The vectors x and y of the same size are perpendicular, $x^T y = 0$.
$:=$	Denotes a definition.
$\mathbb{I}$, $\mathbb{I}_n$	The identity matrix of appropriate size or size $n \times n$.
$\mathbb{R}, \mathbb{R}^+, \mathbb{R}^+_0$	The set of real, positive real, nonnegative real numbers.
$\partial F(y)$	Subdifferential of F at y.
$\mathrm{conv}\{X\}$	Convex hull of a set X.
$\|X\|$	Cardinality of a set X.
$t \to 0$	0 is the limit of t.
$t \uparrow 0$	0 is the limit of t for $t \leq 0$ (left-hand limit).
$t \downarrow 0$	0 is the limit of t for $t \geq 0$ (right-hand limit).
e	Vector of ones.
e_i	Unit vector with the i-th entry being one.

Remaining symbols

For the sake of readability, we try to stay close to the standard notation used in the fields of nonlinear programming, mathematical programs with complementarity constraints, optimal control, parameter estimation, hierarchical dynamic optimization and multi-body systems. Therefore, some symbols have different meanings in different chapters. In the following, we provide a list of the main symbols used in this thesis and their meaning ordered by chapters. We furthermore note that throughout this thesis, dependencies of functions or sets are in some cases neglected for the sake of a compact presentation. The respective dependencies can be derived from the context.

Chapter 2:

f	The objective function of a nonlinear program.
g	Equality constraints of an optimization problem.
h	Inequality constraints of an optimization problem.
J^+	Generalized inverse.
r	Residual function of a least-squares problem.
x	Unknown in a nonlinear program or a least-squares problem.
λ	Lagrange multipliers corresponding to the equality constraints.
μ	Lagrange multipliers corresponding to the inequality constraints.
$\mathcal{A}(x)$	Active set containing indices of inequality constraints that are active at x.
$\mathcal{B}_\epsilon(x)$	Open ball with radius ϵ around x.
$\mathcal{E}$	Set of indices of the equality constraints.
$\mathcal{F}(x)$	Feasible set of a nonlinear program at x.
$\mathcal{I}$	Set of indices of the inequality constraints.
$\mathcal{L}$	Lagrangian.

Chapter 3, Chapter 6:

c	Mixed control-state constraints.
f	Right-hand side of an ordinary differential equation.
G	Derivative of a differential state x with respect to an initial value.
h	Model response function.
H	Hamiltonian.
q	Control values.
$\tilde{Q}$	Riccati matrix.
r	Two-point boundary conditions.
r^{eq}	Multi-point equality conditions.
r^{ieq}	Multi-point inequality conditions.
s	Multiple shooting state parameterization variables.
t	System time, $t \in I := [t_0, T]$.
$t_0, \cdots, t_{n_T}$	Time discretization.
u	Control function.
w	Control discretization variables.
x	Differential state function.
η	Measurement data.

γ	Weights in the objective function of the underlying optimal control problem.
λ	Dual variables.
μ	Dual variables corresponding to control constraints.
$\phi_{\mathcal{M}}$	Optimal control objective function of Mayer-type.
$\phi_{\mathcal{L}}$	Optimal control objective function of Lagrange-type.
Φ	Optimal control objective function of Bolza-type.
$\tilde{\Phi}$	Augmented objective function.

Chapter 4:

b^{upper}	Upper bounds on s, w, q.
b^{lower}	Lower bounds on s, w, q.
b_k	Stage transition condition between model stage k and $k+1$.
c_k	Mixed control-state constraints on model stage k.
f_k	Right-hand side of the ordinary differential equation on model stage k.
h_k	Model response function on model stage k.
p	Upper-level parameter.
q	Control values.
r_k^{eq}	Multi-point equality conditions on model stage k.
r_k^{ieq}	Multi-point inequality conditions on model stage k.
t	System time, $t \in I := [t_0, T]$.
u_k	Control function on model stage k.
x_k	Differential state function on model stage k.
η_k	Measurement data on model stage k.
γ	Upper-level parameters.
σ_k	Standard deviation on model stage k corresponding to measurement η_k.
$\tau_{k,0}, \cdots, \tau_{k,n_k}$	Time discretization on model stage k.
$\tau_{k,0}^m, \cdots, \tau_{k,m_k}^m$	Measurement times on model stage k.
$\phi_{\mathcal{M}}^{ki}$	Optimal control objective functions of Mayer-type on model stage k with $i = 1, \ldots, n_{\mathcal{M}}$.
$\phi_{\mathcal{L}}^{ki}$	Optimal control objective functions of Lagrange-type on model stage k with $i = n_{\mathcal{M}} + 1, \ldots, n_{\mathcal{M}} + n_{\mathcal{L}}$.

Chapter 5, Chapter 9, Chapter 13.4, Chapter 14:

b^{upper}	Upper bounds on s, w, q.
b^{lower}	Lower bounds on s, w, q.
c	Mixed control-state constraints.
C^{eq}	Summarized equality constraints.
C^{ieq}	Summarized inequality constraints (just simple bounds).
f	Right-hand side of the ordinary differential equation.
G_X	Derivative of a differential state x with respect to an initial value X.
h	Model response function.
I_i	Interval $[t_i, t_{i+1}]$.

J	Constraint Jacobian.
M_X	Derivative of the lower-level objective with respect to X.
q	Control values.
q^s	Slack variables.
$\bar{q}$	Control values and slack variables.
r^{eq}	Multi-point equality conditions.
$\hat{r}^{\mathrm{eq}}$	Multi-point equality conditions with a special structure.
$\tilde{r}^{\mathrm{eq}}$	Summarized equality constraints.
r^{ieq}	Multi-point inequality conditions.
$\hat{r}^{\mathrm{ieq}}$	Multi-point inequality conditions with a special structure.
R_X^{eq}	Derivative of equality constraints with respect to X.
R_X^{ieq}	Derivative of inequality constraints with respect to X.
$\hat{R}_X^{\mathrm{eq}}$	Derivative of structured equality constraints with respect to X.
$\hat{R}_X^{\mathrm{ieq}}$	Derivative of structured inequality constraints with respect to X.
s	Multiple shooting state parameterization variables.
t	System time, $t \in I := [t_0, T]$.
$t_0, \cdots, t_{n_T}$	Time discretization.
u	Control function.
$\bar{u}$	Approximation of the control function u described by ξ_i on I_i.
w	Control discretization variables.
x	Differential state function.
y	Summary of lower-level variables.
z	Summary of lower-level and upper-level variables.
η	Measurement data.
γ	Weights in the objective function of the underlying optimal control problem.
λ	Lagrange multipliers corresponding to equality constrains.
$\mathcal{L}$	Lagrangian.
μ	Lagrange multipliers corresponding to inequality constrains.
$\phi_{\mathcal{M}}^i$	Optimal control objective function of Mayer-type with $i = 1, \ldots, n_{\mathcal{M}}$.
$\tilde{\Phi}$	Discretized optimal control objective function.
$\mathfrak{F}$	Residual in a constrained least-squares problem.
$\mathfrak{G}$	Equality constraints in a constrained least-squares problem.

Chapter 7:

$\bar{F}/F$	Least-squares objective/least-squares residual.
g_i	Jacobian of F_i.
H_i	Hessian of F_i.
m	Quadratic model of $\bar{F}$.
q	Quadratic model of F_i.
y	Variables in a constrained least-squares problem.
$\mathcal{Y}$	Set of interpolation points.

Chapter 8, Chapter 13.1-13.3:

f	The objective function of a mathematical program with complementarity constraint/a lower-level nonlinear program.
F	Upper-level objective in a bilevel nonlinear program.
g	Equality constraints of an optimization problem.
h	Inequality constraints of an optimization problem.
P	Penalty term.
v	Lifting variables.
x	Unknown in a mathematical program with complementarity constraint/upper-level variable in a bilevel nonlinear program.
y	Lower-level variable in a bilevel nonlinear program.
λ, μ	Lagrange multipliers.
$\mathcal{L}$	Lagrangian.
$\nu_1, \nu_2, \xi, \nu, z$	Lagrange multipliers.
π	Penalty parameter.
$\mathcal{A}$	Active set containing indices of inequality constraints that are active.

Chapter 10:

$\varkappa$	Contact forces.
ζ	Contact Hessian.
$\mathfrak{G}$	Contact Jacobian.
$\mathfrak{M}$	Mass matrix.
$\mathfrak{N}$	Nonlinear forces.
$\mathfrak{q}$	Generalized coordinates.
$\mathfrak{R}$	Rotation matrix.
$\bar{\mathfrak{R}}$	Augmented rotation matrix.
$\mathfrak{T}$	Translation.
$\bar{\mathfrak{T}}$	Augmented translation matrix.

Chapter 12, Chapter 15:

b_k	Stage transition condition between model stage k and $k+1$.
c_k	Mixed control-state constraints on model stage k.
f_k	Right-hand side of the ordinary differential equation on model stage k.
h_k	Model response function on model stage k.
q	Control values.
r_k^{eq}	Multi-point equality conditions on model stage k.
r_k^{ieq}	Multi-point inequality conditions on model stage k.
t	System time, $t \in I := [t_0, T]$.
T_{right}	Position of the left contact point in the global coordinate system.
T_{left}	Position of the right contact point in the global coordinate system.
u_k	Control function on model stage k.
x_k	Differential state function on model stage k.
η_k	Measurement data on model stage k.
σ_k	Standard deviations on model stage k

	corresponding to measurements η_k.
$\varkappa$	Contact forces.
$\phi^{ki}_{\mathcal{M}}$	Optimal control objective function of Mayer-type on model stage k with $i = 1, \dots, n_{\mathcal{M}}$.
$\phi^{ki}_{\mathcal{L}}$	Optimal control objective function of Lagrange-type on model stage k with $i = n_{\mathcal{M}} + 1, \dots, n_{\mathcal{M}} + n_{\mathcal{L}}$.
γ	Weights in the objective function of the underlying optimal control problem.
Λ_{onefoot}	Momentum acting on one foot at the touchdown.
Λ_{left}	Momentum acting on the left foot at the touchdown.
Λ_{right}	Momentum acting on the right foot at the touchdown.
$\tau_{k,0}, \cdots, \tau_{k,n_k}$	Time discretization on model stage k.
$\tau^m_{k,0}, \cdots, \tau^m_{k,n_k}$	Measurement times on model stage k.
ζ_{right}	Contact Hessian of the right foot.
ζ_{left}	Contact Hessian of the left foot.
$\mathfrak{G}_{\text{onefoot}}$	Contact Jacobian of one contact point.
$\mathfrak{G}_{\text{right}}$	Contact Jacobian of the right foot.
$\mathfrak{G}_{\text{left}}$	Contact Jacobian of the left foot.
$\mathfrak{M}$	Mass matrix.
$\mathfrak{N}$	Nonlinear forces.
$\mathfrak{q}$	Generalized coordinates.

List of figures

List of tables

List of algorithms

Bibliography

[ABB+99] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 3rd edition, 1999.

[ABEvS93] T. Andrzejewski, H.G. Bock, E. Eich, and R. von Schwerin. Recent Advances in the Numerical Integration of Multibody Systems. In W. Schiehlen, editor, *Advanced Multibody System Dynamics, Simulation and Software Tools*, pages 127–151. Kluwer Academic Publishers, Dordrecht, Boston, London, 1993.

[AD05] A.S. Arnold and S.L. Delp. Computer modeling of gait abnormalities in cerebral palsy: application to treatment planning. *Theoretical Issues in Ergonomics Science*, 6(3–4):305–312, 2005.

[Alb10] J. Albersmeyer. *Adjoint based algorithms and numerical methods for sensitivity generation and optimization of large scale dynamic systems.* PhD thesis, Ruprecht-Karls-Universität Heidelberg, 2010.

[ALB12] N. Aghasadeghi, A. Long, and T. Bretl. Inverse optimal control for a hybrid dynamical system with impacts. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4962–4967, 2012.

[Ale84] R.M. Alexander. The Gaits of Bipedal and Quadrupedal Animals. *International Journal of Robotics Research*, 3(2):49–59, 1984.

[Ale96] R.M. Alexander. *Optima for Animals.* Princeton University Press, Princeton, NJ, 1996.

[Ale97] R.M. Alexander. A minimum energy cost hypothesis for human arm trajectories. *Biological Cybernetics*, 76(2):97 – 105, 1997.

[ALS+06] A.S. Arnold, M.Q. Liu, M.H. Schwartz, S. Ounpuu, L.S. Dias, and S.L. Delp. Do the hamstrings operate at increased muscle-tendon lengths and velocities after surgical lengthening? *Journal of Biomechanics*, 39(8):1498 – 506, 2006.

[ALU12] S. Albrecht, M. Leibold, and M. Ulbrich. A bilevel optimization approach to obtain optimal cost functions for human arm movements. *Numerical Algebra, Control and Optimization (NACO)*, 2(1):105 – 127, 2012.

[Ani05] M. Anitescu. On Using the Elastic Mode in Nonlinear Programming Approaches to Mathematical Programs with Complementarity Constraints. *SIAM Journal on Optimization*, 15(4):1203 – 1236, 2005.

[APBLU12] S. Albrecht, S. Glasauer P. Basili, M. Leibold, and M. Ulbrich. Modeling and Analysis of Human Navigation with Crossing Interferer Using Inverse Optimal Control. In *Proceedings of the 7th Vienna International Conference on Mathematical Modelling (Math Mod)*, 2012.

[APS+10] S. Albrecht, C. Passenberg, M. Sabotka, A. Peer, M. Buss, and M. Ulbrich. Optimization Criteria for Human Trajectory Formation in Dynamic Virtual Environments. In *Haptics: Generating and Perceiving Tangible Sensations, Lecture Notes in Computer Science*, volume 6192, pages 257 – 262, 2010.

[ARARU+11] S. Albrecht, K. Ramìrez-Amaro, F. Ruiz-Ugalde, D. Weikersdorfer, M. Leibold, M. Ulbrich, and M. Beetz. Imitating human reaching motions using physically inspired optimization principles. In *Proceedings of the 11th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 602 – 607, 2011.

[AS13] G.A. Allende and G. Still. Solving bilevel programs with the KKT-approach. *Mathematical Programming*, 138(1–2):309 – 332, 2013.

[ATW07] M. Anitescu, P. Tseng, and S. Wright. Elastic-mode algorithms for mathematical programs with equilibrium constraints: global convergence and stationarity properties. *Mathematical Programming*, 110(2):337 – 371, 2007.

[Ayy97] E. Ayyappa. Normal Human Locomotion, Part 1: Basic Concepts and Terminology. *JPO Journal of Prosthetics & Orthotics*, 9(1):10 – 17, 1997.

[Bar74] Y. Bard. *Nonlinear Parameter Estimation.* Academic Press, New York, 1974.

[Bar88] J.F. Bard. Convex two-level optimization. *Mathematical Programming*, 40(1-3):15 – 27, 1988.

[Bar10] J.F. Bard. *Practical Bilevel Optimization: Algorithms and Applications.* Springer, Heidelberg, 2010.

[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*, pages 295 – 340. Springer, Berlin, 2001.

[BC95] R. Beckett and K. Chang. An Evaluation of the Kinematics of Gait by Minimum Energy. In D. Bootzin and H. Muffley, editors, *Biomechanics*, pages 15 – 26. Springer, New York, 1995.

[BCS07] P. Marcotte B. Colson and G. Savard. An overview of bilevel optimization. *Annals of Operation Research*, 153(1):235 – 256, 2007.

[BDD+02] L. S. Blackford, J. Demmel, J. Dongarra, I. Duff, and et. al. An updated set of basic linear algebra subprograms (BLAS). *ACM Trans. Math. Soft.*, 28(2):135 – 151, 2002.

[Ber03] D.P. Bertsekas. *Nonlinear Programming.* Athena Scientific, Belmont, MA, 2003.

[BES88] H.G. Bock, E. Eich, and J.P. Schlöder. Numerical solution of constrained least squares boundary value problems in differential-algebraic equations. In K. Strehmel, editor, *Numerical Treatment of Differential Equations.* Teubner, Leipzig, 1988.

[BH75] A.E. Bryson and Y.-C. Ho. *Applied Optimal Control.* Wiley, New York, 1975.

[Bie10] L.T. Biegler. *Nonlinear Programming: Concepts, Algorithms and Applications to Chemical Processes.* SIAM, Philadelphia, PA, 2010.

[BKS00] H.G. Bock, E.A. Kostina, and J.P. Schlöder. On the role of natural level functions to achieve global convergence for damped Newton methods. In M. Powell et al., editor, *System Modelling and Optimization. Methods, Theory and Applications.* Kluwer, Dordrecht, 2000.

[BKS07] H.G. Bock, E. Kostina, and J.P. Schlöder. Numerical Methods for Parameter Estimation in Nonlinear Differential Algebraic Equations. *GAMM Mitteilungen*, 30(2):352 – 375, 2007.

[BM73] J. Bracken and J. T. McGill. Mathematical Programs with Optimization Problems in the Constraints. *Operations Research*, 21(1):37 – 44, 1973.

[Boc74] H.G. Bock. Numerische Optimierung zustandsbeschränkter parameterabhängiger Prozesse mit linear auftretender Steuerung unter Anwendung der Mehrzielmethode. Master's thesis, Universität zu Köln, 1974.

[Boc78] H.G. Bock. Numerical Solution of Nonlinear Multipoint Boundary Value Problems with Applications to Optimal Control. *Zeitschrift für Angewandte Mathematik und Mechanik*, 58:407, 1978.

[Boc81a] H.G. Bock. Numerical Treatment of Inverse Problems in Chemical Reaction Kinetics. In K.H. Ebert, P. Deuflhard, and W. Jäger, editors, *Modelling of Chemical Reaction Systems*, volume 18 of *Springer Series in Chemical Physics*, pages 102 – 125. Springer, Heidelberg, 1981.

[Boc81b] H.G. Bock. Numerische Behandlung von zustandsbeschränkten und Chebyshev-Steuerungsproblemen. Technical Report R106/81/11, Carl Cranz Gesellschaft, Heidelberg, 1981.

[Boc83] H.G. Bock. Recent advances in parameter identification techniques for ODE. In P. Deuflhard and E. Hairer, editors, *Numerical Treatment of Inverse Problems in Differential and Integral Equations*, pages 95 – 121. Birkhäuser, Boston, MA, 1983.

[Boc87] H.G. Bock. *Randwertproblemmethoden zur Parameteridentifizierung in Systemen nichtlinearer Differentialgleichungen*, volume 183 of *Bonner Mathematische Schriften.* Universität Bonn, 1987.

[BP84] H.G. Bock and K.J. Plitt. A multiple shooting algorithm for direct solution of optimal control problems. In *Proceedings 9th IFAC World Congress Budapest*, pages 242 – 247. Pergamon Press, Oxford, 1984.

[BPL+06] H.G. Bock, K.J. Plitt, D. Leineweber, M. Diehl, and A. Schäfer. MUSCOD: An SQP multiple shooting code for optimal control and design problems in ODE and DAE, Interdisciplinary Center for Scientific Computing, Ruprecht-Karls Universität Heidelberg, 2006.

[BPS06] J.E. Bobrow, F.C. Park, and A. Sideris. Recent advances on the algorithmic optimization of robot motion. In Moritz Diehl and Katja Mombaur, editors, *Fast Motions in Biomechanics and Robotics*, volume 340 of *Lecture Notes in Control and Information Sciences*, pages 21 – 41. Springer, Berlin, Heidelberg, 2006.

[BSSV06] H. Benson, A. Sen, D. Shanno, and R. Vanderbei. Interior-Point Algorithms, Penalty Methods and Equilibrium Problems. *Computational Optimization and Applications*, 34(2):155 – 182, 2006.

[BTK13] Biomechanical toolkit package BTK. http://code.google.com/p/b-tk/, 2013.

[Bul71] R. Bulirsch. Die Mehrzielmethode zur numerischen Lösung von nichtlinearen Randwertproblemen und Aufgaben der optimalen Steuerung. Technical report, Carl-Cranz-Gesellschaft, Oberpfaffenhofen, 1971.

[Cas80] J. Casti. On the general inverse problem of optimal control theory. *Journal of Optimization Theory and Applications*, 32(4):491 – 497, 1980.

[Cla90] F.H. Clarke. *Optimization and Nonsmooth Analysis.* Classics in Applied Mathematics. SIAM, Philadelphia, PA, 1990.

[Cou38] A.A. Cournot. *Recherches sur les principes mathématiques de la théorie des richesses.* L. Hachette, 1838.

[Cra89] J.J. Craig. *Introduction to Robotics: Mechanics and Control.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 2nd edition, 1989.

[CSV09] A. Conn, K. Scheinberg, and L. Vicente. Introduction to Derivative-Free Optimization. In *MPS/SIAM Series on Optimization.* SIAM, Philadelphia, PA, 2009.

[DAA+07] S. Delp, F. Anderson, A. Arnold, P. Loan, A. Habib, C. John, E. Guendelman, and D. Thelen. OpenSim: Open-source software to create and analyze dynamic simulations of movement. *IEEE Trans. Biomed. Engineering*, 54(11):1940 – 1950, 2007.

[DBV+13] T. Dreher, R. Brunner, D. Vegvari, D. Heitzmann, S. Gantz, M. Maier, F. Braatz, and S. Wolf. The effects of muscle-tendon surgery on dynamic electromyographic patterns and muscle tone in children with cerebral palsy. *Gait & Posture*, 38(2):215 – 220, 2013.

[DBW$^+$12] T. Dreher, T. Buccoliero, S. Wolf, D. Heitzmann, S. Gantz, F. Braatz, and W. Wenz. Long-term results after gastrocnemius-soleus intramuscular aponeurotic recession as a part of multilevel surgery in spastic diplegic cerebral palsy. *The Journal of Bone & Joint Surgery*, 94(7):627 – 637, 2012.

[DDW$^+$08] H. Diedam, D. Dimitrov, P. B Wieber, K. Mombaur, and M. Diehl. Online walking gait generation with adaptive foot positioning through linear model predictive control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008 (IROS 2008)*, pages 1121 – 1126, 2008.

[Dem02] S. Dempe. *Foundations of Bilevel Programming.* Kluwer Academic Publishers, Dordrecht, 2002.

[Dem03] S. Dempe. Bilevel programming - a survey. *Preprint TU Bergakademie Freiberg Nr. 2003-11*, 2003.

[DFNS05] A.-V. DeMiguel, M. P. Friedlander, F. J. Nogales, and S. Scholtes. A two-sided relaxation scheme for mathematical programs with equilibrium constraints. *SIAM Journal on Optimization*, 16(1):587 – 609, 2005.

[DGW$^+$12] T. Dreher, M. Götze, S. Wolf, S. Hagmann, D. Heitzmann, S. Gantz, and F. Braatz. Distal rectus femoris transfer as part of multilevel surgery in children with spastic diplegia - a randomized clinical trial. *Gait & Posture*, 36(2):212 – 218, June 2012.

[Dic99] M. H. Dickinson. Bionics: Biological insight into mechanical design. *Proceedings of the National Academy of Sciences*, 96(25):14208 – 14209, 1999.

[Die01] M. Diehl. *Real-Time Optimization for Large Scale Nonlinear Processes.* PhD thesis, Ruprecht-Karls-Universität Heidelberg, 2001.

[dL96] P. de Leva. Adjustments to Zatsiorsky-Seluyanov's segment inertia parameters. *Journal of Biomeachanics*, 29(9):1223 – 1230, 1996.

[DLS01] M. Diehl, D.B. Leineweber, and A.A.S. Schäfer. MUSCOD-II Users' Manual. IWR-Preprint 2001-25, Ruprecht-Karls-Universität Heidelberg, 2001.

[DM02] E. Dolan and J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201 – 213, 2002.

[DMBG07] F. Dobson, M.E. Morris, R. Baker, and H.K. Graham. Gait classification in children with cerebral palsy: a systematic review. *Gait & Posture*, 25(1):140 – 52, 2007.

[Död07] L. Döderlein. *Infantile Zerebralparese.* Steinkopff Verlag, Heidelberg, 2007.

[DS13] A.V. Dmitruk and I.A. Samylovskiy. Optimal synthesis in the Reeds and Shepp problem with a free final direction. *Journal of Dynamical and Control Systems*, 19(3):309 – 325, 2013.

[DZ13] S. Dempe and A.B. Zemkoho. The bilevel programming problem: reformulations, constraint qualifications and optimality conditions. *Mathematical Programming*, 138(1–2):447 – 473, 2013.

[Fel09] M. Felis. Modellierung, Optimierung und Visualisierung menschlicher Gehbewegungen. Master's thesis, Ruprecht-Karls-Universität Heidelberg, 2009.

[Fel12] M. Felis. Rigid Body Dynamics Library. http://rbdl.bitbucket.org/, 2012.

[Fel13] M. Felis. Meshup. https://bitbucket.org/MartinFelis/meshup, 2013.

[FGK02] R. Fourer, D. Gay, and B. Kernighan. *AMPL: A Modeling Language for Mathematical Programming.* Duxbury Press, Pacific Grove, CA, 2002.

[FL02a] R. Fletcher and S. Leyffer. Nonlinear programming without a penalty function. *Mathematical Programming*, 91(2):239 – 269, 2002.

[FL02b] R. Fletcher and S. Leyffer. Numerical experience with solving MPECs as NLPs. Technical report, Numerical Analysis Report NA/210, Department of Mathematics, University of Dundee, Dundee, UK, 2002.

[Fle87] R. Fletcher. *Practical Methods of Optimization.* Wiley, Chichester, 2nd edition, 1987.

[Fle05] M.L. Flegel. *Constraint Qualifications and Stationarity Concepts for Mathematical Programs with Equilibrium Constraints.* PhD thesis, University of Würzburg, 2005.

[FLHS10] F. Fisch, J. Lenz, F. Holzapfel, and G. Sachs. On the solution of bilevel optimal control problems to increase the fairness in air races. In *Proceedings of the AIAA Atmospheric Flight Mechanics Conference*, 2010.

[FLRS06] R. Fletcher, S. Leyffer, D. Ralph, and S. Scholtes. Local convergence of SQP methods for mathematical programs with equilibrium constraints. *SIAM Journal on Optimization*, 17(1):259 – 286, 2006.

[FM12] M. Felis and K. Mombaur. Using optimal control methods to generate human walking motions. In Marcelo Kallmann and Kostas Bekris, editors, *Motion in Games*, volume 7660 of *Lecture Notes in Computer Science*, pages 197 – 207. Springer, Heidelberg, 2012.

[FMKB13] M. Felis, K. Mombaur, H. Kadone, and A. Berthoz. Modeling and identification of emotional aspects of locomotion. *Journal of Computational Science*, 4(4):255 – 261, 2013.

[FO00] R. Featherstone and D. Orin. Robot dynamics: equations and algorithms. In *IEEE International Conference on Robotics and Automation (ICRA 2000)*, volume 1, pages 826 – 834, 2000.

[FRGS00] D. Fehlings, M. Rang, J. Glazier, and C. Steele. An evaluation of botulinum - A toxin injections to improve upper extremity function in children with hemiplegic cerebral palsy. *The Journal of Pediatrics*, 137(3):331 – 337, 2000.

[Gag91] J.R. Gage. *Gait Analysis in Cerebral Palsy.* Clinics in Developmental Medicine. MacKeith Press, London, 1991.

[GCRC98] M. Garcia, A. Chatterjee, A. Ruina, and M. Coleman. The simplest walking model: Stability, complexity, and scaling. *ASME Journal of Biomechanical Engineering*, 120(2):281 – 288, 1998.

[Ger12] M. Gerdts. *Optimal Control of ODEs and DAEs*. De Gruyter, Berlin, 2012.

[GFK09] K. E. Gordon, D. P. Ferris, , and A. D. Kuo. Metabolic and mechanical energy costs of reducing vertical center of mass movement during gait. *Archives of Physical Medicine and Rehabilitation*, 90(1):136 – 144, 2009.

[GL18] H. Gray and W.H. Lewis. *Anatomy of the human body*. Lea & Febiger, Philadelphia, PA, 1918.

[GNU] The GNU General Public License (GPL). Available from http://www.gnu.org/licenses/gpl.html.

[GSB06] H. Geyer, A. Seyfarth, and R. Blickhan. Compliant leg behaviour explains basic dynamics of walking and running. *Proceedings of the Royal Society B: Biological Sciences*, 273(1603):2861 – 2867, 2006.

[Har82] P. Hartman. *Ordinary differential equations*. SIAM, Philadelphia, PA, 1982.

[Hat08] K. Hatz. Estimating parameters in optimal control problems. Master's thesis, Ruprecht-Karls-Universität Heidelberg, 2008.

[Hes66] M.R. Hestenes. *Calculus of variations and optimal control theory*. Wiley, New York, 1966.

[HKS13] T. Hoheisel, C. Kanzow, and A. Schwartz. Theoretical and numerical comparison of relaxation methods for mathematical programs with complementarity constraints. *Mathematical Programming*, 137(1-2):257 – 288, 2013.

[HLSB13] K. Hatz, S. Leyffer, J.P. Schlöder, and H.G. Bock. Regularizing bilevel nonlinear programs by lifting. Technical Report ANL/MCS-P4076-0613, Argonne National Laboratory, Mathematics and Computer Science Division, 2013.

[HNW00] E. Hairer, S. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations I*. Springer, Berlin, Heidelberg, 2000.

[HP88] P. T. Harker and J.-S. Pan. Existence of optimal solutions to mathematical programs with equilibrium constraints. *Operations Research Letters*, 7(2):61 – 64, 1988.

[HR71] G.A. Hicks and W.H. Ray. Approximation methods for optimal control synthesis. *Canadian Journal of Chemical Engineering*, 49(4):522 – 528, 1971.

[HSB12] K. Hatz, J.P. Schlöder, and H.G. Bock. Estimating parameters in optimal control problems. *SIAM Journal on Scientific Computing*, 34(3):A1707 – A1728, 2012.

[HSV95] R.F. Hartl, S.P. Sethi, and R.G. Vickson. A survey of the maximum principles for optimal control problems with state constraints. *SIAM Review*, 37(2):181 – 218, 1995.

[INR05] INRIA. Humans toolbox. http://www.inrialpes.fr/bipop/software/humans/, 2005.

[IPS11] A. F. Izmailov, A. L. Pogosyan, and M. V. Solodov. Semismooth SQP method for equality-constrained optimization problems with an application to the lifted reformulation of mathematical programs with complementarity constraints. *Optimization Methods and Software*, 26(4 – 5):847 – 872, 2011.

[Kal64] R. Kalman. When is a linear control system optimal? *Journal of Basic Engineering*, 86(1):51 – 60, 1964.

[KAS$^+$10] S. Kraus, S. Albrecht, M. Sobotka, B. Heissing, and M. Ulbrich. Optimisation-based identification of situation determined cost functions for the implementation of a human-like driving style in an autonomous car. In *Proceedings of AVEC 10 (10th International Symposium on Advanced Vehicle Control)*, pages 412 – 417, 2010.

[KB06] M. Knauer and C. Büskens. Bilevel optimization of container cranes. In *5th MATHMOD Vienna International Conference on Mathematical Modelling*. 2006. ARGESIM Report Nr. 30(2):SP18.3.1.

[Kir06] C. Kirches. A numerical method for nonlinear robust optimal control with implicit discontinuities and an application to powertrain oscillations. Master's thesis, Ruprecht-Karls-Universität Heidelberg, October 2006.

[Kiw90] K. C. Kiwiel. Proximity control in bundle methods for convex nondifferentiable minimization. *Mathematical Programming*, 46(1-3):105 – 122, 1990.

[KLM$^+$10] M. Kortelainen, T. Lesinski, J. Moré, W. Nazarewicz, J. Sarich, N. Schunck, M. V. Stoitsov, and S. Wild. Nuclear energy density optimization. *Physical Review C*, 82(2):024313, Aug 2010.

[KML10] A. Truong K. Mombaur and J.-P. Laumond. From human to humanoid locomotion – an inverse optimal control approach. *Autonomous Robots*, 28(3):369 – 383, 2010.

[Kna09] M. Knauer. *Bilevel-Optimalsteuerung mittels hybrider Lösungsmethoden am Beispiel eines deckengeführten Regalbediengerätes in einem Hochregallager*. PhD thesis, Universität Bremen, 2009.

[Kna12] M. Knauer. Fast and save container cranes as bilevel optimal control problems. *Mathematical and Computer Modelling of Dynamical Systems*, 18(4):465 – 486, 2012.

[Kos08] J. Koschorrek. Modellierung und Optimierung von Salti und Schrauben beim Turmspringen. Master's thesis, Ruprecht-Karls-Universität Heidelberg, 2008.

[Kuo07] A. Kuo. The six determinants of gait and the inverted pendulum analogy: A dynamic walking perspective. *Human Movement Science*, 26(4):617 – 656, 2007.

[Kö02] S. Körkel. *Numerische Methoden für Optimale Versuchsplanungsprobleme bei nichtlinearen DAE-Modellen.* PhD thesis, Ruprecht-Karls-Universität Heidelberg, 2002.

[LBS+03] D.B. Leineweber, I. Bauer, A.A.S. Schäfer, H.G. Bock, and J.P. Schlöder. An efficient multiple shooting based reduced SQP strategy for large-scale dynamic process optimization (Parts I and II). *Computers and Chemical Engineering*, 27:157 – 174, 2003.

[Lei96] D.B. Leineweber. The theory of MUSCOD in a nutshell. IWR-Preprint 96-19, Ruprecht-Karls-Universität Heidelberg, 1996.

[Lei99] D.B. Leineweber. *Efficient reduced SQP methods for the optimization of chemical processes described by large sparse DAE models*, volume 613 of *Fortschritt-Berichte VDI Reihe 3, Verfahrenstechnik.* VDI Verlag, Düsseldorf, 1999.

[Lem81] C. Lemarechal. A view of line-searches. In Alfred Auslender, Werner Oettli, and Josef Stoer, editors, *Optimization and Optimal Control*, volume 30 of *Lecture Notes in Control and Information Sciences*, pages 59 – 78. Springer, Berlin, Heidelberg, 1981.

[Ley00] S. Leyffer. MacMPEC: AMPL collection of MPECs. www.mcs.anl.gov/leyffer/MacMPEC/, 2000.

[Ley03] S. Leyffer. Complementarity constraints as nonlinear equations: Theory and numerical experiences. Technical Report ANL/MCS-P1054-0603, Argonne National Laboratory, June 2003.

[LF03] G.H. Lin and M. Fukushima. New relaxation method for mathematical programs with complementarity constraints. *Journal of Optimization Theory and Applications*, 118(1):81 – 116, 2003.

[LLCN06] S. Leyffer, G. Lòpez-Calva, and J. Nocedal. Interior methods for mathematical programs with complementarity constraints. *SIAM Journal on Optimization*, 17(1):52 – 77, 2006.

[LM07] S. Leyffer and T.S Munson. A Globally Convergent Filter Method for MPECs. Technical Report ANL/MCS-P1457-0907, Argonne National Laboratory, Mathematics and Computer Science Division, 2007.

[LPR96] Z. Luo, J. Pang, and D. Ralph. *Mathematical Programs with Equlibrium Constraints.* Cambridge University Press, Cambridge, 1996.

[LPR97] Z. Luo, J. Pang, and D. Ralph. *Multilevel Optimization: Algorithms and Applications*, chapter 9, pages 209 – 228. Kluwer Academic Publishers, Dordrecht, 1997.

[Mas68] M. Masak. An inverse problem on decoupling optimal control systems. *IEEE Trans. Autom. Control*, 13(1):109 – 110, 1968.

[Mat13] The MathWorks. Homepage. http://www.mathworks.com/, 2013.

[MBLS01] K. Mombaur, H.G. Bock, R.W. Longman, and J.P. Schlöder. Human-like actuated walking that is asymptotically stable without feedback. In *Proc. International Conference on Robotics and Automation*, 2001.

[MFS12] P. Moreira, P. Flores, and M. Silva. A biomechanical multibody foot model for forward dynamic analysis. In *IEEE 2nd Portuguese Meeting in Bioengineering (ENBENG)*, pages 1 – 6, 2012.

[Mir99] J. Mirrlees. The theory of moral hazard and unobservable behaviour: Part I. *Review of Economic Studies*, 66(1):3 – 21, 1999.

[Mit98] M. Mitchell. *An Introduction to Genetic Algorithms.* MIT Press, Cambridge, MA, USA, 1998.

[MO02] H. Maurer and H. Oberle. Second Order Sufficient Conditions for Optimal Control Problems with Free Final Time: The Riccati Approach. *SIAM Journal on Control and Optimization*, 41(2):380 – 403, 2002.

[MOC13] K. Mombaur, A. Olivier, and A. Crètual. Forward and inverse optimal control of bipedal running. In K. Mombaur and K. Berns, editors, *Modeling, Simulation and Optimization of Bipedal Walking*, volume 18 of *Cognitive Systems Monographs*, pages 165 – 179. Springer, Berlin, Heidelberg, 2013.

[Mom01] K.D. Mombaur. *Stability Optimization of Open-loop Controlled Walking Robots.* PhD thesis, Ruprecht-Karls-Universität Heidelberg, 2001.

[MS10] K. Mombaur and M. Sreenivasa. Inverse optimal control as a tool to understand human yoyo playing. In *ICNAAM 2010: International Conference of Numerical Analysis and Applied Mathematics*, 2010.

[MW09] J. Moré and S. Wild. Benchmarking derivative-free optimization algorithms. *SIAM Journal on Optimization*, 20(1):172 – 191, 2009.

[Neu76] L.W. Neustadt. *Optimization: A Theory of Necessary Conditions.* Princeton University Press, Princeton, NJ, 1976.

[NM65] J.A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308 – 313, 1965.

[NW99] J. Nocedal and S.J. Wright. *Numerical Optimization.* Springer, Heidelberg, 1999.

[OG89] H.J. Oberle and W. Grimm. *BNDSCO: A Program for the Numerical Solution of Optimal Control Problems.* DLR IB / 515: DLR IB. Inst. für Dynamik der Flugsysteme, DLR, 1989.

[OKZ98] J.V. Outrata, Michal Kočvara, and J. Jochem Zowe. *Nonsmooth approach to optimization problems with equilibrium constraints : theory, applications, and numerical results.* Nonconvex optimization and its applications. Kluwer Academic Publishers, Dordrecht, Boston, London, 1998.

[Osb69] M.R. Osborne. On shooting methods for boundary value problems. *Journal of Mathematical Analysis and Applications*, 27:417 – 433, 1969.

[PBGM62] L.S. Pontryagin, V.G. Boltyanski, R.V. Gamkrelidze, and E.F. Miscenko. *The Mathematical Theory of Optimal Processes.* Wiley, Chichester, 1962.

[PBS09] A. Potschka, H.G. Bock, and J.P. Schlöder. A minima tracking variant of semi-infinite programming for the treatment of path constraints within direct solution of optimal control problems. *Optimization Methods and Software*, 24(2):237 – 252, 2009.

[PH05] M. Popovic and H. Herr. Ground reference points in legged locomotion: Definitions, biological trajectories and control implications. *International Journal of Robotics Research*, 24(12):1013 – 1032, 2005.

[PJJB12] A.-S. Puydupin-Jamin, M. Johnson, and T. Bretl. A convex approach to inverse optimal control and its application to modeling human locomotion. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 531 – 536, 2012.

[Pli81] K.J. Plitt. Ein superlinear konvergentes Mehrzielverfahren zur direkten Berechnung beschränkter optimaler Steuerungen. Master's thesis, Universität Bonn, 1981.

[Pot06] A. Potschka. Handling path constraints in a direct multiple shooting method for optimal control problems. Master's thesis, Ruprecht-Karls-Universität Heidelberg, 2006.

[Pow08] M.J.D. Powell. Developments of NEWUOA for unconstrained minimization without derivatives. *IMA Journal of Numerical Analysis*, 28(4):643 – 664, 2008.

[Pow09] M.J.D. Powell. The BOBYQA algorithm for bound constrained optimization without derivatives. Technical Report DAMTP 2009/NA06, Centre for Mathematical Sciences, University of Cambridge, UK, 2009.

[PS04] C.P. Panteliadis and H.M. Strassburg. *Cerebral Palsy: Principles And Management.* Thieme Publishers Series. Thieme Medical Publishers, New York, 2004.

[RB70] J.P. Richter and R.C. Bell. *The Notebooks of Leonardo Da Vinci.* Number 2 in Dover Fine Art Books. Dover Publications, Mineola, NY, 1970.

[RB03] A.U. Raghunathan and L.T. Biegler. Mathematical programs with equilibrium constraints (MPECs) in process engineering. *Computers and Chemical Engineering*, 27(10):1381 – 1392, 2003.

[RB05] A.U. Raghunathan and L.T. Biegler. Interior point methods for Mathematical Programs with Complementarity Constraints (MPCCs). *SIAM Journal on Optimization*, 15(3):720 – 750, 2005.

[Ret12] O. Rettig. *Modellierung der oberen Extremität und Armbewegungen beim Gehen.* PhD thesis, Ruprecht-Karls-Universität Heidelberg, 2012.

[RMS09] M. Raschke, K. Mombaur, and A. Schubert. Jacksonbot - design, simulation and optimal control of an action painting robot. In *ArtsIT 2009, YiLan, Taiwan, Springer Lecture Notes of ICST*, 2009.

[RMS11] M. Raschke, K. Mombaur, and A. Schubert. An optimization-based robot platform for the generation of action paintings. *International Journal of Arts and Technology*, 4(2):181 – 195, 2011.

[Roc97] R.T. Rockafellar. *Convex Analysis (Princeton Mathematical Series).* Princeton University Press, Princeton, 1997.

[RS72] R.D. Russell and L.F. Shampine. A collocation method for boundary value problems. *Numerische Mathematik*, 19(1):1 – 28, 1972.

[RS90] J.A. Reeds and L. A. Shepp. Optimal paths for a car that goes both forwards and backwards. *Pacific Journal of Mathematics*, 145(2):367 – 393, 1990.

[RW04] D. Ralph and S.J. Wright. Some properties of regularization and penalization schemes for MPECs. *Optimization Methods and Software*, 19(5):527 – 556, 2004.

[SAH75] W.J.W. Sharrard, J. M. H. Allen, and S. H. Heaney. Surgical prophylaxis of subluxation and dislocation of the hip in cerebral palsy. *Journal of Bone & Joint Surgery, British Volume*, 57-B(2):160 – 166, 1975.

[SB83] J.P. Schlöder and H.G. Bock. Identification of Rate Constants in Bistable Chemical Reactions. In Deuflhard and Hairer, editors, *Numerical Treatment of Inverse Problems in Differential and Integral Equations, Progress in Scientific Computing*, pages 27 – 47. Birkhäuser, Basel, Boston, Berlin, 1983.

[SB92] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis.* Springer, Heidelberg, 1992.

[Sch88] J.P. Schlöder. *Numerische Methoden zur Behandlung hochdimensionaler Aufgaben der Parameteridentifizierung*, volume 187 of *Bonner Mathematische Schriften.* Universität Bonn, Bonn, 1988.

[Sch91] H. Schwetlick. Nichtlineare Parameterschätzung: Modelle, Schätzkriterien und numerische Algorithmen. *Mitteilungen der Gesellschaft für Angewandte Mathematik und Mechanik*, 2(91):13 – 51, 1991.

[Sch01] S. Scholtes. Convergence properties of a regularization scheme for mathematical programs with complementarity constraints. *SIAM Journal on Optimization*, 11(4):918 – 936, 2001.

[Sch07] G. Schultz. Generation of optimal human running gaits. Master's thesis, Ruprecht-Karls-Universität Heidelberg, 2007.

[Sha05] A.A. Shabana. *Dynamics of Multibody Systems.* Cambridge University Press, Cambridge, 2005.

[Sim98] J. Simon. Modellierung von Kontaktereignissen bei der Simulation von Laufvorgängen. Master's thesis, Ruprecht-Karls-Universität Heidelberg, 1998.

[SM10] G. Schultz and K. Mombaur. Modeling and optimal control of human-like running. *Mechatronics, IEEE/ASME Transactions on*, 15(5):783 – 792, 2010.

[SM13] A. Schubert and K. Mombaur. The role of motion dynamics in abstract painting. In *Proceedings of the Fourth International Conference on Computational Creativity 2013*, 2013.

[SS78] R.W.H. Sargent and G.R. Sullivan. The development of an efficient optimal control package. In J. Stoer, editor, *Proceedings of the 8th IFIP Conference on Optimization Techniques (1977), Part 2.* Springer, Heidelberg, 1978.

[SS00] H. Scheel and S. Scholtes. Mathematical programs with complementarity constraints: Stationarity, optimality and sensitivity. *Mathematics of Operations Research*, 25(1):1 – 22, 2000.

[Ste87] M. Steinbach. Numerische Berechnung optimaler Steuerungen für Industrieroboter. Master's thesis, Ruprecht-Karls-Universität Heidelberg, 1987.

[Ste12] O. Stein. Lifting mathematical programs with complementarity constraints. *Mathematical Programming*, 131(1 – 2):71 – 94, 2012.

[SU10] S. Steffensen and M. Ulbrich. A new relaxation scheme for mathematical programs with equilibrium constraints. *SIAM Journal on Optimization*, 20(5):2504 – 2539, 2010.

[SW89] G. A. F. Seber and C. J. Wild. *Nonlinear Regression.* John Wiley&Sons, Hoboken, NJ, 1989.

[SZ92] H. Schramm and J. Zowe. A version of the bundle idea for minimizing a nonsmooth function: Conceptual idea, convergence analysis, numerical results. *SIAM Journal on Optimization*, 2(1):121 – 152, 1992.

[TM04] P.A. Tipler and G. Mosca. *Physics for Scientists and Engineers.* W.H. Freeman, New York, 2004.

[Tod04] E. Todorov. Optimality principles in sensorimotor control. *Nature Neuroscience*, 7(9):907 – 915, 2004.

[VC94] L. Vicente and P. Calami. Bilevel and multilevel programming: A bibliograpgy review. *Journal on Global Optimization*, 5(3):291 – 306, 1994.

[vdKDH09] M. van der Krogt, C. Doorenbosch, and J. Harlaar. The effect of walking speed on hamstrings length and lengthening velocity in children with spastic cerebral palsy. *Gait & Posture*, 29(4):640 – 644, 2009.

[VIC13] Vicon Motion Systems. http://www.vicon.com, 2013.

[vNM53] J. von Neumann and O. Morgenstern. *Spieltheorie und wirtschaftliches Verhalten.* 1953.

[vS34] H. von Stackelberg. *Marktform und Gleichgewicht.* J. Springer, 1934.

[vS99] R. von Schwerin. *MultiBody System SIMulation: Numerical Methods, Algorithms, and Software.* Lecture Notes in Computational Science and Engineering. Springer, Berlin, Heidelberg, 1999.

[WB06] A. Wächter and L.T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25 – 57, 2006.

[Wie02] P. Wieber. On the stability of walking systems. In *Proceedings of the International Workshop on Humanoid and Human Friendly Robotics*, Tsukuba, Japan, 2002.

[Wik13] Wikipedia. The free encyclopedia. http://en.wikipedia.org/, October 2013.

[Wil08] S. M. Wild. A Derivative-Free Optimization Algorithm Using Minimal Norm Hessians. In *Tenth Copper Mountain Conference on Iterative Methods*, 2008.

[Win99] M. Winckler. *Numerische Werkzeuge zur Simulation, Visulaisierung und Optimierung unstetiger dynamisch Systeme.* PhD thesis, Ruprecht-Karls-Universität Heidelberg, 1999.

[Wol93] S. Wolf. Heidelberg MotionLab, Heidelberg University Hospital, Department Orthopedic Surgery. http://www.heidel-motionlab.de/en/, 1993.

[WP05] R.C. Whaley and A. Petitet. Minimizing development and maintenance costs in supporting persistently optimized BLAS. *Software: Practice and Experience*, 35(2):101–121, February 2005.

[Ye95] J.J. Ye. Necessary conditions for bilevel dynamic optimization problems. *SIAM Journal on Control and Optimization*, 33(4):1208 – 1223, 1995.

[Ye05] J.J. Ye. Necessary and sufficient optimality conditions for mathematical programs with equilibrium constraints. *Journal on Mathematical Analysis and Applications*, 307(1):350 – 369, 2005.