INAUGURAL-DISSERTATION

zur

Erlangung der Doktorwürde

der

Naturwissenschaftlich–Mathematischen Gesamtfakultät

der

RUPRECHT–KARLS–UNIVERSITÄT

HEIDELBERG

vorgelegt von

Dipl.-Math. Andreas Meyer

aus Roth

Tag der mündlichen Prüfung

16. Januar 2020

Numerical Solution of Optimal Control Problems with Explicit and Implicit Switches

Gutachter

PD Dr. Andreas Potschka

Prof. Dr. Ekaterina A. Kostina

Zusammenfassung

Diese Dissertation befasst sich mit der effizienten numerischen Lösung von mitunter gleichzeitig explizit und implizit geschalteten Optimalsteuerungsproblemen. Dazu wird ein Framework entwickelt, in welchem sich beide Problemklassen einheitlich in ein gemischt–ganzzahliges Optimalsteuerungsproblem mit kombinatorischen Nebenbedingungen überführen lassen. Aktuelle Forschungsergebnisse setzen diese Problemklasse in Beziehung zu einem kontinuierlichen Optimalsteuerungsproblem mit verschwindenden Nebenbedingungen, welches wiederum eine bedeutende Unterklasse eines Optimalsteuerungsproblems mit Gleichgewichtsnebenbedingungen darstellt. In der vorliegenden Arbeit bildet dieser Zusammenhang das Fundament für eine numerische Behandlung.

Die verwendeten numerischen Ansätze fußen auf einem direkten Kollokationsansatz und erfordern insbesondere eine möglichst präzise Bestimmung der Schaltstruktur des Ausgangsproblems. Aufgrund der Tatsache, dass die Schaltstruktur im Allgemeinen a priori unbekannt ist, wird diese sukzessive bestimmt. Während dieses Prozesses wird eine Folge von nichtlinearen Programmen, welche von diskretisierten Optimalsteuerungsproblemen abgeleitet werden, approximativ gelöst. Dabei wird nach jeder Iteration das Diskretisierungsgitter gemäß der aktuell geschätzten Schaltstruktur adaptiert.

Neben einer genauen Bestimmung der Schaltstruktur ist es von zentraler Bedeutung den globalen Fehler zu schätzen, der beim näherungsweisen Lösen von Optimalsteuerungsproblemen durch das Kollokationsverfahrens auf den einzelnen Diskretisierungsintervallen ensteht. Dazu werden diskrete Adjungierte benutzt, welche sich mit Hilfe der Lagrange–Multiplikatoren der nichtlinearen Programme extrahieren lassen. Zu diesem Zweck wird mit Hilfe eines funktional analytischen Frameworks die Brücke zwischen Kollokationsverfahren und Petrov–Galerkin Finite–Elemente Verfahren geschlagen. In Analogie zu der im Umfeld von partiellen Differentialgleichungen etablierten Methodik der dual–gewichteten Residuen für Galerkin–Verfahren werden im Anschluss zielorientierte globale Fehlerschätzer abgeleitet. Darauf aufbauend werden Strategien angegeben, die es erlauben die Diskretisierung im Hinblick auf eine möglichst effiziente Reduzierung des globalen Fehlers anzupassen. Dabei ist zu beachten, dass sich die Gitteranpassung bezüglich des globalen Fehlers mit der Adaptierung hinsichtlich der Schaltstruktur vereinbaren lässt und somit auf ein iteratives Lösungsframework führt.

Üblicherweise besitzen einzelne Zustands- und Steuerungskomponenten den gleichen Polynomgrad, wenn sie von einer Kollokations–Diskretisierung stammen. Durch die spezielle Rolle, welche einigen Steuerungskomponenten in dem hier vorgeschlagenen Lösungsframework zukommt, ist es wünschenswert, variierende Polynomgrade zu erlauben. Damit ergeben sich hinsichtlich einer effizienten Implementierung Probleme, welche mittels geschickter Strukturausnutzung sowie einer passenden Permutation von Variablen und Gleichungen behoben werden können. Der resultierende Algorithmus wurde parallel zu dieser Arbeit angefertigt und in einer Software umgesetzt.

Die vorgestellten Methoden werden implementiert und anhand von Benchmark–Problemen

wird ihre Anwendbarkeit und Effektivität demonstriert.
Im Hinblick auf eine zukünftige Einbettung der beschriebenen Verfahren in einen online Optimalsteuerungskontext und die damit verbunden Echtzeitanforderungen wird eine Erweitung der bekannten Multilevel–Iterationsschemata vorgeschlagen. Diese basiert auf der Trapezregel und reduziert den Rechenaufwand im Falle von dünnbesetzten Datenmatrizen gegenüber einer vollständigen Bestimmung erheblich.

Abstract

This dissertation deals with the efficient numerical solution of switched optimal control problems whose dynamics may coincidentally be affected by both explicit and implicit switches. A framework is being developed for this purpose, in which both problem classes are uniformly converted into a mixed–integer optimal control problem with combinatorial constraints. Recent research results relate this problem class to a continuous optimal control problem with vanishing constraints, which in turn represents a considerable subclass of an optimal control problem with equilibrium constraints. In this thesis, this connection forms the foundation for a numerical treatment.

We employ numerical algorithms that are based on a direct collocation approach and require, in particular, a highly accurate determination of the switching structure of the original problem. Due to the fact that the switching structure is a priori unknown in general, our approach aims to identify it successively. During this process, a sequence of nonlinear programs, which are derived by applying discretization schemes to optimal control problems, is solved approximatively. After each iteration, the discretization grid is updated according to the currently estimated switching structure.

Besides a precise determination of the switching structure, it is of central importance to estimate the global error that occurs when optimal control problems are solved numerically. Again, we focus on certain direct collocation discretization schemes and analyze error contributions of individual discretization intervals. For this purpose, we exploit a relationship between discrete adjoints and the Lagrange multipliers associated with those nonlinear programs that arise from the collocation transcription process. This relationship can be derived with the help of a functional analytic framework and by interrelating collocation methods and Petrov–Galerkin finite element methods. In analogy to the dual-weighted residual methodology for Galerkin methods, which is well–known in the partial differential equation community, we then derive goal–oriented global error estimators. Based on those error estimators, we present mesh refinement strategies that allow for an equilibration and an efficient reduction of the global error. In doing so we note that the grid adaption processes with respect to both switching structure detection and global error reduction get along with each other. This allows us to distill an iterative solution framework.

Usually, individual state and control components have the same polynomial degree if they originate from a collocation discretization scheme. Due to the special role which some control components have in the proposed solution framework it is desirable to allow varying polynomial degrees. This results in implementation problems, which can be solved by means of clever structure exploitation techniques and a suitable permutation of variables and equations. The resulting algorithm was developed in parallel to this work and implemented in a software package.

The presented methods are implemented and evaluated on the basis of several benchmark problems. Furthermore, their applicability and efficiency is demonstrated.

With regard to a future embedding of the described methods in an online optimal control context and the associated real-time requirements, an extension of the well–known multi–level iteration schemes is proposed. This approach is based on the trapezoidal rule and, compared to a full evaluation of the involved Jacobians, it significantly reduces the computational costs in case of sparse data matrices.

List of Acronyms

ACQ	Abadie Constraint Qualification
AD	Automatic Differentiation
BDF	Backward Differentiation Formula
BFGS	Broyden–Fletcher–Goldfarb–Shanno
CP	Constraint Programming
CQ	Constraint Qualification
CVP	Constrained Variational Problem
DAE	Differential Algebraic Equation
DP	Disjunctive Programming
DVR	Discrete Variable Representation
DWR	Dual Weighted Residual
EFS	Externally Forced Switch
END	External Numerical Differentiation
FE	Finite Element
FLGR	Flipped Legendre–Gauss–Radau
FRPM	Flipped Radau Pseudospectral Method
GCQ	Guignard Constraint Qualification
GDP	Generalized Disjunctive Programming
GPM	Gauss Pseudospectral Method
HBVP	Hamilton Boundary–Value Problem
IC	Inner Convexification
IFS	Internally Forced Switch
IND	Internal Numerical Differentiation
IVE	Initial Value Embedding

IVP	Initial Value Problem
KKT	Karush–Kuhn–Tucker
LG	Legendre–Gauss
LGL	Legendre–Gauss–Lobatto
LGR	Legendre–Gauss–Radau
LICQ	Linear Independence Constraint Qualification
LMM	Linear Multistep Method
LMPC	Linear Model Predictive Control
LP	Linear Program
LPM	Lobatto Pseudospectral Method
MD–Spline	Multi–Degree Spline
MFCQ	Mangasarian–Fromowitz Constraint Qualification
MILP	Mixed Integer Linear Programming
MINLP	Mixed Integer Nonlinear Programming Problem
MIOCP	Mixed Integer Optimal Control Problem
MIQP	Mixed Integer Quadratic Program
MLD	Mixed Logical Dynamic
MLI	Multi–Level Iteration Scheme
MPC	Model Predictive Control
MPCC	Mathematical Program with Complementarity Constraints
MPEC	Mathematical Program with Equilibrium Constraints
MPVC	Mathematical Program with Vanishing Constraints
MPVC–ACQ	MPVC Abadie Constraint Qualification
MPVC–GCQ	MPVC Guignard Constraint Qualification
MPVC–LICQ	MPVC Linear Independence Constraint Qualification
MPVC–MFCQ	MPVC Mangasarian–Fromowitz Constraint Qualification
NCP	Nonlinear Complementarity Problem

NLP	Nonlinear Programming Problem
NMPC	Nonlinear Model Predictive Control
OC	(Partial) Outer Convexification
OCP	Optimal Control Problem
ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
PMP	Pontryagin's Maximum Principle
PWA	Piecewise Affine
QP	Quadratic Program
RPM	Radau Pseudospectral Method
RTI	Real–Time Iteration Scheme
SLP	Sequential Linear Programming
SNLP	Sequential Nonlinear Programming
SOS	Special Ordered Set
SOS–1	Special Ordered Sets of type 1
SOS–2	Special Ordered Sets of type 2
SQP	Sequential Quadratic Programming
TNLP(x^*)	Tightened Nonlinear Program
TP	Tangential Predictor
VC	Vanishing Constraint
VC–SOS–SUR	Vanishing Constraint SOS–Sum–Up Rounding
VDE	Variational Differential Equation

Contents

Zusammenfassung V

Abstract VII

List of Acronyms IX

Introduction 1

Preface . . . 1
Contributions and Results of this Thesis . . . 9
Thesis Outline . . . 14
Preliminary Notation . . . 17

I State of the Art 19

1 Optimal Control Problems with Switches 21

1.1 Switched Systems and Classification of Switches . . . 22
1.1.1 Initial Value Problems in Ordinary Differential Equations . . . 22
1.1.2 Initial Value Problems in Switched Dynamical Systems . . . 23
1.2 Towards Solutions of Switched Systems . . . 29
1.2.1 Theory of Smooth Initial Value Problems . . . 30
1.2.2 Theory of Initial Value Problems with Switches . . . 31
1.3 Filippov Theory . . . 34
1.4 Sliding Modes . . . 37
1.5 OCPs with Explicit and Implicit Switches . . . 45
1.5.1 Problem Formulation . . . 45
1.5.2 Inner and Outer Convexification in OCPs with Explicit Switches . . . 47
1.5.3 Constraint Formulations . . . 52
1.5.4 Bounds on the Objective Function and Rounding Scheme . . . 56

2 Elements of Real and Functional Analysis 61

2.1 Vector Spaces . . . 61
2.2 Mappings and Dual Spaces . . . 64
2.3 Differentiability in Banach Spaces . . . 65
2.4 Function Spaces . . . 68
2.4.1 Spaces of Continuous Functions . . . 69
2.4.2 Lebesgue Spaces . . . 70

2.4.3 Sobolev Spaces . . . 71
2.4.4 Absolutely Continuous Functions . . . 71
2.4.5 The Function Spaces $\mathcal{Y}^k(\mathcal{I}, \mathbb{R})$ . . . 73
2.4.6 Step Functions . . . 75
2.4.7 Monotone Functions . . . 79
2.4.8 Functions of Bounded Variation . . . 81
2.5 The Lebesgue–Stieltjes Integral . . . 85
2.6 Variational Equalities and Inequalities . . . 92

3 Optimization in Banach Spaces **95**
3.1 Problem Formulation . . . 95
3.2 Existence of a Solution . . . 96
3.3 First–Order Necessary Conditions of Fritz–John Type . . . 96
3.4 Constraint Qualifications . . . 97
3.5 Optimality Conditions for Finite Dimensional Problems . . . 98
3.6 Numerical Methods . . . 103
3.6.1 Lagrange–Newton Method . . . 104
3.6.2 Sequential Quadratic Programming Method . . . 105
3.6.3 Other Approaches . . . 113

4 Mathematical Programs with Vanishing Constraints **115**
4.1 Problem Formulation . . . 115
4.2 Comparison with MPECs . . . 117
4.3 Towards CQs and Necessary Optimality Conditions . . . 118
4.3.1 Tools for MPVC Analysis . . . 118
4.3.2 MPVC–Tailored CQs . . . 119
4.3.3 First–Order Necessary Optimality Conditions . . . 120
4.4 Numerical Approaches . . . 122
4.4.1 Structural Constraint Approach . . . 123
4.4.2 Nonlinear Equation Approach . . . 123

5 Theory of Optimal Control Problems **127**
5.1 Continuous OCPs . . . 127
5.2 OCPs as Infinite Dimensional Optimization Problems . . . 132
5.3 Local Minimum Principle . . . 135
5.4 Solution Structure . . . 137
5.5 Solution Approaches . . . 140
5.5.1 Indirect Methods . . . 140
5.5.2 Direct Methods . . . 141
5.5.3 Direct versus Indirect Methods . . . 141

6 Direct Approach: Problem Discretization **145**
6.1 Derivative Generation . . . 146
6.1.1 Automatic Differentiation . . . 147
6.1.2 IVPs and Sensitivity Generation . . . 148
6.2 Reduced Discretization Approach . . . 155
6.2.1 Control Discretization . . . 156
6.2.2 Direct Single Shooting . . . 157
6.2.3 Direct Multiple Shooting . . . 158
6.3 Full Discretization Approach . . . 164
6.3.1 Local Approach . . . 164
6.3.2 Global Approach . . . 167
6.3.3 Conclusion . . . 185

II Contributions **187**

7 A Local Multi-Degree Pseudospectral Method **189**
7.1 Problem Formulation . . . 190
7.2 Global Collocation . . . 193
7.3 Multi–Degree Global Collocation . . . 199
7.4 Local Collocation . . . 204

8 A Discrete Local Minimum Principle **209**
8.1 Problem Formulation . . . 210
8.2 Representation of Multipliers . . . 216
8.3 Local Minimum Principle . . . 221
8.4 Regularity Conditions . . . 224

9 An Interpretation for Discrete Adjoints of Collocation Methods **227**
9.1 Problem Formulation . . . 228
9.1.1 First Discretize, Then Optimize . . . 229
9.1.2 First Optimize, Then Discretize . . . 230
9.1.3 Auxiliary Results . . . 238
9.2 First Discretize, Then Optimize: Local Collocation Approach . . . 241
9.3 First Optimize, Then Discretize: Petrov–Galerkin Approach . . . 247
9.3.1 Finite Element Spaces . . . 247
9.3.2 Finite Dimensional Optimality Conditions . . . 256
9.4 Synthesis . . . 268

10 A Goal–Oriented Global Error Estimation for Collocation Methods **269**
10.1 Literature Review of Global Error Estimation . . . 270
10.1.1 Error Estimation for Differential Equations . . . 270

10.1.2 Error Estimation for Optimal Control Problems 272
10.2 Goal–Oriented Error Representation 275
10.3 Approximation of the Error Representation 280
10.4 Towards an Adaptive Mesh Refinement 281

11 A Unified Framework for Optimal Control Problems with Switches 285
11.1 Literature Review 285
11.2 Problem Formulation 287
11.3 Optimal Control of Hybrid Systems 287
11.4 Generalized Disjunctive Programming 288
11.5 Mixed Integer Optimal Control Problems 291
11.6 Discretization 293
11.7 MPVC Handling 295
11.8 Sequential Nonlinear Programming 296

III Applications and Numerical Results 297

12 Multi–Degree Pseudospectral Collocation Numerics 299
12.1 An Academic ODE Example 299
12.2 An Academic OCP Example 302

13 Petrov–Galerkin Costate Estimation Numerics 305
13.1 Rayleigh Problem Without Constraints 306
13.2 Rayleigh Problem With Control Bounds 307
13.3 Rayleigh Problem With Mixed Control–State Constraint I 307
13.4 A Minimum Energy Double Integrator 308

14 Goal–Oriented Error Estimation Numerics 317
14.1 Hyper–Sensitive Problem I 317
14.2 Hyper–Sensitive Problem II 319
14.3 Rayleigh Problem With Mixed Control–State Constraint II 321
14.4 Problem with Tangential Path Constraint Exit 322

15 Switched Optimal Control Problem Numerics 329
15.1 A Coulombic Friction Model 329
15.2 A Stick-Slip Model with Known Switching Point 333
15.3 An Alternate Friction Model 336

Conclusion and Outlook 341
Multi–Degree Collocation 341

Costate Estimation . . . 341
Goal–Oriented Error Estimation . . . 342
Switched Optimal Control . . . 342
Nonlinear Model Predictive Control . . . 343

Appendix A Auxiliary Results **347**
A.1 Proof of Lemma 8.5 . . . 347
A.2 Proof of Lemma 9.2 . . . 347
A.3 Towards the Generalized Taylor's Theorem I . . . 349
A.4 Towards the Generalized Taylor's Theorem II . . . 350

Appendix B Numerical Analysis **351**
B.1 Orthogonal Polynomials . . . 351
B.2 Polynomial Interpolation . . . 353
B.3 Numerical Integration . . . 357
B.3.1 Quadrature Using Interpolating Polynomials . . . 358
B.3.2 Gauss Quadrature . . . 358

Appendix C Nonlinear Model Predictive Control **365**
C.1 Feedback Control . . . 365
C.2 The Principle of NMPC . . . 365
C.3 Real–Time Iteration Scheme for NMPC . . . 366
C.3.1 Tangential Predictors . . . 367
C.3.2 Initial Value Embedding . . . 368
C.3.3 RTI for MPC . . . 369
C.4 Multi–Level Iteration Scheme . . . 370
C.4.1 Description of the MLI Levels . . . 370
C.4.2 Convergence Analysis . . . 372

Appendix D Extensions to Multi–Level Iteration Schemes **377**

Danksagung **385**

Bibliography **387**

Nomenclature **426**

List of Figures **430**

List of Tables **431**

Introduction

Preface

Dynamic processes arise naturally or are specifically designed phenomena whose characteristics vary in time. They are commonly described by algebraic equations, ordinary differential equations, partial differential equations, or combinations thereof. Certain characteristics of dynamic processes can be analyzed and often controlled by external impacts.

It seems natural to apply suitable external inputs in a way to achieve certain goals such as the minimization of a selected cost function or maintaining certain physically motivated constraints. Over the years, there has been a rapid growth of scientific disciplines dealing with increasingly complex dynamic processes. Natural sciences, economy, engineering, and even humanities rank among the scientific fields where researchers analyze dynamic processes. This analytical process is commonly subsumed under the general heading "Modeling, Simulation and Optimization".

Certain aspects of dynamic processes make them more complicated to deal with both from a mathematical point of view as well as from a computational point of view, for instance, abrupt changes of the process dynamics triggered by particular in–process states. One may think of a fire damper which is activated when sensors detect smoke particles or extraordinary heat and prevents smoke or fire from spreading to other areas. Process dynamics can also be abruptly affected by external controls that may attain one of just finitely many states which prevents the system from smooth changes. Here, one can imagine a valve of a distillation column that can either be opened or closed by an engineer at every instant.

Depending on whether non–smooth changes of the process dynamics are internally forced or externally forced, we call the respective system implicitly or explicitly switched. One question which arises is how to steer a system that potentially has both explicit and implicit switches such that a prescribed goal is achieved without violating some predefined constraints.

From a mathematical point of view, dynamic processes are modeled by differential equations, where the rate of change of the process state is expressed by a function of the process state itself. Hence, the process state is a dependent variable determined by the differential equation, whereas the time represents an independent variable. Complex differential equations must likely be solved numerically. The continuous time interval is discretized using a finite number of points in time and approximations of states and controls are determined at the resulting temporal grid. Typical algorithms pursue the aim to modify the temporal grid adaptively in a way to balance the inversely affected goals of keeping the approximation error small and limiting the computational effort. In the following, with the help of an illustrative example, we demonstrate that the same error magnitude at different time instants can have different effects on the evolution of the process state.

Imagine first a car driving along a straight road during wintertime. If there were a short section where the car would slip and slide on ice, it would have almost no impact on the arrival time of

the car. If, however, the same would happen when the car is driving along a curvy road, the car would swerve and slow down significantly. As a consequence, this could mean a tremendous delay in arrival. From a mathematical point of view, the effect of small intermediate changes on the final state, such as the impact of short icy road sections on the arrival time, is measured by means of adjoint sensitivities. To obtain adjoint sensitivities, one has to solve a particular adjoint problem. In the first case of our example the adjoint sensitivity is small resulting in minor consequences on the arrival time. In contrast, the adjoint sensitivity is large in the second case resulting in a considerable impact on the arrival time.
The same phenomenon happens when solving differential equations numerically: small local errors caused by a discretization may be propagated differently. Most numerical algorithms deal with estimation and control of local errors exclusively. However, the global error is the crucial quantity that should be investigated.
This doctoral thesis provides a novel step towards a unified treatment of two extremely challenging problem classes, namely explicitly and implicitly switched systems. Furthermore, we contribute an interpretation of adjoint information obtained by a specific discretization scheme. This is achieved by interrelating that adjoint information with costate information coming from a suitable Petrov–Galerkin discretization of a variational formulation of the Hamilton Boundary–Value Problem (HBVP) equations. As a consequence, we are empowered to introduce novel global error estimators based on the Dual Weighted Residual (DWR) methodology. This allows us to control the computational efficiency and accuracy of the numerical solution process. The combination of both aspects, namely the unified framework for switched systems and the error estimation, may even open up new possibilities towards real–time optimal control of switched dynamic processes in the future.
In the remainder of this section, we briefly survey the current state of the art of the issues addressed in this thesis. This allows us to point out the significance and relevance of our research for the field of applied mathematics.

Optimal Control

In this contribution, dynamic processes are described by systems of Ordinary Differential Equations. The optimization of dynamic processes is then referred to as *optimal control*. Numerical algorithms to solve Optimal Control Problems (OCPs) fall into the category of indirect and direct methods, cf. von Stryk and Bulirsch [442], Betts [62].
Indirect methods deal with approximations to the continuous necessary optimality conditions. Well–known algorithms to determine those approximations are multiple shooting (see Osborne [344], Bulirsch [90]) and collocation (see Bock [72], Ascher et al. [17]). A major advantage of indirect methods is their high and guaranteed accuracy. However, they suffer from severe drawbacks such as the necessity to calculate the necessary optimality conditions by hand. This can be an error–prone task, in particular for complex dynamic systems. Furthermore, the region of convergence of the resulting nonlinear systems can be rather small such that accurate initial values are required to initialize the algorithm. This may be cumbersome especially for adjoint variables since their values are in general not physically motivated. Finally, for systems involving path constraints, the switching structure must be known a priori.
Due to the aforementioned disadvantages of indirect methods, direct methods have emerged as

the method of choice for complicated problems. By means of a state or control parametrization the OCP is transcribed into a Nonlinear Programming Problem (NLP). Notable representatives for direct methods are given by direct collocation methods (see Bär [29] and Biegler [68]) and direct multiple shooting methods (see Bock and Plitt [75]). Direct shooting methods parameterize the infinite dimensional control space by a finite number of control parameters and the differential constraints are ensured by explicit numerical integration. In direct collocation methods, both the states and controls are parameterized and piecewise polynomials (see Kraft [279], Hargraves and Paris [223], or von Stryk [440]) or global polynomials (see Vlassenbroeck and Dooren [439]) are used in order to achieve that the differential equation holds at the collocation points. Direct shooting and direct collocation methods transcribe the OCP into a structured and large–scale Nonlinear Programming Problem. After a possible condensing step (see Bock [73] and Cervantes and Biegler [103]), which exploits the special NLP structure and results in a small–scale NLP, numerical algorithms such as active set SQP methods (see Han [221] and Powell [361]) or interior–point methods (see Karmarkar [267], Mehrotra [323], and Wächter and Biegler [444]) can be applied to solve the NLPs.
Since the original problem is transferred into the well–known and well–researched NLP problem class, some disadvantages of indirect methods can be overcome by using direct methods instead: the necessary optimality conditions do not need to be derived, they do not suffer from a small region of convergence, no guess for the adjoint states is required, and the switching structure does not need to be known in advance. However, direct methods are not as accurate as indirect methods and most of them do not provide costate guesses.

Switched Optimal Control

There exist numerous real–world applications whose underlying process dynamics can be described by a system of differential equations involving binary or integer control variables, i.e., controls that can only take values from a finite admissible set. The mathematical discipline dealing with optimization problems constrained by systems of the latter type is commonly known as *switched* or *Mixed–Integer Optimal Control (MIOC)*, and by several authors also referred to as *Mixed–Logic/Mixed–Integer Dynamic Optimization* (see Oldenburg et al. [343]) or *hybrid optimal control* (see Buss et al. [93]).
As one of the first, Bock and Longman [74] deal with MIOCPs: under the assumption of a finite set of admissible accelerations they consider the problem of choosing accelerations of a subway train in an energy–minimizing way. More descriptive examples are given in the following: one can imagine complex chemical plants that allow for design alternatives such as the location of feed trays in distillation columns, or for activating and deactivating specific components such as compressors or heating elements depending on the current state of the plant. Valves, which can be either opened or closed, and buttons, which can be either switched on or off, represent prototypical examples for binary process controls and can be found in various process types. Transmissions, as they are used in many vehicles, allow for selecting one of several discrete gear transmission ratios and can therefore be modeled by means of integer control variables. MIOC applications in the field of chemical engineering and biology can be found in [83, 380, 382, 33] and in [285, 380, 402], respectively. Further applications of practical relevance arise from traffic light optimization (see Göttlich et al. [205]), from

power network topology optimization (see Göttlich et al. [204]), from thermodynamics (see Gräber et al. [209]), or from automotive control (see [187, 383, 273]).
In switched optimal control, we distinguish between *explicit switches*, i.e., explicitly controllable switches, and *implicit switches*, i.e., state dependent switches. For the first class the switchings are degrees of freedom, while for the latter class, the model switching is triggered by the state of the optimization problem. Here, the ground contact of a robot leg or the weir overflow in a distillation column may serve as vivid examples.
In order to illustrate the increased complexity of the MIOC problem class compared to the one of continuous OCPs, we have a short look at both problems after a discretization step. As we have pointed out, OCPs are transcribed into NLPs. In contrast, the finite dimensional counterpart for Mixed Integer Optimal Control Problems (MIOCPs) is given by Mixed Integer Nonlinear Programming Problems (MINLPs), cf. Belotti et al. [43]. This problem class has been proven to be $\mathcal{NP}$–hard, cf. Kannan and Monma [264]. Consequently, MINLPs are at least as hard as the hardest problems in $\mathcal{NP}$ and under the assumption $\mathcal{P} \neq \mathcal{NP}$ deterministic machines are unable to solve certain MINLP instances in polynomial time.
The key challenge in solving MIOCPs is to deal with their combinatorial nature, which, as a consequence, leads to a vast number of possible operation modes. A common way to bypass this difficulty in some chemical engineering applications is to assume the process to be in a phase equilibrium and to solve a static optimization problem (see Duran and Grossmann [144]) or a system with time–dependent dynamic subproblems (see Schweiger and Floudas [400]). Indirect methods to MIOCPs are based on hybrid maximum principles (see [18, 403]) or, by means of a time transformation, on a reformulation into an equivalent continuous OCP (see [142, 143]). By an application of local minimum principles to the continuous OCP one can then derive necessary optimality conditions.
There exists a variety of solution algorithms for switched OCPs in the literature. While we give a rather detailed overview on existing methods in Chapter 11, we just point to a considerable subset for now: approaches that deal with implicitly switched systems can use direct simultaneous methods based on direct multiple shooting and incorporating a switch detecting integrator (see [83, 271]), multi–stage OCP formulations (see [328]), and variational formulations in terms of switching time instants (see [457]). Some algorithms considering systems with explicit switches base on problem specific continuous reformulations of discrete valued controls (see [91]), rounding heuristics (see [424]), switching time optimization techniques (see [188]), branch and bound as well as sparse direct collocation (see [443]), or dynamic programming (see [89]).
In recent years considerable progress has been achieved towards an efficient solution of MIOCPs employing a convexification and a subsequent relaxation step, cf. [380, 381]. This approach yields a lower bound for the MIOCP objective, which is obtained by the solution of a convexified and relaxed control problem. The lower bound can be approximated arbitrary close by MIOCP feasible points. The authors propose an algorithm to find suchlike points for a given tolerance. One can find solutions for many MIOCPs instances with just sub–exponential running time. Kirches [272] extended the approach to systems with path constraints that directly depend on integer control functions. Discretized OCPs of the latter type lead to a special form of NLPs, namely Mathematical Programs with Vanishing Constraints (MPVCs). Among others, the lack of constraint qualification and ill–conditioning make MPVCs a rather chal-

lenging problem class, cf. [3, 238]. MPVCs represent an important subclass of Mathematical Programs with Equilibrium Constraints (MPECs). Problems of the latter type stand out due to their high degree of lack of convexity and smoothness.
All solution approaches to OCPs with explicit or implicit switches, which have been presented so far and will be presented in later chapters, are limited to a particular problem class and do not provide the possibility to solve explicitly and implicitly switched systems in a unified way.

Collocation Methods

A particular class of direct methods discretizing states and controls simultaneously is given by *pseudospectral* (see Elnagar et al. [145] and Canuto et al. [99]) or *orthogonal* (see Cuthrell and Biegler [121]) collocation methods. In pseudospectral methods, states and controls are approximated by a finite basis of global interpolating polynomials and the differential equation is ensured to hold at so–called collocation points by employing respective algebraic constraints. The set of collocation points can be chosen arbitrarily but usually orthogonal collocation is applied. This means that the collocation points are given as roots of linear combinations of certain orthogonal polynomials and their derivatives. Since pseudospectral methods are usually implemented as orthogonal collocation methods – depending on the mathematical community – the terms pseudospectral and orthogonal collocation are used interchangeably.
Pseudospectral collocation methods applied to smooth OCPs typically converge faster than other methods. This is due to the so–called spectral accuracy property (see Trefethen [428]) which guarantees an exponential convergence rate, provided the solution is smooth. Regarding non–smooth problems or problems comprising different model stages the time horizon is split into different finite elements and global orthogonal collocation is applied to each of them. There exist numerous examples in the literature where orthogonal collocation was applied to non–smooth OCPs, cf. Cuthrell and Biegler [120] or Ross and Fahroo [376].
Pseudospectral collocation methods are based on spectral methods. These have been successfully used for the numerical solution of Partial Differential Equations since the 1970's and were derived from the method of weighted residuals (see Finlayson [164]). Early ideas of using spectral representations date back to Fourier [170]. The spectral collocation method was first used by Slater [413] and Kantorovich [265]. Lanczos [283] stressed the necessity of a proper choice for trial functions and the location of collocation points. Gottlieb and Orszag [206] later established a unifying theory for spectral methods. The term pseudospectral method for spectral collocation methods goes back to Orszag. The application of spectral methods to OCPs involving Chebyshev polynomials was first done by Vlassenbroeck and Dooren [439]. A pseudospectral method using Legendre polynomials was developed by Elnagar et al. [145]. Since then, several variations of the Chebyshev pseudospectral method and the Legendre pseudospectral method have been developed, where the sets of collocation points are electively chosen from Gauss–type, Gauss–Radau–type, or Gauss–Lobatto–type nodes. Exemplarily, we refer the reader to [439, 438, 152, 153, 261, 262] and cited references therein.
There exist several implementations employing collocation methods. A list of considerable software packages, which is far from complete, is given by SOCS [64], DIDO [377], DIRCOL [441], GPOPS-II [351].

The discretization granularity of collocation methods directly impacts the size of the resulting NLP. The finer the discretization grid and the higher the polynomial degrees the larger the NLP gets. There are reasonable arguments to use distinct polynomial degrees for single components of the state and control approximations. A control modeling a gear shift, which can only have states from a finite set, can naturally be represented by piecewise constant functions, while a control modeling a gas pedal would be represented by higher degree polynomials. Moreover, an error control procedure could also suggest different polynomial degrees for distinct state and control components. To our knowledge, there are no implementations available which allow for individually adjustable polynomial degrees within a finite element.

Pontryagin's Maximum Principle

Necessary optimality conditions for Optimal Control Problems are known as *maximum principles* or *minimum principles*. As mentioned earlier, they form the core of indirect methods. Starting from the 1950s with early results by Pontryagin [357] and Hestenes [231] they have been intensively studied since then. Necessary optimality conditions involving pure state constraints are stated, for example, in [251, 256, 314, 315]. References studying mixed control–state path constrained OCPs are [140, 466, 308, 338]. Contributions dealing with sufficient conditions are [316, 309].
An excellent survey paper on the maximum principle for OCPs is provided by Hartl et al. [224]. They summarize some issues in context of the maximum principle as follows: "[...] since there exist various forms of the necessary and sufficient optimality conditions. Because the literature on this subject is not comprehensive and is, at times, incorrect or incomplete, it has been hard to understand, especially for people working in applied areas." Later they formulate an informal theorem of the maximum principle "[...] that is used often as a recipe while dealing with optimal control problems with state constraints in an applied setting".
Beigel [41] develops a new functional analytic framework which provides a well–posed variational formulation of IVPs in a Banach space setting which is tailored to common discretization schemes for ODEs. The framework exploits the duality pairing between continuous functions and normalized functions of bounded variation and enables Beigel to define weak adjoint solutions of adjoint IVPs. To our knowledge, nobody has investigated Beigel's framework in an OCP context yet. In particular, it would be appealing to analyze the consequences for OCP and path constraint adjoints in the respective Banach space setting. One could also establish Pontryagin's Maximum Principle (PMP) for functions spaces of practical relevance.

Costate Estimation

Whenever one solves an OCP numerically the question arises under which circumstances – assuming a continuously refined grid – the resulting sequence of approximate solutions converges towards a local minimum of the original continuous–time OCP, cf. [137, 200, 262]. The answer to this question is by no means trivial: Hager [215] could show that a discretization scheme which converges when applied to a certain dynamics fails to converge when applied to an OCP constrained to this dynamics. On the contrary, Betts et al. [67] deal with discretization schemes whose approximate solutions converge towards a local solution when applied to

an OCP but do not converge when used as an integrator on the constrained dynamics.
If one wants to verify the optimality of approximate solutions, the adjoint states play an important role. Moreover, the adjoints can be very helpful for other reasons such as the development of grid refinement strategies (see [62]), or an OCP sensitivity analysis, i.e., investigating the behavior of OCP solutions with respect to disturbance parameters, cf. [310, 92]. Finally, sensitivity differentials of OCPs with respect to disturbance parameters, which can be used for real–time optimal control approximations of perturbed solutions (see [353, 320]), depend on knowing the adjoints as well.
As we have pointed out, adjoint states are not required to constitute consistent discretization schemes of direct methods. Yet several approaches have been developed to retain approximations. In [312, 230], the authors use state and control approximations obtained by the direct approach to solve an adjoint problem providing an adjoint approximation. An approach based on a sensitivity analysis is presented in Seywald and Kumar [401]. Here, a relation between the adjoint states and certain cost function sensitivities is exploited. This is complemented by the fact that the same relation also holds for Lagrange multipliers from a direct approach and the sensitivities of the corresponding discretized cost function. The idea of establishing relationships – based on algebraic mappings – between the continuous adjoints and the Karush–Kuhn–Tucker (KKT) multipliers coming from a direct approach was described, e.g., in Hager [214] for one–step and multistep integration schemes and in von Stryk [440] for collocation schemes. There exist several papers applying the latter approach to pseudospectral methods (see e.g. [154, 375] for Lobatto methods, [262, 179, 172] for Radau methods, and [52, 53, 245] for Gauss methods). Their basic idea can be sketched as follows: the HBVP equations coming from PMP are discretized with the same discretization scheme that is used for the direct approach. Comparing variables from the resulting discretized system with the KKT system variables from the direct method provides a suitable algebraic mapping.
Beigel [41] proposes a different approach for BDF integration schemes and their adjoint IND schemes. Beigel develops a new functional analytic framework which provides a well–posed variational formulation of IVPs in function spaces that are tailored to common integration schemes. Using the framework leads naturally to a definition of weak adjoint solutions of adjoint IVPs. By means of a Petrov–Galerkin discretization approach the new variational formulation with weak adjoints is transcribed into a finite dimensional equation system. Beigel verifies that this system of equations is equivalent to the BDF method together with their discrete adjoint IND schemes, cf. [41, 42].

Error Estimation and Mesh Refinement in Direct Transcription Methods

Error estimates for shooting type transcription methods typically assume the control to be optimal and estimate the error of the integration scheme only, cf. Betts [62]. By varying the step size or other adaptive components, common ODE integrators control just the local error, cf. Shampine [406]. However, good approximations are quantified by the global error and involve the stability of the nominal problem, which can be described by adjoint sensitivity information. The calculation of adjoint sensitivities requires, e.g., the solution of the adjoint IVP, cf. [150, 330, 101, 284, 427]. Early approaches to determine the stability of the nominal problem are based on determining a single global stability constant (see [148, 150]). Later ad-

vancements make use of distributed stability factors (see [37, 38]) given by the adjoint solution. The latter approach, commonly known as the DWR method, has been extended to control the error with respect to a given functional. The respective method is known as the DWR method for goal–oriented error estimation, cf. [39].
Mesh refinement strategies for pseudospectral methods either increase the degree of respective state and control approximating polynomials (p methods) or the number of finite elements (h methods). In general, p methods try to increase the number of collocation points close to domain sections undergoing abrupt changes of the trajectories, cf. e.g. Gong et al. [201]. Criteria to refine the finite element grid in h methods are often based on evaluating certain residua at inner element points, cf. e.g. Darby et al. [122, 124]. Latest results (cf. Liu et al. [299]) imply that it is not only necessary to combine h and p refinement strategies but also to allow for a coarsening of the grid.

Model Predictive Control and Mixed–Integer Nonlinear Model Predictive Control

So far we have considered off–line OCPs where the optimal control is determined before the actual process operation begins. However, for most real–world processes there exists no perfect mathematical model and the process most likely undergoes disturbances such that off-line solutions are only of limited applicability. This justifies the use of *real–time optimization* approaches of which Model Predictive Control (MPC) is an important representative.
The fundamentals of MPC can be summarized as follows: at a certain time instant t_0 one solves an OCP on a prediction horizon $[t_0, t_0 + T]$. The obtained optimal control is fed back to the real process for a short time δ. Hereafter a new OCP is solved on an updated horizon $[t_0+\delta, t_0+\delta+T]$ and the scheme is repeated. The optimization problems are initialized with repeatedly updated data from the real process allowing to react on disturbances.
Linear MPC has intensively been studied and employed to numerous industrial process. Literature on linear MPC can be found, for example, in [178, 286]. There exist many processes whose system behavior is not captured adequately by linear models. In a natural way, this leads to Nonlinear Model Predictive Control (NMPC) where nonlinear models are investigated. Surveys on the theoretical foundations of NMPC can be found in [369, 321, 11]. NMPC applications are investigated in [362, 363, 12].
A crucial aspect for the practicability of NMPC algorithms is the numerical solution of the arising OCPs in real–time. Considerable effort has been put into this task. We refer the reader to [465, 464] for major algorithmic achievements in interior–point based methods and to [131, 132, 451] for SQP based methods.
While MPC is a well–researched field in the linear case and major steps towards maturity are done in the nonlinear case, mixed–integer NMPC is still in its very early stage. In this contribution, we subsume real–time optimization subject to process controls with a finite number of admissible values under the term mixed–integer NMPC. Early approaches to mixed–integer NMPC are based on solving mixed–integer QPs, cf. Allgöwer and Zheng [11]. Literature applying this strategy can be found in [48, 342]. Kirches [272] developed a new approach which can be seen as a combination of Bock's direct multiple shooting method (see [75]), the real–time iteration scheme (see [131, 132]), and a partial outer convexification with a subsequent relaxation step (see [380]) in order to deal with the integer–valued controls.

Contributions and Results of this Thesis

The aim of this thesis is to develop a unified framework for the efficient solution of Optimal Control Problems with explicit and implicit switches. To this end, proceeding from a tailored collocation discretization scheme, we develop a sequential nonlinear programming approach which incorporates an adaptive grid refinement strategy according to the a priori unknown switching structure and controls the homotopy to solve the underlying MPVC instance. Furthermore, we contribute a global goal–oriented a posteriori error estimation approach for OCPs that is based on the DWR methodology and can be used within our sequential approach but also independently of it. We justify the proposed algorithms by means of the underlying theory on the one hand and by sound numerical results on the other hand. Since this thesis covers several areas and describes novel methods as well as advances over previously established results, we outline our contributions in the following.

Literature Survey

In this thesis, we deal with several complex facets of optimal control theory, including switched OCPs, local minimum principles, covector mapping principles, and global error estimation. Those different fields require particular fundamentals. The first part of this thesis is therefore devoted to two tasks, namely to save the reader from looking up the theoretical principles in numerous papers and reference books, and to enable the reader to classify our contributions in literature. Our literature survey involves the fundamentals of switched optimal control theory such as a classification of switching types, Filippov theory, well–known available mathematical representations, and possible convexification approaches. Apart from standard function spaces in optimal control theory, we also introduce some less common ones such as step functions and functions of bounded variation. We provide results on finite and infinite optimization problems: while we rely on finite dimensional KKT conditions in order to derive our covector mapping result, we exploit necessary optimality conditions from infinite optimization theory to establish local minimum principles in a specific Banach space setting. We present elementary results on MPVC theory and algorithms as MPVCs naturally arise in our novel numerical approach to switched optimal control. We also give an introduction to continuous optimal control theory. We motivate this decision by the fact that we deal with both common solution approaches, namely direct and indirect methods. In particular, we describe the direct multiple shooting method since we use it to contribute an extension to the Multi–Level Iteration Scheme (MLI). Moreover, we explain specific pseudospectral collocation as well as Petrov–Galerkin discretization schemes as we use them for discretizing OCPs and necessary optimality conditions in our covector mapping result.

A Multi–Degree Collocation Approach

As we have pointed out before, collocation methods and particularly pseudospectral methods are established in the literature with state and control approximating polynomials all of them having the same polynomial degree on a single finite element. Since the polynomial degrees are directly correlated with the dimension of the resulting collocation NLP instance it is desirable to keep them small. However, different components may require disparate polynomial

degrees. For instance, our novel switched optimal control solution algorithm incorporates dedicated control variables indicating the binary state of a switch and consequently the respective polynomial should be constant on a finite element. We propose a pseudospectral method that enables us to to choose distinct numbers of collocation points for single state and control components. Here, the difficulties do arise from carrying out an efficient numerical realization since a naive implementation of the single–degree version would cause too many right–hand side evaluations and destroy the structure of the NLP Jacobian and Hessian. We develop strategies based on tailored data structures and permutations of NLP variables and constraints to overcome the indicated issues and incorporate them into our multi–degree pseudospectral method software.

A Discrete Local Minimum Principle

Beigel [41] investigates the relation between the discrete adjoints of variable–order variable–step size BDF methods and the costates defined by the adjoint differential equation. To this end, she constructs a Constrained Variational Problem (CVP) in a Banach space setting which is tailored to most common discretization schemes: for a prescribed fixed discretization grid differential states have continuous and piecewise continuously differentiable approximations. Exploiting the duality pairing between the space of continuous functions and the space of normalized functions of bounded variation, Beigel identifies the adjoint of a stationary point of the CVP to be an element of the latter function space as well as the integral of the solution of the adjoint equation. We transfer Beigel's setting from the CVP to the more difficult OCP case: we assume a fixed discretization grid and consider states to be of continuous and piecewise continuously differentiable type and controls to be of piecewise continuous type. Within this function space setting, we derive necessary optimality conditions in terms of local minimum principles of OCPs subject to both boundary constraints and mixed control–state path constraints. Therefore, we adopt a proof technique which was also used by Gerdts [189] in his habilitation thesis within a different function space setting and in a DAE OCP context. First, the OCP is rewritten as an infinite dimensional optimization problem. Then, the prerequisites of Fritz John type necessary conditions for optimality are checked. They are applied and result in a system of variational equalities and inequalities involving multipliers in terms of functionals that are elements of unhandy function spaces. Subsequently, explicit representations of state and path constraint adjoints are derived. In comparison with Beigel, who employs Hahn–Banach's extension theorem and finds adjoints to be functions of bounded variation, we show that the adjoints have even higher regularity.

Covector Mapping for a Collocation Method

The discrete local minimum principle of the previous section provides us with a variational formulation of first–order necessary optimality conditions for a rather general OCP with boundary and path constraints. The solution space of local pseudospectral methods is covered by the function space setting of the local minimum principle. Thus, the question arises if the OCP discretization with a specific local pseudospectral method ("discretize–then–optimize") commutes with a suitable discretization of the necessary optimality conditions ("optimize–then–

discretize"). To this end, we apply Petrov–Galerkin finite element techniques to discretize the infinite dimensional optimality conditions: in terms of appropriately chosen finite dimensional basis functions we replace trial and test functions with finite dimensional functions and derive a system of equations and inequalities from the resulting finite dimensional variational system. We prove the equivalence of the latter system with the KKT system coming from the pseudospectral discretization approach. For this reason, we show the commutation of the "discretize–then–optimize" and the "optimize–then–discretize" approach in our Banach space functional analytic framework. In particular, we obtain the Petrov–Galerkin finite element formulation of the local pseudospectral optimality system including ODE, adjoint, and stationarity discretization schemes.

Goal–Oriented Global Error Estimation

In this thesis, we contribute novel goal–oriented global error estimators for numerical solutions of boundary and path constrained optimal control problems obtained by local pseudospectral methods. As adjoint information describes the correct propagation of the nominal local error onto the global error, it is highly recommended to incorporate discrete adjoint information given in terms of corresponding Lagrange multipliers of the pseudospectral optimality system. However, in order to be able to expect reliable convergence results, the Lagrange multipliers need to approximate ODE and path constraint multipliers of the local minimum principle conditions. Indeed, with the novel Petrov–Galerkin finite element representation of the pseudospectral KKT system, in particular nominal and adjoint conditions, we provide an appropriate framework. Using the dual weighted residual methodology of finite element methods, we derive goal–oriented global error representations incorporating aforementioned Lagrange multipliers. Notably, our novel approach overcomes the need for solving an extra dual problem. Local error quantities turn out to be given in the form of defect integrals of the nominal approximation. For the sake of feasibility, in implementations, we establish approximations of those integrals and describe procedures to use the error estimator for adaptive mesh refinement.

Framework for the Solution of OCPs with Explicit and Implicit Switches

As mentioned above, a majority of algorithms to solve implicitly switched OCPs suffers from the drawback that the switching sequence must be known in advance. Hence, in case there exists no reasonable knowledge about the switching sequence, it must be determined in an upstream step. As a consequence, the approach is of low practicability when applied to closed–loop systems. Moreover, one seldom finds methods that can handle both explicitly and implicitly switched systems. In this contribution, we provide a novel approach that deals with the numerical solution of OCPs subject to ODEs with both implicit and explicit switches as well as a priori unknown switching sequence. In particular, we are able to cover systems undergoing consistent as well as the more difficult inconsistent switches, where the latter type is interpreted in the sense of Filippov. To this end, we exploit techniques from generalized disjunctive programming and transcribe the implicitly switched problem part into a counterpart problem where discontinuities do not appear implicitly anymore. Instead, one obtains a prob-

lem containing discrete control variables and vanishing constraints. The explicitly switched problem part in association with the reformulated implicitly switched problem part fits into the well–established mixed–integer optimal control theory which was developed in Bock's group (see Sager [380], Kirches [272]) and complements it.

Sequential Nonlinear Programming Framework

Recent results in mixed–integer optimal control theory (see Lenders [292]) enable us to drop integrality constraints on integer variables within the previously described unified framework for explicitly and implicitly switched systems. Consequently, we end up with a continuous OCP with the particularity that, according to our construction, we expect certain control variables to attain only discrete values indicating the active switching modes at certain time intervals. Taking this into account, we use a "first discretize, then optimize" approach where we apply our local multi–degree collocation discretization with element wise constant switch mode indication control approximations. The discretization leads to a MPVC. Due to the a priori unknown switching sequence on the one hand and for the purpose of dealing with the MPVC on the other hand, we propose a novel Sequential Nonlinear Programming (SNLP) framework. More specifically, starting from an initial discretization, we approximately solve a sequence of relaxed finite dimensional optimization problems where the respective sequence of solutions is supposed to converge towards a solution of the original problem. We develop algorithms which iteratively adapt the discretization grid according to the switching structure. We furthermore propose strategies to follow the relaxation homotopy path in such a way that infeasible problems are avoided but, at the same time, the relaxation parameter is continuously driven to zero. Our SNLP approach can naturally be extended to include further grid adaption methods, for example methods, based on a posteriori error estimation such as the one developed in this thesis.

Fast Nonlinear Model Predictive Control

In the future, we plan to embed our switched optimal control algorithm into a NMPC context. There exist first ideas for mixed–integer NMPC algorithms based on the real–time iteration scheme, which in turn fits into the more general Multi–Level Iteration Schemes (see [451]). In this work, we contribute a new level to MLI. The new level is in line with the ideas of parareal (see [26, 177]): in direct shooting methods, the Jacobian may be determined by solving a particular matrix–valued differential equation, the so–called Variational Differential Equations (VDEs). It may be favorable to solve the VDEs on a coarser grid than the corresponding shooting equations. To this end, the trapezoidal rule is applied on each shooting interval. Even tough using an implicit scheme, we are capable to avoid solving a nonlinear equation system by means of suitable algebraic reformulations. As a result, the Jacobian approximation is given as the product of a matrix inverse and a matrix. A constraint space transformation of the MLI inexact Newton–type equation system makes the matrix inverse calculation dispensable and our method efficient. Local convergence of the proposed MLI level relies on contractivity conditions for inexact Newton–type methods and can be proven similarly to MLI optimality (level–C) iterations.

Implementations

We implemented the new MLI level in the software package `MLI` [451]. All other developed algorithms are implemented in our software package `grc` which allows for the generic and fast solution of continuous as well as explicitly and implicitly switched optimal control problems. It realizes our Sequential Nonlinear Programming framework including the local multi–degree pseudospectral discretization scheme and the option to choose between automatic and user–driven modes for adaptive mesh refinement and MPVC homotopy strategies. In total, the kernel of `grc` has roughly 25,000 lines of `Matlab` code and 6,000 lines of C code for the mex interface. The software provides interfaces to the software packages `SolvIND` [9], `Ipopt` [444], and SNOPT [197]. The last two allow the user to solve occurring NLP instances with state of the art interior–point and SQP methods. `SolvIND` was developed in Bock's research group and contributes a powerful suite of ODE/DAE solvers with IND. Moreover, it enables the user to set up OCP instances in a very comfortable way and saves him from providing function derivatives since this is achieved by an automatic differentiation interface to the software ADOL-C [445].

Case Studies

We demonstrate the main theoretical contributions of this thesis at the examples of appropriately chosen continuous and switched optimal control problem instances. In particular, we present examples substantiating the reliability of our local multi–degree pseudospectral method. Furthermore, we show results on adjoint approximations obtained from NLP Lagrange multipliers of our collocation approach. Using OCP examples with different constraint types (boundary constraints, mixed control–state constraints, and pure state constraints), we demonstrate the efficacy of our proposed goal–oriented global error estimation approach.

Publications

During the work on this thesis, we contributed the following publications:

Haßkerl et al. [226]	D. Haßkerl, A. Meyer, N. Azadfallah, S. Engell, A. Potschka, L. Wirsching, and H. G. Bock. Study of the performance of the multi-level iteration scheme for dynamic online optimization for a fed-batch reactor example. In *2016 European Control Conference (ECC)*. IEEE, 2016. doi: 10.1109/ecc.2016.7810327
Lindscheid et al. [298]	C. Lindscheid, D. Haßkerl, A. Meyer, A. Potschka, H. G. Bock, and S. Engell. Parallelization of modes of the multi-level iteration scheme for nonlinear model-predictive control of an industrial process. In *2016 IEEE Conference on Control Applications (CCA)*. IEEE, 2016. doi: 10.1109/cca.2016.7588014
Bock et al. [78]	H. G. Bock, C. Kirches, A. Meyer, and A. Potschka. Numerical solution of optimal control problems with explicit and implicit switches. *Optimization Methods and Software*, 33(3):450–474, 2018. doi: 10.1080/10556788.2018.1449843
Kirches et al. [276]	C. Kirches, E. A. Kostina, A. Meyer, and M. Schlöder. Numerical Solution of Optimal Control Problems with Switches, Switching Costs and Jumps. *Optimization Online Preprint 6888*, 2018
Kirches et al. [277]	C. Kirches, E. A. Kostina, A. Meyer, and M. Schlöder. Generation of Optimal Walking-Like Motions Using Dynamic Models with Switches, Switch Costs, and State Jumps. *Optimization Online Preprint 7124*, 2019. (submitted to International Conference on Decision and Control 2019)

Thesis Outline

This thesis is organized in three parts: current state of the art, our contributions, and finally applications and numerical results. The thesis is laid out in fourteen chapters and three appendices as follows.

Chapter 1 is the first chapter of Part I and introduces the problem class of switched Optimal Control Problems (OCPs), classifies the different switching types and analyzes their properties. In particular, we provide the theoretical foundations of switched systems including existence and uniqueness theory, Filippov's theory to handle inconsistent switches, and a classification of the different sliding modes from a geometrical point of view. Finally, we survey several convexification approaches to switched OCPs leading to the mixed–integer optimal control theory, which was developed in Bock's research group and incorporates partial outer convexification as well as vanishing and complementarity constraint reformulations. In Chapter 2 we summarize the fundamentals of real and functional analysis that are used throughout the thesis. We highlight important results about Banach space theory and several function

spaces such as Lebesgue and Sobolev spaces as well as step functions, monotone functions, and functions of bounded variation. The chapter also sketches the Lebesgue–Stieltjes integral and some variational equalities and inequalities. Chapter 3 reviews the elements of finite and infinite optimization theory within a Banach space context. Optimality conditions are provided in terms of Fritz–John type conditions for infinite dimensional optimization problems. In form of KKT conditions we carry over the necessary conditions to the finite dimensional case and describe the Lagrange–Newton as well as the SQP method as common numerical solution algorithms and how they are related. Chapter 4 provides the definition of Mathematical Programs with Equilibrium Constraints and their important subclass of Mathematical Programs with Vanishing Constraints. We establish well–known tailored stationarity concepts and present common numerical solution approaches. In Chapter 5 we introduce Optimal Control Problems (OCPs) subject to dynamic processes encoded by Ordinary Differential Equations (ODEs) in a standard Banach space setting, embed the problem class into the infinite dimensional optimization theory of Chapter 3 which naturally allows us to derive first–order necessary conditions in terms of local minimum principles. The chapter also gives some insight into the solution structure of OCPs and particularly how it is related to switching functions. We present the general solution concepts of OCPs, namely indirect and direct methods, and sketch advantages and drawbacks of each. Chapter 6 concludes Part I and gives a deeper insight into the broad range of direct approaches to OCP. After reviewing some fundamentals about computer–aided derivative generation, numerical calculation of IVP sensitivities including the important IND principles, and common control discretization approaches we distill well–known direct shooting type methods. Last but not least we deal with full discretization approaches, where polynomial parametrizations are used for both states and controls. Based on the weighted residual principles, which are of great importance since they include Galerkin and collocation type methods, we highlight the need for a carefully considered choice of collocation points and the advantages of expanding the approximate solution in a series involving orthogonal polynomials.

Part II, which contains the author's contributions to different fields in applied mathematics, starts with Chapter 7 and the presentation of a particular global pseudospectral collocation method based on Gauss–Radau collocation points. In the further course of the chapter we extend the aforementioned discretization to the multi–degree case which is implemented in our new software package `grc` and goes beyond the functionality of currently available implementations since polynomial order values can be assigned to single components of state and control trajectory approximations without the need for adding additional constraints and an overhead of artificial variables in the problem discretization. However, for an efficient implementation our new approach requires some structure exploitation techniques based on variable and constraint permutations and the use of tailored data structures. This is explained briefly, followed by an embedding of the global method into a local element–wise context as subsequent chapters rely on local discretization formulations. Chapter 8 formulates OCPs within a Banach space setting, motivated by semi–discrete function spaces as they arise from numerical solvers, and considers them as infinite dimensional optimization problems of type

$$F(\boldsymbol{x},\boldsymbol{u}) \to \min \quad \text{s.t.} \quad G(\boldsymbol{x},\boldsymbol{u}) \in \mathcal{K},\ H(\boldsymbol{x},\boldsymbol{u}) = \Theta,$$

with a cone set $\mathcal{K}$ and mappings F, G and H between appropriate Banach spaces. Based on first–order necessary conditions for optimization problems of the latter type we then derive necessary conditions in terms of local minimum principles for OCPs subject to ODEs, boundary constraints, and mixed control–state path constraints. During the proof we exploit the special structure of the considered OCP and this enables us to find an explicit representation for the involved multipliers. Contrary to an alternative approach, which makes use of a function space duality pairing argument and an application of the Hahn–Banach Extension Theorem, the multipliers are more regular. By means of optimality conditions from Chapter 8 we set up a finite dimensional system of equalities and inequalities in Chapter 9 and proof their equivalence with the KKT system of a particular local pseudospectral method. We find the aforementioned system by specifying tailor–made finite element spaces and subsequently applying a corresponding Petrov–Galerkin finite element discretization to a variational formulation of the infinite dimensional optimality conditions. The previous results allow us to show that the "discretize–then–optimize" approach and the "optimize–then–discretize" approach commute in our functional analytic setting. Moreover, we are empowered to distill costate approximations from Lagrange multiplier values coming from the pseudospectral NLP. Chapter 10 is devoted to the derivation of novel goal–oriented global error estimators for local pseudospectral methods. To this end we develop a novel error representation for the specific Petrov–Galerkin finite element discretization of Chapter 9. By incorporating the DWR methodology and by approximating the unknown exact dual weights with suitable values we are able to determine global error approximations in a criterion of interest. Dual weights approximations involve discrete costate information provided by pseudospectral NLP multipliers. To our knowledge we are the first to involve those pseudospectral multipliers in DWR a posteriori error estimators for the goal–oriented global error. By means of the DWR methodology we find error representations summating element–wise nominal local error quantities of ODEs and constraints multiplied by respective adjoint values describing the sensitivity of the quantity of interest on intermediate disturbances. The local error quantities comprise defect integral values of ODEs and constraint nominal approximations. Approximating those defect integrals enables us to derive evaluable global error estimators of practical relevance. The final Chapter 11 of Part II first specifies a broad class of OCPs subject to explicitly and implicitly switched dynamic systems. Hereafter we transcribe implicitly switched system parts into explicit counterparts subject to additional constraints and binary control variables. Here we make use of concepts from generalized disjunctive programming. We establish a connection between the transcribed problem and MIOCPs such that we are enabled to relax integrality constraints. Due to recent results the relaxed problem yields an objective value that can be reached by binary controls up to any prescribed accuracy threshold. We discretize the relaxed problem with the help of a suitable discretization approach taking account of the particular meaning of certain controls. We show that the proposed discretization approach fits into our local multi–degree collocation framework and that the resulting finite dimensional problem is of MPVC type. This motivates us to propose a SNLP approach in which the homotopy parameter of the relaxed MPVC is driven to zero. Moreover, the discretization grid is adaptively refined according to the gradually revealed switching structure. We propose strategies for detecting explicit switches and switches of Filippov type. We briefly outline how our global error estimators may be incorporated into our SNLP framework.

Part III demonstrates the main theoretical contributions of this thesis at the example of several benchmark problems including OCPs of continuous as well as switched type. The numerical results cover the multi–degree pseudospectral method in Chapter 12, the interpretation of adjoint NLP variables in Chapter 13, the global goal–oriented error estimation in Chapter 14, and finally the unified framework for explicitly and implicitly switched OCPs in Chapter 15. Selected examples demonstrate the functionality of our software package `grc`.

The final chapter briefly summarizes the results of this thesis and provides several ideas for future research. In particular, this includes the extension of our new framework for switched OCPs such that it can be used within a closed–loop environment. In this context we also introduce a novel level for the Multi–Level Iteration Schemes that is based on the trapezoidal rule.
This thesis is closed by three appendix chapters. Appendix A states proofs of several auxiliary results for the reader's convenience. In Appendix B, we provide elements of numerical analysis including orthogonal polynomials, polynomial interpolation, and numerical integration. Appendix C deals with the fundamentals of closed–loop systems and outlines the principles of NMPC as a state–of–the–art feedback control approach. In particular, we demonstrate the ideas of tangential predictors and initial value embedding which motivate the Real–Time Iteration Scheme (RTI) as an efficient approach to real–time optimization. We generalize the RTI approach which then leads to the Multi–Level Iteration Scheme.

Computational Environment

All computational results and run times presented in this thesis have been obtained on a 64–bit *Ubuntu© Linux™ 16.04* system powered by an *Intel© Core™ i7-3820* CPU at 3.60 GHz, with 15.6 GB main memory available. A single core of the available eight physical cores of the CPU has been used. All source code is written in *MATLAB 9.2* (release name: R2017a) and *ANSI C99* which is compiled using version 5.4.0 of the *GNU C/C++ compiler collection*, with applicable machine–specific optimization flags enabled.

Preliminary Notation

From Chapter 2 on, we give a profound overview of thesis relevant concepts about finite and infinite dimensional optimization as well as their fundamentals. This includes the associated notation as well. However, the first chapter is supposed to stress the importance of the topic covered in this thesis. We present some preliminary notation at this point of the dissertation in the hope to facilitate reading the first chapter.
The symbol $\mathbb{N}$ denotes the set of natural numbers excluding zero. The set of all integer numbers is denoted with $\mathbb{Z}$. For the set of real numbers we use the symbol $\mathbb{R}$.
Closed, open, and half–open intervals are usually denoted with $\mathcal{I}$ where start and endpoint points are given by the letters a and b. Horizon intervals in the context of differential equations or optimal control problems are usually denoted with $\mathcal{T}$. They are compact intervals $[t_\mathrm{s}, t_\mathrm{f}]$ with starting point t_s and end point t_f. For switching time instants we use the symbol t_σ.

We use capital letters $\mathcal{X}$, $\mathcal{Y}$ for sets. For integer valued sets we use the Greek letter Ω. The cardinality of Ω is denoted with $|\Omega|$. Functions are denoted with bold letters such as $\boldsymbol{F}$, $\boldsymbol{G}$, $\boldsymbol{f}$, $\boldsymbol{g}$, $\boldsymbol{x}$, $\boldsymbol{u}$. The i–th component of a vector–valued function $\boldsymbol{f} : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ is written as $\boldsymbol{f}_i$, i.e., we have $\boldsymbol{f} = [\boldsymbol{f}_1, \ldots, \boldsymbol{f}_m]^T$. In the same way x_i denotes the i–th component of a real vector $x \in \mathbb{R}^n$. For gradient and subdifferential (generalized derivative) of a function $\boldsymbol{f}$ at point x we write $\nabla \boldsymbol{f}(x)$ and $\partial \boldsymbol{f}(x)$, respectively. The partial derivatives of a function $\boldsymbol{f} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_y} \longrightarrow \mathbb{R}^{n_f}$, $(x, y) \mapsto \boldsymbol{f}(x, y)$ at (x, y) are denoted with $\boldsymbol{f}'_x(x, y)$ and $\boldsymbol{f}'_y(x, y)$.
We use the symbol **id** for the identity function. The sign function $\operatorname{sgn}(\cdot)$ is defined as

$$\operatorname{sgn}(x) \stackrel{\text{def}}{=} \begin{cases} -1, & \text{if } x < 0, \\ 0, & \text{if } x = 0, \\ +1, & \text{if } x > 0. \end{cases}$$

For a set $\mathcal{X}$ we denote its convex hull with $\operatorname{conv}(\mathcal{X})$. We use the notation $\mathcal{U}_r(x)$ for the open ball of radius $r > 0$ centered at a point x. For $N \in \mathbb{N}$ we denote the set $\{1, \ldots, N\}$ with $[N]$.
This thesis deals with time dependent processes described by Ordinary Differential Equations. The independent variable of a trajectory $\boldsymbol{x} : \mathcal{T} \longrightarrow \mathbb{R}^{n_x}$ with $\mathcal{T} \subset \mathbb{R}$ is denoted by t and it can be associated with time. The derivative of $\boldsymbol{x}$ with respect to time t is denoted by $\dot{\boldsymbol{x}}(t) \stackrel{\text{def}}{=} \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t)$.

Part I

State of the Art

Chapter 1

Optimal Control Problems with Switches

A broad class of real–world problems deals with systems which operate by switching between different *subsystems* or *modes*. Such systems are called *hybrid systems*. For instance, a valve or the opening and closing of a power switch give rise to hybrid behavior. A Thermostat adjusting the heat naturally leads to hybrid behavior, too. Further examples of hybrid systems can be found e.g. in growing and dividing biological cells, space shuttles which enter, cross or leave atmosphere layers, or in vehicles changing their dynamics abruptly due to locking and unlocking wheels on ice. The literature does not provide a unified and precise definition of the hybrid system problem class. Among others, this results from its high level of generality and the fact that it is an interdisciplinary problem class which is influenced by concepts from engineering, computer science and mathematics. For a comprehensive introduction to hybrid systems we refer the interested reader to the excellent survey paper of Goebel et al. [198] as well as to the textbooks of Lunze and Lamnabhi-Lagarrigue [304], van der Schaft and Schumacher [432], and Liberzon [297].

In this contribution, we consider a particular but rather general class of hybrid systems, namely *discontinuous or switched systems*. Starting in the late 1940s switched systems have been studied intensively in the former Soviet Union and other Eastern European countries. A considerable amount of theory has been developed in the field of switched systems. Meanwhile, there is a broad range of textbooks available considering the results of this research, for instance, the ones by Andronov et al. [14], Filippov [163], or Utkin [430].

The investigation of switched systems was to a substantial part motivated by research in control theory, where systems are manipulated by varying certain input variables. In this thesis, we deal with *switched system optimal control problems* in which input variables of a switched system are chosen properly such that some measure of system performance is optimized. Regarding the theory of this topic, there is plenty of literature available, cf. Branicky et al. [84], Brandt-Pollmann [83], Bengea and DeCarlo [49], Sager [380], Kirches [271], or Kamgarpour and Tomlin [263]. However, most researchers deal only with certain aspects or types of switched systems. The challenge and strength of our approach is it to present a unified framework for the numerical solution of optimal control problems with switches.

In general, different types of switched systems are distinguished: proceeding from the classic *Initial Value Problems with Ordinary Differential Equations*, we start our investigations in Section 1.1 by presenting the most relevant switching types, including *state jumps* as well as *explicit* and *implicit switches*. The literature usually provides solution strategies for systems with a single switching type. The strength of the approach proposed later in this work (see Chapter 11) is that it enables us to readily solve systems which contain different switching types in a unified way. To implement this idea, it is necessary to synthesize the relevant switching types in a unified framework, as this is realized in Section 1.1.

For mathematical problems, especially problems motivated by physical phenomena, it is a crucial factor to determine if they are *well–posed* or *ill–conditioned* in the sense of Hadamard. For this reason Section 1.2 is dedicated to deal with the existence and uniqueness of all relevant problem types that are introduced in Section 1.1. Thereby, it is required to introduce tailored solution concepts for certain problems in order to achieve their well–posedness.

One of these solution concepts is based on a *differential inclusion* setting and leads to Filippov's first–order theory. Due to its importance and complexity, there is a detailed investigation of the solution concept according to Filippov in Section 1.3. Here, one seeks for a solution of an Initial Value Problem in Ordinary Differential Equations in which the right–hand–side varies discontinuously as the solution trajectory reaches one or more surfaces, often called *switching surfaces* or *zero manifolds*, but is smooth otherwise.

In general, there are several possible outcomes as the solution reaches a zero manifold. Loosely speaking, the solution can either cross the zero manifold or may stay on it. In the latter case, a description of the motion on the surface is required. Section 1.4 describes all possibly occurring cases according to Filippov's theory in detail.

Finally, in Section 1.5, we introduce a broad class of switched Optimal Control Problems, i.e., infinite dimensional optimization problems that minimize a quantity of interest subject to switched dynamical systems and possibly additional constraints. Our new solution approach, which is presented in Chapter 11, is based on two central steps. First, the switched system is embedded into a larger family of systems and the optimal control problem is formulated for this larger family. In a second step, the embedded problem is relaxed and the relationship between solutions of the switched optimal control problem and the (relaxed) embedded optimal control problem are investigated. This idea is not new as it has been successfully applied to a simpler class of switched optimal control problems. All relevant previous results are reviewed in Section 1.5 as well.

1.1 Switched Systems and Classification of Switches

1.1.1 Initial Value Problems in Ordinary Differential Equations

Commonly, researchers in applied mathematics deal with systems arising from real–world problems. Systems that can be modeled by nonlinear differential equations are called *nonlinear dynamical systems*. One particular class of differential equations describing a dynamical system is given by systems of *ODEs*. Here, we speak of an ODE if one seeks a differentiable function $\boldsymbol{x} = \boldsymbol{x}(t)$ of one real variable t, whose derivative $\dot{\boldsymbol{x}}$ has to satisfy an equation of the form

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t)). \tag{1.1}$$

Since there occur only first order derivatives of the unknown function $\boldsymbol{x}$, the ODE is said to be of first–order. In general, there exist infinitely many different functions $\boldsymbol{x}$ which satisfy (1.1). In order to obtain a single solution from the set of all solutions, one has to pose additional requirements such as *initial conditions*. The resulting problem is then called Initial Value Problem in ODEs.

Definition 1.1 (Initial Value Problem in ODEs)
Let $\mathcal{T} \stackrel{\text{def}}{=} [t_s, t_f] \subset \mathbb{R}$ be a compact interval with $t_s < t_f$. An *Initial Value Problem (IVP)* in *ODEs* is given as a system of $n \in \mathbb{N}$ first–order ODEs and n initial conditions

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t)), \quad t \in \mathcal{T}, \tag{1.2a}$$

$$\boldsymbol{x}(t_s) = x_s, \tag{1.2b}$$

where the right–hand–side $\boldsymbol{f} : \mathcal{T} \times \mathcal{X} \longrightarrow \mathbb{R}^n$ and the unknown dynamic state $\boldsymbol{x} : \mathcal{T} \longrightarrow \mathcal{X}$ are vector–valued functions. The set $\mathcal{X} \subseteq \mathbb{R}^n$ denotes the range of the states. We call $t \in \mathcal{T}$ the *independent variable* and $x_s \in \mathbb{R}^n$ the *initial state vector* or *initial value*. The component–wise derivative of the dynamic state with respect to time t is denoted by $\dot{\boldsymbol{x}}$. △

The solution to the IVP from Definition 1.1, provided that it exists, constitutes one or more solution curves in the state space through the initial point x_s. Such a solution curve is called a *trajectory* in the state space.
Here and in the remainder of this chapter, $\mathcal{T}$ is used with the same meaning as in Definition 1.1, i.e., it denotes the *horizon* interval of an ODE. Repeated definitions of $\mathcal{T}$ are avoided and therefore often omitted. A special case of ODE systems, namely linear ODE systems, is defined separately.

Definition 1.2 (Linear ODE System)
Let $\boldsymbol{A} : \mathcal{T} \longrightarrow \mathbb{R}^{n \times n}$ be a matrix–valued function and $\boldsymbol{b} : \mathcal{T} \longrightarrow \mathbb{R}^n$ a vector–valued function. Then, an ODE system of the form

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}(t)\boldsymbol{x}(t) + \boldsymbol{b}(t)$$

is called a *system of linear ODEs*. △

1.1.2 Initial Value Problems in Switched Dynamical Systems

A discontinuity in the right–hand–side function $\boldsymbol{f}(\cdot)$ or in the state vector $\boldsymbol{x}(\cdot)$ of an ODE is called *switch*. State discontinuities are called *jumps* in this work. We introduce switched dynamical systems involving jumps for the sake of completeness on the one hand and since our unified framework might be augmented for this problem type in the future on the other hand. For right–hand–side discontinuities, we distinguish different types of switches, namely *explicit* and *implicit* switches.
To explore switched systems not only from an ODE perspective but additionally from a more descriptive point of view, may be helpful for subsequent investigations. Switched dynamical systems consist of a set of dynamical subsystems representing the so–called *modes* of the system and a switching law which determines the active subsystem at each time instant. A switch from one subsystem to another is triggered by a so–called *event signal*. Either external signals or internal signals cause such an event signal. Here, an internal signal means that the state satisfies an internal condition. Depending on whether signals are generated externally or internally, the respective switches are either called *Externally Forced Switches (EFSs)* or *Internally Forced Switches (IFSs)*. EFSs are also known as *controllable* or *explicit switches*, whereas IFSs are often called *implicit switches*.
Switched systems inherently involve discrete decision variables. For this reason, we are continuously confronted with integer or binary valued functions and variables in the following.

Definition 1.3 (Integer/Binary Control Functions and Variables)
Let $\boldsymbol{v} : \mathcal{T} \longrightarrow \mathbb{R}^{n_v}$ be a function and $v \in \mathbb{R}^{n_v}$ a vector. We call the component functions $\boldsymbol{v}_i(\cdot)$ *integer control functions* and v_i *integer variables*, if their values are restricted to the set $\mathbb{Z}$. If their values are even restricted to the set $\{0, 1\}$, then $\boldsymbol{v}_i(\cdot)$ and v_i are called *binary control functions* and *binary variables*, respectively. △

Now, we introduce common formulations for ODEs with explicit and implicit switches separately. One can easily see the similarities with Definition 1.1. Afterwards, we give a definition of switched dynamical systems in a unified form.

Initial Value Problems in Ordinary Differential Equations with Explicit Switches

The following definition provides a first representation of an explicitly switched ODE. Other common formulations are traceable to this one.

Definition 1.4 (IVP in ODEs with Explicit Switches)
Let $\mathcal{T} \stackrel{\text{def}}{=} [t_s, t_f] \subset \mathbb{R}$ be a compact interval with $t_s < t_f$. An *IVP* in *ODEs with explicitly defined switches* is given as a system of $n \in \mathbb{N}$ first–order ODEs and n initial conditions as

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{v}(t)), \quad t \in \mathcal{T}, \tag{1.3a}$$
$$\boldsymbol{x}(t_s) = x_s, \tag{1.3b}$$

where the right–hand–side $\boldsymbol{f} : \mathcal{T} \times \mathcal{X} \times \Omega \to \mathbb{R}^n$ and the unknown dynamic state $\boldsymbol{x} : \mathcal{T} \to \mathcal{X}$ are vector–valued functions. The system is affected by another vector–valued function $\boldsymbol{v} : \mathcal{T} \to \Omega$, which attains only values from a finite discrete set $\Omega \stackrel{\text{def}}{=} \{v^1, v^2, \ldots, v^{n_\omega}\} \subseteq \mathbb{R}^{n_v}$ with cardinality $|\Omega| = n_\omega < \infty$. The initial condition is defined analogously to Definition 1.1. △

Compared to the problem from Definition 1.1, the addition of integrality caused by the discrete valued function $\boldsymbol{v}(\cdot)$ usually means that the system can run in different operation modes. Another common approach of formulating ODE (1.3a) results from the following considerations: let functions $\boldsymbol{f}^i : \mathcal{T} \times \mathbb{R}^n \to \mathbb{R}^n$ be defined as

$$\boldsymbol{f}^i(t, \boldsymbol{x}(t)) \stackrel{\text{def}}{=} \boldsymbol{f}(t, \boldsymbol{x}(t), v^i), \quad i \in [n_\omega].$$

Then ODE (1.3a) can be expressed as

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}^i(t, \boldsymbol{x}(t)), \quad t \in \mathcal{M}^i \stackrel{\text{def}}{=} \{\tau \in \mathcal{T} : \boldsymbol{v}(\tau) = v^i\}, \quad i \in [n_\omega]. \tag{1.4}$$

One can say that an explicitly switched system of the form (1.4) is in mode i if $t \in \mathcal{M}^i$. In a similar fashion, we can write (1.4) by means of an indexed set of differential equations as

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}^{\boldsymbol{i}(t)}(t, \boldsymbol{x}(t)), \quad t \in \mathcal{T}, \tag{1.5}$$

where the integer control function $\boldsymbol{i} : \mathcal{T} \longrightarrow [n_\omega]$ indicates the index of the current mode. Hence, using the sets $\mathcal{M}^i$ defined in (1.4) the function $\boldsymbol{i}(\cdot)$ is given as

$$\boldsymbol{i}(t) = i, \quad t \in \mathcal{M}^i, \quad i \in [n_\omega].$$

Another way to express explicitly switched ODEs makes use of boolean–valued control functions $\boldsymbol{\omega}^i : \mathcal{T} \longrightarrow \{0,1\}$. The $\boldsymbol{\omega}^i(\cdot)$ are defined such that they are equal to 1 whenever $\boldsymbol{\nu}(t) = \nu^i$ and zero otherwise. The switched ODE (1.3a) then can be reformulated as

$$\dot{\boldsymbol{x}}(t) = \sum_{i=1}^{n_\omega} \boldsymbol{\omega}^i(t) \cdot \boldsymbol{f}(t, \boldsymbol{x}(t), \nu^i), \quad t \in \mathcal{T}, \tag{1.6}$$

$$\boldsymbol{\omega}^i(t) = \begin{cases} 1 & t \in \mathcal{M}^i \\ 0 & t \notin \mathcal{M}^i \end{cases}, \qquad i \in [n_\omega].$$

Initial Value Problems in Ordinary Differential Equations with Implicit Switches

The activation time $t_\sigma \in \mathcal{T}$ of an implicit switch $i \in [n_\sigma]$ is implicitly given by a zero–crossing of the corresponding component of a *switching function* $\boldsymbol{\sigma} : \mathcal{X} \longrightarrow \mathbb{R}^{n_\sigma}$. More concrete, for a given state trajectory $\boldsymbol{x} = \boldsymbol{x}(t)$ the sign structure of the time–dependent and vector–valued function

$$t \mapsto \boldsymbol{\sigma}(\boldsymbol{x}(t)) \tag{1.7}$$

uniquely identifies the *activation state* of any switch at any time t of the horizon interval $\mathcal{T}$. Thus, at any time instant $t \in \mathcal{T}$ the i–th switch can take exactly one of three activation states given by $\boldsymbol{\sigma}_i(t) < 0$, $\boldsymbol{\sigma}_i(t) > 0$, and $\boldsymbol{\sigma}_i(t) = 0$. We assume $\boldsymbol{\sigma}(\cdot)$ to be sufficiently smooth in its state component. The implicit function theorem therefore enables us to consider the *switching point* t_σ as a function of (t_s, x_s). By means of $\boldsymbol{\sigma}(\cdot)$ we set up ODEs with implicitly defined switches.

Definition 1.5 (IVP in ODEs with Implicit Switches)
Let $\mathcal{T} \stackrel{\text{def}}{=} [t_s, t_f] \subset \mathbb{R}$ be a compact interval with $t_s < t_f$. An *IVP in ODEs with implicitly defined switches* is given as a system of $n \in \mathbb{N}$ first–order ODEs and n initial conditions as

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \operatorname{sgn}(\boldsymbol{\sigma}(\boldsymbol{x}(t)))), \quad t \in \mathcal{T}, \tag{1.8a}$$

$$\boldsymbol{x}(t_s) = x_s, \tag{1.8b}$$

where the right–hand–side $\boldsymbol{f} : \mathcal{T} \times \mathcal{X} \times \{0, \pm 1\}^{n_\sigma} \to \mathbb{R}^n$ and the unknown dynamic state $\boldsymbol{x} : \mathcal{T} \to \mathcal{X}$ are vector–valued functions. The system is affected by the sign structure of the switching function $\boldsymbol{\sigma}(\cdot)$. The initial condition is defined analogously to Definition 1.1. △

Similarly to the case of explicitly switched systems, the third component of function $\boldsymbol{f}(\cdot)$ in Definition 1.5 introduces different operation modes by its integrality. The only difference is that in implicitly switched systems the current mode of the system is implicitly determined by the state trajectory, whereas it depends just on the definition of $\boldsymbol{\nu}(\cdot)$ in explicitly switched systems. In order to emphasize the close relationship of explicitly and implicitly switched systems, we establish alternative representations of implicitly switched systems in the following.
Alternatively to the form used in Definition 1.5, one could represent implicitly switched dynamic systems by an indexed set of differential equations as

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}^{\boldsymbol{i}(t)}(t, \boldsymbol{x}(t)), \quad t \in \mathcal{T}. \tag{1.9}$$

The current mode of the system is identified by the sign structure of $\boldsymbol{\sigma}(\cdot)$. Hence, we distinguish $|\{0,\pm1\}|^{n_\sigma} = 3^{n_\sigma}$ modes and define the index set of switching modes as $\Omega \stackrel{\text{def}}{=} \{1,\dots,3^{n_\sigma}\}$. At any time instant on the horizon interval $\mathcal{T}$, the function $\boldsymbol{i} : \mathcal{T} \longrightarrow \Omega$ in (1.9) indicates the index of the applicable dynamic right–hand–side. In other words, at any instant the function $\boldsymbol{i}(\cdot)$ specifies the active subsystem.

Assuming that only a finite number of switching events occurs, the function $\boldsymbol{i}(\cdot)$ may be identified with a finite set of tuples $S \stackrel{\text{def}}{=} \{(\tau_0, i_0), (\tau_1, i_1), \dots, (\tau_N, i_N)\}$ with $0 \leqslant N < \infty$. The $\tau_j \in \mathcal{T}$ where $t_\text{s} = \tau_0 \leqslant \tau_1 \leqslant \ldots \leqslant \tau_N \leqslant t_\text{f}$, denote all switching points of the system and the $i_j \in \Omega$ denote the respective subsystem indices for all $j = 0,\dots,N$. Hence, the switching structure is uniquely identified by the switching sequence $\sigma \stackrel{\text{def}}{=} \{i_j\}_{j=0}^N$ and the associated switching instants $\tau \stackrel{\text{def}}{=} \{\tau_j\}_{j=0}^N$. One can also write Problem (1.9) as

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}^i(t, \boldsymbol{x}(t)), \quad t \in \mathcal{M}^i \stackrel{\text{def}}{=} \{\tau \in \mathcal{T} : \boldsymbol{i}(\tau) = i\}, \quad i \in \Omega.$$

In order to point out that ODEs (1.8a) and (1.9) can be converted into each other, we define a bijective function $\boldsymbol{p} : \Omega \longrightarrow \{0,\pm1\}^{n_\sigma}$ which maps the index set $\Omega = \{1,\dots,3^{n_\sigma}\}$ of switching modes to the sign structure induced by the switching function $\boldsymbol{\sigma}(\cdot)$, i.e., we have $\boldsymbol{p}(\boldsymbol{i}(t)) = \operatorname{sgn}(\boldsymbol{\sigma}(\boldsymbol{x}(t)))$ for all t in $\mathcal{T}$. ODE (1.9) in terms of $\boldsymbol{f}(\cdot)$ therefore reads as

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}^{\boldsymbol{i}(t)}(t, \boldsymbol{x}(t)) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{p}(\boldsymbol{i}(t))).$$

The following example illustrates a possible realization of functions $\boldsymbol{p}(\cdot)$, $\boldsymbol{i}(\cdot)$ and $\boldsymbol{f}^{\boldsymbol{i}(t)}(\cdot)$ for IVP (1.8a)–(1.8b) if we investigate the easiest case $n_\sigma = 1$.

Example 1.6
Let us assume $n_\sigma = 1$ for IVP (1.8a)–(1.8b). Then we have 3 modes with $\Omega = \{1,2,3\}$ and define the bijective function $\boldsymbol{p} : \Omega \longrightarrow \{0,\pm1\}$ as

$$\boldsymbol{p}(1) \stackrel{\text{def}}{=} -1, \quad \boldsymbol{p}(2) \stackrel{\text{def}}{=} 0, \quad \boldsymbol{p}(3) \stackrel{\text{def}}{=} +1.$$

According to the definition of $\boldsymbol{p}$ the functions $\boldsymbol{f}^i : \mathcal{T} \times \mathbb{R}^n \longrightarrow \mathbb{R}^n$ are given as

$$\boldsymbol{f}^1(t,x) \stackrel{\text{def}}{=} \boldsymbol{f}(t,x,-1), \quad \boldsymbol{f}^2(t,x) \stackrel{\text{def}}{=} \boldsymbol{f}(t,x,0), \quad \boldsymbol{f}^3(t,x) \stackrel{\text{def}}{=} \boldsymbol{f}(t,x,+1),$$

and the index function $\boldsymbol{i} : \mathcal{T} \longrightarrow \Omega$ implicitly as

$$\boldsymbol{i}(t) \stackrel{\text{def}}{=} \begin{cases} 1, & \text{if } \boldsymbol{\sigma}(\boldsymbol{x}(t)) < 0, \\ 2, & \text{if } \boldsymbol{\sigma}(\boldsymbol{x}(t)) = 0, \\ 3, & \text{if } \boldsymbol{\sigma}(\boldsymbol{x}(t)) > 0. \end{cases}$$

The IVP then reads as

$$\dot{\boldsymbol{x}}(t) \stackrel{\text{def}}{=} \begin{cases} \boldsymbol{f}^1(t, \boldsymbol{x}(t)), & \text{if } \boldsymbol{\sigma}(\boldsymbol{x}(t)) < 0, \\ \boldsymbol{f}^2(t, \boldsymbol{x}(t)), & \text{if } \boldsymbol{\sigma}(\boldsymbol{x}(t)) = 0, \\ \boldsymbol{f}^3(t, \boldsymbol{x}(t)), & \text{if } \boldsymbol{\sigma}(\boldsymbol{x}(t)) > 0, \end{cases} \quad t \in \mathcal{T}, \quad \boldsymbol{x}(t_\text{s}) = x_\text{s}.$$

Unified Framework for IVPs in ODEs with Explicit and Implicit Switches

Next, we aim to combine the switched system types of Definition 1.4 and Definition 1.5 resulting in a system that involves both explicit as well as implicit switches. The new system type will be embedded in an Optimal Control Problem context in the subsequent Section 1.5.
For the reader's convenience, we augment the explicitly and implicitly switched system by so–called *state jumps*, i.e., differential state components may undergo discontinuities. In order to define the respective IVP, it is necessary to introduce some notations. For the sake of simplicity we assume for a moment that only a single switch is present in the problem. The left–hand–side limit of the state vector $\boldsymbol{x}$ at switch time t_σ is denoted by x^-, whereas the right–hand–side limit is denoted by x^+, i.e.,

$$x^-(t_\sigma; t_s, x_s) \stackrel{\text{def}}{=} \lim_{\varepsilon \nearrow 0} \boldsymbol{x}(t_\sigma + \varepsilon; t_s, x_s), \tag{1.10}$$

$$x^+(t_\sigma; x^-, \boldsymbol{\Delta}) \stackrel{\text{def}}{=} \lim_{\varepsilon \searrow 0} \boldsymbol{x}(t_\sigma + \varepsilon; t_\sigma, x^-, \boldsymbol{\Delta}). \tag{1.11}$$

First, it can be stated that the one sided continuity of $\boldsymbol{x}(\cdot)$ at t_σ must hold. Analogously to the case of implicit switches, the parameters (t_s, x_s) of the state vector $\boldsymbol{x}(\cdot)$ in (1.10) express that the switching point t_σ may be considered as a function of (t_s, x_s) by virtue of the implicit function theorem. The right–hand–side limit in (1.11) depends implicitly on x^- and additionally on the *jump function* of the differential states which has the form

$$\boldsymbol{\Delta} : \mathcal{T} \times \mathcal{X} \longrightarrow \mathcal{X}, \quad (t, \boldsymbol{x}^-(t)) \mapsto \boldsymbol{x}^+(t). \tag{1.12}$$

The jump function becomes effective when the switch is activated. Analogously to our notation of state discontinuities, we proceed with right–hand–side discontinuities. The right–hand–side in (t_σ, x^-) and (t_σ, x^+) is denoted with f^- and f^+, respectively, i.e.,

$$f^-(t_\sigma, x^-) \stackrel{\text{def}}{=} \lim_{\varepsilon \nearrow 0} \boldsymbol{f}(t_\sigma + \varepsilon, x^-), \tag{1.13}$$

$$f^+(t_\sigma, x^+) \stackrel{\text{def}}{=} \lim_{\varepsilon \searrow 0} \boldsymbol{f}(t_\sigma + \varepsilon, x^+). \tag{1.14}$$

Finally, we are able to combine the switched system types from Definition 1.4 and Definition 1.5 and additionally augment them with implicitly defined state jumps.

Definition 1.7 (IVP in ODEs with Explicit and Implicit Switches and State Jumps)
Let $\mathcal{T} \stackrel{\text{def}}{=} [t_s, t_f] \subset \mathbb{R}$ be a compact interval with $t_s < t_f$. An *IVP in ODEs with explicitly and implicitly defined switches and state jumps* is given as a system of $n \in \mathbb{N}$ first–order ODEs, n initial conditions and state jump conditions as

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{v}(t), \operatorname{sgn}(\boldsymbol{\sigma}(\boldsymbol{x}(t)))), \quad t \in \mathcal{T}, \tag{1.15a}$$

$$\boldsymbol{x}(t_s) = x_s, \tag{1.15b}$$

$$x^+(t_\sigma) = \boldsymbol{\Delta}_\sigma(t_\sigma, x^-(t_\sigma)), \quad \forall\, t_\sigma \in \mathcal{T} : \exists\, i \in \{1, 2, \ldots, n_\sigma\} : \boldsymbol{\sigma}_i(t_\sigma) = 0, \tag{1.15c}$$

where the right–hand–side $\boldsymbol{f} : \mathcal{T} \times \mathcal{X} \times \Omega \times \{0, \pm 1\}^{n_\sigma} \longrightarrow \mathbb{R}^n$ and the unknown dynamic state $\boldsymbol{x} : \mathcal{T} \longrightarrow \mathcal{X}$ are vector–valued functions. The system is affected by the sign structure of the switching function $\boldsymbol{\sigma}$

and another vector–valued function $\boldsymbol{\nu} : \mathcal{T} \to \Omega$, which attains only values from a finite discrete set $\Omega \overset{\text{def}}{=} \{\nu^1, \nu^2, \ldots, \nu^{n_\omega}\} \subseteq \mathbb{R}^{n_\nu}$ with cardinality $|\Omega| = n_\omega < \infty$. The initial condition is defined analogously to Definition 1.1. The notations of $\boldsymbol{\sigma}$, x^-, x^+ and $\boldsymbol{\Delta}_\sigma$ follow the ones in (1.7), (1.10), (1.11) and (1.12) △

In order to avoid the risk of ill–posed problems we assume at most a finite number of switching times t_σ in Problem 1.15. In the case of switched systems without state jumps we will discuss switches where the zero manifold $\boldsymbol{\sigma}(x) = \mathbf{0}$ is crossed infinitely often in finite time, where the limits f^- or f^+ do not exist, or where state trajectories are sliding along the zero manifold $\boldsymbol{\sigma}(x) = \mathbf{0}$. We will do this in Sections 1.2 and 1.3 and show how we deal with it.
It is quite appealing to solve the problem from Definition 1.7. Most existing algorithms can only deal with a single switching type. In contrast, the methods developed in this contribution deal with discontinuities in the right–hand–side in a unified framework, i.e., systems with explicit and implicit switches but not with state jumps. For this reason, we assume all jump conditions in (1.15c) to be equal to zero from now on, i.e., $\boldsymbol{\Delta}_\sigma \equiv \mathbf{id}$. However, the author believes that the new framework can be extended with state jumps, but this task is left to future research.
Now, we give an alternative definition (see e.g. Xu and Antsaklis [459] or Lunze and Lamnabhi-Lagarrigue [304]) for switched systems which enables us to point out some issues that arise in the following Section 1.2 when it comes to finding solution concepts for the problem from Definition 1.7. A switched system can be regarded as a 3–tuple $\mathcal{S} = (\mathcal{D}, \mathcal{F}, \mathcal{L})$, where

- $\mathcal{D} = (I, E)$ is a direct graph. This graph represents the discrete mode structure of the system. $I = \{1, 2, \ldots, M\}$ is the set of indices for all M subsystems. E is a subset of $I \times I \setminus \{(i, i) : i \in I\}$ which contains the admissible events between different subsystems, i.e., the system switches from $i_1 \in I$ to $i_2 \in I$ if the event $e = (i_1, i_2)$ takes place. The event sets E_E and E_I denote the admissible events for EFSs and IFSs such that $E = E_E \cup E_I$.
- $\mathcal{F} = \{\boldsymbol{f}^i : \mathcal{T} \times \mathcal{X}_i \longrightarrow \mathbb{R}^n : i \in I\}$, and $\boldsymbol{f}^i$ is the vector field for the i–th subsystem. The $\mathcal{X}_i \subseteq \mathbb{R}^n$ denote the state constraint sets for the i–th subsystem.
- $\mathcal{L} = \mathcal{L}_E \cup \mathcal{L}_I$ provides logic constraints relating the continuous state and mode switchings. $\mathcal{L}_E$ is defined as $\mathcal{L}_E \overset{\text{def}}{=} \{\Lambda_e : \Lambda_e \subseteq \mathbb{R}^n,\ \emptyset \neq \Lambda_e = \mathcal{X}_{i_1} \cap \mathcal{X}_{i_2},\ e = (i_1, i_2) \in E_E\}$ and corresponds to the external events. This means that an explicit switch from system i_1 to i_2 is only admissible if $x \in \Lambda_e$ with $e = (i_1, i_2)$. In a similar way to $\mathcal{L}_E$, we define $\mathcal{L}_I$ as $\mathcal{L}_I \overset{\text{def}}{=} \{\Gamma_e : \Gamma_e \subseteq \mathbb{R}^n,\ \emptyset \neq \Gamma_e = \mathcal{X}_{i_1} \cap \mathcal{X}_{i_2},\ e = (i_1, i_2) \in E_I\}$ for the internal events. When subsystem i_1 is active and the state trajectory hits Γ_e with $e = (i_1, i_2) \in E_I$, the event e might be triggered and the system could switch to subsystem i_2.

A switched system can be characterized by the evolution of continuous as well as discrete states. As a consequence thereof, we get a timed sequence of active subsystems which can be encoded by a so–called switching sequence. We were facing switching sequences already in the section about systems with implicit switches. We will see that the same concept can also be applied to systems with both explicit and implicit switches.

Definition 1.8 (Switching Sequence)
A *switching sequence* σ in $\mathcal{T}$ is a timed sequence of active modes combined with its activation time instants, i.e., $\sigma = \{(t_0, i_0), (t_1, i_1), \ldots, (t_N, i_N)\}$, where $0 \leqslant N < \infty$, $t_s = t_0 \leqslant t_1 \leqslant \ldots \leqslant t_N \leqslant t_f$, and $i_n \in I$ for $0 \leqslant n \leqslant N$. △

Starting from subsystem i_0 at t_0, the switching sequence from Definition 1.8 indicates a switch from subsystem i_{n-1} to i_n at t_n for $1 \leqslant n \leqslant N$ and subsystem i_n stays active in $[t_n, t_{n+1}]$. For the switched system to be well–behaved, it is generally assumed that there occur only finitely many switchings in finite amount of time. The case with infinitely many switchings is known as *chattering* or Zeno *phenomenon*.

The switching pairs (t_n, i_n) from σ either originate from explicit switches or from implicit switches. The switching sequence denoted by $\sigma_E = \{(t_0, i_0), (t_1^E, i_1^E), \ldots, (t_{N_1}^E, i_{N_1}^E)\}$ corresponds to the explicit switches and $\sigma_I = \{(t_0, i_0), (t_1^I, i_1^I), \ldots, (t_{N_2}^I, i_{N_2}^I)\}$ to the implicit switches. One obtains the overall switching sequence as the combination of σ_E and σ_I as $\sigma = \sigma_E \cup \sigma_I$.

A switched system is exogenously affected by σ_E, whereas, along with the evolution of the state trajectory $\boldsymbol{x}$, σ_I is generated implicitly. σ_E together with σ_I then lead to the overall σ. An exogenous control σ_E is said to be *valid* if for given initial conditions (x_s, i_0) the evolution of the system generates a non–blocking state trajectory and a non–chattering σ.

The non–blocking property plays an important role in the well–posedness of a switched system. Roughly speaking, we call a switched system non–blocking if there exist infinite executions for all initial values (existence property). Another ingredient of well–posedness is the determinism, i.e., infinite executions are unique in case of existence (uniqueness property). Well–posed problems usually have the property that their solution changes continuously with the input data. However, certain boundaries in the state space of switched systems may separate regions of initial states leading to broadly distinct trajectories across those boundaries. For this reason, continuous dependence of solutions on input data may be a requirement that is too strong for switched systems. Certain aspects of well–posedness that are of relevance for this thesis are discussed in the following Section 1.2. The interested reader finds an extensive discussion e.g. in Lunze and Lamnabhi-Lagarrigue [304].

For the reader's convenience, we specify the switched system type under consideration in this work:

Definition 1.9 (IVP in ODEs with Explicit and Implicit Switches)
Let $\mathcal{T} \stackrel{\text{def}}{=} [t_s, t_f] \subset \mathbb{R}$ be a compact interval with $t_s < t_f$. An *IVP in ODEs with explicitly and implicitly defined switches* is given as a system of $n \in \mathbb{N}$ first–order ODEs and n initial conditions as

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{v}(t), \operatorname{sgn}(\boldsymbol{\sigma}(\boldsymbol{x}(t)))), \quad t \in \mathcal{T}, \tag{1.16a}$$

$$\boldsymbol{x}(t_s) = x_s, \tag{1.16b}$$

where the right–hand–side $\boldsymbol{f} : \mathcal{T} \times \mathcal{X} \times \Omega \times \{0, \pm 1\}^{n_\sigma} \to \mathbb{R}^n$ and the unknown dynamic state $\boldsymbol{x} : \mathcal{T} \to \mathcal{X}$ are vector–valued functions. The system is affected by the sign structure of the switching function $\boldsymbol{\sigma}(\cdot)$ and another vector–valued function $\boldsymbol{v} : \mathcal{T} \to \Omega$, which attains only values from a finite discrete set $\Omega \stackrel{\text{def}}{=} \{v^1, v^2, \ldots, v^{n_\omega}\} \subseteq \mathbb{R}^{n_v}$ with cardinality $|\Omega| = n_\omega < \infty$. The initial condition is defined analogously to Definition 1.1. △

1.2 Towards Solutions of Switched Systems

We start our investigations on well–posedness of dynamical systems by dealing with smooth systems.

1.2.1 Theory of Smooth Initial Value Problems

We briefly investigate the well–posedness of the IVP (1.2). Following the definition of Hadamard, a problem is well–posed if a solution exists, the solution is unique and the solution depends continuously on the input data. The following result guarantees the existence of solutions to IVP (1.2).

Theorem 1.10 (Peano)
Let $\boldsymbol{f}(\cdot)$ be continuous on the region $\mathcal{R} \stackrel{\text{def}}{=} \{(t,x) : t_s \leqslant t \leqslant t_s + a,\ \|x - x_s\| \leqslant b\} \subset \mathbb{R} \times \mathbb{R}^n$ and let M denote the maximum of $\|\boldsymbol{f}(\cdot)\|$ on $\mathcal{R}$. Then there exists at least one solution $\boldsymbol{x}(\cdot)$ of the IVP

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t)), \quad \boldsymbol{x}(t_s) = x_s,$$

for $t \in [t_s, t_s + \alpha]$, where $\alpha = \min\{a, b/M\}$. The solution remains in the set $\{x : \|x - x_s\| \leqslant b\}$. △

Proof See Teschl [425, Theorem 2.19]. □

To show the uniqueness of IVP (1.2), we need Lipschitz continuity of the function $\boldsymbol{f}(\cdot)$.

Definition 1.11 (Lipschitz Continuity)
Let the function $\boldsymbol{f}(\cdot)$ be defined on $\mathcal{D} \subset \mathbb{R} \times \mathbb{R}^n$. We call $\boldsymbol{f}(\cdot)$ Lipschitz continuous on $\mathcal{D}$ with respect to x, if a Lipschitz constant $L > 0$ exists such that

$$\|\boldsymbol{f}(t, x_1) - \boldsymbol{f}(t, x_2)\| \leqslant L \cdot \|x_1 - x_2\|, \quad \forall\, (t, x_1), (t, x_2) \in \mathcal{D}.$$ △

Theorem 1.12 (Picard–Lindelöf)
Let $\boldsymbol{f}(\cdot)$ be continuous on an open subset $\mathcal{D} \subset \mathbb{R} \times \mathbb{R}^n$ and let $\mathcal{R} \subset \mathcal{D}$ where the region $\mathcal{R}$ is defined as $\mathcal{R} \stackrel{\text{def}}{=} \{(t,x) : t_s \leqslant t \leqslant t_s + a,\ \|x - x_s\| \leqslant b\}$. If $\boldsymbol{f}(\cdot)$ is additionally Lipschitz continuous with respect to x, and bounded on $\mathcal{R}$ with $\|\boldsymbol{f}(\cdot)\| \leqslant M$, then

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t)), \quad \boldsymbol{x}(t_s) = x_s,$$

has a unique solution $\boldsymbol{x}(\cdot)$ on $[t_s, t_s + \alpha]$, where $\alpha = \min\{a, b/M\}$. △

Proof See Teschl [425, Theorem 2.2]. □

A sufficient condition for Lipschitz continuity, as it is required in Theorem 1.12, is the differentiability of $\boldsymbol{f}(\cdot)$ with respect to x and the boundedness of $\boldsymbol{f}'_x(\cdot)$ on $\mathcal{R}$. In this case, one can choose L to be a bound on $\boldsymbol{f}'_x(\cdot)$ using any matrix norm, i.e., $L = \sup_{(t,x)\in\mathcal{R}} \left\|\boldsymbol{f}'_x(t,x)\right\|$. The proof of Theorem 1.12 also reveals that the solution $\boldsymbol{x}(\cdot)$ is continuously differentiable.
What remains to show for Hadamard well–posedness of Problem (1.2) is the continuous dependency of the input data. The input data for IVP (1.2) is given by the initial value x_s and the right–hand–side function $\boldsymbol{f}(\cdot)$.

Theorem 1.13 (Shampine and Gordon [407])
Let $\boldsymbol{f}(\cdot)$, $\boldsymbol{g}(\cdot)$ be continuous on the open set $\mathcal{D} \subset \mathbb{R} \times \mathbb{R}^n$ and $\boldsymbol{f}(\cdot)$ Lipschitz continuous in x with Lipschitz constant L. Let us assume that

$$\|\boldsymbol{f}(t, x) - \boldsymbol{g}(t, x)\| \leqslant \varepsilon$$

holds for all $(t, x) \in \mathcal{D}$. If $\boldsymbol{x}(\cdot)$ is a solution of the ODE (1.2a) and $\boldsymbol{y}(\cdot)$ a solution of

$$\dot{\boldsymbol{y}}(t) = \boldsymbol{g}(t, \boldsymbol{y}(t)), \quad t \in \mathcal{T},$$

and $(t, \boldsymbol{x}(t))$ as well as $(t, \boldsymbol{y}(t))$ lie in $\mathcal{D}$, then it holds

$$\|\boldsymbol{x}(t) - \boldsymbol{y}(t)\| \leqslant \{\|\boldsymbol{x}(t_s) - \boldsymbol{y}(t_s)\| + a\,\varepsilon\} \cdot \exp(L(t - t_s)). \qquad \triangle$$

1.2.2 Theory of Initial Value Problems with Switches

Because of the possibly discontinuous right–hand–side functions of IVPs with implicit switches there is no way to apply the standard theory for existence and uniqueness of solutions of differential equations, cf. Theorem 1.12. Indeed, it is easy to come up with examples of implicitly switched IVPs that have no solution in the classic sense. Here, a *classical solution* means a function that is continuously differentiable and satisfies the differential equation, i.e., it is a solution in the sense of the Picard–Lindelöf Theorem. In the following example, we provide a problem instance whose solution is no solution in the classic sense.

Example 1.14
Let us consider the following implicitly switched ODE

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t)) = 2 - \operatorname{sgn}(\boldsymbol{x}(t)) = \begin{cases} 3, & \boldsymbol{x}(t) < 0, \\ 2, & \boldsymbol{x}(t) = 0, \\ 1, & \boldsymbol{x}(t) > 0. \end{cases}$$

For any initial condition $\boldsymbol{x}(t_s) \neq 0$ we obtain a solution of the respective IVP

$$\boldsymbol{x}(t) = \begin{cases} 3t + C_1, & \boldsymbol{x}(t) < 0, \\ t + C_2, & \boldsymbol{x}(t) > 0, \end{cases}$$

where C_1 and C_2 are constants that are determined by the initial condition. Each solution reaches $\boldsymbol{x}(t_\sigma) = 0$ for a finite t_σ if the initial condition is chosen such that $\boldsymbol{x}(t_s) < 0$. Since $\dot{\boldsymbol{x}}(t) > 0$ the solution trajectory hits $\boldsymbol{x} = 0$ at a single instant t_σ. The solution cannot be a solution in the classical sense ($\dot{\boldsymbol{x}}$ is discontinuous at t_σ) as $\lim_{\varepsilon \nearrow 0} \dot{\boldsymbol{x}}(t_\sigma + \varepsilon) \neq \lim_{\varepsilon \searrow 0} \dot{\boldsymbol{x}}(t_\sigma + \varepsilon) \neq 2 - \operatorname{sgn}(0)$.

In the remainder of the current as well as in the following section, we present different solution concepts of switched IVPs and discuss conditions of the respective concepts that guarantee existence and uniqueness of solutions.

Towards Consistent Switches

In order to facilitate notation, we concentrate on the case of a system with a single switch, cf. Example 1.6. If a solution trajectory would fulfill either $\boldsymbol{\sigma}(\boldsymbol{x}(t)) > 0$ or $\boldsymbol{\sigma}(\boldsymbol{x}(t)) < 0$ over the complete horizon, we encounter the case of a standard IVP and therefore existence and uniqueness of a solution were guaranteed by Theorem 1.12. For now, since we are not interested in this case we focus our attention to the case where a solution satisfies $\boldsymbol{\sigma}(\boldsymbol{x}(t_\sigma)) = 0$ at a certain time instance t_σ of the horizon. At first, we deal with the case where $\boldsymbol{\sigma}(\boldsymbol{x}(t)) \neq 0$ for

any time instant t in a vicinity of t_σ and call the respective switches *consistent switches*. For this reason, we consider an IVP of the form

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t)) \stackrel{\text{def}}{=} \begin{cases} \boldsymbol{f}^-(t, \boldsymbol{x}(t)) & \text{if } \boldsymbol{\sigma}(\boldsymbol{x}(t)) < 0 \\ \boldsymbol{f}^+(t, \boldsymbol{x}(t)) & \text{if } \boldsymbol{\sigma}(\boldsymbol{x}(t)) > 0 \end{cases}, \; t \in \mathcal{T} \setminus \{t_\sigma\}, \quad \boldsymbol{x}(t_s) = x_s, \quad (1.17)$$

in the following and analyze it. Note, that the differential equation in (1.17) is not defined at the switching time instant t_σ. However, this is in accordance with our first solution concept as the ODE is solely not defined on a set of Lebesgue measure zero.

A Geometrical Interpretation Before we start with an analysis of the first solution concept, we have a look at the system defined in (1.17) from a geometrical point of view: under mild assumptions on the smoothness of the switching function $\boldsymbol{\sigma} : \mathbb{R}^n \longrightarrow \mathbb{R}$, an $(n-1)$–dimensional differentiable manifold in $\mathbb{R}^n$ is given by

$$\Sigma \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n \, : \, \boldsymbol{\sigma}(x) = 0\}.$$

Σ is defined as the null set of a smooth real–valued function on $\mathbb{R}^n$. Likewise, subspaces $\mathcal{S}^-$ and $\mathcal{S}^+$ can be implicitly defined by means of the switching function $\boldsymbol{\sigma}(\cdot)$ as

$$\mathcal{S}^- \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n \, : \, \boldsymbol{\sigma}(x) < 0\} \quad \text{and} \quad \mathcal{S}^+ \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n \, : \, \boldsymbol{\sigma}(x) > 0\}.$$

The space $\mathbb{R}^n$ can therefore be split into the two subspaces $\mathcal{S}^-$ and $\mathcal{S}^+$ by the hypersurface Σ such that $\mathbb{R}^n = \mathcal{S}^- \dot{\cup} \Sigma \dot{\cup} \mathcal{S}^+$.

A First Solution Concept It is obvious that the dynamics of IVP (1.17) might change abruptly when the state trajectory hits the zero manifold Σ (see e.g. Example 1.14). For this reason, one cannot expect solution trajectories to be differentiable at those points. We present a solution concept that provides solutions under rather mild assumptions. To this end, we investigate the integral equation

$$\boldsymbol{x}(t) = x_s + \int_{t_s}^{t} \boldsymbol{f}(\tau, \boldsymbol{x}(\tau)) \, \mathrm{d}\tau. \quad (1.18)$$

It is a well–known fact that the problem formulations given in (1.17) and (1.18) are equivalent if $\boldsymbol{f}(\cdot)$ is continuous in some (t, x)–domain $\mathcal{D}$. Beyond that, the integral in (1.18) makes also sense for functions $\boldsymbol{f}(\cdot)$ undergoing discontinuities. We use this fact as a leverage point in order to relax the continuity restriction on $\boldsymbol{f}(\cdot)$.

We consider a function $\boldsymbol{x}(\cdot)$ to be a solution of IVP (1.17) in the *extended sense* if $\boldsymbol{x}(\cdot)$ is absolutely continuous (see Definition 2.30), satisfies the ODE almost everywhere on $\mathcal{T}$, and satisfies the initial condition. Note that the second condition makes sense due to the fact that $\boldsymbol{x}(\cdot)$ is assumed to be absolutely continuous and absolute continuity implies the existence of $\dot{\boldsymbol{x}}(\cdot)$ almost everywhere on $\mathcal{T}$.

Differential equations that are interpreted in this way are sometimes called Carathéodory *equations*. A solution of IVP (1.17) in the extended sense is also called Carathéodory *solution*. This solution concept has the advantage that it supersedes the need of specifying the value of

$\boldsymbol{f}(\cdot)$ on the zero manifold Σ, if the solution trajectory hits Σ from one side and immediately leaves it on the other side. One can easily show that the extended solution concept coincides with the ordinary one when $\boldsymbol{f}(\cdot)$ is a continuous functions. The following result provides criteria that guarantee the existence of Carathéodory solutions.

Theorem 1.15 (Carathéodory)
Let $\boldsymbol{f}(\cdot)$ be defined on the region $\mathcal{R} \stackrel{\text{def}}{=} \{(t,x) : |t-t_s| \leqslant a,\ \|x-x_s\| \leqslant b\}$. Let $\boldsymbol{f}(t,x)$ be continuous in x for each fixed t and measurable in t for each fixed x. Let a Lebesgue–integrable function $\boldsymbol{m}(\cdot)$ be defined on the interval $|t-t_s| \leqslant a$ such that

$$\|\boldsymbol{f}(t,x)\| \leqslant \boldsymbol{m}(t), \quad (t,x) \in \mathcal{R}.$$

Then IVP (1.17) has a solution in the sense of Carathéodory on some interval $|t-t_s| \leqslant \alpha$ with $\alpha > 0$.$_\triangle$

Proof See e.g. Coddington and Levinson [114, Theorem 1.1]. □

For further information on generalized ODE solution concepts we refer the reader to the textbooks of Coddington and Levinson [114] and Hale [220]. In particular, Hale [220, Theorem 5.3] provides a uniqueness result for Carathéodory solutions.

Beyond Consistent Switches

Dependent on the vector fields determined by $\boldsymbol{f}^-(\cdot)$ and $\boldsymbol{f}^+(\cdot)$ solution trajectories cross the zero manifold Σ instantaneously or not. Let us therefore suppose that x_σ is a point on the zero manifold Σ. If we consider the vectors $\boldsymbol{f}^-(x_\sigma)$ and $\boldsymbol{f}^+(x_\sigma)$ and their relation to the tangent space of the hypersurface Σ at x_σ, we distinguish the following four cases:

(i) Both vectors point inside $\mathcal{S}^+$. A solution trajectory can only come from $\mathcal{S}^-$ and will continue in $\mathcal{S}^+$. This case is covered by the solution concept of Carathéodory.

(ii) Both vectors point inside $\mathcal{S}^-$. In the same way as in the previous case a state trajectory comes from $\mathcal{S}^+$ and will continue in $\mathcal{S}^-$.

(iii) The vector $\boldsymbol{f}^+(x_\sigma)$ points into $\mathcal{S}^+$ and $\boldsymbol{f}^-(x_\sigma)$ into $\mathcal{S}^-$. This case cannot occur for solution trajectories arriving from $\mathcal{S}^+$ or $\mathcal{S}^-$. If x_σ is taken as initial condition there is no unique solution to (1.18). Switches where solution trajectories may leave Σ in both directions are often called *bifurcations*. Switches of this type are beyond the scope of this thesis.

(iv) The vector $\boldsymbol{f}^+(x_\sigma)$ points into $\mathcal{S}^-$ and $\boldsymbol{f}^-(x_\sigma)$ into $\mathcal{S}^+$. In this case solutions in the sense of Carathéodory do not exist since the set $\{t : \boldsymbol{\sigma}(\boldsymbol{x}(t)) = 0\}$ is not of Lebesgue–measure zero. Hence, there is the need for different solution concepts.

The Filippov solution concept, presented in the following Section 1.3, mainly deals with case (iv). There are two obvious ways to approach this case, both being physically motivated.
The first interpretation assumes the right–hand–side as a simplified version of another function $\widetilde{\boldsymbol{f}}(\cdot)$ which is not discontinuous across the zero manifold Σ, but changes in a steep gradient from $\boldsymbol{f}^+(\cdot)$ on one side to $\boldsymbol{f}^-(\cdot)$ on the other side of Σ. In case (iv), a solution of the system

$\dot{\boldsymbol{x}}(t) = \widetilde{\boldsymbol{f}}(t, \boldsymbol{x}(t))$ will have the tendency to slide along the zero manifold since it is pushed towards it from the vector field in a vicinity.
For the second interpretation, we assume that there is a switching controller monitoring the sign of $\boldsymbol{\sigma}(\boldsymbol{x}(\cdot))$, which is responsible for the transition from the regime described by $\boldsymbol{f}^+(\cdot)$ to the one described by $\boldsymbol{f}^-(\cdot)$. For instance, caused by physical limitations, the switching cannot exactly occur when $\boldsymbol{\sigma}(\boldsymbol{x}(\cdot))$ crosses the zero value, but at some nearby instant of time. In case (iv), this would result in a chattering behavior of a solution trajectory. This means that the dynamics would quickly switch from one dynamics to the other and back again. Just as in the first interpretation, a motion more or less along the zero manifold would take place.
The presented two interpretations illustrate that there are good reasons to allow for solutions along Σ in cases of type (iv). However, due to different physical mechanisms, the situation may be not completely specified by the two functions $\boldsymbol{f}^+(\cdot)$ on $\mathcal{S}^+$ and $\boldsymbol{f}^-(\cdot)$ on $\mathcal{S}^-$.

Example 1.16
Let us consider the following implicitly switched ODE:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t)) = 2 - 3\,\mathrm{sgn}(\boldsymbol{x}(t)) = \begin{cases} 5, & \boldsymbol{x}(t) < 0, \\ 2, & \boldsymbol{x}(t) = 0, \\ -1, & \boldsymbol{x}(t) > 0. \end{cases}$$

For any initial condition $\boldsymbol{x}(t_s) \neq 0$ we obtain a solution of the respective IVP

$$\boldsymbol{x}(t) = \begin{cases} 5t + C_1, & \boldsymbol{x}(t) < 0, \\ -t + C_2, & \boldsymbol{x}(t) > 0, \end{cases}$$

where C_1 and C_2 are constants that are determined by the initial condition. Each solution reaches $\boldsymbol{x}(t_\sigma) = 0$ for a finite t_σ. Since $\dot{\boldsymbol{x}}(t) > 0$ for $\boldsymbol{x}(t) < 0$ and $\dot{\boldsymbol{x}}(t) < 0$ for $\boldsymbol{x}(t) > 0$ the solution trajectory can not leave $\boldsymbol{x} = 0$ when it arrives at it. Hence, the solution will stay at $\boldsymbol{x} = 0$, which implies $\dot{\boldsymbol{x}} = 0$. This cannot be a solution in the classical sense since $0 \neq 2 - 3\,\mathrm{sgn}(0)$.

1.3 Filippov Theory

For Example 1.16 neither the classic solution concept nor solutions in the sense of Carathéodory are applicable. The theory of Filippov provides a generalized definition of the solution of switched systems in the sense that the definition holds for a larger class of differential equations, cf. Filippov [162, 163]. Solutions in the sense of Filippov are continuous in time. Jump conditions are not described by the theory of Filippov.
Before we investigate Filippov's theory for the general case, we convey its idea based on Example 1.16. A natural idea to extend the classic solution concept is to replace the right–hand–side $\boldsymbol{f}(\cdot)$ with a *set–valued* function $\boldsymbol{F}(\cdot)$ such that $\boldsymbol{f}(\cdot)$ and $\boldsymbol{F}(\cdot)$ are identical at points where $\boldsymbol{f}(\cdot)$ is continuous in x. At points for which $\boldsymbol{f}(\cdot)$ is discontinuous in x there is a suitable choice for $\boldsymbol{F}(\cdot)$ required. The differential equation is then replaced by the *differential inclusion*

$$\dot{\boldsymbol{x}}(t) \in \boldsymbol{F}(t, \boldsymbol{x}(t)).$$

The definition of $\boldsymbol{F}(\cdot)$ at points of discontinuity happens by means of the *generalized differential*. For a detailed introduction to the subdifferential calculus the reader is referred to the

excellent textbook of Clarke [112]. Here, the *generalized derivative* of a function $\boldsymbol{x} : \mathbb{R} \longrightarrow \mathcal{X}$ at a point t is defined as *any* value $\dot{\boldsymbol{x}}_\alpha(t)$ included between its left and right derivatives, cf. Curnier [117] and Clarke et al. [113]. Such a representative can be obtained by means of a convex combination of the left and right derivatives as

$$\dot{\boldsymbol{x}}_\alpha(t) = \alpha \cdot \dot{\boldsymbol{x}}^-(t) + (1-\alpha) \cdot \dot{\boldsymbol{x}}^+(t), \quad 0 \leqslant \alpha \leqslant 1.$$

The values $\dot{\boldsymbol{x}}^-(t)$ and $\dot{\boldsymbol{x}}^+(t)$ are given as $\boldsymbol{f}^-(t, \boldsymbol{x}(t))$ and $\boldsymbol{f}^+(t, \boldsymbol{x}(t))$, respectively. The generalized differential of $\boldsymbol{x}(\cdot)$ at t, which is denoted by $\partial \boldsymbol{x}(t)$, is then the set of all the generalized derivatives of $\boldsymbol{x}(\cdot)$ at t. More concrete, it is the convex hull of the derivative extremes, i.e.,

$$\begin{aligned} \partial \boldsymbol{x}(t) &= \operatorname{conv}\left\{\dot{\boldsymbol{x}}^-(t), \dot{\boldsymbol{x}}^+(t)\right\} \\ &= \left\{\dot{\boldsymbol{x}}_\alpha(t) \in \mathbb{R}^n \,:\, \dot{\boldsymbol{x}}_\alpha(t) = \alpha \cdot \dot{\boldsymbol{x}}^-(t) + (1-\alpha) \cdot \dot{\boldsymbol{x}}^+(t),\ \alpha \in [0,1]\right\}, \end{aligned} \tag{1.19}$$

where $\operatorname{conv}(\mathcal{A})$ denotes the smallest closed convex set containing $\mathcal{A}$. We define the set–valued sign function as the generalized differential of $|x|$ such that

$$\operatorname{Sgn}(x) \stackrel{\text{def}}{=} \partial\,|x| = \begin{cases} \{-1\}, & x < 0, \\ [-1, 1], & x = 0, \\ \{1\}, & x > 0. \end{cases}$$

The differential inclusion formulation of the ODE from Example 1.16 is then given as

$$\dot{\boldsymbol{x}}(t) \in 2 - 3\operatorname{Sgn}(\boldsymbol{x}(t)). \tag{1.20}$$

If we solve the problem *forward in time* with the initial condition $\boldsymbol{x}(0) = 0$, we see that $\boldsymbol{x}(t) \equiv 0$ is a unique solution. However, the solutions of (1.20) with initial condition $\boldsymbol{x}(-1) = 1$ and initial condition $\boldsymbol{x}(-1) = -5$ evolve both to $\boldsymbol{x}(0) = 0$. This shows the non–uniqueness of the solution of (1.20) if it is solved *backward in time*.

In the following, the idea of replacing a switched ODE with a differential inclusion is transferred from one dimension to dimension n. For the sake of simplicity, we assume the case with a single switch, cf. Example 1.6. As we have done earlier, we split the space $\mathbb{R}^n$ into two subspaces $\mathcal{S}^-$ and $\mathcal{S}^+$ by a hypersurface Σ such that $\mathbb{R}^n = \mathcal{S}^- \dot\cup \Sigma \dot\cup \mathcal{S}^+$. The subspaces $\mathcal{S}^-$ and $\mathcal{S}^+$ are implicitly defined by means of the switching function $\boldsymbol{\sigma} : \mathbb{R}^n \longrightarrow \mathbb{R}$ as

$$\mathcal{S}^- \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n \,:\, \boldsymbol{\sigma}(x) < 0\}, \quad \mathcal{S}^+ \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n \,:\, \boldsymbol{\sigma}(x) > 0\},$$

and the hypersurface Σ as

$$\Sigma \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n \,:\, \boldsymbol{\sigma}(x) = 0\}.$$

We consider the nonlinear system with discontinuous right–hand–side

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t)) \stackrel{\text{def}}{=} \begin{cases} \boldsymbol{f}^-(t, \boldsymbol{x}(t)), & \text{if } \boldsymbol{x}(t) \in \mathcal{S}^-, \\ \boldsymbol{f}^+(t, \boldsymbol{x}(t)), & \text{if } \boldsymbol{x}(t) \in \mathcal{S}^+, \end{cases} \quad t \in \mathcal{T} \setminus \Sigma, \quad \boldsymbol{x}(t_s) = x_s. \tag{1.21}$$

We assume that $\boldsymbol{f}(\cdot)$ fulfills all assumptions from Theorem 1.12 in $\mathbb{R}^n \setminus \Sigma$, such that the solution $\boldsymbol{x}(\cdot)$ within $\mathcal{S}^-$ and $\mathcal{S}^+$ exists and is unique. Moreover, we assume that the smooth functions $\boldsymbol{f}^-(\cdot)$ and $\boldsymbol{f}^+(\cdot)$ uniquely extend to smooth functions on $\mathcal{S}^- \cup \Sigma$ and $\mathcal{S}^+ \cup \Sigma$, respectively. However, the values $\dot{\boldsymbol{x}}^-(t_\sigma)$ and $\dot{\boldsymbol{x}}^+(t_\sigma)$ at any $t_\sigma \in \Sigma$ do not necessarily coincide leading to discontinuities across Σ.
Function $\boldsymbol{f}(\cdot)$, as defined in Problem (1.21), is not defined for t with $\boldsymbol{x}(t) \in \Sigma$. This means that there is some freedom on how to extend the vector field on Σ. To this end, we investigate the set–valued extension $\boldsymbol{F} : \mathcal{T} \times \mathcal{X} \longrightarrow \mathbb{R}^n$ of $\boldsymbol{f}(\cdot)$ for $x_\sigma \in \Sigma$, which is given as

$$\boldsymbol{F}(t, x_\sigma) \stackrel{\text{def}}{=} \operatorname{conv}\left\{ y \in \mathbb{R}^n \,:\, y = \lim_{x \to x_\sigma} \boldsymbol{f}(t,x),\ x \in \mathbb{R}^n \setminus \Sigma \right\}. \tag{1.22}$$

Due to the assumptions on the function $\boldsymbol{f}(\cdot)$ all the limits exist. The convexification of the switched IVP (1.21) into the convex differential inclusion

$$\dot{\boldsymbol{x}}(t) \in \boldsymbol{F}(t, \boldsymbol{x}(t)) \stackrel{\text{def}}{=} \begin{cases} \boldsymbol{f}^-(t, \boldsymbol{x}(t)), & \text{if } \boldsymbol{x}(t) \in \mathcal{S}^-, \\ \operatorname{conv}\left\{ \boldsymbol{f}^-(t, \boldsymbol{x}(t)), \boldsymbol{f}^+(t, \boldsymbol{x}(t)) \right\}, & \text{if } \boldsymbol{x}(t) \in \Sigma, \\ \boldsymbol{f}^+(t, \boldsymbol{x}(t)), & \text{if } \boldsymbol{x}(t) \in \mathcal{S}^+, \end{cases} \tag{1.23a}$$

$$\boldsymbol{x}(t_s) = x_s, \tag{1.23b}$$

where the convex set from (1.22) can be expressed on Σ by means of the two right–hand–sides $\boldsymbol{f}^-(\cdot)$ and $\boldsymbol{f}^+(\cdot)$ and

$$\operatorname{conv}\{f^-, f^+\} = \left\{ f \in \mathbb{R}^n \,:\, f = \alpha \cdot f^- + (1-\alpha) \cdot f^+,\ \alpha \in [0,1] \right\} \tag{1.24}$$

is known as Filippov*'s convex method.* Considering the IVP (1.21) as a mathematical model of a physical system, it is crucial that we deal with a solution concept which guarantees existence of solutions. Hence, the choice of the set valued extension $\boldsymbol{F}(\cdot)$ of $\boldsymbol{f}(\cdot)$ should be suitable in the sense that the existence of a solution can be guaranteed. The existence of solutions of a differential inclusion can be ensured with the notion of upper semi–continuity of set–valued functions.

Definition 1.17 (Upper Semi–Continuity of Set–Valued Functions)
A set–valued function $\boldsymbol{F}(\cdot)$ is called *upper semi–continuous* in x if

$$\lim_{y \to x} \left(\sup_{a \in \boldsymbol{F}(y)} \inf_{b \in \boldsymbol{F}(x)} \|a - b\| \right) \longrightarrow 0. \qquad \triangle$$

This condition is equivalent to the condition that for all $\varepsilon > 0$ there exists a $\delta > 0$ such that $\|x - y\| < \delta$ implies $\boldsymbol{F}(y) \subset \boldsymbol{F}(x) + \mathcal{U}_\varepsilon(0)$. As with Carathéodory solutions the following result guarantees the existence of differential inclusion solution trajectories that are absolutely continuous (see Definition 2.30).

Theorem 1.18 (Existence of Solution of a Differential Inclusion)
Let $\boldsymbol{F}(\cdot)$ be a set valued function. Assuming $\boldsymbol{F}(\cdot)$ to be upper semi–continuous and $\boldsymbol{F}(t,x)$ to be closed, convex, and bounded for all $t \in \mathbb{R}$ and $x \in \mathbb{R}^n$, then for each $x_s \in \mathbb{R}^n$ there exists a $\tau > 0$ and an

absolutely continuous function $\boldsymbol{x}(\cdot)$ defined on $[t_s, t_s + \tau]$, which is a solution of the IVP

$$\dot{\boldsymbol{x}}(t) \in \boldsymbol{F}(t, \boldsymbol{x}(t)), \quad \boldsymbol{x}(t_s) = x_s. \qquad \triangle$$

Proof See Aubin and Cellina [19]. □

For Example 1.16, we check the conditions of the theorem for the differential inclusion

$$\dot{\boldsymbol{x}}(t) \in 2 - a\,\mathrm{Sgn}(\boldsymbol{x}(t)), \quad |a| > 2.$$

It must hold $0 \in 2 - a\,\mathrm{Sgn}(0)$ to allow for the solution $\boldsymbol{x}(t) = 0$. This is true since $|a| > 2$. Hence, $\mathrm{Sgn}(0)$ must be defined to be the set $[-1, +1]$ in order to guarantee the existence of a solution. Together with the values of $\mathrm{sgn}(x) = \pm 1$ for $x \neq 0$, the set $[-1, +1]$ is upper semi–continuous, closed, convex, and bounded. Therefore, the conditions of Theorem 1.18 are satisfied.
A *solution in the sense of* Filippov for an implicitly switched system of type (1.21) can be defined by means of Filippov's convex method and the existence result from Theorem 1.18.

Definition 1.19 (Solution in the Sense of Filippov)
We call an absolute continuous function $\boldsymbol{x} : [t_s, t_s + \tau] \longrightarrow \mathbb{R}^n$ a *solution* of IVP (1.21) *in the sense of* Filippov if for almost all $t \in [t_s, t_s + \tau]$ it holds that

$$\dot{\boldsymbol{x}}(t) \in \boldsymbol{F}(t, \boldsymbol{x}(t)),$$

where $\boldsymbol{F}(t, \boldsymbol{x}(t))$ is defined as in (1.23). $\triangle$

For solutions $\boldsymbol{x}(\cdot)$ in the sense of Filippov, we review some properties. In a region where $\boldsymbol{x}(\cdot)$ is smooth, i.e., $\boldsymbol{x}(t) \in \mathcal{S}^- \cup \mathcal{S}^+$, it must hold $\boldsymbol{f}(t, \boldsymbol{x}(t)) = \boldsymbol{F}(t, \boldsymbol{x}(t))$. If $\boldsymbol{x}(\cdot)$ slides along a switching boundary, i.e., $\boldsymbol{x}(t) \in \Sigma$, then $\dot{\boldsymbol{x}}(t) \in \boldsymbol{F}(t, \boldsymbol{x}(t))$. However, at time instances t_σ where the solution $\boldsymbol{x}(\cdot)$ enters the switching manifold Σ or leaves from Σ, the state derivative $\dot{\boldsymbol{x}}(t_\sigma)$ is not defined. Here, a solution trajectory $\boldsymbol{x}(\cdot)$ enters or leaves Σ if for any $\varepsilon > 0$ there exists a $t^* \in t_\sigma + \mathcal{U}_\varepsilon(0) \setminus \{0\}$ such that $\boldsymbol{x}(t_\sigma) \in \Sigma$ and $\boldsymbol{x}(t^*) \notin \Sigma$. The set of t for which the solution trajectory $\boldsymbol{x}(t)$ enters or leaves Σ is of Lebesgue measure zero.
Theorem 1.18 guarantees the existence of a solution on an interval $[t_s, t_s + \tau]$ with $\tau > 0$. In order to achieve existence over the whole horizon, we need further assumptions: let $\boldsymbol{f}(t, x)$ be linearly bounded for $x \notin \Sigma$, i.e., there exist positive constants c_1 and c_2 such that for all $t \in [0, \infty)$ and $x \in \mathcal{S}^- \cup \mathcal{S}^+$ it holds

$$\|\boldsymbol{f}(t, x)\| \leqslant c_1 \|x\| + c_2.$$

If additionally $\boldsymbol{F}(\cdot)$ is bounded at values (t, x) for which $\boldsymbol{F}$ is set–valued, then a solution $\boldsymbol{x}(\cdot)$ of IVP (1.23) exists on $[t_s, \infty)$, cf. Aubin and Cellina [19] and Clarke et al. [113]. These assumptions are not sufficient to guarantee the uniqueness of a solution.

1.4 Sliding Modes

The previous section was dedicated to introduce Filippov systems and to answer the question if solutions of this system type exist. Now, we examine the uniqueness of solutions. Let us

consider a solution of IVP (1.23) and suppose that $x_s \notin \Sigma$. Since $\boldsymbol{f}^-(\cdot)$ and $\boldsymbol{f}^+(\cdot)$ are assumed to be smooth such a solution is locally unique.
For this reason, we have to consider the case that $x_s \in \Sigma$ or that the solution trajectory $\boldsymbol{x}(\cdot)$ enters the switching manifold Σ at a finite time instant t_σ. Informally speaking, there are the following two cases that have to be distinguished:

(i) The solution trajectory leaves Σ and enters either $\mathcal{S}^-$ or $\mathcal{S}^+$.

(ii) The solution trajectory remains in Σ.

In order to decide which of these cases may occur and possibly if the resulting solution trajectory can be uniquely determined, we have to characterize the vector field in a vicinity of Σ. Based on the vector field characterization, we will identify three different modes for solution trajectories, namely the transversal intersection mode, the attracting sliding mode, and finally the repulsive sliding mode.
Prior to that, there is some notation required. All switching functions $\boldsymbol{\sigma}(\cdot)$ that we consider are supposed to be at least continuous. It can be easily seen that a switching manifold Σ is not determined by a unique switching function $\boldsymbol{\sigma}(\cdot)$. Different switching functions can define the same Σ. We assume that the switching function $\boldsymbol{\sigma}(\cdot)$ is chosen such that it holds

$$\mathbf{0} \notin \partial \boldsymbol{\sigma}(x) \quad \forall x \in \Sigma.$$

In the case of a locally smooth Σ, the unit normal vector to Σ at a point $x \in \Sigma$ is denoted by $\boldsymbol{n} : \Sigma \longrightarrow \mathbb{R}^n$. The vector $\boldsymbol{n}(x)$, which is perpendicular to the tangent plane at x, is given as

$$\boldsymbol{n}(x) = \frac{\nabla \boldsymbol{\sigma}(x)}{\|\nabla \boldsymbol{\sigma}(x)\|}. \tag{1.25}$$

In contrast, we make use of the generalized differential for locally non–smooth Σ and choose $\boldsymbol{n}(x)$ such that

$$\boldsymbol{n}(x) \in \left\{ \frac{y}{\|y\|} \,:\, y \in \partial \boldsymbol{\sigma}(x) \right\}, \tag{1.26}$$

where $\partial \boldsymbol{\sigma}(x)$ is assumed to be bounded. Note that the projections of $\boldsymbol{f}^-(t,x)$ and $\boldsymbol{f}^+(t,x)$ onto the normal of the zero manifold Σ at $(t,x) \in \mathcal{T} \times \Sigma$ can be expressed by means of $\boldsymbol{n}(x)$ as $\boldsymbol{n}^T(x)\boldsymbol{f}^-(t,x)$ and $\boldsymbol{n}^T(x)\boldsymbol{f}^+(t,x)$, respectively. These notations enable us to characterize the vector field in a vicinity of Σ.

Transversal Intersection Mode

A large class of switched dynamic systems satisfies the *transversality condition*. We say that the transversality condition holds at a point $(t,x) \in \mathcal{T} \times \Sigma$ if

$$\left[\boldsymbol{n}^T(x)\boldsymbol{f}^-(t,x)\right] \cdot \left[\boldsymbol{n}^T(x)\boldsymbol{f}^+(t,x)\right] > 0. \tag{1.27}$$

Let us assume that a solution of IVP (1.23) is given by $\boldsymbol{x}(\cdot)$ and that there is a finite time instant t_σ such that $\boldsymbol{x}(t_\sigma) \in \Sigma$. If the transversality condition (1.27) holds at $(t_\sigma, \boldsymbol{x}(t_\sigma))$, then Σ will be left by the solution trajectory.
More precisely, if $\boldsymbol{n}^T(\boldsymbol{x}(t_\sigma))\boldsymbol{f}^-(t_\sigma, \boldsymbol{x}(t_\sigma)) < 0$ the solution trajectory will enter $\mathcal{S}^-$ and the ODE in (1.23) holds with $\boldsymbol{f} = \boldsymbol{f}^-$. Conversely, if $\boldsymbol{n}^T(\boldsymbol{x}(t_\sigma))\boldsymbol{f}^-(t_\sigma, \boldsymbol{x}(t_\sigma)) > 0$ the solution trajectory will enter $\mathcal{S}^+$ and the ODE in (1.23) holds with $\boldsymbol{f} = \boldsymbol{f}^+$. Note, that $\boldsymbol{n}(x_\sigma)$ points into $\mathcal{S}^+$ since it points into the direction of steepest ascent of $\boldsymbol{\sigma}(\cdot)$ at x_σ.
Any solution of IVP (1.23) with an initial condition starting in $\mathcal{S}^-$ that hits Σ at a finite time instance where the transversality condition is satisfied will cross it transversally and proceed in $\mathcal{S}^+$. Here, the word transversal does not refer to the vector field $\boldsymbol{f}$ but to the solution which is transversal to Σ. Hence, any solution of IVP (1.23) which has an initial value in $\mathcal{S}^-$ and exposes a transversal intersection at Σ exists and is unique. An analogous statement holds for solutions with an initial value in $\mathcal{S}^+$.
The following example shows that the choice of the set in (1.23a) is crucial for the uniqueness of IVP (1.23).

Example 1.20
Let us consider the differential inclusion

$$\dot{\boldsymbol{x}}(t) \in \boldsymbol{F}(\boldsymbol{x}(t)) = \begin{cases} 3 + \operatorname{sgn}(\boldsymbol{x}(t)), & \boldsymbol{x}(t) \neq 0, \\ [-1, 5], & \boldsymbol{x}(t) = 0, \end{cases}$$

with initial condition $\boldsymbol{x}(0) = 0$. Since $\boldsymbol{F}(\cdot)$ is upper semi–continuous, non–empty, closed, convex and bounded for all (t, x), Theorem 1.18 guarantees the existence of solutions. Even though the transversality condition (1.27) holds at the initial point, the solution is not unique. This is due to the fact that $0 \in \boldsymbol{F}(0)$ which allows the solution trajectory to stay on Σ. A second admissible solution is given by $\boldsymbol{x}(t) = 4t$. Note that $\boldsymbol{F}(x)$ is not the smallest convex set containing $3 + \operatorname{sgn}(x)$. It holds $0 \notin \boldsymbol{F}(0)$ for the smallest convex set, which is given as $3 + \operatorname{Sgn}(x)$, and the resulting problem therefore has a unique solution in forward time.

Solutions satisfying the transversality condition are often referred to as "*classical*", and the switching behavior is often called "*consistent*". For classical solutions, the solution concept of Carathéodory is sufficient and one can skip the case (1.23a) of IVP (1.23). Consequently, the resulting problem becomes easier to formulate and to solve numerically.
In order to obtain a valid switching sequence, it was required in Section 1.1.2 that Zeno's phenomenon does not occur, i.e., there are not infinitely many switchings in finite amount of time. Unfortunately, Zeno's phenomenon exists for Filippov systems in form of *accumulation points*, cf. Filippov [163], Heemels [228], or Utkin [430]. A point $\tau \in \mathcal{T}$ is a *right–accumulation point* if there exist switching points τ_i, $i \in \mathbb{N}$, with $\tau_i < \tau$ such that $\tau = \lim_{i \to \infty} \tau_i$. A *left-accumulation point* is defined by changing "<" into ">". The following example describes a time reversed version of a system studied by Filippov which contains a right–accumulation point.

Example 1.21
Let us consider the differential inclusion

$$\dot{\boldsymbol{x}}(t) \in \boldsymbol{F}(\boldsymbol{x}(t)) = \begin{bmatrix} -\operatorname{Sgn}(\boldsymbol{x}_1(t)) + 2\operatorname{Sgn}(\boldsymbol{x}_2(t)) \\ -2\operatorname{Sgn}(\boldsymbol{x}_1(t)) - \operatorname{Sgn}(\boldsymbol{x}_2(t)) \end{bmatrix},$$

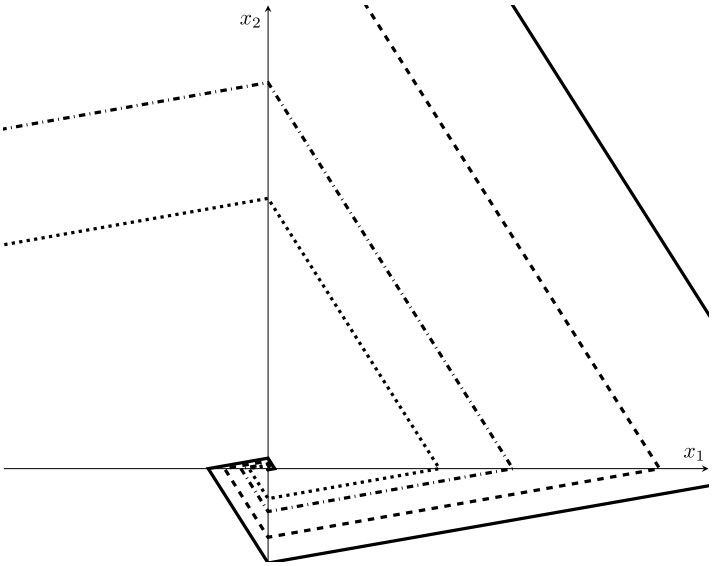


Figure 1.1: The figure depicts several trajectories of the differential inclusion system defined in Example 1.21 with different initial values. All trajectories tend towards the accumulation point at point $[0,0]^T$.

where $\boldsymbol{x}(t) \overset{\text{def}}{=} [\boldsymbol{x}_1(t), \boldsymbol{x}_2(t)]^T$. The piecewise constant system, which is characterized by the zero manifolds $\Sigma_1 = \{x_1 = 0\}$ and $\Sigma_2 = \{x_2 = 0\}$, is spiraling to the origin, where it has its only equilibrium, i.e., it holds $\mathbf{0} \in \boldsymbol{F}(\mathbf{0})$. The origin is located at the only intersection point of the zero manifolds. It can be easily checked that a transversal intersection occurs at $\Sigma_{1,2} \setminus \{\mathbf{0}\}$ at any time a solution trajectory hits one of the zero manifolds $\Sigma_{1,2}$. Several trajectories of the system with different initial values are depicted in Figure 1.1. Now, we aim to find a strict Lyapunov function for the system, i.e., a scalar and continuous function $\boldsymbol{V} : \mathbb{R}^2 \longrightarrow \mathbb{R}$ that is positive definite ($\boldsymbol{V}(\mathbf{0}) = 0$, $\boldsymbol{V}(x) > 0$ for $x \neq \mathbf{0}$), has continuous first partial derivatives, and whose derivative with respect to time along a trajectory of the system is negative ($\dot{\boldsymbol{V}}(\boldsymbol{x}(t)) < 0$). As a candidate we choose the positive definite function $\boldsymbol{V} : \mathbb{R}^2 \longrightarrow \mathbb{R}$, which is defined as the L^1–norm $\boldsymbol{V}(x_1, x_2) \overset{\text{def}}{=} |x_1| + |x_2|$. Its derivative along a solution $\boldsymbol{x}(\cdot)$ of the system can be calculated as

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{V}(\boldsymbol{x}(t)) = \boldsymbol{V}'_{x_1}(\boldsymbol{x}(t))\,\dot{\boldsymbol{x}}_1(t) + \boldsymbol{V}'_{x_2}(\boldsymbol{x}(t))\,\dot{\boldsymbol{x}}_2(t) = -\operatorname{Sgn}(\boldsymbol{x}_1(t))^2 - \operatorname{Sgn}(\boldsymbol{x}_2(t))^2,$$

and thus becomes set–valued when $\boldsymbol{x}_1(\cdot)$ or $\boldsymbol{x}_2(\cdot)$ vanishes. It follows that $\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{V}(\boldsymbol{x}(t)) = -2$ for $\boldsymbol{x}(t) \in \mathbb{R}^2 \setminus \{\Sigma_1 \cup \Sigma_2\}$ and $\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{V}(\boldsymbol{x}(t)) = [-2,-1]$ for $\boldsymbol{x}(t) \in \{\Sigma_1 \cup \Sigma_2\} \setminus \{\mathbf{0}\}$. Since the zero manifolds are crossed transversally outside the origin, it holds $\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{V}(\boldsymbol{x}(t)) = -2$ for almost all t as long as $\boldsymbol{x}(t) \in \mathbb{R}^2 \setminus \{\mathbf{0}\}$. By construction of $\boldsymbol{V}(\cdot)$ it has the function value $|\boldsymbol{x}_1(t_s)| + |\boldsymbol{x}_2(t_s)|$ at initial time t_s and ($\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{V}(\boldsymbol{x}(t)) = -2$) zero at time $\frac{1}{2}\boldsymbol{V}(\boldsymbol{x}(t_s))$. If $\boldsymbol{V}(x_1, x_2) = 0$ it holds $x_1 = x_2 = 0$ such that the system has an equilibrium point. Hence, solutions reach the equilibrium in finite time ΔT with

$$\Delta T = \frac{1}{2}\big(\,|\boldsymbol{x}_1(t_s)| + |\boldsymbol{x}_2(t_s)|\,\big).$$

However, these solutions cannot arrive at the origin without going through an infinite number of mode switches and consequently the event times contain a right–accumulation point.

If we have

$$\left[\boldsymbol{n}^T(x)\boldsymbol{f}^-(t,x)\right]\cdot\left[\boldsymbol{n}^T(x)\boldsymbol{f}^+(t,x)\right]\leqslant 0,$$

at some point $(t,x)\in\mathcal{T}\times\Sigma$, then the transversality condition is violated and one distinguishes several so–called "non–classical" cases.

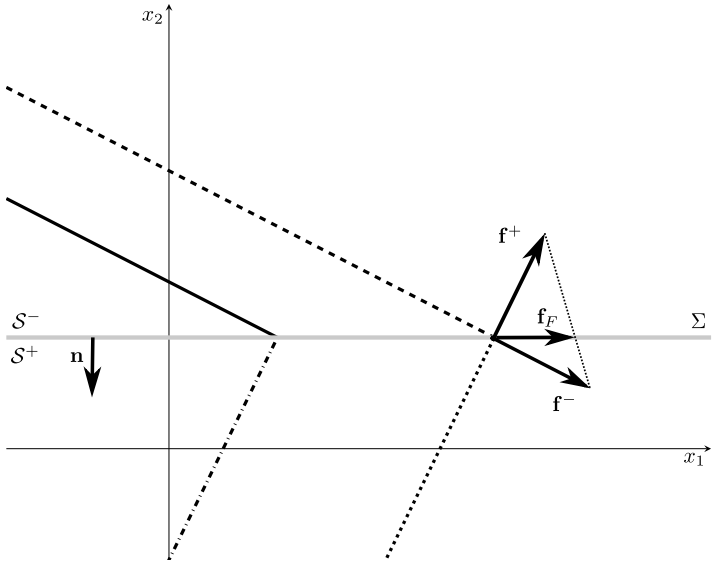


Figure 1.2: The figure illustrates the attracting sliding mode by reference to the system from Example 1.22. Some trajectories are depicted for different initial values. All trajectories coming from either $\mathcal{S}^-$ or $\mathcal{S}^+$ tend towards the zero manifold Σ. When the trajectories hit Σ they slide along the zero manifold. The sliding direction $\boldsymbol{f}_{\mathrm{F}}$ is calculated as convex combination of the vectors $\boldsymbol{f}^+=[2,4]^T$ and $\boldsymbol{f}^-=[4,-2]^T$ as $\boldsymbol{f}_{\mathrm{F}}=\frac{2}{3}\boldsymbol{f}^-+\frac{1}{3}\boldsymbol{f}^+=[10/3,0]^T$.

Sliding Mode

We encounter the so–called *sliding mode* through $(t,x)\in\mathcal{T}\times\Sigma$ if

$$\left[\boldsymbol{n}^T(x)\boldsymbol{f}^-(t,x)\right]\cdot\left[\boldsymbol{n}^T(x)\boldsymbol{f}^+(t,x)\right]<0. \tag{1.28}$$

This case is decomposed into two sub–cases, namely the *attracting sliding mode* and the *repulsive sliding mode.*

(i) Attracting Sliding Mode. Alternatively, switches of this type are often called "*inconsistent*" switches. A solution being in attracting sliding mode will hit Σ but cannot leave

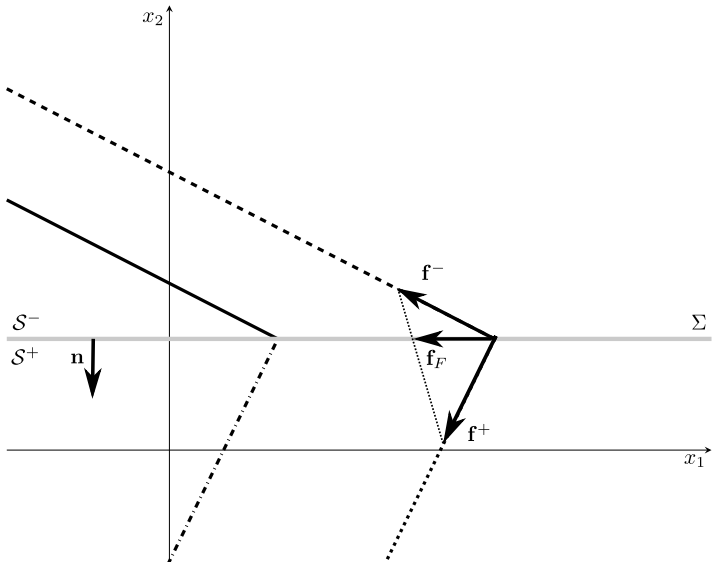


Figure 1.3: The figure illustrates the repulsive sliding mode by reference to the system from Example 1.23. Some trajectories with initial values at the zero manifold Σ are depicted. The trajectories tend either towards $\mathcal{S}^-$ with direction $\boldsymbol{f}^-$ or towards $\mathcal{S}^+$ with direction $\boldsymbol{f}^+$. Alternatively, the trajectories slide along Σ with direction $\boldsymbol{f}_{\mathrm{F}}$ being a convex combination of the vectors $\boldsymbol{f}^+ = [-2,-4]^T$ and $\boldsymbol{f}^- = [-4,+2]^T$, where $\boldsymbol{f}_{\mathrm{F}} = \frac{2}{3}\boldsymbol{f}^- + \frac{1}{3}\boldsymbol{f}^+ = [-10/3, 0]^T$.

it and will consequently slide along the zero manifold Σ. An attracting sliding mode at (t,x) occurs if

$$\boldsymbol{n}^T(x)\boldsymbol{f}^-(t,x) > 0 \quad \text{and} \quad \boldsymbol{n}^T(x)\boldsymbol{f}^+(t,x) < 0. \tag{1.29}$$

An extension to the vector field on Σ is provided by Filippov's theory, which is consistent with the interpretation in (1.19) and gives rise to *sliding motion.* The time derivative $\boldsymbol{f}_{\mathrm{F}} : \mathcal{T} \times \Sigma \longrightarrow \mathbb{R}^n$ of the solution during the sliding motion along Σ is given as

$$\boldsymbol{f}_{\mathrm{F}}(t,x) = \boldsymbol{\alpha}(t,x)\boldsymbol{f}^-(t,x) + (1-\boldsymbol{\alpha}(t,x))\boldsymbol{f}^+(t,x). \tag{1.30}$$

Here, $\boldsymbol{\alpha} : \mathcal{T} \times \Sigma \longrightarrow [0,1]$ is defined such that for any $(t,x) \in \mathcal{T} \times \Sigma$ the vector $\boldsymbol{f}_{\mathrm{F}}(t,x)$ lies in the tangent plane of $\boldsymbol{\sigma}(x)$, i.e., $\boldsymbol{\alpha}(t,x)$ is chosen as the value which implies $\boldsymbol{n}^T(x)\boldsymbol{f}_{\mathrm{F}}(t,x) = 0$. A simple calculation therefore gives

$$\boldsymbol{\alpha}(t,x) = \frac{\boldsymbol{n}^T(x)\boldsymbol{f}^+(t,x)}{\boldsymbol{n}^T(x)(\boldsymbol{f}^+(t,x) - \boldsymbol{f}^-(t,x))}. \tag{1.31}$$

For any time instant $t \in \mathcal{T}$, the scalar $\boldsymbol{\alpha}(t,\boldsymbol{x}(t))$ can be regarded as the value α in (1.24) that chooses $\boldsymbol{f}_{\mathrm{F}}(t,\boldsymbol{x}(t))$ such that it lies along Σ and the trajectory $\boldsymbol{x}(t)$ slides along

Σ. Note that a solution having an attracting sliding mode exists and is unique. The attracting sliding mode is illustrated in the following example.

Example 1.22
Let us consider the differential inclusion

$$\dot{\boldsymbol{x}}(t) \in \boldsymbol{F}(\boldsymbol{x}(t)) = \begin{bmatrix} 3 + 1 \cdot \operatorname{Sgn}(\boldsymbol{x}_2(t) - c) \\ 1 - 3 \cdot \operatorname{Sgn}(\boldsymbol{x}_2(t) - c) \end{bmatrix}, \tag{1.32}$$

where $\boldsymbol{x}(t) = [\boldsymbol{x}_1(t), \boldsymbol{x}_2(t)]^T$. The only zero manifold is $\Sigma = \{x_2 = c\}$ and we can choose $\sigma(x) = c - x_2$ as one admissible switching function. Then the normal $\boldsymbol{n}(x)$ to Σ at any point $x \in \Sigma$ is given by $\boldsymbol{n}(x) = [0, -1]^T$. Some trajectories for different initial conditions are depicted in Figure 1.2. One can see that all trajectories starting from $\mathcal{S}^- = \{x_2 > c\}$ and from $\mathcal{S}^+ = \{x_2 < c\}$ are pushed to Σ by the vector field. At points where trajectories hit Σ they slide along Σ with time derivative determined as in (1.30). For our system we obtain

$$\boldsymbol{f}^- = \begin{bmatrix} 4 \\ -2 \end{bmatrix}, \quad \boldsymbol{f}^+ = \begin{bmatrix} 2 \\ 4 \end{bmatrix}, \quad \boldsymbol{\alpha} = \frac{2}{3}, \quad \boldsymbol{f}_{\mathrm{F}} = \begin{bmatrix} 10/3 \\ 0 \end{bmatrix}.$$

(ii) Repulsive Sliding Mode. Switches of this type indicate a *bifurcation* location. If a solution in repulsive sliding mode starts close to Σ then it will move away from it, but if it starts right on Σ then it can either stay while it obeys Filippov's solution, or it can leave Σ by entering either $\mathcal{S}^-$ or $\mathcal{S}^+$. A repulsive sliding occurs at $(t, x) \in \mathcal{T} \times \Sigma$ if

$$\boldsymbol{n}^T(x)\boldsymbol{f}^-(t, x) < 0 \quad \text{and} \quad \boldsymbol{n}^T(x)\boldsymbol{f}^+(t, x) > 0.$$

Since a solution may leave Σ at any instance of time with $\boldsymbol{f}^-(\cdot)$ or $\boldsymbol{f}^+(\cdot)$, it can not be unique. In this contribution, we do not consider models involving repulsive sliding mode. Nevertheless, we present it for the sake of completeness. The lack of uniqueness of solutions may vanish if one considers systems in the context of optimal control problems, cf. Section 1.5. Moreover, in the author's opinion, the methods developed later in this thesis (see Chapter 11) can also be applied to systems with repulsive sliding mode. The repulsive sliding mode is illustrated in the following example.

Example 1.23
Let us consider the differential inclusion

$$\dot{\boldsymbol{x}}(t) \in \boldsymbol{F}(\boldsymbol{x}(t)) = \begin{bmatrix} -3 - 1 \cdot \operatorname{Sgn}(\boldsymbol{x}_2(t) - c) \\ -1 + 3 \cdot \operatorname{Sgn}(\boldsymbol{x}_2(t) - c) \end{bmatrix}, \tag{1.33}$$

where $\boldsymbol{x}(t) = [\boldsymbol{x}_1(t), \boldsymbol{x}_2(t)]^T$. Note that the vector field of this system is the vector field of (1.32) in reverse time. With the same definitions of the zero manifold Σ, the switching function $\boldsymbol{\sigma}(\cdot)$, and the normal $\boldsymbol{n}(\cdot)$ to Σ as in Example 1.22 several admissible trajectories for system (1.33) are depicted in Figure 1.3. For the Filippov solution one has

$$\boldsymbol{f}^- = \begin{bmatrix} -4 \\ 2 \end{bmatrix}, \quad \boldsymbol{f}^+ = \begin{bmatrix} -2 \\ -4 \end{bmatrix}, \quad \boldsymbol{\alpha} = \frac{2}{3}, \quad \boldsymbol{f}_{\mathrm{F}} = \begin{bmatrix} -10/3 \\ 0 \end{bmatrix}.$$

Higher Order Conditions

In all cases that are not subsumed above, a more careful analysis is required. They are not covered by the first order theory presented. Another case one could imagine is the one where the solution is in attracting sliding mode, but for some time instant t_σ and the associated solution value $x_\sigma = \boldsymbol{x}(t_\sigma)$ condition (1.29) is no longer satisfied. According to Filippov's theory it can be assumed that one of the following cases has occurred:

$$\boldsymbol{n}^T(x_\sigma)\boldsymbol{f}^-(t_\sigma, x_\sigma) = 0 \quad \text{or} \quad \boldsymbol{n}^T(x_\sigma)\boldsymbol{f}^+(t_\sigma, x_\sigma) = 0. \tag{1.34}$$

This means that one of the two vector fields lies already in the tangent space of the zero manifold Σ. According to (1.31) the first case in (1.34) yields $\boldsymbol{\alpha}(t_\sigma, x_\sigma) = 1$ and therefore $\boldsymbol{f}_{\mathrm{F}} = \boldsymbol{f}^-$, and the latter case yields $\boldsymbol{\alpha}(t_\sigma, x_\sigma) = 0$ and $\boldsymbol{f}_{\mathrm{F}} = \boldsymbol{f}^+$, respectively. One would expect that a solution enters $\mathcal{S}^-$ in case of $\boldsymbol{\alpha} = 1$ and in the same way it enters $\mathcal{S}^+$ if $\boldsymbol{\alpha} = 0$. To decide this, one has to analyze higher order conditions. We touch on this topic briefly in the following and point out the central ideas.

An elementary geometric consideration will help us to get the central ideas of higher order conditions. Let $\boldsymbol{g}^-$, $\boldsymbol{g}^+$ and $\boldsymbol{g}$ be defined as

$$\boldsymbol{g}^-(t,x) \stackrel{\text{def}}{=} \nabla\boldsymbol{\sigma}(x)^T\boldsymbol{f}^-(t,x), \quad \boldsymbol{g}^+(t,x) \stackrel{\text{def}}{=} \nabla\boldsymbol{\sigma}(x)^T\boldsymbol{f}^+(t,x),$$

and

$$\boldsymbol{g}(t,x) = \boldsymbol{g}^-(t,x)\cdot\boldsymbol{g}^+(t,x).$$

Note that we assume $\boldsymbol{\sigma}(\cdot)$ to be sufficiently smooth. The sets of *transversality points* and *sliding points* are defined as

$$\Sigma_{\mathrm{T}} \stackrel{\text{def}}{=} \{(t,x) \in \mathcal{T}\times\Sigma : \boldsymbol{g}(t,x) > 0\} \quad \text{and} \quad \Sigma_{\mathrm{S}} \stackrel{\text{def}}{=} \{(t,x) \in \mathcal{T}\times\Sigma : \boldsymbol{g}(t,x) < 0\},$$

respectively. These definitions obviously coincide with the ones from (1.27) and (1.28). By definition, Σ_{T} and Σ_{S} are open and disjoint sets. We define so–called *exit sets* as

$$\mathcal{E}^- \stackrel{\text{def}}{=} \{(t,x) \in \mathcal{T}\times\Sigma : \boldsymbol{g}^-(t,x) = 0\}, \quad \mathcal{E}^+ \stackrel{\text{def}}{=} \{(t,x) \in \mathcal{T}\times\Sigma : \boldsymbol{g}^+(t,x) = 0\},$$

and

$$\mathcal{E} \stackrel{\text{def}}{=} \mathcal{E}^- \cup \mathcal{E}^+ = \{(t,x) \in \mathcal{T}\times\Sigma : \boldsymbol{g}(t,x) = 0\}$$

Note that these definitions coincide with the cases in (1.34). From a geometrical point of view, $x \in \Sigma$ lies on an $(n-1)$–dimensional manifold of points in Σ since we assumed $\nabla\boldsymbol{\sigma}(x) \neq 0$. If a point x belongs to $\mathcal{E}^-$ or $\mathcal{E}^+$, it will usually lie on an $(n-1)$–dimensional manifold of such points in Σ. This is, for example, the case under the mild assumptions that $\nabla\boldsymbol{\sigma}(x)$ is not parallel to $\nabla\boldsymbol{g}^-$ and $\nabla\boldsymbol{g}^+$, respectively. This means that one can expect that a smooth solution trajectory in Σ may encounter $\mathcal{E}^-$ or $\mathcal{E}^+$ but not $\mathcal{E}^- \cap \mathcal{E}^+$. However, a trajectory in $\mathcal{E}^-$ or $\mathcal{E}^+$ may encounter $\mathcal{E}^- \cap \mathcal{E}^+$.

Let us assume that we have a trajectory on Σ which is denoted by $\boldsymbol{x}(\cdot)$ and let $\boldsymbol{x}(\cdot)$ reach $\mathcal{E}$ at some time instant t. Without loss of generality we can assume $t = 0$. For this time we have to decide if the trajectory stays on Σ in sliding mode and we have to determine the associated vector field, or if the trajectory leaves Σ. Also in the latter case, the vector field has to be determined.
No matter if the trajectory stays on Σ or leaves it, the trajectory starts with the value $\boldsymbol{x}(0)$. The basic idea how to continue the trajectory is the following: one looks at the left and right limit expansions of the functions $\boldsymbol{g}^{-}(t, \boldsymbol{x}(t))$ and $\boldsymbol{g}^{+}(t, \boldsymbol{x}(t))$ and enforces smoothness of the solution. Details on how to do this and the resulting multiple cases can be found in the article of Dieci and Lopez [130].

1.5 OCPs with Explicit and Implicit Switches

In this section, we embed the ODEs with explicit and implicit switches from Definition 1.9 into an OCP context. This means that we consider a system that is affected by nonlinear control functions and for which we have to determine a control law such that a certain optimality criterion is achieved. Apart from a switched ODE, the system may include additional constraints for states and controls.

1.5.1 Problem Formulation

We start the section with a formal problem definition of a rather general OCP involving a dynamic system with explicit and implicit switches.

Definition 1.24 (Optimal Control Problem with Explicit and Implicit Switches)
An *OCP with explicit and implicit switches* is a constrained infinite–dimensional optimization problem of the form

$$\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),\boldsymbol{v}(\cdot)} \quad \boldsymbol{\varphi}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})) \tag{1.35a}$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t), \operatorname{sgn}(\boldsymbol{\sigma}(\boldsymbol{x}(t)))), \quad t \in \mathcal{T}, \tag{1.35b}$$

$$\mathbf{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t)), \quad t \in \mathcal{T}, \tag{1.35c}$$

$$\mathbf{0}_{n_\mathrm{d}} \geqslant \boldsymbol{d}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), \quad t \in \mathcal{T}, \tag{1.35d}$$

$$\mathbf{0}_{n_\mathrm{r}} = \boldsymbol{r}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})), \tag{1.35e}$$

$$\boldsymbol{v}(t) \in \Omega \subset \mathbb{R}^{n_\mathrm{v}}, \quad t \in \mathcal{T},\ |\Omega| = n_\omega < \infty, \tag{1.35f}$$

where a dynamic process $\boldsymbol{x} : \mathcal{T} \longrightarrow \mathbb{R}^{n_\mathrm{x}}$ on the time horizon $\mathcal{T} \overset{\text{def}}{=} [t_\mathrm{s}, t_\mathrm{f}] \subset \mathbb{R}$ is determined. A solution $\boldsymbol{x}(\cdot)$ is described by a system of ODEs, where $\boldsymbol{f} : \mathcal{T} \times \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{u}} \times \mathbb{R}^{n_\mathrm{v}} \times \{0, \pm 1\}^{n_\sigma} \longrightarrow \mathbb{R}^{n_\mathrm{x}}$ acts as the right–hand–side function. This system is affected by a continuous and vector–valued control function $\boldsymbol{u} : \mathcal{T} \longrightarrow \mathbb{R}^{n_\mathrm{u}}$ as well as another vector–valued control function $\boldsymbol{v} : \mathcal{T} \longrightarrow \Omega$, which attains only values from a finite discrete set $\Omega \overset{\text{def}}{=} \{v^1, v^2, \ldots, v^{n_\omega}\} \subseteq \mathbb{R}^{n_\mathrm{v}}$ with cardinality $|\Omega| = n_\omega < \infty$. Moreover, the system is affected by an implicit switch given by the sign structure of a switching function $\boldsymbol{\sigma} : \mathbb{R}^{n_\mathrm{x}} \longrightarrow \mathbb{R}^{n_\sigma}$. The controls are to be determined such that a performance index $\boldsymbol{\varphi} : \mathcal{T} \times \mathbb{R}^{n_\mathrm{x}} \times \mathcal{T} \times \mathbb{R}^{n_\mathrm{x}} \longrightarrow \mathbb{R}$ is minimized. Additionally, mode–dependent and mode–independent mixed control–state constraints $\boldsymbol{c} : \mathcal{T} \times \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{u}} \times \mathbb{R}^{n_\mathrm{v}} \longrightarrow \mathbb{R}^{n_\mathrm{c}}$ and $\boldsymbol{d} : \mathcal{T} \times \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{u}} \longrightarrow \mathbb{R}^{n_\mathrm{d}}$ as well as boundary constraints $\boldsymbol{r} : \mathcal{T} \times \mathbb{R}^{n_\mathrm{x}} \times \mathcal{T} \times \mathbb{R}^{n_\mathrm{x}} \longrightarrow \mathbb{R}^{n_\mathrm{r}}$ must be satisfied. △

Note that the function $\boldsymbol{u}(\cdot)$ enters OCP (1.35) as a real and vector–valued control function. Contrary to Definition 1.9, the function $\boldsymbol{v}(\cdot)$, whose values must be chosen from a finite discrete set, is up to optimization now and therefore a control function as well. To emphasize the special character of $\boldsymbol{v}(\cdot)$ compared to $\boldsymbol{u}(\cdot)$ and due to its special status in this thesis, we introduce a new term for this type of control functions.

Definition 1.25 (Integer Control and Binary Control)
The term *integer control* for function $\boldsymbol{v}(\cdot)$ in Definition 1.24 is used for control functions whose image space is a finite discrete set, i.e.,

$$\boldsymbol{v}(t) \in \Omega \stackrel{\text{def}}{=} \{v^1, v^2, \ldots, v^{n_\omega}\},$$

with $\exists \varepsilon > 0, \forall i \neq j : \left\| v^i - v^j \right\| > \varepsilon$. The term *binary control* is used for the special case

$$\boldsymbol{v}(t) \in \{0,1\}^{n_\omega}. \qquad \triangle$$

We use the term *relaxed*, whenever the condition $\boldsymbol{v}(\cdot) \in \Omega$ is relaxed to a superset of Ω. This holds in particular for the case that Ω is replaced with its convex hull $\operatorname{conv}(\Omega)$. For instance, relaxing the condition $\boldsymbol{v}(t) \in \{0,1\}^{n_\omega}$ with the convex hull of $\{0,1\}^{n_\omega}$ would result in the relaxed condition $\boldsymbol{v}(t) \in \operatorname{conv}(\{0,1\}^{n_\omega}) = [0,1]^{n_\omega}$.
One major goal of this thesis is to solve the explicitly and implicitly switched OCP from Definition 1.24. In Chapter 11, we present a new method to solve the problem. This new method is based on an approach which was developed to solve pure explicitly switched OCPs. In the following, we sketch this approach. For this reason, we replace the ODE in Equation (1.35a) with the explicitly switched ODE

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t)). \tag{1.36}$$

Definition 1.26 (Optimal Control Problem with Explicit Switches)
An *OCP with explicit switches* is defined as

$$\begin{aligned}
\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),\boldsymbol{v}(\cdot)} \quad & \boldsymbol{\varphi}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})) && && \text{(1.37a)}\\
\text{s.t.} \quad & \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t)), && t \in \mathcal{T}, && \text{(1.37b)}\\
& \boldsymbol{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t)), && t \in \mathcal{T}, && \text{(1.37c)}\\
& \boldsymbol{0}_{n_\mathrm{d}} \geqslant \boldsymbol{d}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), && t \in \mathcal{T}, && \text{(1.37d)}\\
& \boldsymbol{0}_{n_\mathrm{r}} = \boldsymbol{r}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})), && && \text{(1.37e)}\\
& \boldsymbol{v}(t) \in \Omega \subset \mathbb{R}^{n_\mathrm{v}}, && t \in \mathcal{T},\ |\Omega| = n_\omega < \infty, && \text{(1.37f)}
\end{aligned}$$

where the meaning of all variables and functions is the same as in Definition 1.24. $\triangle$

The discrete character of the explicitly switched OCP (1.37) introduces major difficulties in solving the problem. For this reason, we pursue the plan to replace OCP (1.37) with an equivalent auxiliary problem which allows for a subsequent relaxation step. Thereby, one can overcome the discrete character of OCP (1.37) and come up with a problem that might become easier to solve. As we will see, the solution of the relaxed problem can be used to find good approximate solutions for the problem that we actually aim to solve.

All approaches are based on the same considerations: it would be desirable to find reformulations which enable us to express the logical mode choices by means of logical variables that can be relaxed. The resulting problem is then a so–called *relaxed OCP*.
A relaxed OCP is a purely continuous control problem whose fundamentals are reviewed in Chapter 5. A lot of research has been done on continuous OCPs and powerful numerical solution methods exist to solve this problem type in an efficient way. Chapter 6 presents several well–established methods, whereas Chapter 7 introduces a tailored algorithm for switched OCPs.
The idea of convexification and subsequent relaxation is similar to the notion of *generalized curves* which were introduced by Young [461] to investigate existence questions in the *calculus of variations*. Cesari and Berkovitz used *relaxed controls*, which are also similar in notion, to study existence results for OCPs, cf. Cesari [105], Berkovitz [57], or Berkovitz and Medhin [58].

1.5.2 Inner and Outer Convexification in OCPs with Explicit Switches

For the moment, we deal only with mixed control–state constraints (1.37d) which do not explicitly depend on $\boldsymbol{v}(\cdot)$, but not with mode–dependent constraints (1.37c). Convexification techniques in this setting were studied by Sager in his doctoral thesis, cf. Sager [380], Sager et al. [384]. The task of dealing with switches in path constraints is postponed to the subsequent Section 1.5.3.
There are several options how the logical mode choice defining function $\boldsymbol{v}(\cdot)$ in (1.37b) can be reformulated. Depending on how $\boldsymbol{v}(\cdot)$ is substituted we distinguish between the *Inner Convexification (IC)*, the *(Partial) Outer Convexification (OC)* and the *Perspective Formulation* approaches. In this contribution, we present IC and OC. The perspective formulation was carefully analyzed in the publications of Jung et al. [260] and Jung [259].

Inner Convexification

For the *IC approach* the function $\boldsymbol{v}(\cdot)$, which expresses the time–dependent mode switches, is replaced by means of function $\boldsymbol{w} : \mathcal{T} \longrightarrow [n_\omega]$ as well as function $\boldsymbol{g} : [1, n_\omega] \longrightarrow \mathbb{R}^{n_\mathrm{v}}$, where $\boldsymbol{g}(\cdot)$ has the property $\boldsymbol{g}(i) = v^i$ for $i \in [n_\omega]$. OCP (1.37) with explicit switches and without mode–specific constraints $\boldsymbol{c}(\cdot)$ after IC reformulation reads as

$$\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),\boldsymbol{w}(\cdot)} \quad \varphi(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})) \tag{1.38a}$$

$$\begin{aligned} \text{s.t.} \quad & \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{g}(\boldsymbol{w}(t))), && t \in \mathcal{T}, \\ & \mathbf{0}_{n_\mathrm{d}} \geqslant \boldsymbol{d}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), && t \in \mathcal{T}, \\ & \mathbf{0}_{n_\mathrm{r}} = \boldsymbol{r}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})), \\ & \boldsymbol{w}(t) \in [n_\omega], && t \in \mathcal{T}. \end{aligned} \tag{1.38b}$$

By construction, Problem (1.38) can be relaxed towards $\boldsymbol{w}(t) \in [1, n_\omega]$. There are several possibilities to choose the function $\boldsymbol{g}(\cdot)$. Two common realizations are

(i) piecewise linear representations of the form

$$\boldsymbol{g}(i+\xi_{i+1}) = \xi_i v^i + \xi_{i+1} v^{i+1}$$

with *Special Ordered Sets of type 2 (SOS–2)* variables:

$$\xi_i \in [0,1],\ \sum_i \xi_i = 1, \xi_i \neq 0 \Longrightarrow \xi_j = 0\ \forall j \neq i, i+1.$$

(ii) a convex combination

$$\boldsymbol{g}\left(\sum_{i=1}^{n_\omega} \xi_i i\right) = \sum_{i=1}^{n_\omega} \xi_i v^i$$

with *Special Ordered Sets of type 1 (SOS–1)* variables $\xi_i \in [0,1],\ \sum_i \xi_i = 1$.

A third realization uses fitted smooth convex functions $\boldsymbol{g}(\cdot)$, as suggested by Gerdts [187].

(Partial) Outer Convexification

The *OC approach* has been investigated in the context of OCPs by Sager [380], Sager et al. [381], and Sager et al. [384]. The term partial is due to the exclusive convexification of the integer controls, but not to the rest of the control problem. To realize the OC approach the integer controls $\boldsymbol{v}(\cdot)$ are lifted into a higher dimensional space by introducing binary controls $\boldsymbol{\omega}_i : \mathcal{T} \longrightarrow \{0,1\},\ i \in [n_\omega]$. The value $\boldsymbol{\omega}_i(t)$ indicates if mode i is active ($\boldsymbol{\omega}_i(t) = 1$) or not ($\boldsymbol{\omega}_i(t) = 0$) at time instant $t \in \mathcal{T}$. We have used the same idea in (1.6) to reformulate IVP (1.3). The OC approach consists of an evaluation of all possible right–hand–sides, their multiplication with convex multipliers, and the summation of the products. ODE (1.36) then reads as

$$\dot{\boldsymbol{x}}(t) = \sum_{i=1}^{n_\omega} \boldsymbol{\omega}_i(t) \cdot \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), v^i).$$

In IVP (1.3) the mode sequence was given in advance, whereas it is up to optimization in an OCP context. To ensure that exactly one mode is active at any time instant $t \in \mathcal{T}$ one additionally imposes the SOS–1 constraint

$$\sum_{i=1}^{n_\omega} \boldsymbol{\omega}_i(t) = 1.$$

OCP (1.37) with explicit switches and without mode–specific constraints $\boldsymbol{c}(\cdot)$ after OC reformulation reads as

$$\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),\boldsymbol{\omega}(\cdot)} \quad \boldsymbol{\varphi}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f}))$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}}(t) = \sum_{i=1}^{n_\omega} \boldsymbol{\omega}_i(t) \cdot \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), v^i), \quad t \in \mathcal{T}, \tag{1.39a}$$

$$\mathbf{0}_{n_\mathrm{d}} \geqslant \boldsymbol{d}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), \qquad t \in \mathcal{T},$$

$$\mathbf{0}_{n_\mathrm{r}} = \boldsymbol{r}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})),$$

$$1 = \sum_{i=1}^{n_\omega} \boldsymbol{\omega}_i(t), \ \boldsymbol{\omega}(t) \in \{0,1\}^{n_\omega}, \qquad t \in \mathcal{T}, \tag{1.39b}$$

where we define as usual $\boldsymbol{\omega}(\cdot) \stackrel{\text{def}}{=} [\boldsymbol{\omega}_1(\cdot), \ldots, \boldsymbol{\omega}_{n_\omega}(\cdot)]^T$. By construction, Problem (1.39) can be relaxed towards $\boldsymbol{\omega}(t) \in [0,1]^{n_\omega}$. In later chapters we use the notation $\boldsymbol{\alpha}(t) \in [0,1]^{n_\omega}$ to highlight the difference between the original and the relaxed problem. For the sake of completeness, we state the relaxed counterpart problem of OCP (1.39) in the following:

$$\min_{\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot), \boldsymbol{\alpha}(\cdot)} \quad \boldsymbol{\varphi}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})) \tag{1.40a}$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}}(t) = \sum_{i=1}^{n_\omega} \boldsymbol{\alpha}_i(t) \cdot \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), v^i), \quad t \in \mathcal{T},$$

$$\mathbf{0}_{n_\mathrm{d}} \geqslant \boldsymbol{d}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), \qquad t \in \mathcal{T},$$

$$\mathbf{0}_{n_\mathrm{r}} = \boldsymbol{r}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})),$$

$$1 = \sum_{i=1}^{n_\omega} \boldsymbol{\alpha}_i(t), \ \boldsymbol{\alpha}(t) \in [0,1]^{n_\omega}, \qquad t \in \mathcal{T}.$$

Analogously to $\boldsymbol{\omega}(\cdot)$, the components of $\boldsymbol{\alpha}(\cdot)$ are denoted by $\boldsymbol{\alpha}_i(\cdot)$, $i \in [n_\omega]$. The following remark illustrates a common way to reduce the problem dimensions of OCP (1.39) and OCP (1.40).

Remark 1.27 (Elimination Using the SOS–1 Constraint)
The SOS–1 constraint in (1.39b) allows for an elimination of one binary control function $\boldsymbol{\omega}_j(\cdot)$, $j \in [n_\omega]$, which is then replaced by

$$\boldsymbol{\omega}_j(t) = 1 - \sum_{\substack{i=1 \\ i \neq j}}^{n_\omega} \boldsymbol{\omega}_i(t), \quad t \in \mathcal{T}.$$

The same holds for its relaxed counterparts $\boldsymbol{\alpha}_j(\cdot)$.

Inner versus Outer Convexification

In Problem (1.38), only a function $\boldsymbol{w} : \mathcal{T} \longrightarrow [1, n_\omega]$ enters the control problem after relaxation, whereas n_ω functions $\boldsymbol{\alpha}_i : \mathcal{T} \longrightarrow [0,1]$ enter the relaxed Problem (1.40). For this reason, relaxations of reformulation (1.38) may be faster to solve compared to (1.40): after a discretization step, which transcribes the OCP to a Nonlinear Programming Problem, the number of derivatives to be computed is smaller, and subproblems to be solved in iterations of interior–point or SQP methods are cheaper to solve. Furthermore, the aggregated right–hand–side

$\sum_i \boldsymbol{f}(\cdot, v^i)$ in Problem (1.40) may become more expensive to evaluate. The number of explicit switching modes is reflected by n_ω and may get large.
The number of admissible choices n_ω can be reduced in many cases. For instance, certain separability properties of the right–hand–side function $\boldsymbol{f}(\cdot)$ often decouple integer controls, cf. Gräber et al. [209].

Example 1.28 (Outer Convexification and Separability)
Let an ODE be defined as

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}_1(\cdot, \boldsymbol{v}_1(t)) + \boldsymbol{f}_2(\cdot, \boldsymbol{v}_2(t)), \quad \boldsymbol{v}_1(t) \in \Omega_1, \quad \boldsymbol{v}_2(t) \in \Omega_2.$$

Instead of $n_\omega = n_{\omega 1} \cdot n_{\omega 2}$ controls by enumerating all possible modes in $\Omega_1 \times \Omega_2$, which would result in the problem

$$\begin{aligned}
\dot{\boldsymbol{x}}(t) &= \sum_{i=1}^{n_{\omega 1}} \sum_{j=1}^{n_{\omega 2}} \boldsymbol{\omega}_{i,j}(t) \cdot \left(\boldsymbol{f}_1(\cdot, v_1^i) + \boldsymbol{f}_2(\cdot, v_2^j) \right) \\
1 &= \sum_{i=1}^{n_{\omega 1}} \sum_{j=1}^{n_{\omega 2}} \boldsymbol{\omega}_{i,j}(t), \ \boldsymbol{\omega}_{i,j}(t) \in \{0,1\},
\end{aligned}$$

for all $t \in \mathcal{T}$, we obtain an equivalent OC reformulation, which is given by

$$\begin{aligned}
\dot{\boldsymbol{x}}(t) &= \sum_{i=1}^{n_{\omega 1}} \boldsymbol{\omega}_{1,i}(t) \cdot \boldsymbol{f}_1(\cdot, v_1^i) + \sum_{i=1}^{n_{\omega 2}} \boldsymbol{\omega}_{2,i}(t) \cdot \boldsymbol{f}_2(\cdot, v_2^i) \\
1 &= \sum_{i=1}^{n_{\omega 1}} \boldsymbol{\omega}_{1,i}(t), \ \boldsymbol{\omega}_1(t) \in \{0,1\}^{n_{\omega 1}}, \quad 1 = \sum_{i=1}^{n_{\omega 2}} \boldsymbol{\omega}_{2,i}(t), \ \boldsymbol{\omega}_2(t) \in \{0,1\}^{n_{\omega 2}}
\end{aligned}$$

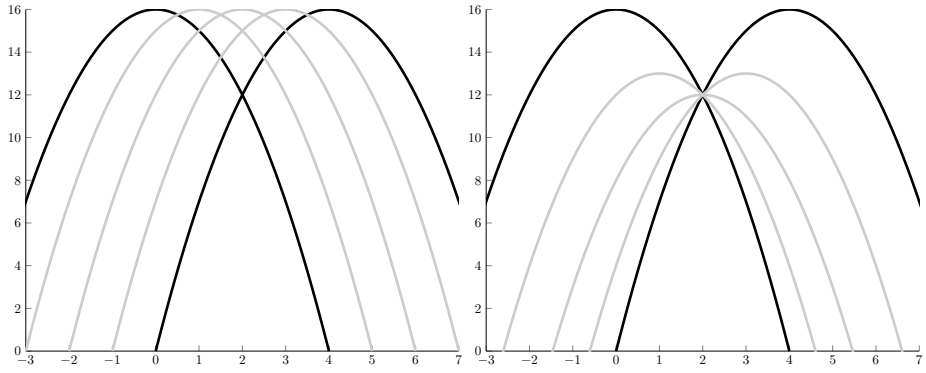
for $t \in \mathcal{T}$, and combines modes from Ω_1 and Ω_2 independently and requires only $n_\omega = n_{\omega 1} + n_{\omega 2}$ controls.

In most real–world applications, n_ω increases linearly with the number of choices, or the integer controls decouple, or the binary control functions enter the problem linearly. For this reason, a modest increase in the number of control functions can be expected, and this increase provides crucial advantages of OC reformulation (1.39) over IC reformulation (1.38):

- It is often impossible to find meaningful IC functions $\boldsymbol{g}(\cdot)$. Black–box simulators can often be evaluated only for values from the domain Ω, but not for values in between. IC reformulations may furthermore lead to numerical issues such as divisions by zero. Since an OC reformulation evaluates the model only for vectors from the admissible set Ω, the aforementioned problems can be overcome.
- The integer gap between optimal solutions of OCP (1.38) and its relaxation may become arbitrary large, cf. Sager [380], whereas the corresponding integer gap for Problem (1.39) is bounded by a multiple of the control discretization grid size, cf. Sager et al. [384].

In order to solve the original problem, it is very important to find a tight relaxation of OCP (1.39). The explicitly switched OCP can be decoupled into a continuous OCP and a mixed–integer linear programming problem, which allows for computational savings and a posteriori

bounds on the gap to the best possible solution, cf. Sager et al. [384]. Compared to a branch & bound approach to solve OCP (1.38), solutions of the relaxed Problem (1.39) are often integral. Based on a benchmark problem posed by Gerdts [187], the two approaches have been compared by Kirches et al. [273] for the first time. Both approaches identify identical solutions, but the OC approach obtains the solution with a speedup of several orders of magnitude compared to the IC approach.



a) Inner Convexification of $\boldsymbol{f}(x,v)$. b) (Partial) Outer Convexification of $\boldsymbol{f}(x,v)$.

Figure 1.4: IC and OC approach applied to the example $\boldsymbol{f}(x,v) = -(x-v)^2 + 16$. (—) shows the function's graph $\operatorname{gr}\boldsymbol{f}(x;v^i)$, $i = 1,2$ for the integer–valued parameter choices $v^1 = 0$ and $v^2 = 4$. For the IC approach (—) shows the function's graph $\operatorname{gr}\boldsymbol{f}(x;\xi v^1 + (1-\xi)v^2)$ for the choices $\xi \in \left\{\frac{1}{4}, \frac{1}{2}, \frac{3}{4}\right\}$. Analogously, for the OC approach (—) shows the function's graph $\operatorname{gr}\alpha\boldsymbol{f}(x;v^1) + (1-\alpha)\boldsymbol{f}(x;v^2)$ for the choices $\alpha \in \left\{\frac{1}{4}, \frac{1}{2}, \frac{3}{4}\right\}$.

Example 1.29 (Inner and Outer Convexification)
To visualize the differences between the IC and the OC approach for nonlinear functions, we exemplarily investigate the function $\boldsymbol{f}(x,v) \stackrel{\text{def}}{=} -(x-v)^2 + 16$ with the continuous variable $x \in \mathbb{R}$ and the integer variable $v \in \mathbb{Z}$. The set of admissible values for v is chosen as $\Omega = \{v^1, v^2\}$ with $v^1 = 0$ and $v^2 = 4$. Figure 1.4 depicts $\operatorname{gr}\boldsymbol{f}(x;v)$ for $v \in \Omega$ as well as function graphs of IC and OC reformulations of $\boldsymbol{f}(\cdot)$ with relaxed choices of v. In case of using the convex combination approach, a subsequent relaxation step and the elimination of one convex multiplier (see Remark 1.27) IC of $\boldsymbol{f}(x,v)$ with respect to the integer variable v yields

$$\begin{aligned} \boldsymbol{f}^{\mathrm{IC}}(x,\xi) &\stackrel{\text{def}}{=} \boldsymbol{f}(x;\xi v^1 + (1-\xi)v^2), \\ &\xi \in [0,1]. \end{aligned}$$

OC including a relaxation step and elimination of one control function (see Remark 1.27) results in

$$\begin{aligned} \boldsymbol{f}^{\mathrm{OC}}(x,\alpha) &\stackrel{\text{def}}{=} \alpha\boldsymbol{f}(x;v^1) + (1-\alpha)\boldsymbol{f}(x;v^2), \\ &\alpha \in [0,1]. \end{aligned}$$

1.5.3 Constraint Formulations

In the previous Section 1.5.2, we excluded the occurrence of logically implied constraints

$$\mathbf{0}_{n_c} \geqslant \boldsymbol{c}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t)) \tag{1.41}$$

in the explicitly switched OCP (1.37). In this section, we seek for reformulations of this constraint type that allow for an additional relaxation step. This setting was discussed by Kirches [272] in his dissertation for the first time. Jung [259], in turn, compared different convexification strategies and interrelated them in his dissertation.
Apart from the aforementioned *IC* and *OC* reformulations we describe the so–called *Vanishing Constraint (VC)* approach. For all of them, we investigate their tightness properties and consider their benefits and drawbacks. Descriptions of further reformulations such as *Big–M* and *Perspective Formulation* can be found in the publications of Jung et al. [260] and Jung [259].
To visualize the different formulations and to demonstrate their effects, we use the following example disjunction of two constraints.

Example 1.30
We consider a disjunction of two quadratic constraints that are given as follows:

$$\left[\boldsymbol{c}(x, v^1) = \boldsymbol{a}_1(v^1)x^2 + \boldsymbol{a}_2(v^1)x + \boldsymbol{a}_3(v^1) \geqslant 0\right] \tag{1.42a}$$
$$\vee \left[\boldsymbol{c}(x, v^2) = \boldsymbol{a}_1(v^2)x^2 + \boldsymbol{a}_2(v^2)x + \boldsymbol{a}_3(v^2) \geqslant 0\right], \tag{1.42b}$$
$$x \in [-1, 4]. \tag{1.42c}$$

The functions $\boldsymbol{a}_i(\cdot)$, $i = 1, 2, 3$ are defined as

$$\boldsymbol{a}_1(v) \overset{\text{def}}{=} -(v-4)^2, \qquad \boldsymbol{a}_2(v) \overset{\text{def}}{=} 3(v-4), \qquad \boldsymbol{a}_3(v) \overset{\text{def}}{=} 4.$$

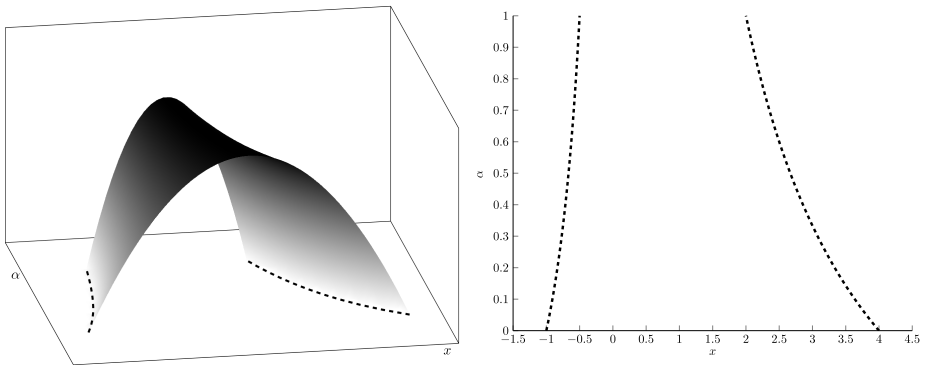
Note that the constraints are convex since $\boldsymbol{a}_1(v) \leqslant 0$ holds for arbitrary chosen v. The admissible set for the integral variable v is chosen to be $\Omega \overset{\text{def}}{=} \{v^1, v^2\}$ with $v^1 = 5$ and $v^2 = 6$.

Inner Convexification

The IC reformulation approach for constraints of type (1.41) works in a similar way as for ODE (1.36). Again, occurrences of the discrete control $\boldsymbol{v}(\cdot)$ are replaced by a convex function $\boldsymbol{g}(\boldsymbol{w}(\cdot))$ of the newly introduced integer control $\boldsymbol{w}(\cdot)$, which attains the values $v \in \Omega$ for integral choices of $\boldsymbol{w}(\cdot)$. If IC is applied to OCPs, one usually chooses the same $\boldsymbol{g}(\cdot)$ for constraints and ODE. Summarizing, the IC reformulation of constraint (1.41) reads as

$$\mathbf{0}_{n_c} \geqslant \boldsymbol{c}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{g}(\boldsymbol{w}(t))).$$

Note that for $\boldsymbol{w}(t) \in \{1, 2, \ldots, n_\omega\}$ there exists $1 \leqslant i \leqslant n_\omega$ such that $\boldsymbol{g}(\boldsymbol{w}(t)) = v^i$.
Similar to the arguments from Section 1.5.2, we conclude that evaluating convex combinations within nonlinear functions may result in optimality of feasible fractional values, whereas neighboring integer values may not be optimal. Hence, relaxations obtained from an IC approach can be expected to be weak.



a) Inner Convexification of $\boldsymbol{c}(x, v)$.

b) Feasible set of IC of $\boldsymbol{c}(x, v)$.

Figure 1.5: IC applied to the two sided disjunction (1.42a)–(1.42c) from Example 1.30. The left figure shows the function's graph $\operatorname{gr} \boldsymbol{c}^{\mathrm{IC}}(x, \xi)$ for $\xi \in [0, 1]$, where $\boldsymbol{c}^{\mathrm{IC}}(x, \xi) = \boldsymbol{c}(x, \xi v^1 + (1-\xi)v^2)$. The right figure depicts the feasible set $\left\{(x, \xi) : \boldsymbol{c}^{\mathrm{IC}}(x, \xi) \geqslant 0\right\}$.

In case of using the convex combination approach, a subsequent relaxation step and the elimination of one convex multiplier (see Remark 1.27), the IC of $\boldsymbol{c}(x, v)$ from Example 1.30 with respect to the integer variable v yields

$$\begin{aligned} &0 \leqslant \boldsymbol{c}^{\mathrm{IC}}(x, \xi) \stackrel{\text{def}}{=} \boldsymbol{c}(x; \xi v^1 + (1-\xi)v^2), \\ &\xi \in [0, 1]. \end{aligned}$$

Figure 1.5 depicts the function graph $\operatorname{gr} \boldsymbol{c}^{\mathrm{IC}}(x, \xi)$ and the feasible set $\left\{(x, \xi) : \boldsymbol{c}^{\mathrm{IC}}(x, \xi) \geqslant 0\right\}$. By means of the feasible set, one can see that the convexity is not retained by the chosen IC reformulation.

(Partial) Outer Convexification

We can apply the OC reformulation from (1.39a)+(1.39b) to the constraint $\boldsymbol{c}(\cdot)$ as well. A convex combination of residuals $\boldsymbol{c}(\cdot, v^i)$ for all admissible modes $v^i \in \Omega$ is imposed resulting in the formulation

$$\mathbf{0}_{n_c} \geqslant \sum_{i=1}^{n_\omega} \boldsymbol{\omega}_i(t) \cdot \boldsymbol{c}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), v^i) \tag{1.43}$$

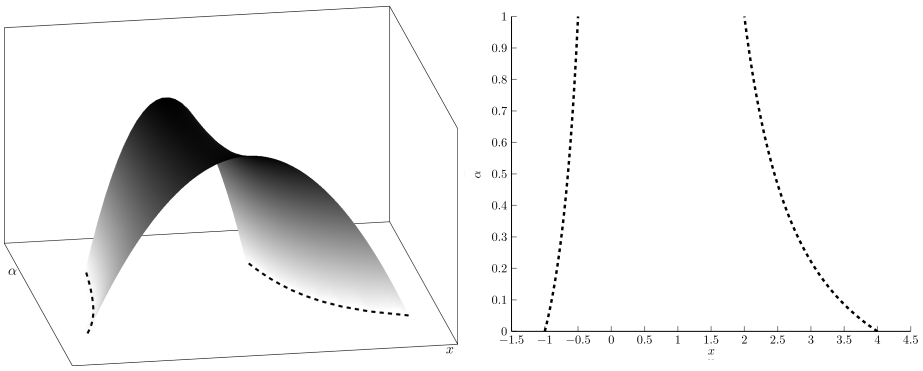
$$1 = \sum_{i=1}^{n_\omega} \boldsymbol{\omega}_i(t), \ \boldsymbol{\omega}_i(t) \in \{0, 1\}. \tag{1.44}$$

The OC reformulation does not require to evaluate the constraint $\boldsymbol{c}(\cdot, v)$ for fractional choices v, and all feasible integer points are guaranteed to be feasible for the original formulation. Often, constraints may be just active for a subset of available modes. This has to be taken into account and one has to ensure that the respective constraints are inactive while these modes

are not active. In general, just constraints of the same type should be combined in the convex combination (1.43).
The relaxed formulation replaces $\boldsymbol{\omega}(t) \in \{0,1\}^{n_\omega}$ with $\boldsymbol{\omega}(t) \in [0,1]^{n_\omega}$ in (1.44). As it was demonstrated in Example 1.28, one should exploit separability properties of the constraints for the purpose of a reduced number of binary control variables.
Kirches [272] has shown that aggregating the constraint residuals $\boldsymbol{c}(\cdot, v^i)$ into a single constraint as this is done in the OC reformulation approach give rise to *compensatory effects* and may lead to feasible residuals for fractional values of the complex multipliers.
The following result states that the feasible set of the relaxed OC reformulation (1.43) and (1.44) projected onto the $(\boldsymbol{x}, \boldsymbol{u})$–space coincides with the union of all feasible sets of the disjunction $\bigvee_{i=1}^{n_\omega} [\mathbf{0}_{n_c} \geqslant \boldsymbol{c}(\cdot, v^i)]$.

Proposition 1.31 (Feasible Region)
There exists $\boldsymbol{\omega}(t)$ such that $(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\omega}(t))$ is feasible for the relaxed version of the OC formulation (1.43)+(1.44) if and only if there exists an index $1 \leqslant i \leqslant n_\omega$ such that $\mathbf{0}_{n_c} \geqslant \boldsymbol{c}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), v^i)$. △

Proof See Jung [259]. □



a) (Partial) Outer Convexification of $\boldsymbol{c}(x, v)$.

b) Feasible set of OC of $\boldsymbol{c}(x, v)$.

Figure 1.6: OC applied to the two sided disjunction (1.42a)–(1.42c) from Example 1.30. The left figure shows the function's graph $\operatorname{gr} \boldsymbol{c}^{\mathrm{OC}}(x, \alpha)$ for $\alpha \in [0,1]$, where $\boldsymbol{c}^{\mathrm{OC}}(x, \alpha) = \alpha \cdot \boldsymbol{c}(x, v^1) + (1-\alpha) \cdot \boldsymbol{c}(x, v^2)$. The right figure depicts the feasible set $\left\{(x, \alpha) \,:\, \boldsymbol{c}^{\mathrm{OC}}(x, \alpha) \geqslant 0\right\}$.

OC applied to $\boldsymbol{c}(x, v)$ from Example 1.30 with respect to the integer variable v, a subsequent relaxation step, and the elimination of one convex multiplier (see Remark 1.27) yields

$$\begin{aligned} &0 \leqslant \boldsymbol{c}^{\mathrm{OC}}(x, \alpha) \stackrel{\text{def}}{=} \alpha \cdot \boldsymbol{c}(x, v^1) + (1-\alpha) \cdot \boldsymbol{c}(x, v^2), \\ &\alpha \in [0,1]. \end{aligned}$$

Figure 1.6 depicts the function graph $\operatorname{gr} \boldsymbol{c}^{\mathrm{OC}}(x, \alpha)$ and the feasible set $\left\{(x, \alpha) \,:\, \boldsymbol{c}^{\mathrm{OC}}(x, \alpha) \geqslant 0\right\}$. As in case of the IC approach, the convexity is not retained by the OC reformulation. Furthermore, Proposition 1.31 guarantees that the feasible set after OC is not connected in the (x, α)–space if the two feasible sets of the disjunction are disjoint in the x–space. The question if the

feasible set is connected or not is crucial when one wants to solve the problem numerically. Local methods are not suited for problems with disconnected feasible sets. Note, however, that the feasible set remains convex for a fixed α, which usually leads to a good convergence of local methods.

Vanishing and Complementarity Constraints

As opposed to the IC and OC approaches, reformulations with vanishing and complementarity constraints avoid aggregating the constraints of the different modes. Instead, one constraint set per mode is added and the constraint (1.41) is replaced with

$$\begin{aligned} \mathbf{0}_{n_c} &\geqslant \boldsymbol{\omega}_i(t) \cdot \boldsymbol{c}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), v^i), \quad 1 \leqslant i \leqslant n_\omega, \\ 1 &= \sum_{i=1}^{n_\omega} \boldsymbol{\omega}_i(t), \ \boldsymbol{\omega}_i(t) \in \{0,1\}. \end{aligned} \tag{1.45}$$

It can easily be seen that the constraint of each mode is enforced if the corresponding multiplier is nonzero.

Remark 1.32 (Tightness of Complementarity Constraint Approach)
Any value $\boldsymbol{\omega}_i(t) > 0$ in (1.45) makes the corresponding vanishing constraint

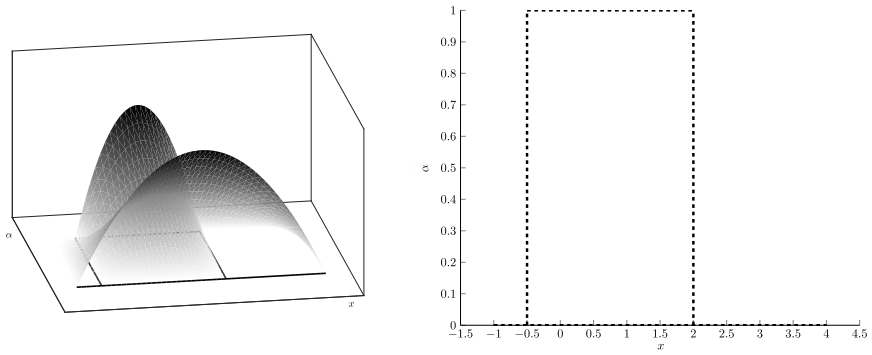
$$\mathbf{0} \geqslant \boldsymbol{c}(\cdot, v^i)$$

being satisfied.

Due to Remark 1.32, vanishing constraints are either fully active (if $\boldsymbol{\omega}_i(t) > 0$) or inactive (if $\boldsymbol{\omega}_i(t) = 0$), but nothing in between. Hence, relaxations $\boldsymbol{\omega}(t) \in [0,1]^{n_\omega}$ are much tighter than the respective ones for OC or IC reformulations.
OC applied to $\boldsymbol{c}(x, v)$ from Example 1.30 with respect to the integer variable v, a subsequent relaxation step, and the elimination of one convex multiplier (see Remark 1.27) yields

$$\begin{aligned} 0 &\leqslant \boldsymbol{c}^{\mathrm{VC}_1}(x, \alpha) \stackrel{\text{def}}{=} \alpha \cdot \boldsymbol{c}(x, v^1), \\ 0 &\leqslant \boldsymbol{c}^{\mathrm{VC}_2}(x, \alpha) \stackrel{\text{def}}{=} (1-\alpha) \cdot \boldsymbol{c}(x, v^2), \\ \alpha &\in [0,1]. \end{aligned}$$

Figure 1.7 depicts the function graphs $\operatorname{gr} \boldsymbol{c}^{\mathrm{VC}_1}(x, \alpha)$ and $\operatorname{gr} \boldsymbol{c}^{\mathrm{VC}_2}(x, \alpha)$, and the feasible set $\left\{(x, \alpha) : \boldsymbol{c}^{\mathrm{VC}_1}(x, \alpha) \geqslant 0 \wedge \boldsymbol{c}^{\mathrm{VC}_2}(x, \alpha) \geqslant 0\right\}$.
The VC formulation produces non–convex feasible sets. Moreover, implied by its special structure, tailored algorithms are required to handle weak stationarity. Alternatively, one can apply smoothing and regularization techniques, accompanied with a homotopy approach to drive the smoothing and regularization parameters to zero. More details about weak stationarity and numerical issues coming along with mathematical problems, which involve Vanishing Constraints, are addressed in Chapter 4.
If the feasible sets resulting from the disjuncts of the original disjunctions are disjoint, then, analogously to the OC formulation, the VC formulation may produce a disconnected feasible



a) Vanishing Constraint of $\boldsymbol{c}(x, \alpha)$.

b) Feasible set of VC of $\boldsymbol{c}(x, v)$.

Figure 1.7: VC applied to the two sided disjunction (1.42a)–(1.42c) from Example 1.30. The left figure shows the function's graphs $\operatorname{gr} \boldsymbol{c}^{\mathrm{VC}_1}(x, \alpha)$ and $\operatorname{gr} \boldsymbol{c}^{\mathrm{VC}_2}(x, \alpha)$ for $\alpha \in [0, 1]$, where $\boldsymbol{c}^{\mathrm{VC}_1}(x, \alpha) = \alpha \cdot \boldsymbol{c}(x, v^1)$ and $\boldsymbol{c}^{\mathrm{VC}_2}(x, \alpha) = (1 - \alpha) \cdot \boldsymbol{c}(x, v^2)$. The right figure depicts the feasible set $\left\{(x, \alpha) : \boldsymbol{c}^{\mathrm{VC}_1}(x, \alpha) \geqslant 0 \wedge \boldsymbol{c}^{\mathrm{VC}_2}(x, \alpha) \geqslant 0\right\}$.

set. In this case, if local methods are used to solve the problem, they would only minimize over the connected component in which the first feasible point was found.
Any solution $\boldsymbol{\omega}(\cdot)$, which is constructed from a fractional and relaxed solution $\boldsymbol{\alpha}(\cdot)$ and where every 0–value is not rounded up, remains feasible for the VCs, given that the states are approximated closely enough. This behavior allows for primal rounding heuristics, which are easily accessible, cf. Jung [259].

1.5.4 Bounds on the Objective Function and Rounding Scheme

Relation Between OCP with Explicit Switches and Relaxed Problem

Based on the Sections 1.5.2 and 1.5.3, we introduce a *convexified* problem which is associated to the explicitly switched OCP (1.37) and allows for a subsequent *relaxation* step. We convexify ODE (1.37b) with OC since it yields tighter relaxations than IC and does not require $\boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \cdot)$ to be defined on the convex hull of Ω. Similar reasons motivate us to use the VC approach to convexify constraint (1.37c).

Definition 1.33 (Convexification and Relaxation)
The *(Partial) Outer Convexification* of OCP (1.37) is given by

$$\begin{aligned}
\min_{\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot), \boldsymbol{\omega}(\cdot)} \quad & \boldsymbol{\varphi}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})) && && \text{(1.46a)}\\
\text{s.t.} \quad & \dot{\boldsymbol{x}}(t) = \sum_{i=1}^{n_\omega} \boldsymbol{\omega}_i(t) \cdot \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), v^i), && t \in \mathcal{T},\\
& \mathbf{0}_{n_\mathrm{c}} \geqslant \boldsymbol{\omega}_i(t) \cdot \boldsymbol{c}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), v^i), && 1 \leqslant i \leqslant n_\omega,\ t \in \mathcal{T},\\
& \mathbf{0}_{n_\mathrm{d}} \geqslant \boldsymbol{d}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), && t \in \mathcal{T},
\end{aligned}$$

$$
\begin{aligned}
\mathbf{0}_{n_\mathrm{r}} &= \boldsymbol{r}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})),\\
1 &= \sum_{i=1}^{n_\omega} \boldsymbol{\omega}_i(t),\ \boldsymbol{\omega}(t) \in \{0,1\}^{n_\omega}, \qquad t \in \mathcal{T}.
\end{aligned}
$$

The *relaxed (Partial) Outer Convexification* of OCP (1.37) is given by

$$
\begin{aligned}
\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),\boldsymbol{\alpha}(\cdot)} \quad & \boldsymbol{\varphi}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})) && && \text{(1.47a)}\\
\text{s.t.} \quad & \dot{\boldsymbol{x}}(t) = \sum_{i=1}^{n_\omega} \boldsymbol{\alpha}_i(t) \cdot \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), v^i), && t \in \mathcal{T},\\
& \mathbf{0}_{n_\mathrm{c}} \geqslant \boldsymbol{\alpha}_i(t) \cdot \boldsymbol{c}(t, \boldsymbol{x}(t), \boldsymbol{u}(t), v^i), && 1 \leqslant i \leqslant n_\omega,\ t \in \mathcal{T},\\
& \mathbf{0}_{n_\mathrm{d}} \geqslant \boldsymbol{d}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), && t \in \mathcal{T},\\
& \mathbf{0}_{n_\mathrm{r}} = \boldsymbol{r}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})),\\
& 1 = \sum_{i=1}^{n_\omega} \boldsymbol{\alpha}_i(t),\ \boldsymbol{\alpha}(t) \in [0,1]^{n_\omega} && t \in \mathcal{T}.
\end{aligned}
$$

△

The binary convexified OCP (1.46) is equivalent to OCP (1.37) in the following sense:

Proposition 1.34
The binary convexified OCP (1.46) has a solution if and only if the explicitly switched OCP (1.37) has a solution. Let $(\boldsymbol{x}_B^*, \boldsymbol{u}_B^*, \boldsymbol{\omega}_B^*)$ be a solution of OCP (1.46). Then $(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{v}^*)$ with $\boldsymbol{x}^* = \boldsymbol{x}_B^*$, $\boldsymbol{u}^* = \boldsymbol{u}_B^*$ and

$$
\boldsymbol{v}^*(t) \overset{\text{def}}{=} \sum_{i=1}^{n_\omega} \boldsymbol{\omega}_i(t) v^i
$$

is a solution of OCP (1.37). △

Proof See Lenders [292, Proposition 6.6]. □

Proposition 1.34 especially states that a solution of the convexified OCP (1.46) provides us with a solution of the explicitly switched OCP (1.37) that we actually want to solve.
In previous sections, we argued that it would be desirable to solve the purely continuous OCP (1.47). But this makes just sense if we could show that for a feasible point of OCP (1.47) there is an essentially feasible point of OCP (1.46) which has essentially the same objective value. The following theorem yields such a result. Note that the arising Lebesgue and Sobolev spaces $L^\infty(\mathcal{T}, \mathbb{R})$ and $W^{1,\infty}(\mathcal{T}, \mathbb{R})$ are introduced in Section 2.4.2 and 2.4.3, respectively.

Theorem 1.35 (Zero Integrality Gap in Function Space)
Let $(\overline{\boldsymbol{x}}, \boldsymbol{u}, \overline{\boldsymbol{\alpha}})$ be feasible for OCP (1.47) and suppose that $t \mapsto \boldsymbol{f}(t, \overline{\boldsymbol{x}}(t), \boldsymbol{u}(t), v^i)$, $i \in [n_\omega]$, are functions of type $W^{1,\infty}(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}})$. Let $\varepsilon > 0$.
Then there exist functions $\boldsymbol{x}^\varepsilon \in W^{1,\infty}(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}})$ and $\boldsymbol{\omega}^\varepsilon \in L^\infty(\mathcal{T}, \{0,1\}^{n_\omega})$ such that

$$
|\boldsymbol{\varphi}(t_\mathrm{s}, \boldsymbol{x}^\varepsilon(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}^\varepsilon(t_\mathrm{f})) - \boldsymbol{\varphi}(t_\mathrm{s}, \overline{\boldsymbol{x}}(t_\mathrm{s}), t_\mathrm{f}, \overline{\boldsymbol{x}}(t_\mathrm{f}))| < \varepsilon
$$

and

$$
\dot{\boldsymbol{x}}^\varepsilon(t) = \sum_{i=1}^{n_\omega} \boldsymbol{\omega}_i^\varepsilon(t) \cdot \boldsymbol{f}(t, \boldsymbol{x}^\varepsilon(t), \boldsymbol{u}(t), v^i), \qquad t \in \mathcal{T},
$$

$$
\begin{aligned}
\boldsymbol{\varepsilon}_{n_c} &\geqslant \boldsymbol{\omega}_i^{\varepsilon}(t) \cdot \boldsymbol{c}(t, \boldsymbol{x}^{\varepsilon}(t), \boldsymbol{u}(t), v^i), && 1 \leqslant i \leqslant n_\omega,\ t \in \mathcal{T},\\
\boldsymbol{\varepsilon}_{n_d} &\geqslant \boldsymbol{d}(t, \boldsymbol{x}^{\varepsilon}(t), \boldsymbol{u}(t)), && t \in \mathcal{T},\\
\boldsymbol{0}_{n_r} &= \boldsymbol{r}(t_s, \boldsymbol{x}^{\varepsilon}(t_s), t_f, \boldsymbol{x}^{\varepsilon}(t_f)),\\
1 &= \sum_{i=1}^{n_\omega} \boldsymbol{\omega}_i^{\varepsilon}(t), && t \in \mathcal{T}.
\end{aligned}
$$

△

Proof See Lenders [292, Theorem 6.7]. □

Given any feasible point of the relaxed OCP (1.47), Theorem 1.35 guarantees that this point can be approximated arbitrary well by a binary feasible point. The prescribed accuracy $\varepsilon > 0$ impacts on the binary feasible point.

Rounding Scheme

What remains is to find a way how we can construct a binary solution from a relaxed solution. Early results in the setting without mode–dependent constraints $\boldsymbol{c}(\cdot)$ were published by Sager [380]. He established a reconstruction algorithm (*Sum–Up Rounding*) which has linear complexity in the size of the temporal grid. Jung [259] developed an algorithm (*Next–Forced Rounding*) with improved approximation properties but quadratic complexity in the size of the temporal grid. Recently, Lenders extended Sager's Sum–Up Rounding algorithm such that it also addresses the case of mode–dependent constraints. This algorithm is called *Vanishing Constraint SOS–Sum–Up Rounding* and sketched in the following definition.

Definition 1.36 (Vanishing Constraint SOS–Sum–Up Rounding Scheme)
Given a temporal grid $t_s = t_0 < t_1 < \ldots < t_N = t_f$, the *Vanishing Constraint SOS–Sum–Up Rounding (VC–SOS–SUR) Scheme* is defined recursively by $\boldsymbol{\omega}^{\mathrm{VC}} \restriction_{(t_i, t_{i+1})} \stackrel{\text{def}}{=} \left(\omega_j^i\right)_{j=1}^{n_\omega}$ with

$$
\omega_j^i \stackrel{\text{def}}{=} \left[j = \operatorname*{argmax}_{\substack{1 \leqslant k \leqslant n_\omega \\ \int_{t_i}^{t_{i+1}} \boldsymbol{\alpha}_k(t)\,\mathrm{d}t > 0}} \int_0^{t_{i+1}} \boldsymbol{\alpha}_k(t)\,\mathrm{d}t - \int_0^{t_i} \boldsymbol{\omega}_k^{\mathrm{VC}}(t)\,\mathrm{d}t \right]. \tag{1.48}
$$

In case the maximum in (1.48) is attained for more than one index k, exactly one must be selected by argmax. △

Compared to Sager's Sum–Up Rounding scheme, the rounding scheme (1.48) adds the term $\int_{t_i}^{t_{i+1}} \boldsymbol{\alpha}_k(t)\,\mathrm{d}t > 0$ when selecting the index. This term plays the role of an additional feasibility requirement that guarantees solutions to stay feasible even after rounding. To specify this more detailed, we introduce the notion of ε–feasible grids.

Definition 1.37 (ε–feasible Grid)
Let $(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\alpha})$ be feasible for the relaxed OCP (1.47) and $\varepsilon > 0$ an acceptable constraint violation. A temporal grid $t_s = t_0 < t_1 < \ldots < t_N = t_f$ is called ε–feasible, if for every $\xi \in L^\infty(\mathcal{T}, \mathbb{R}^{n_x})$ with $\left\|\xi(t) - \boldsymbol{x}(t)\right\| < \varepsilon$, $t \in \mathcal{T}$ the following implication holds:

$$
\int_{t_j}^{t_{j+1}} \boldsymbol{\alpha}_i(t)\,\mathrm{d}t > 0 \qquad \Longrightarrow \qquad \boldsymbol{c}(t, \xi(t), \boldsymbol{u}(t), v^i) \leqslant \boldsymbol{\varepsilon}, \quad t \in [t_j, t_{j+1}]. \tag{1.49}
$$

△

The Lipschitz continuity of the constraint function $\boldsymbol{c}(\cdot)$ implies the existence of an ε–feasible grid. This result is summarized in the following lemma:

Lemma 1.38
Let $(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\alpha})$ be feasible for the relaxed OCP (1.47) and $\varepsilon > 0$ an acceptable constraint violation. Then there exists an ε–feasible grid. △

The VC–SOS–SUR Scheme applied to an ε–feasible grid therefore fulfills the feasibility requirement (1.49). Moreover, it maintains other favorable properties of the Sum–Up Rounding such as preserving the Special Ordered Set (SOS) property and being computational cheap:

Proposition 1.39
Let $(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\alpha})$ be feasible for the relaxed OCP (1.47) and let $t_\mathrm{s} = t_0 < t_1 < \ldots < t_N = t_\mathrm{f}$ be a ε–feasible grid for an acceptable constraint violation $\varepsilon > 0$. Then the VC–SOS–SUR constructed $\boldsymbol{\omega}^{\mathrm{VC}}(\cdot)$ has the following properties:

(i) $\int_{t_i}^{t_{i+1}} \boldsymbol{\alpha}_j(t)\,\mathrm{d}t = 0 \quad \Longrightarrow \quad \boldsymbol{\omega}_j^{\mathrm{VC}}(t) = 0, \quad t \in (t_i, t_{i+1})$.

(ii) the Special Ordered Set Property $\sum_{j=1}^{n_\omega} \boldsymbol{\omega}_j^{\mathrm{VC}}(t) = 1$, $t \in \mathcal{T}$ holds.

(iii) the computational complexity to evaluate $\boldsymbol{\omega}^{\mathrm{VC}}(\cdot)$ is $\mathcal{O}(N)$. △

Proof See Lenders [292, Proposition 6.22]. □

An estimate which relates the distance of the relaxed $\boldsymbol{\alpha}(\cdot)$ and the VC–SOS–SUR generated $\boldsymbol{\omega}^{\mathrm{VC}}(\cdot)$ to the granularity of the temporal grid has not been proven yet. However, based on numerical experiments there is strong evidence for the following conjecture:

Conjecture 1.40
The VC–SOS–SUR Scheme satisfies

$$\sup_{t\in\mathcal{T}} \left\| \int_{t_\mathrm{s}}^{t} \left(\boldsymbol{\alpha}(\tau) - \boldsymbol{\omega}^{\mathrm{VC}}(\tau)\right) \mathrm{d}\tau \right\|_\infty \leqslant \frac{1}{2}(n_\omega - 1) h_{\max}, \quad h_{\max} \stackrel{\mathrm{def}}{=} \max(t_{i+1} - t_i). \qquad \triangle$$

Under Conjecture 1.40 VC–SOS–SUR is a Vanishing Constraint convergent algorithm.

Definition 1.41 (Vanishing Constraint Convergent Algorithm)
A *Vanishing Constraint convergent algorithm* is an algorithm that accepts

- functions $\boldsymbol{\alpha}_j \in L^\infty(\mathcal{T}, [0,1])$ such that $\sum_{j=1}^{n_\omega} \boldsymbol{\alpha}_j(t) = 1$, $t \in \mathcal{T}$
- a temporal grid $t_\mathrm{s} = t_0 < t_1 < \ldots < t_N = t_\mathrm{f}$

as inputs and provides functions $\boldsymbol{\omega}_j \in L^\infty(\mathcal{T}, \{0,1\})$ such that $\sum_{j=1}^{n_\omega} \boldsymbol{\omega}_j(t) = 1$, $t \in \mathcal{T}$ as outputs. Moreover, there exists a constant $C > 0$ such that

$$\int_{t_i}^{t_{i+1}} \boldsymbol{\alpha}_j(t)\,\mathrm{d}t = 0 \quad \Longrightarrow \quad \boldsymbol{\omega}_j(t) = 0, \quad t \in (t_i, t_{i+1}),$$

$$\sup_{t\in\mathcal{T}} \left\| \int_{t_\mathrm{s}}^{t} (\boldsymbol{\alpha}(\tau) - \boldsymbol{\omega}(\tau))\,\mathrm{d}\tau \right\|_\infty \leqslant C \cdot h_{\max}. \qquad \triangle$$

Finally, the following result states that we retrieve an essentially feasible point of OCP (1.46) with essentially the same objective function value if we apply VC–SOS–SUR to a feasible point of the relaxed OCP (1.47).

Proposition 1.42
Let $(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\alpha})$ be feasible for the relaxed OCP (1.47) and $\varepsilon > 0$ an acceptable constraint violation. Under the assumption that a Vanishing Constraint convergent algorithm exists, there exists a ε–feasible temporal grid $t_s = t_0 < t_1 < \ldots < t_N = t_f$ such that the algorithm returns a point $(\boldsymbol{x}^\varepsilon, \boldsymbol{u}, \boldsymbol{\omega}^\varepsilon)$, which fulfills the conclusion of Theorem 1.35, if the input is chosen to be $\boldsymbol{\alpha}(\cdot)$ and $t_0, t_1, \ldots, t_N$. △

Proof See Lenders [292, Proposition 6.19]. □

Chapter 2

Elements of Real and Functional Analysis

This thesis is intended to be self–contained. Hence, the chapter reviews some fundamental concepts of real and functional analysis that are of significance in the remainder of this thesis. The single sections are based on relevant literature and we mainly concentrate on presenting definitions and results. Proofs are dropped for the most part and the interested reader is referred to secondary literature. Readers already familiar with the topic can skip the chapter and just use it as a reference if required.

2.1 Vector Spaces

This section is based on Gerdts [190, Sec. 2.1]. Let $\mathcal{X}$ be a set and $\mathbb{K} = \mathbb{R}$ or $\mathbb{K} = \mathbb{C}$. Algebraic operations $+ : \mathcal{X} \times \mathcal{X} \to \mathcal{X}$ and $\cdot : \mathbb{K} \times \mathcal{X} \to \mathcal{X}$ are called *addition* and *scalar multiplication*, respectively.

Definition 2.1 (Abelian Group)
Let $\mathcal{X}$ be a set and $+ : \mathcal{X} \times \mathcal{X} \to \mathcal{X}$ an addition. The tuple $(\mathcal{X}, +)$ is called an Abelian *group*, if

(i) $(x + y) + z = x + (y + z) \quad \forall x, y, z \in \mathcal{X}$ (associative law);

(ii) $\exists \Theta_{\mathcal{X}} \in \mathcal{X} : \Theta_{\mathcal{X}} + x = x \quad \forall x \in \mathcal{X}$ (existence of null-element);

(iii) $\forall x \in \mathcal{X}\ \exists x' \in X : x + x' = \Theta_{\mathcal{X}}$ (existence of inverse);

(iv) $x + y = y + x \quad \forall x, y \in \mathcal{X}$ (commutative law). △

We call $\Theta_{\mathcal{X}}$ the null–element of $(\mathcal{X}, +)$. If there is no confusion about the set $\mathcal{X}$ we will omit it and simply write Θ instead of $\Theta_{\mathcal{X}}$.

Definition 2.2 (Vector Space)
Let $(\mathcal{X}, +)$ be an Abelian group and $\cdot : \mathbb{K} \times \mathcal{X} \to \mathcal{X}$ a scalar multiplication. The tuple $(\mathcal{X}, +, \cdot)$ is called a *vector space* or *linear space* over $\mathbb{K}$, if the following computational rules hold:

(i) $(s \cdot t) \cdot x = s \cdot (t \cdot x) \quad \forall s, t \in \mathbb{K},\ x \in \mathcal{X}$;

(ii) $s \cdot (x + y) = s \cdot x + s \cdot y \quad \forall s \in \mathbb{K},\ x, y \in \mathcal{X}$;

(iii) $(s + t) \cdot x = s \cdot x + t \cdot x \quad \forall s, t \in \mathbb{K},\ x \in \mathcal{X}$;

(iv) $1 \cdot x = x \quad \forall x \in \mathcal{X}$. △

Null–elements of the special vector spaces $\mathcal{X} = \mathbb{R}^n$ and $\mathcal{X} = \mathbb{R}$ (together with the usual addition and scalar multiplication) are denoted by $\mathbf{0}_n$ and 0, respectively. In case there is no doubt about the value of n we just drop it and write $\mathbf{0}$ instead.

Definition 2.3 (Topology)
A *topology* τ on a set $\mathcal{X}$ is a subset of the power set of $\mathcal{X}$ with the following properties:

(i) $\emptyset$, $\mathcal{X}$ belong to τ.

(ii) The union of arbitrary many elements of τ belongs to τ.

(iii) The intersection of finitely many elements of τ belongs to τ. △

The elements of a topology τ are called *open sets*. The complement of an open set is called *closed set*.
The *topological product* of two topological spaces $(\mathcal{X}, \tau_X)$ and $(\mathcal{Y}, \tau_Y)$ is the topological space $(\mathcal{X} \times \mathcal{Y}, \tau)$. Here $\mathcal{X} \times \mathcal{Y} = \{(x, y) : x \in \mathcal{X}, y \in \mathcal{Y}\}$ is the Cartesian product of the sets $\mathcal{X}$ and $\mathcal{Y}$, and τ consists of arbitrary unions of sets of the form $\{\mathcal{U} \times \mathcal{V} : \mathcal{U} \in \tau_X, \mathcal{V} \in \tau_Y\}$.

Definition 2.4 (Continuity)
Let $(\mathcal{X}, \tau_X)$ and $(\mathcal{Y}, \tau_Y)$ be topological vector spaces. A mapping $F : \mathcal{X} \longrightarrow \mathcal{Y}$ is called *continuous* at $x \in \mathcal{X}$, if for all open sets $\mathcal{V} \in \tau_Y$ containing $y = F(x) \in \mathcal{Y}$ there exists an open set $\mathcal{U} \in \tau_X$ such that $x \in \mathcal{U}$ and $F(\mathcal{U}) \subseteq \mathcal{V}$. △

The mapping F in Definition 2.4 is called continuous on $\mathcal{X}$, if F is continuous at every point $x \in \mathcal{X}$.

Definition 2.5 (Topological Vector Space)
We call a tuple (X, τ) a *topological vector space* over $\mathbb{K}$, if

(i) X is a vector space over $\mathbb{K}$.

(ii) X is equipped with a topology τ.

(iii) addition and scalar multiplication are continuous functions in the given topology. △

Definition 2.6 (Interior Point, Closure, Boundary Point, Dense Set)
Let V be a subset of a topological vector space (X, τ). Then

- a point $x \in V$ is called *interior point* of V, if there exists an open set $U \in \tau$ such that $x \in U$ and $U \subseteq V$. We denote the set of all interior points of V by $\text{int}(V)$.
- the set of all points x satisfying $U \cap V \neq \emptyset$ for all open sets U containing x is called *closure* of V and denoted by $\text{cl}(V)$.
- a point x is called *boundary point* of V, $x \in \partial V$, if $x \in \text{cl}(V)$ and $x \notin \text{int}(V)$.
- V is called *dense* in X, if $\text{cl}(V) = X$. △

Definition 2.7 (Metric Space)
A *metric space* is a tuple (X, d), where X is a set and $d : X \times X \longrightarrow \mathbb{R}$ is a mapping such that for every $x, y, z \in X$ it holds

- $d(x, y) \geqslant 0$ (non-negativity) and $d(x, y) = 0 \Leftrightarrow x = y$ (identity of indiscernibles);
- $d(x, y) = d(y, x)$ (symmetry);
- $d(x, y) \leqslant d(x, z) + d(z, y)$ (triangle inequality).

We call d a metric on X. △

A sequence $\{x_n\}_{n \in \mathbb{N}}$ in a metric space (X, d) is said to *converge* to a point $x \in X$, that is $x_n \to x$, if $\lim_{n \to \infty} d(x_n, x) = 0$.

Definition 2.8 (Cauchy Sequence)
Let (X,d) be a metric space. The sequence $\{x_n\}_{n\in\mathbb{N}}$ is called a *Cauchy sequence*, if

$$\forall \varepsilon > 0\ \exists N(\varepsilon) \in \mathbb{N} : d(x_n, x_m) < \varepsilon \quad \forall n, m > N(\varepsilon).$$ △

Definition 2.9 (Complete Metric Space)
A metric space (X,d) is called *complete*, if every Cauchy sequence in X converges to some point of X. △

Definition 2.10 (Normed Vector Space)
Let X be a vector space over $\mathbb{K}$. We call the tuple $(X, \|\cdot\|_X)$ a *normed vector space*, if $\|\cdot\|_X : X \to \mathbb{R}$ fulfills for all $x, y \in X$ and $s \in \mathbb{K}$ the following conditions:

(i) $\|x\|_X \geqslant 0$ and $\|x\|_X = 0 \Leftrightarrow x = 0$ (positive-definiteness);

(ii) $\|s \cdot x\|_X = |s| \cdot \|x\|_X$ (homogeneity);

(iii) $\|x + y\|_X \leqslant \|x\|_X + \|y\|_X$ (triangle inequality). △

We call the mapping $\|\cdot\|_X : X \to \mathbb{R}$ a *norm* on X. Every norm induces a metric by defining $d(x,y) \stackrel{\text{def}}{=} \|x - y\|_X$. Hence the terminologies 'convergence', 'Cauchy sequence' and 'complete' can be translated to normed vector spaces $(X, \|\cdot\|_X)$.

Definition 2.11 (Banach Space)
A complete normed vector space is called Banach *space.* △

Definition 2.12 (Scalar Product)
Let X be a vector space over $\mathbb{K}$. We call the mapping $\langle\cdot,\cdot\rangle_X : X \times X \to \mathbb{K}$ a *scalar product* or *inner product*, if the following conditions are fulfilled for all $x, y, z \in X$ and $s \in \mathbb{K}$:

(i) $\langle x, y\rangle_X = \overline{\langle y, x\rangle_X}$;

(ii) $\langle x + y, z\rangle_X = \langle x, z\rangle_X + \langle y, z\rangle_X$;

(iii) $\langle x, s \cdot y\rangle_X = s \cdot \langle x, y\rangle_X$;

(iv) $\langle x, x\rangle_X \geqslant 0$ and $\langle x, x\rangle_X = 0 \Leftrightarrow x = \Theta_X$. △

In Definition 2.12, the complex conjugate of a complex number z is denoted by $\overline{z}$. Notice, that $\|x\|_X = \sqrt{\langle x, x\rangle_X}$ induces a norm on a pre–Hilbert space X.

Definition 2.13 (pre–Hilbert Space, Hilbert Space)
Let X be a vector space over $\mathbb{K}$ and $\langle\cdot,\cdot\rangle_X : X \times X \to \mathbb{K}$ a scalar product. The tuple $(X, \langle\cdot,\cdot\rangle_X)$ is called *pre–*Hilbert *space.* A pre–Hilbert space $(X, \langle\cdot,\cdot\rangle_X)$ is called Hilbert*space*, if it is complete with respect to the induced norm $\|\cdot\|_X = \sqrt{\langle\cdot,\cdot\rangle_X}$. △

The tangent cone plays an important role in the context of infinite and finite dimensional optimization. In particular, certain regularity conditions are expressed by means of the tangent cone.

Definition 2.14 (Tangent Cone – Banach Space Version)
For a non–empty subset $\Sigma \subset X$ of a Banach space X the *tangent cone* to Σ at $x \in \Sigma$ is defined as follows:

$$\mathcal{T}(\Sigma, x) \stackrel{\text{def}}{=} \left\{ d \in X \ : \ \exists (x_n)_{n\in\mathbb{N}} \text{ in } \Sigma \text{ with } x_n \to x \text{ and } \exists (t_n)_{n\in\mathbb{N}} \text{ with } t_n \searrow 0, \frac{x_n - x}{t_n} \to d \right\}.$$ △

If the set Σ becomes clear from the context we often drop it and just write $\mathcal{T}(x)$ instead.

2.2 Mappings and Dual Spaces

In this section $(X, \|\cdot\|_X)$, $(Y, \|\cdot\|_Y)$ and $(Z, \|\cdot\|_Z)$ denote real Banach spaces.

Definition 2.15 (Image, Kernel, Preimage)
Let $F : X \longrightarrow Y$ be a mapping from a Banach space $(X, \|\cdot\|_X)$ into Banach space $(Y, \|\cdot\|_Y)$.

(i) The *image* of F is defined as $\operatorname{im}(F) \stackrel{\text{def}}{=} \{F(x) \,:\, x \in X\}$.

(ii) The *kernel* of F is defined as $\ker(F) \stackrel{\text{def}}{=} \{x \in X \,:\, F(x) = \Theta_Y\}$.

(iii) The *preimage* of $S \subseteq Y$ under F is defined as $F^{-1}(S) \stackrel{\text{def}}{=} \{x \in X \,:\, F(x) \in S\}$ △

The mapping $F : X \longrightarrow Y$ is called linear, if

$$F(x_1 + x_2) = F(x_1) + F(x_2) \quad \text{and} \quad F(\lambda x_1) = \lambda F(x_1)$$

holds for all $x_1, x_2 \in X, \lambda \in \mathbb{R}$. F is called bounded, if

$$\|F(x)\|_Y \leqslant C \, \|x\|_X \quad \forall x \in X \tag{2.1}$$

and some $C \geqslant 0$. If $Y = \mathbb{R}$, then F is called a *functional*. The space $\mathscr{L}(X, Y)$ consists of all continuous linear operators L from X to Y. The space is equipped with the norm

$$\|L\|_{\mathscr{L}(X,Y)} \stackrel{\text{def}}{=} \sup_{x \neq \Theta_X} \frac{\|L(x)\|_Y}{\|x\|_X} = \sup_{\|x\|_X \leqslant 1} \|L(x)\|_Y = \sup_{\|x\|_X = 1} \|L(x)\|_Y \,. \tag{2.2}$$

Note that the definition makes sense due to the fact that a linear operator L from X to Y is continuous if and only if L is bounded (see e.g. Yosida [460, Corollary 2, p. 43]). If $L \in \mathscr{L}(X, Y)$ is bijective, then $L^{-1} \in \mathscr{L}(Y, X)$ and L is said to be an isomorphism. We call $L \in \mathscr{L}(X, Y)$ an isometry, if $\|L(x)\|_Y = \|x\|_X$ for all $x \in X$. We call the spaces X and Y isometrically isomorphic, if there exists an isometric isomorphism between X and Y. In this case we write $X \cong Y$.

Definition 2.16 (Upper and Lower Semi–continuity)
A functional $F : X \longrightarrow \mathbb{R}$ is called *upper semi–continuous* at x, if for every sequence $\{x_i\}$ with $x_i \to x$ it holds

$$\limsup_{i \to \infty} F(x_i) \leqslant F(x).$$

A functional $F : X \longrightarrow \mathbb{R}$ is called *lower semi–continuous* at x, if for every sequence $\{x_i\}$ with $x_i \to x$ it holds

$$\liminf_{i \to \infty} F(x_i) \geqslant F(x).$$ △

Definition 2.17 (Dual Space, Adjoint Operator)
The set of all linear continuous functionals on X equipped with the norm

$$\|F\|_{X^*} = \sup_{\|x\|_X \leqslant 1} |F(x)| \tag{2.3}$$

is called *dual space* of X and is denoted by X^* i.e., $X^* = \mathscr{L}(X, \mathbb{R})$. Let $F : X \longrightarrow Y$ be linear. The *adjoint operator* $F^* : Y^* \longrightarrow X^*$ is a linear operator defined by

$$F^*(y^*)(x) = y^*(F(x)) \qquad \forall y^* \in Y^*,\ x \in X. \tag{△}$$

Theorem 2.18
Let $X_1, X_2, \ldots, X_n$ be Banach spaces endowed with the norms $\|\cdot\|_1, \|\cdot\|_2, \ldots, \|\cdot\|_n$. Then the product space

$$X = X_1 \times X_2 \times \ldots \times X_n \tag{2.4}$$

equipped with the norm $\|x\|_X = \max_{1 \leqslant j \leqslant n} \|x_i\|_{X_i}$ is also a Banach space. The dual space of X is given by

$$X^* = \left\{ x^* = (x_1^*, x_2^*, \ldots, x_n^*) \,:\, x_i^* \in X_i^*,\ i \in [n] \right\}, \quad \text{where} \quad x^*(x) = \sum_{i=1}^{n} x_i^*(x_i). \tag{2.5}$$

△

Proof See Wloka [455]. □

Theorem 2.19 (Hahn–Banach Extension Theorem)
Let $(X, \|\cdot\|_X)$ be a normed space and $U \subset X$ be a closed linear subspace of X endowed with the same norm $\|\cdot\|_X$. Then a linear functional $L \in U^*$ on U can be extended to a linear functional $\hat{L}$ on X preserving the norm, i.e. $\hat{L} \restriction_U = L$ and $\left\|\hat{L}\right\|_{X^*} = \|L\|_{U^*} < \infty$. △

Proof See Wloka [455]. □

Definition 2.20 (Duality Pairing)
For a normed space X, the mapping $(\cdot, \cdot)_{X^*,X} : X^* \times X \longrightarrow \mathbb{R}$ given by

$$(x^*, x)_{X^*,X} \stackrel{\text{def}}{=} L(x)$$

is called *duality pairing* of X^* and X. △

Definition 2.21 (Cone and Dual Cone)
Let X be a vector space and $K \subset X$. If $k \in K$ implies $\alpha k \in K$ for all scalar values $\alpha \geqslant 0$, then K is called a *cone* with vertex at Θ_X. For a subset $K \subseteq X$ of a Banach space X we define the *positive dual cone of* K as the set $K^+ \stackrel{\text{def}}{=} \{x^* \in X^* \,:\, x^*(k) \geqslant 0\ \forall\, k \in K\}$. For the *negative dual cone* K^- we just replace '$\geqslant$' with '$\leqslant$'. △

Note that the tangent cone of Definition 2.14 is indeed a cone.

2.3 Differentiability in Banach Spaces

In this section $(X, \|\cdot\|_X)$, $(Y, \|\cdot\|_Y)$, $(Z, \|\cdot\|_Z)$, ... denote Banach spaces over $\mathbb{K}$.

Directional, Gateaux and Fréchet Derivatives

Definition 2.22 (Directional Derivative)
The *directional derivative* of a function $F : X \to Y$ at x in direction $h \in X$ refers to the limit

$$F'(x;h) \stackrel{\text{def}}{=} \lim_{t \searrow 0} \frac{1}{t}\big(F(x+th) - F(x)\big),$$

if it exists. $F'(x;h)$ is then called *directional derivative* of F at x in direction h. Furthermore, the mapping $h \mapsto F'(x;h)$ is called *first variation* of F at x. F is called *directionally differentiable* at x if the limit exists for all $h \in X$. △

Definition 2.23 (Gateaux–Differentiability)
A directionally differentiable mapping $F : X \to Y$ is called Gateaux *differentiable* at x, if there exists a continuous and linear operator $\delta F(x) : X \to Y$ with

$$\lim_{t \searrow 0} \frac{F(x+th) - F(x) - t\delta F(x)(h)}{t} = \Theta_Y$$

for all $h \in X$. The operator $\delta F(x)$ is called Gateaux *differential* of F at x. △

If $F : X \to Y$ is Gateaux–differentiable at x, then F is directionally differentiable and the directional derivative and Gateaux–derivative coincide, i.e. we then have

$$F'(x;h) = \delta F(x)(h).$$

The Gateaux derivative of a functional $F : X \longrightarrow \mathbb{R}$ is an element of the dual space $X^* = \mathscr{L}(X, \mathbb{R})$, i.e.

$$\delta F(x)(h) = (\delta F(x), h)_{X^*,X}. \tag{2.6}$$

Definition 2.24 (Fréchet–Differentiability)
We call the function $F : X \longrightarrow Y$ Fréchet–*differentiable* at $x \in X$ or *differentiable* at $x \in X$, if there exists a continuous and linear operator $F'(x) : X \longrightarrow Y$ with

$$\lim_{\|h\|_X \to 0} \frac{F(x+h) - F(x) - F'(x)(h)}{\|h\|_X} = \Theta_Y$$

for all $h \in X$. △

If $F : X \to Y$ is Fréchet–differentiable at x, then F is continuous and Gateaux differentiable and the Fréchet–derivative and the Gateaux–derivative coincide, i.e. it holds

$$F'(x)(h) = \delta F(x)(h). \tag{2.7}$$

Definition 2.25 (Partial Fréchet–Differentiability)
Let $F : X \times Y \longrightarrow Z$ a mapping. We call F *(partially)* Fréchet–*differentiable* with respect to x at $(x^*, y^*) \in X \times Y$, if $F(\cdot, y^*)$ is Fréchet–differentiable at x^*. The partial derivative of F with respect to x at (x^*, y^*) is denoted by $F'_x(x^*, y^*)$. △

In a similar way we define the partial differential with respect to y.

Higher–Order Derivatives

The definition of higher–order Fréchet–derivatives is constructed recursively: let $k \geqslant 2$ and $F : X \longrightarrow Y$. F is called k–times Fréchet differentiable, if F' is $(k-1)$–times Fréchet differentiable. In particular, F is twice Fréchet–differentiable at $\hat{x}$, if the mapping

$$F'(\cdot) : X \longrightarrow \mathscr{L}(X,Y), \quad x \mapsto F'(x),$$

is Fréchet–differentiable at $\hat{x}$. Hence, $F''(\hat{x})$ is a continuous linear operator from X into $\mathscr{L}(X,Y)$:

$$\begin{aligned} F''(\hat{x}) &\in \mathscr{L}(X, \mathscr{L}(X,Y)) \\ F''(\cdot) &: X \longrightarrow \mathscr{L}(X, \mathscr{L}(X,Y)) \end{aligned}$$

Thus, for every $d_1, d_2 \in X$ it holds

$$\begin{aligned} F''(\hat{x})(d_1)(\cdot) &\in \mathscr{L}(X,Y), \\ F''(\hat{x})(d_1)(d_2) &\in Y. \end{aligned}$$

Definition 2.26 (Bilinear Mapping)
Let X and Y be vector spaces. A mapping $B : X \times Y \longrightarrow Z$ is called *bilinear*, if the partial mappings given by

$$x \mapsto B(x,y) \quad \text{and} \quad y \mapsto B(x,y)$$

are linear for any $y \in Y$ and $x \in X$, respectively. △

$F''(\hat{x})(d_1)(\cdot)$ and $F''(\hat{x})(\cdot)(d_2)$ are linear for every $d_1 \in X$ and $d_2 \in X$, respectively. According to Definition 2.26 $F''(\hat{x})$ is a bilinear mapping.

Basic Theorems of Differential Calculus

Lemma 2.27
If $F : X \times Y \longrightarrow Z$ is Fréchet–differentiable at (x^*, y^*), then the partial Fréchet–derivatives $F'_x(x^*, y^*)$ and $F'_y(x^*, y^*)$ exist at (x^*, y^*). Furthermore, it holds for all $x \in X$ and $y \in Y$ that

$$F'(x^*, y^*)(x,y) = F'_x(x^*, y^*)(x) + F'_y(x^*, y^*)(y). \qquad \triangle$$

Proof See Zeidler [467]. □

Theorem 2.28 (Mean–Value Theorem)
Let X and Y be linear topological spaces and let $U \subset X$ be open. Furthermore, let $F : U \longrightarrow Y$ be Gateaux–differentiable at every point of the interval $[x, x+h] \subset U$.

(i) If $x \mapsto \delta F(x)(h)$ is a continuous mapping for all points in $[x, x+h]$, then

$$F(x+h) - F(x) = \int_0^1 \delta F(x + \tau \cdot h)(h)\, \mathrm{d}\tau.$$

(ii) If X and Y are Banach spaces, it holds

$$\|F(x+h)-F(x)\|_Y \leqslant \sup_{0\leqslant\tau\leqslant 1} \|\delta F(x+\tau\cdot h)\|_{\mathscr{L}(X,Y)} \cdot \|h\|_X \,,$$

and, for any $L \in \mathscr{L}(X,Y)$

$$\|F(x+h)-F(x)-L(h)\|_Y \leqslant \sup_{0\leqslant\tau\leqslant 1} \|\delta F(x+\tau\cdot h)-L\|_{\mathscr{L}(X,Y)} \cdot \|h\|_X \,. \qquad \triangle$$

Proof See Ioffe and Tikhomirov [251, p. 27]. □

2.4 Function Spaces

This section reviews some basic function spaces. We summarize properties of these function spaces and specify norms and scalar products to end up with concrete Banach and Hilbert spaces that will find a use in the remainder of this thesis.
However, we assume readers to be familiar with Lebesgue measure and integration. This is especially important to introduce Lebesgue spaces (see Section 2.4.2) and Sobolev spaces (see Section 2.4.3). Furthermore, it is essential for the definition of the Lebesgue–Stieltjes integral in Section 2.5. For a brief discussion of that theory we refer the reader to Adams and Fournier [6]. Comprehensive surveys can be found in the monographs of Wloka [455], Yosida [460] and Clarke [111].
Distributions (see Schwartz [398, 399]) which is a concept that generalizes the classic notation of a function are not formally introduced in this thesis as well. This is due to the fact that we need distributions just in one single aspect, where we can sketch the idea behind even without a rigorous discussion of distributions. Excellent sources covering the topic of distributions are the monographs of Zeidler [468], Yosida [460], Berezansky et al. [56] and Dret and Lucquin [139].
In the following sections we consider intervals $\mathcal{I}$ with endpoints $a, b \in \overline{\mathbb{R}}$ (unless particularly restricted to finite values) with $a \leqslant b$. The interval $\mathcal{I}$ can be open, half–open or closed. Ω denotes an open subset of $\mathbb{R}^n$, $n \geqslant 1$.
Let $\boldsymbol{f} : \mathbb{R} \longrightarrow \mathbb{R}$ be a function and $t \in \mathbb{R}$. The limit from the left and the limit from the right is denoted by $\boldsymbol{f}(t^-) = \lim_{s\nearrow t} \boldsymbol{f}(s)$ and $\boldsymbol{f}(t^+) = \lim_{s\searrow t} \boldsymbol{f}(s)$, respectively. Furthermore, we use the abbreviations $\boldsymbol{f}(\infty^-)$ and $\boldsymbol{f}((-\infty)^+)$ for the limits of $\boldsymbol{f}$ to ∞ and $-\infty$.
For functions $\boldsymbol{f} : \mathbb{R}^n \longrightarrow \mathbb{R}$ we use the multiindex notation for partial derivatives. Let $\alpha = [\alpha_1, \ldots, \alpha_n]^T \in \mathbb{N}^n$ be a multiindex. We call the integer $|\alpha| = \sum_{i=1}^n \alpha_i$ the *length* of α and set

$$\partial^\alpha \boldsymbol{f} = \frac{\partial^{|\alpha|} \boldsymbol{f}}{\partial x_1^{\alpha_1}\, \partial x_2^{\alpha_2} \cdots \partial x_n^{\alpha_n}}.$$

The *support* of a function $\boldsymbol{f} : \Omega \longrightarrow \mathbb{R}$, $\operatorname{supp} \boldsymbol{f}$, is defined as the complement of the largest open subset of Ω on which $\boldsymbol{f}(\cdot)$ vanishes. Hence, it is a closed subset of Ω. Informally speaking, a subset $\mathcal{K}$ of Ω is compact if and only if it is a closed and bounded subset that does not touch the boundary in the sense that the distance between $\mathcal{K}$ and $\mathbb{R}^n \setminus \Omega$ is greater than zero, i.e., there is a safety gap between $\mathcal{K}$ and $\partial\Omega$.

2.4.1 Spaces of Continuous Functions

Definition 2.29 (k–Times Continuously Differentiable Functions)
Let $\mathcal{I}$ be any interval and $k \in \mathbb{N}$. The space of *k–times continuously differentiable* functions $\mathcal{C}^k(\mathcal{I}, \mathbb{R})$ is denoted by the vector space consisting of all functions $\boldsymbol{f} : \mathcal{I} \longrightarrow \mathbb{R}$ which, together with all their derivatives $\boldsymbol{f}^{(j)}(\cdot)$ of order $j \leqslant k$, are continuous on $\mathcal{I}$. We use the abbreviation $\mathcal{C}(\mathcal{I}, \mathbb{R}) = \mathcal{C}^0(\mathcal{I}, \mathbb{R})$. △

Compact Domain Let $\mathcal{I} = [a, b]$ be any compact interval and $k \in \mathbb{N}$. The space $\mathcal{C}^k(\mathcal{I}, \mathbb{R})$ of all k–times continuous differentiable functions $\boldsymbol{f} : \mathcal{I} \longrightarrow \mathbb{R}$ equipped with the norm

$$\|\boldsymbol{f}\|_{\mathcal{C}^k(\mathcal{I})} \stackrel{\text{def}}{=} \sum_{j=0}^{k} \max_{t \in \mathcal{I}} \left|\boldsymbol{f}^{(j)}(t)\right|,$$

is a Banach space, cf. Gajewski et al. [176]. The space

$$\mathcal{C}^k(\mathcal{I}, \mathbb{R}^n) \stackrel{\text{def}}{=} \underbrace{\mathcal{C}^k(\mathcal{I}, \mathbb{R}) \times \ldots \times \mathcal{C}^k(\mathcal{I}, \mathbb{R})}_{\text{n times}}$$

of k–times continuously differentiable vector–valued functions $\boldsymbol{f} : \mathcal{I} \longrightarrow \mathbb{R}^n$ with the product norm is a Banach space, cf. Gajewski et al. [176].

Half–Open Domain Let $\mathcal{I} = (a, b]$ any half–open interval and $k \in \mathbb{N}$. Since $\mathcal{I}$ is half–open, functions in $\mathcal{C}^k(\mathcal{I}, \mathbb{R})$ need not be bounded on $\mathcal{I}$. We define $\mathcal{C}_b(\mathcal{I}, \mathbb{R})$ to consist of those functions $\boldsymbol{f} : \mathcal{I} \longrightarrow \mathbb{R}$ for which $\boldsymbol{f}^{(j)}(\cdot)$ is bounded on $\mathcal{I}$ for $0 \leqslant j \leqslant k$. Endowed with the norm

$$\|\boldsymbol{f}\|_{\mathcal{C}_b^k(\mathcal{I})} \stackrel{\text{def}}{=} \sum_{j=0}^{k} \sup_{t \in \mathcal{I}} \left\|\boldsymbol{f}^{(j)}(t)\right\|,$$

$\mathcal{C}_b^k(\mathcal{I}, \mathbb{R})$ is a Banach space, cf. Adams and Fournier [6]. The space

$$\mathcal{C}_b^k(\mathcal{I}, \mathbb{R}^n) \stackrel{\text{def}}{=} \underbrace{\mathcal{C}_b^k(\mathcal{I}, \mathbb{R}) \times \ldots \times \mathcal{C}_b^k(\mathcal{I}, \mathbb{R})}_{\text{n times}}$$

of k–times continuously differentiable vector–valued functions $\boldsymbol{f} : \mathcal{I} \longrightarrow \mathbb{R}^n$ with bounded derivatives equipped with the product norm is a Banach space, cf. Adams and Fournier [6].

Extensions Due to their important role functions with compact support deserve their own notation:

$$\mathcal{C}_c^k(\mathcal{I}, \mathbb{R}) \stackrel{\text{def}}{=} \{\boldsymbol{f} \in \mathcal{C}^k(\mathcal{I}, \mathbb{R}) : \operatorname{supp} \boldsymbol{f} \text{ is compact}\}.$$

The space of infinitely differentiable functions on $\mathcal{I}$ is defined by

$$\mathcal{C}^\infty(\mathcal{I},\mathbb{R}^n) \stackrel{\text{def}}{=} \bigcap_{k\in\mathbb{N}} \mathcal{C}^k(\mathcal{I},\mathbb{R}^n).$$

Likewise we define the function spaces $\mathcal{C}_b^\infty(\mathcal{I},\mathbb{R}^n)$ and $\mathcal{C}_c^\infty(\mathcal{I},\mathbb{R}^n)$.

2.4.2 Lebesgue Spaces

Let Ω be a domain in $\mathbb{R}^n$ and let p be a positive real number. The space $L^p(\Omega,\mathbb{R})$ denotes the function space of all measurable functions $\boldsymbol{f} : \Omega \longrightarrow \mathbb{R}$ with

$$\int_\Omega |\boldsymbol{f}(x)|^p \ \mathrm{d}x < \infty, \tag{2.8}$$

where the integral denotes the Lebesgue integral. In $L^p(\Omega,\mathbb{R})$ we identify functions that are equal *almost everywhere* in Ω. Hence, elements of $L^p(\Omega,\mathbb{R})$ are equivalence classes of measurable functions that satisfy (2.8), where two functions are equivalent if they are equal almost everywhere in Ω. For convenience this distinction is ignored in this thesis, and write $\boldsymbol{f} \in L^p(\Omega,\mathbb{R})$ if $\boldsymbol{f}(\cdot)$ satisfies (2.8), and $\boldsymbol{f} = 0$ in $L^p(\Omega,\mathbb{R})$ if $\boldsymbol{f}(x) = 0$ almost everywhere in Ω. The spaces $L^p(\Omega,\mathbb{R})$, endowed with the norm

$$\|\boldsymbol{f}\|_p \stackrel{\text{def}}{=} \left(\int_\Omega |\boldsymbol{f}(x)|^p \ \mathrm{d}x \right)^{1/p}$$

are Banach spaces for $1 \leqslant p < \infty$, cf. Kufner et al. [280, Theorems 2.8.2].
A measurable function $\boldsymbol{f} : \Omega \longrightarrow \mathbb{R}$ is said to be *essentially bounded* on Ω if there exists a constant K such that $\boldsymbol{f}(x) \leqslant K$ almost everywhere on Ω. We call the greatest lower bound of such constants K the *essential supremum* of $|\boldsymbol{f}|$ on Ω and denote it by $\operatorname{ess\,sup}_{x\in\Omega} |\boldsymbol{f}(x)|$.
The space $L^\infty(\Omega,\mathbb{R})$ consists of all essentially bounded functions $\boldsymbol{f} : \Omega \longrightarrow \mathbb{R}$. As we have done before functions in $L^\infty(\Omega,\mathbb{R})$ are identified if they are equal almost everywhere on Ω. The space $L^\infty(\Omega,\mathbb{R})$, equipped with the norm

$$\|\boldsymbol{f}\|_\infty \stackrel{\text{def}}{=} \operatorname*{ess\,sup}_{x\in\Omega} |\boldsymbol{f}(x)|$$

is a Banach space, cf. Kufner et al. [280, Theorem 2.11.7].
For $1 \leqslant p \leqslant \infty$ the space $L^p(\Omega,\mathbb{R}^n)$ is defined as the product space

$$L^p(\Omega,\mathbb{R}^n) \stackrel{\text{def}}{=} \underbrace{L^p(\Omega,\mathbb{R}) \times \ldots \times L^p(\Omega,\mathbb{R})}_{\text{n times}}.$$

Hence, the space $L^p(\Omega,\mathbb{R}^n)$, equipped with the product norm is a Banach space. The Lebesgue spaces admit local versions

$$L^p_{\text{loc}}(\Omega,\mathbb{R}^n) \stackrel{\text{def}}{=} \{\boldsymbol{f} : \boldsymbol{f} \restriction_{\mathcal{K}} \in L^p(\mathcal{K},\mathbb{R}^n) \text{ for all compact } \mathcal{K} \subset \Omega\}.$$

2.4.3 Sobolev Spaces

The spaces of this section are defined over an arbitrary open set $\Omega \subseteq \mathbb{R}^n$. They are vector subspaces of Lebesgue spaces $L^p(\Omega, \mathbb{R})$. For $1 \leqslant p, q \leqslant \infty$ the space $W^{q,p}(\Omega, \mathbb{R})$ consists of all functions of $L^p(\Omega, \mathbb{R})$ that admit all weak derivatives of order at most q:

$$W^{q,p}(\Omega, \mathbb{R}) \stackrel{\text{def}}{=} \{\boldsymbol{f} \in L^p(\Omega, \mathbb{R}) : \partial^\alpha \boldsymbol{f} \in L^p(\Omega, \mathbb{R}) \text{ for all } 0 \leqslant |\alpha| \leqslant q\}.$$

If we equip the spaces $W^{q,p}(\Omega, \mathbb{R})$ with the appropriate Sobolev–norm from

$$\begin{aligned} \|\boldsymbol{f}\|_{q,p} &\stackrel{\text{def}}{=} \left(\sum_{0 \leqslant |\alpha| \leqslant q} \|\partial^\alpha \boldsymbol{f}\|_p^p \right)^{1/p}, \qquad 1 \leqslant p < \infty, \\ \|\boldsymbol{f}\|_{q,\infty} &\stackrel{\text{def}}{=} \max_{0 \leqslant |\alpha| \leqslant q} \|\partial^\alpha \boldsymbol{f}\|_\infty, \end{aligned}$$

they are called Sobolev spaces over Ω. For $1 \leqslant p, q \leqslant \infty$ the space $W^{q,p}(\Omega, \mathbb{R})$, endowed with the norm $\|\cdot\|_{q,p}$ is a Banach space, cf. Kufner et al. [280, Theorem 5.2.2]. Alternatively, we could also define the Sobolev space $W^{q,p}(\Omega, \mathbb{R})$ as the completion of $\{\boldsymbol{f} \in C^q(\Omega, \mathbb{R}) : \|\boldsymbol{f}\|_{q,p} < +\infty\}$ with respect to the Sobolev–norm $\|\cdot\|_{q,p}$, cf. Meyers and Serrin [324].
For $1 \leqslant p, q \leqslant \infty$ the space $W^{q,p}(\Omega, \mathbb{R}^n)$ is defined as the product space

$$W^{q,p}(\Omega, \mathbb{R}^n) \stackrel{\text{def}}{=} \underbrace{W^{q,p}(\Omega, \mathbb{R}) \times \ldots \times W^{q,p}(\Omega, \mathbb{R})}_{\text{n times}}.$$

Hence, the space $W^{q,p}(\Omega, \mathbb{R}^n)$, equipped with the product norm is a Banach space.

2.4.4 Absolutely Continuous Functions

In the context of ODEs and OCPs in particular the function space $W^{1,p}(\mathcal{I}, \mathbb{R}^n)$ with non–empty intervals $\mathcal{I}$ is of relevance. Following Rudin [378] we will characterize this space using absolutely continuous functions.

Definition 2.30 (Absolutely Continuous Functions)
Let $\mathcal{I}$ be an interval in $\mathbb{R}$. We call a function $\boldsymbol{f} : \mathcal{I} \longrightarrow \mathbb{R}$ *absolutely continuous*, if $\boldsymbol{f}(\cdot)$ is continuous in the following sense: for every $\varepsilon > 0$ there exists a $\delta(\varepsilon) > 0$ such that

$$\sum_{j=1}^{N} (b_j - a_j) < \delta(\varepsilon) \qquad \Longrightarrow \qquad \sum_{j=1}^{N} \left| \boldsymbol{f}(b_j) - \boldsymbol{f}(a_j) \right| < \varepsilon,$$

where $N \in \mathbb{N}$ is arbitrary and $(a_j, b_j) \subset \mathcal{I}$, $1 \leqslant j \leqslant N$, are disjoint intervals. The space of all absolutely continuous functions on interval $\mathcal{I}$ is denoted by $\mathcal{AC}(\mathcal{I}, \mathbb{R})$. △

From Definition 2.30 it is obvious that absolutely continuous functions are in particular continuous as one can simply take $N = 1$. $\mathcal{AC}(\mathcal{I}, \mathbb{R}^n)$ denotes the product space

$$\mathcal{AC}(\mathcal{I}, \mathbb{R}^n) \stackrel{\text{def}}{=} \underbrace{\mathcal{AC}(\mathcal{I}, \mathbb{R}) \times \ldots \times \mathcal{AC}(\mathcal{I}, \mathbb{R})}_{\text{n times}}.$$

Differentiability The following result states that for any function $\boldsymbol{f} \in W^{1,p}(\mathcal{I}, \mathbb{R}^n)$ there exists a function $\hat{\boldsymbol{f}} \in \mathcal{AC}(\mathcal{I}, \mathbb{R}^n)$ such that $\boldsymbol{f} = \hat{\boldsymbol{f}}$ almost everywhere and that the derivative of $\hat{\boldsymbol{f}}$ exists almost everywhere.

Theorem 2.31
Suppose $\boldsymbol{f} \in L^p(\mathcal{I}, \mathbb{R}^n)$. Then $\boldsymbol{f} \in W^{1,p}(\mathcal{I}, \mathbb{R}^n)$, $p \geqslant 1$, if and only if $\boldsymbol{f}(\cdot)$ has a representative $\hat{\boldsymbol{f}}(\cdot)$ that is absolutely continuous on almost all segments $\mathcal{J}$ in $\mathcal{I}$ ($\hat{\boldsymbol{f}} \in \mathcal{AC}(\mathcal{J}, \mathbb{R}^n)$) and whose (classical) derivative belongs to $L^p(\mathcal{I}, \mathbb{R}^n)$. △

Proof See e.g. Ziemer [472, Theorem 2.1.4]. □

Due to this result we can identify the space $W^{1,p}(\mathcal{I}, \mathbb{R}^n)$ for an open non–empty interval $\mathcal{I} \subseteq \mathbb{R}$ with absolutely continuous functions $\boldsymbol{f} \in \mathcal{AC}(\mathcal{I}, \mathbb{R}^n)$ such that $\dot{\boldsymbol{f}} \in L^p(\mathcal{I}, \mathbb{R}^n)$. Using this identification we find the following generalized definition of $W^{1,p}(\mathcal{I}, \mathbb{R}^n)$ for arbitrary bounded non–empty intervals $\mathcal{I} \subseteq \mathbb{R}$.

Definition 2.32 (Absolutely Continuous Functions $W^{1,p}(\mathcal{I}, \mathbb{R}^n)$)
Let $\mathcal{I} \subseteq \mathbb{R}$ be a non–empty bounded interval and $1 \leqslant p \leqslant \infty$. The space $W^{1,p}(\mathcal{I}, \mathbb{R}^n)$ consists of all absolutely continuous functions $\boldsymbol{f} : \mathcal{I} \longrightarrow \mathbb{R}^n$ such that $\dot{\boldsymbol{f}} \in L^p(\mathcal{I}, \mathbb{R}^n)$ and $\|\boldsymbol{f}\|_{1,p} < +\infty$, where the norm is given by the Sobolev norm

$$\|\boldsymbol{f}\|_{1,p} = \begin{cases} \left(\|\boldsymbol{f}\|_p^p + \left\|\dot{\boldsymbol{f}}\right\|_p^p \right)^{1/p}, & 1 \leqslant p < \infty, \\ \max\left\{ \|\boldsymbol{f}\|_\infty, \left\|\dot{\boldsymbol{f}}\right\|_\infty \right\}, & p = \infty. \end{cases}$$ △

Fundamental Theorem of Calculus Now we investigate the relation between absolute continuity and the indefinite Lebesgue integral.

Theorem 2.33
Let $\boldsymbol{f} : \mathcal{I} \longrightarrow \mathbb{R}$ be summable, i.e., we have $\boldsymbol{f} \in L^1(\mathcal{I}, \mathbb{R})$. Then the indefinite integral

$$\boldsymbol{F}(t) = \int_a^t \boldsymbol{f}(\tau)\, \mathrm{d}\tau$$

is absolutely continuous. △

Proof See e.g. Kolmogorov and Fomin [278]. □

In a next step we show that the reverse direction of Theorem 2.33 is valid as well.

Theorem 2.34 (Lebesgue)
Let $\boldsymbol{F} : \mathcal{I} \longrightarrow \mathbb{R}$ be absolutely continuous. Then the derivative $\boldsymbol{F}'(\cdot)$ is summable on $\mathcal{I}$ and

$$\boldsymbol{F}(t) = \boldsymbol{F}(a) + \int_a^t \boldsymbol{F}'(\tau)\, \mathrm{d}\tau.$$ △

Proof See e.g. Kolmogorov and Fomin [278]. □

According to Theorems 2.33 and 2.34 the absolutely continuous functions can be identified as precisely those functions for which the *fundamental theorem of calculus* is valid, i.e., a function $\boldsymbol{F} : \mathcal{I} \longrightarrow \mathbb{R}^n$ is absolutely continuous if and only if there exists a function $\boldsymbol{f} \in L^1(\mathcal{I}, \mathbb{R}^n)$ such that

$$\int_a^t \boldsymbol{f}(\tau)\,\mathrm{d}\tau = \boldsymbol{F}(t) - \boldsymbol{F}(a), \qquad t \in \mathcal{I},$$

and $\boldsymbol{F}' = \boldsymbol{f}$ almost everywhere on $\mathcal{I}$. Consequently, absolutely continuous functions are well behaved in the sense that they coincide with the integral of their derivative. For this reason, they are a natural choice for the components of the differential state in ODE and OCP problems.

2.4.5 The Function Spaces $\mathcal{Y}^k(\mathcal{I}, \mathbb{R})$

In this section we investigate function spaces that are tailored to ODE and OCP solution approximations arising from numerical discretization schemes. In particular, state and control trajectory approximations that are generated by pseudospectral collocation methods fall into this algorithm class. The function spaces will allow us to gain some insight into the structure of OCP costates, cf. Chapter 8. They enabled Beigel [41] to shed light on the relationship between the discrete adjoints of BDF methods and the solution of the adjoint ODE. To a certain extent we lift her results from an ODE to an OCP context in Chapter 9.
Let $\mathcal{I} = [t_s, t_f]$ be a compact non–empty interval, where $\mathcal{I}$ usually denotes the optimization horizon $\mathcal{T}$ in the context of OCPs. Given a natural number N we can form a temporal grid

$$t_s = t_0 < t_1 < \ldots < t_N = t_f \tag{2.9}$$

of $N+1$ disjoint points $t_0, t_1, \ldots, t_N \in \mathcal{I}$. The t_n make up a partition of size N and the horizon interval $\mathcal{I}$ is split into intervals as

$$\mathcal{I} = \overline{\mathcal{I}}_1 \cup \ldots \cup \overline{\mathcal{I}}_N, \tag{2.10}$$

where the intervals $\mathcal{I}_n$ are given by open intervals $\mathcal{I}_n \stackrel{\text{def}}{=} (t_{n-1}, t_n)$ for $n \in [N]$. The interval length of interval $\mathcal{I}_n$ is denoted by $h_n = t_n - t_{n-1}$, $n \in [N]$

Generalized Derivatives Numerical OCP solvers usually provide state approximations that are continuous and piecewise continuously differentiable with respect to a grid as defined in (2.10) whereas control approximations are often chosen to be *piecewise continuous*. We call a function piecewise continuous on an interval if the interval can be split into a finite number of subintervals such that the function is continuous on each subinterval (subinterval without endpoints) and has a finite limit at the endpoints of each subinterval. Notice that we always consider a fixed grid in the subsequent discussion.
Anticipating our later definition we denote the state function space as $\mathcal{Y}^1(\mathcal{I}, \mathbb{R})$ and the control function space as $\mathcal{Y}^0(\mathcal{I}, \mathbb{R})$. Intuitively we would expect that differentiating a function from $\mathcal{Y}^1(\mathcal{I}, \mathbb{R})$ would result in a function from $\mathcal{Y}^0(\mathcal{I}, \mathbb{R})$. However, to achieve this we need *generalized derivatives*.

In order to motivate generalized derivatives we recall the classical *integration by parts formula*

$$\int_a^b \boldsymbol{f}(x)\boldsymbol{g}^{(k)}(x)\,\mathrm{d}x = (-1)^k \int_a^b \boldsymbol{f}^{(k)}(x)\boldsymbol{g}(x)\,\mathrm{d}x \tag{2.11}$$

for fixed $\boldsymbol{f} \in \mathcal{C}^k((a,b),\mathbb{R})$ and all $\boldsymbol{g} \in \mathcal{C}_c^\infty((a,b),\mathbb{R})$. Note that no boundary integrals appear in the integration by parts formula due to the choice $\boldsymbol{g} \in \mathcal{C}_c^\infty((a,b),\mathbb{R})$. If we substitute $\boldsymbol{h} = \boldsymbol{f}^{(k)}$ in formula (2.11) then we obtain the key formula

$$\int_a^b \boldsymbol{f}(x)\boldsymbol{g}^{(k)}(x)\,\mathrm{d}x = (-1)^k \int_a^b \boldsymbol{h}(x)\boldsymbol{g}(x)\,\mathrm{d}x \qquad \forall \boldsymbol{g} \in \mathcal{C}_c^\infty((a,b),\mathbb{R}). \tag{2.12}$$

The following definition is based on the idea that (2.12) remains true for certain non–smooth functions $\boldsymbol{f}(\cdot)$ and $\boldsymbol{h}(\cdot)$.

Definition 2.35 (Generalized Derivative)
Let $\mathcal{J} = (a,b)$ be a non–empty open interval and let $\boldsymbol{f}, \boldsymbol{h} \in L^1_{\mathrm{loc}}(\mathcal{J},\mathbb{R})$. We call $\boldsymbol{h}(\cdot)$ a *generalized derivative* of $\boldsymbol{f}(\cdot)$ of order k iff (2.12) holds. In this case we write $\boldsymbol{h}(\cdot) = \boldsymbol{f}^{(k)}(\cdot)$. △

Note that the definition is well posed since classic and generalized derivatives coincide and each generalized derivative is unique up to a change of the values of $\boldsymbol{h}(\cdot)$ on a set of measure zero, cf. Zeidler [469, Proposition 21.2]. Now it is easy to show that the (generalized) derivative of a function $\boldsymbol{f} \in \mathcal{Y}^1(\mathcal{I},\mathbb{R})$ is given by

$$\boldsymbol{h}(x) = \begin{cases} \boldsymbol{f}'(x) & \text{if } x \in \cup_{n\in[N]} \mathcal{I}_n, \\ \text{arbitrary} & \text{otherwise,} \end{cases}$$

and therefore $\boldsymbol{h} \in \mathcal{Y}^0(\mathcal{I},\mathbb{R})$, cf. Zeidler [469, Example 21.5]. In order to obtain unique derivatives of $\mathcal{Y}^1(\mathcal{I},\mathbb{R})$ we define functions from $\mathcal{Y}^0(\mathcal{I},\mathbb{R})$ to be *continuous from the left.* For convenience we assume piecewise continuous functions to be continuous from the left as well and denote the function space as $\mathcal{C}^{-1}(\mathcal{I},\mathbb{R})$.

Function Space Definition Given a temporal grid (2.9) and an non–negative natural number k we define the function space

$$\mathcal{Y}^k(\mathcal{I},\mathbb{R}) \stackrel{\text{def}}{=} \left\{ \boldsymbol{f} \in \mathcal{C}^{k-1}(\mathcal{I},\mathbb{R}) \,:\, \boldsymbol{f} \restriction_{\mathcal{I}_n} \in \mathcal{C}_b^k(\mathcal{I}_n,\mathbb{R}),\ n \in [N] \right\} \tag{2.13}$$

such that single–component state trajectory approximations $\boldsymbol{X}(\cdot)$ are chosen from $\mathcal{Y}^1(\mathcal{I},\mathbb{R})$ and single–component control trajectory approximations $\boldsymbol{U}(\cdot)$ from $\mathcal{Y}(\mathcal{I},\mathbb{R}) \stackrel{\text{def}}{=} \mathcal{Y}^0(\mathcal{I},\mathbb{R})$. Analogously to previous sections we realize the extension of $\mathcal{Y}^k(\mathcal{I},\mathbb{R})$ for vector–valued functions $\boldsymbol{f} : \mathcal{I} \longrightarrow \mathbb{R}^n$ by means of product spaces and denote them as $\mathcal{Y}^k(\mathcal{I},\mathbb{R}^n)$. In order to make $\mathcal{Y}^k(\mathcal{I},\mathbb{R})$ into a normed vector space we equip it with the norm

$$\|\boldsymbol{f}\|_{\mathcal{Y}^k(\mathcal{I})} \stackrel{\text{def}}{=} \max_{n=1,\ldots,N} \|\boldsymbol{f}\|_{\mathcal{C}_b^k(\mathcal{I}_n)}. \tag{2.14}$$

For later chapters, it is necessary that $\mathcal{Y}^1(\mathcal{I},\mathbb{R})$ and $\mathcal{Y}^0(\mathcal{I},\mathbb{R})$ are Banach spaces. Exemplarily this is shown for $\mathcal{Y}^0(\mathcal{I},\mathbb{R})$ in the following lemma.

Lemma 2.36
The function space $\mathcal{Y}^0(\mathcal{I},\mathbb{R})$ endowed with the norm

$$\|\boldsymbol{f}\|_{\mathcal{Y}(\mathcal{I})} = \max_{n\in[N]} \|\boldsymbol{f}\|_{C_b(\mathcal{I}_n)}$$

is a Banach space. △

Proof To prove the completeness of the space $\mathcal{Y}^0(\mathcal{I},\mathbb{R})$ let $(\boldsymbol{f}_k)_{k\in\mathbb{N}}$ be a Cauchy sequence in $\mathcal{Y}^0(\mathcal{I},\mathbb{R})$. By definition there exists for all $\varepsilon > 0$ a natural number M such that for natural numbers $k, l > M$ it holds that $\|\boldsymbol{f}_k - \boldsymbol{f}_l\|_{\mathcal{Y}(\mathcal{I})} < \varepsilon$. By definition of the norm $\|\cdot\|_{\mathcal{Y}(\mathcal{I})}$ in (2.14) this means that $\|\boldsymbol{f}_k - \boldsymbol{f}_l\|_{C_b(\mathcal{I}_n)} < \varepsilon$ for all $n \in [N]$. Hence, $(\boldsymbol{f}_k)_{k\in\mathbb{N}}$ restricted to all intervals $\mathcal{I}_n$ and equipped with the norm $\|\cdot\|_{C_b(\mathcal{I}_n)}$ are Cauchy sequences. Since these spaces are Banach spaces they have a limit in the space. The limit candidate $\boldsymbol{f}$ for $\mathcal{Y}^0(\mathcal{I},\mathbb{R})$ is constructed from the limits on the intervals $\mathcal{I}_n$ by concatenation. We get

$$\|\boldsymbol{f} - \boldsymbol{f}_k\|_{\mathcal{Y}(\mathcal{I})} = \max_{n\in[N]} \|\boldsymbol{f} - \boldsymbol{f}_k\|_{C_b(\mathcal{I}_n)} \longrightarrow 0, \quad \text{for } k \to \infty,$$

where the convergence to 0 when k tends to infinity follows because of the interval wise construction of $\boldsymbol{f}$. Thus $\mathcal{Y}^0(\mathcal{I},\mathbb{R})$ is complete. □

The proof that $\mathcal{Y}^1(\mathcal{I},\mathbb{R})$ is a Banach space looks very similar to the one of Lemma 2.36 and is omitted here. If we choose $(\boldsymbol{x},\boldsymbol{u}) \in \mathcal{Y}^1(\mathcal{I},\mathbb{R}) \times \mathcal{Y}(\mathcal{I},\mathbb{R})$ then the ODE constraint function

$$F(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)) \stackrel{\text{def}}{=} \dot{\boldsymbol{x}}(\cdot) - \boldsymbol{f}(\cdot,\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot))$$

maps $\mathcal{Y}^1(\mathcal{I},\mathbb{R}) \times \mathcal{Y}(\mathcal{I},\mathbb{R})$ to the space $\mathcal{Y}(\mathcal{I},\mathbb{R})$. Thus $F(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot))$ is continuous from the left on $\mathcal{I}$.

2.4.6 Step Functions

Step functions act as a central element in the definition of the Lebesgue integral. Even though we assumed the Lebesgue integral to be well known we will need step functions for the definition of the Lebesgue–Stieltjes integral in Section 2.5 and for some other reasons. Generally speaking, a step function is a function from the real numbers to themselves which is constant everywhere except at a finite number of points. There are several ways to write step functions mathematically.

Definition 2.37 (Step Function)
Let $\mathcal{I}$ be any interval and $\{\mathcal{I}_1,\mathcal{I}_2,\ldots,\mathcal{I}_N\}$ a finite collection of pairwise disjoint intervals such that $\mathcal{I}$ contains the interval union $\mathcal{J} \stackrel{\text{def}}{=} \mathcal{I}_1 \cup \mathcal{I}_2 \cup \ldots \cup \mathcal{I}_N$, i.e. $\mathcal{J} \subseteq \mathcal{I}$. Let $\{a_1,a_2,\ldots,a_N\}$ be a set of finite non–zero real numbers. Then a function $\boldsymbol{\theta} : \mathcal{I} \longrightarrow \mathbb{R}$, given as

$$\boldsymbol{\theta}(t) = \begin{cases} a_n, & \text{if } t \in \mathcal{I}_n,\ n \in [N], \\ 0, & \text{if } t \in \mathcal{I} \setminus \mathcal{J}, \end{cases}$$

is called *step function.* △

It can be easily seen from Definition 2.37 that the zero function is also a step function. We call the set $\mathcal{J}$ on which $\boldsymbol{\theta}$ is nonzero the *support* of $\boldsymbol{\theta}$. We show two ways how to express step functions alternatively.

Characteristic Function The first alternative makes use of characteristic functions.

Definition 2.38 (Characteristic Function)
Let $\mathcal{I}$ be any interval and let $\mathcal{J}$ be a subinterval of $\mathcal{I}$, i.e. $\mathcal{J} \subseteq \mathcal{I}$. A *characteristic function* is an indicator function $\mathcal{X}_{\mathcal{J}} : \mathcal{I} \to \{0, 1\}$, that is defined as

$$\mathcal{X}_{\mathcal{J}}(t) = \begin{cases} 1, & \text{if } t \in \mathcal{J}, \\ 0, & \text{if } t \in \mathcal{I} \setminus \mathcal{J}. \end{cases}$$ △

It can be easily seen from Definition 2.38 that a characteristic function is a step function. By means of characteristic functions any step function $\boldsymbol{\theta} : \mathcal{I} \longrightarrow \mathbb{R}$ can be written as

$$\boldsymbol{\theta}(t) = \sum_{n=1}^{N} a_n \cdot \mathcal{X}_{\mathcal{I}_n}(t),$$

where the same assumptions and notations are assumed to hold as in Definition 2.37.

Heaviside Function The second alternative to write step functions uses linear combinations of translations of the Heaviside function.

Definition 2.39 (Heaviside Function)
The Heaviside *function* is the indicator function $\mathcal{H} : \mathbb{R} \to \{0, 1\}$

$$\mathcal{H}(t) = \begin{cases} 0, & \text{if } t < 0, \\ 1, & \text{if } t \geqslant 0. \end{cases}$$ △

It can be seen from the definition of the Heaviside function that it is a step function. Specifically the function $\mathcal{H}(\cdot)$ is sometimes called the Heaviside step function. According to Definition 2.39 the Heaviside function is continuous from the right. Alternatively one could also define it as continuous from the left or even discontinuous in 0. For reasons that become clearer in later chapters we prefer the chosen definition. Now we extend the Heaviside function to be defined not on the whole real line but on subsets and to have a discontinuity in other points than 0. Let therefore $\mathcal{I}$ be any interval and let s be a point in $\mathcal{I}$. The Heaviside function with jump in s is the indicator function $\mathcal{H}_s : \mathcal{I} \to \{0, 1\}$, that is defined as

$$\mathcal{H}_s(t) \stackrel{\text{def}}{=} \mathcal{H}(t - s) = \begin{cases} 0, & \text{if } t < s, \\ 1, & \text{if } t \geqslant s. \end{cases}$$

Let M be a positive natural number and let $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_M$ be step functions on the same interval $\mathcal{I}$. Moreover, we assume the $\boldsymbol{\theta}_m$ to have supports of finite total length. It can be easily seen

that for finite real numbers $h_1, h_2, \ldots, h_M$ the function $\boldsymbol{\theta} : \mathcal{I} \longrightarrow \mathbb{R}$ given as

$$\boldsymbol{\theta}(t) = \sum_{m=1}^{M} h_m \cdot \boldsymbol{\theta}_m(t) \tag{2.15}$$

is also a step function on $\mathcal{I}$ and that the support of $\boldsymbol{\theta}$ has finite length. If we choose Heaviside functions $\mathcal{H}_{t_m}$ for the $\boldsymbol{\theta}_m$ in (2.15) then we end up with a step function that is continuous from the right and has a jump of height h_m at points t_m.

Dirac Delta Distribution We are interested to find a derivative of $\mathcal{H}_s(\cdot)$. But there does not exist any function which is the (generalized) derivative of $\mathcal{H}_s(\cdot)$ on $\mathbb{R}$. However, using the so–called Dirac $\boldsymbol{\delta}$–distribution at $t = s$ enables us to define a derivative in the sense of the theory of distributions. A rigorous definition of distributions is beyond the scope of this thesis and we only sketch the idea behind it.

The key to understanding the relation between classical functions $\boldsymbol{f}$ and distributions F is the following formula:

$$F(\boldsymbol{g}) = \int_a^b \boldsymbol{f}(x)\boldsymbol{g}(x)\,\mathrm{d}x \qquad \forall \boldsymbol{g} \in \mathcal{C}_c^\infty((a,b),\mathbb{R}). \tag{2.16}$$

The theory of distributions is based on the principle to express properties of functions $\boldsymbol{f}$ in terms of distributions F via (2.16). By means of this approach distributions can be regarded as *generalized functions*. For instance, let us consider the function $\boldsymbol{f}(\cdot)$ with classic derivative $\boldsymbol{h} = \boldsymbol{f}^{(k)}$ such that $\boldsymbol{h}(\cdot)$ corresponds to the distribution

$$H(\boldsymbol{g}) = \int_a^b \boldsymbol{f}^{(k)}(x)\boldsymbol{g}(x)\,\mathrm{d}x \qquad \forall \boldsymbol{g} \in \mathcal{C}_c^\infty((a,b),\mathbb{R}).$$

Integration by parts yields

$$\int_a^b \boldsymbol{f}^{(k)}(x)\boldsymbol{g}(x)\,\mathrm{d}x = (-1)^k \int_a^b \boldsymbol{f}(x)\boldsymbol{g}^{(k)}(x)\,\mathrm{d}x,$$

and thus

$$H(\boldsymbol{g}) = (-1)^k F(\boldsymbol{g}^{(k)}).$$

In a natural way this motivates us defining the derivative $F^{(k)}$ of F as $F^{(k)} \stackrel{\text{def}}{=} H$ such that

$$F^{(k)}(\boldsymbol{g}) = (-1)^k F(\boldsymbol{g}^{(k)}) \qquad \forall \boldsymbol{g} \in \mathcal{C}_c^\infty((a,b),\mathbb{R}). \tag{2.17}$$

Now we come back to the $\boldsymbol{\delta}$–distribution and its motivation: the so–called $\boldsymbol{\delta}$–"function" describes the density of a mass point with mass $m = 1$ at $t = s$. Following this physical inter-

pretation one would like to set

$$\boldsymbol{\delta}(t-s) = \begin{cases} 0 & \text{if } t \neq s, \\ +\infty & \text{if } t = s, \end{cases}$$

and

$$\int_{-\infty}^{+\infty} \boldsymbol{\delta}(t-s)\boldsymbol{\varphi}(t)\,\mathrm{d}t = \boldsymbol{\varphi}(s) \qquad \forall \boldsymbol{\varphi} \in \mathcal{C}_c^{\infty}(\mathcal{I},\mathbb{R}).$$

Since there is no function satisfying these two properties this definition make no sense from a mathematical point of view. However, it is possible with the help of distributions: for fixed $s \in \mathbb{R}$, we define

$$\boldsymbol{\delta}_s(\boldsymbol{\varphi}) \stackrel{\text{def}}{=} \boldsymbol{\varphi}(s) \qquad \forall \boldsymbol{\varphi} \in \mathcal{C}_c^{\infty}(\mathcal{I},\mathbb{R}).$$

A nice property of distributions is that they possess derivatives of arbitrary order. This allows us to identify the derivative of the Heaviside function $\mathcal{H}_s(\cdot)$ in the sense of the theory of distributions as the $\boldsymbol{\delta}$–distribution $\boldsymbol{\delta}_s$: set

$$T_s(\boldsymbol{\varphi}) = \int_a^b \mathcal{H}_s(t)\,\boldsymbol{\varphi}(t)\,\mathrm{d}t \qquad \forall \boldsymbol{\varphi} \in \mathcal{C}_c^{\infty}(\mathcal{I},\mathbb{R}).$$

By definition (see (2.17)), we have $T_s'(\boldsymbol{\varphi}) = -T_s(\boldsymbol{\varphi}')$ for all $\boldsymbol{\varphi} \in \mathcal{C}_c^{\infty}(\mathcal{I},\mathbb{R})$ and thus we state

$$T_s'(\boldsymbol{\varphi}) = -\int_s^b \boldsymbol{\varphi}'(t)\,\mathrm{d}t = \boldsymbol{\varphi}(s)$$

such that $T_s'(\boldsymbol{\varphi}) = \boldsymbol{\delta}_s(\boldsymbol{\varphi})$.

Example 2.40
Let $a = t_0 < t_1 < \ldots < t_N = b$ be a fixed partition of the compact interval $\mathcal{I} = [a,b]$. The derivative (in the sense of distributions) of the step function

$$\boldsymbol{\Lambda}^h(t) = \sum_{n=1}^{N} \lambda_n h_n \cdot \mathcal{H}_{t_n}(t), \qquad h_n = t_n - t_{n-1}, \tag{2.18}$$

is then given by the Dirac measures at $\{t_n\}_{n=1}^N$ with heights $\{\lambda_n h_n\}_{n=1}^N$.

Area of a Step Function If the support of a step function $\boldsymbol{\theta}$ has finite total length, then we associate with $\boldsymbol{\theta}$ the area $A(\boldsymbol{\theta})$ between the graph of $\boldsymbol{\theta}$ and the x–axis. Here we use the usual convention that areas below the x–axis have a negative sign. Due to linearity of $A(\cdot)$ we get for the step function in (2.15) the area

$$A(\boldsymbol{\theta}) = \sum_{m=1}^{M} h_m \cdot A(\boldsymbol{\theta}_m).$$

Example 2.41
Let us consider the same environment as in Example 2.40 with a partition $a = t_0 < t_1 < \ldots < t_N = b$ of the compact interval $\mathcal{I} = [a, b]$. Let a step function in terms of characteristic functions be given as

$$\boldsymbol{\lambda}^h(t) = \boldsymbol{\theta}_N(t) = \sum_{n=1}^{N} \lambda_n \, \mathcal{X}_{\mathcal{I}_n}(t), \qquad \mathcal{I}_n = [t_{n-1}, t_n). \tag{2.19}$$

Furthermore, let us consider the functional $\boldsymbol{f} \mapsto F(\boldsymbol{f})$ that integrates a function over the interval $(-\infty, t]$

$$F(\boldsymbol{f}) = \int_{-\infty}^{t} \boldsymbol{f}(t) \, \mathrm{d}t$$

Then it holds

$$F(\boldsymbol{\lambda}^h)(t_m) = \int_{-\infty}^{t_m} \boldsymbol{\lambda}^h(t) \, \mathrm{d}t = \sum_{n=1}^{m} \lambda_n h_n, \qquad h_n = t_n - t_{n-1},$$

such that the area of $\boldsymbol{\lambda}^h(\cdot)$ is given by $A(\boldsymbol{\lambda}^h) = F(\boldsymbol{\lambda}^h)(t_N)$. Moreover, we see that the values $F(\boldsymbol{\lambda}^h)(t_m)$, $0 \leqslant m \leqslant N$ represent the areas of the step functions $\boldsymbol{\theta}_m(\cdot)$ such that $F(\boldsymbol{\lambda}^h)(\cdot)$ acts as a sort of discrete antiderivative of $\boldsymbol{\lambda}^h(\cdot)$. Another form of this discrete antiderivative is given by $\boldsymbol{\Lambda}^h(\cdot)$ in (2.18) since $\boldsymbol{\Lambda}^h(t_m) = F(\boldsymbol{\lambda}^h)(t_m)$, $0 \leqslant m \leqslant N$.

2.4.7 Monotone Functions

Definition 2.42 (Monotone (Increasing/Decreasing) Function)
Let $\boldsymbol{f} : \mathbb{R} \longrightarrow \mathbb{R}$ be a function. We call $\boldsymbol{f}$ a *monotone increasing function* if $\boldsymbol{f}(t_1) \leqslant \boldsymbol{f}(t_2)$ for any $t_1 < t_2$. The function $\boldsymbol{f}$ is called a *monotone decreasing function*, if $\boldsymbol{f}(t_1) \geqslant \boldsymbol{f}(t_2)$ whenever $t_1 < t_2$. We call $\boldsymbol{f}$ a *monotone function*, if it is either monotone increasing or monotone decreasing. △

A function $\boldsymbol{f}$ may be monotone increasing or monotone decreasing on a particular interval $\mathcal{I}$ rather than on the entire real line. In this case we say that $\boldsymbol{f}$ is monotone increasing or monotone decreasing on $\mathcal{I}$. The following theorem summarizes some important properties of monotone functions.

Theorem 2.43
Let $\boldsymbol{f} : \mathbb{R} \longrightarrow \mathbb{R}$ be a monotone function. Then $\boldsymbol{f}(t^+)$ and $\boldsymbol{f}(t^-)$ exist and are finite for all $t \in \mathbb{R}$. Moreover, for all $t \in \mathbb{R}$, it holds that

(i) $\boldsymbol{f}(t^-) \leqslant \boldsymbol{f}(t) \leqslant \boldsymbol{f}(t^+)$, if $\boldsymbol{f}$ is monotone increasing;

(ii) $\boldsymbol{f}(t^-) \geqslant \boldsymbol{f}(t) \geqslant \boldsymbol{f}(t^+)$, if $\boldsymbol{f}$ is monotone decreasing.

The limits $\boldsymbol{f}(\infty^-)$ and $\boldsymbol{f}((-\infty)^+)$ also exist, but are not necessarily finite. △

Proof See van Brunt and Carter [431]. □

Corollary 2.44
Let $\boldsymbol{f} : \mathbb{R} \longrightarrow \mathbb{R}$ be a monotone function. Then

(i) $\boldsymbol{f}(a^+) \leqslant \boldsymbol{f}(b^-)$, if $\boldsymbol{f}$ is monotone increasing and $a, b \in \overline{\mathbb{R}}$ with $a < b$.

(ii) $\boldsymbol{f}(a^+) \geqslant \boldsymbol{f}(b^-)$, if $\boldsymbol{f}$ is monotone decreasing and $a, b \in \overline{\mathbb{R}}$ with $a < b$. △

Proof See van Brunt and Carter [431]. □

The following result, due to Lebesgue, reveals differentiability properties of monotone functions.

Theorem 2.45 (Lebesgue)
A monotone function $f : [a, b] \longrightarrow \mathbb{R}$ has a finite derivative almost everywhere on $[a, b]$. △

Proof See e.g. Kolmogorov and Fomin [278, Theorem 6, p. 321]. □

Discontinuity Points of Monotone Functions According to Theorem 2.43 the values $f(t^-)$, $f(t)$ $f(t^+)$ all exist for any real t, if f is a monotone function. Hence, the only discontinuities that a monotone function can have are jump discontinuities.

Definition 2.46 (Jump Discontinuity)
A function $f : \mathbb{R} \longrightarrow \mathbb{R}$ is said to have a *jump discontinuity* at t if

(i) the values $f(t^-)$, $f(t)$ and $f(t^+)$ all exist and are finite; and

(ii) $f(t^-)$, $f(t)$ and $f(t^+)$ are not all equal. △

Functions may fail to be continuous at a point t, for example, because the limit is not finite or $f(t)$ has not been defined. The following theorem specifies the number of discontinuities that a monotone function can have.

Theorem 2.47
Let $f : \mathbb{R} \longrightarrow \mathbb{R}$ be a monotone function. The set of points at which f is discontinuous is either empty, finite, or countably infinite. △

Proof See van Brunt and Carter [431]. □

Definition 2.48 (Jump of a Function)
Let $f : \mathbb{R} \longrightarrow \mathbb{R}$ be a monotone function. The *jump of function* f at $t \in \mathbb{R}$ is defined as

$$\triangle_f(t) \stackrel{\text{def}}{=} f(t^+) - f(t^-).$$ △

Jump Function Now we consider a monotone function $f : [a, b] \longrightarrow \mathbb{R}$ and we assume $f(\cdot)$ to be normed by the condition of its continuity from the right such that

$$f(t) = f(t^+) \qquad \forall t \in [a, b]. \tag{2.20}$$

In case this condition does not hold its validity can always be guaranteed by changing the values of $f(\cdot)$ at all points of discontinuity, i.e., we set $f(t_n) = f(t_n^+)$ for all (possibly infinitely many) discontinuity points t_n. From condition (2.20) we conclude

$$\triangle_f(t) = f(t) - f(t^-) \qquad \forall t \in [a, b].$$

Definition 2.49 (Jump Function)
Let $f : [a, b] \longrightarrow \mathbb{R}$ be a monotone function that is continuous from the right and let the $t_1, t_2, \ldots$ denote its discontinuity points. The *jump function* of $f(\cdot)$ is a function $f_d : [a, b] \longrightarrow \mathbb{R}$ given by the relations

$$f_d(t) = \sum_{n:t_n \leqslant t} \triangle_f(t_n), \quad t \in [a, b]. \qquad \triangle$$

It can be shown that the jump function of a right–continuous nondecreasing (nonincreasing) function that is defined on an interval $[a, b]$ is also a right–continuous nondecreasing (nonincreasing), cf. Berezansky et al. [56, Theorem 13.3]. Note that we assumed monotone functions and their jump functions to be continuous from the right, a fact whose importance becomes clear later on. However, analogous considerations would also hold under the assumption of a continuity from the left.

Remark 2.50
If it is possible to enumerate the points of discontinuity of $f(\cdot)$ either in increasing or decreasing order, then this would imply that its jump function $f_d(\cdot)$ is constant between adjacent discontinuity points. Consequently, $f_d(\cdot)$ is a step function, cf. Section 2.4.6. But in general jump functions may have a more complicated structure.

The following example demonstrates that jump functions may not have any interval of constancy.

Example 2.51
Let an enumeration of the rational numbers be given by $\mathbb{Q} = \{q_n : n \in \mathbb{N}\}$, and let a function $f : \mathbb{R} \longrightarrow \mathbb{R}$ be defined as

$$f(t) = \sum_{n:q_n \leqslant t} \frac{1}{2^n}.$$

The definition of $f(\cdot)$ is well–defined since $\sum_{n \geqslant 0} \frac{1}{2^n}$ is absolutely convergent. For any $u < v$ there is some $q_n \in (u, v)$ such that $f(\cdot)$ is monotone increasing because we have $f(v) \geqslant f(u) + \frac{1}{2^n}$. But $f(\cdot)$ is continuous at the irrational points, and discontinuous at every rational:

$$f(q_n^-) = \sum_{k:q_k < q_n} \frac{1}{2^k} < \sum_{k:q_k \leqslant q_n} \frac{1}{2^k} = f(q_n).$$

Continuous Part of a Monotone Function The following definition helps in specifying some characteristics of monotone functions.

Definition 2.52 (Continuous Part of Monotone Functions)
Let $f : [a, b] \longrightarrow \mathbb{R}$ be a monotone function and $f_d(\cdot)$ the associate jump function. Then we call the function $f_c = f - f_d$ the *continuous part of* $f(\cdot)$. $\triangle$

The term "continuous part" in Definition 2.52 is justified by the fact that $f_c(\cdot)$ is a monotone and continuous function on the interval $[a, b]$, cf. Berezansky et al. [56, Theorem 13.4].

2.4.8 Functions of Bounded Variation

In this section, we introduce the theory of an important class of functions, namely the functions of bounded variation. These are strictly related to monotone functions. We begin with some definitions.

Definition 2.53 (Interval Partition)
A *partition* of the closed interval $\mathcal{I} = [a, b]$ of size $N \in \mathbb{N}$ is a finite sequence $\tau_0, \tau_1, \ldots, \tau_N \in \mathcal{I}$ of $N+1$ real numbers such that

$$a = \tau_0 < \tau_1 < \ldots \tau_N = b.$$

The set of *all partitions* of $\mathcal{I}$ *of size* N is denoted by

$$\mathcal{P}_N(\mathcal{I}) \stackrel{\text{def}}{=} \left\{\tau = \{\tau_n\}_{n=0}^N \,:\, a = \tau_0 < \tau_1 < \ldots \tau_N = b\right\}.$$

The *norm* or *mesh* of $\tau \in \mathcal{P}_N(\mathcal{I})$ is defined as $h(\tau) \stackrel{\text{def}}{=} \max_{1 \leqslant n \leqslant N}(\tau_n - \tau_{n-1})$ and the size N of τ is denoted by $|\tau|$. The set of *all partitions* on $\mathcal{I}$ is denoted by

$$\mathcal{P}(\mathcal{I}) \stackrel{\text{def}}{=} \bigcup_{1 \leqslant N < \infty} \mathcal{P}_N(\mathcal{I}). \qquad \triangle$$

Definition 2.54 (Variation)
Let $\mathcal{I}$ be a closed interval. The *variation* of a function $\boldsymbol{f} : \mathcal{I} \longrightarrow \mathbb{R}$ with respect to a partition $\tau \in \mathcal{P}(\mathcal{I})$ is defined as

$$V(\boldsymbol{f}, \tau, \mathcal{I}) \stackrel{\text{def}}{=} \sum_{n=1}^{|\tau|} |\boldsymbol{f}(\tau_n) - \boldsymbol{f}(\tau_{n-1})|. \qquad \triangle$$

Definition 2.55 (Total Variation)
The *total variation* of a function $\boldsymbol{f} : \mathcal{I} \longrightarrow \mathbb{R}$, where $\mathcal{I}$ is a closed interval, is given by

$$TV(\boldsymbol{f}, \mathcal{I}) \stackrel{\text{def}}{=} \sup_{\tau \in \mathcal{P}(\mathcal{I})} V(\boldsymbol{f}, \tau, \mathcal{I}). \qquad \triangle$$

Function Space Definition With these definitions in hand we can define the functions of bounded variation.

Definition 2.56 (Function of Bounded Variation)
Let $\mathcal{I} = [a, b]$ be any closed interval. A function $\boldsymbol{f} : \mathcal{I} \longrightarrow \mathbb{R}$ is said to be of *bounded variation*, if there exists a constant $K > 0$ such that for any partition $\tau \in \mathcal{P}(\mathcal{I})$ it holds

$$V(\boldsymbol{f}, \tau, \mathcal{I}) = \sum_{j=n}^{|\tau|} |\boldsymbol{f}(\tau_n) - \boldsymbol{f}(\tau_{n-1})| \leqslant K. \qquad \triangle$$

We denote the space of all functions of bounded variation on $\mathcal{I}$ with $\mathcal{BV}(\mathcal{I}, \mathbb{R})$. The space $\mathcal{BV}(\mathcal{I}, \mathbb{R})$ can be endowed with the norm $\|\boldsymbol{f}\|_{\mathcal{BV}(\mathcal{I})} \stackrel{\text{def}}{=} |\boldsymbol{f}(a)| + TV(\boldsymbol{f}, \mathcal{I})$. The space $\mathcal{BV}(\mathcal{I}, \mathbb{R})$ with the norm $\|\cdot\|_{\mathcal{BV}(\mathcal{I})}$ is a Banach space, c.f. Adams [5].
As usual we lift the case of single–valued functions to the one of vector–valued functions $\boldsymbol{f} : \mathcal{I} \longrightarrow \mathbb{R}^n$ by using product spaces such that $\mathcal{BV}(\mathcal{I}, \mathbb{R}^n)$ denotes the product space

$$\mathcal{BV}(\mathcal{I}, \mathbb{R}^n) \stackrel{\text{def}}{=} \underbrace{\mathcal{BV}(\mathcal{I}, \mathbb{R}) \times \ldots \times \mathcal{BV}(\mathcal{I}, \mathbb{R})}_{\text{n times}}. \tag{2.21}$$

Relation to Monotone Functions The following result sheds light on the relation between monotone functions and functions of bounded variation.

Theorem 2.57
A monotone function on the closed interval $\mathcal{I}$ has bounded variation on $\mathcal{I}$ △

Proof See e.g. Natanson [337]. □

Next, we investigate a very important connection between functions of bounded variation and monotone functions. Slightly varying the appropriate part of the proof of Theorem 2.43 shows that if $f(\cdot)$ is monotone on $\mathcal{I}$, where $\mathcal{I}$ is an interval with endpoints a, b, then $f(t^-)$ and $f(t^+)$ exist and are finite for all t such that $a < t < b$, and also $f(a^+)$ and $f(b^-)$ exist (but are not necessarily finite). Furthermore, $f(a^+)$ and $f(b^-)$ are both finite if and only if $\sup\{f(t) : t \in \mathcal{I}\}$ and $\inf\{f(t) : t \in \mathcal{I}\}$ are both finite.

Theorem 2.58
Let $\mathcal{I}$ be any interval. Then a function $f : \mathcal{I} \longrightarrow \mathbb{R}$ has bounded variation on $\mathcal{I}$ if and only if f can be expressed as a difference

$$f = h_1 - h_2,$$

where the functions $h_1, h_2 : \mathcal{I} \longrightarrow \mathbb{R}$ are both monotone increasing on $\mathcal{I}$, and $\sup\{h_1(t) : t \in \mathcal{I}\}$, $\inf\{h_1(t) : t \in \mathcal{I}\}$, $\sup\{h_2(t) : t \in \mathcal{I}\}$ and $\inf\{h_2(t) : t \in \mathcal{I}\}$ are all finite. △

Proof See van Brunt and Carter [431]. □

Corollary 2.59
Let $\mathcal{I}$ be an interval with endpoints a, b and let $f : \mathcal{I} \longrightarrow \mathbb{R}$ be a function with bounded variation on $\mathcal{I}$. Then $f(t^-)$ and $f(t^+)$ exist and are finite for all t such that $a < t < b$, and also $f(a^+)$ and $f(b^-)$ exist and are finite. △

Proof See van Brunt and Carter [431]. □

Note that the expression of a particular function of bounded variation as a difference of monotone increasing functions is by no means unique. For instance, just replacing h_1 and h_2 by $h_1 + k$ and $h_2 + k$, where k is a constant, gives an infinite number of different expressions of this kind.

Corollary 2.60
Every function of bounded variation has a finite derivative almost everywhere. △

Proof Follows as an immediate consequence of Theorems 2.45 and 2.58. □

Corollary 2.61
For every function $f \in L^1(\mathcal{I}, \mathbb{R})$, where $\mathcal{I} = [a, b]$ is a closed interval, the indefinite integral

$$F(t) = \int_a^t f(\tau)\, \mathrm{d}\tau$$

is a function of bounded variation. △

Proof See e.g. Kolmogorov and Fomin [278]. □

Relation to Absolutely Continuous Functions The following result states that the function space of absolutely continuous functions is a subspace of the one of functions of bounded variation.

Theorem 2.62
Let $\mathcal{I}$ be an interval with finite endpoints a, b and let $\boldsymbol{f} : \mathcal{I} \longrightarrow \mathbb{R}$ be absolutely continuous on $\mathcal{I}$, then $\boldsymbol{f}$ has bounded variation on $\mathcal{I}$. △

Proof See van Brunt and Carter [431]. □

In Section 2.4.4 we stated that absolutely continuous functions are exactly the functions for which the fundamental theorem of calculus holds, cf. Theorems 2.33+2.34. However, Corollary 2.61 might suggest that the fundamental theorem can be generalized to functions of bounded variation as well. But if one takes for instance the famous Cantor function as an example it can be easily verified (see e.g. Berezansky et al. [56]) that this is not the case. Another basic example is presented in the following.

Example 2.63
Let us consider the function $\boldsymbol{f} : [0, 1] \longrightarrow \mathbb{R}$ which is defined as $\boldsymbol{f}(t) = \mathcal{H}_1(t)$. Since $\boldsymbol{f}(\cdot)$ is nondecreasing it is of bounded variation and it holds $\boldsymbol{f}(t) = 0$ almost everywhere. But the fundamental theorem of calculus does not hold since

$$0 = \int_0^1 \boldsymbol{f}'(t)\,\mathrm{d}t \neq \boldsymbol{f}(1) - \boldsymbol{f}(0) = 1.$$

The Lebesgue Decomposition By using Definition 2.52 and the discussion afterwards in combination with Theorem 2.58 any function $\boldsymbol{f} \in \mathcal{BV}(\mathcal{I}, \mathbb{R})$ can be represented as a sum

$$\boldsymbol{f}(t) = \boldsymbol{f}_c(t) + \boldsymbol{f}_d(t), \tag{2.22}$$

where $\boldsymbol{f}_c(t)$ is a continuous function of bounded variation and $\boldsymbol{f}_d$ is a jump function. Now let functions $\boldsymbol{f}_a(\cdot)$ and $\boldsymbol{f}_s(\cdot)$ be defined in terms of $\boldsymbol{f}_c(\cdot)$ as

$$\boldsymbol{f}_a(t) \stackrel{\text{def}}{=} \int_a^t \boldsymbol{f}_c'(\tau)\,\mathrm{d}\tau, \tag{2.23}$$

$$\boldsymbol{f}_s(t) \stackrel{\text{def}}{=} \boldsymbol{f}_c(t) - \boldsymbol{f}_a(t) \tag{2.24}$$

such that $\boldsymbol{f}_a(\cdot)$ is an absolutely continuous function and $\boldsymbol{f}_s(\cdot)$ is a function of bounded variation whose derivative vanishes almost everywhere since it holds

$$\boldsymbol{f}_s'(t) = \boldsymbol{f}_c'(t) - \frac{\mathrm{d}}{\mathrm{d}t}\int_a^t \boldsymbol{f}_c'(\tau)\,\mathrm{d}\tau = 0.$$

Functions of the same type as $\boldsymbol{f}_s(\cdot)$ have their own name:

Definition 2.64 (Singular Function)
We call a continuous function of bounded variation a *singular function* if its derivative vanishes almost everywhere. △

Combining Equations (2.22)–(2.24) provides the so–called Lebesgue decomposition which gives a deeper insight into the relation of $\mathcal{BV}(\mathcal{I},\mathbb{R})$ and $\mathcal{AC}(\mathcal{I},\mathbb{R})$:

Theorem 2.65 (Lebesgue Decomposition)
Every function $\boldsymbol{f} \in \mathcal{BV}(\mathcal{I},\mathbb{R})$ can be represented as

$$\boldsymbol{f}(t) = \boldsymbol{f}_a(t) + \boldsymbol{f}_d(t) + \boldsymbol{f}_s(t), \tag{2.25}$$

where $\boldsymbol{f}_a(\cdot)$ is absolutely continuous, $\boldsymbol{f}_d(\cdot)$ is a jump function and $\boldsymbol{f}_s(\cdot)$ is singular. △

Corollary 2.66
Let $\mathcal{I}$ be any interval. Any step function $\boldsymbol{\theta} : \mathcal{I} \longrightarrow \mathbb{R}$ is a function of bounded variation on $\mathcal{I}$. △

Remark 2.67
Note that differentiating (2.25) yields

$$\boldsymbol{f}'(t) = \boldsymbol{f}_a'(t)$$

almost everywhere. For this reason we do not recover a function of bounded variation from an integration of its derivative, but only its absolutely continuous part. The other two parts, namely the jump function and the singular function, simply disappear when they are differentiated.

Normalization An important subspace of the function space of functions of bounded variation is presented in the following definition.

Definition 2.68 (Normalized Function of Bounded Variation)
Let $\mathcal{I} = [a,b]$ be any closed interval. The space of *normalized functions of bounded variation* consists of all functions $\boldsymbol{f} \in \mathcal{BV}(\mathcal{I},\mathbb{R})$, which satisfy $\boldsymbol{f}(a) = 0$ and are continuous from the right on (a,b). It is denoted by $\mathcal{NBV}(\mathcal{I},\mathbb{R})$. △

Definition 2.68 requires functions from $\mathcal{NBV}(\mathcal{I},\mathbb{R})$ to be continuous from the right. In general we could also assume continuity from the left. For reasons that become clear later continuity from the right is more convenient in this thesis. The space $\mathcal{NBV}(\mathcal{I},\mathbb{R}^n)$ is defined in a similar way as described in (2.21) for $\mathcal{BV}(\mathcal{I},\mathbb{R}^n)$. A norm on $\mathcal{NBV}(\mathcal{I},\mathbb{R})$ is given by

$$\|\boldsymbol{f}\|_{\mathcal{NBV}(\mathcal{I})} \stackrel{\text{def}}{=} TV(\boldsymbol{f},\mathcal{I}).$$

The space $\mathcal{NBV}(\mathcal{I},\mathbb{R})$ with the norm $\|\cdot\|_{\mathcal{NBV}(\mathcal{I})}$ is a Banach space, cf. Kolmogorov and Fomin [278].

2.5 The Lebesgue–Stieltjes Integral

We assume the reader to be familiar with the concepts of Riemann- and Lebesgue–Integrals. To refresh those topics we recommend reading the monographs of Adams and Fournier [6], Rudin [378], or Kolmogorov and Fomin [278]. Essential parts of this section about the Lebesgue–Stieltjes integral follow van Brunt and Carter [431]. For most results of this section we omit the proofs and refer instead to the literature cited in [431]. Note that we introduce the Lebesgue–Stieltjes integral in a function space setting where the measure function

is monotone increasing and the integrand function of bounded type. The monotone increasing restriction is then relaxed to functions of bounded variation. The function space setting assumed in this contribution fits into this one.
Let $\boldsymbol{\mu} : \mathbb{R} \longrightarrow \mathbb{R}$ be a monotone increasing function, and let $\mathcal{I}$ be an interval with endpoints a, b. The $\boldsymbol{\mu}$*–measure* of $\mathcal{I}$, denoted by $\mu_{\boldsymbol{\mu}}(\mathcal{I})$, is given by

$$\begin{aligned}
\mu_{\boldsymbol{\mu}}([a,b]) &= \boldsymbol{\mu}(b^+) - \boldsymbol{\mu}(a^-),\\
\mu_{\boldsymbol{\mu}}((a,b]) &= \boldsymbol{\mu}(b^+) - \boldsymbol{\mu}(a^+),\\
\mu_{\boldsymbol{\mu}}([a,b)) &= \boldsymbol{\mu}(b^-) - \boldsymbol{\mu}(a^-),
\end{aligned} \tag{2.26}$$

and if $a < b$

$$\mu_{\boldsymbol{\mu}}((a,b)) = \boldsymbol{\mu}(b^-) - \boldsymbol{\mu}(a^+).$$

For the open interval (a,a), which is of course the empty set, we have the convention $\mu_{\boldsymbol{\mu}}((a,a)) = 0$.

Definition 2.69 (Simple Set)
A *simple set* is a subset of $\mathbb{R}$ that can be expressed as the union of a finite collection of disjoint intervals.$_\triangle$

Let $\mathcal{G}$ be a simple set, where $\mathcal{G} = \bigcup_{n=1}^{N} \mathcal{I}_n$ with disjoint intervals $\mathcal{I}_1, \ldots, \mathcal{I}_N$ and $\boldsymbol{\mu} : \mathbb{R} \longrightarrow \mathbb{R}$ be a monotone increasing function. Then the $\boldsymbol{\mu}$–measure of $\mathcal{G}$ is defined as

$$\mu_{\boldsymbol{\mu}}(\mathcal{G}) \stackrel{\text{def}}{=} \sum_{n=1}^{N} \mu_{\boldsymbol{\mu}}(\mathcal{I}_n).$$

Let $\boldsymbol{\mu} : \mathbb{R} \longrightarrow \mathbb{R}$ be a monotone increasing function, and let $\mathcal{I}$ be any interval. It is clear from the relevant definitions that the support of a step function $\boldsymbol{\theta} : \mathcal{I} \longrightarrow \mathbb{R}$ is a simple set. We call $\boldsymbol{\theta}$ $\boldsymbol{\mu}$*–summable* if the support of $\boldsymbol{\theta}$ is $\boldsymbol{\mu}$–finite. In that case we associate with $\boldsymbol{\theta}$ a real number $A_{\boldsymbol{\mu}}(\boldsymbol{\theta})$ defined by

$$A_{\boldsymbol{\mu}}(\boldsymbol{\theta}) = \sum_{n=1}^{N} h_n \mu_{\boldsymbol{\mu}}(\mathcal{I}_n).$$

Definition of the Integral

In this section we assume $\mathcal{I}$ to be a given interval with endpoints a, b and $\boldsymbol{\mu} : \mathbb{R} \longrightarrow \mathbb{R}$ be a monotone increasing function. Let $\boldsymbol{f} : \mathcal{I} \longrightarrow \mathbb{R}$ be a function that is non–negative on $\mathcal{I}$. We call a sequence $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\theta}_3, \ldots$ *admissible for* $\boldsymbol{f}$, if the following conditions are fulfilled:

(i) $\boldsymbol{\theta}_n$ is an $\boldsymbol{\mu}$–summable step function on $\mathcal{I}$ for $n = 1, 2, 3, \ldots$;

(ii) $\boldsymbol{\theta}_n(t) \geqslant 0$ for $t \in \mathcal{I}$ and for $n = 1, 2, 3, \ldots$;

(iii) $0 \leqslant \boldsymbol{f}(t) \leqslant \sum_{n=1}^{\infty} \boldsymbol{\theta}_n(t)$ for $t \in \mathcal{I}$.

Theorem 2.70
For any non–negative function $f : \mathcal{I} \longrightarrow \mathbb{R}$ there exists an admissible sequence. △

Proof See van Brunt and Carter [431]. □

For any non–negative function $f : \mathcal{I} \longrightarrow \mathbb{R}$ we associate an extended real number $L_\mu(f)$ given by

$$L_\mu(f) \stackrel{\text{def}}{=} \inf\left\{\sum_{n=1}^{\infty} A_\mu(\theta_n)\right\},$$

where the greatest lower bound is taken over all sequences $\theta_1, \theta_2, \theta_3, \ldots$ that are admissible for f. According to Theorem 2.70 the set $\left\{\sum_{n=1}^{\infty} A_\mu(\theta_n)\right\}$ is non–empty and has 0 as lower bound. Hence, $L_\mu(f)$ exists and $L_\mu(f) \geqslant 0$ for any non–negative function $f : \mathcal{I} \longrightarrow \mathbb{R}$.

Theorem 2.71
For any function $f : \mathcal{I} \longrightarrow \mathbb{R}$ we have:

(i) $L_\mu(f^+) \leqslant L_\mu(|f|)$ and $L_\mu(-f^-) \leqslant L_\mu(|f|)$;

(ii) $L_\mu(|s \cdot f|) = |s| \cdot L_\mu(|f|)$ for any finite nonzero real number s (and for $s = 0$, provided that $L_\mu(|f|)$ is finite). △

Proof See van Brunt and Carter [431]. □

In Theorem 2.71 and the remainder of this section, f^+ and f^- denote the positive and negative part of the real–valued function f, i.e., it holds $f^+(t) = \max(f(t), 0)$ and $f^-(t) = \max(-f(t), 0)$. In later chapters f^+ and f^- appear with a different meaning. However, the reader can easily infer the actual meaning from the context.

Theorem 2.72
Let $f_1, f_2, f_3, \ldots$ be a sequence of functions such that $f_n : \mathcal{I} \longrightarrow \mathbb{R}$ and $L_\mu(|f_n|)$ is finite for each $n \in \mathbb{N}$. Let $f : \mathcal{I} \longrightarrow \mathbb{R}$ be such that $L_\mu(|f - f_n|) \to 0$ as $n \to \infty$. Then it holds:

(i) $L_\mu(|f|)$, $L_\mu(f^+)$ and $L_\mu(-f^-)$ are all finite;

(ii) $L_\mu(\left|f^+ - f_n^+\right|) \to 0$, $L_\mu(\left|f^- - f_n^-\right|) \to 0$ and $L_\mu(\left||f| - |f_n|\right|) \to 0$;

(iii) $L_\mu(|f_n|) \to L_\mu(|f|)$, $L_\mu(f_n^+) \to L_\mu(f^+)$ and $L_\mu(-f_n^-) \to L_\mu(-f^-)$. △

Proof See van Brunt and Carter [431]. □

Definition 2.73 (Lebesgue–Stieltjes Integral)
Let $\mu : \mathbb{R} \longrightarrow \mathbb{R}$ be a monotone increasing function and let $f : \mathcal{I} \longrightarrow \mathbb{R}$ be a function with the property that there is a sequence $\theta_1, \theta_2, \theta_3, \ldots$ of μ–summable step functions defined on $\mathcal{I}$ such that $L_\mu(|f - f_n|) \to 0$. The Lebesgue–Stieltjes *integral* of f over $\mathcal{I}$ with respect to μ is defined as

$$\int_{\mathcal{I}} f(t)\, \mathrm{d}\mu(t) \stackrel{\text{def}}{=} L_\mu(f^+) - L_\mu(-f^-).$$ △

Note that under the assumptions of Definition 2.73 $L_\mu(f^+)$, $L_\mu(-f^-)$ and $L_\mu(|f|)$ are all finite according to part (i) of Theorem 2.72.
It is important to ensure that $\int_{\mathcal{I}} \theta(t)\, \mathrm{d}\mu(t) = A_\mu(\theta)$ for any μ–summable step function $\theta : \mathcal{I} \longrightarrow \mathbb{R}$, because only then the integral can justifiably be regarded as an extension of the concept of the area under a graph as defined for step functions. The following Theorem gives a positive answer to this question.

Theorem 2.74
For any μ–summable step function $\theta : \mathcal{I} \longrightarrow \mathbb{R}$, we have $\int_{\mathcal{I}} \theta(t)\, \mathrm{d}\mu(t) = A_\mu(\theta)$. △

The case of vector–valued functions $f, \mu : \mathcal{I} \longrightarrow \mathbb{R}^n$ is reduced to the single–valued case by the definition

$$\int_{\mathcal{I}} f(t)^T \mathrm{d}\mu(t) \stackrel{\text{def}}{=} \sum_{i=1}^{n} \int_{\mathcal{I}} f_i(t)\, \mathrm{d}\mu_i(t).$$

Properties of the Lebesgue–Stieltjes Integral

Measure Functions of Bounded Variation Suppose $\mu : \mathcal{I} \longrightarrow \mathbb{R}$ is a function of bounded variation. According to Corollary 2.59 the one sided limits $\mu(a^+)$ and $\mu(b^-)$ exist and are finite, if a and b are endpoints of a proper subset of $\mathbb{R}$. The function μ can be extended to a function of bounded variation on $\mathbb{R}$ as follows: if a is finite and $a \in \mathcal{I}$, then define $\mu(t)$ to be equal to $\mu(a^+)$ for $t < a$. If a is finite and $a \notin \mathcal{I}$, then define $\mu(t)$ to be equal to $\mu(a^+)$ for $t \leqslant a$. Analogously, we deal with the interval to the right of $\mathcal{I}$. If b is finite and $b \in \mathcal{I}$, then define $\mu(t)$ to be equal to $\mu(b^-)$ for $t > b$. If b is finite and $b \notin \mathcal{I}$, then define $\mu(t)$ to be equal to $\mu(b^-)$ for $t \geqslant b$. Theorem 2.58 allows us to express μ as a difference

$$\mu = \mu_1 - \mu_2,$$

where $\mu_1, \mu_2 : \mathbb{R} \longrightarrow \mathbb{R}$ are both monotone increasing. Those considerations enable us to define the Lebesgue–Stieltjes integral with respect to a measure that is affiliated with any function of bounded variation.
To this end, let $\mathcal{J}$ be any subinterval of $\mathcal{I}$. Then we call a function $f : \mathcal{J} \longrightarrow \mathbb{R}$ integrable over $\mathcal{J}$ with respect to μ, if f is integrable over $\mathcal{J}$ with respect to both μ_1 and μ_2. We define naturally

$$\int_{\mathcal{J}} f(t)\, \mathrm{d}\mu(t) \stackrel{\text{def}}{=} \int_{\mathcal{J}} f(t)\, \mathrm{d}\mu_1(t) - \int_{\mathcal{J}} f(t)\, \mathrm{d}\mu_2(t).$$

It can be proven that the value of $\int_{\mathcal{J}} f(t)\, \mathrm{d}\mu(t)$ does not depend on the particular way in which μ is expressed as a difference of monotone increasing functions.

Elementary Results The following fundamental properties of the Lebesgue–Stieltjes are given without proof. Instead, we refer the reader to the publication of Francis [171] and the monograph of van Brunt and Carter [431] as well as the literature cited therein.

Theorem 2.75 (Linearity of the Integral)
Let $f_n : \mathcal{I} \longrightarrow \mathbb{R}$ be integrable over $\mathcal{I}$ with respect to μ for $n \in [N]$, $N \in \mathbb{N}$, and let the s_n be finite real numbers. Then the sum $\sum_{n=1}^{N} s_n f_n$ is integrable over $\mathcal{I}$ with respect to μ, and it holds

$$\int_{\mathcal{I}} \left(\sum_{n=1}^{N} s_n f_n(t) \right) \mathrm{d}\mu(t) = \sum_{n=1}^{N} s_n \int_{\mathcal{I}} f_n(t) \, \mathrm{d}\mu(t). \qquad \triangle$$

Theorem 2.76
Let $\mathcal{I}$ be a finite number of N pairwise disjoint intervals

$$\mathcal{I} = \mathcal{I}_1 \cup \ldots \cup \mathcal{I}_N.$$

Then it holds

$$\int_{\mathcal{I}} f(t) \, \mathrm{d}\mu(t) = \sum_{n=1}^{N} \int_{\mathcal{I}_n} f(t) \, \mathrm{d}\mu(t)$$

in the sense that if one side exists, then so does the other, and the two are equal. $\triangle$

Theorem 2.77
Let $\mu = \sum_{n=1}^{N} s_n \mu_n$, where each $\mu_n : \mathbb{R} \longrightarrow \mathbb{R}$ is a monotone increasing function and the s_n, $n \in [N]$, are non–negative finite real numbers. If a function $f : \mathcal{I} \longrightarrow \mathbb{R}$ is integrable over $\mathcal{I}$ with respect to each of the $\mu_1, \ldots, \mu_n$, then it is integrable over $\mathcal{I}$ with respect to μ, and it holds

$$\int_{\mathcal{I}} f(t) \, \mathrm{d}\mu(t) = \sum_{n=1}^{N} s_n \int_{\mathcal{I}} f(t) \, \mathrm{d}\mu_n(t). \qquad \triangle$$

Theorem 2.78
(i) Let μ be continuous at a. Then it holds

$$\int_{[a,b]} f(t) \, \mathrm{d}\mu(t) = \int_{(a,b]} f(t) \, \mathrm{d}\mu(t) \quad \text{and} \quad \int_{[a,b)} f(t) \, \mathrm{d}\mu(t) = \int_{(a,b)} f(t) \, \mathrm{d}\mu(t)$$

in the sense that if one side of the equation exists, then so does the other, and both are equal.

(ii) Let μ be continuous at b. Then it holds

$$\int_{[a,b]} f(t) \, \mathrm{d}\mu(t) = \int_{[a,b)} f(t) \, \mathrm{d}\mu(t) \quad \text{and} \quad \int_{(a,b]} f(t) \, \mathrm{d}\mu(t) = \int_{(a,b)} f(t) \, \mathrm{d}\mu(t)$$

in the sense that if one side of the equation exists, then so does the other, and both are equal. $\triangle$

Theorem 2.79
For any function $f(\cdot)$ that is defined at a point c it holds

$$\int_{[c,c]} f(t) \, \mathrm{d}\mu(t) = f(c)\,(\mu(c^+) - \mu(c^-)). \qquad \triangle$$

Proof See e.g. van Brunt and Carter [431, Theorem 6.1.6]. $\square$

We just state at this point that the well–known convergence theorems (Monotone Convergence, Dominated Convergence, Fatou's Lemma) of the Lebesgue theory also hold in a Lebesgue–Stieltjes setting.

Theorem 2.80 (Partial Integration Rule)
Let $f, g : \mathcal{I} \longrightarrow \mathbb{R}$ be functions of bounded variation, and let $\mathcal{G}$ be the set of common points of discontinuity. Then it holds

$$\int_{\mathcal{I}} f(t)\mathrm{d}g(t) + \int_{\mathcal{I}} g(t)\mathrm{d}f(t) = \mu_{fg}(\mathcal{I}) + \sum_{t\in\mathcal{G}} A(t),$$

where

$$A(t) = \left[f(t) - \frac{1}{2}(f(t^+) + f(t^-))\right]\mu_g(\{t\}) + \left[g(t) - \frac{1}{2}(g(t^+) + g(t^-))\right]\mu_f(\{t\}).$$

In particular,

(i) if $\mathcal{G}$ is empty, or if $f(t) = \frac{1}{2}(f(t^+) + f(t^-))$ and $g(t) = \frac{1}{2}(g(t^+) + g(t^-))$ for all $t \in \mathcal{G}$, then

$$\int_{\mathcal{I}} f(t)\mathrm{d}g(t) + \int_{\mathcal{I}} g(t)\mathrm{d}f(t) = \mu_{fg}(\mathcal{I}).$$

(ii) if f and g are continuous on the right at all points of $\mathcal{G}$, then

$$\int_{\mathcal{I}} f(t)\mathrm{d}g(t) + \int_{\mathcal{I}} g(t)\mathrm{d}f(t) = \mu_{fg}(\mathcal{I}) + \sum_{t\in\mathcal{G}} \mu_f(\{t\})\mu_g(\{t\}).$$

(iii) if f and g are continuous on the left at all points of $\mathcal{G}$, then

$$\int_{\mathcal{I}} f(t)\mathrm{d}g(t) + \int_{\mathcal{I}} g(t)\mathrm{d}f(t) = \mu_{fg}(\mathcal{I}) - \sum_{t\in\mathcal{G}} \mu_f(\{t\})\mu_g(\{t\}).$$ △

Theorem 2.81 (Change of Variables)
Let $\mathcal{I}$ be any interval. Let $u : \mathbb{R} \longrightarrow \mathbb{R}$ be a function that is continuous and strictly increasing on the interval $\mathcal{I}$. Then

$$\int_{\mathcal{I}} (f \circ u)(t)\mathrm{d}u(t) = \int_{u(\mathcal{I})} f(t)\mathrm{d}t,$$

where $f \circ u$ denotes the composition of f and u, i.e. $(f \circ u)(t) \stackrel{\text{def}}{=} f(u(t))$ for all $t \in \mathcal{I}$. Additionally, if u is differentiable on $\mathcal{I}$, then we get

$$\int_{\mathcal{I}} (f \circ u)(t)\dot{u}(t)\mathrm{d}t = \int_{u(\mathcal{I})} f(t)\mathrm{d}t.$$

Furthermore, if $\mu : \mathbb{R} \longrightarrow \mathbb{R}$ is monotone increasing, then

$$\int_{\mathcal{I}} (f \circ u)(t)\mathrm{d}(\mu \circ u)(t) = \int_{u(\mathcal{I})} f(t)\mathrm{d}\mu(t).$$

All three results hold in the sense that if one side exists, then so does the other, and both are equal. △

The condition in Theorem 2.81 that u should be strictly increasing on $\mathcal{I}$ is just a theoretical matter. If u is not strictly increasing on $\mathcal{I}$ the interval of integration can usually be split up into subintervals on which u is either strictly increasing or strictly decreasing, or constant, and each of these cases can be dealt with separately. Note that if u is strictly decreasing then $-u$ is strictly increasing and the theorem can still be used with the obvious modifications.

Lemma 2.82
Let $\mu : \mathcal{I} \longrightarrow \mathbb{R}$ be a jump function and let f be a μ–summable function. Then the Lebesgue–Stieltjes integral reduces to a sum given by

$$\int_{\mathcal{I}} f(t)\,\mathrm{d}\mu(t) = \sum_n h_n f(t_n). \tag{2.27}$$

△

Theorem 2.83
If μ is differentiable at all points in an open interval $\mathcal{I}$, then

$$\int_{\mathcal{I}} f(t)\,\mathrm{d}\mu(t) = \int_{\mathcal{I}} f(t)\dot{\mu}(t)\,\mathrm{d}t$$

in the sense that if one side of the equation exists, then so does the other, and both are equal. △

As Kolmogorov and Fomin [278, p. 364f.] point out, Theorem 2.83 also holds for absolutely continuous μ. In later chapters of this contribution we often deal with functions of bounded variations as measure functions. By means of the Lebesgue decomposition (see Theorem 2.65) applied to the measure function $\mu \in \mathcal{BV}(\mathcal{I}, \mathbb{R})$, where $\mu(t) = \mu_a(t) + \mu_d(t) + \mu_s(t)$, and under the assumption of a non–existent singular function part $\mu_s(\cdot)$, we can combine the results of Lemma 2.82 and Theorem 2.83 such that it holds

$$\int_{\mathcal{I}} f(t)\,\mathrm{d}\mu(t) = \int_{\mathcal{I}} f(t)\dot{\mu}_a(t)\,\mathrm{d}t + \sum_n h_n f(t_n).$$

Francis [171, p. 949] states the existence of the integrals in Theorem 2.83 for bounded and measurable f. In particular, this holds true for $f \in \mathcal{BV}(\mathcal{I}, \mathbb{R})$.

Towards the Riemann–Stieltjes Integral

Since our research has some links to the one of Beigel [41] and since she works within a Riemann–Stieltjes setting, we review some results on this integral type. For an introduction to Riemann–Stieltjes integral theory the reader is referred to the monographs of Kolmogorov and Fomin [278] and Natanson [337].

Duality pairing of $\mathcal{NBV}(\mathcal{I}, \mathbb{R})$ and $\mathcal{C}(\mathcal{I}, \mathbb{R})$ Since Riemann–Stieltjes integrals are defined on compact intervals $\mathcal{I} = [a, b]$ we assume $\mathcal{I}$ to be of this kind in the remainder of this section. The following theorem characterizes the dual space of the space of continuous functions.

Theorem 2.84 (Riesz Representation Theorem)
Let $L : \mathcal{C}(\mathcal{I},\mathbb{R}) \longrightarrow \mathbb{R}$ be a linear and continuous functional. Then there exists a $\boldsymbol{\mu} \in \mathcal{BV}(\mathcal{I},\mathbb{R})$ such that for every $\boldsymbol{f} \in \mathcal{C}(\mathcal{I},\mathbb{R})$ it holds

$$L(\boldsymbol{f}) = \int_a^b \boldsymbol{f}(t)\,\mathrm{d}\boldsymbol{\mu}(t). \tag{2.28}$$

Furthermore we get $\|L\|_{\mathscr{L}(\mathcal{C}(\mathcal{I},\mathbb{R}),\mathbb{R})} = \|\boldsymbol{\mu}\|_{\mathcal{NBV}(\mathcal{I})}$. △

Proof See Luenberger [303]. □

$\boldsymbol{\mu}$ is defined almost everywhere in $\mathcal{I}$ with exception of an additive constant, cf. Luenberger [303]. The uniqueness of $\boldsymbol{\mu}$ in Theorem 2.84 only holds if the normalized space $\mathcal{NBV}(\mathcal{I},\mathbb{R})$ of $\mathcal{BV}(\mathcal{I},\mathbb{R})$ is used. Thus, the dual of $\mathcal{C}(\mathcal{I},\mathbb{R})$ is isometrically isomorphic to the normalized space of all function of bounded variation, i.e. $(\mathcal{C}(\mathcal{I},\mathbb{R}))^* = \mathscr{L}(\mathcal{C}(\mathcal{I},\mathbb{R}),\mathbb{R}) \cong \mathcal{NBV}(\mathcal{I},\mathbb{R})$. The duality pairing (Definition 2.20) is given as

$$(\boldsymbol{\mu},\boldsymbol{f})_{\mathcal{NBV}(\mathcal{I},\mathbb{R}),\mathcal{C}(\mathcal{I},\mathbb{R})} = \int_a^b \boldsymbol{f}(t)\,\mathrm{d}\boldsymbol{\mu}(t).$$

Duality pairing of $\mathcal{NBV}(\mathcal{I},\mathbb{R})$ and $\mathcal{Y}(\mathcal{I},\mathbb{R})$ In order to find the appropriate duality pairing between $\mathcal{NBV}(\mathcal{I},\mathbb{R})$ and $\mathcal{Y}(\mathcal{I},\mathbb{R})$, Beigel [41] extended the linear functional given in Equation (2.28) from $\mathcal{C}(\mathcal{I},\mathbb{R})$ to $\bigcup_{n=1}^{N} \mathcal{C}_b(\mathcal{I}_n,\mathbb{R})$ by generalizing the definition of the Riemann–Stieltjes integral to allow for integrands that are continuous from the left.
The existence of an extension $\hat{L}$ of the linear functional L defined in (2.28) from $\mathcal{C}(\mathcal{I},\mathbb{R})$ to $\mathcal{Y}(\mathcal{I},\mathbb{R})$ is guaranteed by the Hahn–Banach Extension Theorem (Theorem 2.19). A suitable extension is provided by

$$\hat{L}(\boldsymbol{f}) \overset{\text{def}}{=} \sum_{n=1}^{N} \int_{\mathcal{I}_n} \boldsymbol{f}(t)\,\mathrm{d}\boldsymbol{\mu}(t) \tag{2.29}$$

using the extended Riemann–Stieltjes integral. This extension $\hat{L}$, restricted to the continuous functions $\boldsymbol{f} \in \mathcal{C}(\mathcal{I},\mathbb{R})$, coincides with L given by (2.28). The extended Riemann–Stieltjes integral was introduced by Beigel [41, Section 5.3.1] and splits the standard Riemann–Stieltjes integral into a sum whose parts have continuous integrands.

2.6 Variational Equalities and Inequalities

In this section, we present some fundamental results in the calculus of variations. We formulate them employing the function spaces $\mathcal{Y}^k(\mathcal{I},\mathbb{R})$ as they are relevant in this thesis.

Lemma 2.85 (Variational Lemma)
Let $\boldsymbol{f},\boldsymbol{g} \in \mathcal{Y}(\mathcal{I},\mathbb{R})$. The relation

$$\int_a^b \left\{\boldsymbol{f}(t)\boldsymbol{h}(t) + \boldsymbol{g}(t)\dot{\boldsymbol{h}}(t)\right\}\,\mathrm{d}t = 0 \tag{2.30}$$

holds for all $\boldsymbol{h} \in \mathcal{Y}^1(\mathcal{I}, \mathbb{R})$ with $\boldsymbol{h}(a) = \boldsymbol{h}(b) = 0$ if, and only if, there exists a constant $C \in \mathbb{R}$ such that

$$\boldsymbol{g}(t) = -\int_t^b \boldsymbol{f}(\tau)\,\mathrm{d}\tau - C. \tag{2.31}$$

△

Proof First, let condition (2.30) hold for all $\boldsymbol{h} \in \mathcal{Y}^1(\mathcal{I}, \mathbb{R})$ with $\boldsymbol{h}(a) = \boldsymbol{h}(b) = 0$ and let $\widetilde{t} \in \mathcal{I} \setminus \{b\}$ not be a point of discontinuity of $\boldsymbol{g}(\cdot)$, i.e., it is $\widetilde{t} \in \mathcal{I} \setminus \{t_i\}_{i=0}^N$. Now, we choose an ε such that $0 < \varepsilon < (b - \widetilde{t})/2$. We define

$$\begin{aligned}
&\boldsymbol{h}(t) = 0 && (a \leqslant t \leqslant \widetilde{t}),\\
&\boldsymbol{h}(t) = \frac{1}{\varepsilon}(t - \widetilde{t}) && (\widetilde{t} \leqslant t \leqslant \widetilde{t} + \varepsilon),\\
&\boldsymbol{h}(t) = 1 && (\widetilde{t} + \varepsilon \leqslant t \leqslant b - \varepsilon),\\
&\boldsymbol{h}(t) = \frac{1}{\varepsilon}(b - t) && b - \varepsilon \leqslant t \leqslant b.
\end{aligned}$$

The function $\boldsymbol{h}(\cdot)$ is in $\mathcal{Y}^1(\mathcal{I}, \mathbb{R})$ and it holds $\boldsymbol{h}(a) = \boldsymbol{h}(b) = 0$. Furthermore, it holds

$$0 = \int_a^b \left\{\boldsymbol{f}(t)\boldsymbol{h}(t) + \boldsymbol{g}(t)\dot{\boldsymbol{h}}(t)\right\}\,\mathrm{d}t = \int_{\widetilde{t}}^b \boldsymbol{f}(t)\boldsymbol{h}(t)\,\mathrm{d}t + \frac{1}{\varepsilon}\int_{\widetilde{t}}^{\widetilde{t}+\varepsilon} \boldsymbol{g}(t)\,\mathrm{d}t - \frac{1}{\varepsilon}\int_{b-\varepsilon}^b \boldsymbol{g}(t)\,\mathrm{d}t.$$

Taking the limit $\varepsilon \to 0$ yields

$$0 = \int_{\widetilde{t}}^b \boldsymbol{f}(t)\,\mathrm{d}t + \boldsymbol{g}(\widetilde{t}) - \boldsymbol{g}(b).$$

Hence, (2.31) holds with $C = -\boldsymbol{g}(b)$. For continuity reasons the statement holds for all t. Let us now assume that (2.31) is satisfied such that it holds $\dot{\boldsymbol{g}} = \boldsymbol{f}$. We calculate

$$\int_a^b \left\{\boldsymbol{f}(t)\boldsymbol{h}(t) + \boldsymbol{g}(t)\dot{\boldsymbol{h}}(t)\right\}\,\mathrm{d}t = \int_a^b \frac{\mathrm{d}}{\mathrm{d}t}\{\boldsymbol{g}(t)\boldsymbol{h}(t)\}\,\mathrm{d}t = \boldsymbol{g}(t)\boldsymbol{h}(t)\big|_a^b = 0$$

since $\boldsymbol{h}(a) = \boldsymbol{h}(b) = 0$. □

A result involving variational inequalities is presented in the following lemma.

Lemma 2.86
Let $\boldsymbol{f} \in \mathcal{Y}(\mathcal{I}, \mathbb{R})$. If the inequality

$$\int_a^b \boldsymbol{f}(t)\boldsymbol{h}(t)\,\mathrm{d}t \geqslant 0$$

holds for all $\boldsymbol{h} \in \mathcal{Y}(\mathcal{I}, \mathbb{R})$ with $\boldsymbol{h}(t) \geqslant 0$ almost everywhere in $\mathcal{I}$, then $\boldsymbol{f}(t) \geqslant 0$ almost everywhere in $\mathcal{I}$. △

Proof We prove the lemma by contradiction and assume $\boldsymbol{f}(t) < 0$ on an interval $\mathcal{I}_0 \subset \mathcal{I}$ with positive measure whose interval boundaries we denote with a_0 and b_0. By choosing $c_0 \stackrel{\text{def}}{=} (a_0 + b_0)/2$ and $\delta \stackrel{\text{def}}{=} (b_0 - a_0)/2$ we define a function

$$\boldsymbol{h}(t) = 0 \qquad (a \leqslant t \leqslant a_0),$$

$$\begin{aligned}
\boldsymbol{h}(t) &= \frac{1}{\delta}(t - a_0) && (a_0 \leqslant t \leqslant c_0),\\
\boldsymbol{h}(t) &= \frac{1}{\delta}(b_0 - t) && (c_0 \leqslant t \leqslant b_0),\\
\boldsymbol{h}(t) &= 0 && b_0 \leqslant t \leqslant b.
\end{aligned}$$

Since $\boldsymbol{h} \in \mathcal{Y}(\mathcal{I}, \mathbb{R})$ and $0 \leqslant \boldsymbol{h}(t) \leqslant 1$ on $\mathcal{I}$ we calculate

$$\int_a^b \boldsymbol{f}(t)\boldsymbol{h}(t)\,\mathrm{d}t = \int_{\mathcal{I}_0} \boldsymbol{f}(t)\boldsymbol{h}(t)\,\mathrm{d}t \leqslant \int_{\mathcal{I}_0} \boldsymbol{f}(t)\,\mathrm{d}t < 0,$$

which is contradictory to the assumption. □

Chapter 3

Optimization in Banach Spaces

This chapter deals with some basic optimization theory. We review the general Banach space setting as well as the special case of finite dimensional nonlinear problems. In our presentation of the material we refer mainly to Gerdts [190] and Nocedal and Wright [341]. We present several important necessary optimality conditions and relevant regularity conditions which guarantee that a solution satisfies those conditions. This is more or less the topic from Section 3.1 to Section 3.5. In Section 3.6, we illustrate how finite dimensional optimization problems are commonly solved numerically.
In this chapter $(X, \|\cdot\|_X)$, $(Y, \|\cdot\|_Y)$, $(Z, \|\cdot\|_Z)$, ... denote Banach spaces over $\mathbb{R}$.

3.1 Problem Formulation

We start with a rather general formulation of an optimization problem:

Definition 3.1 (General Minimization Problem)
Let $f : X \to \mathbb{R}$ be a functional and $\emptyset \neq \Sigma \subseteq X$ a set. A *general minimization problem* is an optimization problem of the form

$$\begin{aligned} \min_x \quad & f(x) \\ \text{s. t.} \quad & x \in \Sigma. \end{aligned} \tag{3.1}$$

f is called objective function and a vector x is called admissible or feasible for problem (3.1), if $x \in \Sigma$. We call Σ the admissible set or feasible set for problem (3.1). Problem (3.1) is called unconstrained, if $\Sigma = X$ holds. Finally, we call problem (3.1) convex, if f and Σ are convex. △

We consider minimization problems exclusively. But this is no restriction, since maximization problems can be transformed into equivalent minimization problems. In problem (3.1) we differ the types of minima given in Definition 3.2.

Definition 3.2 ((Strict) Global Minimum, (Strict) Local Minimum)
The following minimum types for general minimization problems are considered:

(i) $x^* \in \Sigma$ is called *global minimum* of problem (3.1), if

$$f(x^*) \leqslant f(x) \quad \forall x \in \Sigma. \tag{3.2}$$

$x^* \in \Sigma$ is called *strict global minimum* of problem (3.1), if '$<$' holds in (3.2) for all $x \in \Sigma$, $x \neq x^*$.

(ii) $x^* \in \Sigma$ is called *local minimum* of problem (3.1), if there exists a $\varepsilon > 0$ such that

$$f(x^*) \leqslant f(x) \quad \forall x \in \Sigma \cap U_\varepsilon(x^*). \tag{3.3}$$

$x^* \in \Sigma$ is called *strict local minimum* of problem (3.1), if '$<$' holds in (3.3) for all $x \in \Sigma \cap U_\varepsilon(x^*)$, $x \neq x^*$. △

Definition 3.3 (Standard Nonlinear Minimization Problem)
Let $f : X \to \mathbb{R}$ be a functional, $g : X \to Y$, $h : X \to Z$ operators, $S \subseteq X$ a closed, convex set, and $K \subseteq Y$ a closed convex cone with vertex at Θ_Y. A *standard nonlinear minimization problem* is an optimization problem of the form

$$\begin{aligned} \min_x \quad & f(x) \\ \text{s.t.} \quad & g(x) \in K, \\ & h(x) = \Theta_Z, \\ & x \in S. \end{aligned} \tag{3.4}$$

△

Notice, that problem (3.4) is a special case of problem (3.1) with

$$\Sigma = g^{-1}(K) \cap h^{-1}(\Theta_Z) \cap S, \tag{3.5}$$

where $g^{-1}(K) \stackrel{\text{def}}{=} \{x \in X \mid x \in K\}$ is the preimage of K under g and $h^{-1}(K) \stackrel{\text{def}}{=} \{x \in X \mid h(x) = \Theta_Z\}$ denotes the preimage of Θ_Z under h.

3.2 Existence of a Solution

Theorem 3.4 (Weierstrass)
Let Σ be a compact subset of a normed linear space X and $f : \Sigma \to \mathbb{R}$ a lower semi–continuous functional. Then, f achieves its minimum on Σ. △

Proof See e.g. Luenberger [303, p. 40]. □

The following theorem can be found in Alt [13] and provides a generalization of Weierstrass's extreme value theorem.

Theorem 3.5
Let Σ be a subset of a normed linear space X and $f : \Sigma \to \mathbb{R}$ lower semi–continuous. Let the set

$$\operatorname{lev}(f, f(y)) \cap \Sigma = \{x \in \Sigma \mid f(x) \leqslant f(y)\} \tag{3.6}$$

be a nonempty and compact set for some $y \in \Sigma$. Then, f achieves its minimum on Σ. △

3.3 First–Order Necessary Conditions of Fritz–John Type

Theorem 3.6
Let $f : X \to \mathbb{R}$ be Fréchet-differentiable at x^* and let x^* be a local minimum of problem (3.1). Then

$$f'(x^*)(d) \geqslant 0 \quad \forall d \in T(\Sigma, x^*). \tag{3.7}$$

△

Proof See e.g. Clarke [111, Proposition 1.39]. □

Theorem 3.7 (First–Order Necessary Conditions)
Let $f : X \to \mathbb{R}$ and $g : X \to Y$ be Fréchet-differentiable and $h : X \to Z$ continuously Fréchet-differentiable. Let x^* be a local minimum of problem (3.4), $S \subseteq X$ a closed convex set, where $\operatorname{int}(S) \neq \emptyset$, and $K \subseteq Y$ a closed convex cone with vertex at Θ_Y, where $\operatorname{int}(K) \neq \emptyset$. Assume that $\operatorname{im}(h'(x^*))$ is not a proper dense subset of Z. Then there exist nontrivial multipliers $(l_0, \lambda^*, \mu^*) \neq (0, \Theta_{Y^*}, \Theta_{Z^*})$ such that

$$l_0 \geqslant 0, \tag{3.8a}$$
$$\lambda^* \in K^+, \tag{3.8b}$$
$$\lambda^*(g(x^*)) = 0, \tag{3.8c}$$
$$l_0 f'(x^*)(d) - \lambda^*(g'(x^*)(d)) - \mu^*(h'(x^*)(d)) \geqslant 0, \quad \forall d \in S \setminus \{x^*\}. \tag{3.8d}$$

△

Proof See e.g. Gerdts [190, Theorem 2.3.24.]. □

Every point $(x, l_0, \lambda^*, \mu^*) \in X \times \mathbb{R} \times Y^* \times Z^*$, $(l_0, \lambda^*, \mu^*) \neq \Theta$ satisfying the Fritz–John conditions (3.8) is called Fritz–John point of problem (3.4). We call l_0, λ^* and μ^* *Lagrange multipliers* or simply *multipliers*.
Notice, that multipliers $(l_0, \lambda^*, \mu^*) = \Theta$ trivially fulfill the Fritz–John conditions. Hence, the main statement of Theorem 3.7 is that there exist nontrivial multipliers $(l_0, \lambda^*, \mu^*) \neq \Theta$. Unfortunately, the case $l_0 = 0$ cannot be excluded. In this case the objective function f does not enter into the Fritz–John conditions. If $l_0 \neq 0$ we call a Fritz–John point $(x, l_0, \lambda^*, \mu^*)$ KKT point. The following section provides regularity conditions that guarantee a nonzero multiplier l_0.
Note that the conditions in Theorem 3.7 can equivalently be modified such that $\lambda^* \in K^-$. As a consequence, the sign that belongs to λ^* in (3.8d) switches. Sometimes, in particular for the finite dimensional special case, we will use this alternative formulation of Theorem 3.7.

3.4 Constraint Qualifications

Conditions ensuring Fritz–John points to be KKT points are called *regularity conditions* or *Constraint Qualifications (CQs).* In this case, l_0 can be normalized to one, since the multipliers enter (3.8d) linearly.
Robinson [373] postulated the following CQ in the context of stability analysis for generalized inequalities.

Definition 3.8 (Constraint Qualification of Robinson [373])
The Robinson constraint qualification holds at x^* if

$$\begin{bmatrix} \Theta_Y \\ \Theta_Z \end{bmatrix} \in \operatorname{int}\left\{ \begin{bmatrix} g(x^*) + g'(x^*)(x - x^*) - k \\ h'(x^*)(x - x^*) \end{bmatrix} : \; x \in S, \; k \in K \right\}. \tag{3.9}$$

△

The validity of CQ (3.9) ensures a nonzero multiplier l_0.

Theorem 3.9 (KKT–Conditions, Gerdts [190])
Let the assumptions of Theorem 3.7 be satisfied. If the constraint qualifications of Robinson are satisfied at x^* then the assertions of Theorem 3.7 hold with $l_0 = 1$. △

Corollary 3.10 (Linear Independence Constraint Qualification, Gerdts [190])
Let $x^* \in \operatorname{int}(S)$ and let the operator

$$T : X \to Y \times Z, \qquad T \stackrel{\text{def}}{=} (g'(x^*), h'(x^*)) \tag{3.10}$$

be surjective. Then the Robinson constraint qualification (3.9) is fulfilled. △

The following result provides another sufficient condition for the Robinson constraint qualification.

Corollary 3.11 (Mangasarian–Fromowitz Constraint Qualification, Gerdts [190])
Let $g : X \to Y$ and $h : X \to Z$ be Fréchet-differentiable at x^*, $K \subseteq Y$ a closed convex cone with vertex at zero and $\operatorname{int}(K) \neq \emptyset$, $g(x^*) \in K$, $h(x^*) = \Theta_Z$. Furthermore, let the following conditions be fulfilled:

1. Let $h'(x^*)$ be surjective.
2. Let there exist some $d^* \in \operatorname{int}(S \setminus \{x^*\})$ with

$$h'(x^*)(d^*) = \Theta_Z, \tag{3.11}$$
$$g'(x^*)(d^*) \in \operatorname{int}(K \setminus \{g(x^*)\}). \tag{3.12}$$

Then the Robinson constraint qualification (3.9) holds. △

One can show that the Mangasarian–Fromowitz constraint qualification is identical with the Robinson constraint qualifications for problems of type (3.4).

3.5 Optimality Conditions for Finite Dimensional Problems

In this section we address an important special case of Problem (3.4), namely the one with finite dimensional spaces X, Y and Z. Let therefore n be a natural number and let $\mathcal{E}$ and $\mathcal{I}$ be two disjoint finite sets of indices such that $X = \mathbb{R}^n$, $Y = \mathbb{R}^{|\mathcal{I}|}$, $Z = \mathbb{R}^{|\mathcal{E}|}$ and $S \subseteq \mathbb{R}^n$. Furthermore, let

$$\begin{aligned} \boldsymbol{f} &: \mathbb{R}^n \longrightarrow \mathbb{R}, \\ \boldsymbol{c} &: \mathbb{R}^n \longrightarrow \mathbb{R}^{|\mathcal{E} \cup \mathcal{I}|}, \end{aligned}$$

be continuously differentiable functions. The resulting special case of Problem (3.4) is called Nonlinear Programming Problem and is explicitly stated in the following definition.

Definition 3.12 (Nonlinear Programming Problem)
A *Nonlinear Programming Problem (NLP)* is an optimization problem of the form

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & \boldsymbol{f}(x) \\ \text{s. t.} \quad & \boldsymbol{c}_i(x) = 0, \quad i \in \mathcal{E}, \\ & \boldsymbol{c}_i(x) \leqslant 0, \quad i \in \mathcal{I}, \\ & x \in S, \end{aligned} \tag{3.13}$$

where $\boldsymbol{f}$ is called *objective function*, whereas $\boldsymbol{c}_i$, $i \in \mathcal{E}$, are the *equality constraints* and $\boldsymbol{c}_i$, $i \in \mathcal{I}$, are the *inequality constraints*. △

The *feasible set* Σ of Problem (3.13) is the set of points satisfying all of its constraints, i.e.,

$$\Sigma \stackrel{\text{def}}{=} \left\{ x \in S \ : \ \boldsymbol{c}_i(x) = 0,\ i \in \mathcal{E},\ \boldsymbol{c}_i(x) \leqslant 0,\ i \in \mathcal{I} \right\}$$

so that Problem (3.13) can be rewritten more compactly as

$$\min_{x \in \Sigma} \qquad \boldsymbol{f}(x).$$

To characterize solutions of constrained optimization problems we introduce important items of terminology. We start by defining the active set of a NLP.

Definition 3.13 (Active Set)
Let $x \in \Sigma$ be feasible for problem (3.13). The *active set* $\mathcal{A}(x)$ at point x is then given as the equality constraint indices $\mathcal{E}$ together with the inequality constraint indices i for which $\boldsymbol{c}_i(x) = 0$, i.e.,

$$\mathcal{A}(x) \stackrel{\text{def}}{=} \mathcal{E} \cup \{ i \in \mathcal{I} \ : \ \boldsymbol{c}_i(x) = 0 \}. \qquad \triangle$$

We call an inequality constraint $i \in \mathcal{I}$ *active* at a feasible point x if $\boldsymbol{c}_i(x) = 0$ and *inactive* if the strict inequality $\boldsymbol{c}_i(x) < 0$ is satisfied.

Constraint Qualifications for NLPs

We start this section by introducing the *tangent cone* $\mathcal{T}(\Sigma, x)$ to the closed convex set Σ at a point $x \in \Sigma$ and the *linearized feasibility cone* $\mathcal{F}(\Sigma, x)$ of first–order feasible directions at x.

Definition 3.14 (Tangent Cone – Finite Dimensional Version)
Let $x \in \Sigma$ be feasible for Problem (3.13). Then the *tangent cone* of Σ at x is given as

$$\mathcal{T}(\Sigma, x) \stackrel{\text{def}}{=} \left\{ d \in \mathbb{R}^n \ : \ \exists (x_n)_{n \in \mathbb{N}} \text{ in } \Sigma \text{ and } \exists (t_n)_{n \in \mathbb{N}} \text{ with } t_n \searrow 0, \frac{x_n - x}{t_n} \to d \right\}. \qquad \triangle$$

Definition 3.15 (Linearized Feasibility Cone)
Let $x \in \Sigma$ be feasible for Problem (3.13). Then the *linearized feasibility cone* of Σ at x is given as

$$\mathcal{F}(\Sigma, x) \stackrel{\text{def}}{=} \bigcap_{i \in \mathcal{E}} \left\{ d \ : \ d^T \nabla \boldsymbol{c}_i(x) = 0 \right\} \quad \cap \quad \bigcap_{i \in \mathcal{A}(x) \cap \mathcal{I}} \left\{ d \ : \ d^T \nabla \boldsymbol{c}_i(x) \leqslant 0 \right\}. \qquad \triangle$$

It is easy to verify that $\mathcal{T}(\Sigma, x)$ and $\mathcal{F}(\Sigma, x)$ are indeed cones. Note that the definition of the tangent cone does not rely on the algebraic specification of the feasible set Σ, but only on its geometry. However, the linearized feasibility cone depends on the concrete representation of Σ, given by $\boldsymbol{c}_i$, $i \in \mathcal{E} \cup \mathcal{I}$. It holds the inclusion $\mathcal{T}(\Sigma, x) \subseteq \mathcal{F}(\Sigma, x)$.
Constraint Qualifications in the finite dimensional case are conditions under which $\mathcal{T}(\Sigma, x)$ and $\mathcal{F}(\Sigma, x)$ are similar or even identical. This ensures that the linearized feasibility cone $\mathcal{F}(\Sigma, x)$, which is constructed by the algebraic representation $\boldsymbol{c}$ of the feasible set Σ, captures the geometric features of Σ, as represented by the tangent cone $\mathcal{T}(\Sigma, x)$, in the vicinity of x.
With the aid of CQs we will be able to formulate first–order optimality conditions for Problem (3.13). The following definitions summarize some common CQs. However, there are CQs that are unsuited to be checked in algorithms and therefore just relevant from a theoretical perspective.

Definition 3.16 (Constraint Qualifications, Finite Case)
Let $x \in \Sigma$ be feasible for Problem (3.13).

- **GCQ**: Guignard Constraint Qualification, [212] holds at x if $\mathcal{T}(\Sigma, x)^- = \mathcal{F}(\Sigma, x)^-$.
- **ACQ**: Abadie Constraint Qualification, [1] holds at x if $\mathcal{T}(\Sigma, x) = \mathcal{F}(\Sigma, x)$. △

The Abadie Constraint Qualification can be found in most textbooks like, for example, Nocedal and Wright [341], whereas the Guignard Constraint Qualification can hardly be found in standard textbooks. However, it was noted by Gould and Tolle [207] that it is the weakest CQ which guarantees that, at a local minimum of an optimization problem, there exist Lagrange multipliers such that the KKT conditions (see Theorem 3.9) are first–order optimality conditions.

Definition 3.17 (Linear Independence Constraint Qualification, Finite Case)
Let $x \in \Sigma$ be feasible for Problem (3.13). The *Linear Independence Constraint Qualification (LICQ)*, [231] holds at x, if the following conditions are fulfilled:

(i) $x \in \text{int}(S)$.

(ii) The set of active constraint gradients $\{\nabla \boldsymbol{c}_i(x),\ i \in \mathcal{A}(x)\}$ is linearly independent. △

A useful generalization of the LICQ is the Mangasarian–Fromowitz Constraint Qualification (MFCQ).

Definition 3.18 (Mangasarian–Fromowitz Constraint Qualification, Finite Case)
Let $x \in \Sigma$ be feasible for Problem (3.13). The Mangasarian–Fromowitz *Constraint Qualification (MFCQ)*, [311] holds at x, if the following conditions are satisfied:

(i) The gradients $\nabla \boldsymbol{c}_i(x)$, $i \in \mathcal{E}$, are linearly independent.

(ii) There exists a vector $d \in \text{int}(S \setminus \{x\})$ with

$$d^T \nabla \boldsymbol{c}_i(x) = 0,\ i \in \mathcal{E} \quad \text{and} \quad d^T \nabla \boldsymbol{c}_i(x) < 0,\ i \in \mathcal{A}(x) \cap \mathcal{I}.$$ △

It can be easily shown that MFCQ is a weaker condition than LICQ. In Peterson [354] several CQs are reviewed and their relationship is analyzed. For the aforementioned CQs the following implications hold:

$$\text{LICQ} \implies \text{MFCQ} \implies \text{ACQ} \implies \text{GCQ}.$$

First–Order Optimality Conditions

In this section, we state first–order necessary conditions for x^* to be a local minimizer for the NLP (3.13). To facilitate the formulation of the necessary conditions we introduce the Lagrange function. The Lagrange function associated to Problem (3.13) is the function $\mathcal{L} : \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^{|\mathcal{E}|} \times \mathbb{R}^{|\mathcal{I}|} \longrightarrow \mathbb{R}$ defined by

$$\mathcal{L}(x, l_0, \lambda, \mu) \stackrel{\text{def}}{=} l_0 \boldsymbol{f}(x) - \sum_{i \in \mathcal{E}} \lambda_i \boldsymbol{c}_i(x) - \sum_{i \in \mathcal{I}} \mu_i \boldsymbol{c}_i(x).$$

A first necessary condition can be provided by Fritz–John type conditions as introduced for Banach spaces in Theorem 3.7. Now we apply the theorem to the NLP (3.13).

Theorem 3.19 (First–Order Necessary Conditions of Fritz–John Type, Finite Case)
Let x^* be a local minimum of Problem (3.13) and S closed and convex with $\operatorname{int}(S) \neq \emptyset$. Then there exist multipliers $l_0 \geqslant 0$, $\lambda \in \mathbb{R}^{|\mathcal{E}|}$ and $\mu \in \mathbb{R}^{|\mathcal{I}|}$ not all zero such that

$$\mathcal{L}'_x(x^*, l_0, \lambda, \mu)(x - x^*) \geqslant \mathbf{0} \quad \forall x \in S, \tag{3.14a}$$
$$\mu_i \cdot \mathbf{c}_i(x^*) = \mathbf{0}, \quad i \in \mathcal{I}, \tag{3.14b}$$
$$\mu_i \leqslant \mathbf{0}, \quad i \in \mathcal{I}. \tag{3.14c}$$

△

Note that $(l_0, \lambda, \mu) = \mathbf{0}$ trivially satisfies the Fritz–John conditions (3.14a)–(3.14c). However, Theorem 3.19 guarantees the existence of a nontrivial vector $(l_0, \lambda, \mu) \neq \mathbf{0}$ to fulfill the Fritz–John conditions. Unfortunately, the case $l_0 = 0$ may occur and this implies that the objective function $\boldsymbol{f}$ does not enter into the Fritz–John conditions.
To overcome the issue of a potential vanishing l_0 one requires additionally the validity of a CQ. Then a Fritz–John point becomes a KKT point and due to the linearity of Lagrange multipliers in the Lagrange function l_0 can be normalized.

Theorem 3.20 (KKT Conditions with MFCQ, Finite Case)
Let the assumptions of Theorem 3.19 be satisfied and let the MFCQ be fulfilled at x^*. Then, the assertions of Theorem 3.19 hold with $l_0 = 1$. △

Theorem 3.21 (KKT Conditions with LICQ, Finite Case)
Let the assumptions of Theorem 3.19 be satisfied and let the LICQ be fulfilled at x^*. Then, the assertions of Theorem 3.19 hold with $l_0 = 1$ and in particular

$$\nabla_x \mathcal{L}(x^*, l_0, \lambda, \mu) = \mathbf{0}.$$

Furthermore, the multipliers λ and μ are unique. △

The condition $\nabla_x \mathcal{L}(x^*, l_0, \lambda, \mu) = \mathbf{0}$ in Theorem 3.21 holds due to the condition $x^* \in \operatorname{int}(S)$ which is required for LICQ to hold. The condition $x^* \in \operatorname{int}(S)$ is trivially satisfied for the choice $S = \mathbb{R}^n$. This important special case is particularly convenient for the design of numerical methods. Hence, we restrict our discussion to this case from now on. We state the KKT conditions assuming the Guignard Constraint Qualification (GCQ) holds and investigate simpler CQs that imply GCQ afterwards. Furthermore, we will discuss the influence of CQs on the multiplier properties.

Theorem 3.22 (KKT Conditions, Finite Case, Karush [268], Kuhn and Tucker [281])
Let $x^* \in \Sigma$ be a local minimizer of Problem (3.13) with $S = \mathbb{R}^n$ such that the GCQ is satisfied in x^*. Then there exist multipliers $\lambda \in \mathbb{R}^{|\mathcal{E}|}$ and $\mu \in \mathbb{R}^{|\mathcal{I}|}$ such that

$$\nabla_x \mathcal{L}(x^*, \lambda, \mu) = \mathbf{0}, \tag{3.15a}$$
$$\mu_i \cdot \mathbf{c}_i(x^*) = \mathbf{0}, \quad i \in \mathcal{I}, \tag{3.15b}$$
$$\mu_i \leqslant \mathbf{0}, \quad i \in \mathcal{I}. \tag{3.15c}$$

△

For a fixed local minimizer $x^* \in \Sigma$, multipliers λ and μ satisfying the KKT conditions (3.15a)–(3.15c) need not necessarily be unique. The following result reveals the relationship between different CQs and the resulting multiplier uniqueness.

Theorem 3.23 (Constraint Qualifications and Uniqueness of Multipliers)
Let $x^* \in \Sigma$ a local minimizer of Problem (3.13). Let $\Lambda \stackrel{\text{def}}{=} \{(\lambda, \mu) \,:\, (x^*, \lambda, \mu) \text{ is KKT point}\}$. Then Λ is

(i) closed and convex.

(ii) not the empty set, if GCQ holds.

(iii) a compact set if and only if MFCQ holds.

(iv) a singleton if LICQ holds. △

Second–Order Conditions

The previous section introduced criteria that characterize the relationship of the objective function gradient and the active constraints at a solution point x^* of Problem (3.13). Roughly speaking, the first–order approximation to the objective function along any vector x from $\mathcal{F}(x^*)$ either increases (i.e., $x^T \nabla \boldsymbol{f}(x^*) > 0$), or keeps this value the same (i.e., $x^T \nabla \boldsymbol{f}(x^*) = 0$). For the directions $x \in \mathcal{F}(x^*)$ for which $x^T \nabla \boldsymbol{f}(x^*) = 0$ we cannot determine from first–order information alone whether the objective function value increases or decreases if we move along x. Hence, it is necessary to take second–order information into account.
An important quantity in the second–order conditions is the critical cone $\mathcal{C}(\Sigma, x^*, \lambda, \mu)$. This is the cone of directions d in the linearized feasible set $\mathcal{F}(\Sigma, x^*)$ for which the KKT conditions alone do not tell us whether the objective function increases along d.

Definition 3.24 (Critical Cone)
Let (x^*, λ, μ) be a KKT point for Problem (3.13). Then the *critical cone* is defined as follows:

$$\mathcal{C}(\Sigma, x^*, \lambda, \mu) \stackrel{\text{def}}{=} \left\{ d \in \mathcal{F}(\Sigma, x^*) \,:\, \nabla \boldsymbol{c}_i(x^*)^T d = 0,\ i \in \mathcal{A}(x^*) \cap \mathcal{I} \text{ with } \mu_i < 0 \right\}. \qquad \triangle$$

From the KKT conditions we have for $d \in \mathcal{C}(\Sigma, x^*, \lambda, \mu)$ that

$$d^T \nabla \boldsymbol{f}(x^*) = \sum_{i \in \mathcal{E}} \lambda_i \cdot d^T \nabla \boldsymbol{c}_i(x^*) + \sum_{i \in \mathcal{I}} \mu_i \cdot d^T \nabla \boldsymbol{c}_i(x^*) = 0.$$

The following theorem gives a second–order necessary condition. It states that that the Hessian of the Lagrangian has non–negative curvature along critical directions at a local solution x^*.

Theorem 3.25 (Second–Order Necessary Conditions)
Let $\boldsymbol{f}$ and $\boldsymbol{c}_i$, $i \in \mathcal{E} \cup \mathcal{I}$ be twice continuously differentiable and let $S = \mathbb{R}^n$. Suppose that x^* is a local solution of NLP (3.13) and that the LICQ is satisfied at x^*. Let furthermore (λ, μ) be Lagrange multipliers such that (x^*, λ, μ) is a KKT point. Then

$$d^T \nabla^2_{xx} \mathcal{L}(x^*, \lambda, \mu) d \geqslant 0, \quad \forall d \in \mathcal{C}(\Sigma, x^*, \lambda, \mu). \tag{3.16}$$

△

Proof See Nocedal and Wright [341, Theorem 12.5]. □

Sufficient conditions are conditions on $\boldsymbol{f}$ and $\boldsymbol{c}_i$, $i \in \mathcal{E} \cup \mathcal{I}$, that guarantee that x^* is a local solution of Problem (3.13). The following result provides a sufficient condition which is similar to the previous introduced necessary condition, but it differs in that the CQ is not required, and the inequality in (3.16) is replaced by a strict inequality.

Theorem 3.26 (Second–Order Sufficient Conditions)
Let f and c_i, $i \in \mathcal{E} \cup \mathcal{I}$ be twice continuously differentiable. Let $S = \mathbb{R}^n$ and let (x^*, λ, μ) be a KKT point of Problem (3.13) with

$$d^T \nabla^2_{xx} \mathcal{L}(x^*, \lambda, \mu) d > 0 \quad \forall d \in \mathcal{C}(\Sigma, x^*, \lambda, \mu). \tag{3.17}$$

Then there exists a neighborhood $\mathcal{U}$ of x^* and some $\alpha > 0$ such that

$$f(x) \geqslant f(x^*) + \alpha \|x - x^*\|^2 \quad \forall x \in \Sigma \cap \mathcal{U}. \qquad \triangle$$

Proof See Nocedal and Wright [341, Theorem 12.6]. □

The conditions (3.16) and (3.17) are unhandy for most problems since they involve the critical cone. Thus, Theorem 3.25 and Theorem 3.26 are often stated in a weaker form: if the strict complementarity condition holds and the KKT multipliers λ and μ are unique, then $\mathcal{C}(\Sigma, x^*, \lambda, \mu)$ can be expressed as the kernel of the matrix whose rows are built of active constraint gradients at x^*, i.e., it holds $\mathcal{C}(\Sigma, x^*, \lambda, \mu) = \ker(A(x^*))$ for the matrix

$$A(x^*) \overset{\text{def}}{=} \left[\nabla c_i(x^*)^T \right]_{i \in \mathcal{A}(x^*)}.$$

If LICQ holds at x^* the matrix $A(x^*)$ has full row rank. Let Z be a matrix with full column rank and let the columns of Z build a basis for $\ker(A(x^*))$, i.e., it holds $A(x^*)\, Z = \mathbf{0}$. Hence, we can write the critical cone as

$$\mathcal{C}(\Sigma, x^*, \lambda, \mu) = \left\{ Z\, d \,:\, d \in \mathbb{R}^{|\mathcal{A}(x^*)|} \right\}.$$

We conclude that the conditions (3.16) and (3.17) are fulfilled if

$$d^T\, Z^T\, \nabla^2_{xx} \mathcal{L}(x^*, \lambda, \mu)\, Z\, d \geqslant 0 \quad \forall d \qquad \text{resp.} \qquad d^T\, Z^T\, \nabla^2_{xx} \mathcal{L}(x^*, \lambda, \mu)\, Z\, d > 0 \quad \forall d$$

holds, i.e., the Hessian $\nabla^2_{xx} \mathcal{L}(x^*, \lambda, \mu)$ is positive semidefinite/positive definite on the null space of the active constraint Jacobian matrix.
In order to determine the matrix Z one usually calculates a QR factorization (see e.g. Stoer et al. [419, Section 6.6.4]) to the matrix $A(x^*)$: we find a square upper triangle matrix R and an orthogonal matrix $Q = [Q_1\, Q_2]$ such that the transpose of $A(x^*)$ can be expressed as follows:

$$A(x^*)^T = [Q_1\, Q_2] \begin{bmatrix} R \\ \mathbf{0} \end{bmatrix} = Q_1 R.$$

In case of a regular matrix R one can choose $Z = Q_2$. Otherwise, one needs to apply a column pivoting procedure during the QR factorization process.

3.6 Numerical Methods

In this section we give a brief introduction how NLPs can be solved numerically. Well known methods are Sequential Quadratic Programming (SQP), interior–point methods and

multiplier–penalty methods. Most textbooks in nonlinear optimization address those methods, see e.g. Fiacco and McCormick [161], Fletcher [166], Geiger and Kanzow [186], Gill et al. [196], or Nocedal and Wright [341].
To achieve convergence from arbitrary starting points the aforementioned methods are usually expanded by globalization strategies such as line–search methods, trust–region methods, or filter methods. All of them have been investigated for many years and assuming certain requirements most of them show at least global convergence to KKT points and locally superlinear convergence.
No method can be taken over any other per se from a purely mathematical point of view. The decision to use one method rather depends on its concrete implementation and in particular on the strategies, how it deals with numerical difficulties such as ill–conditioned problems, bad scaling, sparsity, or warm–start handling.
In this thesis we focus on the Lagrange–Newton method and the SQP method. More extended reference literature for SQP methods is provided by Han [222], Powell [360], Gill et al. [196], Stoer [418], Schittkowski [387, 388]. There exist several implementations of the SQP method, e.g. Schittkowski [389], Philip et al. [355], Gill et al. [197].

3.6.1 Lagrange–Newton Method

In this section we restrict the discussion to purely equality constrained NLPs

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & \boldsymbol{f}(x) \\ \text{s.t.} \quad & \boldsymbol{c}_i(x) = \mathbf{0}, \quad i \in \mathcal{E}, \end{aligned} \tag{3.18}$$

where we make the same assumptions for the objective function $\boldsymbol{f}$ and the constraint functions $\boldsymbol{c}_i$, $i \in \mathcal{E}$ as for our standard NLP problem (3.13). Let us assume that x^* is a local minimum of Problem (3.18) and that the LICQ is satisfied at x^*, i.e., the gradients $\nabla \boldsymbol{c}_i(x^*)$, $i \in \mathcal{E}$, are linearly independent. According to Theorem 3.22 and Theorem 3.23 there exist unique multipliers $\lambda_i^* \in \mathbb{R}$, $i \in \mathcal{E}$, such that

$$\begin{aligned} \mathbf{0} = \nabla_x \mathcal{L}(x^*, \lambda^*) &= \nabla \boldsymbol{f}(x^*) - \sum_{i\in\mathcal{E}} \lambda_i^* \nabla \boldsymbol{c}_i(x^*), \\ 0 = \boldsymbol{c}_i(x^*), & \qquad\qquad i \in \mathcal{E}. \end{aligned}$$

This is a nonlinear equation for x^* and λ^* and can be rewritten as $\boldsymbol{F}(x^*, \lambda^*) = \mathbf{0}$, where the function $\boldsymbol{F} : \mathbb{R}^n \times \mathbb{R}^{|\mathcal{E}|} \longrightarrow \mathbb{R}^{n+|\mathcal{E}|}$ is defined as

$$\boldsymbol{F}(x, \lambda) \overset{\text{def}}{=} \begin{bmatrix} \nabla_x \mathcal{L}(x, \lambda) \\ \boldsymbol{c}(x) \end{bmatrix}. \tag{3.19}$$

The Lagrange–Newton method solves this nonlinear equation with the well known Newton method. In fact, the Lagrange–Newton method tries to find a KKT point of Problem (3.18).
For the sake of completeness we review the well–known convergence results for Newton's method (see e.g. Nocedal and Wright [341, Theorem 11.2]) in terms of the linear equation

Algorithm 1 Lagrange–Newton Method

1: Choose $x^{(0)} \in \mathbb{R}^n$ and $\lambda^{(0)} \in \mathbb{R}^{|\mathcal{E}|}$, $k \leftarrow 0$
2: **while** $\boldsymbol{F}(x^{(k)}, \lambda^{(k)}) \neq \mathbf{0}$ **do**
3: Solve the linear equation

$$\begin{bmatrix} \nabla^2_{xx}\mathcal{L}(x^{(k)}, \lambda^{(k)}) & \boldsymbol{c}'(x^{(k)})^T \\ \boldsymbol{c}'(x^{(k)}) & \mathbf{0} \end{bmatrix} \cdot \begin{bmatrix} \delta_x \\ \delta_\lambda \end{bmatrix} = - \begin{bmatrix} \nabla_x \mathcal{L}(x^{(k)}, \lambda^{(k)}) \\ \boldsymbol{c}(x^{(k)}) \end{bmatrix} \tag{3.20}$$

4: $x^{(k+1)} \leftarrow x^{(k)} + \delta_x$, $\lambda^{(k+1)} \leftarrow \lambda^{(k)} + \delta_\lambda$
5: $k \leftarrow k + 1$
6: **end while**

(3.20).

Theorem 3.27 (Local Convergence of Lagrange–Newton Method)
Let (x^*, λ^*) be a KKT point and let $\boldsymbol{f}$ and $\boldsymbol{c}_i$, $i \in \mathcal{E}$ be twice continuously differentiable with Lipschitz continuous second derivatives. Furthermore, let the matrix

$$\begin{bmatrix} \nabla^2_{xx}\mathcal{L}(x^{(k)}, \lambda^{(k)}) & \boldsymbol{c}'(x^{(k)})^T \\ \boldsymbol{c}'(x^{(k)}) & \mathbf{0} \end{bmatrix} \tag{3.21}$$

be non–singular. Then there exists $\varepsilon > 0$ such that the Lagrange–Newton method converges for all $(x^{(0)}, \lambda^{(0)}) \in \mathcal{U}_\varepsilon(x^*, \lambda^*)$. The algorithm has a quadratic convergence rate, i.e., there exists a constant $C \geqslant 0$ such that

$$\left\|(x^{(k+1)}, \lambda^{(k+1)}) - (x^*, \lambda^*)\right\| \leqslant C \cdot \left\|(x^{(k)}, \lambda^{(k)}) - (x^*, \lambda^*)\right\|^2$$

for all k sufficiently large. △

The matrix (3.21) is called Karush–Kuhn–Tucker–matrix. A sufficient condition for the non–singularity of the KKT–matrix is provided if the gradients $\boldsymbol{c}_i$, $i \in \mathcal{E}$, are linearly independent and it holds

$$d^T \nabla^2_{xx}\mathcal{L}(x^*, \lambda^*)d > 0$$

for all $\mathbf{0} \neq d \in \mathbb{R}^n$ with

$$\boldsymbol{c}'(x^*)\, d = \mathbf{0}.$$

3.6.2 Sequential Quadratic Programming Method

Sequential Quadratic Programming algorithms solve a sequence of quadratic subproblems to generate iterates. Those subproblems are composed of a quadratic model of the objective subject to constraint linearizations of the NLP at the actual iterate. The following section is dedicated to this particular class of optimization problems.

Quadratic Programs

In a Quadratic Program a quadratic function is minimized over a polyhedron. A formal definition of what a Quadratic Program is, is provided below.

Definition 3.28 (Quadratic Program)
Let $a \in \mathbb{R}^{n_x}$, $b \in \mathbb{R}^{n_b}$ and $c \in \mathbb{R}^{n_c}$ be vectors and let $A \in \mathbb{R}^{n_x \times n_x}$, $B \in \mathbb{R}^{n_b \times n_x}$ and $C \in \mathbb{R}^{n_c \times n_x}$ be matrices. A *Quadratic Program (QP)* is a NLP having a quadratic objective function and affine constraint functions, i.e., problems of the following type:

$$\begin{aligned} \min_{x \in \mathbb{R}^{n_x}} \quad & \frac{1}{2} x^T A x + a^T x \\ \text{s.t.} \quad & \mathbf{0}_{n_b} = Bx + b, \\ & \mathbf{0}_{n_c} \geqslant Cx + c. \end{aligned} \tag{3.22}$$

△

A standard characterization of QPs is based on the Hessian matrix A: QP (3.22) is called a *convex QP* if A is positive semidefinite. In case of positive definite A it is called a *strictly convex QP* and for an indefinite matrix A it is called a *non–convex QP*.
In case of affine objective ($A \equiv \mathbf{0}_{n_x \times n_x}$) and affine constraint functions we call the problem a *Linear Program (LP)*. Applying KKT conditions to the QP (3.22) a point (x^*, λ^*, μ^*) is a KKT if the following equations hold:

$$\begin{aligned} \mathbf{0}_{n_x} &= Ax^* + a + B^T \lambda^* + C^T \mu^*, \\ \mathbf{0}_{n_b} &= Bx^* + b, \\ \mathbf{0}_{n_c} &\geqslant Cx^* + c, \\ \mathbf{0}_{n_c} &\leqslant \mu^*, \\ 0 &= \mu_i^* (C_{i,\cdot} x^* + c_i), \quad i \in [n_c]. \end{aligned}$$

Hence, the KKT conditions for pure equality constrained QPs are given by the linear system

$$\begin{bmatrix} A & B^T \\ B & \mathbf{0} \end{bmatrix} \begin{bmatrix} x^* \\ \lambda^* \end{bmatrix} = - \begin{bmatrix} a \\ b \end{bmatrix}.$$

The block matrix on the left side is called the *KKT matrix*. Note that the equality constrained QP has a unique KKT point if the KKT matrix is invertible. The following lemma specifies conditions assuring a nonsingular KKT matrix.

Lemma 3.29 (Invertibility of the KKT Matrix)
Let $A \in \mathbb{R}^{n_x \times n_x}$ be a symmetric matrix and let $B \in \mathbb{R}^{n_b \times n_x}$ be a matrix such that it has full rank and it holds $n_b \leqslant n_x$. Let furthermore A be positive definite on the null space of B. Then the matrix

$$\begin{bmatrix} A & B^T \\ B & \mathbf{0} \end{bmatrix}$$

is invertible. △

Proof See Nocedal and Wright [341, Lemma 16.1]. □

According to Lemma 3.29 the KKT matrix is regular if B has full rank n_b and A is positive definite on the null space of B. Under those assumptions it can be easily verified that the second–order sufficient conditions (see Theorem 3.26) hold at (x^*, λ^*). Hence, x^* is a strict local minimizer of the equality constrained QP.

Similar results towards existence and uniqueness of a KKT point can also be shown for inequality constrained QPs, e.g. under the additional assumptions of a non–empty feasible set and that the combined constraint matrix $[B^T, C^T]$ has full rank $n_b + n_c$. Sufficient conditions for x^* to be a local minimizer are satisfied if (x^*, λ^*, μ^*) is a KKT point and if A is positive definite on the null space of the active constraint Jacobian matrix, i.e., the matrix whose rows comprises the active constraint gradients at x^*. A sufficient condition for x^* to be a global solution of QP (3.22) the point (x^*, λ^*, μ^*) needs to be a KKT point and A needs to be positive semidefinite, cf. Nocedal and Wright [341, Theorem 16.4].

A presentation of the general theory about solution approaches to QPs lies beyond the scope of this thesis and we refer the reader to the textbook of Nocedal and Wright [341, Chapter 16] and the cited references therein. In general, however, the following can be stated: QP solutions can be determined in a finite number of computational steps or the infeasibility of the QP can be shown. The numerical effort to find solutions depends on the number of inequality constraints and the QP type. In case of convex or strictly convex QPs the computational complexity is comparable to the one that is required to solve LPs (see Nocedal and Wright [341, Chapter 13+14]). In case of non–convex QPs the solution process becomes more challenging due to the possible existence of several stationary points, cf. Gould et al. [208]. For non–convex QPs it was even shown that it is $\mathcal{NP}$–hard to decide whether a given feasible point is a global minimizer, cf. Murty and Kabadi [335]. Likewise, it is $\mathcal{NP}$–hard to decide if a given point is a local minimizer, Vavasis [433].

Well–established QP solvers are `Gurobi` [213], CPLEX [248], and `qpOASES` [160].

The Full Step Exact Hessian SQP Method

The Sequential Quadratic Programming (SQP) method can be interpreted as an extension of the Lagrange–Newton method to general nonlinear optimization problems with inequality constraints. To make this clear, we derive the Lagrange–Newton method in a second way. To this end, we consider again the purely equality constrained NLP (3.18) and approximate the problem at some point $(x^{(k)}, \lambda^{(k)})$ by the QP

$$\begin{aligned} \min_{d \in \mathbb{R}^n} \quad & \frac{1}{2} d^T \nabla^2_{xx} \mathcal{L}(x^{(k)}, \lambda^{(k)}) d + \nabla f(x^{(k)})^T d \\ \text{s.t.} \quad & \mathbf{0} = \boldsymbol{c}(x^{(k)}) + \boldsymbol{c}'(x^{(k)}) d. \end{aligned} \tag{3.23}$$

The Lagrange function for QP (3.23) is given by

$$\frac{1}{2} d^T \nabla^2_{xx} \mathcal{L}(x^{(k)}, \lambda^{(k)}) d + f'(x^{(k)}) d - \eta^T \left(\boldsymbol{c}(x^{(k)}) + \boldsymbol{c}'(x^{(k)}) d \right).$$

The KKT conditions from Theorem 3.22 applied to the QP leads to

$$\begin{bmatrix} \nabla^2_{xx}\mathcal{L}(x^{(k)},\lambda^{(k)}) & \boldsymbol{c}'(x^{(k)})^T \\ \boldsymbol{c}'(x^{(k)}) & \mathbf{0} \end{bmatrix} \cdot \begin{bmatrix} d \\ \eta \end{bmatrix} = -\begin{bmatrix} \nabla \boldsymbol{f}(x^{(k)}) \\ \boldsymbol{c}(x^{(k)}) \end{bmatrix}. \tag{3.24}$$

Subtracting $\boldsymbol{c}'(x^{(k)})^T\lambda^{(k)}$ on both sides of the first equation in (3.24) yields

$$\begin{bmatrix} \nabla^2_{xx}\mathcal{L}(x^{(k)},\lambda^{(k)}) & \boldsymbol{c}'(x^{(k)})^T \\ \boldsymbol{c}'(x^{(k)}) & \mathbf{0} \end{bmatrix} \cdot \begin{bmatrix} d \\ \eta - \lambda^{(k)} \end{bmatrix} = -\begin{bmatrix} \nabla_x \mathcal{L}(x^{(k)},\lambda^{(k)}) \\ \boldsymbol{c}(x^{(k)}) \end{bmatrix}. \tag{3.25}$$

A comparison of linear equations (3.20) and (3.25) reveals that they are equivalent if we identify $\delta_x \overset{\text{def}}{=} d$ and $\delta_\lambda \overset{\text{def}}{=} \eta - \lambda^{(k)}$. If we update the states according to line 4 of Algorithm 1 we get

$$x^{(k+1)} = x^{(k)} + \delta_x = x^{(k)} + d, \qquad \lambda^{(k+1)} = \lambda^{(k)} + \delta_\lambda = \eta.$$

We have seen that for purely equality constrained NLPs the Lagrange–Newton method coincides with the just presented method of repeatedly solving QPs, if we use the Lagrange multiplier η of the QP subproblem as new approximation for the multiplier λ.
This observation suggests an obvious extension for NLP (3.13) with $S = \mathbb{R}^n$: the QP (3.23) is augmented with an additional quadratic approximation term for the inequality constraints. The resulting algorithm is summarized in Algorithm 2.

Algorithm 2 Local SQP Method

1: Choose $(x^{(0)},\lambda^{(0)},\mu^{(0)}) \in \mathbb{R}^n \times \mathbb{R}^{|\mathcal{E}|} \times \mathbb{R}^{|\mathcal{I}|}$, $k \leftarrow 0$
2: **while** $(x^{(k)},\lambda^{(k)},\mu^{(k)})$ is not a KKT point of NLP (3.13) with $S = \mathbb{R}^n$ **do**
3: Compute a KKT point $(d^{(k)},\lambda^{(k+1)},\mu^{(k+1)}) \in \mathbb{R}^n \times \mathbb{R}^{|\mathcal{E}|} \times \mathbb{R}^{|\mathcal{I}|}$ of the QP

$$\begin{aligned} \min_{d\in\mathbb{R}^n} \quad & \frac{1}{2} d^T \nabla^2_{xx}\mathcal{L}(x^{(k)},\lambda^{(k)},\mu^{(k)})d + \nabla \boldsymbol{f}(x^{(k)})^T d \\ \text{s.t.} \quad & \mathbf{0} = \boldsymbol{c}_i(x^{(k)}) + \boldsymbol{c}_i'(x^{(k)})d, \quad i \in \mathcal{E}, \\ & \mathbf{0} \geqslant \boldsymbol{c}_i(x^{(k)}) + \boldsymbol{c}_i'(x^{(k)})d, \quad i \in \mathcal{I}. \end{aligned} \tag{3.26}$$

4: $x^{(k+1)} \leftarrow x^{(k)} + d^{(k)}$
5: $k \leftarrow k+1$
6: **end while**

Similar to the Lagrange–Newton method we provide a convergence result for the local SQP method as proposed in Algorithm 2.

Theorem 3.30 (Local Convergence of SQP Method)
Let the following conditions hold:

(i) The point x^* is a local minimum of NLP (3.13) with $S = \mathbb{R}^n$.

(ii) The functions $\boldsymbol{f}$ and $\boldsymbol{c}_i$, $i \in \mathcal{E} \cup \mathcal{I}$ are twice continuously differentiable with Lipschitz continuous second derivatives.

(iii) The LICQ is satisfied at x^*, i.e., the gradients $\nabla \boldsymbol{c}_i(x^*)$, $i \in \mathcal{A}(x^*)$ are linearly independent.

(iv) The strict complementarity condition $\mu_i^* - \boldsymbol{c}_i(x^*) > 0$ holds for all $i \in \mathcal{A}(x^*) \cap \mathcal{I}$.

(v) The inequality

$$d^T \nabla^2_{xx} \mathcal{L}(x^*, \lambda^*, \mu^*) d > 0$$

holds for all $\mathbf{0} \neq d \in \mathbb{R}^n$ with

$$\boldsymbol{c}_i'(x^*)\, d = \mathbf{0}, \quad i \in \mathcal{A}(x^*).$$

Then there exists $\varepsilon > 0$ such that all QPs (3.26) have a locally unique solution $d^{(k)}$ with unique multipliers $\lambda^{(k)}$ and $\mu^{(k)}$ for arbitrary $(x^{(0)}, \lambda^{(0)}, \mu^{(0)}) \in \mathcal{U}_\varepsilon(x^*, \lambda^*, \mu^*)$. Moreover, the sequence $(x^{(k)}, \lambda^{(k)}, \mu^{(k)})$ converges quadratically to (x^*, λ^*, μ^*). △

The proof of the Theorem 3.30 follows from Theorem 3.27 and the results of the following section.

Active Set Determination

In general, there are two established approaches to put SQP methods for a general NLP (3.13) into practice.

IQP Approach The first approach is called IQP (*inequality–constrained QP*) approach and is presented in Algorithm 2, i.e., in each iteration the inequality constrained QP (3.26) is solved to determine new iterates. The multipliers obtained from the QP solver are used to come up with a guess of the optimal active set. To make this clear we assume that (x^*, λ^*, μ^*) denotes a KKT point of NLP (3.13) such that the KKT conditions can be stated as

$$\begin{aligned}
\mathbf{0} &= \nabla \boldsymbol{f}(x^*) + [\nabla \boldsymbol{c}_i(x^*)]_{i\in\mathcal{E}}\ \lambda^* + [\nabla \boldsymbol{c}_i(x^*)]_{i\in\mathcal{I}}\ \mu^*, \\
0 &= \boldsymbol{c}_i(x^*), && i \in \mathcal{E}, \\
0 &\geqslant \boldsymbol{c}_i(x^*), && i \in \mathcal{I}, \\
0 &\leqslant \mu_i^*, && i \in \mathcal{I}, \\
0 &= \mu_i^* \cdot \boldsymbol{c}_i(x^*), && i \in \mathcal{I}.
\end{aligned}$$

Let us assume that QP (3.26) is initialized at the primal KKT point x^* of NLP (3.13) with a Hessian matrix A_0. Denoting the KKT point of the QP by (d_0, λ_0, μ_0), we can write its KKT conditions as

$$\begin{aligned}
\mathbf{0} &= A_0\, d_0 + \nabla \boldsymbol{f}(x^*) + [\nabla \boldsymbol{c}_i(x^*)]_{i\in\mathcal{E}}\ \lambda_0 + [\nabla \boldsymbol{c}_i(x^*)]_{i\in\mathcal{I}}\ \mu_0, \\
0 &= \boldsymbol{c}_i(x^*) + \nabla \boldsymbol{c}_i(x^*)^T\, d_0, && i \in \mathcal{E}, \\
0 &\geqslant \boldsymbol{c}_i(x^*) + \nabla \boldsymbol{c}_i(x^*)^T\, d_0, && i \in \mathcal{I}, \\
0 &\leqslant \mu_{0,i}, && i \in \mathcal{I}, \\
0 &= \mu_{0,i} \cdot \left(\boldsymbol{c}_i(x^*) + \nabla \boldsymbol{c}_i(x^*)^T\, d_0\right), && i \in \mathcal{I}.
\end{aligned}$$

It is obvious that those conditions hold if we choose $(d_0, \lambda_0, \mu_0) = (\mathbf{0}, \lambda^*, \mu^*)$. If we assume additionally that A_0 is positive definite on the null space of the active constraint Jacobian matrix, that point is even a unique local minimizer. Thus, the active constraint set and the multipliers can be determined from just knowing the primal KKT point x^*.

One might expect that Algorithm 2 not just identifies active set and multipliers when initialized at (x^*, λ^*, μ^*) but also in a vicinity thereof. Moreover, if the active set does not change near the KKT point then Algorithm 2 acts like a Newton method for equality–constrained optimization (Newton method applied to (3.19)). In this case the proof of Theorem 3.30 would follow from Theorem 3.27. Indeed, the following theorem provides such a result.

Theorem 3.31 (Robinson [372])
Let the assumptions (i)–(v) of Theorem 3.30 be satisfied. Then it holds that the active set $\mathcal{A}(x_k)$ associated with iterates (x_k, λ_k, μ_k) of Algorithm 2 sufficiently close to (x^*, λ^*, μ^*) coincide with the active set $\mathcal{A}(x^*)$ of NLP (3.13) at x^*. △

The practicability of the IQP approach depends strongly on the computational cost to find a solution for QP (3.26) which can be high, in particular, for large–scale problem instances. It is crucial to constitute warm–start strategies, i.e., to initialize the current QP with solution data obtained from the previous QP.

EQP Approach The second SQP based approach is called EQP (*equality–constrained QP*) approach and is based on the idea to split the active set determination and the calculation of iterates into two separate tasks. One possible realization can be sketched as follows: one sets up a LP by omitting the quadratic term $\frac{1}{2}\, d^T\, \nabla^2_{xx} \mathcal{L}(x^{(k)}, \lambda^{(k)}, \mu^{(k)})\, d$ in QP (3.26) and augments the problem with a trust–region constraint $\|d\|_\infty \leqslant \Delta_k$. The active set of the resulting LP is used as the working set for the current iteration. The SQP step is then calculated by solving an equality constrained QP with that working set.

For further details about both the IQP and the EQP approach we refer the reader to the textbook of Nocedal and Wright [341, Chapter 18] and the references therein.

Approximation of Hessian

Due to numerical aspects there are severe reasons to replace the exact Hessian of the Lagrangian $\nabla^2_{xx} \mathcal{L}(x^{(k)}, \lambda^{(k)}, \mu^{(k)})$ in QP (3.26) by an approximation: one reason is that the Hessian is not known explicitly in most practical applications and the computation of the Hessian by finite differences can be very expensive (see Section 6.1 about numerical derivate generation). Another reason may be a possibly indefinite Hessian what causes issues to find a solution of the QP numerically.

Hence, the Hessian of the Lagrangian in iteration k is replaced by a suitable matrix B_k. The modified BFGS–update formula

$$B_{k+1} = B_k + \frac{q^{(k)} q^{(k)^T}}{q^{(k)^T} s^{(k)}} - \frac{B_k s^{(k)} s^{(k)^T} B_k}{s^{(k)^T} B_k s^{(k)}}, \tag{3.27}$$

where

$$
\begin{aligned}
s^{(k)} &= x^{(k+1)} - x^{(k)}, \\
q^{(k)} &= \theta_k \cdot \eta^{(k)} + (1-\theta_k) \cdot B_k s^{(k)}, \\
\eta^{(k)} &= \nabla_x \mathcal{L}(x^{(k+1)}, \lambda^{(k)}, \mu^{(k)}) - \nabla_x \mathcal{L}(x^{(k)}, \lambda^{(k)}, \mu^{(k)}), \\
\theta_k &= \begin{cases} 1, & \text{if } {s^{(k)}}^T \eta^{(k)} \geqslant 0.2 \cdot {s^{(k)}}^T B_k s^{(k)}, \\ \frac{0.8 \cdot {s^{(k)}}^T B_k s^{(k)}}{{s^{(k)}}^T B_k s^{(k)} - {s^{(k)}}^T \eta^{(k)}}, & \text{otherwise.} \end{cases}
\end{aligned}
\tag{3.28}
$$

was suggested by Powell [360] and is a well established choice for B_k. This update formula ensures a symmetric and positive definite matrix B_{k+1} if B_k was symmetric and positive definite. If we choose $\theta_k = 1$ we retrieve the well–known Broyden–Fletcher–Goldfarb–Shanno (BFGS) update formula which is used in quasi–Newton methods for unconstrained optimization problems, see Broyden [87], Fletcher [165], Goldfarb [199], Shanno [408].
Using the exact Hessian Theorem 3.30 guarantees a quadratic convergence rate for the SQP algorithm. On the other hand we only achieve a super–linear convergence for the modified BFGS update formula, cf. Nocedal and Wright [341].
If NLP (3.13) is large–scale and sparse then the modified BFGS update formula is not the method of choice since it tends to generate dense matrices. We sketch a few alternatives for this case:

- Set $B_k \overset{\text{def}}{=} \boldsymbol{I}_n$ for $k \geqslant 0$, what results in a Sequential Linear Programming (SLP) method and a linear convergence rate.

- Use the regularized exact Hessian

  $$B_k \overset{\text{def}}{=} \nabla^2_{xx} \mathcal{L}(x^{(k)}, \lambda^{(k)}, \mu^{(k)}) + \kappa_k \cdot \boldsymbol{I}_n$$

  with a positive real κ_k such that B_k is guaranteed to be positive definite. This is achieved if κ_k is larger than the modulus of the smallest negative eigenvalue of the Hessian matrix. For more information about how to estimate the sparse Hessian matrix and their eigenvalues the reader is referred to the publications of Betts [62], Coleman et al. [115] and Gerschgorin [193].

- The L–BFGS updating formula (see Nocedal [340]) is based on the BFGS updating formula and the Hessian matrix is constructed from only recent data by means of curvature information. For the sake of saving data storage previous curvature information is omitted.

Globalization of the Local SQP Method

The presented convergence results for the Lagrange–Newton method in Theorem 3.27 and the SQP method in Theorem 3.30 were of local type, i.e., both algorithms converge if the chosen starting values are within some neighborhood of a local minimum. In practice, neither the local minimum nor its neighborhood is known a priori in most cases. We therefore describe one

approach that guarantees the convergence of the methods to a local minimum for any starting value under suitable conditions.
The idea is to determine an appropriate step length $t_k > 0$ in iteration k such that the new iterate $x^{(k+1)}$ is given by the formula

$$x^{(k+1)} = x^{(k)} + t_k \cdot d^{(k)}.$$

Compared to the local SQP method the step length is now obtained by a so–called line search in the direction $d^{(k)}$. The line search is performed with the aid of a suitable penalty function or merit function. These act as a sort of measure function for "improvement" of iteration $x^{(k+1)}$. An iterate improves if either a sufficient decrease of the objective function value or of the total constraint violation is achieved while the respective other value is not substantially declined. Often used merit functions are given by

- the non–differentiable l_1–penalty function (see e.g. Powell [360])

$$l_1(x;\alpha) \stackrel{\text{def}}{=} f(x) + \alpha \cdot \sum_{i\in\mathcal{E}} |c_i(x)| + \alpha \cdot \sum_{i\in\mathcal{I}} \max\{0, c_i(x)\}.$$

- the differentiable augmented Lagrange function (see e.g. Schittkowski [387, 388])

$$\begin{aligned}\mathcal{L}_a(x,\lambda,\mu;\alpha) \stackrel{\text{def}}{=} f(x) &+ \sum_{i\in\mathcal{E}} \left(\lambda_i \cdot c_i(x) + \frac{\alpha}{2} \cdot c_i(x)^2\right) \\ &+ \sum_{i\in\mathcal{I}} \begin{cases} \mu_i \cdot c_i(x) + \frac{\alpha}{2} \cdot c_i(x)^2, & \text{if } \mu_i + \alpha \cdot c_i(x) \geqslant 0, \\ -\frac{\mu_i^2}{2\alpha}, & \text{otherwise.} \end{cases}\end{aligned}$$

Under suitable assumptions both functions are exact penalty functions, i.e., there exists a finite positive parameter α^* such that every local minimum x^* of NLP (3.13) with $S = \mathbb{R}^n$ is also a local minimum of the penalty function for all $\alpha > \alpha^*$.

Algorithm 3 Global SQP Method

1: Choose $(x^{(0)}, \lambda^{(0)}, \mu^{(0)}) \in \mathbb{R}^n \times \mathbb{R}^{|\mathcal{E}|} \times \mathbb{R}^{|\mathcal{I}|}$, $B_0 \in \mathbb{R}^{n\times n}$ symmetric and positive definite, $\alpha > 0$, $\beta \in (0,1)$, $\sigma \in (0,1)$, $k \leftarrow 0$
2: **while** $(x^{(k)}, \lambda^{(k)}, \mu^{(k)})$ is not a KKT point of NLP (3.13) with $S = \mathbb{R}^n$ **do**
3: Compute a KKT point $(d^{(k)}, \lambda^{(k+1)}, \mu^{(k+1)}) \in \mathbb{R}^n \times \mathbb{R}^{|\mathcal{E}|} \times \mathbb{R}^{|\mathcal{I}|}$ of the QP (3.26), with $\nabla^2_{xx}\mathcal{L}(x^{(k)}, \lambda^{(k)}, \mu^{(k)})$ replaced by B_k.
4: Determine a step size $t_k = \max\{\beta^j \ : \ j \geqslant 0\}$ such that

$$l_1(x^{(k)} + t_k \cdot d^{(k)}; \alpha) \leqslant l_1(x^{(k)}; \alpha) + \sigma\, t_k \cdot l_1'(x^{(k)}; d^{(k)}; \alpha).$$

5: Compute B_{k+1} according to (3.27), $x^{(k+1)} \leftarrow x^{(k)} + t_k \cdot d^{(k)}$
6: $k \leftarrow k + 1$
7: **end while**

3.6.3 Other Approaches

Interior–Point Methods

Interior–Point methods are also known as *barrier methods* and represent a common and efficient way to solve NLPs. For an extensive introduction to these algorithms we refer to the textbook of Nocedal and Wright [341, Chapter 19]. In our software package `grc` we also integrated the state–of–the–art interior–point `Ipopt` [444] as numerical NLP solver.
As a first step to interior–Point methods we reformulate the NLP problem by introducing a slack variable $s = [s_i]_{i \in \mathcal{I}}$:

$$\begin{aligned} \min_{x,s} \quad & f(x) \\ \text{s.t.} \quad & c_i(x) = 0, \quad i \in \mathcal{E}, \\ & c_i(x) + s_i = 0, \quad i \in \mathcal{I}. \\ & s \geqslant \mathbf{0}. \end{aligned} \tag{3.29}$$

Interior–Point methods exploit the KKT conditions of Theorem 3.19 (we have $S \in \mathbb{R}^n$):

$$\begin{aligned} \mathbf{0} &= \nabla f(x) - {J_E}^T(x)\, y - {J_I}^T(x) z, \\ \mathbf{0} &= S Z - \mu \mathbf{1}, \\ 0 &= c_i(x), && i \in \mathcal{E}, \\ 0 &= c_i(x) + s_i, && i \in \mathcal{I}, \\ \mathbf{0} &\leqslant s, \ \mathbf{0} \leqslant z, \ 0 = \mu. \end{aligned}$$

Here, $J_E(x)$ and $J_I(x)$ denote the Jacobian matrices of the equality and inequality constraint functions. The matrices S and Z denote the diagonal matrices with diagonal entries s and z, respectively.
The condition $S Z = \mu \mathbf{1}$ with $\mu = 0$ makes the condition hard to solve since combinatorial aspects are introduced (determination of the optimal active set). Interior–point methods circumvent this issue by a parametrization of the equation system with μ as a parameter. Then they solve a sequence of equation systems with strictly positive assignments to μ and drive the parameter iteratively to zero. This explains the word "interior" in the name since $s, z > 0$ is guaranteed throughout the whole solution process. Hence, the iterates are kept in the interior of the admissible area. In the described homotopy, we solve perturbed KKT equation systems.

Penalty Methods

In the penalty approach to NLP, we remove the constraints from the NLP and augment the objective function with a penalization term which penalizes constraint violations. Taking the NLP

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{s.t.} \quad & c_i(x) = 0, \quad i \in \mathcal{E}, \end{aligned} \tag{3.30}$$

and a special quadratic penalization approach we define the penalty function

$$p(x;\mu) \stackrel{\text{def}}{=} f(x) + \frac{\mu}{2} \sum_{i \in \mathcal{E}} c_i^2(x)$$

with the penalization parameter $\mu > 0$. If we drive μ to infinity, the degree of the constraint violation penalization is steadily increased. Penalty methods start with a small value for μ and employ handy μ–adaption–strategies to solve the constrained optimization problem. Further details on penalty methods can be found in the textbook of Nocedal and Wright [341, Chapter 17].

Chapter 4

Mathematical Programs with Vanishing Constraints

In this chapter Mathematical Programs with Vanishing Constraints (MPVCs), a challenging class of NLPs, are investigated. We provide a definition of the problem class as well as a review of the most important properties, in particular stationarity conditions and the lack of CQs. We present existing solution strategies for the problem class. Furthermore we analyze the cohesion between MPVCs and the even more challenging class of MPECs. In fact, it is easy to show that MPVC is a subclass of MPEC. In later chapters we will show that certain discretization strategies of OCPs with explicit as well as implicit switches lead to MPVCs.

A wide range of real–world problems can be naturally modeled by MPVCs. For this reason the problem class has attracted a lot of research interest both theoretically and algorithmically in recent years. The presentation of MPVCs in this chapter is based on the theses of Kirches [272] and Lenders [292] and the references they cited (see Scheel and Scholtes [386], Scholtes [392, 393], Outrata [347, 348], Izmailov and Solodov [252], Achtziger and Kanzow [3], Hoheisel and Kanzow [239, 240, 241], Hoheisel [238], Hoheisel et al. [242], Kanzow and Schwartz [266], Luo et al. [307], Pang and Fukushima [350], Gfrerer [194]). These references provide an excellent overview of MPVCs and MPECs.

4.1 Problem Formulation

In this thesis MPVCs arise after a discretization step of OCPs with explicit and implicit switches. Numerical methods to solve NLPs usually rely on the satisfaction of CQs such as e.g. MFCQ or LICQ. Due to a non–convex feasible set and violation of CQs the problem class of MPVCs is challenging.

Definition 4.1 (MPVC)
Let $\mathcal{E}$ and $\mathcal{I}$ be disjoint index sets with $n_c \overset{\text{def}}{=} |\mathcal{E}|+|\mathcal{I}|$ and w.l.o.g. $\mathcal{E}\cup\mathcal{I} = \{1,\dots,n_c\}$. Let $\boldsymbol{f} : \mathbb{R}^{n_x}\times\mathbb{R}^{n_s} \to \mathbb{R}$, $\boldsymbol{c} : \mathbb{R}^{n_x}\times\mathbb{R}^{n_s} \longrightarrow \mathbb{R}^{n_c}$ and $\boldsymbol{g} : \mathbb{R}^{n_x}\times\mathbb{R}^{n_s} \longrightarrow \mathbb{R}^{n_s}$ be continuously differentiable functions. An NLP of the form

$$\begin{aligned} \min_{x\in\mathbb{R}^{n_x}, s\in\mathbb{R}^{n_s}} \quad & \boldsymbol{f}(x,s) \\ \text{s.t.} \quad & 0 = \boldsymbol{c}_i(x,s), && i\in\mathcal{E}, \\ & 0 \geqslant \boldsymbol{c}_i(x,s), && i\in\mathcal{I}, \\ & 0 \geqslant s_i\cdot\boldsymbol{g}_i(x,s), && i\in[n_s], \\ & \mathbf{0}_{n_s} \geqslant s, \end{aligned} \tag{4.1}$$

is called a *Mathematical Program with Vanishing Constraints*. △

By introducing slack variables, problems with two–sided general constraints of the form

$$\begin{aligned} \min_{x\in\mathbb{R}^{n_x}} \quad & \boldsymbol{f}(x) && && (4.2)\\ \text{s.t.} \quad & 0 = \boldsymbol{c}_i(x), && i\in\mathcal{E},\\ & 0 \geqslant \boldsymbol{c}_i(x), && i\in\mathcal{I},\\ & 0 \geqslant \boldsymbol{g}_i(x)\cdot\boldsymbol{h}_i(x), && i\in[n_s],\\ & 0 \geqslant \boldsymbol{h}_i(x), && i\in[n_s], \end{aligned}$$

as used e.g. in Hoheisel [238] can be equivalently reformulated as a problem of type (4.1).
To point out the meaning of a vanishing constraint we have a closer look at the constraint $0 \geqslant s_i \cdot \boldsymbol{g}_i(x,s)$ in problem (4.1): if $s_i = 0$, the constraint $0 \geqslant s_i \cdot \boldsymbol{g}_i(x,s)$ "vanishes" since it is fulfilled regardless of $\boldsymbol{g}_i(x,s)$. Hence, problem (4.1) can be equivalently written as

$$\begin{aligned} \min_{x\in\mathbb{R}^{n_x},s\in\mathbb{R}^{n_s}} \quad & \boldsymbol{f}(x,s) && && \\ \text{s.t.} \quad & 0 = \boldsymbol{c}_i(x,s), && i\in\mathcal{E},\\ & 0 \geqslant \boldsymbol{c}_i(x,s), && i\in\mathcal{I},\\ & \mathbf{0}_{n_s} \geqslant s, && && \\ & 0 > s_i \quad \Rightarrow \quad 0 \geqslant \boldsymbol{g}_i(x,s), && i\in[n_s]. && (4.3a) \end{aligned}$$

The concept of vanishing constraints can also be found in the context of Mixed Integer Linear Programmings (MILPs). Depending on the value of a binary variable $\omega \in \{0,1\}$ a linear constraint $\alpha \cdot x \leqslant \beta$ is either active or inactive. These constraints are also known as *Indicator Constraints* and can be studied in detail e.g. in Belotti et al. [44].
A mathematical program with logical constraints as (4.3a) is a special case of a so called *Constraint Programming (CP)*. CPs in the context of MILPs are investigated by Achterberg [2].
Due to the product terms $0 \geqslant s_i \cdot \boldsymbol{g}_i(x,s)$, $i \in [n_s]$, MPVCs are non–convex regardless of the curvature types of constraint functions $\boldsymbol{c}$ and $\boldsymbol{g}$. Furthermore, standard CQs like LICQ, MFCQ or even the Abadie Constraint Qualification (ACQ) are not guaranteed to hold.

Lemma 4.2 (Violation of LICQ and MFCQ for MPVCs)
Let $x \in \mathbb{R}^{n_x}$ and $s \in \mathbb{R}^{n_s}$ be feasible for problem (4.1). Then

(i) LICQ is violated in (x,s), if $\{i : s_i = 0\}$ is not the empty set.

(ii) MFCQ is violated in (x,s), if $\{i : s_i = 0 \text{ and } 0 \geqslant \boldsymbol{g}_i(x,s)\}$ is not the empty set. △

Proof See Achtziger and Kanzow [3]. □

Achtziger and Kanzow [3] show that the assumption $\{i : s_i = 0 \text{ and } 0 \geqslant \boldsymbol{g}_i(x,s)\} \neq \emptyset$ is quite reasonable for MPVCs and satisfied for a big class of applications of truss topology optimization. Hence, LICQ and MFCQ are too strong assumptions for MPVCs.

4.2 Comparison with MPECs

In this section another class of optimization problems which is closely related to MPVCs is investigated, namely *Mathematical Programs with Equilibrium Constraints (MPECs)*.

Definition 4.3 (MPEC)
Let $\mathcal{E}$ and $\mathcal{I}$ be disjoint index sets with $n_c \stackrel{\text{def}}{=} |\mathcal{E}| + |\mathcal{I}|$ and w.l.o.g. $\mathcal{E} \cup \mathcal{I} = \{1, \ldots, n_c\}$. Let the functions $\boldsymbol{f} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_s} \times \mathbb{R}^{n_s} \longrightarrow \mathbb{R}$ and $\boldsymbol{c} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_s} \times \longrightarrow \mathbb{R}^{n_c}$ be continuously differentiable. An NLP of the form

$$\begin{aligned} \min_{x \in \mathbb{R}^{n_x}, s \in \mathbb{R}^{n_s}, t \in \mathbb{R}^{n_s}} \quad & \boldsymbol{f}(x,s,t) \\ \text{s.t.} \quad & 0 = \boldsymbol{c}_i(x,s,t), \quad i \in \mathcal{E}, \\ & 0 \geqslant \boldsymbol{c}_i(x,s,t), \quad i \in \mathcal{I}, \\ & 0 \geqslant s \perp t \leqslant 0, \end{aligned} \tag{4.4}$$

is called a *Mathematical Program with Equilibrium Constraints* or *Mathematical Program with Complementarity Constraints*. We use the notation "$0 \geqslant s \perp t \leqslant 0$" for $0 \geqslant s, 0 \geqslant t,\ s^T t = 0$. △

Combinatorial structures on the characteristic constraints imply a non–convexity in the sense of Scholtes [393] for MPECs. As it has already been done for MPVCs in Lemma 4.2, the following result states that for MPECs LICQ and MFCQ are always violated at any feasible point.

Lemma 4.4 (Violation of Standard Constraint Qualifications for MPEC)
Let (x,s,t) be a feasible point for (4.4). Then MFCQ is violated for the NLP formulation of (4.4):

$$\begin{aligned} \min_{x \in \mathbb{R}^{n_x}, s \in \mathbb{R}^{n_s}, t \in \mathbb{R}^{n_s}} \quad & \boldsymbol{f}(x,s,t) \\ \text{s.t.} \quad & 0 = \boldsymbol{c}_i(x,s,t), \quad i \in \mathcal{E}, \\ & 0 \geqslant \boldsymbol{c}_i(x,s,t), \quad i \in \mathcal{I}, \\ & 0 \geqslant s,\ 0 \geqslant t,\ s^T t = 0. \end{aligned} \tag{4.5}$$

△

Proof See Chen et al. [107], Scheel and Scholtes [386]. □

If we compare MPVCs and MPECs regarding CQs the situation is even worse for MPECs. Due to the non–convexity and the violation of standard CQs it is in general more challenging to find MPEC solutions than MPVC solutions.
In principle, an MPVC can be reformulated as an MPEC by introducing slack variables: the problem (4.1) has a solution if and only if the problem

$$\begin{aligned} \min_{x \in \mathbb{R}^{n_x}, s \in \mathbb{R}^{n_s}, t \in \mathbb{R}^{n_s}} \quad & \boldsymbol{f}(x,s) \\ \text{s.t.} \quad & 0 = \boldsymbol{c}_i(x,s), \quad i \in \mathcal{E}, \\ & 0 \geqslant \boldsymbol{c}_i(x,s), \quad i \in \mathcal{I}, \\ & 0 \geqslant \boldsymbol{g}(x,s) + t, \\ & 0 \geqslant s \perp t \leqslant 0, \end{aligned} \tag{4.6}$$

has a solution. One could have the idea to reformulate an MPVC as an MPEC and use a numerical solver from the MPEC machinery.
There are severe reasons not to use this reformulation strategy: First of all, it has been shown e.g. by Achtziger and Kanzow [3] and Achtziger et al. [4] that an MPEC is even more difficult to tackle than an MPVC in many situations. Moreover, the reformulation with slack variables increases the problem dimension. Note also the non–uniqueness of the slack variables, which illustrates that MPVCs are truly a different class of problems than MPECs.

4.3 Towards CQs and Necessary Optimality Conditions

In Section 4.1 we pointed out the violation of standard NLP CQs such as LICQ and MFCQ for MPVCs. This section is dedicated to introduce some MPVC–tailored CQs. With the aid of these CQs we derive necessary optimality conditions for MPVCs.

4.3.1 Tools for MPVC Analysis

We start with a list of index sets, that will be intensively used in the remainder of this chapter. For feasible points x^* of MPVC (4.2) we define the index sets

$$\begin{aligned} I_g &\overset{\text{def}}{=} \{i \in \mathcal{I} : \boldsymbol{c}_i(x^*) = 0\}, \\ I_+ &\overset{\text{def}}{=} \{i : \boldsymbol{h}_i(x^*) > 0\}, \\ I_0 &\overset{\text{def}}{=} \{i : \boldsymbol{h}_i(x^*) = 0\}. \end{aligned} \tag{4.7}$$

The index set I_+ is split into two subsets as

$$\begin{aligned} I_{+0} &\overset{\text{def}}{=} \{i : \boldsymbol{h}_i(x^*) > 0, \boldsymbol{g}_i(x^*) = 0\}, \\ I_{+-} &\overset{\text{def}}{=} \{i : \boldsymbol{h}_i(x^*) > 0, \boldsymbol{g}_i(x^*) < 0\}, \end{aligned} \tag{4.8}$$

and in the same fashion the index set I_0 is split up as

$$\begin{aligned} I_{0+} &\overset{\text{def}}{=} \{i : \boldsymbol{h}_i(x^*) = 0, \boldsymbol{g}_i(x^*) > 0\}, \\ I_{00} &\overset{\text{def}}{=} \{i : \boldsymbol{h}_i(x^*) = 0, \boldsymbol{g}_i(x^*) = 0\}, \\ I_{0-} &\overset{\text{def}}{=} \{i : \boldsymbol{h}_i(x^*) = 0, \boldsymbol{g}_i(x^*) < 0\}. \end{aligned} \tag{4.9}$$

The subscript in (4.7) and the first subscript in (4.8) and (4.9) indicates the sign of $\boldsymbol{h}_i(x^*)$ whereas the second subscript in (4.8) and (4.9) indicates the sign of $\boldsymbol{g}_i(x^*)$. Note furthermore that the index sets in (4.7)–(4.9) depend on the chosen feasible point x^*. For our purposes the value for x^* will be clear from the particular context.

Definition 4.5 (TNLP(x^*))
Let x^* be feasible for MPVC (4.2). The *Tightened Nonlinear Program (TNLP(x^*))* at x^* is defined as

$$\min_{x \in \mathbb{R}^{n_x}} \quad \boldsymbol{f}(x) \tag{TNLP(x^*)}$$

$$
\begin{aligned}
\text{s.t.} \quad & 0 = \boldsymbol{c}_i(x), \quad i \in \mathcal{E}, \\
& 0 \geqslant \boldsymbol{c}_i(x), \quad i \in \mathcal{I}, \\
& 0 = \boldsymbol{h}_i(x), \quad i \in I_{0+} \cup I_{00}, \\
& 0 \leqslant \boldsymbol{h}_i(x), \quad i \in I_{0-} \cup I_{+}, \\
& 0 \geqslant \boldsymbol{g}_i(x), \quad i = 1, \ldots, n_s.
\end{aligned}
$$
△

The reason why problem (TNLP(x^*)) is called tightened is that its feasible set is obviously contained in the feasible set of the original problem. We will use TNLP(x^*) in the following sections to identify MPVC–tailored CQs. For the same purpose one can use a similar tightened NLP in the context of MPECs. The interested reader can find details in the article of Scheel and Scholtes [386].

4.3.2 MPVC–Tailored CQs

We start the section with a definition of the LICQ counterpart for MPVCs and denote it with MPVC Linear Independence Constraint Qualification (MPVC–LICQ).

Definition 4.6 (MPVC–LICQ)
If the gradients

$$
\begin{aligned}
& \nabla \boldsymbol{c}_i(x^*), \quad i \in \mathcal{E}, \\
& \nabla \boldsymbol{c}_i(x^*), \quad i \in I_g, \\
& \nabla \boldsymbol{h}_i(x^*), \quad i \in I_0, \\
& \nabla \boldsymbol{g}_i(x^*), \quad i \in I_{00} \cup I_{+0},
\end{aligned}
$$

are linearly independent at a feasible point x^* of MPVC (4.2) then *MPVC–LICQ* is fulfilled at x^*. △

The following result, which follows immediately from the definitions of LICQ, MPVC–LICQ and TNLP(x^*), shows that MPVC–LICQ is standard LICQ of the problem TNLP(x^*).

Lemma 4.7
Let x^* be feasible for problem (4.2). Then MPVC–LICQ is fulfilled at point x^* if and only if LICQ holds at point x^* for TNLP(x^*). △

Proof Follows from Definitions 3.17, 4.5, and 4.6. □

Lemma 4.7 motivates the following definition of MPVC–MFCQ, which acts as an MPVC analogue of MFCQ.

Definition 4.8 (MPVC–MFCQ)
Let x^* be feasible for problem (4.2). We say that MPVC Mangasarian–Fromowitz Constraint Qualification (MPVC–MFCQ) is fulfilled at point x^* if MFCQ is fulfilled at point x^* for TNLP(x^*). △

The definition of MPVC–MFCQ implies immediately that if MPVC–LICQ holds at a feasible point x^* for (4.2) then MPVC–MFCQ holds at x^*. Now we state an explicit characterization of MPVC–MFCQ. For this reason let x^* be feasible for (4.2). Then MPVC–MFCQ holds at x^* if and only if

$$
\nabla \boldsymbol{c}_i(x^*),\ i \in \mathcal{E} \quad \text{and} \quad \nabla \boldsymbol{h}_i(x^*),\ i \in I_{0+} \cup I_{00}
$$

are linearly independent, and if there exists a vector d such that

$$\begin{aligned}
&0 > \nabla \boldsymbol{c}_i(x^*)^T d, \quad i \in I_g,\\
&0 < \nabla \boldsymbol{h}_i(x^*)^T d, \quad i \in I_{0-},\\
&0 > \nabla \boldsymbol{g}_i(x^*)^T d, \quad i \in I_{+0} \cup I_{00},\\
&0 = \nabla \boldsymbol{c}_i(x^*)^T d, \quad i \in \mathcal{E},\\
&0 = \nabla \boldsymbol{h}_i(x^*)^T d, \quad i \in I_{0+} \cup I_{00}.
\end{aligned}$$

If GCQ holds, then the KKT conditions are necessary optimality conditions for a point to be a local minimizer of a standard NLP. Hence, it is desirable to derive conditions which imply GCQ. Hoheisel [238, Theorem 4.3.2] shows, that MPVC–LICQ implies GCQ. For the definition of the remaining CQs we need to introduce the MPVC–linearized feasibility cone

$$\begin{aligned}
\mathcal{F}_{MPVC}(x^*) \overset{\text{def}}{=} \{d \in \mathbb{R}^n : \; & 0 \geqslant \nabla \boldsymbol{c}_i(x^*)^T d, && i \in I_g,\\
& 0 = \nabla \boldsymbol{c}_i(x^*)^T d, && i \in \mathcal{E},\\
& 0 = \nabla \boldsymbol{h}_i(x^*)^T d, && i \in I_{0+},\\
& 0 \leqslant \nabla \boldsymbol{h}_i(x^*)^T d, && i \in I_{00} \cup I_{0-},\\
& 0 \geqslant \nabla \boldsymbol{g}_i(x^*)^T d, && i \in I_{+0},\\
& 0 \geqslant (\nabla \boldsymbol{h}_i(x^*)^T d)(\nabla \boldsymbol{g}_i(x^*)^T d), && i \in I_{00}\}.
\end{aligned}$$

Note that the only difference between the standard linearized feasibility cone (see Definition 3.15) and $\mathcal{F}_{MPVC}(x^*)$ is adding the quadratic term in the last line of the definition of $\mathcal{F}_{MPVC}(x^*)$. The definition of MPVC Abadie Constraint Qualification (MPVC–ACQ) and MPVC Guignard Constraint Qualification (MPVC–GCQ) is then straightforward.

Definition 4.9
For a feasible point x^* of MPVC (4.1), we say that

(i) **MPVC–GCQ** holds at x^* if $\mathcal{T}(x^*)^- = \mathcal{F}_{MPVC}(x^*)^-$.

(ii) **MPVC–ACQ** holds at x^* if $\mathcal{T}(x^*) = \mathcal{F}_{MPVC}(x^*)$. △

One can show (see Hoheisel [238, p. 38]) that the following implications hold:

$$\text{GCQ} \Longleftarrow \text{MPVC–LICQ} \Longrightarrow \text{MPVC–MFCQ} \Longrightarrow \text{MPVC–ACQ} \Longrightarrow \text{MPVC–GCQ}. \tag{4.10}$$

4.3.3 First–Order Necessary Optimality Conditions

In this section we investigate first–order optimality conditions for MPVCs. We focus on optimality conditions that involve two different types of stationarity: the first one, called *strong stationarity*, will be seen to be equivalent to the KKT conditions. The second one is called *M–stationarity*, which is a weaker condition and holds under milder assumptions such as all MPVC–tailored CQs introduced in the previous section.

Strong Stationarity

We start with the definition of strong stationarity.

Definition 4.10 (Strong Stationarity)
Let x^* be a feasible point for problem (4.2). We call x^* *strongly stationary* if there exist Lagrange multipliers $(\lambda, \mu, \eta^g, \eta^h) \in \mathbb{R}^{|\mathcal{I}|} \times \mathbb{R}^{|\mathcal{E}|} \times \mathbb{R}^{n_s} \times \mathbb{R}^{n_s}$ such that

$$0 = \nabla f(x^*) + \sum_{i \in \mathcal{I}} \lambda_i \nabla c_i(x^*) + \sum_{i \in \mathcal{E}} \mu_i \nabla c_i(x^*) - \sum_{i=1}^{n_s} \eta_i^h \nabla h_i(x^*) + \sum_{i=1}^{n_s} \eta_i^g \nabla g_i(x^*) \tag{4.11}$$

and

$$\begin{aligned} &0 = h_i(x^*), \quad i \in \mathcal{E}, \\ &0 \geqslant g_i(x^*), \quad i \in \mathcal{I}, \qquad \lambda_i \geqslant 0,\ \lambda_i g_i(x^*) = 0, \quad i \in \mathcal{I}, \\ &\qquad\qquad \eta_i^h = 0\ (i \in I_+),\ \eta_i^h \geqslant 0\ (i \in I_{00} \cup I_{0-}),\ \eta_i^h \text{ free } (i \in I_{0+}), \\ &\qquad\qquad \eta_i^g = 0\ (i \in I_0 \cup I_{+-}),\ \eta_i^g \geqslant 0\ (i \in I_{+0}). \end{aligned} \tag{4.12}$$

△

Strong stationarity is derived from the KKT conditions of the MPVC (4.2). Achtziger and Kanzow [3] have shown that a feasible point x^* of problem (4.2) is strongly stationary if and only if it is a KKT point. Due to the equivalence of KKT conditions and strong stationarity we conclude that strong stationarity is a necessary optimality criterion for an MPVC, if there holds a CQ which implies GCQ.

Theorem 4.11
Let x^* be a local minimizer of MPVC (4.2) such that GCQ is fulfilled at x^*. Then x^* is a strongly stationary point for (4.2). △

Proof See e.g. Hoheisel [238, Proposition 6.1.3]. □

From the previous section, where we reviewed the result that MPVC–LICQ implies GCQ, we conclude immediately:

Corollary 4.12
Let x^* be a local minimizer of MPVC (4.2) such that MPVC–LICQ is fulfilled at x^*. Then x^* is a strongly stationary point for problem (4.2) with unique multipliers $(\lambda, \mu, \eta^g, \eta^h)$ such that (4.11) and (4.12) hold. △

M–Stationarity

Since most standard CQs, apart from GCQ, are too strong for MPVCs, the tailored MPVC–CQs were introduced. However, apart from MPVC–LICQ none of these implies GCQ. Consequently, the conditions from Definition 4.10 cannot be expected to be necessary optimality conditions if one of those holds. Thus, it is important to find necessary optimality conditions that hold under MPVC–GCQ.

Theorem 4.13
Let x^* be a local minimizer for problem (4.2) such that MPVC–GCQ is fulfilled. Then there exist multipliers $(\lambda, \mu, \eta^g, \eta^h)$ such that

$$0 = \nabla f(x^*) + \sum_{i \in \mathcal{I}} \lambda_i \nabla c_i(x^*) + \sum_{i \in \mathcal{E}} \mu_i \nabla c_i(x^*) - \sum_{i=1}^{n_s} \eta_i^h \nabla h_i(x^*) + \sum_{i=1}^{n_s} \eta_i^g \nabla g_i(x^*) \tag{4.13}$$

and

$$
\begin{aligned}
0 \geqslant \boldsymbol{g}_i(x^*), \quad i \in \mathcal{I}, \qquad & 0 \leqslant \lambda_i,\ \lambda_i \boldsymbol{g}_i(x^*) = 0, \quad i \in \mathcal{I}, \\
& 0 = \eta_i^h\ (i \in I_+),\ \eta_i^h \geqslant 0\ (i \in I_{0-}),\ \eta_i^h \text{ free } (i \in I_{0+}), \\
& 0 = \eta_i^g\ (i \in I_{+-} \cup I_{0-} \cup I_{0+}),\ \eta_i^g \geqslant 0\ (i \in I_{+0} \cup I_{00}), \\
& 0 = \eta_i^h \cdot \eta_i^g\ (i \in I_{00}).
\end{aligned} \tag{4.14}
$$

△

Proof See Hoheisel [238]. □

We call the conditions (4.13) and (4.14) *M–stationarity conditions* of an MPVC. This is due to an analogous terminology for MPECs that was introduced by Scholtes [391]. They are slightly weaker than the strong stationarity conditions (4.11) and (4.12) from Definition 4.10. This can be easily seen since for strong stationarity it is required to hold $\eta_i^h \geqslant 0$ and $\eta_i^g = 0$ for all $i \in I_{00}$ whereas we just have $\eta_i^g \geqslant 0$ and $\eta_i^h \cdot \eta_i^g = 0$ for all $i \in I_{00}$ in the case of M–stationarity. Note that M– and strong stationary are identical as soon as I_{00} is the empty set.

Definition 4.14 (M–Stationarity)
Let x^* be feasible for problem (4.2). Then we say that x^* is *M–stationary* if there exist multipliers $(\lambda, \mu, \eta^g, \eta^h)$ such that

$$
0 = \nabla \boldsymbol{f}(x^*) + \sum_{i \in \mathcal{I}} \lambda_i \nabla \boldsymbol{c}_i(x^*) + \sum_{i \in \mathcal{E}} \mu_i \nabla \boldsymbol{c}_i(x^*) - \sum_{i=1}^{n_s} \eta_i^h \nabla \boldsymbol{h}_i(x^*) + \sum_{i=1}^{n_s} \eta_i^g \nabla \boldsymbol{g}_i(x^*)
$$

and

$$
\begin{aligned}
0 = \boldsymbol{h}_i(x^*), \quad & i \in \mathcal{E}, \\
0 \geqslant \boldsymbol{g}_i(x^*), \quad & i \in \mathcal{I}, \qquad 0 \leqslant \lambda_i,\ \lambda_i \boldsymbol{g}_i(x^*) = 0, \quad i \in \mathcal{I}, \\
& \qquad\quad 0 = \eta_i^h\ (i \in I_+),\ \eta_i^h \geqslant 0\ (i \in I_{0-}),\ \eta_i^h \text{ free } (i \in I_{0+}), \\
& \qquad\quad 0 = \eta_i^g\ (i \in I_{+-} \cup I_{0-} \cup I_{0+}),\ \eta_i^g \geqslant 0\ (i \in I_{+0} \cup I_{00}), \\
& \qquad\quad 0 = \eta_i^h \cdot \eta_i^g\ (i \in I_{00}).
\end{aligned} \tag{4.15}
$$

△

Since MPVC–GCQ is the weakest (see (4.10)) of the MPVC–tailored constraint qualifications, M–stationarity becomes a necessary optimality condition in the presence of any of these CQs.

Corollary 4.15
Let x^* be a local minimizer of MPVC (4.2) such that either MPVC–ACQ or MPVC–MFCQ is fulfilled. Then x^* is M–stationary. △

4.4 Numerical Approaches

MPECs arise in various branches of applied sciences with applications in chemical engineering (see Baumrucker et al. [34]), optimal shape design (see Haslinger and Neittaanmäki [225]), economics (see Murphy et al. [334]), automotive engineering (see Kirches et al. [273]), or medicine (see Hatz [227]). Their importance is also backed by the growing number of problem instance suites, cf. Leyffer [293], Dirkse [135].

In the literature there have been developed a vast number of algorithms to solve MPECs. The referenced algorithms are taken from survey [129], monographs [305, 346], and theses written in our group [272, 227, 292]. These algorithms fall into certain categories whose principles are briefly reviewed in the following.

4.4.1 Structural Constraint Approach

Most approaches to solve MPECs numerically rely on reformulating the complementarity constraint as a nonlinear equation. However, the structural constraint approach considers the complementarity constraint of a MPEC as a structural constraint.
Scholtes [393] suggests a problem formulation that decouples the two intrinsic aspects of a MPEC, namely the nonlinear aspect and the combinatorial aspect. Both aspects are then treated separately by a suitable generalization of the well–established Lagrangian framework. He proposes a SQPEC method where the complementarity constraint is passed to the quadratic subproblem as a structural constraint. However, Leyffer and Munson [295] point out that Scholtes's method can also converge to spurious stationary points. That is why they propose a Filter SLPEC method that converges globally to B–stationary points (see Luo et al. [306]). In order to obtain an active set estimate they first solve a linear program with complementarity constraints inside a trust region. With fixed active constraints they then solve an equality–constrained QP. A three–dimensional filter separating constraint violations into complementarity constraint contributions and general constraint contributions is used to achieve global convergence. A method similar in fashion was proposed by Lenders [292] who carries over the SLEQP method of Nocedal and Waltz [94, 95] to MPEC. Under certain assumptions he could establish global convergence to B–stationary points.
Kirches [272, 275] introduces a SQPVC method where he uses a non–convex parametric active set method to solve the arising QPVC subproblem. Assuming MPVC–LICQ convergence to strong stationary points of this subproblem can be ensured. Benko and Gfrerer [50] present a SQPEC method where a QP with linear complementarity constraints is solved in each iteration. An active set method is proposed that finds at least a strongly M–stationary point of this subproblem. The sequence of iterates is obtained by reducing a merit function. Under certain boundedness assumptions it can be ensured that limit points of the method are at least M–stationary. A similar approach (see Benko and Gfrerer [51]) was analyzed in a MPVC context. Under additional assumptions it provides limit points that are stationary in an even stronger sense than M–stationary points.

4.4.2 Nonlinear Equation Approach

In general, the nonlinear equation approach, does not handle the complementarity constraint $0 \geqslant s \perp t \leqslant 0$ as a structural constraint but as nonlinear equation, i.e., it is reformulated e.g. as $0 \geqslant s, 0 \geqslant t, s^T t \leqslant 0$.
In the following, we sketch several solution techniques. Bard [30] reformulated a bilevel optimization problem as a problem subject to complementarity constraints and applied a branch and bound scheme to solve it. A branch and bound method seeking the global minimum of a MPEC is described in Muu and Oettli [336]. Izmailov et al. [253] present a lifting approach to

MPEC resulting in a semi–smooth equation system that can be solved by suitable non–smooth methods. Since certain subclasses of the nonlinear equation approach have an importance of their own we dedicate separate sections to them.

NLP Approach

As a first guess one might think to find solutions to NLPs arising from the nonlinear equation approach by an application of a standard NLP solver. However, constraint qualifications such as LICQ or MFCQ are not satisfied by the NLP, cf. Chen and Florian [109]. Since convergence proofs for NLP algorithms usually ensure convergence only if constraint qualifications hold one cannot expect iterates to converge towards a local optimum. Indeed, a violation of MFCQ causes unbounded multiplier sets, nonexistence of central paths, or inconsistent NLP linearizations arbitrary close to the solution leading to ill–conditioned or unbounded subproblems. Accordingly, early experiments (see Bard [30]) showed a poor performance. By contrast, Fletcher and Leyffer [167] could solve a large class of MPECs in a reliable and efficient way by employing a SQP solver. The theoretical justification for this benevolent behavior was provided by Fletcher et al. [168] who could show a local superlinear convergence to strongly stationary points of SQP methods under certain assumptions.
Benson et al. [54] suggest a filter–based interior–point method. Numerical test prompt that this approach seems to be superior to approaches based on the usage of merit functions.

Piecewise SQP Approach

Piecewise SQP methods can be considered as canonic extensions of the standard SQP method for NLPs to solve MPECs. Hence, a QP with linear complementarity constraints is solved per iteration.
In case of multiplier uniqueness but without requiring the strict complementarity condition, Luo et al. [307] could show local superlinear convergence for a piecewise SQP method. For the sake of enhancing the piecewise SQP approach, Zhang and Liu [470] use an extreme point technique to find better search directions.

Bundle Approach

The bundle approach to solve MPECs makes use of the non–smooth calculus to derive optimality conditions, i.e., the mixed complementarity problem is considered as a semi–smooth system of equations that is solved by suitable algorithms.
Dirkse and Ferris [136] employ a Newton type method to the non–smooth equations, present tailored path–following and pathsearch damping techniques for the method, and derive a global convergence result. Arising linear complementarity problems are solved by simplex–like pivotal algorithms requiring a direct factorization and rank–1 updates. Hence, they are of low practicability when applied to general large–scale problems. In contrast, Munson et al. [333] use a semi–smooth algorithm in which a single linear system must be evaluated per iteration in terms of the Newton system. An approach applying a semi–smooth Newton method to MPECs with equilibria defined by implicit complementarity problems can be found in Outrata et al. [346].

Penalty Function Approach

Penalty function methods rely on the nonlinear equation form of a MPEC. In the style of common penalization techniques the complementarity constraints are dropped from the MPEC. Instead, the MPEC objective function is augmented with them multiplied by a penalty coefficient, i.e., the complementarity constraints become soft constraints.

Luo et al. [305, Chapter 6.1] describe a penalty interior–point algorithm. The method combines interior–point and SQP aspects in the sense that it calculates iterates such that $0 > s$, $0 > t$ and solves a quadratic direction–finding problem. The step size is determined by a backtracking line–search approach. A sufficient reduction in a quadratic penalty function enforces global convergence. Under considerable assumptions the authors obtained a convergence result. Leyffer [294] provides a simple example where the algorithm does not converge to a stationary point. Leyffer et al. [296] propose an interior–penalty method whose global convergence to strongly stationary points is established under common assumptions.

Hu and Ralph [244] employ the convergence analysis of a smoothing approach and a regularization approach in order to derive convergence of iterates, which are generated by a penalty framework for MPECs, to B–stationary points. Scholtes and Stöhr [394] investigate an exact penalization approach to MPECs. Then, they derive a global convergence result for a trust region optimization algorithm applied to a function space that includes exact penalty functions arising from MPECs.

Benson et al. [55] combine an interior–point NLP method and a penalty approach in order to bound the optimal multipliers. The penalty method is realized with a ℓ_1 as well as a ℓ_∞ penalty function. Under certain regularity assumptions Anitescu et al. [15] show convergence to C–stationary points of iterates generated by an approach penalizing the complementarity constraints. Arising QP subproblems are constructed in a way to make them feasible by relaxing certain constraints.

Smoothing–Regularization Approach

For smoothing–regularization methods one incorporates Nonlinear Complementarity Problem (NCP) functions, i.e., (non–smooth) functions $\boldsymbol{\phi}\colon \mathbb{R}^2 \to \mathbb{R}$ satisfying $\boldsymbol{\phi}\ (s,t) = 0$ if and only if $0 \leqslant s$, $0 \leqslant t$, $s \cdot t = 0$. MPEC complementarity constraints are replaced with NCP function formulations. The resulting problem is embedded into a parameterized family of smooth NLPs in the sense that $\boldsymbol{\phi}\ (\cdot)$ is replaced with smooth parametric functions $\boldsymbol{\phi}^\tau\colon \mathbb{R}^2 \to \mathbb{R}$. They are constructed such that $\boldsymbol{\phi}^\tau \to \boldsymbol{\phi}$ as $\tau \to 0$ aiming to end up with a set of NLPs that satisfy constraint qualifications. Hence, in order to determine MPEC solution approximations one generates iterates either by solving a single NLP while driving τ to zero or by solving a sequence of NLPs and a reduction of τ in between NLP solution steps.

Facchinei et al. [151] could show convergence to a global stationary point of the MPEC problem if the iterates are comprised of global stationary points of the smooth problems. Scholtes [391] establishes convergence to C–stationary points under MPEC–LICQ. Under the additional assumption of second order necessary conditions he could show M–stationarity of accumulation points. Finally, adding a certain nondegeneracy assumption, namely the lower–level strict complementarity, ensures B–stationarity. Convergence to B–stationary could also be

established by Fukushima et al. [175] without the need for the aforementioned strict complementarity conditions but under a certain weak nondegeneracy assumption.
As opposed to the aforementioned results, where the considered algorithms are conceptual and global convergence is not established for an implementable method, Fukushima and Tseng [174] propose an implementable ε–active set algorithm. If uniform LICQ holds for the ε–feasible they show global convergence to B–stationary points.
Hoheisel et al. [242] compare several state of the art relaxation methods for MPECs in terms of satisfaction of constraint qualifications and their numerical performance when applied to a common test problem suite. Moreover, convergence results of some algorithms are improved.
Hoheisel [238] analyzes an algorithm very similar in nature to the one of Fukushima et al. [175] within a MPVC setting. However, convergence to a B–stationary point can be established without the need for the second–order condition and under a weaker LICQ type assumption. Likewise, comparing the results with the ones of Scholtes [391] there is also no need for the lower–level strict complementarity assumption. Regarding convergence results of numerical approaches to MPVC problems as a particular class of MPEC problems Hoheisel concludes that stronger results hold under weaker assumptions.
An interior–point method in which τ is driven to zero in conjunction with the barrier parameter is described in Raghunathan and Biegler [365]. An approach relaxing both the complementarity and the nonnegativity constraints is described by DeMiguel et al. [128]. The relaxation parameter is driven to zero in such a way that the arising problems have a strictly feasible interior what makes interior–point point methods applicable.
Stein [416] considers a slightly different setting involving a smooth function $\boldsymbol{\phi}(\cdot)$ such as $\boldsymbol{\phi}_{\mathrm{FB}}^2(\cdot)$ where $\boldsymbol{\phi}_{\mathrm{FB}}(s,t) \overset{\text{def}}{=} s + t - \sqrt{s^2 + t^2}$ denotes the well–known Fischer–Burmeister function. The resulting lack of LICQ at the origin is then overcome by lifting the set $\{(s,t) \in \mathbb{R}^2 : \boldsymbol{\phi}(s,t) = 0\}$ into a higher dimension. This is achieved by reformulating it in terms of an orthogonal projection of a certain smooth set in $\mathbb{R}^3$. Another lifting method applicable to bilevel optimization problems was proposed by Hatz [227]. Inequality constraints of the lower–level problem are lifted which results in a MPEC fulfilling MPEC–LICQ. Algorithms applied to those problems are expect to show good convergence behavior.

Chapter 5

Theory of Optimal Control Problems

This chapter addresses a class of continuous OCPs, interprets it as an infinite dimensional optimization problem and derives first–order optimality conditions. Finally, an overview of common solution strategies for the problem class is presented.
The first section introduces a sufficiently general class of OCPs such that we have recourse to this problem class in the remainder of this thesis. Besides giving some notations we show potential extensions that fit into the aforementioned problem class.
The next section is dedicated to the interpretation of the OCP class as an optimization problem in Banach spaces.
Based on results presented in Section 3.3 about first–order necessary conditions in the Banach space context the third section derives necessary optimality conditions for our OCP class.
The two final sections sketch the two main solution approaches for OCPs, namely the indirect approach and the direct approach. Moreover, we balance pros and cons of both approaches and give an outline of the strategies pursued in this thesis.

5.1 Continuous OCPs

Let $\mathcal{T} = [t_\mathrm{s}, t_\mathrm{f}] \subset \mathbb{R}$ be a compact and non–empty time interval with $t_\mathrm{s} < t_\mathrm{f}$. For the most part of the thesis both t_s and t_f are assumed to be fix. In case t_s or t_f are variable this is emphasized in the particular context. Let

$$
\begin{aligned}
\boldsymbol{\varphi} &: \mathbb{R} \times \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R} \times \mathbb{R}^{n_\mathrm{x}} \longrightarrow \mathbb{R}, \\
\boldsymbol{\psi} &: \mathcal{T} \times \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{u}} \longrightarrow \mathbb{R}, \\
\boldsymbol{f} &: \mathcal{T} \times \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{u}} \longrightarrow \mathbb{R}^{n_\mathrm{x}}, \\
\boldsymbol{c} &: \mathcal{T} \times \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{u}} \longrightarrow \mathbb{R}^{n_\mathrm{c}}, \\
\boldsymbol{r} &: \mathbb{R} \times \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R} \times \mathbb{R}^{n_\mathrm{x}} \longrightarrow \mathbb{R}^{n_\mathrm{r}}
\end{aligned}
$$

be sufficiently smooth functions.

Definition 5.1 (Continuous Optimal Control Problem in Standard Form)
A *continuous optimal control problem* is a constrained infinite–dimensional optimization problem which minimizes a *cost functional* (5.1a) with the *dynamic process* $\boldsymbol{x} \in W^{1,\infty}(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}})$, described by a system of ODEs (5.1b) with right–hand side function $\boldsymbol{f}$ and affected by a *control* $\boldsymbol{u} \in L^\infty(\mathcal{T}, \mathbb{R}^{n_\mathrm{u}})$ over the *horizon* interval $\mathcal{T}$ such that the *mixed control–state constraints* (5.1c) and the *boundary conditions* (5.1d) are satisfied:

$$
\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)} \quad J(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)) \overset{\text{def}}{=} \boldsymbol{\varphi}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})) + \int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{\psi}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) \,\mathrm{d}t \tag{5.1a}
$$

$$\text{s.t.} \qquad \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), \qquad t \in \mathcal{T}, \tag{5.1b}$$

$$\mathbf{0}_{n_c} \geqslant \boldsymbol{c}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), \qquad t \in \mathcal{T}, \tag{5.1c}$$

$$\mathbf{0}_{n_r} = \boldsymbol{r}(t_s, \boldsymbol{x}(t_s), t_f, \boldsymbol{x}(t_f)). \tag{5.1d}$$

△

Due to the unknowns, namely the control trajectory $\boldsymbol{u}(\cdot)$ and the controlled state trajectory $\boldsymbol{x}(\cdot)$, the OCP (5.1) is an infinite dimensional problem. Note, that the path constraint formulation (5.1c) covers pure state and pure control path constraints as well as those of mixed type.

There is the possibility to transfer constraints to the objective as an additional penalization term. These constraints are then called *soft constraints*. This does not guarantee that they are satisfied in the solution but usually they hold with little violation. One normally implements this for constraints that are not crucial for the process dynamics but prevent undesirable behavior.

A pair $(\boldsymbol{x}, \boldsymbol{u}) \in W^{1,\infty}(\mathcal{T}, \mathbb{R}^{n_x}) \times L^{\infty}(\mathcal{T}, \mathbb{R}^{n_u})$ is called *admissible* or *feasible* for the OCP (5.1), if it fulfills the constraints (5.1b)–(5.1d). An admissible pair

$$(\boldsymbol{x}^*, \boldsymbol{u}^*) \in W^{1,\infty}(\mathcal{T}, \mathbb{R}^{n_x}) \times L^{\infty}(\mathcal{T}, \mathbb{R}^{n_u}) \tag{5.2}$$

is called a *weak local minimum* of problem (5.1), if there exists an $\varepsilon > 0$ such that

$$J(\boldsymbol{x}^*, \boldsymbol{u}^*) \leqslant J(\boldsymbol{x}, \boldsymbol{u}) \tag{5.3}$$

holds for all $(\boldsymbol{x}, \boldsymbol{u})$ with $\|\boldsymbol{x} - \boldsymbol{x}^*\|_{1,\infty} < \varepsilon$ and $\|\boldsymbol{u} - \boldsymbol{u}^*\|_{\infty} < \varepsilon$. An admissible pair (5.2) is called *strong local minimum* of problem (5.1), if there exists an $\varepsilon > 0$ such that (5.3) holds for all admissible $(\boldsymbol{x}, \boldsymbol{u})$ with $\|\boldsymbol{x} - \boldsymbol{x}^*\|_{\infty} < \varepsilon$. Note that all strong local minima are also weak local minima, but the converse is not true.

The horizon endpoints t_s and t_f in Problem (5.1) can either be fixed or unknown. In the latter case they are additional optimization variables. If t_s is not fixed, then t_s is called *free initial time* and Problem (5.1) is an OCP with free initial time. If t_f is not fixed, then t_f is called *free final time* and Problem (5.1) is an OCP with free final time.

Problem (5.1) is called *autonomous* if the functions φ, ψ, $\boldsymbol{f}$, $\boldsymbol{c}$ and $\boldsymbol{r}$ do not explicitly depend on the time, otherwise it is called *non–autonomous*.

In the remainder of this section we will present some transformation techniques. On the one hand they possibly allow us to write Problem (5.1) even more compactly. For example, it is shown that every non–autonomous OCP can be transformed into an equivalent autonomous one by augmenting the state vector. On the other hand ostensibly more complex problems such as OCPs with global parameters can be equivalently written in standard form (5.1).

Such a transformation technique is possibly accompanied with an increased problem size or may introduce additional nonlinearities. Hence, tailored numerical solution methods for the original problem may be the preferred choice.

Transformation to Autonomous Problem

A non–autonomous Problem (5.1) can be transformed into an equivalent autonomous problem by introducing an additional state $\boldsymbol{t}(\cdot)$ according to the IVP

$$\dot{\boldsymbol{t}}(t) = 1, \quad \boldsymbol{t}(t_\mathrm{s}) = t_\mathrm{s}. \tag{5.4}$$

By replacing the function argument t in $\boldsymbol{\psi}$, $\boldsymbol{f}$ and $\boldsymbol{c}$ with $\boldsymbol{t}(t)$ and adding (5.4) to Problem (5.1) we end up with the *autonomous* problem

$$\begin{aligned}
\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)} \quad & \boldsymbol{\varphi}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})) + \int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{\psi}(\boldsymbol{t}(t), \boldsymbol{x}(t), \boldsymbol{u}(t)) \,\mathrm{d}t \\
\text{s.t.} \quad & \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{t}(t), \boldsymbol{x}(t), \boldsymbol{u}(t)), \quad t \in \mathcal{T}, \\
& \dot{\boldsymbol{t}}(t) = 1, \quad t \in \mathcal{T}, \\
& \mathbf{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(\boldsymbol{t}(t), \boldsymbol{x}(t), \boldsymbol{u}(t)), \quad t \in \mathcal{T}, \\
& \mathbf{0}_{n_\mathrm{r}} = \boldsymbol{r}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})), \\
& \boldsymbol{t}(t_\mathrm{s}) = t_\mathrm{s}.
\end{aligned}$$

Objective Functions

The performance index

$$\boldsymbol{\varphi}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})) + \int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{\psi}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) \,\mathrm{d}t$$

in (5.1a) is given in the so called Bolza form of an objective functional. The Bolza form is a combination of

- the Mayer type form objective $\boldsymbol{\varphi}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f}))$, which is a mixed start– and end–point contribution, and
- the Lagrange type form objective, which is an integral contribution of the states and controls with integrand $\boldsymbol{\psi}(t, \boldsymbol{x}(t), \boldsymbol{u}(t))$ evaluated on the time horizon $\mathcal{T}$.

By introducing additional states, differentiation and integration Bolza, Mayer and Lagrange type functionals can be transformed into each other.

Global Parameters

A vector of model parameters $p \in \mathbb{R}^{n_\mathrm{p}}$ may enter the objective function $\boldsymbol{\psi}$, the ODE function $\boldsymbol{f}$ or the constraint function $\boldsymbol{c}$ in problem (5.1) as $\boldsymbol{\psi}(\cdot, \boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot), p)$, $\boldsymbol{f}(\cdot, \boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot), p)$ and $\boldsymbol{c}(\cdot, \boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot), p)$. Model parameters can be handled in different ways: we can either treat them as additional controls which are constant over the complete time horizon $\mathcal{T}$, or we can handle them similar to (5.4) by introducing the ODE equations

$$\dot{\boldsymbol{p}}(t) = 0, \quad \boldsymbol{p}(t_\mathrm{s}) = p,$$

and add them to problem (5.1). This results then in the problem

$$\begin{aligned}
\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),p} \quad & \boldsymbol{\varphi}(t_\mathrm{s},\boldsymbol{x}(t_\mathrm{s}),t_\mathrm{f},\boldsymbol{x}(t_\mathrm{f}))+\int_{t_\mathrm{s}}^{t_\mathrm{f}}\boldsymbol{\psi}(t,\boldsymbol{x}(t),\boldsymbol{u}(t),\boldsymbol{p}(t))\,\mathrm{d}t \\
\text{s.t.} \quad & \dot{\boldsymbol{x}}(t)=\boldsymbol{f}(t,\boldsymbol{x}(t),\boldsymbol{u}(t),\boldsymbol{p}(t)), \quad t\in\mathcal{T}, \\
& \dot{\boldsymbol{p}}(t)=0, \quad t\in\mathcal{T}, \\
& \mathbf{0}_{n_\mathrm{c}}\geqslant\boldsymbol{c}(t,\boldsymbol{x}(t),\boldsymbol{u}(t),\boldsymbol{p}(t)), \quad t\in\mathcal{T}, \\
& \mathbf{0}_{n_\mathrm{r}}=\boldsymbol{r}(t_\mathrm{s},\boldsymbol{x}(t_\mathrm{s}),t_\mathrm{f},\boldsymbol{x}(t_\mathrm{f})), \\
& \boldsymbol{p}(t_\mathrm{s})=p.
\end{aligned}$$

Depending on the numerical method to solve an OCP, either formulation may be more efficient. For performance reasons parameters that may attain only one constant value should rather be considered as part of the model equations.

Transformation to Fixed Time Interval

When we introduced the OCP in standard form (Problem (5.1)) we assumed the initial and final time of the horizon to be fixed. Now we show that this assumption can be made without loss of generality.

We demonstrate that an OCP can be transformed into an equivalent one by a linear time transformation of the horizon interval, which is unknown in advance, to a normalized interval. As a result the free initial or final time enter the OCP as additional optimization variables.

For reasons that become clear later we use $[-1,+1]$ as the normalized interval in this thesis, whereas often the interval $[0,1]$ is chosen in the literature. The linear time transformation of the horizon is realized as follows: one defines a linear mapping $\boldsymbol{t}:[-1,1]\longrightarrow\mathcal{T}$ as

$$\boldsymbol{t}(\tau)\overset{\text{def}}{=}\frac{t_f+t_s}{2}+\tau\cdot\frac{t_f-t_s}{2},$$

where $\boldsymbol{t}(\cdot)$ maps a point from the normalized interval $[-1,+1]$ into the original horizon interval $[t_\mathrm{s},t_\mathrm{f}]$. Let furthermore $\widetilde{\boldsymbol{x}}:[-1,+1]\longrightarrow\mathbb{R}^{n_\mathrm{x}}$ and $\widetilde{\boldsymbol{u}}:[-1,+1]\longrightarrow\mathbb{R}^{n_\mathrm{u}}$ be defined as

$$\widetilde{\boldsymbol{x}}(\tau)\overset{\text{def}}{=}\boldsymbol{x}(\boldsymbol{t}(\tau)) \qquad \text{and} \qquad \widetilde{\boldsymbol{u}}(\tau)\overset{\text{def}}{=}\boldsymbol{u}(\boldsymbol{t}(\tau)).$$

Now we express the ODE (5.1b) in terms of the new states $\widetilde{\boldsymbol{x}}(\cdot)$ and controls $\widetilde{\boldsymbol{u}}(\cdot)$ as

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}\tau}\widetilde{\boldsymbol{x}}(\tau)&=\dot{\boldsymbol{x}}(\boldsymbol{t}(\tau))\cdot\frac{\mathrm{d}}{\mathrm{d}\tau}\boldsymbol{t}(\tau)=\frac{t_f-t_s}{2}\cdot\boldsymbol{f}(\boldsymbol{t}(\tau),\boldsymbol{x}(\boldsymbol{t}(\tau)),\boldsymbol{u}(\boldsymbol{t}(\tau))) \\
&=\frac{t_f-t_s}{2}\cdot\boldsymbol{f}(\boldsymbol{t}(\tau),\widetilde{\boldsymbol{x}}(\tau),\widetilde{\boldsymbol{u}}(\tau)).
\end{aligned}$$

The constraints (5.1c) and (5.1d) are handled in a similar way. In general there are two ways how free initial and final time can enter the transformed OCP: t_s and t_f can either be viewed

as additional constant states $\boldsymbol{t}_s(\cdot)$ and $\boldsymbol{t}_f(\cdot)$, where

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}\tau}\boldsymbol{t}_s(\tau) &= 0, \quad \boldsymbol{t}_s(-1) \text{ free},\\ \frac{\mathrm{d}}{\mathrm{d}\tau}\boldsymbol{t}_f(\tau) &= 0, \quad \boldsymbol{t}_f(-1) \text{ free},\end{aligned}$$

or more efficiently as additional scalar optimization variables. Problem (5.1) with initial time t_s and final time t_f as optimization variables then reads as

$$\begin{aligned}\min_{\widetilde{\boldsymbol{x}}(\cdot),\widetilde{\boldsymbol{u}}(\cdot),t_s,t_f} \quad & \boldsymbol{\varphi}(t_s,\widetilde{\boldsymbol{x}}(-1),t_f,\widetilde{\boldsymbol{x}}(+1)) + \int_{-1}^{+1} \frac{t_f - t_s}{2}\cdot\boldsymbol{\psi}(\boldsymbol{t}(\tau),\widetilde{\boldsymbol{x}}(\tau),\widetilde{\boldsymbol{u}}(\tau))\,\mathrm{d}\tau \\ \text{s.t.} \quad & \frac{\mathrm{d}}{\mathrm{d}\tau}\widetilde{\boldsymbol{x}}(\tau) = \frac{t_f - t_s}{2}\cdot\boldsymbol{f}(\boldsymbol{t}(\tau),\widetilde{\boldsymbol{x}}(\tau),\widetilde{\boldsymbol{u}}(\tau)), \quad \tau\in[-1,1],\\ & \mathbf{0}_{n_c} \geqslant \boldsymbol{c}(\boldsymbol{t}(\tau),\widetilde{\boldsymbol{x}}(\tau),\widetilde{\boldsymbol{u}}(\tau)), \quad \tau\in[-1,1],\\ & \mathbf{0}_{n_r} = \boldsymbol{r}(t_s,\widetilde{\boldsymbol{x}}(-1),t_f,\widetilde{\boldsymbol{x}}(+1)).\end{aligned} \tag{5.5}$$

Whenever OCPs of the form (5.5) appear in this thesis we drop the tilde and simply use $\boldsymbol{x}(\cdot)$ and $\boldsymbol{u}(\cdot)$ for state and control variables on the normalized interval to avoid notational clutter.

Differential Algebraic Equations

The standard OCP form may even cover optimization subject to Differential Algebraic Equation (DAE) systems. We restrict our discussion to *semi–explicit DAEs*, i.e., a DAE of type

$$\begin{aligned}\dot{\boldsymbol{x}}(t) &= \boldsymbol{f}(t,\boldsymbol{x}(t),\boldsymbol{y}(t),\boldsymbol{u}(t)), \quad t\in\mathcal{T},\\ \mathbf{0} &= \boldsymbol{g}(t,\boldsymbol{x}(t),\boldsymbol{y}(t),\boldsymbol{u}(t)), \quad t\in\mathcal{T},\end{aligned} \tag{5.6}$$

where $\boldsymbol{x}(\cdot)$ and $\boldsymbol{u}(\cdot)$ have the previous meaning and $\boldsymbol{y}:\mathcal{T}\longrightarrow\mathbb{R}^{n_y}$ are called *algebraic states*. More general DAEs such as $\boldsymbol{F}(t,\boldsymbol{z}(t),\dot{\boldsymbol{z}}(t),\boldsymbol{u}(t)) = 0$ can be transformed formally into a semi–explicit DAE by introducing an artificial algebraic variable. The index of a DAE measures its singularity when compared to an ODE. Today a number of definitions with different emphasis exist.

One common example is the *differential index*, which measures how often the algebraic equation (5.6) has to be differentiated at least until one obtains an ODE. We illustrate this for a specific case: let us assume that $\boldsymbol{g}$ and $\boldsymbol{u}$ are continuously differentiable. Let furthermore $\boldsymbol{g}_y'(t,x,y,u)$ be non–singular and $\boldsymbol{g}_y'(t,x,y,u)^{-1}$ be bounded for all $(t,x,y,u)\in\mathcal{T}\times\mathbb{R}^{n_x}\times\mathbb{R}^{n_y}\times\mathbb{R}^{n_u}$. When we apply the well known implicit function theorem we obtain a function $\boldsymbol{y}(\cdot,\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot))$ that solves (5.6). We get an explicit ODE system through one differentiation of (5.6) as

$$\begin{aligned}&\mathbf{0} = \boldsymbol{g}_t'[t] + \boldsymbol{g}_x'[t]\dot{\boldsymbol{x}}(t) + \boldsymbol{g}_y'[t]\dot{\boldsymbol{y}}(t) + \boldsymbol{g}_u'[t]\dot{\boldsymbol{u}}(t)\\ \Longrightarrow\quad &\dot{\boldsymbol{y}}(t) = -\boldsymbol{g}_y'[t]^{-1}\left(\boldsymbol{g}_t'[t] + \boldsymbol{g}_x'[t]\boldsymbol{f}[t] + \boldsymbol{g}_u'[t]\dot{\boldsymbol{u}}(t)\right),\end{aligned}$$

where, e.g. $\boldsymbol{g}_t'[t]$ is an abbreviation for $\boldsymbol{g}_t'(t, \boldsymbol{x}(t), \boldsymbol{y}(t), \boldsymbol{u}(t))$. Since it was necessary to differentiate (5.6) once in order to obtain an ODE the differentiation index is one.

Multistage Systems

For many practical problems a system cannot be described by a single set of model equations. Transitions as well as changes in the dynamics may occur. In case the sequence of the different dynamics is unknown in advance we deal with a switched OCP which is the main topic of this thesis. However, if the sequence of the dynamics is known in advance - this is the easier case - we can handle it by using so called *multistage OCPs*.
To formulate such a multistage system we introduce a finite number N which determines the number of model stages. Moreover, an ordered set $\{t_0, t_1, \ldots, t_N\}$ of time points $t_n \in \mathcal{T}$ is defined, where

$$t_s = t_0 \leqslant t_1 \leqslant \ldots \leqslant t_N = t_f.$$

The t_n indicate the time points, when a new model stage occurs. A multistage OCP is a constrained optimization problem of the form:

$$\begin{aligned}
\min_{\boldsymbol{x}_n(\cdot), \boldsymbol{u}_n(\cdot)} \quad & \sum_{n=1}^{N} J_n(\boldsymbol{x}_n(\cdot), \boldsymbol{u}_n(\cdot)) \\
\text{s.t.} \quad & \dot{\boldsymbol{x}}_n(t) = \boldsymbol{f}_n(t, \boldsymbol{x}_n(t), \boldsymbol{u}_n(t)), && t \in [t_{n-1}, t_n], \\
& \boldsymbol{0} \geqslant \boldsymbol{c}_n(t, \boldsymbol{x}_n(t), \boldsymbol{u}_n(t)), && t \in [t_{n-1}, t_n], \\
& \boldsymbol{0} = \boldsymbol{r}(t_s, \boldsymbol{x}_1(t_s), t_f, \boldsymbol{x}_N(t_f)), \\
& \boldsymbol{x}_n(t_{n-1}) = \boldsymbol{j}_n(\boldsymbol{x}_{n-1}(t_{n-1})), && 2 \leqslant n \leqslant N.
\end{aligned}$$

The objective Bolza functionals

$$J_n(\boldsymbol{x}_n(\cdot), \boldsymbol{u}_n(\cdot)) \stackrel{\text{def}}{=} \boldsymbol{\varphi}_n(t_{n-1}, \boldsymbol{x}_n(t_{n-1}), t_n, \boldsymbol{x}_n(t_n)) + \int_{t_{n-1}}^{t_n} \boldsymbol{\psi}_n(t, \boldsymbol{x}_n(t), \boldsymbol{u}_n(t)) \, \mathrm{d}t$$

as well as the designators $\boldsymbol{\varphi}_n$, $\boldsymbol{\psi}_n$, $\boldsymbol{f}_n$, $\boldsymbol{c}_n$, $\boldsymbol{r}$, $\boldsymbol{x}_n$, $\boldsymbol{u}_n$ and t correspond to those in Definition 5.1 without the index n. The dimensions of state and control vectors change towards n_{x_n} and n_{u_n} respectively. Their values may change from one model stage to another.

5.2 OCPs as Infinite Dimensional Optimization Problems

This section is dedicated to provide the connection between ODE optimal control and infinite dimensional optimization problems, cf. Chapter 3. We restrict our analysis to OCPs in standard form (5.1) on a fixed horizon interval $\mathcal{T}$. Transformation techniques in Section 5.1 show that this can be done without loss of generality.

Transformation

In order to derive first–order necessary optimality conditions for Problem (5.1), as this will be done in the following Section 5.3, it is convenient to rewrite the OCP as an infinite dimensional optimization problem in appropriate Banach spaces. To this end, the vector $z \stackrel{\text{def}}{=} (\boldsymbol{x}, \boldsymbol{u})$ of optimization variables in problem (5.1) is an element of the Banach space $(Z, \|\cdot\|_Z)$, where

$$\begin{aligned} Z &\stackrel{\text{def}}{=} W^{1,\infty}(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}}) \times L^\infty(\mathcal{T}, \mathbb{R}^{n_\mathrm{u}}), \\ \|(\boldsymbol{x}, \boldsymbol{u})\|_Z &\stackrel{\text{def}}{=} \max\left\{\|\boldsymbol{x}\|_{1,\infty}, \|\boldsymbol{u}\|_\infty\right\}. \end{aligned} \tag{5.8}$$

The Bolza type objective functional in Problem (5.1) defines a mapping $J : Z \longrightarrow \mathbb{R}$. The equality constraints (5.1b) and (5.1d) of the OCP define the operator equation

$$H(\boldsymbol{x}, \boldsymbol{u}) = \Theta_V, \tag{5.9}$$

where $H = (H_1, H_2) : Z \longrightarrow V$ is defined by

$$H_1(\boldsymbol{x}, \boldsymbol{u}) \stackrel{\text{def}}{=} \boldsymbol{f}(\cdot, \boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot)) - \dot{\boldsymbol{x}}(\cdot), \tag{5.10}$$

$$H_2(\boldsymbol{x}, \boldsymbol{u}) \stackrel{\text{def}}{=} -\boldsymbol{r}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})), \tag{5.11}$$

and the Banach space $(V, \|\cdot\|_V)$ is given as

$$\begin{aligned} V &\stackrel{\text{def}}{=} L^\infty(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}}) \times \mathbb{R}^{n_\mathrm{r}}, \\ \|(v_1, v_2)\|_V &\stackrel{\text{def}}{=} \max\left\{\|v_1\|_\infty, \|v_2\|_2\right\}. \end{aligned}$$

The set

$$\mathcal{K} \stackrel{\text{def}}{=} \left\{k \in L^\infty(\mathcal{T}, \mathbb{R}^{n_\mathrm{c}}) \,:\, k(t) \geqslant \mathbf{0}_{n_\mathrm{c}} \text{ a.e. in } \mathcal{T}\right\}$$

is a convex cone with non–empty interior in the Banach space $(W, \|\cdot\|_W)$, where W is given as

$$\begin{aligned} W &\stackrel{\text{def}}{=} L^\infty(\mathcal{T}, \mathbb{R}^{n_\mathrm{c}}), \\ \|w\|_W &\stackrel{\text{def}}{=} \|w\|_\infty. \end{aligned}$$

The inequality constraint (5.1c) of the OCP defines the cone constraint

$$G(\boldsymbol{x}, \boldsymbol{u}) \in \mathcal{K}, \tag{5.12}$$

where $G : Z \longrightarrow W$ is given by

$$G(\boldsymbol{x}, \boldsymbol{u}) \stackrel{\text{def}}{=} -\boldsymbol{c}(\cdot, \boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot)). \tag{5.13}$$

Summarizing, the OCP (5.1) is equivalent to the infinite dimensional optimization problem

$$\begin{aligned} \min_{(\boldsymbol{x},\boldsymbol{u})\in Z} \quad & J(\boldsymbol{x},\boldsymbol{u}) \\ \text{s. t.} \quad & G(\boldsymbol{x},\boldsymbol{u}) \in K, \\ & H(\boldsymbol{x},\boldsymbol{u}) = \Theta_V. \end{aligned} \tag{5.14}$$

Differentiability Properties and Notation

We conclude this section by investigating differentiability properties of the functionals J, G and H on the one hand and introduce some notations on the other hand which allow us to write formulas more compactly in the remainder of this thesis. Assumption 5.2 summarizes the smoothness requirements that we put on the functions in Problem (5.1).

Assumption 5.2
Let the functions $\boldsymbol{\varphi}$, $\boldsymbol{\psi}$, $\boldsymbol{f}$, $\boldsymbol{c}$ and $\boldsymbol{r}$ in Problem (5.1) satisfy the following smoothness conditions:

1. $\boldsymbol{\varphi}$ and $\boldsymbol{r}$ are continuously differentiable with respect to all arguments.
2. Let $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}}) \in W^{1,\infty}(\mathcal{T}, \mathbb{R}^{n_x}) \times L^{\infty}(\mathcal{T}, \mathbb{R}^{n_u})$ be given and let M be a sufficiently large convex compact neighborhood of

$$\left\{(\hat{\boldsymbol{x}}(t), \hat{\boldsymbol{u}}(t)) \in \mathbb{R}^{n_x+n_u} \,:\, t \in \mathcal{T}\right\}.$$

   a) The mappings $t \mapsto \boldsymbol{\psi}(t,x,u)$ and

$$t \mapsto \boldsymbol{f}(t,x,u), \qquad t \mapsto \boldsymbol{c}(t,x,u)$$

   are measurable for every $(x,u) \in M$.

   b) The mappings $(x,u) \mapsto \boldsymbol{\psi}(t,x,u)$ and

$$(x,u) \mapsto \boldsymbol{f}(t,x,u), \qquad (x,u) \mapsto \boldsymbol{c}(t,x,u)$$

   are continuously differentiable in M uniformly for $t \in \mathcal{T}$.

   c) The derivatives

$$\boldsymbol{\psi}'_{(x,u)}, \quad \boldsymbol{f}'_{(x,u)}, \quad \boldsymbol{c}'_{(x,u)}$$

   are bounded in $\mathcal{T} \times M$.

Fréchet–differentiability of functions J, G and H in problem (5.14) can be obtained under Assumption 5.2. The following result illustrates this for a simplified setting.

Theorem 5.3 (see Gerdts [190], Theorem 2.2.9)
Let $\hat{\boldsymbol{x}} \in W^{1,\infty}(\mathcal{T}, \mathbb{R}^{n_x})$ be given and let $\boldsymbol{f} : \mathcal{T} \times \mathbb{R}^{n_x} \longrightarrow \mathbb{R}^{n_x}$, $(t,x) \mapsto \boldsymbol{f}(t,x)$ be a function satisfying the conditions in Assumptions 5.2 with

$$M \stackrel{\text{def}}{=} \{x \in \mathbb{R}^{n_x} \,:\, \exists t \in \mathcal{T}, \|x - \hat{\boldsymbol{x}}(t)\| \leqslant r\}, \quad r > 0.$$

Then the mapping $T : W^{1,\infty}(\mathcal{T}, \mathbb{R}^{n_x}) \longrightarrow L^{\infty}(\mathcal{T}, \mathbb{R}^{n_x})$ defined by

$$T(\boldsymbol{x}(\cdot)) \stackrel{\text{def}}{=} \dot{\boldsymbol{x}}(\cdot) - \boldsymbol{f}(\cdot, \boldsymbol{x}(\cdot))$$

is continuously Fréchet–differentiable in $\hat{\boldsymbol{x}}$ with derivative

$$T'(\hat{\boldsymbol{x}})(\boldsymbol{x}) = \dot{\boldsymbol{x}}(\cdot) - \boldsymbol{f}'_x(\cdot, \hat{\boldsymbol{x}}(\cdot))\boldsymbol{x}(\cdot). \qquad \triangle$$

By using similar arguments one can show Fréchet–differentiability for J, G and H. The derivatives are given as

$$J'(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}})(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{\varphi}'_{x_s}\boldsymbol{x}(t_s) + \boldsymbol{\varphi}'_{x_f}\boldsymbol{x}(t_f) + \int_{t_s}^{t_f} \boldsymbol{\psi}'_x[t]\boldsymbol{x}(t) + \boldsymbol{\psi}'_u[t]\boldsymbol{u}(t)\mathrm{d}t$$

for the Bolza type objective functional,

$$\begin{aligned} H'_1(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}})(\boldsymbol{x}, \boldsymbol{u}) &= \boldsymbol{f}'_x[\cdot]\boldsymbol{x}(\cdot) + \boldsymbol{f}'_u[\cdot]\boldsymbol{u}(\cdot) - \dot{\boldsymbol{x}}(\cdot) \\ H'_1(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}})(\boldsymbol{x}, \boldsymbol{u}) &= -\boldsymbol{r}'_{x_s}\boldsymbol{x}(t_s) - \boldsymbol{r}'_{x_f}\boldsymbol{x}(t_f) \end{aligned}$$

for the equality constraint functional and

$$G'(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}})(\boldsymbol{x}, \boldsymbol{u}) = -\boldsymbol{c}'_x[\cdot]\boldsymbol{x}(\cdot) + \boldsymbol{c}'_u[\cdot]\boldsymbol{u}(\cdot).$$

for the inequality constraint functional. For notational convenience we used the abbreviations

$$\boldsymbol{\varphi}'_{x_s} \stackrel{\text{def}}{=} \boldsymbol{\varphi}'_{x_s}(t_s, \hat{\boldsymbol{x}}(t_s), t_f, \hat{\boldsymbol{x}}(t_f)), \quad \boldsymbol{\psi}'_x[t] \stackrel{\text{def}}{=} \boldsymbol{\psi}'_x(t, \hat{\boldsymbol{x}}(t), \hat{\boldsymbol{u}}(t)).$$

In a similar fashion are $\boldsymbol{\varphi}'_{x_f}$, $\boldsymbol{\psi}'_u[t]$, $\boldsymbol{f}'_x[t]$, $\boldsymbol{f}'_u[t]$, $\boldsymbol{c}'_x[t]$, $\boldsymbol{c}'_u[t]$, $\boldsymbol{r}'_{x_s}$ and $\boldsymbol{r}'_{x_f}$ defined for the respective derivatives. These abbreviations together with

$$\boldsymbol{\psi}[t] \stackrel{\text{def}}{=} \boldsymbol{\psi}(t, \hat{\boldsymbol{x}}(t), \hat{\boldsymbol{u}}(t))$$

and analogously $\boldsymbol{f}[t]$, $\boldsymbol{c}[t]$ are used for particular cases in the remaining chapters.

5.3 Local Minimum Principle

First–order necessary optimality conditions in terms of a local minimum principle are often expressed by means of the Hamilton Function resp. the augmented Hamilton Function.

Definition 5.4 (Hamilton Function)
The Hamilton *function* $\mathcal{H} : \mathcal{T} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_x} \times \mathbb{R} \longrightarrow \mathbb{R}$ for OCP (5.1) is defined by

$$\mathcal{H}(t, x, u, \lambda, l_0) \stackrel{\text{def}}{=} l_0\, \psi(t, x, u) + \lambda^T \boldsymbol{f}(t, x, u). \qquad \triangle$$

Definition 5.5 (Augmented Hamilton Function)
The *augmented* Hamilton *function* $\hat{\mathcal{H}} : \mathcal{T} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_c} \times \mathbb{R} \longrightarrow \mathbb{R}$ for OCP (5.1) is defined by

$$\hat{\mathcal{H}}(t, x, u, \lambda, \mu, l_0) \stackrel{\text{def}}{=} \mathcal{H}(t, x, u, \lambda, l_0) - \mu^T\, \mathbf{c}(t, x, u). \qquad \triangle$$

In order to formulate the local minimum principle we need the following assumption:

Assumption 5.6
We assume that $\operatorname{rank}(\boldsymbol{c}_u'[t]) = n_c$ holds almost everywhere in $\mathcal{T}$ and that the *pseudo–inverse* of $\boldsymbol{c}_u'[t]$,

$$\big(\boldsymbol{c}_u'[t]\big)^+ \stackrel{\text{def}}{=} \boldsymbol{c}_u'[t]^T\big(\boldsymbol{c}_u'[t]\boldsymbol{c}_u'[t]^T\big)^{-1},$$

is essentially bounded in $\mathcal{T}$.

Theorem 5.7 (Local Minimum Principle)
Let the Assumptions 5.2 and 5.6 hold and let $(\boldsymbol{x}^*, \boldsymbol{u}^*)$ be a local minimum of Problem (5.1). Then there exist multipliers

$$l_0 \geqslant 0, \quad \nu \in \mathbb{R}^{n_r}, \quad \boldsymbol{\lambda} \in \mathcal{BV}(\mathcal{T}, \mathbb{R}^{n_x}), \quad \boldsymbol{\mu} \in L^\infty(\mathcal{T}, \mathbb{R}^{n_c})$$

such that the following conditions hold:

(i) $l_0 \geqslant 0, (l_0, \nu, \boldsymbol{\lambda}, \boldsymbol{\mu}) \neq \Theta$

(ii) Adjoint equations:

$$\boldsymbol{\lambda}(t) = \boldsymbol{\lambda}(t_f) + \int_t^{t_f} \hat{\mathcal{H}}_x'(\tau, \boldsymbol{x}^*(\tau), \boldsymbol{u}^*(\tau), \boldsymbol{\lambda}(\tau), \boldsymbol{\mu}(\tau), l_0)^T \, \mathrm{d}\tau \quad \text{a.e. } t \in \mathcal{T}$$

(iii) Transversality conditions:

$$\begin{aligned}\boldsymbol{\lambda}(t_s) &= -l_0 \boldsymbol{\varphi}_{x_s}'(\boldsymbol{x}^*(t_s), \boldsymbol{x}^*(t_f))^T - \boldsymbol{r}_{x_s}'(\boldsymbol{x}^*(t_s), \boldsymbol{x}^*(t_f))^T \, \nu \\ \boldsymbol{\lambda}(t_f) &= +l_0 \boldsymbol{\varphi}_{x_f}'(\boldsymbol{x}^*(t_s), \boldsymbol{x}^*(t_f))^T + \boldsymbol{r}_{x_f}'(\boldsymbol{x}^*(t_s), \boldsymbol{x}^*(t_f))^T \, \nu\end{aligned}$$

(iv) Stationarity of augmented Hamilton function:

$$\hat{\mathcal{H}}_u'(t, \boldsymbol{x}^*(t), \boldsymbol{u}^*(t), \boldsymbol{\lambda}(t), \boldsymbol{\mu}(t), l_0) = \boldsymbol{0}^T \quad \text{a.e. } t \in \mathcal{T} \tag{5.15}$$

(v) Complementarity conditions:

$$\boldsymbol{\mu}(t)^T \boldsymbol{c}(t, \boldsymbol{x}^*(t), \boldsymbol{u}^*(t)) = 0, \quad \boldsymbol{\mu}(t) \leqslant \boldsymbol{0} \quad \text{a.e. } t \in \mathcal{T} \qquad \triangle$$

Proof See e.g. Gerdts [190, Theorem 3.4.4]. □

Note, that $\boldsymbol{\lambda}(\cdot)$ is differentiable almost everywhere in $\mathcal{T}$ and thus it holds

$$\dot{\boldsymbol{\lambda}}(t) = -\hat{\mathcal{H}}_x'(t, \boldsymbol{x}^*(t), \boldsymbol{u}^*(t), \boldsymbol{\lambda}(t), \boldsymbol{\mu}(t), l_0)^T \quad \text{a.e. } t \in \mathcal{T}. \tag{5.16}$$

In order to conclude the section we need to state conditions which guarantee that we can set $l_0 = 1$ in Theorem 5.7. A first step towards this goal is provided by the following lemma.

Lemma 5.8
Let

$$\operatorname{rank}(\boldsymbol{r}_{x_s}' \boldsymbol{\Phi}(t_s) + \boldsymbol{r}_{x_f}' \boldsymbol{\Phi}(t_f)) = n_r,$$

with $\boldsymbol{\Phi}(\cdot)$ being the fundamental solution of the homogeneous linear differential equation

$$\dot{\boldsymbol{\Phi}}(t) = \boldsymbol{f}_x'[t]\boldsymbol{\Phi}(t), \quad \boldsymbol{\Phi}(t_s) = \boldsymbol{I}_{n_x}, \quad t \in \mathcal{T}.$$

Then $H'(\boldsymbol{x}^*, \boldsymbol{u}^*)$ with H in (5.10)–(5.11) is surjective. △

Proof See e.g. Gerdts [189, Lemma 3.1.12]. □

The Mangasarian–Fromowitz Constraint Qualification as stated in Corollary 3.11 together with Lemma 5.8 can be translated as follows for Problem (5.1), since the interior of

$$K = \left\{\boldsymbol{k} \in L^\infty(\mathcal{T}, \mathbb{R}^{n_c}) : \boldsymbol{k}(t) \geqslant \boldsymbol{0}_{n_c} \text{ a.e. in } \mathcal{T}\right\}$$

is given by

$$\begin{aligned}\operatorname{int}(K) &= \{\boldsymbol{k} \in L^\infty(\mathcal{T}, \mathbb{R}^{n_c}) : \exists\, \varepsilon > 0 \text{ with } \mathcal{U}_\varepsilon(\boldsymbol{k}) \subseteq K\} \\ &= \{\boldsymbol{k} \in L^\infty(\mathcal{T}, \mathbb{R}^{n_c}) : \exists\, \varepsilon > 0 \text{ with } \boldsymbol{k}_i(t) \geqslant \varepsilon \text{ a.e. in } \mathcal{T},\ i \in [n_c]\}\end{aligned}$$

Lemma 5.9
Let the assumptions of Theorem 5.7 and Lemma 5.8 hold and let there exist $\boldsymbol{x} \in W^{1,\infty}(\mathcal{T}, \mathbb{R}^{n_x})$ and $\boldsymbol{u} \in L^\infty(\mathcal{T}, \mathbb{R}^{n_u})$ with

$$\begin{aligned}-\varepsilon \cdot \boldsymbol{1} &\geqslant \boldsymbol{c}[t] + \boldsymbol{c}_x'[t]\boldsymbol{x}(t) + \boldsymbol{c}_u'[t]\boldsymbol{u}(t) \quad \text{a.e. in } \mathcal{T}, \\ \boldsymbol{0} &= \boldsymbol{f}_x'[t]\boldsymbol{x}(t) + \boldsymbol{f}_u'[t]\boldsymbol{u}(t) - \dot{\boldsymbol{x}}(t) \quad \text{a.e. in } \mathcal{T}, \\ \boldsymbol{0} &= \boldsymbol{r}_{x_s}'\boldsymbol{x}(t_s) + \boldsymbol{r}_{x_f}'\boldsymbol{x}(t_f).\end{aligned}$$

Then the Mangasarian–Fromowitz Constraint Qualification holds for Problem (5.1) and $l_0 = 1$ can be chosen in Theorem 5.7. △

In later chapters, we extensively work with the local minimum principle. Usually, we assume – without explicitly mentioning – constraint qualifications to be satisfied. In this case we can omit the l_0 in the (augmented) Hamilton function.

5.4 Solution Structure

In this section we outline how the structure of an optimal control $\boldsymbol{u}^*(\cdot)$ can be determined. To this end, we start by introducing some notation. In this section we assume the OCP under investigation to be autonomous and with Mayer term objective. This enables us to write occurring formulas and equations in a compact form. Due to the transformation techniques presented in Section 5.1 this is possible without loss of generality.
The set of all admissible control functions $\boldsymbol{u}(\cdot)$ such that the mixed control–state constraints (5.1c) hold evaluated at time t with state $\boldsymbol{x}(t)$ is called *admissible region* and denoted by $\mathcal{R}(t, \boldsymbol{x}(t))$. The boundary $\partial\mathcal{R}(t, \boldsymbol{x}(t))$ is the set of all controls $\boldsymbol{u}(\cdot)$ such that $\boldsymbol{c}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) = 0$ whereas the interior $\operatorname{int}(\mathcal{R}(t, \boldsymbol{x}(t)))$ is the union of all controls $\boldsymbol{u}(\cdot)$ with $\boldsymbol{c}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) < 0$. A constraint $\boldsymbol{c}_i$ with $\boldsymbol{c}_i = 0$ is called active and inactive at time t if $\boldsymbol{c}_i < 0$, $i \in [n_c]$.

General Nonlinear Case

To learn more about the structure of an optimal control $\boldsymbol{u}^*(\cdot)$ of Problem (5.1) we can use the first–order optimality conditions. A question that has to be investigated is if a control lies in

the interior or on the boundary of the admissible region. Hence, it has to be investigated if a constraint $\boldsymbol{c}_i$ is active or not.
Let us assume we have an optimal control $\boldsymbol{u}(\cdot)$, corresponding Lagrange multipliers $\boldsymbol{\lambda}(\cdot)$, $\boldsymbol{\mu}(\cdot)$, ν and state $\boldsymbol{x}(\cdot)$. Therefore it holds (5.15) but also point– and componentwise

$$0 = \hat{\mathcal{H}}'_{u_i}[t] = \boldsymbol{\lambda}^T(t)\boldsymbol{f}'_{u_i}[t] - \boldsymbol{\mu}^T(t)\boldsymbol{c}'_{u_i}[t], \quad i \in [n_\mathrm{u}].$$

For each control $\boldsymbol{u}_i$ two cases can be distinguished: either it is

$$\boldsymbol{\lambda}^T(t)\boldsymbol{f}'_{u_i}[t] \neq 0 \qquad \text{or} \qquad \boldsymbol{\lambda}^T(t)\boldsymbol{f}'_{u_i}[t] = 0. \tag{5.17}$$

In order to take account its importance for the rest of this section we call the term $\boldsymbol{\lambda}^T(t)\boldsymbol{f}'_u[t]$, which is identical with the derivative of the Hamilton function with respect to the control component, switching function.

Definition 5.10 (Switching Function)
The n_u–dimensional *switching function* is given by

$$\boldsymbol{\sigma}^T[t] \stackrel{\text{def}}{=} \mathcal{H}'_u[t] = \boldsymbol{\lambda}^T(t)\boldsymbol{f}'_u[t] \qquad \triangle$$

We start by investigating the first case in (5.17) which means that the i–th entry of the switching function is not equal to zero at time t. This implies $\boldsymbol{\mu}(t)^T\boldsymbol{c}'_{u_i}[t] \neq 0$. Since $\boldsymbol{\mu}(t) \leqslant 0$, at least one entry of $\boldsymbol{\mu}(t)$ has to be strictly negative. Due to the complementarity conditions, this implies that one constraint must be active and also explicitly depends on u_i since $\boldsymbol{c}'_{u_i}[t]$ is not equal to zero per assumption. Accordingly, this constraint can be adduced to determine an analytic expression for $\boldsymbol{u}_i(t)$ by means of the implicit function theorem. In literature $\boldsymbol{u}_i(t)$ is often called *constraint–seeking*, cf. Srinivasan et al. [415]. For numerical methods it plays an important role if t lies on a boundary arc or if it is a touch point.

Definition 5.11 (Boundary Arc, Junction Points, Contact Point, Touch Point)
Let $(\boldsymbol{x}(t), \boldsymbol{u}(t))$ be feasible for Problem (5.1).

(i) A compact interval $[t_1, t_2] \subseteq \mathcal{T}$ with $t_1 < t_2$ is called *boundary arc* of $\boldsymbol{c}_i$ if $\boldsymbol{c}_i$ is active on $[t_1, t_2]$. $[t_1, t_2]$ is called *free arc* of $\boldsymbol{c}_i$ if $\boldsymbol{c}_i$ is inactive on $[t_1, t_2]$. The point t_1 is called *junction point* of the boundary arc $[t_1, t_2]$ of $\boldsymbol{c}_i$ if there is a $\delta > 0$ such that $\boldsymbol{c}_i$ is inactive for all $t \in [t_1 - \delta, t_1)$.

(ii) A point $t_1 \in \mathcal{T}$ is called *contact point* of $\boldsymbol{c}_i$ if $\boldsymbol{c}_i$ is active at t_1 and there is a $\delta > 0$ such that $\boldsymbol{c}_i$ is inactive for all $t \in [t_1 - \delta, t_1 + \delta] \setminus \{t_1\}$.

(iii) A contact point t_1 of $\boldsymbol{c}_i$ is called *touch point* if $\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{c}_i$ is continuous at t_1. $\triangle$

We continue by investigating the second case in (5.17) which means that the i–th entry of the switching function is equal to zero at time t. Now we have to distinguish two more cases: if $\boldsymbol{\lambda}^T(t)\boldsymbol{f}'_{u_i}[t]$ does explicitly depend on u_i the control can be determined from the equation $\boldsymbol{\lambda}^T(t)\boldsymbol{f}'_{u_i}[t] = 0$. Otherwise we differentiate $\hat{\mathcal{H}}'_{u_i}[t]$ with respect to time. The resulting derivatives must vanish since it is $\hat{\mathcal{H}}'_{u_i}[t] = 0$ for all t. We get

$$\frac{\mathrm{d}}{\mathrm{d}t}\hat{\mathcal{H}}'_{u_i}[t] = \dot{\boldsymbol{\lambda}}^T(t)\boldsymbol{f}'_{u_i}[t] + \boldsymbol{\lambda}^T(t)\left(\frac{\partial \boldsymbol{f}_{u_i}}{\partial x}[t]\dot{\boldsymbol{x}}(t) + \frac{\partial \boldsymbol{f}_{u_i}}{\partial u}[t]\dot{\boldsymbol{u}}(t)\right)$$

$$
\begin{aligned}
&-\dot{\boldsymbol{\mu}}^T(t)\boldsymbol{c}'_{u_i}[t]-\boldsymbol{\mu}^T(t)\left(\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{c}'_{u_i}[t]\right)\\
&=\dot{\boldsymbol{\lambda}}^T(t)\boldsymbol{f}'_{u_i}[t]+\boldsymbol{\lambda}^T(t)\left(\frac{\partial \boldsymbol{f}_{u_i}}{\partial x}[t]\dot{\boldsymbol{x}}(t)+\frac{\partial \boldsymbol{f}_{u_i}}{\partial u}[t]\dot{\boldsymbol{u}}(t)\right).
\end{aligned}
$$

The latter equality holds as $\boldsymbol{\mu}^T(t)\boldsymbol{c}'_{u_i}[t]=0$ and the complementarity conditions. Now we replace $\dot{\boldsymbol{x}}(t)$ and $\dot{\boldsymbol{\lambda}}(t)$ with the respective expressions in (5.1b) and (5.16) and obtain

$$
\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\hat{\mathcal{H}}'_{u_i}[t]&=\boldsymbol{\lambda}^T(t)\left(\frac{\partial \boldsymbol{f}_{u_i}}{\partial x}[t]\boldsymbol{f}[t]-\frac{\partial \boldsymbol{f}}{\partial x}[t]\boldsymbol{f}_{u_i}[t]+\frac{\partial \boldsymbol{f}_{u_i}}{\partial u}[t]\dot{\boldsymbol{u}}(t)\right)+\boldsymbol{\mu}^T(t)\frac{\partial \boldsymbol{c}}{\partial x}[t]\boldsymbol{f}_{u_i}[t]\\
&=\boldsymbol{\lambda}^T(t)\Delta^1\boldsymbol{f}_{u_i}[t]+\boldsymbol{\mu}^T(t)\frac{\partial \boldsymbol{c}}{\partial x}[t]\boldsymbol{f}_{u_i}[t], \qquad (5.18)
\end{aligned}
$$

where the operator Δ^1 is defined as

$$
\Delta^1\boldsymbol{F}\overset{\text{def}}{=}\frac{\partial \boldsymbol{F}}{\partial x}[t]\boldsymbol{f}[t]-\frac{\partial \boldsymbol{f}}{\partial x}[t]\boldsymbol{F}[t]+\frac{\partial \boldsymbol{F}}{\partial u}[t]\dot{\boldsymbol{u}}(t),
$$

and represents the time differentiation of a vector field $\boldsymbol{F}$ along the trajectories of the OCP dynamic system. This operator is studied in the literature by means of Lie algebra tools. If we repeat differentiation (5.18) $j-1$ more times this leads to an expression which consists of a system dependent part and a constraint dependent one as

$$
\frac{\mathrm{d}^j}{\mathrm{d}t^j}\hat{\mathcal{H}}'_{u_i}[t]=\boldsymbol{\lambda}^T(t)\Delta^j\boldsymbol{f}_{u_i}[t]+\boldsymbol{\mu}^T(t)\frac{\partial \boldsymbol{c}}{\partial x}[t]\Delta^{j-1}\boldsymbol{f}_{u_i}[t], \qquad (5.19)
$$

where the operator Δ^j is defined recursively as $\Delta^j\overset{\text{def}}{=}\Delta^1(\Delta^{j-1})$ with $\Delta^0\overset{\text{def}}{=}id$. The differentiation with respect to t in (5.19) is repeated until one of two cases occurs: if we have $\boldsymbol{\lambda}^T(t)\Delta^j\boldsymbol{f}_{u_i}[t]\neq 0$ then we conclude with a similar argumentation as above that the control is constraint–seeking. In the second case, where the control is called *sensitivity–seeking*, we have $\boldsymbol{\lambda}^T(t)\Delta^j\boldsymbol{f}_{u_i}[t]=0$ and u_i appears explicitly in the expression.

Control–Affine Systems

Now we have a short look at the important case of control–affine systems. In these systems the control $\boldsymbol{u}(\cdot)$ enters linearly in the system dynamics and mixed control–state constraints, i.e., $\boldsymbol{f}$ and $\boldsymbol{c}$ have the form

$$
\begin{aligned}
\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})&=\boldsymbol{f}^1(\boldsymbol{x})+\boldsymbol{f}^2(\boldsymbol{x})\cdot\boldsymbol{u},\\
\boldsymbol{c}(\boldsymbol{x},\boldsymbol{u})&=\boldsymbol{c}^1(\boldsymbol{x})+\boldsymbol{c}^2(\boldsymbol{x})\cdot\boldsymbol{u}.
\end{aligned}
$$

Definition 5.12 (Singularity)
We call a control function $\boldsymbol{u}(\cdot)$ *singular* of rank r over a non–zero time interval $[t_1,t_2]$ with $t_1<t_2$, if r components of $\boldsymbol{u}(\cdot)$ cannot be determined from the condition $\boldsymbol{\sigma}[t]=\boldsymbol{0}$ over this interval. We call $\boldsymbol{u}(\cdot)$ to be *non–singular* if $r=0$. A control input $\boldsymbol{u}_i(\cdot)$ has a degree of singularity r if $\boldsymbol{u}_i(\cdot)$ appears for the

first time in the $(r+1)$–th time derivative of $\boldsymbol{\sigma}(\cdot)$. △

As a result of second–order necessary optimality conditions one can show that in the singular case $\boldsymbol{u}_i(\cdot)$ first appears in an even derivative of $\boldsymbol{\sigma}(\cdot)$. The control $\boldsymbol{u}_i(\cdot)$ is called *bang–bang control* in an interval $[t_1, t_2] \subseteq \mathcal{T}$ if it is non–singular and determined by a bound constraint, i.e., $\boldsymbol{u}_i(t) = u_i^{\max}$ or $\boldsymbol{u}_i(t) = u_i^{\min}$ for $t \in [t_1, t_2]$. Let us consider the case where $\boldsymbol{c}$ consists of bounds on the controls only, i.e.,

$$\boldsymbol{c}(\boldsymbol{u}) = \begin{bmatrix} \boldsymbol{u} - u^{\max} \\ u^{\min} - \boldsymbol{u} \end{bmatrix}.$$

The switching function of control $\boldsymbol{u}_i(\cdot)$ is given as

$$\boldsymbol{\sigma}_i[t] = \boldsymbol{\lambda}^T(t) \boldsymbol{f}_{\cdot i}^2[t],$$

where $\boldsymbol{f}_{\cdot i}^2$ denotes the i–th column of $\boldsymbol{f}^2$. If it is strictly positive, the pointwise minimizer of the Hamilton function $\boldsymbol{u}_i(\cdot)$ must be as small as possible and therefore at its lower bound, i.e., $\boldsymbol{u}_i(t) = u^{\min}$. Conversely, if it is strictly negative it holds with the same argumentation that $\boldsymbol{u}_i(t) = u^{\max}$. In both cases we obtain a bang–bang control.
If the switching function vanishes we cannot determine $\boldsymbol{u}_i(\cdot)$ from this expression since it is $\mathcal{H}''_{uu}[t] = 0$. In the singular case $\boldsymbol{\sigma}_i[t]$ has to be differentiated with respect to time until the degree of singularity of $\boldsymbol{u}_i(\cdot)$ is reached if we assume that it is finite. Apart from pathological cases singular controls lie in the interior of the admissible region.

5.5 Solution Approaches

We distinguish the *discretization approach* and the *function space approach* to solve an OCP. In the discretization approach the infinite dimensional problem is approximated by a finite dimensional one i.e., applying a suitable discretization scheme to transform the OCP into a finite dimensional optimization problem. The discretization approach is also known as *direct approach* or as *'first discretize, then optimize approach'*. A detailed investigation of the direct approach is done in Chapter 6.
On the other hand, the function space approach considers the OCP as an infinite dimensional optimization problem, usually in a suitable Banach space setting. The function space approach is also known as *indirect approach* or as *'first optimize, then discretize approach'*. An indirect approach setting is investigated in the Section 5.2.

5.5.1 Indirect Methods

The classical *indirect approach* for OCPs is a subbranch of the function space approach and can be considered a semi–analytical method. In indirect methods the first–order necessary optimality conditions, that were introduced in Section 5.3, are exploited to setup a nonlinear multipoint boundary value problem which has to be satisfied by a minimizer necessarily. The boundary value problem is solved, e.g. by multiple shooting, cf. Bulirsch [90] and Bock [69].

The aforementioned exploitation of the minimum principle can usually not be executed automatically by an algorithm but must be user provided. Especially in the presence of mixed control–state constraints an optimal solution consists of several arcs. As discussed earlier in this chapter there are either constraint–seeking or sensitivity–seeking arcs, cf. Section 5.4.
The transition from one arc type to another is modeled as follows: one introduces switching times t_i which act as additional variables. The necessary optimality conditions are expanded by so called *switching conditions* $\boldsymbol{s}(\boldsymbol{x}(t_i), \boldsymbol{u}(t_i)) = 0$ that determine the switching time variables resulting in a multipoint boundary value problem. It is a challenging task to determine correct and numerically stable switching conditions. Special cases that have to be distinguished are listed in Section 5.4. Especially mixed control–state constraints are accompanied by an a priori unknown switching structure. Tailored numerical solution methods have to deal with constraints becoming active or inactive, the distinction between touch points and boundary arcs for active constraints as well as jumps in the adjoint variables. The interested reader can find some related material in the works of Bock [70] or Hartl et al. [224].

5.5.2 Direct Methods

Compared to indirect methods, the *direct methods* do not rely on the application of minimum principles but on a suitable discretization of the infinite dimensional OCP. The discretization process results in a finite dimensional optimization problem, namely in a NLP as this was introduced in Section 3.5. These NLPs have often a special structure and can be solved efficiently by tailored numerical methods such as SQP or interior–point methods, cf. Section 3.6.
Direct methods for OCPs are classified according to the discretization strategy that they use. Based on this strategy the resulting NLPs differ in their problem dimensions and one receives more or less numerically stable systems. In the succeeding Chapter 6 we will present a deep insight into common OCP discretization techniques with a focus on those that are in particular relevant for the remainder of this thesis.
Direct methods excel especially in solving large–scale OCPs and problems, where one is unable to setup minimum principles. They enable users without deep knowledge in optimal control theory to solve OCPs. Moreover, numerical experiments evidently identify direct methods to be numerically robust and sufficiently accurate.

5.5.3 Direct versus Indirect Methods

The preceding sections suggested different approaches to solve OCPs numerically. Now we aggregate the strengths and weaknesses of the different approaches and based on this analysis we derive a methodology that we can follow in this thesis.

Advantages of Indirect Approach

The state and control trajectories that one obtains when using indirect methods to solve OCPs are very accurate. This is due to the fact that the infinite–dimensional problem has been solved. In particular, compared with direct methods there is no need for an approximation of the controls. The high accuracy of the obtained solutions is the main advantage of indirect methods.

The boundary value problems that arise from indirect methods have only a dimension $2n_x$. As optimal controls are calculated analytically their degrees of freedom vanish and therefore it is worth using indirect methods in case of a high number of control functions when compared to direct methods. In contrast, it is advantageous to use direct methods if the number of states significantly exceeds the number of controls.

Disadvantages of Indirect Approach

First of all the user has to compute derivatives of the Hamilton function to obtain first order optimality conditions. Even if the user is able to get over this error–prone task it is sometimes impossible or at least very difficult to construct these expressions for complicated black box systems. Furthermore, it is impossible to develop a general–purpose solver since the aforementioned derivatives have to be derived each time a new problem is posed.
Problems involving path constraints result in the necessity of finding a good guess of the constrained–arc sequence. Unfortunately, this can be quite cumbersome without prior knowledge of the system. An unknown number of the constrained subarcs results in an unknown number of iteration variables. Furthermore, an unknown sequence of constrained and unconstrained arcs makes it very difficult to find the arc boundaries.
In order to ensure convergence of Newton's method the switching times t_i have to stay in the multiple shooting intervals. We are just able to transform the problem onto fixed switching times if the switching sequence is guessed correctly in advance and if it does not change during iterations of the multiple shooting algorithm.
The third issue with indirect methods is their lack of robustness. To start the routine the user needs a guess for the adjoint variables. This is a very difficult task since adjoint variables are no physical quantities. But even with a reasonable initialization of the adjoint variables, the numerical solution of the adjoint equations can be very ill–conditioned.

Synthesis

Finally, we want to point out why direct methods that will be introduced in detail in the following Chapter 6 are usually superior over indirect methods in practice and outline our procedure.
One major advantage of direct methods is their numerical robustness. Moreover, there is the possibility to derive approximations for adjoint variables by employing multipliers of the discretized problem. This can be achieved by comparing the necessary optimality conditions of the discretized problem and of the original problem.
For this reason it could be one reasonable approach to combine direct and indirect methods. At first one could solve the OCP with a direct method and then use the obtained solution to initialize an indirect method. This includes an initial guess for the switching structure as well as the adjoint variables.
For different reasons such as aforementioned complicated black box and large–scale systems we are often unable to setup the necessary optimality conditions which makes it impossible to apply indirect methods. Hence, we propose a different way which is sketched briefly. Iteratively we solve a sequence of finite dimensional optimization problems arising from OCP

discretizations. The process is initialized with a coarse discretization grid and is continuously adapted. The grid refinements are conducted according to a global error estimation with respect to a predefined quantity of interest. The error estimation algorithm involves the mentioned approximation of the adjoint variables. The algorithm terminates when the global error is smaller than a certain threshold.

Chapter 6

Direct Approach: Problem Discretization

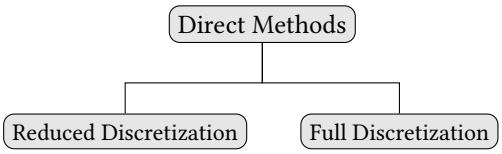


Figure 6.1: Direct Methods Overview

In Section 5.5.3 we discussed some severe difficulties of using the indirect approach for solving OCPs. In this chapter we present the *direct approach*, which has become very popular in recent years, as an alternative. Direct Methods do not formulate optimality conditions but *transcribe* the original infinite–dimensional optimization problem into a finite dimensional NLP with algebraic constraints and solve the resulting system. For this reason, direct methods are often called *direct transcription methods*. Another well known notion is "first discretize, then optimize" approach. More concrete, direct methods consist of repeatedly iterating three fundamental steps:

(i) **Problem Discretization:** *Transcribe* the OCP into an NLP by discretizing the OCP.

(ii) **Solve NLP:** Solve the *structured* NLP resulting from the OCP discretization.

(iii) **Mesh Refinement:** If necessary adapt the discretization by either *refining* or *coarsening* the mesh based on an *error estimation*.

The direct approach can therefore be seen as a SNLP algorithm. This chapter addresses the first step of the SNLP algorithm. Another discretization approach that is based on techniques developed in this chapter will be presented in Chapter 7. Aside from describing the discretization algorithm itself we will motivate it and point out the challenges that arise for implementations. NLPs as they arise in the second step have already been extensively analyzed: corresponding optimality conditions have been derived in Section 3.5. The numerical solution of NLPs is discussed in Section 3.6.
In direct transcription methods we omit forming necessary optimality conditions explicitly. But this results in a lack of values for the adjoint variables, and as a consequence we cannot

asses the accuracy of the solution because we have no clue about the adjoint equation. Hence, we need good strategies to deal with the mesh refinement step of the SNLP algorithm. There is a whole bunch of approaches available in the literature. In this thesis we develop a strategy that is based on two central steps: first we describe a way how to retrieve *discrete representations* of the *adjoint variables* in Chapter 9. Then we use them in a second step to derive a *global goal–oriented error estimation* in Chapter 10.
Direct methods finitely parametrize the states and/or controls of the original infinite dimensional OCP to approximate them in some appropriate manner. We call a method *control parametrization method* if just the controls are approximated. If a method approximates states and controls at the same time we call it *state and control parametrization method.* In either a control parametrization method or a state and control parametrization method the OCP is transcribed to a NLP.

6.1 Derivative Generation

No matter if the control parametrization or the state and control parametrization approach is used to transcribe an OCP into a NLP at least first–order derivative information is required by NLP solvers. We just sketch an overview about how derivatives can be calculated. For further information and details about the topic the reader is referred to e.g. Albersmeyer [8] and the excellent textbook of Griewank and Walther [210].
Fully discretized systems, i.e., if states and controls are parametrized, require derivatives of all occurring functions. There are three well–established ways to do this.

- *Analytic Differentiation* calculates all derivatives by hand and implements them as functions. This includes also the so–called *Symbolic Differentiation*, where computer algebra systems like `Maple` [329] and `Mathematica` [250] are used to derive a symbolic expression for the desired derivative and translates the symbolic derivative expression into e.g. `Fortran` or C source code. It is obvious that derivatives are exact up to machine precision. But calculating derivatives by hand is error–prone and symbolic expressions may not always be available. Moreover, symbolic expressions for the derivative may be inefficient in terms of evaluation time, cf. Speelpenning [414].

- *Finite Differences* uses the principle of central difference scheme to approximate directional derivatives and requires only multiple evaluations of the model function in perturbed evaluation points. However, it potentially suffers from limited accuracy and high computational effort.

- *Automatic Differentiation (AD)* just as Symbolic Differentiation decomposes the function to be differentiated into a concatenation of certain elemental functions. Systematic application of the chain rule yields the derivative. As opposed to Symbolic Differentiation, where this process is applied to the symbolic expression tree, in AD it takes place while the function is evaluated in a given point. Early results on AD are published by Wengert [448] and Kedem [269]. A comprehensive reference about AD is provided by Griewank and Walther [210].

6.1.1 Automatic Differentiation

AD is based on *factorable programming* formulation where all functions are *factorable functions*, cf. McCormick [322], Shayan [409], and Jackson [255]. Factorable functions are defined by a recursive composition of elemental operators such as addition, subtraction, multiplication, division on the one hand and a given library of elemental functions such as exponential or trigonometric functions on the other hand. Every elemental function is locally an analytic function.

Evaluating a factorable function $\boldsymbol{f}(\cdot)$ in a point x to get $y = \boldsymbol{f}(x)$ requires subsequently calculating the composition of all elemental operations. This process can be interpreted descriptively as an evaluation graph where intermediate results appear as vertices and elemental operations as edges.

The computation of derivatives with this approach is based on the idea that all elemental functions are analytical functions. Analytical functions have by definition a convergent power series expansion. Likewise, compositions of convergent power series expansions yield a convergent power series expansion. For this reason the composition of power series expansions of analytical functions yields the power series expansion of their composition. Hence, the aforementioned evaluation graph can be extended to a lifted computational graph where the edges are given by compositions of power series and the nodes by intermediate power series. One obtains the original evaluation graph by projecting onto the constant coefficient.

Forward Mode In *forward mode* of AD the lifted computational graph is used and the power series expansion is truncated up to a certain order k. The evaluation of $\boldsymbol{f}(\cdot)$ in point $x + td$ yields $\sum_{i=0}^{k} \frac{\boldsymbol{f}^{(i)}(x)d^i}{i!} t^i$ where $\boldsymbol{f}^{(i)}(x)d^i$ denotes the i–fold contraction of $\boldsymbol{f}^{(i)}$ with d. The forward mode traverses the computational graph from independent to dependent variables while accumulating derivative information.

Forward/Reverse Mode The application of the reverse mode requires one forward sweep where the forward mode is used with truncation order k and input $x + td$. All intermediate results are stored on a *tape*. The reverse mode requires a direction y of dependent values as additional input and allows for the calculation of $\frac{\mathrm{d}}{\mathrm{d}x} \sum_{i=0}^{k+1} \frac{y^T \boldsymbol{f}^{(i)}(x)d^i}{i!} t^i$ by transversing the computational graph backwards from dependent to independent variables.

Both forward and forward/reverse mode only require a small multiple of the computational cost of evaluating $\boldsymbol{f}(x)$. The forward/reverse mode may additionally require significant amount of storage to store the tape. Using the AD approach allows for calculating the Taylor coefficients $\frac{\boldsymbol{f}^{(i)}(x)d^i}{i!}$ and $\frac{\mathrm{d}}{\mathrm{d}x} \frac{y^T \boldsymbol{f}^{(i)}(x)d^i}{i!}$ up to machine precision.

First–Order Derivatives In order to calculate Jacobians $\frac{\mathrm{d}f}{\mathrm{d}x}$ we use the forward mode with $k = 1$ and $d = e_i$ for all basis vectors e_i. The extraction of the first Taylor coefficient yields $\frac{\mathrm{d}f}{\mathrm{d}x} e_i$. There are n evaluations of forward directional derivatives required if $\boldsymbol{f}(\cdot)$ is just defined on a subset of $\mathbb{R}^n$.

Second–Order Derivatives In order to calculate Hessian–vector products $\frac{\mathrm{d}^2}{\mathrm{d}x^2}\lambda^T \boldsymbol{f} v$ we use the forward/reverse mode with $k = 1$, $d = v$ and $y = \lambda$. The extraction of the first Taylor coefficient yields the desired vector product.
For our implementations we use the AD–software package ADOL–C [445] to calculate first– and second–order derivatives. ADOL–C uses operator overloading techniques to provide Taylor coefficient propagation. It can be applied to functions defined in C++ that are composed from a collection of smooth elemental functions. It allows for the calculation of function derivatives up to machine precision.

6.1.2 IVPs and Sensitivity Generation

Direct transcription methods based on the control parametrization approach (see Section 6.2) require the solution $\boldsymbol{x}_n(\cdot\,; s_n, q_n)$ of IVPs

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), q_n), \quad \boldsymbol{x}(t_n) = s_n, \quad t \in [t_n, t_{n+1}], \tag{6.1}$$

evaluated at t_{n+1}, where q_n denotes a time–independent n_q–dimensional parameter vector and s_n the n_x–dimensional initial state. Hence, the solution to IVP (6.1) evaluated at a point $t \in [t_n, t_{n+1}]$ is a function of the $n_\mathrm{x} + n_\mathrm{q}$ variables $[s_n, q_n]^T$ and we denote it by $\boldsymbol{x}_n(s_n, q_n; t)$. The special case $t = t_{n+1}$ is denoted by $\boldsymbol{x}_n(s_n, q_n)$.
In order to solve NLPs arising from aforementioned direct transcription methods it is necessary to compute first– and second–order derivatives of the ODE solution $\boldsymbol{x}_n(s_n, q_n)$ with respect to the initial values s_n and parameters q_n.

Adaptive Discretization Schemes for ODE IVPs

We obtain numerical solutions of IVP (6.1) by using an (adaptive) discretization scheme for ODEs. Most schemes can be classified as *one–step methods* or *multistep methods*. As a representative for a class of one–step methods we will introduce Runge–Kutta methods. BDF methods will be discussed to show the principle of multistep methods.
No matter which discretization approach is used to determine approximate solutions of the IVP (6.1) they have in common a discretization of the time interval $\mathcal{T} = [t_n, t_{n+1}]$ by a partition or *grid*

$$\mathbb{G}_K \stackrel{\text{def}}{=} \{t_n = \tau^0 < \tau^1 < \ldots < \tau^K = t_{n+1}\}$$

with *grid points* τ^k, $k = 0, 1, \ldots, K$, $K \in \mathbb{N}$, and step sizes $h^k \stackrel{\text{def}}{=} \tau^{k+1} - \tau^k$, $k = 0, 1, \ldots, K-1$. An equidistant grid, i.e., a grid with step size $h^k \stackrel{\text{def}}{=} h \stackrel{\text{def}}{=} (t_{n+1} - t_n)/N$ for $k = 0, 1, \ldots, K-1$, seems to be a natural choice. However, adaptive grids may be a lot more efficient for practical applications.
A discretization scheme produces a *grid function* $\boldsymbol{x}_K : \mathbb{G}_K \longrightarrow \mathbb{R}^{n_\mathrm{x}}$ with $\tau \mapsto \boldsymbol{x}_K(\tau)$ for $\tau \in \mathbb{G}_K$ that approximates the solution $\boldsymbol{x}(\cdot)$ of IVP (6.1) on the grid $\mathbb{G}_K$, i.e., we have

$$\boldsymbol{x}_K(\tau^k) \simeq \boldsymbol{x}(\tau^k), \qquad k = 0, \ldots, K.$$

Since the grid function is only defined on the $K+1$ grid points in $\mathbb{G}_K$ we introduce values η^k such that

$$\eta^k \stackrel{\text{def}}{=} \boldsymbol{x}_K(\tau^k), \qquad k = 0, \ldots, K.$$

Most discretization schemes only use the η^k notation for their definitions. Nonetheless, the interpretation in terms of a grid function may be useful to analyze convergence of discretization methods. In particular it might be of some interest, whether the sequence $\{\boldsymbol{x}_K(\cdot)\}_{K\in\mathbb{N}}$ of grid functions converges to a solution $\boldsymbol{x}(\cdot)$ of IVP (6.1) if K tends to infinity, i.e., the grid size tends to zero. This requires the grid size

$$h \stackrel{\text{def}}{=} \max_{k=0,\ldots,K-1} h^k$$

to tend to zero. Since we use numerical integration methods only like a tool from a toolbox in this thesis we just outline the basics about them in the following. In order to gain a deeper insight into this topic we refer the reader to Gear [183], Gear and Petzold [184], Hairer et al. [219], Shampine [405], Stoer et al. [419] and the references therein. Specific details about one–step methods can be found in Brenan and Petzold [86], Hairer et al. [218]. Curtiss and Hirschfelder [119], Geart and Watanabe [185], Lötstedt and Petzold [301], Brenan and Engquist [85], Beigel et al. [42] deal in particular with multistep methods.

One–Step Methods For the definition of one–step methods and the subsequent definition of Runge–Kutta methods we suppress the parameter q_n in the ODE of IVP (6.1) in favor of an increased readability.

Definition 6.1 (One–Step Method)
Let (t_n, s_n) be an initial time and value for IVP (6.1) and let $\{\tau^k\}_{k=0}^K$ define a discretization grid. Given a function $\boldsymbol{\Phi}(t,h,x)$, a *one–step method* defines a sequence of approximations $\{\eta^k\}$ to the exact solutions $\{\boldsymbol{x}_n(\tau^k; s_n)\}$ of IVP (6.1) recursively as

$$\eta^{k+1} \stackrel{\text{def}}{=} \eta^k + h^k \boldsymbol{\Phi}(\tau^k, h^k, \eta^k), \qquad h^k \stackrel{\text{def}}{=} \tau^{k+1} - \tau^k, \qquad k = 0, \ldots, K-1.$$

The one–step method is started with $(\tau^0, \eta^0) \stackrel{\text{def}}{=} (t_n, s_n)$. △

One–step methods differ in the choice of their *generating function* $\boldsymbol{\Phi}(\cdot)$. A popular family of one–step methods is called the s–stage Runge–Kutta scheme.

Definition 6.2 (Runge–Kutta Method)
A one–step method is called Runge–Kutta *method* with $s \in \mathbb{N}$ stages if its generating function has the form

$$\boldsymbol{\Phi}(t,h,x) \stackrel{\text{def}}{=} \sum_{i=1}^{s} c_i k_i, \quad k_i \stackrel{\text{def}}{=} \boldsymbol{f}\left(t + \alpha_i h, x + h \sum_{j=1}^{s} B_{i,j} k_j\right),$$

where coefficients $c = [c_1, \ldots, c_s]^T \in \mathbb{R}^s$, $\alpha = [\alpha_1, \ldots, \alpha_s]^T \in \mathbb{R}^s$ and $B = \left(B_{i,j}\right)_{i,j} \in \mathbb{R}^{s\times s}$ are chosen suitably. △

A Runge–Kutta scheme is called *explicit* if $B_{i,j} = 0$ for $j \geqslant i$ and *implicit* otherwise. Three common examples of s–stage Runge–Kutta schemes are

- Euler Method (explicit, $s = 1$)

$$\eta^{k+1} = \eta^k + h^k \boldsymbol{f}(\tau^k, \eta^k)$$

- Trapezoidal Method (implicit, $s = 2$)

$$\eta^{k+1} = \eta^k + \frac{h^k}{2}\left(\boldsymbol{f}(\tau^k, \eta^k) + \boldsymbol{f}(\tau^{k+1}, \eta^{k+1})\right) \tag{6.2}$$

- Classical Runge–Kutta Method (explicit, $s = 4$)

$$\begin{aligned} k_1 &= h^k \boldsymbol{f}(\tau^k, \eta^k) \\ k_2 &= h^k \boldsymbol{f}(\tau^k + \frac{h^k}{2}, \eta^k + \frac{1}{2}k_1) \\ k_3 &= h^k \boldsymbol{f}(\tau^k + \frac{h^k}{2}, \eta^k + \frac{1}{2}k_2) \\ k_4 &= h^k \boldsymbol{f}(\tau^{k+1}, \eta^k + k_3) \\ \eta^{k+1} &= \eta^k + \frac{1}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right) \end{aligned}$$

Multistep Methods The following definition states the general form of Linear Multistep Methods with variable order and variable step size.

Definition 6.3 (Linear Multistep Method)
Let $\{\tau^k\}_{k=0}^K$ define a discretization grid. For given start values $\eta^1, \ldots, \eta^m$ with a fixed m the *Linear Multistep Method (LMM)* calculates a sequence of approximations $\{\eta^k\}_{k=m+1}^K$ to the exact solution $\{\boldsymbol{x}_n(\tau^k; s_n)\}$ of IVP (6.1) as

$$\sum_{i=0}^{l_k} \alpha_i^{(k)} \eta^{k+1-i} = h^k \sum_{i=0}^{l_k} \beta_i^{(k)} \boldsymbol{f}(\tau^{k+1-i}, \eta^{k+1-i}), \quad k = m, \ldots, K-1$$

with $\alpha_0^{(k)} \neq 0$ and $\left|\alpha_i^{(k)}\right| + \left|\alpha_i^{(k)}\right| > 0$. We call the LMM *explicit* if $\beta_0^k = 0$ and *implicit* if $\beta_0^k \neq 0$. △

Well–established members of the multistep class are given by the Adams schemes whose basic idea rest on approximating $\boldsymbol{f}(\cdot)$ by interpolating polynomials (see Appendix B.2). An explicit representative is given by the Adams–Bashforth method (see Bashforth and Adams [31]), whereas an implicit one can be found in the Adams–Moulton method (see Moulton [331]). BDF methods were invented by Curtiss and Hirschfelder [119] and belong to the class of implicit LMMs. Particularly with regard to their effectiveness in solving stiff IVPs they became very popular with the work of Gear [183] in 1971. The basic idea of BDF methods can be described as follows: an interpolating polynomial that is constructed from past approximations and an unknown new approximation has to be determined such that it satisfies the ODE

in the new time point. The method requires information at previous points and this implies that some method has to be used to launch the process. The application of several steps with a one–step method is a common approach. A respective strategy using a Runge–Kutta method is then called Runge–Kutta starter, cf. Bauer [32].

Definition 6.4 (Backward Differentiation Formula Method)
Let $\{\tau^k\}_{k=0}^K$ define a discretization grid and let a starting procedure determine start values $\eta^1, \ldots, \eta^m$. The *Backward Differentiation Formula (BDF) method* calculates approximations $\{\eta^k\}_{k=m+1}^K$ to the exact solution $\{\boldsymbol{x}_n(\tau^k; s_n)\}$ of IVP (6.1) as

$$\sum_{i=0}^{l_k} \alpha_i^{(k)} \eta^{k+1-i} = h^k \boldsymbol{f}(\tau^{k+1}, \eta^{k+1}), \quad k = m, \ldots, K-1 \tag{6.3}$$

with step sizes $h^k = \tau^{k+1} - \tau^k$ and orders l_k. By means of the Lagrange polynomials

$$\boldsymbol{L}_i^{(k)}(\tau) \stackrel{\text{def}}{=} \prod_{\substack{j=0 \\ j \neq i}}^{l_k} \frac{\tau - \tau^{k+1-j}}{\tau^{k+1-i} - \tau^{k+1-j}}$$

the coefficients $\alpha_i^{(k)}$ are determined by

$$\alpha_i^{(k)} = h^k \dot{\boldsymbol{L}}_i^{(k)}(\tau^{k+1}). \tag{6.4}$$

△

Due to their definition BDF methods are implicit LMMs since $\beta_0^{(k)} = 1$ and $\beta_i^{(k)} = 0$ for $i = 1, \ldots, l_k$. In our definition of BDF methods the step size and order may vary. Thus, it is necessary for implementations to take care for both efficient and well–conditioned coefficient calculations.
Alternatively to Runge–Kutta starters one could also choose a so–called *self–starting procedure* in Definition 6.4: it begins with setting $l_0 = 1$ in (6.3) and posing additionally the initial constraint $\boldsymbol{x}(\tau^0) = s_n$ what results in an implicit Euler step. Then the integration step order l_k is successively increased until the maximum order is reached.
Due to the fact that the BDF method is implicit and the function $\boldsymbol{f}(\cdot)$ is nonlinear a system of nonlinear equations has to be solved in each iteration step (6.3). If step size h^k and order l_k are chosen in (6.3) such that

$$\left| \frac{h^k}{\alpha_0^k} \right| \cdot \left\| \boldsymbol{f}_x'(\tau^{k+1}, \eta^{k+1}) \right\| < 1$$

holds then (6.3) possesses a unique solution η^{k+1}, cf. Henrici [229]. Approximations of η^{k+1} are usually obtained by applying Newton–type methods.

Automatic Step Size Selection Regardless of using either a one–step or multistep method the accuracy of the solution has to be addressed, i.e., how well is the real solution $\{\boldsymbol{x}_n(\tau^k; s_n)\}$ approximated by the discrete solution $\{\eta^k\}$. This topic has undergone intensive research and produced some mechanism for adjusting the integration step size and order to control the

integration error, see e.g. Dormand and Prince [138], Fehlberg [156, 157], Verner [435], Shampine and Gordon [407], Gear [183].

In order to evaluate the solution trajectory $\boldsymbol{x}(\cdot)$ approximately at points independent of the discretization grid $\{\tau^k\}$ a continuous representation of the discrete values $\{\eta^k\}$ is required. This issue is usually addressed by using interpolation polynomials (Appendix B.2). Further details can be found in Enright [147], Owren and Zennaro [349], Shampine [404] and the references therein.

Sensitivities of Initial Value Problems

Now we deal with numerical approaches for the computation of IVP sensitivities, i.e., we approximate the derivative of $\boldsymbol{x}_n(s_n, q_n)$ with respect to the initial value s_n and parameters q_n. More precisely we are interested in constructing the matrix $X_n = \left[X_n^s, X_n^q\right]$ with

$$X_n^s \stackrel{\text{def}}{=} \left[\frac{\partial}{\partial (s_n)_j}(\boldsymbol{x}_n)_i(s_n, q_n)\right]_{i,j\in[n_\mathrm{x}]}, \quad X_n^q \stackrel{\text{def}}{=} \left[\frac{\partial}{\partial (q_n)_j}(\boldsymbol{x}_n)_i(s_n, q_n)\right]_{i\in[n_\mathrm{x}],j\in[n_\mathrm{q}]}$$

For the sake of simplicity we describe approaches based on finite differences in the following but mention extensions to AD if applicable. If we define perturbation vectors

$$d_k^{s\,T} \stackrel{\text{def}}{=} [0, \ldots, 0, \delta_k^s, 0, \ldots, 0]^T, \quad d_l^{q\,T} \stackrel{\text{def}}{=} [0, \ldots, 0, \delta_l^q, 0, \ldots, 0]^T$$

for $k \in [n_\mathrm{x}]$ and $l \in [n_\mathrm{q}]$, then a forward difference approximation to column k of X_n^s and column l of X_n^q is of the form

$$\left(X_n^s\right)_{\cdot,k} \simeq \frac{1}{\delta_k^s}\left[\boldsymbol{x}_n(s_n + d_k^s, q_n) - \boldsymbol{x}_n(s_n, q_n)\right], \tag{6.5}$$

$$\left(X_n^q\right)_{\cdot,l} \simeq \frac{1}{\delta_l^q}\left[\boldsymbol{x}_n(s_n, q_n + d_l^q) - \boldsymbol{x}_n(s_n, q_n)\right]. \tag{6.6}$$

There are several ways how approximations $\boldsymbol{\eta}_n(s_n, q_n)$ of the *nominal point* $\boldsymbol{x}_n(s_n, q_n)$ and its perturbations $\boldsymbol{\eta}_n(s_n + d_k^s, q_n) \simeq \boldsymbol{x}_n(s_n + d_k^s, q_n)$ and $\boldsymbol{\eta}_n(s_n, q_n + d_l^q) \simeq \boldsymbol{x}_n(s_n, q_n + d_l^q)$ can be calculated using adaptive discretization schemes as introduced in the previous section. We pick three of them, discuss their basic ideas and present respective benefits and drawbacks.

External Numerical Differentiation *External Numerical Differentiation (END)* calculates all required values $\boldsymbol{\eta}_n(s_n, q_n)$, $\boldsymbol{\eta}_n(s_n + d_k^s, q_n)$ and $\boldsymbol{\eta}_n(s_n, q_n + d_l^q)$ separately and applies the forward difference rules (6.5)+(6.6) afterwards. The approximation of X_n^s using END can be described as follows:

- **Nominal Point**
  Compute $\boldsymbol{\eta}_n(s_n, q_n)$: integrate ODE (6.1) from t_n to t_{n+1} with initial conditions s_n
- **Perturbations** For $k = 1, \ldots, n_x$
  Set $\bar{s} = s_n + d_k^s$
  Compute $\boldsymbol{\eta}_n(\bar{s}, q_n)$: integrate ODE (6.1) from t_n to t_{n+1} with initial conditions $\bar{s}$
- **Derivative Evaluation** For $k = 1, \ldots, n_x$
  Construct $\left(X_n^s\right)_{\cdot,k}$ according to (6.5)

While varying the IVP initial conditions the END approach treats the IVP solution method as "black box" to which finite differences are applied *externally*. If the discretization scheme, which generates the approximation $\boldsymbol{\eta}_n(\cdot)$, would be a composition of smooth functions one could also consider applying AD techniques to the composed function in order to calculate sensitivities. However, for most numerical codes this assumption is not satisfied.
It is a well–known fact that adaptive discretization schemes result in composed functions that are in general non–smooth with respect to initial values s_n and parameters q_n. If they are varied there are jumps of the order of the integrator tolerance possible in the output. The discrete character of the step size and order selection strategy introduces the non–smoothness in a natural way: a change in the input parameters (s_n, q_n) may cause the algorithm to follow a different path of execution, e.g. there are possibly different choices of the discretization grid $\{\tau_k\}$. In particular this holds for nominal initial values (s_n, q_n) and its perturbed values $(s_n + d_k^s, q_n)$, $(s_n, q_n + d_k^q)$.
The END approach is easy to implement, but it has the disadvantage that END computed derivatives are not consistent in the sense that increasing the integrator accuracy does not necessarily result in a convergence to the exact derivatives. Furthermore the non–smooth behavior of $\boldsymbol{\eta}_{n+1}(\cdot)$ induces inaccuracies of the finite difference approximations (6.5)+(6.6), unless the integrations are performed with extraordinary high accuracy.

Internal Numerical Differentiation We identified the non–smooth output caused by adaptive discretization schemes as one main issue of END. This can be overcome by the *Internal Numerical Differentiation (IND)* approach whose idea is presented in the following.
The basic concept of IND is to compute the derivative of the adaptive discretization scheme itself and not by trying to differentiate the integrator output. The original idea of IND was proposed by Bock [71, 72], where an adaptive discretization scheme is applied to the nominal trajectory. Then the discretization scheme is "frozen" for the integration of the perturbed trajectories. As a consequence the output becomes a differentiable function of the inputs. In contrast to END the discretization scheme can not exist as a "black box" but one needs some insights to control the sequence of orders and step sizes.
Another possible way to realize the IND principle, that allows for reusing many matrix evaluations and factorizations, is not to integrate the nominal and perturbed trajectories one at a time, but all of them at the same time. The numerical integration method usually just con-

trols the error with respect to the nominal trajectory but not with respect to the perturbed trajectories. Analogously to the description of END we direct our attention to the case of approximating X_n^s since the extension to X_n^q is straightforward.
We define an augmented system of ODEs by making $1 + n_x$ copies of (6.1)

$$\dot{\boldsymbol{y}}^k(t) = \boldsymbol{f}(t, \boldsymbol{y}^k(t), q_n), \quad k = 0, 1, \ldots, n_x. \tag{6.7}$$

By construction the augmented system (6.7) involves $n_x(1+n_x)$ differential variables, i.e., $\boldsymbol{y}_n(\cdot)$ is the $n_x(1 + n_x)$–dimensional vector $\boldsymbol{y}_n{}^T \stackrel{\text{def}}{=} [\boldsymbol{y}_n^{0\,T}, \ldots, \boldsymbol{y}_n^{n_x T}] = [\boldsymbol{x}_n{}^T, \ldots, \boldsymbol{x}_n{}^T]$. The initial conditions for the augmented system then read as

$$\boldsymbol{y}_n{}^T(t_n) = [s_n{}^T, (s_n + d_1^s)^T, \ldots, (s_n + d_{n_x}^s)^T]. \tag{6.8}$$

Note that there is the following relationship between the original and the augmented system:

$$[\boldsymbol{y}_n^{0\,T}, \boldsymbol{y}_n^{1\,T}, \ldots, \boldsymbol{y}_n^{n_x T}]^T(t_{n+1}) = [\boldsymbol{x}_n(s_n, q_n)^T, \boldsymbol{x}_n(s_n + d_1^s, q_n)^T, \ldots, \boldsymbol{x}_n(s_n + d_{n_x}^s, q_n)^T]^T.$$

Similar to END we obtain approximations of $\boldsymbol{x}_n(s_n, q_n)$ and the $\boldsymbol{x}_n(s_n + d_k^s, q_n)$ by applying numerical integration methods and finally we can state the following IND algorithm:

- **Propagation** Compute $\boldsymbol{\eta}_n(s_n, q_n)$, $\boldsymbol{\eta}_n(s_n + d_1^s, q_n), \ldots, \boldsymbol{\eta}_n(s_n + d_{n_x}^s, q_n)$: integrate the augmented ODE (6.7) from t_n to t_{n+1} with initial conditions (6.8)
- **Derivative Evaluation** For $k = 1, \ldots, n_x$ Construct $\left(X_n^s\right)_{\cdot,k}$ according to (6.5)

Derivatives generated according to the IND principle are *consistent* with the discretization scheme. Since IND delivers the "exact" derivative of an adaptive discretization scheme, it is stable even for low integration accuracies. The accuracy of the numerical integration can therefore be chosen to be of the same order of magnitude as the one required for the sensitivities. This is opposed to our respective comments about END.
The IND principle does not rely on a specific discretization scheme and has especially been implemented for one–step and multistep methods. It can also be applied in an AD context. More details about arbitrary–order forward and forward/reverse mode sensitivity generation using IND involving a variable order and variable step size *BDF method* as discretization scheme can be found in Albersmeyer [8]. An implementation thereof is DAESOL-II which is part of the C++ package SolvIND. SolvIND is an ODE/DAE solver suite that allows for generating forward and forward/reverse mode sensitivity information of the solutions using IND. The generation of derivatives of the model functions is done via built–in ADOL-C support.

Variational Differential Equation The *sensitivity equation approach* or *VDE approach* represents another way how X_n^s and X_n^q can be calculated. Let us consider the time dependent matrix–valued functions

$$\boldsymbol{X}_n^s(t) \stackrel{\text{def}}{=} \left[\frac{\partial}{\partial (s_n)_j}(\boldsymbol{x}_n)_i(s_n, q_n; t)\right]_{i,j\in[n_x]}, \quad \boldsymbol{X}_n^q(t) \stackrel{\text{def}}{=} \left[\frac{\partial}{\partial (q_n)_j}(\boldsymbol{x}_n)_i(s_n, q_n; t)\right]_{i\in[n_x],j\in[n_q]},$$

describing the derivatives of the state at time instant t with respect to s_n and q_n. By construction we have $\boldsymbol{X}_n^s(t_{n+1}) = X_n^s$ and $\boldsymbol{X}_n^q(t_{n+1}) = X_n^q$. Differentiating $\boldsymbol{X}_n^s(\cdot)$ and $\boldsymbol{X}_n^q(\cdot)$ leads to the VDEs

$$\dot{\boldsymbol{X}}_n^s(t) = \boldsymbol{f}_x^{'}(t, \boldsymbol{x}_n(s_n, q_n; t), q_n)\, \boldsymbol{X}_n^s(t), \tag{6.9}$$

$$\dot{\boldsymbol{X}}_n^q(t) = \boldsymbol{f}_x^{'}(t, \boldsymbol{x}_n(s_n, q_n; t), q_n)\, \boldsymbol{X}_n^q(t) + \boldsymbol{f}_q^{'}(t, \boldsymbol{x}_n(s_n, q_n; t), q_n). \tag{6.10}$$

The initial conditions for the VDEs are

$$\boldsymbol{X}_n^s(t_n) = \boldsymbol{I}, \qquad \boldsymbol{X}_n^q(t_n) = \boldsymbol{0}.$$

By defining the block matrix

$$\boldsymbol{X}_n(t) \stackrel{\text{def}}{=} \begin{bmatrix} \boldsymbol{X}_n^s(t) & \boldsymbol{X}_n^q(t) \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} \tag{6.11}$$

we can write the IVP (6.9)+(6.10)+(6.11) as linear ODE

$$\dot{\boldsymbol{X}}_n(t) = \begin{bmatrix} \boldsymbol{f}_x^{'}(t, \boldsymbol{x}_n(s_n, q_n; t), q_n) & \boldsymbol{f}_q^{'}(t, \boldsymbol{x}_n(s_n, q_n; t), q_n) \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \boldsymbol{X}_n(t), \tag{6.12}$$

together with initial conditions

$$\boldsymbol{X}_n(t_n) = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix}. \tag{6.13}$$

It is obvious that the principle of IND can be applied using the VDE: it is possible to set up the VDE and solve it along with the nominal system, i.e., we can use the order and step size sequence of the nominal solution.

6.2 Reduced Discretization Approach

We start our investigations about direct transcription methods with methods that discretize the controls exclusively, and propagate the dynamics across the horizon using the control approximation. The process of choosing an initial value for the ODE such that the dynamics satisfies the boundary conditions after propagating the dynamics reminds vaguely of aiming a cannon such that the cannonball hits its target. That is why these methods are called "*shooting methods*".

Figure 6.2 shows that two types of shooting methods are distinguished. In this thesis we just deal with the multiple shooting method. But since the single shooting method forms the basis

for the multiple shooting method we present both of them.

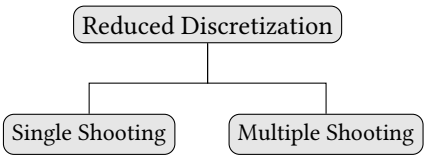


Figure 6.2: Overview: Direct Methods based on control discretization.

All shooting methods have in common that they use an embedded ODE solver in order to eliminate the continuous time dynamics. Relevant results, which includes references and further reading, have been presented in the previous section.
The control parametrization approach is presented by reference to the OCP

$$\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)} \quad \varphi(t_\mathrm{f},\boldsymbol{x}(t_\mathrm{f})) + \int_{t_\mathrm{s}}^{t_\mathrm{f}} \psi(t,\boldsymbol{x}(t),\boldsymbol{u}(t))\,\mathrm{d}t \tag{6.14a}$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t,\boldsymbol{x}(t),\boldsymbol{u}(t)), \quad t \in \mathcal{T}, \tag{6.14b}$$

$$\boldsymbol{x}(t_\mathrm{s}) = x_\mathrm{s}, \tag{6.14c}$$

$$\mathbf{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(t,\boldsymbol{x}(t),\boldsymbol{u}(t)), \quad t \in \mathcal{T}, \tag{6.14d}$$

$$\mathbf{0}_{n_\mathrm{r}} \geqslant \boldsymbol{r}(t_\mathrm{f},\boldsymbol{x}(t_\mathrm{f})). \tag{6.14e}$$

6.2.1 Control Discretization

The aim of this section is to introduce ways how the (infinite dimensional) space of feasible control functions $\boldsymbol{u}(\cdot)$ can be approximated by a finite dimensional subspace. We start by partitioning the control horizon $\mathcal{T}$ into N (not necessarily equidistant) intervals

$$t_\mathrm{s} = t_0 < t_1 < \ldots < t_N = t_\mathrm{f}$$

such that the $\{t_n\}$ define a so–called *shooting grid*. For each component $1 \leqslant i \leqslant n_\mathrm{u}$ of the control trajectory $\boldsymbol{u}(\cdot)$ and on each interval $[t_n, t_{n+1}]$, $0 \leqslant n \leqslant N-1$, we choose a vector of base functions $\xi_n(t,q_n) = \left[\left(\xi_n\right)_1(t,q_n^1),\ldots,\left(\xi_n\right)_{n_\mathrm{u}}(t,q_n^{n_\mathrm{u}})\right]^T$ with $q_n \overset{\text{def}}{=} \left[{q_n^1}^T,\ldots,{q_n^{n_\mathrm{u}}}^T\right]^T$ and $\left(\xi_n\right)_i : \mathcal{T} \times \mathbb{R}^{n_\mathrm{q}^{ni}} \longrightarrow \mathbb{R}$. It is often desirable to guarantee separability (see page 161) of the discretization over shooting grid intervals. For this reason the $\left(\xi_n\right)_i(\cdot)$ are chosen to have local support and they are parameterized by a vector of finitely many control parameters $q_n^i \in \mathbb{R}^{n_\mathrm{q}^{ni}}$. Popular choices for base functions are

- **piecewise constant controls** ($n_\mathrm{q}^{ni} = 1$):

$$\left(\xi_n\right)_i : [t_n, t_{n+1}] \times \mathbb{R}^{n_\mathrm{q}^{ni}} \longrightarrow \mathbb{R}, \quad (t, q_n^i) \mapsto q_n^i$$

- **piecewise linear controls** ($n_{\mathrm{q}}^{ni} = 2$):

$$\big(\xi_n\big)_i : [t_n, t_{n+1}] \times \mathbb{R}^{n_{\mathrm{q}}^{ni}} \longrightarrow \mathbb{R}, \quad (t, q_n^i) \mapsto \frac{t_{n+1} - t}{t_{n+1} - t_n} \big(q_n^i\big)_1 + \frac{t - t_n}{t_{n+1} - t_n} \big(q_n^i\big)_2$$

- **piecewise cubic spline controls** ($n_{\mathrm{q}}^{ni} = 4$):

$$\big(\xi_n\big)_i : [t_n, t_{n+1}] \times \mathbb{R}^{n_{\mathrm{q}}^{ni}} \longrightarrow \mathbb{R}, \quad (t, q_n^i) \mapsto \sum_{k=1}^{4} \big(q_n^i\big)_k \beta_k \left(\frac{t - t_n}{t_{n+1} - t_n} \right)^{k-1},$$

  where the spline function coefficients β_k are chosen appropriately.

For each of the n_{u} control trajectory components there may be chosen different discretization types. For certain control discretization choices such as piecewise linear controls it can be desired that the discretized control trajectory is continuous over the complete control horizon. This can be realized for the control trajectory component $\boldsymbol{u}_i(\cdot)$ by imposing additional control continuity conditions

$$0 = \big(\xi_n\big)_i (t_{n+1}, q_n^i) - \big(\xi_{n+1}\big)_i (t_{n+1}, q_{n+1}^i)$$

for all points of the control discretization grid $\{t_n\}$, $n \in [N-1]$. As it is shown e.g. by Kirches et al. [273] the control discretization choice may have some impact on the approximation quality of the discretized OCP.

6.2.2 Direct Single Shooting

The *direct single shooting* method, earliest presented by Hicks and Ray [236], Sargent and Sullivan [385], first parametrizes the control function $\boldsymbol{u}(\cdot)$ with techniques as presented in Section 6.2.1. We denote the control parametrization by $\xi(\,\cdot\,; q)$, where q denotes the parameter to be determined by the optimization.
For convenience we introduce the single shooting method for the choice of piecewise constant controls. For the temporal grid $t_{\mathrm{s}} = t_0 < t_1 < \ldots < t_N = t_{\mathrm{f}}$ we define parameters $q_n \in \mathbb{R}^{n_{\mathrm{u}}}$, $n \in [N]$. Then we set the control parametrization as

$$\xi(t; q) \overset{\text{def}}{=} q_n, \quad \text{if } t \in [t_n, t_{n+1}), \quad 0 \leqslant n \leqslant N-1.$$

Hence, the dimension of the parameter vector $q \overset{\text{def}}{=} [{q_1}^T, \ldots, {q_N}^T]^T$ is $N \cdot n_{\mathrm{u}}$. For completeness, the control at the final time is defined as $\xi(t_{\mathrm{s}}; q) \overset{\text{def}}{=} q_N \overset{\text{def}}{=} q_{N-1}$, where the vector q_N is introduced for notational convenience exclusively and will not regarded as an additional parameter, but just as another name for q_{N-1}.
In direct single shooting the states $\boldsymbol{x}(\cdot)$ are regarded as dependent variables of the controls $\boldsymbol{u}(\cdot)$ respectively their parametrization $\xi(\,\cdot\,; q)$ together with the initial state s_0, i.e., the states are obtained by a forward integration of the IVP

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \xi(t; q)), \quad t \in \mathcal{T}, \tag{6.15a}$$

$$\boldsymbol{x}(t_s) = s_0. \tag{6.15b}$$

The objective function as well as control and path constraints are usually discretized and enforced only on the control discretization grid $\{t_n\}$.
With this approach we obtain a NLP in the $n_x + N \cdot n_u$ unknowns $[{s_0}^T, {q_0}^T, \ldots, {q_{N-1}}^T]^T$ which can be solved e.g. by SQP techniques as introduced in Section 3.6. It is obvious that IVP (6.15) has the form as IVP (6.1) and thus sensitivities as required by NLP solvers can be determined with techniques from Section 6.1.2.

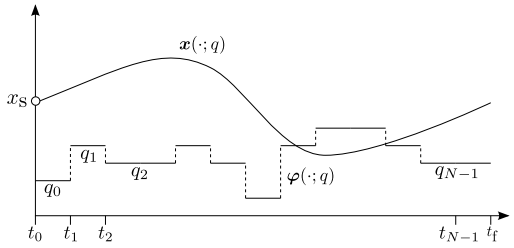


Figure 6.3: Illustration of the direct single shooting discretization applied to the optimal control problem.

The direct single shooting methods has some advantages over other methods: the initialization of the NLP variables is restricted to the initial state s_0 and the control parameters q. If e.g. the END approach is used for the sensitivity generation any state–of–the–art ODE solver can be used for the forward integration and derivative calculations of the arising IVPs. Even for large ODE systems the resulting NLP has few degrees of freedom.
But the direct single shooting method also suffers from several severe drawbacks: since the NLP variables are restricted to the control parameters the initialization of the state trajectory - based on prior knowledge about the dynamic process - is impossible. The main drawback of the single shooting method is the potential infeasibility of the numerical integration, which might break down in the course of the integration. This can happen due to a very stiff or unstable set of differential equations or induced by a singularity in time. Usually the existence of a numerical solution of highly nonlinear and unstable OCPs can only be guaranteed for a parameter initialization that is already very close to the optimal solution. The ODE solution $\boldsymbol{x}(\cdot; s_0, q)$ can depend very nonlinearly on s_0 and q. Finally, the convergence of the NLP solver is effectively influenced by the nonlinearity of the underlying ODE system.
Nevertheless, the direct single shooting method is often used in practice.

6.2.3 Direct Multiple Shooting

The *direct multiple shooting* method was originally developed by Bock and Plitt [75] and Plitt [356] and can be seen as an extension of the direct single shooting method as introduced

in the previous section. The optimal control package MUSCOD-II, which was implemented by Leineweber [291], is based on the multiple shooting method.
Roughly speaking, in a direct multiple shooting approach the horizon interval $\mathcal{T}$ is split into N subintervals and then the single shooting method is applied on each subinterval independently. Here the continuity of the state trajectories is guaranteed by augmenting the resulting NLP with additional continuity constraints.

OCP Discretization

Let a multiple shooting grid be defined by $t_s = t_0 < t_1 < \ldots < t_N = t_f$.

Control Discretization The control discretization is performed in the same way as in the single shooting method. The control approximation is defined with local support as

$$\xi(t;q) \stackrel{\text{def}}{=} \xi_n(t;q_n), \quad \text{if } t \in [t_n, t_{n+1}), \quad 0 \leqslant n \leqslant N-1.$$

For completeness, we introduce an additional discretized control $\xi_N(t;q_N)$ for the final point t_f. It is defined to have the final control value of the previous shooting interval,

$$\xi_N(t_f;q_N) \stackrel{\text{def}}{=} \xi_{N-1}(t_f;q_{N-1}).$$

State Parametrization In contrast to the single shooting method the ODE system is not solved over the complete horizon $\mathcal{T}$ but on each subinterval $[t_n, t_{n+1}]$ separately. This can be realized by introducing artificial initial values $s_n \in \mathbb{R}^{n_x}$ and solving the N IVPs

$$\dot{\boldsymbol{x}}_n(t) = \boldsymbol{f}(t, \boldsymbol{x}_n(t), \xi_n(t;q_n)), \quad t \in [t_n, t_{n+1}], \quad 0 \leqslant n \leqslant N-1, \tag{6.16a}$$
$$\boldsymbol{x}_n(t_n) = s_n, \tag{6.16b}$$

The solutions of IVPs (6.16) are N independent trajectories $\boldsymbol{x}_n(\cdot)$ on $[t_n, t_{n+1}]$, which are functions of s_n and q_n exclusively. For this reason we denote the solution trajectories of IVP (6.16) on $[t_n, t_{n+1}]$ often with $\boldsymbol{x}_n(\cdot\,; s_n, q_n)$.
If we substitute the independent trajectories $\boldsymbol{x}_n(\cdot)$ into the Lagrange term $\boldsymbol{\psi}(\cdot)$ in (6.14a) the interval wise integral objective contributions $\boldsymbol{\psi}_n(s_n, q_n)$, defined by

$$\boldsymbol{\psi}_n(s_n, q_n) \stackrel{\text{def}}{=} \int_{t_n}^{t_{n+1}} \boldsymbol{\psi}(t, \boldsymbol{x}_n(t; s_n, q_n), \xi_n(t;q_n))\, \mathrm{d}t, \quad 0 \leqslant n \leqslant N-1,$$

can be calculated simultaneously.
By introducing the values s_n we added non–physical degrees of freedom that have to be removed by adding appropriate constraints: continuity of the state trajectories can be enforced by introducing so–called *matching conditions*, i.e., each *node value* s_n should be equal to the final value of the preceding trajectory $\boldsymbol{x}_n(\cdot\,; s_n, q_n)$:

$$\boldsymbol{0} = \boldsymbol{x}_n(t_{n+1}; s_n, q_n) - s_{n+1}, \quad 0 \leqslant n \leqslant N-1. \tag{6.17}$$

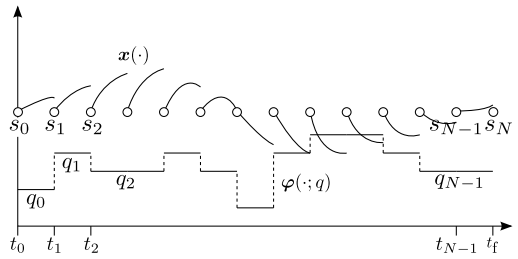


Figure 6.4: Illustration of the direct multiple shooting discretization applied to the optimal control problem. All shooting nodes were initialized identically and the solution of the N IVPs violates the matching conditions.

For the initial condition (6.14c) we require that the first node s_0 is equal to the initial value x_s:

$$s_0 = x_s. \tag{6.18}$$

The additionally introduced degrees of freedom represented by the parameters s_n, $0 \leqslant n \leqslant N$, are therefore removed by the constraints (6.17)+(6.18). It is not required that these constraints must be fulfilled during the optimization process, but rather it is one crucial strength of the direct multiple shooting method that infeasible initial guesses of the variables s_n can be handled.

Discretization of Path Constraints The infinite–dimensional mixed control–state inequality constraints (6.14d) are enforced to hold at the multiple shooting grid points $\{t_n\}$ which results in the $N+1$ inequality constraints

$$\mathbf{0} \geqslant \boldsymbol{c}(t_n, s_n, \xi_n(t_n; q_n)), \quad 0 \leqslant n \leqslant N.$$

Since the path constraints must hold only at shooting grid points and not over the full horizon the feasible set of the discretized OCP is usually enlarged compared to that of the continuous one. The multiple shooting method applied to real world problems shows in general at most mild violations of path constraints in the interior of shooting intervals. If strict violations are observed one could either restart the optimization process with an adapted and possibly finer shooting grid, or could use a semi–infinite programming approach which tracks constraint violations in the interior of shooting intervals. The interested reader can find a detailed description of this approach in the contributions of Potschka [358] and Potschka et al. [359].

Structured NLP

Multiple Shooting NLP The multiple shooting parametrized OCP (6.14) reads as

$$\min_{s_n, q_n} \quad \sum_{n=0}^{N} \psi_n(s_n, q_n) \tag{6.19a}$$

$$\begin{aligned}
\text{s.t.} \quad & \mathbf{0}_{n_x} = x_s - s_0, && (6.19b)\\
& \mathbf{0}_{n_x} = \boldsymbol{x}_n(t_{n+1}; s_n, q_n) - s_{n+1}, \quad 0 \leqslant n \leqslant N-1, && (6.19c)\\
& \mathbf{0}_{n_c} \geqslant \boldsymbol{c}(t_n, s_n, \xi_n(t_n; q_n)), \quad 0 \leqslant n \leqslant N, && (6.19d)\\
& \mathbf{0}_{n_r} \geqslant \boldsymbol{r}(t_N, s_N), && (6.19e)
\end{aligned}$$

where the Mayer term $\varphi(t_N, s_N)$ is written as final term $\psi_N(s_N, q_N)$. By defining the vectors $w_n \overset{\text{def}}{=} [{s_n}^T, {q_n}^T]^T$ and $w \overset{\text{def}}{=} [{w_0}^T, \ldots, {w_N}^T]^T$ we can write NLP (6.19) in a compact form as

$$\begin{aligned}
\min_w \quad & \boldsymbol{F}(w) && (6.20)\\
\text{s.t.} \quad & \mathbf{0} = \boldsymbol{G}(w) + [\boldsymbol{I}_{n_x} \mathbf{0} \ldots \mathbf{0}]^T x_s,\\
& \mathbf{0} \geqslant \boldsymbol{H}(w),
\end{aligned}$$

where $\boldsymbol{F}(w) \overset{\text{def}}{=} \sum_{n=0}^{N} \psi_n(s_n, q_n)$ denotes the objective function.

NLP Constraints The equality constraints (6.19b)+(6.19c) are collected into a function

$$\boldsymbol{G}(w) \overset{\text{def}}{=} \begin{bmatrix} -s_0 \\ \boldsymbol{x}_0(t_1; s_0, q_0) - s_1 \\ \vdots \\ \boldsymbol{x}_{N-1}(t_N; s_{N-1}, q_{N-1}) - s_N \end{bmatrix},$$

and the inequality constraints (6.19d) into a function

$$\boldsymbol{H}(w) \overset{\text{def}}{=} \begin{bmatrix} \boldsymbol{c}(t_0, s_0, \xi_0(t_0; q_0)) \\ \vdots \\ \boldsymbol{c}(t_N, s_N, \xi_N(t_N; q_N)) \\ \boldsymbol{r}(t_N, s_N) \end{bmatrix}.$$

The evaluation of $\boldsymbol{G}(\cdot)$ requires the integration of the dynamic system equations and this makes it computationally expensive. However, evaluating the n–th matching condition only involves w_{n-1} nonlinearly and s_n linearly for $n \in [N]$, and is therefore *separable*. We call a function of NLP (6.20) separable if unknowns w_n are decoupled from unknowns w_m, $0 \leqslant n, m \leqslant N$, $n \neq m$ in the sense that for function components depending on w_n the w_m are either absent or appear linearly. As we will see in the following a block structure of the Jacobian of the respective function with respect to w is a consequence. Tailored algorithms can be used to exploit the separability efficiently, cf. Kirches [272].

When we introduced the control discretization for shooting methods in Section 6.2.1 we stressed the importance of base functions with local support. This choice of the control discretization and similarly the state parametrization implies the objective function $\boldsymbol{F}(\cdot)$ as well as the inequality constraint function $\boldsymbol{H}(\cdot)$ to be separable with respect to w_n.

NLP Constraint Jacobian The Jacobians of $\boldsymbol{G}(\cdot)$ and $\boldsymbol{H}(\cdot)$ are

$$\frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w) = \begin{bmatrix} -\boldsymbol{I}_{n_x} & & & & & & \\ X_0^s & X_0^q & -\boldsymbol{I}_{n_x} & & & & \\ & & X_1^s & X_1^q & -\boldsymbol{I}_{n_x} & & \\ & & & \ddots & & & \\ & & & & X_{N-1}^s & X_{N-1}^q & -\boldsymbol{I}_{n_x} \end{bmatrix} \tag{6.21}$$

and

$$\frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{H}(w) = \begin{bmatrix} Y_0^s & Y_0^q & & & & \\ & & Y_1^s & Y_1^q & & \\ & & & \ddots & & \\ & & & & Y_N^s & Y_N^q \\ & & & & Z_N^s & \end{bmatrix}$$

with

$$\begin{aligned} X_n^s &= \frac{\partial}{\partial s_n}\boldsymbol{x}_n(t_{n+1}; s_n, q_n), \qquad X_n^q = \frac{\partial}{\partial q_n}\boldsymbol{x}_n(t_{n+1}; s_n, q_n), \qquad 0 \leqslant n \leqslant N-1, \\ Y_n^s &= \frac{\partial}{\partial s_n}\boldsymbol{c}(t_{n+1}, s_n, q_n), \qquad Y_n^q = \frac{\partial}{\partial q_n}\boldsymbol{c}(t_{n+1}, s_n, q_n), \qquad 0 \leqslant n \leqslant N, \\ Z_N^s &= \frac{\partial}{\partial s_N}\boldsymbol{r}(t_N, s_n). \end{aligned}$$

NLP Hessian of the Lagrangian In order to define the Lagrangian function of NLP (6.19) we introduce multipliers $\lambda \stackrel{\text{def}}{=} [(\lambda^{\mathrm{s}})^T, (\lambda^{\mathrm{m}})^T]^T$ for the equality constraints $\boldsymbol{G}(\cdot)$ with λ^{s} and $\lambda^{\mathrm{m}} \stackrel{\text{def}}{=} [(\lambda_0^{\mathrm{m}})^T, \ldots, (\lambda_{N-1}^{\mathrm{m}})^T]^T$ being the multipliers for the initial conditions and the matching conditions, respectively. Multipliers for the inequality constraints $\boldsymbol{H}(\cdot)$ are denoted by $\mu \stackrel{\text{def}}{=} [(\mu^{\mathrm{c}})^T, (\mu^{\mathrm{r}})^T]^T$ with $\mu^{\mathrm{c}} \stackrel{\text{def}}{=} [(\mu_0^{\mathrm{c}})^T, \ldots, (\mu_N^{\mathrm{c}})^T]^T$ and μ^{r} being the multipliers for the mixed control–state constraints $\boldsymbol{c}(\cdot)$ and the terminal point constraints $\boldsymbol{r}(\cdot)$, respectively. The Lagrangian function of NLP (6.19) then reads as:

$$\begin{aligned} \mathcal{L}(w, \lambda, \mu) &\stackrel{\text{def}}{=} \sum_{n=0}^{N} \boldsymbol{\psi}_n(w_n) - (\lambda^{\mathrm{s}})^T (x_{\mathrm{s}} - s_0) - \sum_{n=0}^{N-1} (\lambda_n^{\mathrm{m}})^T (\boldsymbol{x}_n(t_{n+1}; w_n) - s_{n+1}) \\ &\quad - \sum_{n=0}^{N} (\mu_n^{\mathrm{c}})^T \boldsymbol{c}[w_n] - (\mu^{\mathrm{r}})^T \boldsymbol{r}[w_N] \\ &= \sum_{n=0}^{N} \mathcal{L}_n(w_n, \lambda, \mu) \end{aligned} \tag{6.22}$$

with

$$\begin{aligned}
\mathcal{L}_0(w_0,\lambda,\mu) &= \boldsymbol{\psi}_0(w_0) - (\lambda^s)^T(x_s - s_0) - (\lambda_0^m)^T \boldsymbol{x}_0(t_1;w_0) - (\mu_0^c)^T \boldsymbol{c}[w_0],\\
\mathcal{L}_n(w_n,\lambda,\mu) &= \boldsymbol{\psi}_n(w_n) - (\lambda_n^m)^T \boldsymbol{x}_n(t_{n+1};w_n) + (\lambda_{n-1}^m)^T s_n - (\mu_n^c)^T \boldsymbol{c}[w_n],\ n \in [N-1],\\
\mathcal{L}_N(w_N,\lambda,\mu) &= \boldsymbol{\psi}_N(w_N) + (\lambda_{N-1}^m)^T s_N - (\mu_N^c)^T \boldsymbol{c}[w_N] - (\mu^r)^T \boldsymbol{r}[w_N].
\end{aligned}$$

With notation (6.22) it is obvious that the Lagrangian function is separable with respect to w_n, i.e., $\frac{\partial^2}{\partial w_n \partial w_m}\mathcal{L}(w,\lambda,\mu) = 0$, $n \neq m$. Similar to the Jacobians of equality constraint function $\boldsymbol{G}(\cdot)$ and inequality constraint function $\boldsymbol{H}(\cdot)$ this separability yields the following block–diagonal structure of the Hessian matrix with respect to w of the Lagrangian function:

$$\nabla_w^2 \mathcal{L}(w,\lambda,\mu) = \begin{bmatrix} \nabla_{w_0}^2 \mathcal{L}_0(w_0,\lambda,\mu) & & \\ & \ddots & \\ & & \nabla_{w_N}^2 \mathcal{L}_N(w_N,\lambda,\mu) \end{bmatrix}$$

with

$$\nabla_{w_n}^2 \mathcal{L}_n(w_n,\lambda,\mu) = \begin{bmatrix} \frac{\partial^2}{\partial s_n^2}\mathcal{L}_n(w_n,\lambda,\mu) & \frac{\partial^2}{\partial q_n \partial s_n}\mathcal{L}_n(w_n,\lambda,\mu) \\ \frac{\partial^2}{\partial s_n \partial q_n}\mathcal{L}_n(w_n,\lambda,\mu) & \frac{\partial^2}{\partial q_n^2}\mathcal{L}_n(w_n,\lambda,\mu) \end{bmatrix}, \qquad 0 \leqslant n \leqslant N.$$

An application of the BFGS update formula (3.27) at $w^k = [{w_0^k}^T, \ldots, {w_N^k}^T]^T$ might result in an approximation of the Hessian of the Lagrangian

$$B^k \simeq \operatorname{diag}\left(\nabla_{w_n}^2 \mathcal{L}_n(w_n^k, \lambda^k, \mu^k)\right)$$

that has a destroyed sparsity pattern which was imposed by the separability of the Lagrangian function. One possible way to overcome this problem was proposed by Bock and Plitt [75] and Plitt [356] and is called *high rank BFGS updating*. Here the initial approximation of the Hessian is chosen to be

$$B^0 = \begin{bmatrix} B_0^0 & & \\ & \ddots & \\ & & B_N^0 \end{bmatrix},$$

where the B_n^0, $0 \leqslant n \leqslant N$ are initial approximations of $\nabla_{w_n}^2 \mathcal{L}_n(w_n^0, \lambda^0, \mu^0)$. The step part and the Lagrangian gradient difference part are chosen to be

$$\theta_n^k = \alpha \Delta w_n^k \qquad \text{and} \qquad \eta_n^k = \nabla_{w_n} \mathcal{L}_n(w_n^{k+1}, \lambda^{k+1}, \mu^{k+1}) - \nabla_{w_n} \mathcal{L}_n(w_n^k, \lambda^{k+1}, \mu^{k+1}).$$

The BFGS update formula (3.27) is then applied to each of the submatrices B_n^0, $0 \leqslant n \leqslant N$, separately. High rank BFGS updates can be interpreted as a rank $2N+2$ approximation of the matrix B^k, cf. Bock and Plitt [75], Plitt [356], Leineweber [290], and Leineweber [291].

QP Subproblem of the SQP Method The QP of the k–th SQP iteration (see Algorithm 2 in Section 3.6.2) has the form

$$\begin{aligned} \min_{\Delta w} \quad & \frac{1}{2}\Delta w^T B^k \Delta w + \nabla \boldsymbol{F}(w^k)^T \Delta w \\ \text{s.t.} \quad & \mathbf{0} = \frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w^k)\,\Delta w + \boldsymbol{G}(w^k) + [\boldsymbol{I}_{n_x}\mathbf{0}\ldots\mathbf{0}]^T x_s, \\ & \mathbf{0} \geqslant \frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{H}(w^k)\,\Delta w + \boldsymbol{H}(w^k). \end{aligned}$$

It plays a central role in the sections about real–time optimization for Nonlinear Model Predictive Control, cf. Appendix C and Appendix D.

6.3 Full Discretization Approach

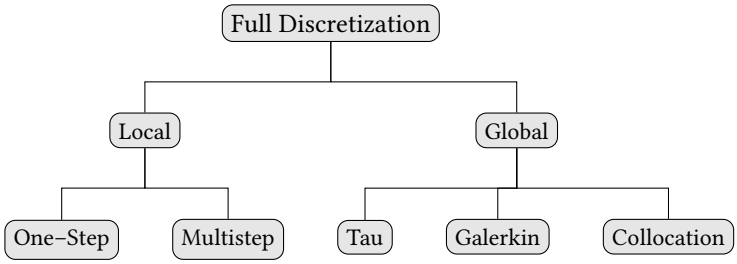


Figure 6.5: Direct Methods with state and control discretization Overview

The OCP that we adduce to discuss the full discretization approach is the continuous OCP in standard form which was introduced in Definition 5.1. We assume the OCP to be given with a Mayer term objective functional for the local approach.

6.3.1 Local Approach

For a local approach to solve an OCP the horizon interval $\mathcal{T} = [t_s, t_f]$ is divided into N intervals

$$\mathbb{G}_N \overset{\text{def}}{=} \{t_s = t_0 < t_1 < \ldots < t_N = t_f\},$$

where N is a natural number. We use the term *Finite Element (FE)* for each of the intervals $[t_n, t_{n+1}]$. The FE boundaries are referred to as *nodes*, *mesh* or *grid points*.
To transcribe the OCP into a NLP we use w as a set of NLP variables which arise from discretizations of states and controls. The optimal control constraints are replaced by the NLP constraints

$$c_l \leqslant \boldsymbol{c}(w) \leqslant c_u,$$

where

$$\boldsymbol{c}(w) = [d_1, d_2, \ldots, d_N, c_0, c_1, \ldots, c_N, r]^T$$

and

$$c_l = [\mathbf{0}, \ldots, \mathbf{0}, -\boldsymbol{\infty}, \ldots, -\boldsymbol{\infty}, \mathbf{0}]^T, \qquad c_u = [\mathbf{0}, \ldots, \mathbf{0}, \mathbf{0}, \ldots, \mathbf{0}, \mathbf{0}]^T.$$

The first $N \cdot n_{\mathrm{x}}$ constraints require the defect vectors arising from an ODE discretization to be equal to zero. The nonlinear path constraints are imposed at the grid points and results therefore in $(N+1) \cdot n_{\mathrm{c}}$ additional constraints. The boundary conditions are enforced directly by the constraints $\boldsymbol{r}$. Simple bounds on states $\boldsymbol{x}(\cdot)$ or controls $\boldsymbol{u}(\cdot)$ can be directly translated to simple bounds on the corresponding NLP variables.

Control Discretization

The control discretization for the full discretization approach can be done in the same way as we have done for the reduced discretization approach in Section 6.2.1. This means in particular that different control discretizations are possible for each control component. For the sake of simplicity this is not done in the following considerations. We rather assume $\{\mathcal{B}_1(\cdot), \ldots, \mathcal{B}_M(\cdot)\}$ to span an M–dimensional subspace $\mathcal{U}_M$ of the control space, where M usually depends on the number N of intervals in $\mathbb{G}_N$. Then every $\xi_M \in \mathcal{U}_M^{n_{\mathrm{u}}}$ can be expressed by means of coefficients $q \stackrel{\text{def}}{=} [{q_1}^T, \ldots, {q_M}^T]^T \in \mathbb{R}^{M \cdot n_{\mathrm{u}}}$ as

$$\xi_M(t) \stackrel{\text{def}}{=} \sum_{m=1}^{M} q_m \, \mathcal{B}_m(t).$$

As we have done before we indicate the dependence on the vector q by using the notation

$$\xi_M(t) \stackrel{\text{def}}{=} \xi_M(t; q) \stackrel{\text{def}}{=} \xi_M(t; q_1, \ldots, q_M).$$

Sometimes we may even identify $\xi_M(\cdot)$ and q.

State Discretization

Local methods mainly differ in the way how they discretize the ODE equations. Depending on the discretization approach (see Section 6.1.2) they are categorized into *one–step methods* and *multistep methods*. We survey both approaches briefly but refer the reader to the literature for a deeper insight.

For the state discretization we use the grid function notation again which we introduced for discretization schemes in Section 6.2.1. For this reason we try to determine a grid function $\boldsymbol{x}_N : \mathbb{G}_N \longrightarrow \mathbb{R}^{n_{\mathrm{x}}}$ such that $t_n \mapsto \boldsymbol{x}_N(t_n)$ and $\eta_n \stackrel{\text{def}}{=} \boldsymbol{x}_N(t_n)$. Consequently, the NLP variables from a local approach consist of the control parametrization coefficients q and state discretization function values $\{\eta_n\}_{n=0}^N$. We summarize them in the NLP variable w.

Local Approach using One–Step Methods For this approach we apply the general one–step discretization scheme for ODE IVPs from Definition 6.1 to discretize the ODE from the OCP under consideration such that for a given control approximation $\xi_M(\cdot)$ the η_n are determined according to the equation

$$\eta_{n+1} = \eta_n + h_n \Phi(t_n, h_n, \eta_n, \xi_M(t_n; q)), \qquad h_n \stackrel{\text{def}}{=} t_{n+1} - t_n, \qquad n = 0, \ldots, N-1.$$

Notice that the generating function $\Phi(\cdot)$ depends on the control coefficients q. The fully discretized OCP (5.1) with Mayer term objective functional is given as

$$\begin{aligned}
\min_{w} \quad & \varphi(t_0, \eta_0, t_N, \eta_N) && && \text{(6.23a)}\\
\text{s.t.} \quad & \mathbf{0}_{n_x} = \eta_n + h_n \Phi(t_n, h_n, \eta_n, q) - \eta_{n+1}, && n = 0, \ldots, N-1, && \text{(6.23b)}\\
& \mathbf{0}_{n_c} \geqslant \boldsymbol{c}(t_n, \eta_n, q), && n = 0, \ldots, N-1, && \text{(6.23c)}\\
& \mathbf{0}_{n_r} = \boldsymbol{r}(t_0, \eta_0, t_N, \eta_N). && && \text{(6.23d)}
\end{aligned}$$

A notation similarly to NLP (6.20) for the Direct Multiple Shooting approach can be found easily. This includes respective definitions for the functions $\boldsymbol{F}(\cdot)$, $\boldsymbol{G}(\cdot)$ and $\boldsymbol{H}(\cdot)$. For details we refer the reader to Gerdts [189, Section 5.1.2].
NLP (6.23) is large–scale, but using tailored numerical optimization algorithms which allow for exploiting the sparse structure of gradient $\boldsymbol{F}'(w)$ and Jacobians $\boldsymbol{G}'(w)$, $\boldsymbol{H}'(w)$ it can be solved very efficiently, cf. Betts and Huffman [63, 66] and Betts [62].

Local Approach using Multistep Methods We illustrate this approach by reference to the BDF method (see Definition 6.4) with self–starting procedure. Given some control approximation function $\xi_M(\cdot)$ the BDF method finds state approximations η_n by solving the system of equations

$$\sum_{i=0}^{l_n} \alpha_i^{(n)} \eta_{n+1-i} = h_n \boldsymbol{f}(t_{n+1}, \eta_{n+1}, \xi_M(t_{n+1}; q)), \quad h_n \stackrel{\text{def}}{=} t_{n+1} - t_n, \quad n = 0, \ldots, N-1,$$

where the definition of coefficients $\alpha_i^{(n)}$ follows the one from (6.4) in Definition 6.4. The integration step orders l_n are set according to its self–starting character, i.e., the procedure starts with $l_0 = 1$, which makes the first equation an implicit Euler step, and then gradually increases the order until the maximum order is reached.
Putting everything together the fully discretized OCP (5.1) with Mayer term objective functional and an BDF based ODE discretization is given as

$$\begin{aligned}
\min_{w} \quad & \varphi(t_0, \eta_0, t_N, \eta_N) && && \text{(6.24a)}\\
\text{s.t.} \quad & \mathbf{0}_{n_x} = \sum_{i=0}^{l_n} \alpha_i^{(n)} \eta_{n+1-i} - h_n \boldsymbol{f}(t_{n+1}, \eta_{n+1}, q), && n = 0, \ldots, N-1, && \text{(6.24b)}
\end{aligned}$$

$$\mathbf{0}_{n_c} \geqslant \boldsymbol{c}(t_n, \eta_n, q), \qquad n = 0, \ldots, N-1, \tag{6.24c}$$

$$\mathbf{0}_{n_r} = \boldsymbol{r}(t_0, \eta_0, t_N, \eta_N). \tag{6.24d}$$

6.3.2 Global Approach

Global methods are also known as *spectral methods* or *Discrete Variable Representation (DVR)* methods and are a class of numerical methods that are extensively used for solving Partial Differential Equations (PDEs) numerically. Analytic studies of differential equations by means of spectral representations have been used since the days of Fourier [170]. Early applications of spectral methods to the numerical solutions of ODEs go back at least to Lanczos [283].

Spectral methods approximate the differential state $\boldsymbol{x}(\cdot)$ by a polynomial of order $N = N_p - 1$

$$\boldsymbol{x}(t) \simeq \boldsymbol{X}_N(t) \stackrel{\text{def}}{=} \sum_{n=1}^{N_p} a_n \boldsymbol{\psi}_n(t), \tag{6.25}$$

where the $\boldsymbol{\psi}_n(\cdot)$ are smooth basis functions and the a_n act as parameters. The $\boldsymbol{\psi}_n(\cdot)$ are usually called *trial functions* or *expansion functions*. This is due to the fact that they function as truncated series expansion of the solution. For this reason the a_n are also called *expansion coefficients*. The control $\boldsymbol{u}(\cdot)$ is approximated in a similar way as $\boldsymbol{x}(\cdot)$.

Numerical Analysis In order to do some numerical analysis theory in this section we introduce weighted Sobolev spaces of Hilbert type over the open interval $\mathcal{I} \stackrel{\text{def}}{=} (-1, +1)$. Sobolev spaces have been presented in Section 2.4.3. Given a weight function $\boldsymbol{\omega}(\cdot)$ on $\mathcal{I}$ that fulfills $\boldsymbol{\omega} \in L^1(\mathcal{I}, \mathbb{R})$ and $\boldsymbol{\omega}(t) > 0$ for $t \in \mathcal{I}$ we define the function space

$$L^2_{\boldsymbol{\omega}}(\mathcal{I}, \mathbb{R}) \stackrel{\text{def}}{=} \{\boldsymbol{\phi} : \mathcal{I} \longrightarrow \mathbb{R} : \boldsymbol{\phi}\,(\cdot) \text{ is measurable and } \langle \boldsymbol{\phi}, \boldsymbol{\phi} \rangle_{\boldsymbol{\omega}} < +\infty\},$$

where the scalar product is given as

$$\langle \boldsymbol{\phi}, \boldsymbol{\psi} \rangle_{\boldsymbol{\omega}} \stackrel{\text{def}}{=} \int_{\mathcal{I}} \boldsymbol{\omega}(t)\ \boldsymbol{\phi}\,(t) \boldsymbol{\psi}(t)\, \mathrm{d}t.$$

For any $q \in \mathbb{N}$ we define the function space

$$H^q_{\boldsymbol{\omega}}(\mathcal{I}, \mathbb{R}) \stackrel{\text{def}}{=} \{\boldsymbol{\phi} \in L^2_{\boldsymbol{\omega}}(\mathcal{I}, \mathbb{R}) : \|\boldsymbol{\phi}\|_{H^q_{\boldsymbol{\omega}}} < +\infty\}$$

with

$$\|\boldsymbol{\phi}\|^2_{H^q_{\boldsymbol{\omega}}} = \sum_{i=0}^{q} \int_{\mathcal{I}} \boldsymbol{\omega}(t) \left|\boldsymbol{\phi}^{(i)}\,(t)\right|^2 \mathrm{d}t. \tag{6.26}$$

Finally, we need the semi–norms (see e.g. Canuto et al. [100])

$$|\boldsymbol{\phi}|^2_{H^q_{\boldsymbol{\omega}}} = \int_{\mathcal{I}} \boldsymbol{\omega}(t) \left|\boldsymbol{\phi}^{(q)}(t)\right|^2 \mathrm{d}t, \quad |\boldsymbol{\phi}|^2_{H^{q;N}_{\boldsymbol{\omega}}} = \sum_{i=\min(q,N+1)}^{q} \int_{\mathcal{I}} \boldsymbol{\omega}(t) \left|\boldsymbol{\phi}^{(i)}(t)\right|^2 \mathrm{d}t. \tag{6.27}$$

Note that for the choice $\boldsymbol{\omega} \equiv 1$ we systematically drop the subscript $\boldsymbol{\omega}$. For this case one can easily verify that $L^2_{\boldsymbol{\omega}}(\mathcal{I},\mathbb{R})$ is identical with $L^2(\mathcal{I},\mathbb{R})$. Moreover, the spaces $H^q_{\boldsymbol{\omega}}(\mathcal{I},\mathbb{R})$ and $W^{q,2}(\mathcal{I},\mathbb{R})$ coincide and whenever $N \geqslant q-1$, one has

$$|\boldsymbol{\phi}|_{H^q_{\boldsymbol{\omega}}} = \left\|\boldsymbol{\phi}^{(q)}\right\|_2 = |\boldsymbol{\phi}|_{H^{q;N}_{\boldsymbol{\omega}}}. \tag{6.28}$$

For a domain $\mathcal{I}$ of *finite width* (as we assumed) it can be shown that the above norm (6.26) and semi–norms (6.27) are equivalent:

Lemma 6.5
For $\mathcal{I}$ of finite length $|\cdot|_{H^{q;N}_{\boldsymbol{\omega}}}$ is a norm on $H^q_{\boldsymbol{\omega}}(\mathcal{I},\mathbb{R})$ which is equivalent to the standard norm $\|\cdot\|_{H^q_{\boldsymbol{\omega}}}$, i.e., there exists a pair of real numbers $0 < C_1 \leqslant C_2$ such that

$$C_1 |\boldsymbol{\phi}|_{H^{q;N}_{\boldsymbol{\omega}}} \leqslant \|\boldsymbol{\phi}\|_{H^q_{\boldsymbol{\omega}}} \leqslant C_2 |\boldsymbol{\phi}|_{H^{q;N}_{\boldsymbol{\omega}}} \quad \forall\, \boldsymbol{\phi} \in H^q_{\boldsymbol{\omega}}(\mathcal{I},\mathbb{R}). \qquad \triangle$$

Proof See e.g. Adams and Fournier [6, Corollary 6.31]. □

For now let us consider the system $\{\boldsymbol{\psi}_n\}_{n=0}^{\infty}$ – with $\deg(\boldsymbol{\psi}_n) = n$ – of the orthonormal polynomials in $L^2_{\boldsymbol{\omega}}(\mathcal{I},\mathbb{R})$. Choosing $\boldsymbol{\omega}(t) = (1-t^2)^{-1/2}$ the associated orthonormal system is given by means of Chebyshev polynomials of first kind $\boldsymbol{T}_n$ as $\{\gamma_n \boldsymbol{T}_n\}_{n=0}^{\infty}$ (with $\gamma_0 = \pi^{-1/2}$ and $\gamma_n = (2/\pi)^{1/2}$ for $n \geqslant 1$), cf. Canuto and Quarteroni [98]. For the choice $\boldsymbol{\omega}(t) = 1$ we will identify the according orthonormal system to be scaled Legendre polynomials that we denote by $\{\widetilde{\boldsymbol{P}}_n\}_{n=0}^{\infty}$, cf. (6.36)+(6.36). Since algorithms presented in this contribution are based on the choice $\boldsymbol{\omega}(t) = 1$ we concentrate on this case in the remainder of this section for the most part. Nevertheless the reader should keep in mind that convergence properties as presented subsequently can be found for Chebyshev systems as well, cf. Canuto and Quarteroni [98]. Moreover, the theory introduced in this work might be investigated in a Chebyshev context in future contributions.
It is well–known that the aforementioned orthonormal systems are complete in $L^2_{\boldsymbol{\omega}}(\mathcal{I},\mathbb{R})$, cf. Szegö [421]; any $\boldsymbol{x} \in L^2_{\boldsymbol{\omega}}(\mathcal{I},\mathbb{R})$ can be written as

$$\boldsymbol{x}(t) = \sum_{n=0}^{\infty} \widetilde{x}_n \boldsymbol{\psi}_n(t), \quad \widetilde{x}_n = \langle \boldsymbol{x}, \boldsymbol{\psi}_n \rangle_{\boldsymbol{\omega}} = \int_{-1}^{+1} \boldsymbol{\omega}(t)\, \boldsymbol{x}(t)\, \boldsymbol{\psi}_n(t)\, \mathrm{d}t \tag{6.29}$$

with

$$\|\boldsymbol{x}\|^2_{0,\boldsymbol{\omega}} = \sum_{n=0}^{\infty} |\widetilde{x}_n|^2,$$

i.e., the $\widetilde{x}_n$ are the orthogonal projections of $\boldsymbol{x}(\cdot)$ onto the $\boldsymbol{\psi}_n(\cdot)$ with respect to the scalar product $\langle \cdot,\cdot \rangle_{\boldsymbol{\omega}}$. Depending on the concrete choice of $\boldsymbol{\omega}(\cdot)$ the series (6.29) have different names

such as Chebyshev series for $\boldsymbol{\omega}(t) = (1-t^2)^{-1/2}$ and Legendre series for $\boldsymbol{\omega}(t) = 1$. By means of the set of all polynomials of degree less than or equal to N, namely

$$\mathcal{S}_N \stackrel{\text{def}}{=} \mathcal{S}_N(\mathcal{I}) \stackrel{\text{def}}{=} \{\boldsymbol{\psi}_n \,:\, 0 \leqslant n \leqslant N\},$$

we denote by $P_N : L^2_{\boldsymbol{\omega}}(\mathcal{I},\mathbb{R}) \longrightarrow \mathcal{S}_N$ the orthogonal projection on $\mathcal{S}_N$ in $L^2_{\boldsymbol{\omega}}(\mathcal{I},\mathbb{R})$ such that

$$P_N(\boldsymbol{x}) = \sum_{n=0}^{N} \langle \boldsymbol{x}, \boldsymbol{\psi}_n \rangle_{\boldsymbol{\omega}} \; \boldsymbol{\psi}_n(t).$$

In accordance with respective designators for (6.29) we call the series truncated Chebyshev series $\boldsymbol{\omega}(t) = (1-t^2)^{-1/2}$ and truncated Legendre series for $\boldsymbol{\omega}(t) = 1$.

Rates of Convergence For the upcoming polynomial approximation theory a precise classification of the rate of convergence seems to be helpful, cf. Boyd [81, 82]. Note that all definitions must be understood asymptotically, i.e., they are based on series coefficient behavior for large N. The rate of convergence of a series of type (6.29) is usually defined in terms of a so–called *algebraic index of convergence*:

Definition 6.6 (Algebraic Index of Convergence)
A series of type (6.29) with coefficients $\{a_n\}$ has an *algebraic index of convergence* k if k is the largest number such that

$$\lim_{n\to\infty} n^k \, |a_n| < \infty.$$

Alternatively, k is the algebraic index if $a_n = \mathcal{O}\left(1/n^k\right)$. △

Definition 6.7 (Exponential/Spectral Convergence)
A series of type (6.29) with coefficients $\{a_n\}$ converges *exponentially* or *spectrally* if the algebraic index k is unbounded, i.e., a_n decreases faster than $1/n^k$ for any finite k. △

Now we come back to the spectral method approach and formulate the following crucial questions that arise if one wants to determine state and control approximations $\boldsymbol{X}_N(\cdot)$ and $\boldsymbol{U}_N(\cdot)$ of form (6.25):

(i) How should the expansion coefficients a_n be determined?

(ii) From which function class should the $\boldsymbol{\psi}_n(\cdot)$ be chosen?

The Weighted Residual Method

We start by answering the first question. A necessary condition for $\boldsymbol{x}(\cdot)$ and $\boldsymbol{u}(\cdot)$ to be admissible for the OCP requires the residual $\dot{\boldsymbol{x}}(\cdot) - \boldsymbol{f}(\cdot, \boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot))$ to be equal to zero on the horizon interval $\mathcal{T}$ (w.l.o.g. $\mathcal{T}$ can be assumed to be the interval $[-1,+1]$). This usually does not hold for the residual

$$\boldsymbol{\varrho}_N(t) = \dot{\boldsymbol{X}}_N(t) - \boldsymbol{f}(t, \boldsymbol{X}_N(t), \boldsymbol{U}_N(t))$$

with (finite dimensional) approximations $\boldsymbol{X}_N(t) \simeq \boldsymbol{x}(t)$ and $\boldsymbol{U}_N(t) \simeq \boldsymbol{u}(t)$. For this reason it is the goal to make the residual $\boldsymbol{\varrho}_N(\cdot)$ small. To do so we define a space of so called *test functions*, $\{\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_{N_p}\}$, and require that the residual is orthogonal to all test functions in this space, i.e.,

$$\langle \boldsymbol{\varrho}_N, \boldsymbol{\phi}_n \rangle = \int_{-1}^{+1} \boldsymbol{\varrho}_N(t)\, \boldsymbol{\phi}_n(t)\, \mathrm{d}t = 0, \quad n = 1, \ldots, N_p.$$

If N is increased then the obtained solution is close to the real one. According to the choice of the test function space spectral methods can be classified. Here we present three of them, namely Tau method, Galerkin method and collocation method.

Tau Method The *tau method* (see Lanczos [283]) chooses the test function space to be equivalent to the trial function space, i.e., one chooses $\boldsymbol{\phi}_n = \boldsymbol{\psi}_n$. The orthogonality conditions for the residual become

$$\langle \boldsymbol{\varrho}_N, \boldsymbol{\psi}_n \rangle = 0, \quad n \in [N_p].$$

Boundary conditions are enforced by additional constraints.

Galerkin Method The Galerkin *method* combines the original basis functions into a new set, $\widetilde{\boldsymbol{\psi}}_n$, $n \in [N_p]$, in which all the functions satisfy the boundary conditions. The expansion coefficients are then determined as

$$\left\langle \boldsymbol{\varrho}_N, \widetilde{\boldsymbol{\psi}}_n \right\rangle = 0, \quad n \in [N_p],$$

meaning that the residual is orthogonal to the new basis functions.

Collocation Method The *collocation method* is often called *pseudospectral method*. In this method, we imagine the test functions to be zero everywhere but at one *collocation point*, i.e., we have $\boldsymbol{\phi}_n = \boldsymbol{\delta}(t - t_n)$, where $\boldsymbol{\delta}(\cdot)$ denotes the $\boldsymbol{\delta}$–function that has been investigated in Section 2.4.6 on page 77. Using the collocation approach yields the equations

$$\langle \boldsymbol{\varrho}_N, \boldsymbol{\phi}_n \rangle = \dot{\boldsymbol{X}}_N(t_n) - \boldsymbol{f}(t_n, \boldsymbol{X}_N(t_n), \boldsymbol{U}_N(t_n)) = 0, \quad n \in [N_p]. \tag{6.30}$$

If the collocation points are chosen to be roots of orthogonal polynomials or combinations thereof the resulting method is also called *orthogonal collocation method*.

Note that global methods based on the collocation approach are often applied to solve OCPs. In (6.29) we have indicated the common strategy to approximate functions with orthogonal projections. Due to their construction collocation methods suggest a second way of approximation: in fact equations (6.30) require the ODE residual to vanish at collocation points t_n which gives $\boldsymbol{X}_N(\cdot)$ and $\boldsymbol{U}_N(\cdot)$ an *interpolatory* character. For this reason our later convergence analysis not only comprises the projection operator P_N but also an interpolation operator I_N which will be introduced soon.

Function classes

Now we address the second question that arises when it comes to find proper approximations $\boldsymbol{X}_N(\cdot) \simeq \boldsymbol{x}(\cdot)$ and $\boldsymbol{U}_N(\cdot) \simeq \boldsymbol{u}(\cdot)$, namely how the basis functions $\boldsymbol{\psi}_n(\cdot)$ in (6.25) should be chosen. We establish some criteria that have to be fulfilled by a proper choice for the $\boldsymbol{\psi}_n(\cdot)$. We do this in the form of a wish list:

(i) There should exist a fast and numerically stable way to convert between expansion coefficients a_n and the approximate function values $x_n \stackrel{\text{def}}{=} \boldsymbol{X}_N(t_n)$ at distinct nodes t_n, $n \in [N_p]$.

(ii) For given coefficients $\{a_n\}_{n=1}^{N_p}$ it should be easy to calculate coefficients $\{b_n\}_{n=1}^{N_p}$ such that

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\sum_{n=1}^{N_p} a_n \boldsymbol{\psi}_n(t)\right) = \sum_{n=1}^{N_p} b_n \boldsymbol{\psi}_n(t). \tag{6.31}$$

(iii) For sufficiently smooth functions $\boldsymbol{x}(\cdot)$ their approximation $\boldsymbol{X}_N(\cdot) = \sum_{n=1}^{N_p} a_n \boldsymbol{\psi}_n(\cdot)$ must converge *fast* towards $\boldsymbol{x}(\cdot)$.

Expansion Coefficient Transformation As an obvious choice for functions $\boldsymbol{\psi}_n(\cdot)$ in (6.25) one could consider the *monomial basis* ($\boldsymbol{\psi}_n(t) = t^{n-1}$) such that $\boldsymbol{X}_N(\cdot)$ becomes a truncated Taylor expansion

$$\boldsymbol{X}_N(t) = \sum_{n=1}^{N_p} \widetilde{x}_n t^{n-1}.$$

For this choice one could determine coefficients $\widetilde{x}_n$ by an $\mathcal{L}^2$–projection such that

$$\langle \boldsymbol{x}, \boldsymbol{\psi}_m \rangle = \sum_{n=1}^{N_p} \widetilde{x}_n \langle \boldsymbol{\psi}_n, \boldsymbol{\psi}_m \rangle \tag{6.32}$$

must hold for all N_p basis functions $\boldsymbol{\psi}_m(\cdot)$. Note that we do not use a_n to denote expansion coefficients but $\widetilde{x}_n$ which is done to express that the coefficients are related to projections. With the definitions

$$M_{i,j} = \left\langle \boldsymbol{\psi}_i, \boldsymbol{\psi}_j \right\rangle, \quad \overline{x}_i = \langle \boldsymbol{x}, \boldsymbol{\psi}_i \rangle, \quad \widetilde{x} = [\widetilde{x}_1, \ldots, \widetilde{x}_N]^T, \quad \overline{x} = [\overline{x}_1, \ldots, \overline{x}_N]^T \tag{6.33}$$

we can rewrite (6.32) as the linear equation

$$M\widetilde{x} = \overline{x} \tag{6.34}$$

with N_p equations for the N_p unknown expansion coefficients, $\widetilde{x}_n$. The so called *mass matrix* M, where

$$M_{i,j} = \frac{1}{i+j-1}\left[1+(-1)^{i+j}\right], \tag{6.35}$$

is reminiscent of a Hilbert matrix which is known to be very poorly conditioned. Table 6.1 depicts the condition number, $\kappa(M)$, of mass matrix M for increasing order of approximation, N, and one can observe a rapid growth of condition number values that is caused by an increasing close to linear dependence of the basis for increasing (i,j) due to the coefficient $(i+j-1)^{-1}$ in (6.35). Hence, even at moderately high order it is hard to recover $\widetilde{x}$ accurately. This implies that $\boldsymbol{X}_N(\cdot)$ is not a good polynomial representation of $\boldsymbol{x}(\cdot)$.

N	2	4	8	16
$\kappa(M)$	1.4×10^1	3.6×10^2	3.1×10^5	3.0×10^{11}

Table 6.1: Condition number of mass matrix M based on the monomial basis for different values N.

To overcome the problem of an ill–conditioned mass matrix one could consider an orthonormal basis. We obtain such an orthonormal basis if we apply an $\mathcal{L}^2$–based Gram–Schmidt orthogonalization approach to the monomial basis, t^n. The resulting basis is given as

$$\boldsymbol{\psi}_n(t) = \widetilde{\boldsymbol{P}}_{n-1}(t) = \frac{\boldsymbol{P}_{n-1}(t)}{\sqrt{\gamma_{n-1}}}, \tag{6.36}$$

where $\boldsymbol{P}_n(\cdot)$ is the Legendre polynomial of order n (see Appendix B.1) and

$$\gamma_n = \frac{2}{2n+1} \tag{6.37}$$

is the normalization. The new basis can be obtained by the three term recursion

$$t\widetilde{\boldsymbol{P}}_n(t) = a_n\widetilde{\boldsymbol{P}}_{n-1}(t) + a_{n+1}\widetilde{\boldsymbol{P}}_{n+1}(t), \quad \widetilde{\boldsymbol{P}}_0(t) = \frac{1}{\sqrt{2}}, \quad \widetilde{\boldsymbol{P}}_1(t) = \sqrt{\frac{3}{2}}t,$$

where

$$a_n = \sqrt{\frac{n^2}{(2n+1)(2n-1)}}.$$

With the chosen basis the conditioning problem is solved since the mass matrix M is the identity matrix. Then (6.34) becomes $\widetilde{x} = \overline{x}$ and we need to recover $\widetilde{x}_n$ which is given as

$$\widetilde{x}_n = \langle \boldsymbol{x}, \boldsymbol{\psi}_n \rangle = \left\langle \boldsymbol{x}, \widetilde{\boldsymbol{P}}_{n-1} \right\rangle. \tag{6.38}$$

In general, there exists no closed formula to evaluate the scalar product such that this has to be done numerically. Appendix B.3 shows some approaches for numerical integration. Applying the Legendre–Gauss quadrature from Appendix B.3.2 to (6.38) leads to an approximation

$$\widetilde{x}_n \simeq \hat{x}_n = \sum_{i=1}^{N_p} \omega_i \, \boldsymbol{x}(\tau_i) \widetilde{\boldsymbol{P}}_{n-1}(\tau_i), \tag{6.39}$$

where the τ_i and ω_i denote the quadrature points and weights, respectively. Gauss quadrature of this type has the important property that polynomials of degree up to $2N_p - 1$ are integrated exactly.
Due to their nature the expansion coefficients $\widetilde{x}_n$ are also known as *continuous* expansion coefficients in the literature. In comparison the $\hat{x}_n$ are often called *discrete* expansion coefficients. Replacing the continuous expansion coefficients $\widetilde{x}_n$ with Gauss quadrature based expansion coefficients allows us to deduce a crucial relation between the projection operator P_N which is constructed with coefficients $\widetilde{x}_n$ and the interpolation operator $I_N : L^2(\mathcal{I}, \mathbb{R}) \longrightarrow \mathcal{S}_N$ which is defined by means of the $\hat{x}_n$ as

$$I_N(\boldsymbol{x})(t) = \sum_{n=1}^{N_p} \hat{x}_n \widetilde{\boldsymbol{P}}_{n-1}(t), \quad \hat{x}_n = \sum_{i=1}^{N_p} \omega_i \, \boldsymbol{x}(\tau_i) \widetilde{\boldsymbol{P}}_{n-1}(\tau_i). \tag{6.40}$$

The fact that we call I_N interpolation operator can be justified by the following result:

Theorem 6.8
Let $\boldsymbol{x} : \mathcal{I} \longrightarrow \mathbb{R}$ be defined for all points in the interval $\mathcal{I}$, and let the discrete expansion coefficients be given by (6.39). Then $I_N(\boldsymbol{x})$ interpolates $\boldsymbol{x}(\cdot)$ at all Gauss quadrature points τ_n, i.e., it holds

$$I_N(\boldsymbol{x})(\tau_n) = \boldsymbol{x}(\tau_n), \quad n \in [N_p]. \qquad \triangle$$

Proof See Hesthaven et al. [234, Theorem 5.3]. □

Our previous considerations regarding collocation methods advise that function approximations which are interpolatory might be more suitable than a projection based approach. Thus, we assume functions $\boldsymbol{X}_n(\cdot)$ to be of type (6.40) from now on.
One major goal of this section is it to establish an easy way to convert coefficients $\hat{x}_n$ into approximations $x_n = \boldsymbol{X}_N(t_n)$ and vice versa computationally cheap. The strategy to do so is that we define $\hat{x}_n$ such that the approximation is interpolatory (see Appendix B.2); that is

$$\boldsymbol{x}(t_i) = \sum_{j=1}^{N_p} \hat{x}_j \, \widetilde{\boldsymbol{P}}_{j-1}(t_i),$$

where the t_i form a set of N distinct grid nodes. The *modes* or *modal values*, $\hat{x}_i$, and the associated *nodal values*, $x_i = \boldsymbol{x}(t_i)$, are connected via the linear equation

$$V\hat{x} = x, \tag{6.41}$$

where

$$V_{i,j} = \widetilde{\boldsymbol{P}}_{j-1}(t_i), \quad \hat{x} = [\hat{x}_1, \ldots, \hat{x}_N]^T, \quad x = [x_1, \ldots, x_N]^T. \tag{6.42}$$

The matrix V is recognized as a generalized Vandermonde matrix and for reasons that become clear later it is important that it is well conditioned.

We have already established that $\widetilde{\boldsymbol{P}}_n(\cdot)$ is a reasonable basis. To define the Vandermonde matrix we need to determine the grid points t_i. There is some freedom to do so and we want to find reasonable criteria.

With the Lagrange interpolating polynomials $\{\boldsymbol{L}_n(\cdot)\}_{n=1}^{N_p}$ we can express $\boldsymbol{X}_N(\cdot)$ in two different ways:

$$\boldsymbol{X}_N(t) = \sum_{n=1}^{N_p} \hat{x}_n \widetilde{\boldsymbol{P}}_{n-1}(t) = \sum_{n=1}^{N_p} x_n \boldsymbol{L}_n(t). \tag{6.43}$$

By defining the Lebesgue constant Λ_N as

$$\Lambda_N \overset{\text{def}}{=} \max_{-1 \leqslant t \leqslant +1} \sum_{n=1}^{N_p} |\boldsymbol{L}_n(t)|$$

we find – apart from (B.5) – a second upper bound for the interpolation error which is given by

$$\|\boldsymbol{x} - \boldsymbol{X}_N\|_\infty \leqslant (1 + \Lambda_N) \left\|\boldsymbol{x} - \boldsymbol{X}_N^*\right\|_\infty, \tag{6.44}$$

where $\|\cdot\|_\infty$ denotes the maximum norm and $\boldsymbol{X}_N^*(\cdot)$ the best polynomial approximation of order N to $\boldsymbol{x}(\cdot)$ on the interval $[-1, +1]$. Hence, Λ_N quantifies how much larger the interpolation error $\|\boldsymbol{x} - \boldsymbol{X}_N\|_\infty$ is compared to the smallest possible error, which is given by $\left\|\boldsymbol{x} - \boldsymbol{X}_N^*\right\|_\infty$, in the worst case. In literature this error formula is known as Lebesgue inequality. We derive (6.44) as follows: from the uniqueness of interpolating polynomials we know that

$$\boldsymbol{X}_N(t) = \sum_{n=1}^{N_p} \boldsymbol{x}(t_n) \boldsymbol{L}_n(t), \qquad \boldsymbol{X}_N^*(t) = \sum_{n=1}^{N_p} \boldsymbol{X}_N^*(t_n) \boldsymbol{L}_n(t),$$

and calculate by subtracting $\boldsymbol{X}_N(t)$ from $\boldsymbol{X}_N^*(t)$

$$\begin{aligned} \left|\boldsymbol{X}_N^*(t) - \boldsymbol{X}_N(t)\right| &= \left| \sum_{n=1}^{N_p} \left(\boldsymbol{X}_N^*(t_n) - \boldsymbol{x}(t_n)\right) \boldsymbol{L}_n(t) \right| \\ &\leqslant \sum_{n=1}^{N_p} |\boldsymbol{L}_n(t)| \cdot \max_{1 \leqslant n \leqslant N_p} \left|\boldsymbol{X}_N^*(t_n) - \boldsymbol{x}(t_n)\right|. \end{aligned}$$

Therefore we conclude

$$\left\|\boldsymbol{X}_N^* - \boldsymbol{X}_N\right\|_\infty \leqslant \Lambda_N \left\|\boldsymbol{x} - \boldsymbol{X}_N^*\right\|_\infty .$$

We retrieve (6.44) as

$$\begin{aligned}\|\boldsymbol{x} - \boldsymbol{X}_N\|_\infty = \left\|\boldsymbol{x} - \boldsymbol{X}_N^* + \boldsymbol{X}_N^* - \boldsymbol{X}_N\right\|_\infty &\leqslant \left\|\boldsymbol{x} - \boldsymbol{X}_N^*\right\|_\infty + \left\|\boldsymbol{X}_N^* - \boldsymbol{X}_N\right\|_\infty \\ &\leqslant (1+\Lambda_N)\left\|\boldsymbol{x} - \boldsymbol{X}_N^*\right\|_\infty .\end{aligned}$$

As a consequence of the Lebesgue inequality we state that the interpolating polynomial $\boldsymbol{X}_N(\cdot)$ tends to $\boldsymbol{x}(\cdot)$ as $(1+\Lambda_N)\left\|\boldsymbol{x} - \boldsymbol{X}_N^*\right\|_\infty$ tends to zero. Hence, convergence can just be expected if – for increasing polynomial order N – the quantity $\left\|\boldsymbol{x} - \boldsymbol{X}_N^*\right\|_\infty$ decreases faster than the Lebesgue constant Λ_N increases. Here, the value of $\left\|\boldsymbol{x} - \boldsymbol{X}_N^*\right\|_\infty$ quantifies how well $\boldsymbol{x}(\cdot)$ can be approximated in the chosen polynomial basis. Relevant convergence properties will be addressed in the associated section. Note however that it does not depend on the grid point selection. On the contrary Λ_N is determined by the grid points exclusively. For this reason we have to identify grid points such that the Lebesgue constant is minimized.

One could consider equidistant points as an obvious choice. But for this set of grid points, namely $\mathcal{E} = \left\{t_n \,:\, t_n = -1 + \frac{2(n-1)}{N},\ n \in [N_p]\right\}$ the associated Lebesgue constant $\Lambda_N(\mathcal{E})$ grows exponentially fast with asymptotic estimate (see Turetskii [429], Schönhage [395])

$$\Lambda_N(\mathcal{E}) \simeq \frac{2^{N+1}}{\mathrm{e}n(\log N + \gamma)}, \quad N \to \infty, \qquad \gamma = \lim_{N\to\infty}\left(\sum_{n=1}^{N}\frac{1}{n} - \log N\right) = 0.577\ldots,$$

which implies that better choices for the grid point selection are necessary. A similar observation can be made for polynomial interpolation at equidistant grids: a poorly behaved interpolation results from an exponentially fast growth of $\boldsymbol{L}_n(\cdot)$ between the support points. This is caused by nearly singular Vandermonde matrix for just moderate number of support points. This problem, which is known as Runge phenomenon (see the relevant discussion in Appendix B.2) in literature, can be resolved by using support points based on roots of Jacobi polynomials $\boldsymbol{P}_n(x;\alpha,\beta)$ whose most important representatives can be found in Legendre and Chebyshev polynomials. In a similar spirit, we will introduce grid points that show a benevolent asymptotic behavior of the Lebesgue constant.

Minimizing the Lebesgue constant is strictly related to the conditioning of the generalized Vandermonde matrix V, which can be concluded from uniqueness of the polynomial interpolation: setting $\hat{x} = e_i$ in (6.43) and (6.41) we have ${V_{\cdot,i}}^T \boldsymbol{L}(t) = \widetilde{\boldsymbol{P}}_{i-1}(t)$, where the vector $\boldsymbol{L}(t) \in \mathbb{R}^{N_p}$ is given as $\boldsymbol{L}(t) = [\boldsymbol{L}_1(t), \ldots, \boldsymbol{L}_{N_p}(t)]^T$ and $V_{\cdot,i}$ denotes the i–th column of V. Varying $\hat{x} = e_i$ for $i \in [N_p]$ then yields the linear system

$$V^T \boldsymbol{L}(t) = \widetilde{\boldsymbol{P}}(t)$$

with $\widetilde{\boldsymbol{P}}(t) = [\widetilde{\boldsymbol{P}}_0(t), \ldots, \widetilde{\boldsymbol{P}}_N(t)]^T$. We seek a solution $\boldsymbol{L}(t)$ in order to be able to determine reasonable ways how Λ_N can be minimized. Recalling Cramer's rule for solving linear equation

systems we get

$$\boldsymbol{L}_i(t) = \frac{\mathrm{Det}\left[(V^T)_{\cdot,1}, \ldots, (V^T)_{\cdot,i-1}, \widetilde{\boldsymbol{P}}(t), (V^T)_{\cdot,i+1}, \ldots, (V^T)_{\cdot,N_p}\right]}{\mathrm{Det}(V^T)},$$

which suggests to calculate the grid points t_n such that the denominator (i.e., the determinant of V) is maximized. The associated node set is given in a simple form (see Hesthaven [232] and Hesthaven et al. [234]) as the N_p zeros of function

$$(1-t^2)\widetilde{\boldsymbol{P}}'_N(t)$$

or, in other words the interval endpoints -1 and $+1$ together with the candidates for extrema of the N–th order (normalized) Legendre polynomials. This grid point set $\mathcal{F}$ is known as the Fekete node set and coincides with Legendre–Gauss–Lobatto quadrature points (see Appendix B.3.2). According to Sündermann [420] an upper bound for $\Lambda_N(\mathcal{F})$ is given by

$$\Lambda_N(\mathcal{F}) \leqslant C \log(N+1)$$

with a positive (undetermined) constant C. Based on numerical experiments Hesthaven [232] conjectures the upper bound

$$\Lambda_N(\mathcal{F}) \leqslant \frac{2}{\pi}\log(N+1) + 0.685.$$

So far, we found that the Lebesgue constant grows like 2^N for the equidistant nodes and like $\log N$ in case of Fekete grid points. Deeper investigations (see e.g. Hesthaven [233]) have shown that a merely logarithmic growth of the Lebesgue constant similar to the one of Fekete type can be found for other families of points. These include Gauss–Lobatto points for the symmetric Jacobi polynomials $\boldsymbol{P}_n(x;\alpha,\alpha)$ with Legendre polynomials ($\alpha = 0$) and Chebyshev polynomials ($\alpha = -\frac{1}{2}$) as special cases. Lebesgue constants Λ_N^{LG} and Λ_N^{CG} of Legendre–Gauss resp. Chebyshev–Gauss quadrature points are $\mathcal{O}\left(\sqrt{N}\right)$ and $\mathcal{O}(\log N)$, respectively, cf. Szegö [421, p. 336] and Rivlin [371, p. 90]. Vértesi [437] could show that Λ_N^{CG} is very close to the smallest possible Lebesgue constant Λ_N^*, where

$$\Lambda_N^{CG} = \frac{2}{\pi}\left(\log N + \gamma + \log\frac{8}{\pi}\right) + \mathrm{o}(1) \quad \text{and} \quad \Lambda_N^* = \frac{2}{\pi}\left(\log N + \gamma + \log\frac{4}{\pi}\right) + \mathrm{o}(1).$$

In 1981 Vértesi [436] found a bound for the Lebesgue constants of Gauss quadrature points – based on the roots of a Jacobi polynomial – augmented by either $t = -1$ or $t = +1$. This bound is $\mathcal{O}\left(\log(N)\cdot\sqrt{N}\right)$ and coincides with the one for (flipped) Radau quadrature points, cf. Vértesi [436, Theorem 2.1]. Recently this bound was sharpened by Hager et al. [216] to $\mathcal{O}\left(\sqrt{N}\right)$. Moreover, it was shown that the Lebesgue constants of (flipped) Radau quadrature points augmented by the missing interval endpoint are even $\mathcal{O}(\log N)$, and therefore are of the same order of magnitude as Fekete/Lobatto points, cf. Hager et al. [216, Theorem 6.1].

Derivatives Expansion Coefficient Transformation So far we have discussed the advantages of using orthogonal polynomials to construct a modal basis. By applying an orthogonalization approach to the monomial basis we found the Legendre polynomials $\widetilde{\boldsymbol{P}}_n(\cdot)$ as a representative of an orthogonal basis. Chebyshev polynomials of first kind represent another well–known orthogonal basis, cf. Appendix B.1. They are orthogonal with respect to the scalar product $\langle\cdot,\cdot\rangle_{\boldsymbol{\omega}}$ with $\boldsymbol{\omega}(t) = \frac{1}{\sqrt{1-t^2}}$.
Regarding Equation (6.31) we require that expansion coefficients $\{\hat{x}_n\}_{n=0}^N$ can be transferred to derivative expansion coefficients $\{\hat{y}_n\}_{n=0}^N$ computationally cheap. Both, Legendre polynomials and Chebyshev polynomials meet this requirement: formulae (see e.g. Gottlieb and Orszag [206, Appendix]) relating expansion coefficients $\hat{x}_n$ in the approximation series

$$\boldsymbol{X}_N(t) = \sum_{n=0}^{N} \hat{x}_n \boldsymbol{\psi}_n(t)$$

to expansion coefficients $\hat{y}_n$ of

$$\dot{\boldsymbol{X}}_N(t) = \sum_{n=0}^{N} \hat{y}_n \boldsymbol{\psi}_n(t)$$

are given as follows:

(i) Legendre polynomials ($\boldsymbol{\psi}_n = \boldsymbol{P}_n$):

$$\hat{y}_n = (2n+1) \sum_{\substack{m=n+1 \\ m+n \text{ odd}}}^{N} \hat{x}_m, \quad n = 0, \dots, N.$$

(ii) Chebyshev polynomials ($\boldsymbol{\psi}_n = \boldsymbol{T}_n$):

$$\hat{y}_n = p_n \sum_{\substack{m=n+1 \\ m+n \text{ odd}}}^{N} m \cdot \hat{x}_m, \quad n = 0, \dots, N, \quad p_n = \begin{cases} 1, & n = 0, \\ 2 & \text{otherwise.} \end{cases}$$

Convergence Properties Now we discuss how well functions and their derivatives can be approximated by polynomials. The associated theory is called *polynomial approximation theory* and one has to distinguish between approximations of smooth functions and approximations of functions involving discontinuities. Theory regarding the former case was developed mainly by Canuto and Quarteroni (see e.g. [97, 98]), and by Bernardi and Maday (see e.g. [59, 60, 61]). Excellent surveys dealing with polynomial approximation theory for smooth functions are provided by the overview paper of Bernardi and Maday [61] and the texts of Canuto [96] and Canuto et al. [100].
Fundamentals of polynomial approximation are mandatory in order to be able to to analyze our global approach, where we replace functions $\boldsymbol{x}(\cdot)$ by global polynomials $\boldsymbol{X}_N(\cdot)$ of order N: we found the estimate (6.44) which shows that the approximation quality measured in the

maximum norm is bounded by the product of the Lebesgue constant Λ_N and $\left\|\boldsymbol{x} - \boldsymbol{X}_N^*\right\|_\infty$, namely the error between the function $\boldsymbol{x}(\cdot)$ and its best polynomial approximation of order N that we denote by $\boldsymbol{X}_N^*$. The analysis of Λ_N which is strictly related to the choice of the grid points has been done in previous sections. It is still left to deal with the approximability of functions by polynomials.
On page 173 we argued that it is reasonable to deal with *interpolations* of $\boldsymbol{x}(\cdot)$. However, before analyzing polynomial approximations which are interpolatory we start our discussion by deriving estimates which are related to *projections*.
We consider the question how well a function $\boldsymbol{x} \in L^2(\mathcal{I}, \mathbb{R})$ can be represented by the polynomial function

$$\widetilde{\boldsymbol{X}}_N(t) \stackrel{\text{def}}{=} P_N(\boldsymbol{x}) = \sum_{n=0}^{N} \widetilde{x}_n \widetilde{\boldsymbol{P}}_n(t), \quad \widetilde{x}_n = \left\langle \boldsymbol{x}, \widetilde{\boldsymbol{P}}_n \right\rangle = \int_{-1}^{+1} \boldsymbol{x}(t) \widetilde{\boldsymbol{P}}_n(t)\, \mathrm{d}t. \tag{6.45}$$

Note that – compared to previous representation (6.38)+(6.43) – the sum index is now running from 0 to N rather than from 1 to $N_p = N + 1$. This is due to simplicity of notation and to use conventional terms. As an immediate consequence of the representations (6.29) and (6.45) we find

$$\left\|\boldsymbol{x} - \widetilde{\boldsymbol{X}}_N\right\|_0^2 = \sum_{n=N+1}^{\infty} |\widetilde{x}_n|^2,$$

which can be recognized as Parseval's identity. Since (6.45) coincide with the first $N+1$ addends of the infinite sum in (6.29) we call $\boldsymbol{x} - \widetilde{\boldsymbol{X}}_N$ *truncation error*.

Exponential Convergence for Smooth Functions A fundamental result quantifying the truncation error (with respect to the L^2–norm) is provided by the following result:

Theorem 6.9
For any real $p \geqslant 0$ there exists a constant C such that

$$\left\|\boldsymbol{x} - \widetilde{\boldsymbol{X}}_N\right\|_0 \leqslant C\, N^{-p}\, |\boldsymbol{x}|_{H^p;N} \quad \forall \boldsymbol{x} \in H^p(\mathcal{I}, \mathbb{R}). \tag{6.46}$$

△

Proof See Canuto and Quarteroni [98, Theorem 2.3]. □

Note that Theorem 6.9 was proven in Canuto and Quarteroni [98] for the full norm $\|\cdot\|_{H^p}$ on the right–hand side. However, according to Lemma 6.5 the norm is equivalent to the semi–norm $|\cdot|_{H^p;N}$ in our setting such that Theorem 6.9 holds true. Trivially it holds $|\boldsymbol{x}|_{H^p;N} \leqslant \|\boldsymbol{x}\|_{H^p}$ such that for all $\boldsymbol{x} \in H^p(\mathcal{I}, \mathbb{R})$ we have

$$\left\|\boldsymbol{x} - \widetilde{\boldsymbol{X}}_N\right\|_0 \leqslant C\, N^{-p}\, \|\boldsymbol{x}\|_{H^p}.$$

with the same constant C as (6.46).
Using the estimate by means of the semi–norm enables us to show that the projection operator P_N is exact for polynomials in $\mathcal{S}_N$: if we choose $p = N + 1$ in (6.46) then this yields (see (6.28))

that the condition $|\boldsymbol{x}|_{H^{N+1;N}} = \left\|\boldsymbol{x}^{(N+1)}\right\|_2 = 0$ is equivalent to $\boldsymbol{x}^{(N+1)}$ vanishing identically in $\mathcal{I}$. The latter condition in turn is equivalent to $\boldsymbol{x}(\cdot)$ being a polynomial of degree $\leqslant N$. Hence, estimate (6.46) states that for $\boldsymbol{x} \in \mathcal{S}_N$ (for which $|\boldsymbol{x}|_{H^{N+1;N}} = 0$ holds) it holds $\boldsymbol{x} - P_N(\boldsymbol{x}) = 0$, i.e., $P_N(\boldsymbol{x}) = \boldsymbol{x}$.
The next result extends Theorem 6.9 to higher order Sobolev norms and therefore investigates those cases, where the truncation error of the derivatives must be considered. Similar to (6.46) we present the estimate with respect to the semi–norm whereas the original result was proven using the full norm.

Theorem 6.10
For any real q and p fulfilling $0 \leqslant q \leqslant p$ it holds for a constant C that

$$\left\|\boldsymbol{x} - \widetilde{\boldsymbol{X}}_N\right\|_{H^q} \leqslant C\, N^{\boldsymbol{e}(q)-p}\, |\boldsymbol{x}|_{H^{p;N}} \qquad \forall \boldsymbol{x} \in H^p(\mathcal{I}, \mathbb{R}),$$

where $\boldsymbol{e}(q)$ is given by

$$\boldsymbol{e}(q) = \begin{cases} \frac{3}{2}q, & 0 \leqslant q \leqslant 1, \\ 2q - \frac{1}{2}, & q \geqslant 1. \end{cases}$$

△

Proof See Canuto and Quarteroni [98, Theorem 2.4]. □

In particular we conclude from Theorem 6.10 that convergence is very fast for smooth functions $\boldsymbol{x}(\cdot)$ (this means $\boldsymbol{x} \in H^p(\mathcal{I}, \mathbb{R})$ for a p large), and even exponential for analytic functions, cf. Tadmor [422]. An exponentially fast error decay for increasing N is also one of the appealing trademarks of classic spectral methods, cf. Hesthaven et al. [234].
Since first order derivatives are involved in our OCPs we try to get a deeper insight into function approximation theory. Schwab [396] provides the estimate

$$\left\|\boldsymbol{x} - \widetilde{\boldsymbol{X}}_N\right\|_0 \leqslant \left[\frac{(N+1-s)!}{(N+1+s)!}\right]^{\frac{1}{2}} \left\|\boldsymbol{x}^{(s)}\right\|_{L^2_s} \text{ with } \left\|\boldsymbol{x}^{(s)}\right\|^2_{L^2_s} \stackrel{\text{def}}{=} \int_{-1}^{+1} \left|\boldsymbol{x}^{(s)}(t)\right|^2 \left(1-t^2\right)^s \, \mathrm{d}t,$$

which holds for all $0 \leqslant s \leqslant \min(p, N+1)$. It was also shown by Schwab [396] that for $\boldsymbol{x} \in H^p(\mathcal{I}, \mathbb{R})$, $p \geqslant 1$, which can be expressed in the form (6.45), it holds

$$\int_{-1}^{+1} \left|\boldsymbol{x}^{(s)}(t)\right|^2 \left(1-t^2\right)^s \, \mathrm{d}t = \sum_{n \geqslant s} |\widetilde{x}_n|^2 \, \frac{(n+s)!}{(n-s)!} \leqslant |\boldsymbol{x}|^2_{H^s}, \qquad 0 \leqslant s \leqslant p.$$

This yields the estimate

$$\left\|\boldsymbol{x} - \widetilde{\boldsymbol{X}}_N\right\|_0 \leqslant \left[\frac{(N+1-s)!}{(N+1+s)!}\right]^{\frac{1}{2}} |\boldsymbol{x}|_{H^s}, \qquad s = \min(p, N+1),$$

which is generalized in the following result.

Lemma 6.11
For $\boldsymbol{x} \in H^p(\mathcal{I}, \mathbb{R})$ with $p \geqslant 1$ it holds

$$\left\| \boldsymbol{x}^{(q)} - \widetilde{\boldsymbol{X}}_N^{(q)} \right\|_0 \leqslant \left[\frac{(N+1-s)!}{(N+1+s-4q)!} \right]^{\frac{1}{2}} |\boldsymbol{x}|_{H^s} \tag{6.47}$$

with $s = \min(N+1, p)$ and $q \leqslant p$. △

Proof See Hesthaven [233, Lemma 4.4]. □

When applying the well–known Stirling formula to inequality (6.47) we can state that

$$\left\| \boldsymbol{x}^{(q)} - \widetilde{\boldsymbol{X}}_N^{(q)} \right\|_0 \leqslant N^{2q-p} \, |\boldsymbol{x}|_{H^p}$$

in the limit of $N \gg p$.
Inequality (6.44) shows the importance of quantifying the best approximation error $\boldsymbol{X}_N^*(\cdot)$. The best approximation polynomial of $\boldsymbol{x}(\cdot)$ with respect to the L^2–norm is given by the truncated Legendre polynomial $\widetilde{\boldsymbol{X}}_N(\cdot)$. This can be generalized as follows: let X any normed vector space and $\boldsymbol{x} \in X$ arbitrary. Then it is a well–known fact that there exists a polynomial $\boldsymbol{X}^* \in \mathcal{S}_N$ such that

$$\|\boldsymbol{x} - \boldsymbol{X}^*\|_X = \inf_{\boldsymbol{X} \in \mathcal{S}_N} \|\boldsymbol{x} - \boldsymbol{X}\|_X \,,$$

and we call $\boldsymbol{X}^*(\cdot)$ the *best approximation polynomial* of $\boldsymbol{x}(\cdot)$ in the norm of X. In particular we are interested in the cases $X = L^p(\mathcal{I}, \mathbb{R})$, $2 < p \leqslant \infty$, for which $\boldsymbol{X}^*(\cdot)$ is unique (see Nikol'skii [339, Theorem 1.3.6] if $1 < p < \infty$, and Timan [426, p. 35–40] if $p = 1$ or $p = \infty$). The following result, which was proven in Jackson [254] for $p = \infty$ and in its full generality in Quarteroni [364], provides estimates for the best approximation error in any L^p–norm and verifies a decay of this error as in the L^2–norm.

Theorem 6.12
Let $\boldsymbol{x} \in W^{q,p}(\mathcal{I}, \mathbb{R})$ for some $q \geqslant 0$ and $2 \leqslant p \leqslant \infty$. Then for any $N \geqslant 0$ there exists a positive C which is independent of N and $\boldsymbol{x}(\cdot)$ such that

$$\inf_{\boldsymbol{X} \in \mathcal{S}_N} \|\boldsymbol{x} - \boldsymbol{X}\|_p \leqslant C N^{-q} \left[\sum_{i=\min(q,N+1)}^{q} \left\| \boldsymbol{x}^{(i)} \right\|_p^p \right]^{1/p} . \tag{6.48}$$

△

Proof See Quarteroni [364, Theorem 3]. □

Note, however, that the truncation error in L^p–norms, $p > 2$, converges not as fast as the best approximation does. If we consider for example a function $\boldsymbol{x}(\cdot)$ whose q–th derivative is of bounded variation, one can show (see Jackson [254, Theorem XV]) that

$$\left\| \boldsymbol{x} - \widetilde{\boldsymbol{X}}_N \right\|_\infty \leqslant C \, N^{1/2-q} \, TV(\boldsymbol{x}^{(q)}, \mathcal{I}) \tag{6.49}$$

holds. Comparing estimates (6.49) and (6.48) for $p = \infty$ shows a rate of convergence of the best approximation which is faster by at least a factor of $\sqrt{N}$.
Up to now we have concentrated on convergence results that were related to projections. In the next step we deal with the interpolation operator. First we recall the transformation between modal values $\hat{x}$ and nodal values x that are interrelated (see (6.41)+(6.42)) via the equation

$$x = V\hat{x}.$$

In order to distinguish between polynomial representations arising from interpolation and projection we use the notations

$$\hat{\boldsymbol{X}}_N(t) = I_N(\boldsymbol{x}) = \sum_{n=0}^{N} \hat{x}_n \widetilde{\boldsymbol{P}}_n(t), \qquad \widetilde{\boldsymbol{X}}_N(t) = P_N(\boldsymbol{x}) = \sum_{n=0}^{N} \widetilde{x}_n \widetilde{\boldsymbol{P}}_n(t).$$

Exploiting the interpolation property we calculate

$$(V\hat{x})_i = \hat{\boldsymbol{X}}_N(t_i) = \boldsymbol{x}(t_i) = \sum_{n=0}^{\infty} \widetilde{x}_n \widetilde{\boldsymbol{P}}_n(t_i) = \sum_{n=0}^{N} \widetilde{x}_n \widetilde{\boldsymbol{P}}_n(t_i) + \sum_{n=N+1}^{\infty} \widetilde{x}_n \widetilde{\boldsymbol{P}}_n(t_i)$$

such that it holds

$$V\hat{x} = V\widetilde{x} + \sum_{n=N+1}^{\infty} \widetilde{x}_n \widetilde{\boldsymbol{P}}_n(t), \quad t = [t_0, \ldots, t_N]^T.$$

A reformulation of the previous equation yields

$$\hat{\boldsymbol{X}}_N(t) = \widetilde{\boldsymbol{X}}_N(t) + \widetilde{\boldsymbol{P}}^T(t) V^{-1} \sum_{n=N+1}^{\infty} \widetilde{x}_n \widetilde{\boldsymbol{P}}_n(t), \quad \widetilde{\boldsymbol{P}} = [\widetilde{\boldsymbol{P}}_0, \ldots, \widetilde{\boldsymbol{P}}_N]^T. \tag{6.50}$$

The difference between $\hat{\boldsymbol{X}}_N(\cdot)$ and $\widetilde{\boldsymbol{X}}_N(\cdot)$ is known as the *aliasing error*:

$$A_N(\boldsymbol{x}) \stackrel{\text{def}}{=} I_N(\boldsymbol{x}) - P_N(\boldsymbol{x}) = \hat{\boldsymbol{X}}_N(t) - \widetilde{\boldsymbol{X}}_N(t).$$

Under the assumption $\boldsymbol{x} \in H^p(\mathcal{I}, \mathbb{R})$ with $p > \frac{1}{2}$ (see Schwab [396]) and using (6.50) we find the following expression for the aliasing error:

$$A_N(\boldsymbol{x}) = \widetilde{\boldsymbol{P}}^T(t) V^{-1} \sum_{n=N+1}^{\infty} \widetilde{x}_n \widetilde{\boldsymbol{P}}_n(t) = \sum_{n=N+1}^{\infty} \widetilde{x}_n \left[\widetilde{\boldsymbol{P}}^T(t) V^{-1} \widetilde{\boldsymbol{P}}_n(t) \right], \tag{6.51}$$

where the expression in brackets can be written in terms of the first $N+1$ polynomials $\widetilde{\boldsymbol{P}}_m(\cdot)$, $0 \leqslant m \leqslant N$ as

$$\widetilde{\boldsymbol{P}}^T(t) V^{-1} \widetilde{\boldsymbol{P}}_n(t) = \sum_{m=0}^{N} \widetilde{y}_m \widetilde{\boldsymbol{P}}_m(t), \qquad V\widetilde{y} = \widetilde{\boldsymbol{P}}_n(t). \tag{6.52}$$

Hence, the aliasing error can be interpreted as those higher–order contributions (n > N) that look like lower order modes on the grid. The aliasing error $A_N(\boldsymbol{x})$ contains the modes with numbers $n \leqslant N$, while the reminder term $\boldsymbol{x} - \widetilde{\boldsymbol{X}}_N$ contains only the modes with $n > N$. For this reason they are orthogonal and the Pythagoras theorem can be applied such that we obtain

$$\left\|\boldsymbol{x} - \hat{\boldsymbol{X}}_N\right\|_2^2 = \left\|\boldsymbol{x} - \widetilde{\boldsymbol{X}}_N - A_N(\boldsymbol{x})\right\|_2^2 = \left\|\boldsymbol{x} - \widetilde{\boldsymbol{X}}_N\right\|_2^2 + \|A_N(\boldsymbol{x})\|_2^2 .$$

Combining (6.51) and (6.52) we identify the representation

$$A_N(\boldsymbol{x}) = \sum_{n=N+1}^{\infty} \left[I_N\left(\widetilde{\boldsymbol{P}}_n\right)\right] \widetilde{x}_n$$

such that the aliasing error can be also interpreted as the error which is introduced by using the interpolation of the basis, rather than the basis itself to represent the higher order modes. The fact that it cannot be distinguished between lower and higher order modes on a finite grid justifies the term aliasing error.
Next we give some estimates for the interpolation error $\boldsymbol{x} - \hat{\boldsymbol{X}}_N$. Analogously to Theorem 6.9, which quantifies the truncation error, the following result holds:

Theorem 6.13
For any real $p \geqslant 1$ there exists a constant C such that

$$\left\|\boldsymbol{x} - \hat{\boldsymbol{X}}_N\right\|_2 \leqslant C\, N^{-p}\, |\boldsymbol{x}|_{H^{p;N}} \quad \forall \boldsymbol{x} \in H^p(\mathcal{I}, \mathbb{R}). \tag{6.53}$$

△

Proof See Bernardi and Maday [60]. □

Considering Theorems 6.9 and 6.13 one can see that truncation and interpolation error behave asymptotically equivalent with respect to the L^2–norm. In accordance with Theorem 6.10 for the truncation error the generalization of (6.53) is presented in the following result:

Theorem 6.14
For any real q and p fulfilling $1 \leqslant q \leqslant p$ it holds for a constant C that

$$\left\|\boldsymbol{x} - \hat{\boldsymbol{X}}_N\right\|_{H^q} \leqslant C\, N^{2q-1/2-p}\, |\boldsymbol{x}|_{H^{p;N}} \quad \forall \boldsymbol{x} \in H^p(\mathcal{I}, \mathbb{R}),$$

△

Proof See Bernardi and Maday [60]. □

Since first–order derivatives of state trajectory functions $\boldsymbol{x}(\cdot)$ enter the ODEs in our OCPs we need to know how well they are approximated if we apply a global approach. It is clear that those derivatives are usually approximated by the Legendre *projection derivative*, $P_N(\boldsymbol{x})' = \frac{\mathrm{d}}{\mathrm{d}t}\widetilde{\boldsymbol{X}}_N(t)$, or by the Legendre *interpolation derivative*, $I_N(\boldsymbol{x})' = \frac{\mathrm{d}}{\mathrm{d}t}\hat{\boldsymbol{X}}_N(t)$, where projection and interpolation are different from each other in general. We can estimate (see Bernardi and Maday [60]) the error between $\dot{\boldsymbol{x}}(\cdot)$ and the Legendre interpolation derivative of $\boldsymbol{x}(\cdot)$ in terms of N and the regularity of $\boldsymbol{x}(\cdot)$ as

$$\left\|\dot{\boldsymbol{x}} - \frac{\mathrm{d}}{\mathrm{d}t}\hat{\boldsymbol{X}}_N\right\|_2 \leqslant C\, N^{1-p}\, |\boldsymbol{x}|_{H^{p;N}} .$$

So far we have considered the polynomial approximation of smooth functions and have shown that convergence is very fast and even exponential (spectral convergence) for analytic functions. In the final section of this chapter we briefly investigate the case of approximating non–smooth functions by polynomials.

Gibbs Phenomenon We have seen that the global approach is suitable for problems that have smooth solutions. Since we are dealing with OCPs, that are subject to explicit and implicit switches, we expect at least discontinuities of certain differential state trajectory derivatives and even jumps of control trajectories (e.g. bang–bang controls) indicating explicit or implicit switching modes. For this reason we investigate the polynomial approximation of non–smooth functions in this section.
Based on the example of the sign function, which can be seen as a prototypical jump function, the polynomial approximation of non–smooth functions is analyzed. To this end let us consider the Legendre polynomial expansion of the sign function which is given as

$$\operatorname{sgn}(t) = \sum_{n=0}^{\infty} \frac{(-1)^n (4n+3)(2n)!}{2^{2n+1}(n+1)!n!} \boldsymbol{P}_{2n+1}(t). \tag{6.54}$$

The partial sums, $P_N(\operatorname{sgn})$, for some N are plotted in Figure 6.6. The overshoot, which can be observed around $t = 0$ in Figure 6.6, occurs whenever functions having discontinuities are expanded in or interpolated with smooth functions.
Nowadays this observation, which was noted by Wilbraham [449] for the first time, is known as Gibbs phenomenon. Literature failed to notice Wilbraham's discovery. In 1898 Michelson and Stratten, who developed a mechanical Fourier analyzer, detected overshoots in output plots of their analyzer, cf. Michelson and Stratton [325]. As a consequence Michelson sent a letter to *Nature* in which he inquired convergence properties of Fourier series for discontinuous functions. In reply, Gibbs published a paper where he could describe the overshoot at the point of discontinuity, cf. Gibbs [195]. For more historical background information we refer the reader to Hewitt and Hewitt [235].
Coming back to our example we observe three features:

(i) The Gibbs phenomenon near $t = 0$ in Figure 6.6 shows the same structural behavior as that for the Fourier series in Michelson and Stratton [325].

(ii) For $|t| < 1$, $t \neq 0$ the approximation error behaves like $\frac{1}{N}$ after N terms: for the $(2n+1)$-st Legendre coefficient, a_n, in (6.54) it holds

$$a_n = \frac{(-1)^n (4n+3)(2n)!}{2^{2n+1}(n+1)!n!} = \mathcal{O}\left(\frac{1}{\sqrt{n}}\right), \qquad n \to \infty,$$

and for fixed $|t| < 1$ we have (see (B.2) in Appendix B.1)

$$\boldsymbol{P}_n(t) = \mathcal{O}\left(\frac{1}{\sqrt{n}}\right), \qquad n \to \infty.$$

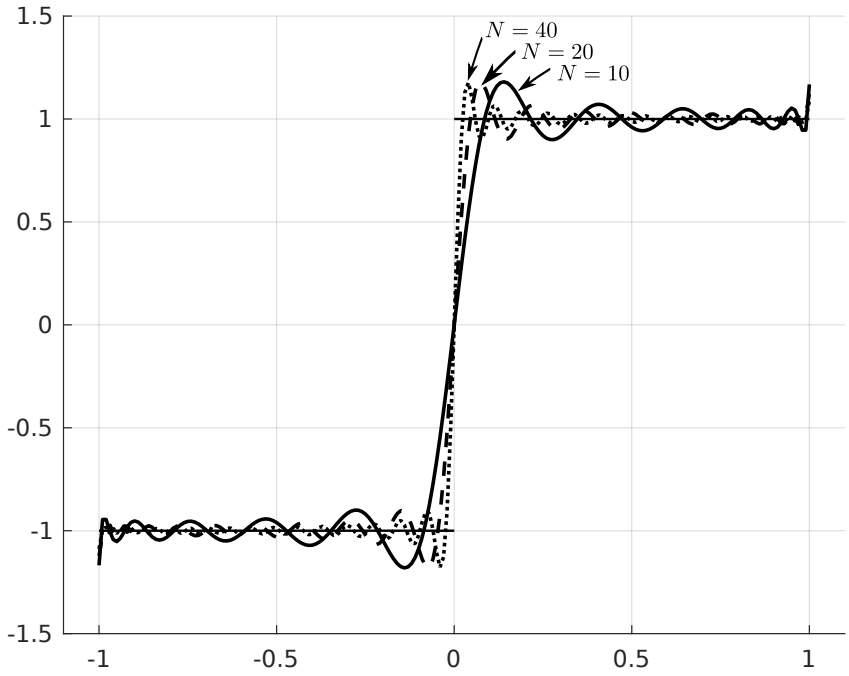


Figure 6.6: A plot depicting several partial sums of the Legendre series expansion for $\operatorname{sgn}(\cdot)$ (see (6.54)). More specifically we plot $P_N(\operatorname{sgn})$ for $N = 10, 20, 40$. Moreover, the function $\operatorname{sgn}(\cdot)$ itself is plotted. Note that the Gibbs phenomenon occurs around $t = 0$. Note furthermore that for fixed t with $t \neq 0, \pm 1$ the series converges like $\frac{1}{N}$. Around $t = \pm 1$ the series converges like $\frac{1}{\sqrt{N}}$.

For fixed $t \neq 0$ the series (6.54) is alternating such that the error after N terms is at most of order $a_N \boldsymbol{P}_N = \mathcal{O}\left(\frac{1}{N}\right)$.

(iii) For $t = \pm 1$ the series (6.54) converges only like $\frac{1}{\sqrt{N}}$ since $\boldsymbol{P}_n(\pm 1) = (\pm 1)^n$ for all n. From this insight we can conclude that the Gibbs phenomenon arising from interval–interior function discontinuities has a sweeping impact on the rate of convergence even at the interval endpoints $t = \pm 1$. Contrary, the error of the Chebyshev expansion of $\operatorname{sgn}(\cdot)$ decays like $\frac{1}{N}$ at $t = \pm 1$, cf. Gottlieb and Orszag [206]. For this reason, the boundary errors of Chebyshev expansions decay to zero roughly a factor $\frac{1}{\sqrt{N}}$ faster than Legendre expansion boundary errors.

The results that we found for the Legendre series expansion of the $\operatorname{sgn}(\cdot)$ function can be generalized. Table 6.2, which can be found in Fornberg [169, p. 13], shows the order of the maximum norm of errors away from function irregularities (discontinuities of the n–th

order derivative $f^{(n)}(\cdot)$, $n \geqslant 0$). They coincide with the decay rates of Legendre expansion coefficients.

	Max–norm of errors (order)	
Function	Near irregularity	Away from irregularity
f discontinuous	1	$1/N$
f' discontinuous	$1/N$	$1/N^2$
f'' discontinuous	$1/N^2$	$1/N^3$
$\vdots$	$\vdots$	$\vdots$
f analytic	$\exp(-C \cdot N)$, $C > 0$	

Table 6.2: Order of max–norm error for Gibbs phenomenon depending on the order of function irregularities.

6.3.3 Conclusion

In this section we draw some conclusions from what we found out about both the local and the global approach such that we are able to develop a numerical solution algorithm that would be most suitable for solving OCPs with explicit and implicit switches within our new framework (see Chapter 11).

Local approach algorithms are employed as *h methods* where states and controls are approximated by fixed low–degree polynomials (depends on the chosen numerical integration method), and the problem is divided into segments (finite elements). Convergence of the discretization is then achieved by increasing the number of finite elements. This is usually done in such a way that grid refinement takes place in regions of the horizon where errors are the largest, cf. Betts and Huffman [65], Jain and Tsiotras [257], Betts [62], Zhao and Tsiotras [471]. A novel strategy, which allows for a goal–oriented error estimation, is presented in this thesis (see Chapter 10).

In recent years, global approach algorithms and in particular pseudospectral collocation methods have gained in popularity, cf. Elnagar et al. [145], Benson [52], Huntington [245], Kameswaran and Biegler [262], Garg [179], Francolin [172]. Here, the collocation points are chosen to be quadrature points of accurate quadrature rules (Gauss quadrature). Basis functions are usually of Chebyshev or Legendre type. Contrary to h methods pseudospectral collocation methods are usually employed as *p methods*, where there exists just a single segment, and convergence of the discretization scheme is achieved by increasing the degree p of the polynomial. Under the assumption of well–behaved and smooth problems p methods – in accordance with our analysis about approximation theory of smooth functions – converge exponentially, cf. Fornberg [169], Canuto et al. [99]. Representatives of p methods can be found in the *Gauss Pseudospectral Method (GPM)* (see [52, 53]), the *Radau Pseudospectral Method (RPM)* (see [262, 181]), and the *Lobatto Pseudospectral Method (LPM)* (see [145]).

Pseudospectral methods applied as p methods suffer from several drawbacks. For smooth problems accurate solutions can often be just achieved if the polynomial degrees are chosen

very large. Furthermore, our problem formulation of OCPs with explicit and implicit switches results in non–smooth state and control trajectories such that convergence rates of p methods may be rather poor. Using high degree polynomials in p methods makes NLP constraint Jacobians and Hessians growing a lot faster in both size and density than the number of collocation points. Even though convergence of p methods could be obtained we would advise against this approach since the number of non–zeros in NLP derivatives would make a solution computationally intractable and inefficient.
As a solution of the aforementioned problems we propose a discretization approach that uses elements from both h methods and p methods: the horizon is split into finite elements and a p method is applied to each of these elements. We obtain a full horizon solution by interlinking the single finite elements solutions by enforcing matching conditions – a technique that we have already seen when we were describing the transition from direct single shooting to direct multiple shooting. The resulting approach is then called a *hp pseudospectral collocation method* where convergence can be achieved by increasing both the number of finite elements and the polynomial degree in single elements. Here we pursue the following strategy: we apply our novel goal–oriented error estimation and a switch detection algorithm to a fixed problem discretization. Switches indicate bang–bang controls and as a consequence non–smooth solutions. We therefore modify the finite element grid in regions containing switches. In regions, that are presumably smooth, the polynomial degree is increased in order to exploit the spectral convergence of smooth solutions. The element–wise error contributions provided by our a posteriori error estimation allow for an equidistribution of the local discretization error.

Part II

Contributions

Chapter 7

A Local Multi-Degree Pseudospectral Method

Based on the concluding remarks of the previous chapter we present a local pseudospectral method for a rather general OCP formulation. The continuous OCP under consideration is introduced in Section 7.1. In order to end up with a proper formulated OCP and for the sake of convenience the original OCP formulation with free start and final time is transformed into an OCP on a fixed interval.

In Section 7.2 we apply a global orthogonal collocation approach to our OCP formulation. The collocation points are chosen to be flipped LGR quadrature points. This approach, namely the *Flipped Radau Pseudospectral Method (FRPM)*, has proven to be successful and numerically stable, cf. Huntington et al. [246] and Garg et al. [180]. Garg et al. [181] could show that the FRPM defines an implicit integration scheme, and therefore it provides the ability to obtain highly accurate solution approximations particularly in the presence of stiff ODE systems (L–stable, see Huynh [247]). Moreover, the scheme is also algebraically stable, cf. Ascher and Petzold [16]. For some OCPs the FRPM is superior to GPM since the latter method leads to oscillatory trajectories while the former one produces much smoother trajectories, cf. Bausa and Tsatsaronis [35]. Also for inherent stability reasons we prefer the FRPM rather than the GPM approach. FRPM has been used to solve challenging problems in the fields of chemical process control and optimal control of aerospace systems, cf. Cervantes-Peredo [104], Raghunathan et al. [366].

In the literature, pseudospectral methods are direct discretization approaches where the polynomial degrees of state approximating polynomials are identical for all state components. The same holds for polynomials approximating the controls. In fact, even the polynomial degrees of differential states and controls coincide. Numerous ODE models arising from real world problems have state trajectories where some components require a rather high polynomial degree to obtain a sufficiently accurate approximation whereas for other components a low polynomial degree is already sufficient. Similarly, the switch–mode indicating control components in our novel framework are naturally chosen to be constant while other components might require higher degree polynomials. Section 7.3 addresses the question how a global collocation approach can be constructed where components of state and control approximations using global polynomials are of different polynomial degree. Difficulties arise especially with respect to efficient numerical implementations. Parallel to the research presented in this thesis we implemented a software that overcomes these difficulties by applying new tailored techniques.

Due to the nature of an OCP incorporating switches, the solutions are non–smooth in general. That is why a global collocation approach might show a poor convergence rate. For this reason we equip the global approach with elements from a local approach in the sense that the FRPM is applied segmentwise. By combining the resulting solutions from single horizon segments

via continuity constraints, we then obtain a full horizon solution. The details of this process are explained in Section 7.4.

7.1 Problem Formulation

This section introduces the continuous OCP formulation which is considered in this chapter. Note that – in the context of this thesis – the concrete choice of function spaces for states and controls in our OCP formulation does not affect subsequent discretization approaches. We therefore omit them.

Continuous–Time Bolza Optimal Control Problem In this chapter we investigate a rather general form of an OCP with free start and final time. We start this section with a naive and mathematically improper but coherent formulation of the problem. Let

$$
\begin{aligned}
\boldsymbol{\varphi} &: \mathbb{R} \times \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R} \times \mathbb{R}^{n_\mathrm{x}} \longrightarrow \mathbb{R}, \\
\boldsymbol{\psi} &: \mathcal{T} \times \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{u}} \longrightarrow \mathbb{R}, \\
\boldsymbol{f} &: \mathcal{T} \times \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{u}} \longrightarrow \mathbb{R}^{n_\mathrm{x}}, \\
\boldsymbol{c} &: \mathcal{T} \times \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{u}} \longrightarrow \mathbb{R}^{n_\mathrm{c}}, \\
\boldsymbol{r} &: \mathbb{R} \times \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R} \times \mathbb{R}^{n_\mathrm{x}} \longrightarrow \mathbb{R}^{n_\mathrm{r}}
\end{aligned}
$$

be mappings. In this contribution we only consider OCPs with finite horizon. In order to define an OCP with free start and final time we introduce scalar variables t_s and t_f and require them to be bounded, i.e., we have $t_\mathrm{s} \in [t_{s,l}, t_{s,u}]$ and $t_\mathrm{f} \in [t_{f,l}, t_{f,u}]$. With the additional condition $t_\mathrm{s} < t_\mathrm{f}$ a compact non–empty horizon interval is given by $\mathcal{T} \stackrel{\text{def}}{=} [t_\mathrm{s}, t_\mathrm{f}] \subset \mathbb{R}$. We consider the free–time continuous Bolza OCP

$$
\begin{aligned}
\min_{t_\mathrm{s}, t_\mathrm{f}, \boldsymbol{x}, \boldsymbol{u}} \quad & \boldsymbol{\varphi}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})) + \int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{\psi}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) \, \mathrm{d}t \\
\text{s. t.} \quad & \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), \qquad t \in \mathcal{T}, \\
& \boldsymbol{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), \qquad t \in \mathcal{T}, \\
& \boldsymbol{0}_{n_\mathrm{r}} = \boldsymbol{r}(t_\mathrm{s}, \boldsymbol{x}(t_\mathrm{s}), t_\mathrm{f}, \boldsymbol{x}(t_\mathrm{f})).
\end{aligned}
\tag{7.1}
$$

For OCP (7.1) we determine the control $\boldsymbol{u} : \mathcal{T} \longrightarrow \mathbb{R}^{n_\mathrm{u}}$, the state $\boldsymbol{x} : \mathcal{T} \longrightarrow \mathbb{R}^{n_\mathrm{x}}$, the start time t_s, and the final time t_f, that minimizes a Bolza objective functional subject to an ODE system, mixed control–state path constraints, and boundary constraints. Since t_s and t_f are variables of OCP (7.1) this is not a well–posed formulation. A time transformation to a fixed interval, such as $[-1, +1]$, will help us to overcome this problem.

Scaled Continuous–Time Optimal Control Problem – Global Approach An idea that we described in Section 5.1 enables us to transform OCP (7.1) into an equivalent OCP on a fixed time interval. The variable time domain $\mathcal{T} = [t_\mathrm{s}, t_\mathrm{f}]$ is mapped to the fixed interval $[-1, +1]$

by using the linear mapping $\boldsymbol{t} : [-1,+1] \longrightarrow \mathcal{T}$ which is defined as

$$\boldsymbol{t}(\tau; t_\mathrm{s}, t_\mathrm{f}) \stackrel{\text{def}}{=} \frac{t_f + t_s}{2} + \tau \cdot \frac{t_f - t_s}{2}, \tag{7.2}$$

and whose derivatives (including partial derivatives with respect to t_s and t_f) are given by

$$\boldsymbol{t}'(\tau; t_\mathrm{s}, t_\mathrm{f}) = \frac{t_f - t_s}{2}, \ \frac{\partial}{\partial t_\mathrm{s}}\boldsymbol{t}(\tau; t_\mathrm{s}, t_\mathrm{f}) = \frac{1}{2}(1-\tau), \ \frac{\partial}{\partial t_\mathrm{f}}\boldsymbol{t}(\tau; t_\mathrm{s}, t_\mathrm{f}) = \frac{1}{2}(1+\tau). \tag{7.3}$$

Note that the mapping $\boldsymbol{t}(\cdot; t_\mathrm{s}, t_\mathrm{f})$ is still valid for free start and final times. We use the linear mapping to convert OCP (7.1) to the time domain $[-1,+1]$: reparametrizations of $\boldsymbol{x}(\cdot)$ and $\boldsymbol{u}(\cdot)$ are then given by the functions $\widetilde{\boldsymbol{x}} : [-1,+1] \longrightarrow \mathbb{R}^{n_\mathrm{x}}$ and $\widetilde{\boldsymbol{u}} : [-1,+1] \longrightarrow \mathbb{R}^{n_\mathrm{u}}$ where $\widetilde{\boldsymbol{x}}(\tau) \stackrel{\text{def}}{=} \boldsymbol{x}(\boldsymbol{t}(\tau))$ and $\widetilde{\boldsymbol{u}}(\tau) \stackrel{\text{def}}{=} \boldsymbol{u}(\boldsymbol{t}(\tau))$. Using the definition

$$\boldsymbol{f}(\tau, \boldsymbol{x}(\tau), \boldsymbol{u}(\tau); t_\mathrm{s}, t_\mathrm{f}) \stackrel{\text{def}}{=} \boldsymbol{f}(\boldsymbol{t}(\tau; t_\mathrm{s}, t_\mathrm{f}), \widetilde{\boldsymbol{x}}(\tau), \widetilde{\boldsymbol{u}}(\tau)) \tag{7.4}$$

and in a similar fashion for $\boldsymbol{\psi}(\cdot)$ and $\boldsymbol{c}(\cdot)$ we can rewrite OCP (7.1) as follows: minimize the objective functional

$$J(\widetilde{\boldsymbol{x}}(\cdot), \widetilde{\boldsymbol{u}}(\cdot)) = \boldsymbol{\varphi}(t_\mathrm{s}, \widetilde{\boldsymbol{x}}(-1), t_\mathrm{f}, \widetilde{\boldsymbol{x}}(+1)) + \frac{t_f - t_s}{2} \int_{-1}^{+1} \boldsymbol{\psi}(\tau, \widetilde{\boldsymbol{x}}(\tau), \widetilde{\boldsymbol{u}}(\tau); t_\mathrm{s}, t_\mathrm{f}) \, \mathrm{d}\tau,$$

subject to the system dynamic constraints

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\widetilde{\boldsymbol{x}}(\tau) = \frac{t_f - t_s}{2} \cdot \boldsymbol{f}(\tau, \widetilde{\boldsymbol{x}}(\tau), \widetilde{\boldsymbol{u}}(\tau); t_\mathrm{s}, t_\mathrm{f}),$$

the mixed control–state constraints

$$\mathbf{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(\tau, \widetilde{\boldsymbol{x}}(\tau), \widetilde{\boldsymbol{u}}(\tau); t_\mathrm{s}, t_\mathrm{f}),$$

and the boundary conditions

$$\mathbf{0}_{n_\mathrm{r}} = \boldsymbol{r}(t_\mathrm{s}, \widetilde{\boldsymbol{x}}(-1), t_\mathrm{f}, \widetilde{\boldsymbol{x}}(+1)).$$

Without using the tilde notation the transformed OCP in a compact form reads as

$$\min_{t_\mathrm{s}, t_\mathrm{f}, \boldsymbol{x}, \boldsymbol{u}} \quad \boldsymbol{\varphi}(t_\mathrm{s}, \boldsymbol{x}(-1), t_\mathrm{f}, \boldsymbol{x}(+1)) + \frac{h}{2} \int_{-1}^{+1} \boldsymbol{\psi}(\tau, \boldsymbol{x}(\tau), \boldsymbol{u}(\tau); t_\mathrm{s}, t_\mathrm{f}) \, \mathrm{d}\tau \tag{7.5a}$$

$$\text{s.t.} \quad \frac{\mathrm{d}}{\mathrm{d}\tau}\boldsymbol{x}(\tau) = \frac{h}{2} \cdot \boldsymbol{f}(\tau, \boldsymbol{x}(\tau), \boldsymbol{u}(\tau); t_\mathrm{s}, t_\mathrm{f}), \quad \tau \in [-1,1], \tag{7.5b}$$

$$\mathbf{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(\tau, \boldsymbol{x}(\tau), \boldsymbol{u}(\tau); t_\mathrm{s}, t_\mathrm{f}), \quad \tau \in [-1,1], \tag{7.5c}$$

$$\mathbf{0}_{n_\mathrm{r}} = \boldsymbol{r}(t_\mathrm{s}, \boldsymbol{x}(-1), t_\mathrm{f}, \boldsymbol{x}(+1)), \tag{7.5d}$$

where we established the notation $h = t_\mathrm{f} - t_\mathrm{s}$. For later use we finally provide the partial derivatives of $\boldsymbol{f}(\tau, \boldsymbol{x}(\tau), \boldsymbol{u}(\tau); t_\mathrm{s}, t_\mathrm{f})$ with respect to t_s and t_f. Exploiting the definition (7.4) and (7.3) we find

$$\frac{\partial}{\partial t_\mathrm{s}} \boldsymbol{f}(\tau, \boldsymbol{x}(\tau), \boldsymbol{u}(\tau); t_\mathrm{s}, t_\mathrm{f}) = \frac{1}{2}(1-\tau) \cdot \boldsymbol{f}_t'(\tau, \boldsymbol{x}(\tau), \boldsymbol{u}(\tau); t_\mathrm{s}, t_\mathrm{f}),$$
$$\frac{\partial}{\partial t_\mathrm{f}} \boldsymbol{f}(\tau, \boldsymbol{x}(\tau), \boldsymbol{u}(\tau); t_\mathrm{s}, t_\mathrm{f}) = \frac{1}{2}(1+\tau) \cdot \boldsymbol{f}_t'(\tau, \boldsymbol{x}(\tau), \boldsymbol{u}(\tau); t_\mathrm{s}, t_\mathrm{f}).$$

Scaled Continuous–Time Optimal Control Problem – Local Approach Proceeding from OCP (7.5) we introduce a temporal grid in a next step, i.e., given any natural number N let $-1 = t_0 < t_1 < \ldots < t_N = +1$ define a fixed temporal grid. Single segments of the grid are denoted by $\mathcal{I}_n = [t_{n-1}, t_n]$, $1 \leqslant n \leqslant N$. By introducing segment–wise defined functions $\boldsymbol{x}^{(n)} : \mathcal{I}_n \longrightarrow \mathbb{R}^{n_\mathrm{x}}$ for states, and $\boldsymbol{u}^{(n)} : \mathcal{I}_n \longrightarrow \mathbb{R}^{n_\mathrm{u}}$ for controls such that

$$\boldsymbol{x}^{(n)} = \boldsymbol{x} \restriction_{\mathcal{I}_n}, \qquad \boldsymbol{u}^{(n)} = \boldsymbol{u} \restriction_{\mathcal{I}_n}, \qquad 1 \leqslant n \leqslant N,$$

holds we can rewrite OCP (7.5) in the form

$$\begin{aligned}
\min_{t_\mathrm{s}, t_\mathrm{f}, \boldsymbol{x}, \boldsymbol{u}} \quad & \varphi(t_\mathrm{s}, \boldsymbol{x}^{(1)}(-1), t_\mathrm{f}, \boldsymbol{x}^{(N)}(+1)) + \frac{h}{2} \sum_{n=1}^{N} \int_{\mathcal{I}_n} \psi(t, \boldsymbol{x}^{(n)}(t), \boldsymbol{u}^{(n)}(t); t_\mathrm{s}, t_\mathrm{f}) \, \mathrm{d}t \qquad (7.6)\\
\text{s.t.} \quad & \dot{\boldsymbol{x}}^{(n)}(t) = \frac{h}{2} \cdot \boldsymbol{f}(t, \boldsymbol{x}^{(n)}(t), \boldsymbol{u}^{(n)}(t); t_\mathrm{s}, t_\mathrm{f}), \quad n \in [N],\ t \in \mathcal{I}_n,\\
& \mathbf{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(t, \boldsymbol{x}^{(n)}(t), \boldsymbol{u}^{(n)}(t); t_\mathrm{s}, t_\mathrm{f}), \qquad n \in [N],\ t \in \mathcal{I}_n,\\
& \mathbf{0}_{n_\mathrm{r}} = \boldsymbol{r}(t_\mathrm{s}, \boldsymbol{x}^{(1)}(-1), t_\mathrm{f}, \boldsymbol{x}^{(N)}(+1)),\\
& \mathbf{0}_{n_\mathrm{x}} = \boldsymbol{x}^{(n+1)}(t_n) - \boldsymbol{x}^{(n)}(t_n), \qquad n \in [N-1],
\end{aligned}$$

where we put additional matching conditions to ensure continuous trajectories over the full time horizon. In a next step, we transform the single intervals $\mathcal{I}_n$ to the unit interval $[-1, +1]$. Similarly to (7.2) this can be achieved by means of linear time transformations $\boldsymbol{t}_n : [-1, +1] \longrightarrow \mathcal{I}_n$ which are defined as

$$\boldsymbol{t}_n(\tau) \stackrel{\mathrm{def}}{=} \frac{t_n + t_{n-1}}{2} + \tau \cdot \frac{t_n - t_{n-1}}{2}, \qquad 1 \leqslant n \leqslant N.$$

As opposed to $\boldsymbol{t}(\cdot)$, we do not write t_{n-1} or t_n as function arguments in $\boldsymbol{t}_n(\cdot)$ since we assume the temporal grid $\{t_n\}$ to be fixed. This is due to the fact that our approach to solve OCPs numerically is based on the idea to refine the discretization grid adaptively after solving the NLP. This assorts well with the tailored function spaces $\mathcal{Y}^k(\mathcal{T}, \mathbb{R})$ (see Section 2.4.5). They play a central role in Chapter 8. Note that other approaches use the finite element locations as additional optimization variables.

Now we use the linear mappings $\boldsymbol{t}_n(\cdot)$ to transform the domains of all functions $\boldsymbol{x}^{(n)}(\cdot)$ and $\boldsymbol{u}^{(n)}(\cdot)$ to the unitary domain $[-1, +1]$: their reparametrizations are given by the functions

$\widetilde{\boldsymbol{x}}^{(n)} : [-1,+1] \longrightarrow \mathbb{R}^{n_x}$ and $\widetilde{\boldsymbol{u}}^{(n)} : [-1,+1] \longrightarrow \mathbb{R}^{n_u}$ where $\widetilde{\boldsymbol{x}}^{(n)}(\tau) \stackrel{\text{def}}{=} \boldsymbol{x}^{(n)}(\boldsymbol{t}_n(\tau))$ and $\widetilde{\boldsymbol{u}}^{(n)}(\tau) \stackrel{\text{def}}{=} \boldsymbol{u}^{(n)}(\boldsymbol{t}_n(\tau))$. Furthermore, we employ the notation

$$\boldsymbol{f}_n(\tau, \boldsymbol{x}(\tau), \boldsymbol{u}(\tau); t_s, t_f) \stackrel{\text{def}}{=} \boldsymbol{f}(\boldsymbol{t}_n(\tau), \boldsymbol{x}(\tau), \boldsymbol{u}(\tau); t_s, t_f),$$

and in a similar fashion notations for the functions $\boldsymbol{\psi}(\cdot)$ and $\boldsymbol{c}(\cdot)$. As a result we can rewrite OCP (7.6) as follows: minimize the objective functional

$$\boldsymbol{\varphi}(t_s, \widetilde{\boldsymbol{x}}^{(1)}(-1), t_f, \widetilde{\boldsymbol{x}}^{(N)}(+1)) + \frac{h}{2} \sum_{n=1}^{N} \frac{t_n - t_{n-1}}{2} \int_{-1}^{+1} \boldsymbol{\psi}_n(\tau, \widetilde{\boldsymbol{x}}^{(n)}(\tau), \widetilde{\boldsymbol{u}}^{(n)}(\tau); t_s, t_f)\, \mathrm{d}\tau,$$

subject to the system dynamic constraints

$$\frac{\mathrm{d}}{\mathrm{d}\tau} \widetilde{\boldsymbol{x}}^{(n)}(\tau) = \frac{h}{2} \frac{t_n - t_{n-1}}{2} \cdot \boldsymbol{f}_n(\tau, \widetilde{\boldsymbol{x}}^{(n)}(\tau), \widetilde{\boldsymbol{u}}^{(n)}(\tau); t_s, t_f), \qquad n \in [N],$$

the mixed control–state constraints

$$\boldsymbol{0}_{n_c} \geqslant \boldsymbol{c}_n(\tau, \widetilde{\boldsymbol{x}}^{(n)}(\tau), \widetilde{\boldsymbol{u}}^{(n)}(\tau); t_s, t_f), \qquad n \in [N],$$

and boundary as well as matching conditions

$$\boldsymbol{0}_{n_r} = \boldsymbol{r}(t_s, \widetilde{\boldsymbol{x}}^{(1)}(-1), t_f, \widetilde{\boldsymbol{x}}^{(N)}(+1)) \quad \text{and} \quad \boldsymbol{0}_{n_x} = \widetilde{\boldsymbol{x}}^{(n+1)}(-1) - \widetilde{\boldsymbol{x}}^{(n)}(+1).$$

Without using the tilde notation, and by introducing the notation $h_n = t_n - t_{n-1}$ we can rewrite the transformed OCP in a compact form as

$$\min_{t_s, t_f, \boldsymbol{x}, \boldsymbol{u}} \quad \boldsymbol{\varphi}(t_s, \boldsymbol{x}^{(1)}(-1), t_f, \boldsymbol{x}^{(N)}(+1)) + \frac{h}{2} \sum_{n=1}^{N} \frac{h_n}{2} \int_{-1}^{+1} \boldsymbol{\psi}_n(\tau, \boldsymbol{x}^{(n)}(\tau), \boldsymbol{u}^{(n)}(\tau); t_s, t_f)\, \mathrm{d}\tau \tag{7.7a}$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}}^{(n)}(\tau) = \frac{h}{2} \frac{h_n}{2} \cdot \boldsymbol{f}_n(\tau, \boldsymbol{x}^{(n)}(\tau), \boldsymbol{u}^{(n)}(\tau); t_s, t_f), \qquad 1 \leqslant n \leqslant N,\ \tau \in [-1,1], \tag{7.7b}$$

$$\boldsymbol{0}_{n_c} \geqslant \boldsymbol{c}_n(\tau, \boldsymbol{x}^{(n)}(\tau), \boldsymbol{u}^{(n)}(\tau); t_s, t_f), \qquad 1 \leqslant n \leqslant N,\ \tau \in [-1,1], \tag{7.7c}$$

$$\boldsymbol{0}_{n_r} = \boldsymbol{r}(t_s, \boldsymbol{x}^{(1)}(-1), t_f, \boldsymbol{x}^{(N)}(+1)), \tag{7.7d}$$

$$\boldsymbol{0}_{n_x} = \boldsymbol{x}^{(n+1)}(-1) - \boldsymbol{x}^{(n)}(+1), \qquad 1 \leqslant n \leqslant N-1.$$

7.2 Global Collocation

In this section we give details on one concrete realization of the global approach, which was presented in Section 6.3.2, and apply it to the transformed continuous OCP (7.5). More specifically, we introduce an orthogonal collocation method whose collocation points are based on *Flipped Legendre–Gauss–Radau (FLGR) quadrature points* (see Section B.3). For an OCP, state as well as control trajectories are approximated by means of a global interpolating polynomial basis. Pseudospectral methods differ mainly in their choice of *discretization points* as well as *collocation points*. Discretization points are used to discretize states and controls, and thus,

to characterize variables that are fed into the NLP. On the other hand, collocation points are points which are used to collocate the ODE, i.e., to guarantee that the system dynamics have been met.
Specifically for the *Flipped Radau Pseudospectral Method (FRPM)* we consider the set of collocation points, $\mathcal{K} = \{\tau_1, \tau_2, \ldots, \tau_K\}$, consisting of the K FLGR points which correspond to the roots of the polynomial $\boldsymbol{P}_K - \boldsymbol{P}_{K-1}$ (see Appendix B.3.2). These FLGR points lie on the half–open interval $(-1, +1]$ such that $\tau_1, \ldots, \tau_K$ are strictly increasing, and $\tau_K = +1$. Next, by appending the point $\tau_0 = -1$ to the set $\mathcal{K}$, we define the superset $\mathcal{N} = \{\tau_0\} \cup \mathcal{K}$ such that $\mathcal{N}$ contains $K+1$ points on the interval $[-1, +1]$. Concerning our pseudospectral method, $\mathcal{N}$ defines the set of discretization points.

Direct Transcription Formulation

Let us suppose that $\boldsymbol{x} : [-1, +1] \longrightarrow \mathbb{R}^{n_x}$ denotes the state trajectory arising from our transformed continuous OCP (7.5). Then we can form a (polynomial) approximation, $\boldsymbol{X}(\cdot)$, by means of a basis of $|\mathcal{N}| = K+1$ Lagrange interpolating polynomials $\boldsymbol{L}_i(\cdot)$, $0 \leqslant i \leqslant K$:

$$\boldsymbol{x}(\tau) \simeq \boldsymbol{X}(\tau) = \sum_{i=0}^{K} \boldsymbol{X}(\tau_i)\, \boldsymbol{L}_i(\tau), \tag{7.8}$$

where

$$\boldsymbol{L}_i(\tau) = \prod_{\substack{j=0 \\ j \neq i}}^{K} \frac{\tau - \tau_j}{\tau_i - \tau_j}. \tag{7.9}$$

We end up with a state approximation which is interpolatory at all points within $\mathcal{N}$, i.e., it holds $\boldsymbol{x}(\tau_i) = \boldsymbol{X}(\tau_i)$, $0 \leqslant i \leqslant K$. Our pseudospectral method requires the system dynamics (7.5b) to be fulfilled exactly at the FLGR quadrature points. We approximate the left–hand side of the system dynamics by the derivative of the state approximation in (7.8), i.e., for $\tau_j \in \mathcal{K}$, $j \in [K]$ we obtain

$$\dot{\boldsymbol{x}}(\tau_j) \simeq \dot{\boldsymbol{X}}(\tau_j) = \sum_{i=0}^{K} \boldsymbol{X}(\tau_i)\, \dot{\boldsymbol{L}}_i(\tau_j) = \sum_{i=0}^{K} \boldsymbol{X}(\tau_i)\, D_{j,i}. \tag{7.10}$$

Note that the *FLGR differentiation matrix*, $D = [D_{j,i}] \in \mathbb{R}^{K \times (K+1)}$ is given as

$$D_{j,i} = \dot{\boldsymbol{L}}_i(\tau_j) = \begin{cases} \dfrac{\dot{\boldsymbol{L}}(\tau_j)}{\dot{\boldsymbol{L}}(\tau_i) \cdot (\tau_j - \tau_i)}, & \text{if } i \neq j, \\ \dfrac{\ddot{\boldsymbol{L}}(\tau_i)}{2\,\dot{\boldsymbol{L}}(\tau_i)}, & \text{if } i = j, \end{cases} \tag{7.11}$$

where $\boldsymbol{L}(\tau) = (1+\tau) \cdot [\boldsymbol{P}_K(\tau) - \boldsymbol{P}_{K-1}(\tau)]$ and the $\tau_i \in \mathcal{N}$ are the discretization points (see Appendix B.2). Since collocation equations for system dynamics and path constraints involve

the control solely at the set $\mathcal{K}$ of collocation points (control at starting point is simply extrapolated), we approximate the control trajectory $\boldsymbol{u} : [-1,+1] \longrightarrow \mathbb{R}^{n_u}$ with the aid of K Lagrange interpolating polynomials, $\overline{\boldsymbol{L}}_i(\cdot)$, $i \in [K]$, as

$$\boldsymbol{u}(\tau) \simeq \boldsymbol{U}(\tau) = \sum_{i=1}^{K} \boldsymbol{U}(\tau_i)\overline{\boldsymbol{L}}_i(\tau), \tag{7.12}$$

where

$$\overline{\boldsymbol{L}}_i(\tau) = \prod_{\substack{j=1 \\ j\neq i}}^{K} \frac{\tau-\tau_j}{\tau_i-\tau_j}. \tag{7.13}$$

By means of Gauss quadrature (see Appendix B.3.2) with FLGR quadrature nodes $\{\tau_i\}$ and associated weights $\{\omega_i\}$ the objective functional (7.5a) is then transcribed to

$$\boldsymbol{\varphi}(t_s, \boldsymbol{X}(\tau_0), t_f, \boldsymbol{X}(\tau_K)) + \frac{t_f - t_s}{2} \sum_{i=1}^{K} \omega_i \cdot \boldsymbol{\psi}(\tau_i, \boldsymbol{X}(\tau_i), \boldsymbol{U}(\tau_i); t_s, t_f),$$

the system dynamics (7.5b) to

$$\boldsymbol{0}_{n_x} = \sum_{i=0}^{K} \boldsymbol{X}(\tau_i)\, D_{j,i} - \frac{t_f - t_s}{2} \cdot \boldsymbol{f}(\tau_j, \boldsymbol{X}(\tau_j), \boldsymbol{U}(\tau_j); t_s, t_f), \qquad j \in [K],$$

the mixed control–state path constraints (7.5c) to

$$\boldsymbol{0}_{n_c} \geqslant \boldsymbol{c}(\tau_j, \boldsymbol{X}(\tau_j), \boldsymbol{U}(\tau_j); t_s, t_f), \qquad j \in [K],$$

and finally the boundary constraints (7.5d) to

$$\boldsymbol{0}_{n_r} = \boldsymbol{r}(t_s, \boldsymbol{X}(\tau_0), t_f, \boldsymbol{X}(\tau_K)).$$

FRPM Discretization of an OCP

Based on the previous discussion, we are now able to set up an NLP arising from a FRPM discretization of transformed continuous OCP (7.5). To this end, let us determine as NLP variables, the (scalar) variables t_s (start time), and t_f (final time), as well as the (vector–valued) discretized state variables, $x_j \in \mathbb{R}^{n_x}$, $0 \leqslant j \leqslant K$, and as discretized control variables, $u_j \in \mathbb{R}^{n_u}$, $1 \leqslant j \leqslant K$. Here we can bring together the approximations (7.8)+(7.12) and the x_j resp. the u_j by identifying

$$x_j \equiv \boldsymbol{X}(\tau_j), \quad 0 \leqslant j \leqslant K, \qquad u_j \equiv \boldsymbol{U}(\tau_j), \quad 1 \leqslant j \leqslant K. \tag{7.14}$$

By combining NLP variables corresponding to

(i) states, $x = [{x_0}^T, \ldots, {x_K}^T]^T \in \mathbb{R}^{(K+1)n_x}$,

(ii) controls, $u = [{u_1}^T, \ldots, {u_K}^T]^T \in \mathbb{R}^{K n_\mathrm{u}}$

we define the overall NLP variable $w = [x^T, u^T, t_\mathrm{s}, t_\mathrm{f}]^T \in \mathbb{R}^{n_\mathrm{w}}$ with $n_\mathrm{w} = (K+1)\, n_\mathrm{x} + K n_\mathrm{u} + 2$. The NLP objective function $\boldsymbol{\Phi} : \mathbb{R}^{n_\mathrm{w}} \longrightarrow \mathbb{R}$ is written as

$$\boldsymbol{\Phi}(w) = \boldsymbol{\varphi}(t_\mathrm{s}, x_0, t_\mathrm{f}, x_K) + \frac{t_f - t_s}{2} \sum_{i=1}^{K} \omega_i \cdot \boldsymbol{\psi}(\tau_i, x_i, u_i; t_\mathrm{s}, t_\mathrm{f}), \tag{7.15}$$

the NLP constraint functions $\boldsymbol{F}_j : \mathbb{R}^{n_\mathrm{w}} \longrightarrow \mathbb{R}^{n_\mathrm{x}}$ corresponding to the system dynamics (7.5b) as

$$\boldsymbol{F}_j(w) = \sum_{i=0}^{K} x_i\, D_{j,i} - \frac{t_f - t_s}{2} \cdot \boldsymbol{f}(\tau_j, x_j, u_j; t_\mathrm{s}, t_\mathrm{f}), \qquad j \in [K], \tag{7.16}$$

the NLP constraint functions $\boldsymbol{C}_j : \mathbb{R}^{n_\mathrm{w}} \longrightarrow \mathbb{R}^{n_\mathrm{c}}$ corresponding to the path constraints (7.5c) as

$$\boldsymbol{C}_j(w) = \boldsymbol{c}(\tau_j, x_j, u_j; t_\mathrm{s}, t_\mathrm{f}), \qquad j \in [K], \tag{7.17}$$

and finally the NLP constraint function $\boldsymbol{R} : \mathbb{R}^{n_\mathrm{w}} \longrightarrow \mathbb{R}^{n_\mathrm{r}}$ corresponding to the boundary condition (7.5d) as

$$\boldsymbol{R}(w) = \boldsymbol{r}(t_\mathrm{s}, x_0, t_\mathrm{f}, x_K). \tag{7.18}$$

Similarly to the NLP variables we combine the NLP constraint functions corresponding to

(i) system dynamics, $\boldsymbol{F}(w) = [\boldsymbol{F}_1(w)^T, \ldots, \boldsymbol{F}_K(w)^T]^T \in \mathbb{R}^{K n_\mathrm{x}}$,

(ii) path constraints, $\boldsymbol{C}(w) = [\boldsymbol{C}_1(w)^T, \ldots, \boldsymbol{C}_K(w)^T]^T \in \mathbb{R}^{K n_\mathrm{c}}$, and

(iii) boundary conditions, $\boldsymbol{R}(w) \in \mathbb{R}^{n_\mathrm{r}}$,

and introduce an overall NLP equality constraint function $\boldsymbol{G} : \mathbb{R}^{n_\mathrm{w}} \longrightarrow \mathbb{R}^{n_\mathrm{G}}$, $n_\mathrm{G} = K n_\mathrm{x} + n_\mathrm{r}$, as well as an overall NLP inequality constraint function $\boldsymbol{H} : \mathbb{R}^{n_\mathrm{w}} \longrightarrow \mathbb{R}^{n_\mathrm{H}}$, $n_\mathrm{H} = K n_\mathrm{c}$, where

$$\boldsymbol{G}(w) = [\boldsymbol{F}(w)^T, \boldsymbol{R}(w)^T]^T, \qquad \boldsymbol{H}(w) = \boldsymbol{C}(w).$$

We finally obtain an NLP of the following form:

$$\begin{aligned} \min_{w} \quad & \boldsymbol{\Phi}(w) \\ \text{s.t.} \quad & \mathbf{0} = \boldsymbol{G}(w), \\ & \mathbf{0} \geqslant \boldsymbol{H}(w). \end{aligned} \tag{7.19}$$

Implementation

In order to solve OCP instances numerically by means of the FRPM discretization approach, one needs a reliable and efficient implementation. Our implementation employs optionally the interior–point solver `Ipopt` [444] or the SQP solver SNOPT [197] to solve the FRPM

NLP (7.19). There are several criteria affecting the running time of those software packages: first, the arising data structures should be used in a cache efficient way. In particular, the arrays holding the NLP variables w, the NLP constraint function evaluation $[\boldsymbol{G}(w)^T, \boldsymbol{H}(w)^T]^T$, and the NLP constraint Jacobian evaluation $[\nabla \boldsymbol{G}(w), \nabla \boldsymbol{H}(w)]^T$ should be ordered such that they are accessed block–wise. Secondly, the sparse structure of the NLP constraint Jacobian and the Lagrangian of the Hessian should be exploited. Finally, the NLP has to be well scaled. We deal with those three aspects in the following. As a consequence thereof, we are enabled to point out the differences between the single–degree collocation approach and the multi–degree collocation approach described in the subsequent Section 7.3.

NLP Variable and Constraint Arrays An investigation of the NLP objective and constraint functions (7.15), (7.16), and (7.17) shows that $\boldsymbol{\psi}(\cdot)$, $\boldsymbol{f}(\cdot)$, and $\boldsymbol{c}(\cdot)$ are evaluated at points (τ_i, x_i, u_i) where the vectors x_i and u_i are state and control approximations evaluated at their respective collocation point τ_i. This nice representation of the approximation evaluations is due to their nodal representation (see (7.8), (7.12), and (7.14)). Hence,

$$x_i = [x_{1,i}, \ldots, x_{n_\mathrm{x},i}]^T \quad \text{and} \quad u_j = [u_{1,j}, \ldots, u_{n_\mathrm{x},j}]^T$$

with $0 \leqslant i \leqslant K$ and $1 \leqslant j \leqslant K$ should be placed at one stretch in the variable array and we identify the NLP variable vector w and the array holding the actual values of a NLP solver iteration. Likewise, the equality and inequality constraint evaluation arrays are assembled in accordance with $\boldsymbol{G}(\cdot)$ and $\boldsymbol{H}(\cdot)$. We use the notations

$$F_i = [F_{1,i}, \ldots, F_{n_\mathrm{x},i}]^T \quad \text{and} \quad C_i = [C_{1,i}, \ldots, C_{n_\mathrm{x},i}]^T, \qquad 1 \leqslant i \leqslant K,$$

where F_i denotes the vector $\boldsymbol{F}_i(w)$, i.e., the function $\boldsymbol{F}(\cdot)$ evaluated at the actual NLP solver iterate w. Likewise, C_i denotes the vector $\boldsymbol{C}_i(w)$. The arrays F, C, R, G, and H are composed in a canonical way.

NLP Constraint Jacobian As we will see, the NLP constraint Jacobian is rather sparse. Sparse Jacobians can be exploited by modern NLP solvers. We investigate the sparsity pattern by reference to a handy example which can be easily extended to the general case.

Example 7.1
We consider an ODE system with $n_\mathrm{x} = 2$ differential states and $n_\mathrm{u} = 1$ control. Assuming the case with two collocation points, we obtain the NLP variable array

$$w = [x_{1,0}, x_{2,0}, x_{1,1}, x_{2,1}, x_{1,2}, x_{2,2}, u_{1,1}, u_{1,2}]^T$$

and the NLP constraint array

$$F = [F_{1,1}, F_{2,1}, F_{1,2}, F_{2,2}]^T.$$

If we furthermore assume that we have an autonomous system with fixed time horizon $[-1, +1]$ then

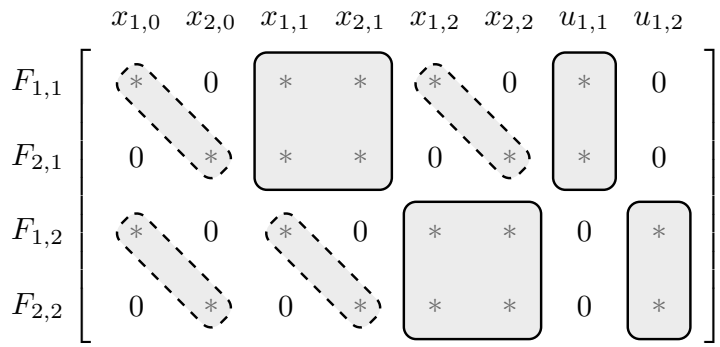


Figure 7.1: The figure depicts the NLP constraint Jacobian sparsity pattern of the system considered in Example 7.1. Blocks being independent of the right–hand–side function $\boldsymbol{f}(\cdot)$ are surrounded by dashed lines, dependent blocks by solid lines.

the entries $F_{k,l}$ are calculated as

$$F_{k,l} = \sum_{i=0}^{2} x_{k,i}\, D_{l,i} - \boldsymbol{f}(x_l, u_l).$$

The sparsity pattern of the associated NLP constraint Jacobian is depicted in Figure 7.1. Note that the diagonal blocks, surrounded by dashed lines, do not depend on the ODE system function $\boldsymbol{f}(\cdot)$, but only on differentiation matrix entries $D_{l,i}$. For instance, the entries $(F_{1,1}, x_{1,0})$ and $(F_{2,1}, x_{2,0})$ are given by $D_{1,0}$. Partial derivatives $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial u}$ enter the blocks surrounded by solid lines. As an example, we take the blocks

$$\begin{bmatrix} (F_{1,1}, x_{1,1}) & (F_{1,1}, x_{2,1}) \\ (F_{2,1}, x_{1,1}) & (F_{2,1}, x_{2,1}) \end{bmatrix} \hookleftarrow \frac{\partial}{\partial x}\boldsymbol{f}(x_1, u_1), \quad \begin{bmatrix} (F_{1,1}, u_{1,1}) \\ (F_{2,1}, u_{1,1}) \end{bmatrix} \hookleftarrow \frac{\partial}{\partial u}\boldsymbol{f}(x_1, u_1).$$

The array holding the non–zero entries should be ordered accordingly.

Similarly to the NLP constraint Jacobian, the NLP Lagrangian of the Hessian exhibits a specific sparse structure that is exploited in our implementation. Further details are beyond the scope of this thesis.

NLP Scaling Poor scaling may have tremendous effects on convergence rate, termination criteria, and numerical conditioning. For instance, variables representing a product concentration range in the interval $[0, 1]$, while variables that measure distances between cities may range from 0 to 10^6 meters. Hence, an aspect of scaling comprises making the variable ranges uniform. As an example, we consider a variable $z \in [a, b]$ and apply a variable scaling such that the range of the scaled variable $\widetilde{z}$ is $[0, 1]$. The scaling is realized by the affine transformation

$$\widetilde{z} = v_z\, z + r_z,$$

where the *variable scale* v_z and the *variable shift* r_z are defined as

$$v_z = \frac{1}{b-a} \qquad \text{and} \qquad r_z = -\frac{a}{b-a}.$$

The generalization to our state and control vectors x and u is straightforward and given as

$$\begin{bmatrix} \widetilde{x} \\ \widetilde{u} \end{bmatrix} = \begin{bmatrix} V_x & \mathbf{0} \\ \mathbf{0} & V_u \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} + \begin{bmatrix} r_x \\ r_u \end{bmatrix},$$

where the diagonal matrices V_x and V_u contain appropriately chosen state and control variable scale weights. The associated shift vectors are given by the vectors r_x and r_u. Analogously, one can scale the NLP constraints. Considering the case without path constraints C and boundary constraints R, one obtains the scaled defect constraints $\widetilde{F} = W_F F$, where the diagonal matrix W_F consists of suitable ODE constraint scale weights. If the associated Jacobian is denoted with J then its scaled counterpart is given by

$$\widetilde{J} = W_F J \begin{bmatrix} V_x & \mathbf{0} \\ \mathbf{0} & V_u \end{bmatrix}^{-1}.$$

To set $W_F = V_x$ is a common choice. In general, there are several other approaches to end up with a well–scaled problem. One can normalize the Jacobian rows and columns to be of the same magnitude. Another approach is based on the idea to make the condition number of the KKT matrix close to one by an appropriately chosen objective function scale weight. Here, a guess for the condition number can be obtained by means of Gerschgorin estimates for the smallest and largest eigenvalues of the Hessian.
The ideas presented in this section and realized in our implementation are by no means new but can be found e.g. in the textbooks of Gill et al. [196] and Betts [62]. They are also realized in other OCP software packages such as SOCS [64] and GPOPS-II [351]. For the sake of completeness and to stress its importance, scaling is also mentioned in this thesis.
Note that, however, there is probably no one–size–fits–all approach to NLP scaling. Moreover, even if there exists a good scaling at a certain point the same scaling might be poor at other points.

NLP Function and Derivative Generation System functions for objective, ODE, and constraints are set up within the software framework SolvIND [9] that provides an interface to the automatic differentiation tool ADOL-C [445]. The foundations of automatic differentiation are briefly discussed in Section 6.1.1.

7.3 Multi–Degree Global Collocation

In this section, we provide our extension to the standard global collocation approach that was introduced in the previous Section 7.2. We call our approach *multi–degree global collocation* since single state and control approximation components may differ in terms of their polynomial degree. We restrict our analysis to the case of systems without constraints since an

extension is straightforward.

Direct Transcription Formulation

Now, state and control approximations are defined as

$$\boldsymbol{X}(\tau) = [\boldsymbol{X}_1(\tau), \ldots, \boldsymbol{X}_{n_\mathrm{x}}(\tau)]^T \quad \text{and} \quad \boldsymbol{U}(\tau) = [\boldsymbol{U}_1(\tau), \ldots, \boldsymbol{U}_{n_\mathrm{u}}(\tau)]^T,$$

where

$$\boldsymbol{X}_i(\tau) = \sum_{j=0}^{K_i^x} x_{i,j}\, \boldsymbol{L}_{i,j}(\tau), \quad i \in [n_\mathrm{x}], \quad \text{and} \quad \boldsymbol{U}_i(\tau) = \sum_{j=1}^{K_i^u} u_{i,j}\, \overline{\boldsymbol{L}}_{i,j}(\tau), \quad i \in [n_\mathrm{u}]. \tag{7.20}$$

The Lagrange interpolating polynomials are

$$\boldsymbol{L}_{i,j}(\tau) = \prod_{\substack{k=0 \\ k \neq j}}^{K_i^x} \frac{\tau - \tau_{i,k}^x}{\tau_{i,j}^x - \tau_{i,k}^x} \qquad \text{and} \qquad \overline{\boldsymbol{L}}_{i,j}(\tau) = \prod_{\substack{k=1 \\ k \neq j}}^{K_i^u} \frac{\tau - \tau_{i,k}^u}{\tau_{i,j}^u - \tau_{i,k}^u}.$$

Analogously to Section 7.2, the $\tau_{i,j}^x$ and the $\tau_{i,j}^u$ are chosen such that the sets

$$\mathcal{K}_i^x = \left\{\tau_{i,1}^x, \tau_{i,2}^x, \ldots, \tau_{i,K_i^x}^x\right\} \quad \text{and} \quad \mathcal{K}_i^u = \left\{\tau_{i,1}^u, \tau_{i,2}^u, \ldots, \tau_{i,K_i^x}^u\right\}$$

consist of FLGR points. The $\tau_{i,0}^x$ are equal to -1. For the i–th component of the ODE we choose the set of collocation points, i.e., the time instants where the respective ODE component must hold exactly for state and control approximations, to be identical with $\mathcal{K}_i^x$. We denote evaluation point sets for objective, ODE, and path constraints with $\mathcal{K}^\psi$, $\mathcal{K}_i^f$, and $\mathcal{K}_i^c$, respectively. According to our choice, it holds that $\mathcal{K}_i^x = \mathcal{K}_i^f$. We use the canonic notations $K_i^x \overset{\text{def}}{=} \left|\mathcal{K}_i^x\right|$ and likewise for u, ψ, f, and c. With ω_i^ψ we denote the respective quadrature weight.

The NLP Formulation

While the direct transcription formulation stays unchanged compared to the one from Section 7.2, this does not hold for the NLP formulation derived from the equations (7.15)–(7.18). This is due to the fact that (7.14) cannot be transferred to the new setting.
Now, the overall NLP variable $w = [x^T, u^T, t_\mathrm{s}, t_\mathrm{f}]^T \in \mathbb{R}^{n_\mathrm{w}}$ is comprised of

(i) state variable, $x = [x_{1,0}, \ldots, x_{1,K_1^x}, \cdots, x_{n_\mathrm{x},0}, \ldots, x_{n_\mathrm{x},K_{n_\mathrm{x}}^x}]^T$, and

(ii) control variable, $u = [u_{1,1}, \ldots, u_{1,K_1^u}, \cdots, u_{n_\mathrm{u},1}, \ldots, u_{n_\mathrm{u},K_{n_\mathrm{u}}^u}]^T$,

where $n_\mathrm{w} = \sum_{i=1}^{n_\mathrm{x}} \left(K_i^x + 1\right) + \sum_{i=1}^{n_\mathrm{u}} K_i^u$. We reuse the notation of Section 7.2 to assemble a NLP of the same type as NLP 7.19. The NLP objective function $\boldsymbol{\Phi} : \mathbb{R}^{n_\mathrm{w}} \longrightarrow \mathbb{R}$ is written as

$$\boldsymbol{\Phi}(w) = \boldsymbol{\varphi}(t_\mathrm{s}, \boldsymbol{X}(-1), t_\mathrm{f}, \boldsymbol{X}(+1)) + \frac{t_f - t_s}{2} \sum_{i=1}^{K^\psi} \omega_i^\psi \cdot \boldsymbol{\psi}\left(\tau_i^\psi, \boldsymbol{X}(\tau_i^\psi), \boldsymbol{U}(\tau_i^\psi); t_\mathrm{s}, t_\mathrm{f}\right),$$

the NLP constraint functions $\boldsymbol{F}_{i,j} : \mathbb{R}^{n_\mathrm{w}} \longrightarrow \mathbb{R}$ corresponding to the system dynamics (7.5b) as

$$\boldsymbol{F}_{i,j}(w) = \sum_{k=0}^{K_i^x} x_{i,k}\, \dot{\boldsymbol{L}}_{i,k}(\tau_{i,j}^x) - \frac{t_f - t_s}{2} \cdot \boldsymbol{f}\left(\tau_{i,j}^x, \boldsymbol{X}(\tau_{i,j}^x), \boldsymbol{U}(\tau_{i,j}^x); t_\mathrm{s}, t_\mathrm{f}\right),$$

with $i \in [n_\mathrm{x}]$ and $j \in \left[K_i^x\right]$, the NLP constraint functions $\boldsymbol{C}_{i,j} : \mathbb{R}^{n_\mathrm{w}} \longrightarrow \mathbb{R}$ corresponding to the path constraints (7.5c) as

$$\boldsymbol{C}_{i,j}(w) = \boldsymbol{c}\left(\tau_{i,j}^c, \boldsymbol{X}(\tau_{i,j}^c), \boldsymbol{U}(\tau_{i,j}^c); t_\mathrm{s}, t_\mathrm{f}\right), \qquad i \in [n_\mathrm{c}], \quad j \in \left[K_i^c\right],$$

and finally the NLP constraint function $\boldsymbol{R} : \mathbb{R}^{n_\mathrm{w}} \longrightarrow \mathbb{R}^{n_\mathrm{r}}$ corresponding to the boundary condition (7.5d) as

$$\boldsymbol{R}(w) = \boldsymbol{r}(t_\mathrm{s}, \boldsymbol{X}(-1), t_\mathrm{f}, \boldsymbol{X}(+1)).$$

Implementation

From a theoretical point of view, there do not arise any difficulties for the multi–degree collocation approach compared to the standard approach. However, there are some pitfalls when it comes to an efficient implementation. In order to avoid more notational clutter, we restrict our analysis to an example that catches important aspects and allows the reader to extend it to the general case.

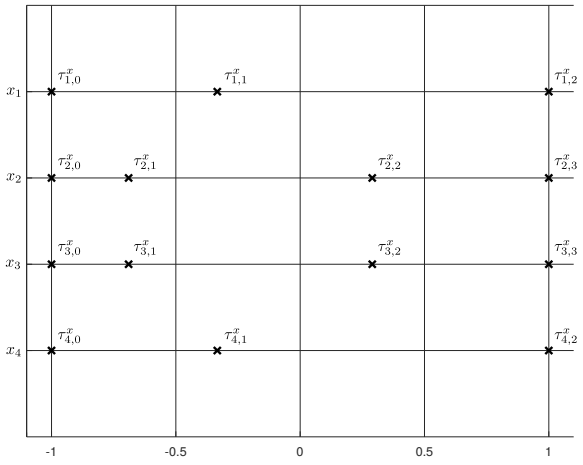


Figure 7.2: The figure depicts the discretization point distribution for the system considered in Example 7.2. States x_1 and x_4 have the same discretization points, namely two FLGR collocation points augmented with -1. States x_2 and x_3 have the same discretization points, namely three FLGR collocation points augmented with -1.

Example 7.2
We consider the problem with $n_x = 4$ differential states and the number of collocation points are given as $K_1^x = K_4^x = 2$ and $K_2^x = K_3^x = 3$. For an autonomous system on the fixed time horizon $[-1, +1]$ the NLP constraint functions are

$$\boldsymbol{F}_{i,j}(w) = \sum_{k=0}^{K_i^x} x_{i,k}\, \dot{\boldsymbol{L}}_{i,k}(\tau_{i,j}^x) - \boldsymbol{f}\left(\boldsymbol{X}(\tau_{i,j}^x)\right), \qquad i \in [4], \quad j \in \left[K_i^x\right].$$

NLP Variable and Constraint Arrays Writing the NLP variables of Example 7.2 as they are encoded in w yields the array

$$w = [x_{1,0}, x_{1,1}, x_{1,2}, x_{2,0}, x_{2,1}, x_{2,2}, x_{2,3}, x_{3,0}, x_{3,1}, x_{3,2}, x_{3,3}, x_{4,0}, x_{4,1}, x_{4,2}]^T.$$

The NLP constraint array is assembled as

$$F = [F_{1,1}, F_{1,2}, F_{2,1}, F_{2,2}, F_{2,3}, F_{3,1}, F_{3,2}, F_{3,3}, F_{4,1}, F_{4,2}]^T.$$

A naive implementation to evaluate the necessary values $\boldsymbol{X}(\tau_{i,j}^x)$ and therewith the values in array F would be inefficient. Instead, we increase the efficiency by an exploitation of similarities in components one and four as well as two and three. As one can easily see in Figure 7.2, the discretization grid of components x_1 and x_4 match. The same holds for components x_2 and x_3. We call the sets $\{x_1, x_4\}$ and $\{x_2, x_3\}$ *cliques* and denote their grids with $\tau_{i,j}$, i.e., it holds

$$\tau_{1,j} = \tau_{1,j}^x = \tau_{4,j}^x, \quad 0 \leqslant j \leqslant 2, \qquad \tau_{2,j} = \tau_{2,j}^x = \tau_{3,j}^x, \quad 0 \leqslant j \leqslant 3.$$

Based on the formulation of state approximation $\boldsymbol{X}(\cdot)$ in (7.20) we find

$$\begin{bmatrix} \boldsymbol{X}_1(\tau_{1,j}) \\ \boldsymbol{X}_4(\tau_{1,j}) \end{bmatrix} = \begin{bmatrix} x_{1,j} \\ x_{4,j} \end{bmatrix}, \quad 0 \leqslant j \leqslant 2, \qquad \begin{bmatrix} \boldsymbol{X}_2(\tau_{2,j}) \\ \boldsymbol{X}_3(\tau_{2,j}) \end{bmatrix} = \begin{bmatrix} x_{2,j} \\ x_{3,j} \end{bmatrix}, \quad 0 \leqslant j \leqslant 3.$$

In order to obtain the remainder of the state approximation evaluations we introduce the matrices

$$V_1 = \begin{bmatrix} \boldsymbol{L}_{1,0}(\tau_{2,1}) & \boldsymbol{L}_{1,0}(\tau_{2,2}) & \boldsymbol{L}_{1,0}(\tau_{2,3}) \\ \boldsymbol{L}_{1,1}(\tau_{2,1}) & \boldsymbol{L}_{1,1}(\tau_{2,2}) & \boldsymbol{L}_{1,1}(\tau_{2,3}) \\ \boldsymbol{L}_{1,2}(\tau_{2,1}) & \boldsymbol{L}_{1,2}(\tau_{2,2}) & \boldsymbol{L}_{1,2}(\tau_{2,3}) \end{bmatrix} \quad \text{and} \quad V_2 = \begin{bmatrix} \boldsymbol{L}_{2,0}(\tau_{1,1}) & \boldsymbol{L}_{2,0}(\tau_{1,2}) \\ \boldsymbol{L}_{2,1}(\tau_{1,1}) & \boldsymbol{L}_{2,1}(\tau_{1,2}) \\ \boldsymbol{L}_{2,2}(\tau_{1,1}) & \boldsymbol{L}_{2,2}(\tau_{1,2}) \\ \boldsymbol{L}_{2,3}(\tau_{1,1}) & \boldsymbol{L}_{2,3}(\tau_{1,2}) \end{bmatrix}.$$

Note that those matrices do not depend on the ODE system but only on the discretization grid. Hence, they can be calculated before the NLP is initiated. In each NLP solver iteration, one has to evaluate

$$\begin{bmatrix} \boldsymbol{X}_1(\tau_{1,0}) & \boldsymbol{X}_1(\tau_{1,1}) & \boldsymbol{X}_1(\tau_{1,2}) \\ \boldsymbol{X}_4(\tau_{1,0}) & \boldsymbol{X}_4(\tau_{1,1}) & \boldsymbol{X}_4(\tau_{1,2}) \end{bmatrix} = \begin{bmatrix} x_{1,0} & x_{1,1} & x_{1,2} \\ x_{4,0} & x_{4,1} & x_{4,2} \end{bmatrix} \stackrel{\text{def}}{=} X_1,$$

$$\begin{bmatrix} \boldsymbol{X}_1(\tau_{2,1}) & \boldsymbol{X}_1(\tau_{2,2}) & \boldsymbol{X}_1(\tau_{2,3}) \\ \boldsymbol{X}_4(\tau_{2,1}) & \boldsymbol{X}_4(\tau_{2,2}) & \boldsymbol{X}_4(\tau_{2,3}) \end{bmatrix} = \begin{bmatrix} x_{1,0} & x_{1,1} & x_{1,2} \\ x_{4,0} & x_{4,1} & x_{4,2} \end{bmatrix} V_1 = X_1 V_1$$

for clique one and

$$\begin{bmatrix} \boldsymbol{X}_2(\tau_{2,0}) & \boldsymbol{X}_2(\tau_{2,1}) & \boldsymbol{X}_2(\tau_{2,2}) & \boldsymbol{X}_2(\tau_{2,3}) \\ \boldsymbol{X}_3(\tau_{2,0}) & \boldsymbol{X}_3(\tau_{2,1}) & \boldsymbol{X}_3(\tau_{2,2}) & \boldsymbol{X}_3(\tau_{2,3}) \end{bmatrix} = \begin{bmatrix} x_{2,0} & x_{2,1} & x_{2,2} & x_{2,3} \\ x_{3,0} & x_{3,1} & x_{3,2} & x_{3,3} \end{bmatrix} \stackrel{\text{def}}{=} X_2,$$

$$\begin{bmatrix} \boldsymbol{X}_2(\tau_{1,1}) & \boldsymbol{X}_2(\tau_{1,2}) \\ \boldsymbol{X}_3(\tau_{1,1}) & \boldsymbol{X}_3(\tau_{1,2}) \end{bmatrix} = \begin{bmatrix} x_{2,0} & x_{2,1} & x_{2,2} & x_{2,3} \\ x_{3,0} & x_{3,1} & x_{3,2} & x_{3,3} \end{bmatrix} V_2 = X_2 V_2$$

for clique two. A fast access to the matrices X_1 and X_2 is mandatory. For this reason, their entries should be stored as blocks. This also saves unnecessary copy operations. Assuming a column–major order for storing multidimensional arrays we permute the NLP variable vector according to

$$w = [x_{1,0}, x_{4,0}, x_{1,1}, x_{4,1}, x_{1,2}, x_{4,2}, x_{2,0}, x_{3,0}, x_{2,1}, x_{3,1}, x_{2,2}, x_{3,2}, x_{2,3}, x_{3,3}]^T$$

and

$$F = [F_{1,1}, F_{4,1}, F_{1,2}, F_{4,2}, F_{2,1}, F_{3,1}, F_{2,2}, F_{3,2}, F_{2,3}, F_{3,3}]^T.$$

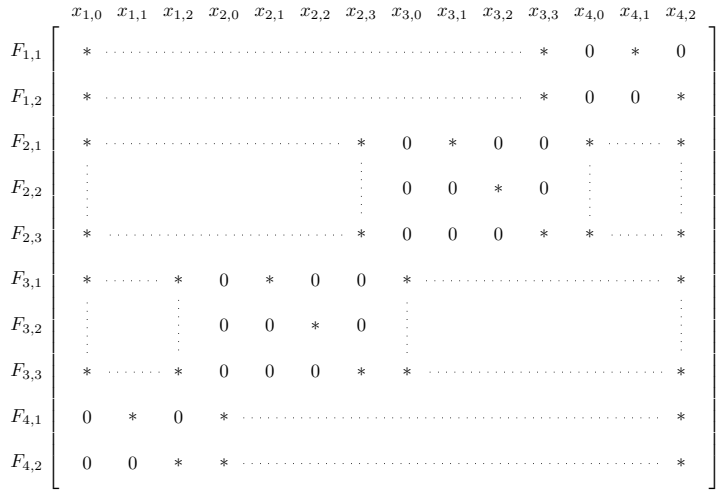


Figure 7.3: The figure depicts the NLP constraint Jacobian sparsity pattern of the system from Example 7.2 without a permutation of variables and constraints.

NLP Constraint Jacobian The arguments from the previous section also hold for the evaluations of the NLP constraint Jacobian and the Hessian. In Figure 7.3 one can see the structure of the constraint Jacobian without permutation of variables and constraints. In contrast, Figure 7.4 shows the constraint Jacobian after the permutation step. It is obvious that the structure

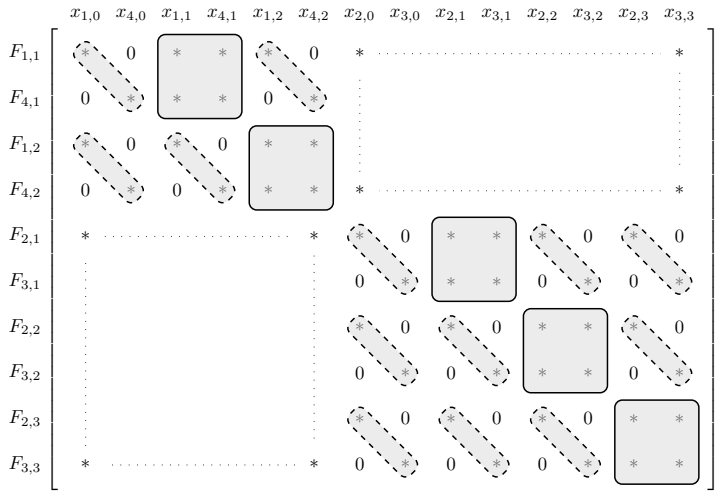


Figure 7.4: The figure depicts the NLP constraint Jacobian sparsity pattern of the system from Example 7.2. NLP variables and constraints are permuted. The upper–left and the lower–right block have the same structure as NLP constraint Jacobians of a standard global collocation discretization (see Figure 7.1).

of the uniform polynomial degree case (see Figure 7.1) is retained for the upper left and lower right matrix blocks. The upper right and lower left matrix blocks are dense. For this reason, the polynomial degrees have to be chosen carefully such that one does not end up with dense Jacobians or Hessians. However, as our numerical experiments in Chapter 12 show there is the potential for a significant speedup if applied carefully and under appropriate circumstances. For instance, one can think of an ODE where the differential states fall into two categories. The extremely volatile trajectories of the first category need a high degree polynomial to be sufficiently well approximated. In contrast, the trajectories of the second category are almost constant over the horizon and the approximating polynomial degree can be small. Then, the savings of variables and the constraints may exceed the not fully sparse NLP constraint Jacobians and Hessians.

7.4 Local Collocation

Next we describe a way how to employ a (global) pseudospectral method in order to end up with a local OCP discretization algorithm. For that purpose, we basically apply the global collocation approach from section 7.2 to all finite elements from OCP (7.7). This allows also for using different polynomial orders on distinct finite elements.

Direct Transcription Formulation Let $K^{(n)}$ denote the number of collocation points on the n–th finite element of transformed continuous OCP (7.7). Then the discretization grid $\mathcal{N}_n$,

which includes the collocation grid $\mathcal{K}_n$, has the form

$$-1 = \tau_0^{(n)} < \tau_1^{(n)} < \ldots < \tau_{K^{(n)}}^{(n)} = +1.$$

In order to define the local approximation polynomials for states and controls, we need Lagrange interpolating polynomials in a similar way to how we have done it in (7.9) and (7.13):

$$\boldsymbol{L}_i^{(n)}(\tau) \stackrel{\text{def}}{=} \prod_{\substack{j=0 \\ j\neq i}}^{K^{(n)}} \frac{\tau - \tau_j^{(n)}}{\tau_i^{(n)} - \tau_j^{(n)}}, \quad \deg\left(\boldsymbol{L}_i^{(n)}\right) = K^{(n)}, \qquad 1 \leqslant n \leqslant N,\ 0 \leqslant i \leqslant K^{(n)},$$

$$\overline{\boldsymbol{L}}_i^{(n)}(\tau) \stackrel{\text{def}}{=} \prod_{\substack{j=1 \\ j\neq i}}^{K^{(n)}} \frac{\tau - \tau_j^{(n)}}{\tau_i^{(n)} - \tau_j^{(n)}}, \quad \deg\left(\overline{\boldsymbol{L}}_i^{(n)}\right) = K^{(n)} - 1, \quad 1 \leqslant n \leqslant N,\ 1 \leqslant i \leqslant K^{(n)}.$$

Approximations of the functions $\boldsymbol{x}^{(n)} : [-1,+1] \longrightarrow \mathbb{R}^{n_\mathrm{x}}$ and $\boldsymbol{u}^{(n)} : [-1,+1] \longrightarrow \mathbb{R}^{n_\mathrm{u}}$ have the form

$$\boldsymbol{X}^{(n)}(\tau) \stackrel{\text{def}}{=} \sum_{i=0}^{K^{(n)}} \boldsymbol{X}^{(n)}(\tau_i)\, \boldsymbol{L}_i^{(n)}(\tau), \qquad 1 \leqslant n \leqslant N,\ \tau \in [-1,+1],$$

$$\boldsymbol{U}^{(n)}(\tau) \stackrel{\text{def}}{=} \sum_{i=1}^{K^{(n)}} \boldsymbol{U}^{(n)}(\tau_i)\, \overline{\boldsymbol{L}}_i^{(n)}(\tau), \qquad 1 \leqslant n \leqslant N,\ \tau \in [-1,+1].$$

The FLGR differentiation matrices $D^{(n)} = [D_{j,i}^{(n)}] \in \mathbb{R}^{K^{(n)} \times (K^{(n)}+1)}$ can be determined analogously to (7.11) such that the derivative of $\boldsymbol{X}^{(n)}(\cdot)$ evaluated at the collocation points can be calculated for $1 \leqslant n \leqslant N$ according to

$$\dot{\boldsymbol{x}}^{(n)}(\tau_j) \simeq \dot{\boldsymbol{X}}^{(n)}(\tau_j) = \sum_{i=0}^{K^{(n)}} \boldsymbol{X}^{(n)}(\tau_i)\, \dot{\boldsymbol{L}}_i^{(n)}(\tau_j) = \sum_{i=0}^{K^{(n)}} \boldsymbol{X}^{(n)}(\tau_i)\, D_{j,i}^{(n)}, \quad 1 \leqslant j \leqslant K^{(n)}.$$

We apply Gauss quadrature element–wise to the integrals occurring in OCP (7.7): for any $1 \leqslant n \leqslant N$ let $\{\tau_i^{(n)}\}$, $1 \leqslant i \leqslant K^{(n)}$, denote the set of FLGR quadrature nodes, and let the associated quadrature weights be denoted by $\{\omega_i^{(n)}\}$. We then transcribe the objective functional (7.7a) to

$$\varphi(t_\mathrm{s}, \boldsymbol{X}^{(1)}(-1), t_\mathrm{f}, \boldsymbol{X}^{(N)}(+1)) + \frac{h}{2} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)}\, \boldsymbol{\psi}_n(\tau_i^{(n)}, \boldsymbol{X}^{(n)}(\tau_i^{(n)}), \boldsymbol{U}^{(n)}(\tau_i^{(n)}); t_\mathrm{s}, t_\mathrm{f}),$$

the system dynamics (7.7b) for $1 \leqslant n \leqslant N$ and $1 \leqslant j \leqslant K^{(n)}$ to

$$\mathbf{0}_{n_\mathrm{x}} = \sum_{i=0}^{K^{(n)}} \boldsymbol{X}^{(n)}(\tau_i^{(n)})\, D_{j,i}^{(n)} - \frac{h}{2}\frac{h_n}{2} \cdot \boldsymbol{f}_n\left(\tau_j^{(n)}, \boldsymbol{X}^{(n)}(\tau_j^{(n)}), \boldsymbol{U}^{(n)}(\tau_j^{(n)}); t_\mathrm{s}, t_\mathrm{f}\right),$$

the mixed control–state path constraints (7.7c) to

$$\mathbf{0}_{n_c} \geqslant \boldsymbol{c}_n\left(\tau_j^{(n)}, \boldsymbol{X}^{(n)}(\tau_j^{(n)}), \boldsymbol{U}^{(n)}(\tau_j^{(n)}); t_s, t_f\right), \quad 1 \leqslant n \leqslant N,\ 1 \leqslant j \leqslant K^{(n)},$$

and the boundary constraints (7.7d) to

$$\mathbf{0}_{n_r} = \boldsymbol{r}(t_s, \boldsymbol{X}^{(1)}(-1), t_f, \boldsymbol{X}^{(N)}(+1)).$$

Unlike in Section 7.2 the state and control approximations are local polynomials now. State trajectories as solutions of ODEs are usually chosen to be in certain function spaces such that they are continuous over the full horizon. In order to take this into account we equip our previous discretization approach additionally with continuity conditions,

$$\mathbf{0}_{n_x} = \boldsymbol{X}^{(n+1)}(-1) - \boldsymbol{X}^{(n)}(+1), \qquad 1 \leqslant n \leqslant N-1, \tag{7.21}$$

a technique similar in fashion to the direct multiple shooting approach (see section 6.2.3).

FRPM Discretization of an OCP Now we determine the NLP arising from the previous discretization approach. We define scalar variables for the start time, t_s, and for the final time, t_f. Furthermore, we introduce vector–valued variables $x_j^{(n)} \in \mathbb{R}^{n_x}$ and $u_j^{(n)} \in \mathbb{R}^{n_u}$ representing approximate values of state and control trajectories evaluated at discretization and collocation points, respectively, i.e., for $n \in [N]$ we have

$$x_j^{(n)} \equiv \boldsymbol{X}^{(n)}(\tau_j^{(n)}), \quad 0 \leqslant j \leqslant K^{(n)}, \qquad u_j^{(n)} \equiv \boldsymbol{U}^{(n)}(\tau_j^{(n)}), \quad 1 \leqslant j \leqslant K^{(n)}.$$

We obtain the overall NLP variable $w = [x^T, u^T, t_s, t_f]^T \in \mathbb{R}^{n_w}$ by collecting NLP variables corresponding to

(i) states, $x = \left[x^{(n)^T}\right]_{1\leqslant n\leqslant N}^T$, $x^{(n)} = \left[x_0^{(n)^T}, \ldots, x_{K^{(n)}}^{(n)^T}\right]^T$, and

(ii) controls, $u = \left[u^{(n)^T}\right]_{1\leqslant n\leqslant N}^T$, $u^{(n)} = \left[u_1^{(n)^T}, \ldots, u_{K^{(n)}}^{(n)^T}\right]^T$.

The dimension of x and u are denoted by n_s and n_q with

$$n_s = \sum_{n=1}^{N} (K^{(n)} + 1)\, n_x, \qquad n_q = \sum_{n=1}^{N} K^{(n)}\, n_u$$

such that $n_w = n_s + n_q + 2$. The NLP objective function $\boldsymbol{\Phi} : \mathbb{R}^{n_w} \longrightarrow \mathbb{R}$ is written as

$$\boldsymbol{\Phi}(w) = \varphi(t_s, x_0^{(1)}, t_f, x_{K^{(N)}}^{(N)}) + \frac{h}{2} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)}\, \psi_n(\tau_i^{(n)}, x_i^{(n)}, u_i^{(n)}; t_s, t_f),$$

the NLP constraint functions $\boldsymbol{F}_j^{(n)} : \mathbb{R}^{n_\mathrm{w}} \longrightarrow \mathbb{R}^{n_\mathrm{x}}$ with $1 \leqslant n \leqslant N$ and $1 \leqslant j \leqslant K^{(n)}$ corresponding to the system dynamics (7.7b) as

$$\boldsymbol{F}_j^{(n)}(w) = \sum_{i=0}^{K^{(n)}} x_i^{(n)} D_{j,i}^{(n)} - \frac{h}{2}\frac{h_n}{2} \cdot \boldsymbol{f}_n\left(\tau_j^{(n)}, x_j^{(n)}, u_j^{(n)}; t_\mathrm{s}, t_\mathrm{f}\right),$$

the NLP constraint functions $\boldsymbol{C}_j^{(n)} : \mathbb{R}^{n_\mathrm{w}} \longrightarrow \mathbb{R}^{n_\mathrm{c}}$ corresponding to the path constraints (7.7c) as

$$\boldsymbol{C}_j^{(n)}(w) = \boldsymbol{c}_n\left(\tau_j^{(n)}, x_j^{(n)}, u_j^{(n)}; t_\mathrm{s}, t_\mathrm{f}\right), \quad 1 \leqslant n \leqslant N,\ 1 \leqslant j \leqslant K^{(n)},$$

and finally the NLP constraint function $\boldsymbol{R} : \mathbb{R}^{n_\mathrm{w}} \longrightarrow \mathbb{R}^{n_\mathrm{r}}$ corresponding to the boundary condition (7.7d) as

$$\boldsymbol{R}(w) = \boldsymbol{r}(t_\mathrm{s}, x_0^{(1)}, t_\mathrm{f}, x_{K^{(N)}}^{(N)}).$$

The matching conditions (7.21) impose the constraint functions $\boldsymbol{M}^{(n)} : \mathbb{R}^{n_\mathrm{w}} \longrightarrow \mathbb{R}^{n_\mathrm{x}}$, where

$$\boldsymbol{M}^{(n)}(w) = x_0^{(n+1)} - x_{K^{(n)}}^{(n)}, \qquad 1 \leqslant n \leqslant N-1.$$

Similarly to the NLP variables we combine the NLP constraint functions corresponding to

(i) system dynamics, $\boldsymbol{F}(w) = \left[\boldsymbol{F}^{(n)}(w)^T\right]_{1\leqslant n\leqslant N}^T$, $\boldsymbol{F}^{(n)}(w) = \left[\boldsymbol{F}_1^{(n)}(w)^T, \ldots, \boldsymbol{F}_{K^{(n)}}^{(n)}(w)^T\right]^T$,

(ii) path constraints, $\boldsymbol{C}(w) = \left[\boldsymbol{C}^{(n)}(w)^T\right]_{1\leqslant n\leqslant N}^T$, $\boldsymbol{C}^{(n)}(w) = \left[\boldsymbol{C}_1^{(n)}(w)^T, \ldots, \boldsymbol{C}_{K^{(n)}}^{(n)}(w)^T\right]^T$,

(iii) boundary conditions, $\boldsymbol{R}(w)$, and

(iv) matching conditions, $\boldsymbol{M}(w) = \left[\boldsymbol{M}^{(n)}(w)^T\right]_{1\leqslant n\leqslant N-1}^T$.

The dimensions of $\boldsymbol{F}(w)$, $\boldsymbol{C}(w)$, $\boldsymbol{R}(w)$, and $\boldsymbol{M}(w)$ are denoted by n_F, n_C, n_R, and n_M. It is easy to check that

$$n_\mathrm{F} = \sum_{n=1}^{N} K^{(n)} n_\mathrm{x}, \qquad n_\mathrm{C} = \sum_{n=1}^{N} K^{(n)} n_\mathrm{c}, \qquad n_\mathrm{R} = n_\mathrm{r}, \qquad n_\mathrm{M} = (N-1)\, n_\mathrm{x}.$$

By introducing an overall NLP equality constraint function $\boldsymbol{G} : \mathbb{R}^{n_\mathrm{w}} \longrightarrow \mathbb{R}^{n_\mathrm{G}}$ ($n_\mathrm{G} = n_\mathrm{F} + n_\mathrm{M} + n_\mathrm{R}$) as well as an overall NLP inequality constraint function $\boldsymbol{H} : \mathbb{R}^{n_\mathrm{w}} \longrightarrow \mathbb{R}^{n_\mathrm{H}}$ ($n_\mathrm{H} = n_\mathrm{C}$), where

$$\boldsymbol{G}(w) = [\boldsymbol{F}(w)^T, \boldsymbol{M}(w)^T, \boldsymbol{R}(w)^T]^T, \qquad \boldsymbol{H}(w) = \boldsymbol{C}(w),$$

we can write the NLP in the form of Problem (7.19). Since we make extensive use of the NLP later in Chapter 9, we formulate its full version here:

$$\min_{t_s,t_f,x,u} \quad \varphi(t_s, x_0^{(1)}, t_f, x_{K^{(N)}}^{(N)}) + \frac{h}{2}\sum_{n=1}^{N}\frac{h_n}{2}\sum_{i=1}^{K^{(n)}}\omega_i^{(n)}\psi_n(\tau_i^{(n)}, x_i^{(n)}, u_i^{(n)}; t_s, t_f) \tag{7.22}$$

$$\begin{aligned}
\text{s.t.} \quad & \mathbf{0}_{n_x} = \sum_{i=0}^{K^{(n)}} x_i^{(n)} D_{j,i}^{(n)} - \frac{h}{2}\frac{h_n}{2}\cdot \boldsymbol{f}_n\left(\tau_j^{(n)}, x_j^{(n)}, u_j^{(n)}; t_s, t_f\right), && n\in[N],\ j\in[K^{(n)}],\\
& \mathbf{0}_{n_c} \geqslant \boldsymbol{c}_n\left(\tau_j^{(n)}, x_j^{(n)}, u_j^{(n)}; t_s, t_f\right), && n\in[N],\ j\in[K^{(n)}],\\
& \mathbf{0}_{n_r} = \boldsymbol{r}(t_s, x_0^{(1)}, t_f, x_{K^{(N)}}^{(N)}),\\
& \mathbf{0}_{n_x} = x_0^{(n+1)} - x_{K^{(n)}}^{(n)}, && n\in[N-1].
\end{aligned}$$

Note that the local approach can be equipped with the multi–degree approach in a straightforward manner. Our software `grc` is based on the local multi–degree pseudospectral method. The `SolvIND` model equations for Problem (12.1) look as follows:

```
template<typename T>
svLong ffcn_sincos(TArgs_ffcn<T>& args, TDependency* depends)
{
  const T lam = args.p[0];

  args.rhs[0] = +args.xd[1];
  args.rhs[1] = -args.xd[0];
  args.rhs[2] = +lam*args.xd[3];
  args.rhs[3] = -lam*args.xd[2];

  return 0;
}
```

The `grc` mesh – including the number of FEs and possibly distinct number of collocation points per FE – is realized as follows:

```
function mesh = usr_init_mesh

mesh = [];
n_fe = 1;

% 1 - Finite element fraction
mesh.fe_frac = (1/n_fe) * ones(1,n_fe);

% 2 - Collocation points
mesh.n_cp.x = [30;30;60;60] * ones(1,n_fe);

% 3 - Initial guess
mesh.guess.x = [0;1;0;1] * ones(1,n_fe);
mesh.guess.p = 10;

end
```

Chapter 8

A Discrete Local Minimum Principle

We consider this chapter as a first step in extending the functional analytic framework for OCPs as it was introduced by Beigel [41]. In this work, the author investigates solutions of IVPs in ODEs (see IVP (1.2) in Section 1.1) as well as solutions of associated adjoint IVPs. In order to establish the framework Beigel embeds the problem class into the problem class of *CVPs*, i.e., she considers problems of the form

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \varphi(\boldsymbol{x}(t_\mathrm{f})) \\ \text{s.t.} \quad & \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t)), \quad t \in [t_\mathrm{s}, t_\mathrm{f}], \\ & \boldsymbol{x}(t_\mathrm{s}) = x_\mathrm{s}. \end{aligned} \tag{8.1}$$

It is obvious that the feasible set of CVP (8.1) consists of a single element, namely the unique solution of the nominal IVP (1.2). In Section 1.2 we discussed the theory of IVPs and presented the Picard–Lindelöf theorem (see Theorem 1.12) which guarantees that solutions are continuously differentiable in a classic environment. For this reason, CVP (8.1) was also investigated in the function space of continuously differentiable functions by Beigel [41]. By means of Riesz's representation theorem (see Theorem 2.84) the Lagrange multiplier could be identified to be an element of the space of normalized functions with bounded variation, cf. Section 2.4.8.

From a practical point of view focusing on continuously differentiable functions means a restriction in the sense that most numerical integrators give approximations to the solution of IVP (1.2) that are not continuously differentiable on the whole interval $[t_\mathrm{s}, t_\mathrm{f}]$ but rather continuous and piecewise continuously differentiable. Consequently, Beigel investigated solutions of CVP (8.1) also in the function space $\mathcal{Y}^1([t_\mathrm{s}, t_\mathrm{f}], \mathbb{R}^n)$, cf. Section 2.4.5. The duality pairing (see Section 2.5) of $\mathcal{NBV}([t_\mathrm{s}, t_\mathrm{f}], \mathbb{R})$ and $\mathcal{Y}([t_\mathrm{s}, t_\mathrm{f}], \mathbb{R})$ enabled Beigel to characterize the Lagrange multiplier in the new function space setting.

Even though Beigel used the derived functional analytic framework to analyze BDF methods and their adjoint IND schemes it is by no means restricted to this case but rather allows for analyzing integration methods that provide at least a continuous and piecewise continuously differentiable approximation to the solution of IVP (1.2).

In this chapter we extend Beigel's theory in two directions: on the one hand we treat OCPs involving controls instead of CVP (8.1) and on the other hand we augment the problem class with additional constraints such as boundary constraints and mixed control–state constraints. Moreover, we show that the Lagrange multipliers have a higher regularity than actually predicted by Beigel's theory.

In Section 5.3 we have presented a local minimum principle (see Theorem 5.7) in the classic function space setting $(\boldsymbol{x}, \boldsymbol{u}) \in W^{1,\infty}(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}}) \times L^\infty(\mathcal{T}, \mathbb{R}^{n_\mathrm{u}})$. It provided us with a characterization of the Lagrange multipliers. An elegant way to derive those results in a DAE setting was proposed by Gerdts [189] in his habilitation thesis. We adapt his ideas while focusing on function spaces that are of particular practical relevance, namely state functions $\boldsymbol{x} \in \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}})$ and control functions $\boldsymbol{u} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_\mathrm{u}})$. This enables us to derive a specific semi–discrete local minimum principle and to complement Beigel's work in the aforementioned sense.

In subsequent chapters we can make use of those results in the following sense: we analyze numerical OCP solution methods such as pseudospectral collocation methods that generate continuous and piecewise differentiable approximations of the solution. To put it more concretely, we establish the interpretation of a particular orthogonal pseudospectral discretization scheme as a Petrov–Galerkin discretization of the variational formulation of the local minimum principle equations. As a consequence thereof, we establish a *covector mapping theorem* that relates NLP multipliers and costates coming from the local minimum principle. Furthermore, we derive a novel global goal–oriented a posteriori *error estimation* approach based on the DWR methodology.

In Section 8.1 we specify the OCP considered in this chapter and rewrite the problem as an infinite dimensional optimization problem. Then we show that the image of the first order derivative of the equality constraint operator is closed. For this reason solution formulas for linear ODEs have to be exploited. By means of these auxiliary results we are able to apply the first–order necessary optimality conditions of Fritz John type.

When applying necessary optimality conditions the arising multipliers appear as elements of dual spaces. Section 8.2 deals with deducing explicit representations of the aforementioned Lagrange multipliers.

Finally, in Section 8.3 we derive a local minimum principle for the OCP that was introduced in Section 8.1.

8.1 Problem Formulation

In this chapter we investigate a special case of Problem (5.1). Let $\mathcal{T} \stackrel{\mathrm{def}}{=} [t_\mathrm{s}, t_\mathrm{f}] \subset \mathbb{R}$ be a compact non-empty time interval with $t_\mathrm{s} < t_\mathrm{f}$. Let

$$
\begin{aligned}
\boldsymbol{\varphi} &: \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{x}} \longrightarrow \mathbb{R}, \\
\boldsymbol{\psi} &: \mathcal{T} \times \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{u}} \longrightarrow \mathbb{R}, \\
\boldsymbol{f} &: \mathcal{T} \times \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{u}} \longrightarrow \mathbb{R}^{n_\mathrm{x}}, \\
\boldsymbol{c} &: \mathcal{T} \times \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{u}} \longrightarrow \mathbb{R}^{n_\mathrm{c}}, \\
\boldsymbol{r} &: \mathbb{R}^{n_\mathrm{x}} \times \mathbb{R}^{n_\mathrm{x}} \longrightarrow \mathbb{R}^{n_\mathrm{r}}
\end{aligned}
$$

be sufficiently smooth mappings (see later Assumption 8.1). The OCP under consideration is given as follows:

$$\begin{aligned} \min_{(\boldsymbol{x},\boldsymbol{u})\in Z} \quad & \varphi(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})) + \int_{t_\mathrm{s}}^{t_\mathrm{f}} \psi(t,\boldsymbol{x}(t),\boldsymbol{u}(t))\,dt \\ \text{s.t.} \quad & \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t,\boldsymbol{x}(t),\boldsymbol{u}(t)), \quad t\in\mathcal{T}, \\ & \mathbf{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(t,\boldsymbol{x}(t),\boldsymbol{u}(t)), \quad t\in\mathcal{T}, \\ & \mathbf{0}_{n_\mathrm{r}} = \boldsymbol{r}(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})). \end{aligned} \tag{8.2}$$

First, we choose function spaces for differential state variables $\boldsymbol{x}(\cdot)$ and control variables $\boldsymbol{u}(\cdot)$. To this end, we define $Z \stackrel{\text{def}}{=} \mathcal{Y}^1(\mathcal{T},\mathbb{R}^{n_\mathrm{x}}) \times \mathcal{Y}(\mathcal{T},\mathbb{R}^{n_\mathrm{u}})$. Next, it is our goal to apply the first order necessary optimality conditions of Theorem 3.7 to OCP (8.2). For this reason, we proceed in the same way as described in Section 5.2 and reformulate OCP (8.2) as an infinite dimensional optimization problem

$$\begin{aligned} \min_{(\boldsymbol{x},\boldsymbol{u})\in Z} \quad & J(\boldsymbol{x},\boldsymbol{u}) \\ \text{s.t.} \quad & G(\boldsymbol{x},\boldsymbol{u}) \in \mathcal{K}, \\ & H(\boldsymbol{x},\boldsymbol{u}) = \Theta_V, \end{aligned} \tag{8.3}$$

with convex cone

$$\mathcal{K} = \left\{ k\in\mathcal{Y}(\mathcal{T},\mathbb{R}^{n_\mathrm{c}}) \,:\, k(t) \geqslant \mathbf{0}_{n_\mathrm{c}} \right\}, \tag{8.4}$$

function spaces

$$\begin{aligned} Z &= \mathcal{Y}^1(\mathcal{T},\mathbb{R}^{n_\mathrm{x}}) \times \mathcal{Y}(\mathcal{T},\mathbb{R}^{n_\mathrm{u}}), \\ V &= \mathcal{Y}(\mathcal{T},\mathbb{R}^{n_\mathrm{x}}) \times \mathbb{R}^{n_\mathrm{r}}, \\ W &= \mathcal{Y}(\mathcal{T},\mathbb{R}^{n_\mathrm{c}}), \end{aligned}$$

and mappings $J : Z \longrightarrow \mathbb{R}$, $H : Z \longrightarrow V$, and $G : Z \longrightarrow W$, given as

$$\begin{aligned} J(\boldsymbol{x},\boldsymbol{u}) &= \varphi(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})) + \int_{t_\mathrm{s}}^{t_\mathrm{f}} \psi(t,\boldsymbol{x}(t),\boldsymbol{u}(t))\,dt, \\ H(\boldsymbol{x},\boldsymbol{u}) &= \begin{bmatrix} H_1(\boldsymbol{x},\boldsymbol{u}) \\ H_2(\boldsymbol{x},\boldsymbol{u}) \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}(\cdot,\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)) - \dot{\boldsymbol{x}}(\cdot) \\ -\boldsymbol{r}(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})) \end{bmatrix}, \\ G(\boldsymbol{x},\boldsymbol{u}) &= -\boldsymbol{c}(\cdot,\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)). \end{aligned} \tag{8.5}$$

In order to analyze differentiability properties of the mappings J, H and G we state the following smoothness conditions:

Assumption 8.1
Let the following smoothness conditions hold for the functions $\varphi, \psi, \boldsymbol{f}, \boldsymbol{c}, \boldsymbol{r}$:

1. φ and $\boldsymbol{r}$ are continuously differentiable with respect to all arguments.

2. Let $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}}) \in \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}}) \times \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_\mathrm{u}})$ be given and let M be a sufficiently large convex compact neighborhood of

$$\left\{(\hat{\boldsymbol{x}}(t), \hat{\boldsymbol{u}}(t)) \in \mathbb{R}^{n_\mathrm{x}+n_\mathrm{u}} \,:\, t \in \mathcal{T}\right\}.$$

   a) The mappings $t \mapsto \boldsymbol{\psi}(t, x, u)$ and

   $$t \mapsto \boldsymbol{f}(t, x, u), \qquad t \mapsto \boldsymbol{c}(t, x, u)$$

   are measurable for every $(x, u) \in M$.

   b) The mappings $(x, u) \mapsto \boldsymbol{\psi}(t, x, u)$ and

   $$(x, u) \mapsto \boldsymbol{f}(t, x, u), \qquad (x, u) \mapsto \boldsymbol{c}(t, x, u)$$

   are continuously differentiable in M uniformly for $t \in \mathcal{T}$.

   c) The derivatives

   $$\boldsymbol{\psi}'_{(x,u)}, \quad \boldsymbol{f}'_{(x,u)}, \quad \boldsymbol{c}'_{(x,u)}$$

   are bounded in $\mathcal{T} \times M$.

The following theorem establishes Fréchet–differentiability under Assumption 8.1 for a simplified mapping. However, Fréchet–differentiability of the mappings J, H and G can be established in a similar way and is therefore not demonstrated explicitly.

Theorem 8.2
Let $\hat{\boldsymbol{x}} \in \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}})$ be given and let $\boldsymbol{f} : \mathcal{T} \times \mathbb{R}^{n_\mathrm{x}} \longrightarrow \mathbb{R}^{n_\mathrm{x}}$, $(t, \boldsymbol{x}) \mapsto \boldsymbol{f}(t, \boldsymbol{x})$ be a function satisfying the conditions in Assumptions 8.1 with

$$M \stackrel{\text{def}}{=} \{x \in \mathbb{R}^{n_\mathrm{x}} \,:\, \exists t \in \mathcal{T}, \|\boldsymbol{x} - \hat{\boldsymbol{x}}(t)\| \leqslant r\}, \quad r > 0.$$

Then the mapping $F : \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}}) \longrightarrow \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}})$ defined by

$$F(\boldsymbol{x}(\cdot)) \stackrel{\text{def}}{=} \dot{\boldsymbol{x}}(\cdot) - \boldsymbol{f}(\cdot, \boldsymbol{x}(\cdot))$$

is continuously Fréchet–differentiable in $\hat{\boldsymbol{x}}$ with derivative

$$F'(\hat{\boldsymbol{x}})(\boldsymbol{x}) = \dot{\boldsymbol{x}}(\cdot) - \boldsymbol{f}'_x(\cdot, \hat{\boldsymbol{x}}(\cdot))\boldsymbol{x}(\cdot). \qquad \triangle$$

Proof In a first step we show linearity and continuity of $F'(\hat{\boldsymbol{x}})(\boldsymbol{x})(\cdot)$. The linearity of the operator is obvious and it remains to show the continuity. Let $1 \leqslant n \leqslant N$ be arbitrary. Then it holds for almost every $t \in \mathcal{I}_n$ and $\boldsymbol{x} \in \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}})$

$$\begin{aligned}
\left\|F'(\hat{\boldsymbol{x}})(\boldsymbol{x})(t)\right\| &\leqslant \|\dot{\boldsymbol{x}}(t)\| + \left\|\boldsymbol{f}'_x(t, \hat{\boldsymbol{x}}(t))\right\| \cdot \|\boldsymbol{x}(t)\| \\
&\leqslant \sup_{t \in \mathcal{I}_n} \|\dot{\boldsymbol{x}}(t)\| + C_{\mathcal{I}_n} \cdot \sup_{t \in \mathcal{I}_n} \|\boldsymbol{x}(t)\| \\
&\leqslant (1 + C_{\mathcal{I}_n}) \cdot \left\{\sup_{t \in \mathcal{I}_n} \|\boldsymbol{x}(t)\| + \sup_{t \in \mathcal{I}_n} \|\dot{\boldsymbol{x}}(t)\|\right\} \\
&= (1 + C_{\mathcal{I}_n}) \cdot \|\boldsymbol{x}\|_{C_b^1(\mathcal{I}_n)}. \qquad (8.6)
\end{aligned}$$

Thus, it holds for almost every $t \in \mathcal{T}$ and $\boldsymbol{x} \in \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_x})$

$$\begin{aligned}\left\|F'(\hat{\boldsymbol{x}})(\boldsymbol{x})(t)\right\| &\leqslant \max_{n\in[N]} \left\{(1 + C_{\mathcal{I}_n}) \cdot \|\boldsymbol{x}\|_{\mathcal{C}_b^1(\mathcal{I}_n)}\right\} \\ &\leqslant (1 + C) \cdot \max_{n\in[N]} \|\boldsymbol{x}\|_{\mathcal{C}_b^1(\mathcal{I}_n)} \\ &= (1 + C) \cdot \|\boldsymbol{x}\|_{\mathcal{Y}^1(\mathcal{T})}.\end{aligned}$$

Hence, $F'(\hat{\boldsymbol{x}})(\boldsymbol{x})(\cdot)$ is continuous. The continuity of $F'(\cdot)$ in M follows because $\boldsymbol{f}_x'$ is supposed to be continuous with respect to x and bounded in M uniformly with respect to $t \in \mathcal{T}$. In the second step of this proof we show that

$$\lim_{\|\boldsymbol{x}\|_{\mathcal{Y}^1(\mathcal{T})}\to 0} \frac{\|F(\hat{\boldsymbol{x}} + \boldsymbol{x}) - F(\hat{\boldsymbol{x}}) - F'(\hat{\boldsymbol{x}})(\boldsymbol{x})\|_{\mathcal{Y}(\mathcal{T})}}{\|\boldsymbol{x}\|_{\mathcal{Y}^1(\mathcal{T})}} = 0.$$

With the mean–value theorem (see Theorem 2.28) we get for almost every $t \in \mathcal{T}$

$$\begin{aligned}\Delta(\boldsymbol{x})(t) &\stackrel{\text{def}}{=} \big(F(\hat{\boldsymbol{x}} + \boldsymbol{x}) - F(\hat{\boldsymbol{x}}) - F'(\hat{\boldsymbol{x}})(\boldsymbol{x})\big)(t) \\ &= -\big(\boldsymbol{f}(t, \hat{\boldsymbol{x}}(t) + \boldsymbol{x}(t)) - \boldsymbol{f}(t, \hat{\boldsymbol{x}}(t))\big) + \boldsymbol{f}_x'(t, \hat{\boldsymbol{x}}(t))\boldsymbol{x}(t) \\ &= -\int_0^1 \big(\boldsymbol{f}_x'(t, \hat{\boldsymbol{x}}(t) + \tau\boldsymbol{x}(t)) - \boldsymbol{f}_x'(t, \hat{\boldsymbol{x}}(t))\big)\boldsymbol{x}(t)\mathrm{d}\tau\end{aligned}$$

and conclude

$$\|\Delta(\boldsymbol{x})(t)\| \leqslant \sup_{\tau\in[0,1]} \left\|\boldsymbol{f}_x'(t, \hat{\boldsymbol{x}}(t) + \tau\boldsymbol{x}(t)) - \boldsymbol{f}_x'(t, \hat{\boldsymbol{x}}(t))\right\| \cdot \|\boldsymbol{x}\|_{\mathcal{Y}^1(\mathcal{T})}.$$

According to 2b in Assumption 8.1, $\boldsymbol{f}_x'(t, \cdot)$ is uniformly continuous on the compact set M. Hence, for every $\varepsilon > 0$ there exists $\delta \in (0, r]$ with

$$\left\|\boldsymbol{f}_x'(t, x_1) - \boldsymbol{f}_x'(t, x_2)\right\| \leqslant \varepsilon \quad \forall x_1, x_2 \in M,\ t \in \mathcal{T},\ \|x_1 - x_2\| \leqslant \delta.$$

Let $\|\boldsymbol{x}\|_{\mathcal{Y}^1(\mathcal{T})} \leqslant \delta$ and $\tau \in [0, 1]$. Then,

$$\hat{\boldsymbol{x}}(t),\ \hat{\boldsymbol{x}}(t) + \tau \cdot \boldsymbol{x}(t) \in M \quad \forall t \in \mathcal{T}$$

and

$$\|\hat{\boldsymbol{x}}(t) + \tau \cdot \boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)\| = \tau \cdot \|\boldsymbol{x}(t)\| \leqslant \|\boldsymbol{x}\|_{\mathcal{Y}^1(\mathcal{T})} \leqslant \delta.$$

The choice of δ implies

$$\|\Delta(\boldsymbol{x})\|_{\mathcal{Y}(\mathcal{T})} \leqslant \varepsilon \cdot \|\boldsymbol{x}\|_{\mathcal{Y}^1(\mathcal{T})} \quad \forall\, \|\boldsymbol{x}\|_{\mathcal{Y}^1(\mathcal{T})} \leqslant \delta,$$

and hence

$$\lim_{\|\boldsymbol{x}\|_{\mathcal{Y}^1(\mathcal{T})}\to 0} \frac{\|F(\hat{\boldsymbol{x}} + \boldsymbol{x}) - F(\hat{\boldsymbol{x}}) - F'(\hat{\boldsymbol{x}})(\boldsymbol{x})\|_{\mathcal{Y}(\mathcal{T})}}{\|\boldsymbol{x}\|_{\mathcal{Y}^1(\mathcal{T})}} = 0. \qquad \square$$

In order to be able to apply Theorem 3.7 we have to ensure the non–density assumption of the theorem. To this end, we prove the following auxiliary lemma which directly implies that the

Fréchet–derivative $H'(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}})$ is surjective. The lemma involves the linear homogeneous IVP

$$\dot{\boldsymbol{\Phi}}(t) = \boldsymbol{A}(t)\,\boldsymbol{\Phi}(t), \qquad \boldsymbol{\Phi}(t_\mathrm{s}) = \boldsymbol{I}_{n_\mathrm{x}}, \tag{8.7}$$

with $\boldsymbol{A} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_\mathrm{x} \times n_\mathrm{x}})$, i.e., $\boldsymbol{A}(\cdot)$ is not continuous, but only piecewise continuous. Hence, it makes no sense to expect the existence of a continuously differentiable function $\boldsymbol{\Phi} : \mathcal{T} \longrightarrow \mathbb{R}^{n_\mathrm{x} \times n_\mathrm{x}}$ that satisfies the IVP. Instead, we use a solution concept that is similar to the one of Carathéodory solutions for switched systems with consistent switches, cf. Section 1.2.2. We consider a function $\boldsymbol{\Phi} : \mathcal{T}_\Phi \longrightarrow \mathbb{R}^{n_\mathrm{x} \times n_\mathrm{x}}$ as a solution of IVP (8.7) if it satisfies

$$\boldsymbol{\Phi}(t) = \boldsymbol{I}_{n_\mathrm{x}} + \int_{t_\mathrm{s}}^{t} \boldsymbol{A}(s)\,\boldsymbol{\Phi}(s)\,\mathrm{d}s \quad \forall t \in \mathcal{T}_\Phi,$$

where $\mathcal{T}_\Phi \subset \mathcal{T}$ denotes an interval with $t_\mathrm{s} \in \mathcal{T}_\Phi$. One can show (see Logemann and Ryan [300, Theorems A.30+A.31]) that $\boldsymbol{\Phi} : \mathcal{T}_\Phi \longrightarrow \mathbb{R}^{n_\mathrm{x} \times n_\mathrm{x}}$ is a solution of IVP (8.7) if and only if $\boldsymbol{\Phi}(\cdot)$ is piecewise continuously differentiable with

$$\dot{\boldsymbol{\Phi}}(t) = \boldsymbol{A}(t)\,\boldsymbol{\Phi}(t) \quad \forall t \in \mathcal{T}_\Phi \setminus \mathcal{N}, \qquad \boldsymbol{\Phi}(t_\mathrm{s}) = \boldsymbol{I}_{n_\mathrm{x}},$$

where $\mathcal{N}$ denotes the finite set (i.e., $\mu(\mathcal{N}) = 0$) of discontinuities of $\boldsymbol{A}(\cdot)$ in $\mathcal{T}$. The existence of a solution of IVP (8.7) is guaranteed (see Logemann and Ryan [300, Theorems 2.5]) and the solution can be characterized by means of a transition matrix function.

Lemma 8.3
Let a vector $b \in \mathbb{R}^{n_\mathrm{r}}$ and matrices $E_\mathrm{s}, E_\mathrm{f} \in \mathbb{R}^{n_\mathrm{r} \times n_\mathrm{x}}$ be given and let the fundamental system $\boldsymbol{\Phi}(t) \in \mathbb{R}^{n_\mathrm{x} \times n_\mathrm{x}}$ be the solution of the IVP

$$\dot{\boldsymbol{\Phi}}(t) = \boldsymbol{A}(t)\,\boldsymbol{\Phi}(t), \qquad \boldsymbol{\Phi}(t_\mathrm{s}) = \boldsymbol{I}_{n_\mathrm{x}}, \tag{8.8}$$

where $\boldsymbol{A} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_\mathrm{x} \times n_\mathrm{x}})$ is a time dependent matrix function. Let the fundamental solution $\boldsymbol{\Phi}(\cdot)$ satisfy the condition

$$\operatorname{rank}(E_\mathrm{s}\boldsymbol{\Phi}(t_\mathrm{s}) + E_\mathrm{f}\boldsymbol{\Phi}(t_\mathrm{f})) = n_\mathrm{r}. \tag{8.9}$$

Let a boundary value problem be defined as

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}(t)\,\boldsymbol{x}(t) + \boldsymbol{h}(t), \tag{8.10}$$

$$b = E_\mathrm{s}\,\boldsymbol{x}(t_\mathrm{s}) + E_\mathrm{f}\,\boldsymbol{x}(t_\mathrm{f}), \tag{8.11}$$

where $\boldsymbol{h} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}})$ is a time dependent vector function. Then IVP (8.10)–(8.11) has a solution $\boldsymbol{x} \in \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}})$ for every $\boldsymbol{h} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}})$ and $b \in \mathbb{R}^{n_\mathrm{r}}$. △

Proof We start by parameterizing the ODE (8.10) with the initial value condition $\boldsymbol{x}(t_\mathrm{s}) = x_\mathrm{s}$, i.e., we consider the IVP

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}(t)\,\boldsymbol{x}(t) + \boldsymbol{h}(t), \quad \boldsymbol{x}(t_\mathrm{s}) = x_s.$$

It can be easily checked that its solution $\boldsymbol{x}(\cdot)$ can be expressed by means of the solution of IVP (8.8) as

$$\boldsymbol{x}(t) = \boldsymbol{\Phi}(t)\left(x_s + \int_{t_s}^{t} \boldsymbol{\Phi}^{-1}(\tau)\,\boldsymbol{h}(\tau)\,\mathrm{d}\tau \right), \quad t \in \mathcal{T}.$$

The boundary condition (8.11) holds if

$$\begin{aligned} b &= E_s\,\boldsymbol{x}(t_s) + E_f\,\boldsymbol{x}(t_f) \\ &= E_s\,x_s + E_f\,\boldsymbol{\Phi}(t_f)\left(x_s + \int_{t_s}^{t_f} \boldsymbol{\Phi}^{-1}(\tau)\,\boldsymbol{h}(\tau)\,\mathrm{d}\tau \right) \\ &= (E_s + E_f\,\boldsymbol{\Phi}(t_f))\,x_s + E_f\,\boldsymbol{\Phi}(t_f) \int_{t_s}^{t_f} \boldsymbol{\Phi}^{-1}(\tau)\,\boldsymbol{h}(\tau)\,\mathrm{d}\tau \end{aligned}$$

If we reformulate terms and exploit $\boldsymbol{\Phi}(t_s) = \boldsymbol{I}_{n_x}$ then we get

$$(E_s\,\boldsymbol{\Phi}(t_s) + E_f\,\boldsymbol{\Phi}(t_f))\,x_s = b - E_f\,\boldsymbol{\Phi}(t_f) \int_{t_s}^{t_f} \boldsymbol{\Phi}^{-1}(\tau)\,\boldsymbol{h}(\tau)\,\mathrm{d}\tau.$$

Due to the rank condition (8.9) the matrix $E_s\,\boldsymbol{\Phi}(t_s) + E_f\,\boldsymbol{\Phi}(t_f)$ has full rank, and therefore the equation is solvable for every $b \in \mathbb{R}^{n_b}$ which completes the proof. □

Thus, Assumption 8.1 ensures that Theorem 3.7 can be applied to OCP (8.2) and we can summarize:

Theorem 8.4 (Necessary Optimality Conditions for OCP (8.2)**)**
Let us suppose that the following assumptions hold for OCP (8.2):

(i) Assumption 8.1 holds.

(ii) $(\boldsymbol{x}^*, \boldsymbol{u}^*)$ is a local minimum of OCP (8.2).

Then there exist nontrivial multipliers $l_0 \geqslant 0$, $\lambda^* \in V^*$ and $\mu^* \in W^*$ such that

$$\mu^* \in \mathcal{K}^+ \qquad \text{and} \qquad \mu^*(G(\boldsymbol{x}^*, \boldsymbol{u}^*)) = \boldsymbol{0}, \tag{8.12}$$

and

$$\boldsymbol{0} = l_0\,J'(\boldsymbol{x}^*, \boldsymbol{u}^*)(\boldsymbol{x}, \boldsymbol{u}) - \lambda^*(H'(\boldsymbol{x}^*, \boldsymbol{u}^*)(\boldsymbol{x}, \boldsymbol{u})) - \mu^*(G'(\boldsymbol{x}^*, \boldsymbol{u}^*)(\boldsymbol{x}, \boldsymbol{u})) \tag{8.13}$$

for all $(\boldsymbol{x}, \boldsymbol{u}) \in Z$. △

The multipliers $\lambda^* \overset{\text{def}}{=} (\lambda_f^*, \nu)$, where $\nu \in \mathbb{R}^{n_r}$ denotes the boundary value constraint multiplier, and μ^* are elements of the dual spaces

$$V^* = \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_x})^* \times (\mathbb{R}^{n_r})^* \qquad \text{and} \qquad W^* = \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_c})^*.$$

The following section deals with the problem to find an explicit representation of the multipliers. The variational equation (8.13) states that

$$0 = \left(\boldsymbol{\varphi}'_{x_s} + \nu^T \boldsymbol{r}'_{x_s} \right) \boldsymbol{x}(t_s) + \left(\boldsymbol{\varphi}'_{x_f} + \nu^T \boldsymbol{r}'_{x_f} \right) \boldsymbol{x}(t_f)$$

$$+\int_{t_s}^{t_f} \psi_x'[t]\boldsymbol{x}(t)\,\mathrm{d}t + \lambda_f^*\left(\dot{\boldsymbol{x}}(\cdot) - \boldsymbol{f}_x'[\cdot]\boldsymbol{x}(\cdot)\right) + \mu^*\left(\boldsymbol{c}_x'[\cdot]\boldsymbol{x}(\cdot)\right) \tag{8.14}$$

$$0 = \int_{t_s}^{t_f} \psi_u'[t]\boldsymbol{u}(t)\,\mathrm{d}t - \lambda_f^*\left(\boldsymbol{f}_u'[\cdot]\boldsymbol{u}(\cdot)\right) + \mu^*\left(\boldsymbol{c}_u'[\cdot]\boldsymbol{u}(\cdot)\right) \tag{8.15}$$

hold for every $\boldsymbol{x} \in \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_x})$ and every $\boldsymbol{u} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_u})$.

8.2 Representation of Multipliers

One could argue that the variational equations (8.14)+(8.15) are of little practical use because the multipliers λ_f^* and μ^* appear as elements of the dual spaces $\mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_x})^*$ and $\mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_c})^*$. However, in this section we find explicit representations for them which enable us to formulate a discrete local minimum principle in the following section.
According to the duality pairing of $\mathcal{NBV}(\mathcal{T}, \mathbb{R})$ and $\mathcal{Y}(\mathcal{T}, \mathbb{R})$ the functional λ_f^* possesses the explicit representation

$$\lambda_f^*(\boldsymbol{h}(\cdot)) = \sum_{n=1}^{N} \int_{\mathcal{I}_n} \boldsymbol{h}(t)^T \mathrm{d}\,\boldsymbol{\Lambda}(t) \tag{8.16}$$

for every vector function $\boldsymbol{h} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_x})$. Herein, the components of $\boldsymbol{\Lambda}(\cdot)$ are functions of bounded variation. Likewise, the functional μ^* possesses the explicit representation

$$\mu^*(\boldsymbol{h}(\cdot)) = \sum_{n=1}^{N} \int_{\mathcal{I}_n} \boldsymbol{h}(t)^T \mathrm{d}\,\mathbf{M}(t) \tag{8.17}$$

for every vector function $\boldsymbol{h} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_c})$. Also in this case the components of $\mathbf{M}(\cdot)$ are functions of bounded variation. In order to make the representation unique we choose both $\boldsymbol{\Lambda}(\cdot)$ and $\mathbf{M}(\cdot)$ to be normalized such that we have $\boldsymbol{\Lambda} \in \mathcal{NBV}(\mathcal{T}, \mathbb{R}^{n_x})$ and $\mathbf{M} \in \mathcal{NBV}(\mathcal{T}, \mathbb{R}^{n_c})$. In the following Chapter 9 we employ the representations (8.16) and (8.17) to interrelate a direct approach based on a pseudospectral method and an indirect approach based on a Petrov–Galerkin discretization scheme.
Next, we intensify our analysis of the multipliers λ_f^* and μ^*. We start with an exploitation of Equation (8.14) to find a representation of the functional λ_f^*. For arbitrary $\boldsymbol{h} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_x})$ we consider the IVP

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{f}_x'[t]\,\boldsymbol{x}(t) + \boldsymbol{h}(t), \quad t \in \mathcal{T}, \\ \boldsymbol{x}(t_s) &= x_s. \end{aligned}$$

As we have seen in the proof of Lemma 8.3 its solution is given as

$$\boldsymbol{x}(t) = \boldsymbol{\Phi}(t)\left(x_s + \int_{t_s}^{t} \boldsymbol{\Phi}^{-1}(\tau)\boldsymbol{h}(\tau)\,\mathrm{d}\tau\right), \quad t \in \mathcal{T}, \tag{8.18}$$

where $\boldsymbol{\Phi}(\cdot)$ denotes the fundamental solution of the IVP

$$\dot{\boldsymbol{\Phi}}(t) = \boldsymbol{f}'_x[t]\boldsymbol{\Phi}(t), \quad \boldsymbol{\Phi}(t_\mathrm{s}) = \boldsymbol{I}_{n_x}.$$

For an arbitrary $\boldsymbol{h} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_x})$ Equation (8.14) reads as

$$\begin{aligned} 0 = &\left(\boldsymbol{\varphi}'_{x_s} + \nu^T \boldsymbol{r}'_{x_s}\right)\boldsymbol{x}(t_\mathrm{s}) + \left(\boldsymbol{\varphi}'_{x_f} + \nu^T \boldsymbol{r}'_{x_f}\right)\boldsymbol{x}(t_\mathrm{f}) \\ &+ \int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{\psi}'_x[t]\boldsymbol{x}(t)\,\mathrm{d}t + \lambda_\mathrm{f}^*(\boldsymbol{h}(\cdot)) + \mu^*\left(\boldsymbol{c}'_x[\cdot]\boldsymbol{x}(\cdot)\right). \end{aligned} \tag{8.19}$$

Now, we introduce the solution formula (8.18) into Equation (8.19) and rearrange terms as follows:

$$\begin{aligned} 0 = &\left(\boldsymbol{\varphi}'_{x_s} + \nu^T \boldsymbol{r}'_{x_s}\right)x_s + \left(\boldsymbol{\varphi}'_{x_f} + \nu^T \boldsymbol{r}'_{x_f}\right)\boldsymbol{\Phi}(t_\mathrm{f})x_s \\ &+ \int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{\psi}'_x[t]\boldsymbol{\Phi}(t)x_s\,\mathrm{d}t + \left(\boldsymbol{\varphi}'_{x_f} + \nu^T \boldsymbol{r}'_{x_f}\right)\boldsymbol{\Phi}(t_\mathrm{f})\int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{\Phi}^{-1}(t)\boldsymbol{h}(t)\,\mathrm{d}t \\ &+ \int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{\psi}'_x[t]\boldsymbol{\Phi}(t)\left(\int_{t_\mathrm{s}}^{t} \boldsymbol{\Phi}^{-1}(\tau)\boldsymbol{h}(\tau)\,\mathrm{d}\tau\right)\mathrm{d}t \\ &+ \lambda_\mathrm{f}^*(\boldsymbol{h}(\cdot)) + \mu^*\left(\boldsymbol{c}'_x[\cdot]\boldsymbol{x}(\cdot)\right). \end{aligned} \tag{8.20}$$

For the reader's convenience, we provide the following result:

Lemma 8.5
Let functions $\boldsymbol{f} : \mathcal{T} \longrightarrow \mathbb{R}$ and $\boldsymbol{g} : \mathcal{T} \longrightarrow \mathbb{R}$ (both piecewise continuous) be given. Then it holds

$$\int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{f}(t)\left(\int_{t_\mathrm{s}}^{t} \boldsymbol{g}(\tau)\,\mathrm{d}\tau\right)\mathrm{d}t = \int_{t_\mathrm{s}}^{t_\mathrm{f}} \left(\int_{t}^{t_\mathrm{f}} \boldsymbol{f}(\tau)\,\mathrm{d}\tau\right)\boldsymbol{g}(t)\,\mathrm{d}t. \qquad \triangle$$

Proof See Appendix A.1. □

Applying Lemma 8.5 to Equation (8.20) and rearranging terms yields

$$\begin{aligned} 0 = &\left(\boldsymbol{\varphi}'_{x_s} + \nu^T \boldsymbol{r}'_{x_s}\right)x_s + \left(\boldsymbol{\varphi}'_{x_f} + \nu^T \boldsymbol{r}'_{x_f}\right)\boldsymbol{\Phi}(t_\mathrm{f})x_s + \int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{\psi}'_x[t]\boldsymbol{\Phi}(t)x_s\,\mathrm{d}t \\ &+ \int_{t_\mathrm{s}}^{t_\mathrm{f}} \left\{\left(\boldsymbol{\varphi}'_{x_f} + \nu^T \boldsymbol{r}'_{x_f}\right)\boldsymbol{\Phi}(t_\mathrm{f}) + \int_{t}^{t_\mathrm{f}} \boldsymbol{\psi}'_x[\tau]\boldsymbol{\Phi}(\tau)\,\mathrm{d}\tau\right\}\boldsymbol{\Phi}^{-1}(t)\boldsymbol{h}(t)\,\mathrm{d}t \\ &+ \lambda_\mathrm{f}^*(\boldsymbol{h}(\cdot)) + \mu^*\left(\boldsymbol{c}'_x[\cdot]\boldsymbol{x}(\cdot)\right). \end{aligned}$$

We can write the equation equivalently as

$$0 = \zeta^T x_s + \int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{p}_\mathrm{f}(t)^T \boldsymbol{h}(t)\,\mathrm{d}t + \lambda_\mathrm{f}^*(\boldsymbol{h}(\cdot)) + \mu^*\left(\boldsymbol{c}'_x[\cdot]\boldsymbol{x}(\cdot)\right), \tag{8.21}$$

where

$$\begin{aligned}\zeta^T &\stackrel{\text{def}}{=} \left(\varphi'_{x_s} + \nu^T \boldsymbol{r}'_{x_s}\right) + \left(\varphi'_{x_f} + \nu^T \boldsymbol{r}'_{x_f}\right)\boldsymbol{\Phi}(t_\mathrm{f}) + \int_{t_\mathrm{s}}^{t_\mathrm{f}} \psi'_x[t]\boldsymbol{\Phi}(t)\,\mathrm{d}t \\ &= \left(\varphi'_{x_s} + \nu^T \boldsymbol{r}'_{x_s}\right) + \boldsymbol{p}_\mathrm{f}(t_\mathrm{s}), \\ \boldsymbol{p}_\mathrm{f}(t)^T &\stackrel{\text{def}}{=} \left\{\left(\varphi'_{x_f} + \nu^T \boldsymbol{r}'_{x_f}\right)\boldsymbol{\Phi}(t_\mathrm{f}) + \int_t^{t_\mathrm{f}} \psi'_x[\tau]\boldsymbol{\Phi}(\tau)\,\mathrm{d}\tau\right\}\boldsymbol{\Phi}^{-1}(t).\end{aligned}$$

Exploiting (8.21) together with (8.15) provides us with an explicit representation of functionals λ_f^* and μ^*. In particular, we deduce the function spaces of the functions $\boldsymbol{\lambda}(\cdot)$ and $\boldsymbol{\mu}(\cdot)$ that characterize λ_f^* and μ^*.

Corollary 8.6 (Explicit Representation of Multipliers)
Let the assumptions of Theorem 8.4 be satisfied and let

$$\operatorname{rank}\left(\boldsymbol{c}'_u[t]\right) = n_\mathrm{c} \tag{8.22}$$

be almost everywhere in $\mathcal{T}$. Furthermore, let the pseudo–inverse of $\boldsymbol{c}'_u[t]$

$$\left(\boldsymbol{c}'_u[t]\right)^+ \stackrel{\text{def}}{=} \boldsymbol{c}'_u[t]^T\left(\boldsymbol{c}'_u[t]\boldsymbol{c}'_u[t]^T\right)^{-1}$$

be essentially bounded. Then there exist functions

$$\boldsymbol{\lambda} \in \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}}), \quad \boldsymbol{\mu} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_\mathrm{c}}),$$

with

$$\begin{aligned}\lambda_\mathrm{f}^*(\boldsymbol{h}(\cdot)) &= -\int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{\lambda}(t)^T \boldsymbol{h}(t)\,\mathrm{d}t, \\ \mu^*(\boldsymbol{k}(\cdot)) &= \int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{\mu}(t)^T \boldsymbol{k}(t)\,\mathrm{d}t\end{aligned}$$

for every $\boldsymbol{h} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}})$ and every $\boldsymbol{k} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_\mathrm{c}})$. △

Proof Choosing $\boldsymbol{h}(\cdot) = \boldsymbol{f}'_u[\cdot]\boldsymbol{u}(\cdot)$ and $x_s = 0$ in (8.21) yields

$$\begin{aligned}& -\lambda_\mathrm{f}^*(\boldsymbol{h}(\cdot)) = \int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{p}_\mathrm{f}(t)^T \boldsymbol{h}(t)\,\mathrm{d}t + \mu^*\left(\boldsymbol{c}'_x[\cdot]\boldsymbol{x}(\cdot)\right) \\ \iff \quad & -\lambda_\mathrm{f}^*\left(\boldsymbol{f}'_u[\cdot]\boldsymbol{u}(\cdot)\right) = \int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{p}_\mathrm{f}(t)^T \boldsymbol{f}'_u[t]\boldsymbol{u}(t)\,\mathrm{d}t + \mu^*\left(\boldsymbol{c}'_x[\cdot]\boldsymbol{x}(\cdot)\right),\end{aligned} \tag{8.23}$$

where $\boldsymbol{x}(\cdot)$ is the solution of the IVP

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}'_x[t]\boldsymbol{x}(t) + \boldsymbol{f}'_u[t]\boldsymbol{u}(t), \quad \boldsymbol{x}(t_\mathrm{s}) = 0. \tag{8.24}$$

Now we write (8.15) as

$$\lambda_{\mathrm{f}}^{*}\left(\boldsymbol{f}_{u}'[\cdot]\boldsymbol{u}(\cdot)\right)=\int_{t_{\mathrm{s}}}^{t_{\mathrm{f}}}\boldsymbol{\psi}_{u}'[t]\boldsymbol{u}(t)\,\mathrm{d}t+\mu^{*}\left(\boldsymbol{c}_{u}'[\cdot]\boldsymbol{u}(\cdot)\right), \tag{8.25}$$

sum up both (8.23) and (8.25), exploit the linearity of the functional μ^{*}, and write the result as

$$\begin{aligned}0&=\int_{t_{\mathrm{s}}}^{t_{\mathrm{f}}}\left(\boldsymbol{\psi}_{u}'[t]+\boldsymbol{p}_{\mathrm{f}}(t)^{T}\boldsymbol{f}_{u}'[t]\right)\boldsymbol{u}(t)\,\mathrm{d}t+\mu^{*}\left(\boldsymbol{c}_{x}'[\cdot]\boldsymbol{x}(\cdot)+\boldsymbol{c}_{u}'[\cdot]\boldsymbol{u}(\cdot)\right)\\&=\int_{t_{\mathrm{s}}}^{t_{\mathrm{f}}}\mathcal{H}_{u}'[t]\boldsymbol{u}(t)\,\mathrm{d}t+\mu^{*}\left(\boldsymbol{k}(\cdot)\right),\end{aligned} \tag{8.26}$$

where

$$\mathcal{H}_{u}'[t]\overset{\text{def}}{=}\boldsymbol{\psi}_{u}'[t]+\boldsymbol{p}_{\mathrm{f}}(t)^{T}\boldsymbol{f}_{u}'[t]$$

and

$$\boldsymbol{k}(t)\overset{\text{def}}{=}\boldsymbol{c}_{x}'[t]\boldsymbol{x}(t)+\boldsymbol{c}_{u}'[t]\boldsymbol{u}(t). \tag{8.27}$$

The rank assumption (8.22) enables us to express $\boldsymbol{u}(\cdot)$ in Equation (8.27) explicitly as

$$\boldsymbol{u}(t)=\left(\boldsymbol{c}_{u}'[t]\right)^{+}\left(\boldsymbol{k}(t)-\boldsymbol{c}_{x}'[t]\boldsymbol{x}(t)\right), \tag{8.28}$$

where $\left(\boldsymbol{c}_{u}'[t]\right)^{+}$ denotes the pseudo–inverse of $\boldsymbol{c}_{u}'[t]$. Introducing (8.28) into Equation (8.26) results in

$$0=\int_{t_{\mathrm{s}}}^{t_{\mathrm{f}}}\mathcal{H}_{u}'[t]\left(\boldsymbol{c}_{u}'[t]\right)^{+}\left(\boldsymbol{k}(t)-\boldsymbol{c}_{x}'[t]\boldsymbol{x}(t)\right)\,\mathrm{d}t+\mu^{*}\left(\boldsymbol{k}(\cdot)\right). \tag{8.29}$$

Likewise we introduce (8.28) into Equation (8.24) and get

$$\begin{aligned}&\dot{\boldsymbol{x}}(t)=\boldsymbol{f}_{x}'[t]\boldsymbol{x}(t)+\boldsymbol{f}_{u}'[t]\left(\boldsymbol{c}_{u}'[t]\right)^{+}\left(\boldsymbol{k}(t)-\boldsymbol{c}_{x}'[t]\boldsymbol{x}(t)\right),\quad \boldsymbol{x}(t_{\mathrm{s}})=0\\\Longleftrightarrow\quad&\dot{\boldsymbol{x}}(t)=\hat{\boldsymbol{f}}_{x}[t]\boldsymbol{x}(t)+\hat{\boldsymbol{h}}(t),\quad \boldsymbol{x}(t_{\mathrm{s}})=0\end{aligned} \tag{8.30}$$

where

$$\hat{\boldsymbol{f}}_{x}[t]\overset{\text{def}}{=}\boldsymbol{f}_{x}'[t]-\boldsymbol{f}_{u}'[t]\left(\boldsymbol{c}_{u}'[t]\right)^{+}\boldsymbol{c}_{x}'[t]$$

and

$$\hat{\boldsymbol{h}}(t)\overset{\text{def}}{=}\boldsymbol{f}_{u}'[t]\left(\boldsymbol{c}_{u}'[t]\right)^{+}\boldsymbol{k}(t). \tag{8.31}$$

We use the well known solution formula for Equation (8.30) and write its solution as

$$\boldsymbol{x}(t)=\hat{\boldsymbol{\Phi}}(t)\int_{t_{\mathrm{s}}}^{t}\hat{\boldsymbol{\Phi}}^{-1}(\tau)\hat{\boldsymbol{h}}(\tau)\,\mathrm{d}\tau, \tag{8.32}$$

where $\hat{\boldsymbol{\Phi}}(\cdot)$ solves the IVP

$$\dot{\hat{\boldsymbol{\Phi}}}(t)=\hat{\boldsymbol{f}}_{x}[t]\hat{\boldsymbol{\Phi}}(t),\quad \hat{\boldsymbol{\Phi}}(t_{\mathrm{s}})=\boldsymbol{I}_{n_{\boldsymbol{x}}}.$$

Substituting (8.31) into the solution formula (8.32) yields

$$\boldsymbol{x}(t) = \hat{\boldsymbol{\Phi}}(t) \int_{t_s}^{t} \boldsymbol{\omega}(\tau)^T \boldsymbol{k}(\tau)\, \mathrm{d}\tau \tag{8.33}$$

with

$$\boldsymbol{\omega}(t)^T \stackrel{\text{def}}{=} \hat{\boldsymbol{\Phi}}^{-1}(t) \boldsymbol{f}_u'[t] \big(\boldsymbol{c}_u'[t]\big)^+ .$$

Now we insert (8.33) into (8.29) and integrate by parts, which results in

$$\begin{aligned}
-\mu^*(\boldsymbol{k}(\cdot)) &= \int_{t_s}^{t_f} \mathcal{H}_u'[t] \big(\boldsymbol{c}_u'[t]\big)^+ \big(\boldsymbol{k}(t) - \boldsymbol{c}_x'[t]\boldsymbol{x}(t)\big)\, \mathrm{d}t \\
&= \int_{t_s}^{t_f} \mathcal{H}_u'[t] \big(\boldsymbol{c}_u'[t]\big)^+ \boldsymbol{k}(t)\, \mathrm{d}t \\
&\quad - \int_{t_s}^{t_f} \mathcal{H}_u'[t] \big(\boldsymbol{c}_u'[t]\big)^+ \boldsymbol{c}_x'[t] \hat{\boldsymbol{\Phi}}(t) \left(\int_{t_s}^{t} \boldsymbol{\omega}(\tau)^T \boldsymbol{k}(\tau)\, \mathrm{d}\tau \right) \mathrm{d}t \\
&= \int_{t_s}^{t_f} \mathcal{H}_u'[t] \big(\boldsymbol{c}_u'[t]\big)^+ \boldsymbol{k}(t)\, \mathrm{d}t \\
&\quad - \int_{t_s}^{t_f} \left(\int_{t}^{t_f} \mathcal{H}_u'[\tau] \big(\boldsymbol{c}_u'[\tau]\big)^+ \boldsymbol{c}_x'[\tau] \hat{\boldsymbol{\Phi}}(\tau)\, \mathrm{d}\tau \right) \boldsymbol{\omega}(t)^T \boldsymbol{k}(t)\, \mathrm{d}t.
\end{aligned}$$

By introducing the function $\boldsymbol{\mu}(\cdot)$ as

$$\boldsymbol{\mu}(t)^T \stackrel{\text{def}}{=} \left(\int_{t}^{t_f} \mathcal{H}_u'[\tau] \big(\boldsymbol{c}_u'[\tau]\big)^+ \boldsymbol{c}_x'[\tau] \hat{\boldsymbol{\Phi}}(\tau)\, \mathrm{d}\tau \right) \boldsymbol{\omega}(t)^T - \mathcal{H}_u'[t] \big(\boldsymbol{c}_u'[t]\big)^+$$

we find the representation

$$\mu^*(\boldsymbol{k}(\cdot)) = \int_{t_s}^{t_f} \boldsymbol{\mu}(t)^T \boldsymbol{k}(t)\, \mathrm{d}t.$$

We substitute the representation into (8.21), integrate by parts, and exploit the solution formula (8.18) which yields

$$\begin{aligned}
-\lambda_f^*(\boldsymbol{h}(\cdot)) &= \zeta^T x_s + \int_{t_s}^{t_f} \boldsymbol{p}_f(t)^T \boldsymbol{h}(t)\, \mathrm{d}t + \int_{t_s}^{t_f} \boldsymbol{\mu}(t)^T \boldsymbol{c}_x'[t] \boldsymbol{x}(t)\, \mathrm{d}t \\
&= \zeta^T x_s + \int_{t_s}^{t_f} \boldsymbol{p}_f(t)^T \boldsymbol{h}(t)\, \mathrm{d}t + \int_{t_s}^{t_f} \boldsymbol{\mu}(t)^T \boldsymbol{c}_x'[t] \boldsymbol{\Phi}(t) \left(x_s + \int_{t_s}^{t} \boldsymbol{\Phi}^{-1}(\tau) \boldsymbol{h}(\tau)\, \mathrm{d}\tau \right) \mathrm{d}t \\
&= \left(\zeta^T + \int_{t_s}^{t_f} \boldsymbol{\mu}(t)^T \boldsymbol{c}_x'[t] \boldsymbol{\Phi}(t)\, \mathrm{d}t \right) x_s + \int_{t_s}^{t_f} \boldsymbol{p}_f(t)^T \boldsymbol{h}(t)\, \mathrm{d}t \\
&\quad + \int_{t_s}^{t_f} \left(\int_{t}^{t_f} \boldsymbol{\mu}(\tau)^T \boldsymbol{c}_x'[\tau] \boldsymbol{\Phi}(\tau)\, \mathrm{d}\tau \right) \boldsymbol{\Phi}^{-1}(t) \boldsymbol{h}(t)\, \mathrm{d}t.
\end{aligned}$$

Alternatively we can write it as

$$-\lambda_{\mathrm{f}}^{*}(\boldsymbol{h}(\cdot)) = \hat{\zeta}^{T} x_{\mathrm{s}} + \int_{t_{\mathrm{s}}}^{t_{\mathrm{f}}} \boldsymbol{\lambda}(t)^{T} \boldsymbol{h}(t)\,\mathrm{d}t$$

with

$$\hat{\zeta}^{T} \overset{\text{def}}{=} \zeta^{T} + \int_{t_{\mathrm{s}}}^{t_{\mathrm{f}}} \boldsymbol{\mu}(t)^{T} \boldsymbol{c}_{x}'[t]\,\boldsymbol{\Phi}(t)\,\mathrm{d}t$$

and

$$\boldsymbol{\lambda}(t)^{T} \overset{\text{def}}{=} \boldsymbol{p}_{\mathrm{f}}(t)^{T} + \left(\int_{t}^{t_{\mathrm{f}}} \boldsymbol{\mu}(\tau)^{T} \boldsymbol{c}_{x}'[\tau]\,\boldsymbol{\Phi}(\tau)\,\mathrm{d}\tau \right) \boldsymbol{\Phi}^{-1}(t).$$

Setting $x_{\mathrm{s}} = 0$ yields

$$\lambda_{\mathrm{f}}^{*}(\boldsymbol{h}(\cdot)) = -\int_{t_{\mathrm{s}}}^{t_{\mathrm{f}}} \boldsymbol{\lambda}(t)^{T} \boldsymbol{h}(t)\,\mathrm{d}t.$$ □

8.3 Local Minimum Principle

In this section we derive first–order necessary optimality conditions for OCP (8.2) in terms of a local minimum principle. They are expressed by means of the associated augmented Hamilton function.

Definition 8.7 (Hamilton Function for OCP (8.2))
Considering OCP (8.2) the Hamilton *function* $\mathcal{H} : \mathcal{T} \times \mathbb{R}^{n_{\mathrm{x}}} \times \mathbb{R}^{n_{\mathrm{u}}} \times \mathbb{R}^{n_{\mathrm{x}}} \times \mathbb{R} \longrightarrow \mathbb{R}$ is defined as

$$\mathcal{H}(t, x, u, \lambda, \mu, l_0) \overset{\text{def}}{=} l_0\, \psi(t, x, u) + \lambda^{T} \boldsymbol{f}(t, x, u).$$ △

Now, the augmented Hamilton function for OCP (8.2) is presented.

Definition 8.8 (Augmented Hamilton Function for OCP (8.2))
Considering OCP (8.2) the *augmented* Hamilton *function* $\hat{\mathcal{H}} : \mathcal{T} \times \mathbb{R}^{n_{\mathrm{x}}} \times \mathbb{R}^{n_{\mathrm{u}}} \times \mathbb{R}^{n_{\mathrm{x}}} \times \mathbb{R}^{n_{\mathrm{c}}} \times \mathbb{R} \longrightarrow \mathbb{R}$ is defined as

$$\begin{aligned}\hat{\mathcal{H}}(t, x, u, \lambda, \mu, l_0) &\overset{\text{def}}{=} \mathcal{H}(t, x, u, \lambda, \mu, l_0) + \mu^{T} \boldsymbol{c}(t, x, u) \\ &= l_0\, \psi(t, x, u) + \lambda^{T} \boldsymbol{f}(t, x, u) + \mu^{T} \boldsymbol{c}(t, x, u).\end{aligned}$$ △

Theorem 8.9 (Discrete Local Minimum Principle)
Let the following assumptions be satisfied by the OCP (8.2):

(i) Let the functions φ, ψ, $\boldsymbol{f}$, $\boldsymbol{c}$ and $\boldsymbol{r}$ be continuous with respect to all arguments and continuously differentiable with respect to $\boldsymbol{x}$ and $\boldsymbol{u}$.

(ii) Let $(\boldsymbol{x}^{*}, \boldsymbol{u}^{*})$ be a weak local minimum of the OCP.

(iii) Let

$$\operatorname{rank}(\boldsymbol{c}_{u}'(t, \boldsymbol{x}^{*}(t), \boldsymbol{u}^{*}(t))) = n_{\mathrm{c}}$$

almost everywhere in $\mathcal{T}$.

(iv) Let the pseudo–inverse of $\boldsymbol{c}_u'[t]$

$$\left(\boldsymbol{c}_u'[t]\right)^+ = \boldsymbol{c}_u'[t]^T\left(\boldsymbol{c}_u'[t]\boldsymbol{c}_u'[t]^T\right)^{-1}$$

be essentially bounded.

Then there exist multipliers $\nu \in \mathbb{R}^{n_\mathrm{r}}$,

$$\boldsymbol{\lambda} \in \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}}), \quad \boldsymbol{\mu} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_\mathrm{c}})$$

such that the following conditions hold:

(i) $(l_0, \boldsymbol{\lambda}, \boldsymbol{\mu}, \nu) \neq \Theta$

(ii) Adjoint equations: almost everywhere in $\mathcal{T}$ it holds

$$\dot{\boldsymbol{\lambda}}(t) = -\mathcal{H}_x'(t, \boldsymbol{x}^*(t), \boldsymbol{u}^*(t), \boldsymbol{\lambda}(t), \boldsymbol{\mu}(t))^T.$$

(iii) Transversality conditions:

$$\boldsymbol{\lambda}(t_\mathrm{s})^T = -\left(\boldsymbol{\varphi}_{x_\mathrm{s}}'(\boldsymbol{x}^*(t_\mathrm{s}), \boldsymbol{x}^*(t_\mathrm{f})) + \nu^T \boldsymbol{r}_{x_\mathrm{s}}'(\boldsymbol{x}^*(t_\mathrm{s}), \boldsymbol{x}^*(t_\mathrm{f}))\right) \tag{8.34}$$

$$\boldsymbol{\lambda}(t_\mathrm{f})^T = \boldsymbol{\varphi}_{x_\mathrm{f}}'(\boldsymbol{x}^*(t_\mathrm{s}), \boldsymbol{x}^*(t_\mathrm{f})) + \nu^T \boldsymbol{r}_{x_\mathrm{f}}'(\boldsymbol{x}^*(t_\mathrm{s}), \boldsymbol{x}^*(t_\mathrm{f})) \tag{8.35}$$

(iv) Stationarity of Hamilton function: almost everywhere in $\mathcal{T}$ it holds

$$\mathbf{0}_{n_\mathrm{u}} = \mathcal{H}_u'(t, \boldsymbol{x}^*(t), \boldsymbol{u}^*(t), \boldsymbol{\lambda}(t), \boldsymbol{\mu}(t))^T.$$

(v) Complementarity condition: almost everywhere in $\mathcal{T}$ it holds

$$\mathbf{0}_{n_\mathrm{c}} \leqslant \boldsymbol{\mu}(t) \qquad \text{and} \qquad \boldsymbol{\mu}(t)^T \boldsymbol{c}(t, \boldsymbol{x}^*(t), \boldsymbol{u}^*(t)) = 0. \tag{8.36}$$

△

Proof Under the assumptions of this theorem Corollary 8.6 ensures that there exist Lagrange multipliers $\boldsymbol{\lambda} \in \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}})$ and $\boldsymbol{\mu} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_\mathrm{c}})$ such that the dual functionals λ_f^* and μ^* can be expressed as

$$\lambda_\mathrm{f}^*(\boldsymbol{h}(\cdot)) = -\int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{\lambda}(t)^T \boldsymbol{h}(t)\,\mathrm{d}t,$$

$$\mu^*(\boldsymbol{k}(\cdot)) = \int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{\mu}(t)^T \boldsymbol{k}(t)\,\mathrm{d}t$$

for every $\boldsymbol{h} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_\mathrm{x}})$ and every $\boldsymbol{k} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_\mathrm{c}})$. Equation (8.14) is therefore equivalent to

$$\begin{aligned} 0 = &\left(\boldsymbol{\varphi}_{x_\mathrm{s}}' + \nu^T \boldsymbol{r}_{x_\mathrm{s}}'\right)\boldsymbol{x}(t_\mathrm{s}) + \left(\boldsymbol{\varphi}_{x_f}' + \nu^T \boldsymbol{r}_{x_f}'\right)\boldsymbol{x}(t_\mathrm{f}) \\ &+ \int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{\psi}_x'[t]\boldsymbol{x}(t)\,\mathrm{d}t - \int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{\lambda}(t)^T\left(\dot{\boldsymbol{x}}(t) - \boldsymbol{f}_x'[t]\boldsymbol{x}(t)\right)\,\mathrm{d}t + \int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{\mu}(t)^T \boldsymbol{c}_x'[t]\boldsymbol{x}(t)\,\mathrm{d}t \end{aligned}$$

for all $\boldsymbol{x} \in \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_x})$. Using the definition of the Hamilton function it can be written as

$$\begin{aligned} 0 = &\left(\boldsymbol{\varphi}'_{x_s} + \nu^T \boldsymbol{r}'_{x_s}\right)\boldsymbol{x}(t_s) + \left(\boldsymbol{\varphi}'_{x_f} + \nu^T \boldsymbol{r}'_{x_f}\right)\boldsymbol{x}(t_f) \\ &+ \int_{t_s}^{t_f} \mathcal{H}'_x[t]\,\boldsymbol{x}(t)\,\mathrm{d}t - \int_{t_s}^{t_f} \boldsymbol{\lambda}(t)^T \dot{\boldsymbol{x}}(t)\,\mathrm{d}t \end{aligned}$$

for all $\boldsymbol{x} \in \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_x})$. Applying Lemma 2.85 it holds for some constant vector C and for all t that

$$C = \boldsymbol{\lambda}(t) - \int_t^{t_f} \mathcal{H}'_x[\tau]^T\,\mathrm{d}\tau.$$

Evaluating this equation at $t = t_f$ yields $C = \boldsymbol{\lambda}(t_f)$ and

$$\boldsymbol{\lambda}(t) = \boldsymbol{\lambda}(t_f) + \int_t^{t_f} \mathcal{H}'_x[\tau]^T\,\mathrm{d}\tau. \tag{8.37}$$

This shows the validity of the adjoint equation. If we apply properties of the Stieltjes integral (see Theorem 2.83) we have

$$\begin{aligned} 0 = &\left(\boldsymbol{\varphi}'_{x_s} + \nu^T \boldsymbol{r}'_{x_s}\right)\boldsymbol{x}(t_s) + \left(\boldsymbol{\varphi}'_{x_f} + \nu^T \boldsymbol{r}'_{x_f}\right)\boldsymbol{x}(t_f) \\ &+ \int_{t_s}^{t_f} \mathcal{H}'_x[t]\,\boldsymbol{x}(t)\,\mathrm{d}t - \int_{t_s}^{t_f} \boldsymbol{\lambda}(t)^T \mathrm{d}\boldsymbol{x}(t). \end{aligned}$$

Integration by parts yields

$$\begin{aligned} 0 = &\left(\boldsymbol{\varphi}'_{x_s} + \nu^T \boldsymbol{r}'_{x_s} + \boldsymbol{\lambda}(t_s)^T\right)\boldsymbol{x}(t_s) + \left(\boldsymbol{\varphi}'_{x_f} + \nu^T \boldsymbol{r}'_{x_f} - \boldsymbol{\lambda}(t_f)^T\right)\boldsymbol{x}(t_f) \\ &+ \int_{t_s}^{t_f} \mathcal{H}'_x[t]\,\boldsymbol{x}(t)\,\mathrm{d}t + \int_{t_s}^{t_f} \boldsymbol{x}(t)^T \mathrm{d}\boldsymbol{\lambda}(t) \end{aligned}$$

for all $\boldsymbol{x} \in \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_x})$. This can be equivalently written as

$$\begin{aligned} 0 = &\left(\boldsymbol{\varphi}'_{x_s} + \nu^T \boldsymbol{r}'_{x_s} + \boldsymbol{\lambda}(t_s)^T\right)\boldsymbol{x}(t_s) + \left(\boldsymbol{\varphi}'_{x_f} + \nu^T \boldsymbol{r}'_{x_f} - \boldsymbol{\lambda}(t_f)^T\right)\boldsymbol{x}(t_f) \\ &+ \int_{t_s}^{t_f} \boldsymbol{x}(t)^T\,\mathrm{d}\left(\boldsymbol{\lambda}(t) - \int_t^{t_f} \mathcal{H}'_x[\tau]^T\,\mathrm{d}\tau\right). \end{aligned}$$

for all $\boldsymbol{x} \in \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_x})$. By a substitution of (8.37) into the last equation we get

$$0 = \left(\boldsymbol{\varphi}'_{x_s} + \nu^T \boldsymbol{r}'_{x_s} + \boldsymbol{\lambda}(t_s)^T\right)\boldsymbol{x}(t_s) + \left(\boldsymbol{\varphi}'_{x_f} + \nu^T \boldsymbol{r}'_{x_f} - \boldsymbol{\lambda}(t_f)^T\right)\boldsymbol{x}(t_f).$$

Since this holds for all variations of $\boldsymbol{x}(\cdot)$ we obtain the transversality conditions (8.34) and (8.35). We can write (8.15) as

$$\begin{aligned} 0 &= \int_{t_s}^{t_f} \boldsymbol{\psi}'_u[t]\,\boldsymbol{u}(t)\,\mathrm{d}t + \int_{t_s}^{t_f} \boldsymbol{\lambda}(t)^T \boldsymbol{f}'_u[t]\,\boldsymbol{u}(t)\,\mathrm{d}t + \int_{t_s}^{t_f} \boldsymbol{\mu}(t)^T \boldsymbol{c}'_u[t]\,\boldsymbol{u}(t)\,\mathrm{d}t \\ &= \int_{t_s}^{t_f} \mathcal{H}'_u[t]\,\boldsymbol{u}(t)\,\mathrm{d}t \end{aligned}$$

for all $\boldsymbol{u} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_u})$. An application of the variational Lemma 2.85 with $\boldsymbol{g} \equiv 0$ yields

$$\mathbf{0} = \int_t^{t_f} \mathcal{H}_u'[\tau]^T \, \mathrm{d}\tau,$$

showing the validity of the Hamilton function stationarity. Next, we investigate the conditions (8.12). It holds

$$\mu^* \in \mathcal{K}^+ \quad \Longleftrightarrow \quad \int_{t_s}^{t_f} \boldsymbol{\mu}(t)^T \boldsymbol{k}(t) \, \mathrm{d}t \geqslant 0 \text{ for } \boldsymbol{k} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_c}),\ \boldsymbol{k}(t) \geqslant 0 \text{ a.e. in } \mathcal{T}.$$

The application of Lemma 2.86 shows that $\boldsymbol{\mu}(t) \geqslant 0$ almost everywhere in $\mathcal{T}$. Thus, the first part of (8.36) holds. The complementarity condition can be written as

$$\mu^*(G(\boldsymbol{x}^*, \boldsymbol{u}^*)) = \mathbf{0} \quad \Longleftrightarrow \quad \int_{t_s}^{t_f} \boldsymbol{\mu}(t)^T \boldsymbol{c}(t, \boldsymbol{x}^*(t), \boldsymbol{u}^*(t)) \, \mathrm{d}t = 0,$$

showing the validity of the second part of (8.36). □

8.4 Regularity Conditions

It is the goal of this section to state conditions such that $l_0 \neq 0$ is satisfied. Before, we formulate a result that follows from Lemma 8.3 and does therefore not require a proof.

Corollary 8.10 (Surjectivity of $H'(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}})$)
Let the rank condition

$$\operatorname{rank}(\boldsymbol{r}_{x_s}' \boldsymbol{\Phi}(t_s) + \boldsymbol{r}_{x_f}' \boldsymbol{\Phi}(t_f)) = n_r$$

be satisfied for the fundamental solution $\boldsymbol{\Phi}(\cdot)$ of the IVP

$$\dot{\boldsymbol{\Phi}}(t) = \boldsymbol{f}_x'[t] \boldsymbol{\Phi}(t), \quad t \in \mathcal{T}, \qquad \boldsymbol{\Phi}(t_s) = \boldsymbol{I}_{n_x}.$$

Then $H'(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}})$ with the functional H from (8.5) is surjective. △

The Mangasarian–Fromowitz constraint qualifications given in Corollary 3.11 provide us with constraint qualifications for OCP 8.2. We exploit the fact that the interior of the cone $\mathcal{K}$ from (8.4) can be written as

$$\operatorname{int}(\mathcal{K}) = \left\{ k \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_c}) : k(t) > \mathbf{0}_{n_c},\ t \in \mathcal{T} \right\}.$$

Lemma 8.11
Let the assumptions of Corollary 8.10 and Theorem 8.9 be fulfilled. Let functions $\boldsymbol{x} \in \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_x})$ and $\boldsymbol{u} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_u})$ exist such that

$$\begin{aligned}
\mathbf{0}_{n_x} &= \boldsymbol{f}_x'[t] \boldsymbol{x}(t) + \boldsymbol{f}_u'[t] \boldsymbol{u}(t) - \dot{\boldsymbol{x}}(t) \quad t \in \mathcal{T}, \\
\mathbf{0}_{n_r} &= \boldsymbol{r}_{x_s}' \boldsymbol{\Phi}(t_s) + \boldsymbol{r}_{x_f}' \boldsymbol{\Phi}(t_f), \\
\mathbf{0}_{n_c} &> \boldsymbol{c}[t] + \boldsymbol{c}_x'[t] \boldsymbol{x}(t) + \boldsymbol{c}_u'[t] \boldsymbol{u}(t) \quad t \in \mathcal{T}.
\end{aligned}$$

Then the Mangasarian–Fromowitz constraint qualification is satisfied and one can choose $l_0 = 1$ in Theorem 8.9. △

Chapter 9

An Interpretation for Discrete Adjoints of Collocation Methods

In Section 5.5.3 we analyzed benefits and drawbacks of both indirect and direct solution methods. As we have seen, one major advantage of using indirect methods is that obtained solution approximations have a high accuracy. Indirect methods do not only include state and control approximations but also costate approximations. Having a good costate approximation helps in verifying the optimality of solutions or in generating meaningful mesh refinement strategies.

Since direct methods do not include equations of the local minimum principle they usually do not approximate the costates. Recently there has been put much effort to estimate costates for many direct methods. There have been developed different approaches: some costate approximation approaches are based on a post–processing step, cf. Martell and Lawton [312], Herman and Conway [230]. As can be seen e.g. in the work of Seywald and Kumar [401], a further approach for approximating costate variables is based on an interpretation of the costate variables as sensitivities connected to the gradient of the cost function. Enright and Conway [146] proposed a method to estimate costates in combination with direct collocation methods, where they used multipliers in the discretized problem associated with the boundary conditions in order to estimate the costate evaluated at the terminal point. A backward integration of the adjoint differential equations with this initial value estimate then provides the costate approximation. Other authors relate the continuous costate dynamics and the KKT conditions associated with the NLP arising from the direct transcription. Here KKT multipliers can be algebraically mapped via a simple calculation to the discrete costates, which result from a certain discretization of the PMP. In the following we briefly provide references for this approach applied with different discretization strategies: Euler discretizations were used by Gerdts [190, Section 5.4], whereas different one–step and multistep integration schemes were investigated by Hager [214]. Cubic collocation at Lobatto points was examined by von Stryk [440]. Recently, strategies have been established involving global orthogonal collocation methods to discretize both the OCP and the PMP. Here, one can use methods based on Legendre–Gauss (LG) collocation points (see e.g. Benson [52], Benson et al. [53]), on Legendre–Gauss–Radau (LGR) collocation points (see e.g. Garg et al. [182]), or on Legendre–Gauss–Lobatto (LGL) collocation points (see e.g. Fahroo and Ross [154], Gong et al. [200]). Local orthogonal collocation methods were used by Kameswaran and Biegler [261, 262], Darby et al. [123] to estimate costates.

The approach we propose in this contribution interrelates the KKT multipliers obtained by a local orthogonal collocation discretization method with associated costates arising from a PMP discretization. Thereby the discretization approach for the PMP equations takes some

insights into account which stem from the novel functional analytic framework that has been developed in the previous chapter. A Petrov–Galerkin approximation is applied to the PMP to transfer the infinite dimensional conditions into a finite dimensional system of equations. The choice of function spaces for test functions and solution functions is motivated by the minimum principle in the function space $\mathcal{Y}(\mathcal{T}, \mathbb{R})$, cf. Section 8.3. Afterwards, we show the equivalence of the equations arising from the Petrov–Galerkin approach and the KKT conditions resulting from the collocation methods. Our approach adopts and extends an idea presented by Beigel [41, Chapter 6]. She could show for the case of ODE IVPs that a certain Petrov–Galerkin approximation of necessary conditions is equivalent to the BDF method and its discrete adjoint IND scheme.

The structure of this chapter looks as follows: the first section introduces a rather general OCP formulation that will be investigated in the remainder of the chapter. In order to facilitate the subsequent analysis the problem with free time is transformed into an equivalent problem with fixed time horizon by means of standard reformulations as they were presented in Section 5.1. Furthermore, Section 9.1 contains some required auxiliary results.

In Section 9.2 we first reformulate the OCP and apply a time transformation rule. Afterwards the problem is discretized using a pseudospectral discretization approach involving Flipped Legendre–Gauss–Radau collocation points. Finally, we derive the KKT conditions of the resulting NLP.

Section 9.3 provides a detailed description of the discretized equations which result from a Petrov–Galerkin transcription of the minimum principle conditions that we found in Chapter 8.

The final section of this chapter deals with the extraction of a mapping between the KKT conditions from Section 9.2 and the discretized first–order necessary conditions from Section 9.3.

9.1 Problem Formulation

Continuous–Time Bolza Optimal Control Problem In this chapter we investigate the OCP with free start and final time from Chapter 7. To this end let sufficiently smooth mappings be given as follows:

$$
\begin{aligned}
\boldsymbol{\varphi} &: \mathbb{R} \times \mathbb{R}^{n_{\mathrm{x}}} \times \mathbb{R} \times \mathbb{R}^{n_{\mathrm{x}}} \longrightarrow \mathbb{R}, \\
\boldsymbol{\psi} &: \mathcal{T} \times \mathbb{R}^{n_{\mathrm{x}}} \times \mathbb{R}^{n_{\mathrm{u}}} \longrightarrow \mathbb{R}, \\
\boldsymbol{f} &: \mathcal{T} \times \mathbb{R}^{n_{\mathrm{x}}} \times \mathbb{R}^{n_{\mathrm{u}}} \longrightarrow \mathbb{R}^{n_{\mathrm{x}}}, \\
\boldsymbol{c} &: \mathcal{T} \times \mathbb{R}^{n_{\mathrm{x}}} \times \mathbb{R}^{n_{\mathrm{u}}} \longrightarrow \mathbb{R}^{n_{\mathrm{c}}}, \\
\boldsymbol{r} &: \mathbb{R} \times \mathbb{R}^{n_{\mathrm{x}}} \times \mathbb{R} \times \mathbb{R}^{n_{\mathrm{x}}} \longrightarrow \mathbb{R}^{n_{\mathrm{r}}}.
\end{aligned}
$$

As usual $\mathcal{T}$ denotes the horizon interval. Based on our reasoning of previous chapters we would like to solve our OCP with state function $\boldsymbol{x}(\cdot)$ and control function $\boldsymbol{u}(\cdot)$ to be chosen in the function spaces $\boldsymbol{x} \in \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_{\mathrm{x}}})$ and $\boldsymbol{u} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_{\mathrm{u}}})$, respectively. We consider the free–time OCP

$$\begin{aligned}
\min_{t_s,t_f,\boldsymbol{x},\boldsymbol{u}} \quad & \varphi(t_s,\boldsymbol{x}(t_s),t_f,\boldsymbol{x}(t_f)) + \int_{t_s}^{t_f} \psi(t,\boldsymbol{x}(t),\boldsymbol{u}(t))\,\mathrm{d}t && (9.1)\\
\text{s.t.} \quad & \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t,\boldsymbol{x}(t),\boldsymbol{u}(t)), \qquad t \in \mathcal{T},\\
& \mathbf{0}_{n_c} \geqslant \boldsymbol{c}(t,\boldsymbol{x}(t),\boldsymbol{u}(t)), \qquad t \in \mathcal{T},\\
& \mathbf{0}_{n_r} = \boldsymbol{r}(t_s,\boldsymbol{x}(t_s),t_f,\boldsymbol{x}(t_f)).
\end{aligned}$$

In the following, we describe briefly the way to go in order to be able to interrelate KKT multipliers arising from a local pseudospectral discretization approach with discretized adjoint variables from the local minimum principle. The goal is achieved by applying a direct method on the one hand, and an indirect method on the other hand.

The direct approach is described in Section 9.1.1 and Section 9.2: in Section 9.1.1 we recall the equivalent reformulation of OCP (9.1) to a scaled OCP (see Section 7.1) taking into consideration the special structure of the function spaces $\mathcal{Y}^1$ and $\mathcal{Y}$. Section 9.2 recalls the NLP arising from a local pseudospectral discretization of the scaled OCP (see Section 7.4) and derives its KKT system.

We investigate the indirect approach in Section 9.1.2 and Section 9.3: first we apply the local minimum principle, which was derived in Chapter 8, to a scaled form of OCP (9.1) in Section 9.1.2. Afterwards, we derive a weak formulation, a well–established technique in the PDE community. This leads to an infinite dimensional variational inequality problem. A discretization of the problem with a Petrov–Galerkin approach in Section 9.3 results in a finite dimensional variational inequality problem, which can be transferred to a nonlinear system of equalities and inequalities.

Section 9.4 is dedicated to assembling the direct and indirect approach: to this end we show the equivalence of the direct approach KKT system and the indirect approach nonlinear system in the following sense: given a solution of one system there exists a mapping such that the mapped solution results in a solution to the other system.

9.1.1 First Discretize, Then Optimize

For the direct approach we consider the following reformulation of OCP (9.1):

$$\begin{aligned}
\min_{t_s,t_f,\boldsymbol{x},\boldsymbol{u}} \quad & \varphi(t_s,\boldsymbol{x}^{(1)}(-1),t_f,\boldsymbol{x}^{(N)}(+1)) + \frac{h}{2}\sum_{n=1}^{N}\frac{h_n}{2}\int_{-1}^{+1}\psi_n(\tau,\boldsymbol{x}^{(n)}(\tau),\boldsymbol{u}^{(n)}(\tau);t_s,t_f)\,\mathrm{d}\tau && (9.2)\\
\text{s.t.} \quad & \dot{\boldsymbol{x}}^{(n)}(\tau) = \frac{h}{2}\frac{h_n}{2}\cdot\boldsymbol{f}_n(\tau,\boldsymbol{x}^{(n)}(\tau),\boldsymbol{u}^{(n)}(\tau);t_s,t_f), && 1 \leqslant n \leqslant N,\ \tau \in [-1,1],\\
& \mathbf{0}_{n_c} \geqslant \boldsymbol{c}_n(\tau,\boldsymbol{x}^{(n)}(\tau),\boldsymbol{u}^{(n)}(\tau);t_s,t_f), && 1 \leqslant n \leqslant N,\ \tau \in [-1,1],\\
& \mathbf{0}_{n_r} = \boldsymbol{r}(t_s,\boldsymbol{x}^{(1)}(-1),t_f,\boldsymbol{x}^{(N)}(+1)),\\
& \mathbf{0}_{n_x} = \boldsymbol{x}^{(n+1)}(-1) - \boldsymbol{x}^{(n)}(+1), && 1 \leqslant n \leqslant N-1.
\end{aligned}$$

The terms are defined in accordance with the ones from OCP (7.7). Note that the fixed 'hidden' temporal grid $-1 = t_0 < t_1 < \ldots < t_N = +1$ in OCP (9.2) is crucial since the interval

boundaries of $\mathcal{T}$ are variables by definition. Hence, it is improper to seek $\boldsymbol{x} \in \mathcal{Y}^1(\mathcal{T}, \mathbb{R}^{n_x})$ and $\boldsymbol{u} \in \mathcal{Y}(\mathcal{T}, \mathbb{R}^{n_u})$. In OCP (9.2) we have overcome the problem and with the aforementioned temporal grid we search solutions with $\boldsymbol{x} \in \mathcal{Y}^1([-1,+1], \mathbb{R}^{n_x})$ and $\boldsymbol{u} \in \mathcal{Y}([-1,+1], \mathbb{R}^{n_u})$. Finally, by applying an additional scaling step we assemble the full horizon trajectories $\boldsymbol{x}(\cdot)$ and $\boldsymbol{u}(\cdot)$ from pieces $\boldsymbol{x}^{(n)} \in \mathcal{C}_b^1([-1,+1], \mathbb{R}^{n_x})$ and $\boldsymbol{u}^{(n)} \in \mathcal{C}_b^0([-1,+1], \mathbb{R}^{n_u})$. Continuity of the state trajectories is enforced by additionally employing matching conditions in OCP (9.2).

9.1.2 First Optimize, Then Discretize

For the indirect approach we consider the following reformulation of OCP (9.1):

$$\begin{aligned}
\min_{t_s, t_f, \boldsymbol{x}, \boldsymbol{u}} \quad & \varphi(t_s, \boldsymbol{x}(-1), t_f, \boldsymbol{x}(+1)) + \frac{h}{2} \int_{-1}^{+1} \psi(\tau, \boldsymbol{x}(\tau), \boldsymbol{u}(\tau); t_s, t_f) \, \mathrm{d}\tau \\
\text{s.t.} \quad & \dot{\boldsymbol{x}}(\tau) = \frac{h}{2} \cdot \boldsymbol{f}(\tau, \boldsymbol{x}(\tau), \boldsymbol{u}(\tau); t_s, t_f), \quad \tau \in [-1,1], \\
& \boldsymbol{0}_{n_c} \geqslant \boldsymbol{c}(\tau, \boldsymbol{x}(\tau), \boldsymbol{u}(\tau); t_s, t_f), \quad \tau \in [-1,1], \\
& \boldsymbol{0}_{n_r} = \boldsymbol{r}(t_s, \boldsymbol{x}(-1), t_f, \boldsymbol{x}(+1)),
\end{aligned} \tag{9.3}$$

The terms are defined in accordance with the ones from OCP (7.5). In contrast, the direct approach we employ $\boldsymbol{x} \in \mathcal{Y}^1([-1,+1], \mathbb{R}^{n_x})$ and $\boldsymbol{u} \in \mathcal{Y}([-1,+1], \mathbb{R}^{n_u})$ without an additional scaling step.

The Local Minimum Principle

For OCP (9.3) we set up the necessary optimality conditions, which have been derived in Chapter 8: to this end, let us consider the associated Hamilton function $\mathcal{H} : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_x} \times \mathbb{R} \times \mathbb{R} \longrightarrow \mathbb{R}$ with

$$\mathcal{H}(t, x, u, \lambda; t_s, t_f) = \psi(t, x, u; t_s, t_f) + \lambda^T \boldsymbol{f}(t, x, u; t_s, t_f)$$

and the augmented Hamilton function $\hat{\mathcal{H}} : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_c} \times \mathbb{R} \times \mathbb{R} \longrightarrow \mathbb{R}$ with

$$\begin{aligned}
\hat{\mathcal{H}}(t, x, u, \lambda, \mu; t_s, t_f) &= \mathcal{H}(t, x, u, \lambda; t_s, t_f) + \mu^T \boldsymbol{c}(t, x, u; t_s, t_f) \\
&= \psi(t, x, u; t_s, t_f) + \lambda^T \boldsymbol{f}(t, x, u; t_s, t_f) + \mu^T \boldsymbol{c}(t, x, u; t_s, t_f).
\end{aligned} \tag{9.4}$$

The PMP can be expressed in terms of $\hat{\mathcal{H}}(\cdot)$ and it states that the following conditions hold:

(i) Adjoint equations: almost everywhere in $[-1,+1]$ it holds

$$\begin{aligned}
\boldsymbol{0}_{n_x} &= \dot{\boldsymbol{\lambda}}(t) + \frac{h}{2} \cdot \hat{\mathcal{H}}_x'(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\lambda}(t), \boldsymbol{\mu}(t); t_s, t_f)^T \\
&= \dot{\boldsymbol{\lambda}}(t) + \frac{h}{2} \cdot \left\{ \psi_x'[t] + \boldsymbol{f}_x'[t] \, \boldsymbol{\lambda}(t) + \boldsymbol{c}_x'[t] \, \boldsymbol{\mu}(t) \right\}.
\end{aligned} \tag{9.5}$$

(ii) Transversality conditions: it holds

$$\mathbf{0}_{n_x} = \boldsymbol{\lambda}(-1) + \boldsymbol{r}'_{x_s}\ \nu + \boldsymbol{\varphi}'_{x_s}, \tag{9.6}$$
$$\mathbf{0}_{n_x} = \boldsymbol{\lambda}(+1) - \boldsymbol{r}'_{x_f}\ \nu - \boldsymbol{\varphi}'_{x_f}, \tag{9.7}$$
$$0 = \hat{\mathcal{H}}[-1] - \boldsymbol{r}'_{t_s}\ \nu - \boldsymbol{\varphi}'_{t_s}, \tag{9.8}$$
$$0 = \hat{\mathcal{H}}[+1] + \boldsymbol{r}'_{t_f}\ \nu + \boldsymbol{\varphi}'_{t_f}. \tag{9.9}$$

(iii) Stationarity of augmented Hamilton function: almost everywhere in $[-1,+1]$ it holds

$$\begin{aligned}\mathbf{0}_{n_u} &= \hat{\mathcal{H}}'_u(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\lambda}(t), \boldsymbol{\mu}(t); t_s, t_f)^T \\ &= \boldsymbol{\psi}'_u[t] + \boldsymbol{f}'_u[t]\,\boldsymbol{\lambda}(t) + \boldsymbol{c}'_u[t]\,\boldsymbol{\mu}(t).\end{aligned} \tag{9.10}$$

(iv) Complementarity condition: almost everywhere in $[-1,+1]$ it holds

$$\mathbf{0}_{n_c} \leqslant \boldsymbol{\mu}(t) \qquad \text{and} \qquad 0 = \boldsymbol{\mu}(t)^T \boldsymbol{c}(t, \boldsymbol{x}(t), \boldsymbol{u}(t); t_s, t_f). \tag{9.11}$$

Additionally, the constraints of OCP (9.3) have to be satisfied:

$$\begin{aligned}\mathbf{0}_{n_x} &= \dot{\boldsymbol{x}}(t) - \frac{h}{2}\,\hat{\mathcal{H}}'_\lambda(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\lambda}(t), \boldsymbol{\mu}(t); t_s, t_f)^T \\ &= \dot{\boldsymbol{x}}(t) - \frac{h}{2}\,\boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t); t_s, t_f),\end{aligned} \tag{9.12}$$
$$\mathbf{0}_{n_c} \geqslant \boldsymbol{c}(t, \boldsymbol{x}(t), \boldsymbol{u}(t); t_s, t_f), \tag{9.13}$$
$$\mathbf{0}_{n_r} = \boldsymbol{r}(t_s, \boldsymbol{x}(-1), t_f, \boldsymbol{x}(+1)). \tag{9.14}$$

Weak Formulation

Now we formulate variational equalities and inequalities from the PMP conditions. We have described a similar concept involving trial and test functions in Section 6.3.2, namely the weighted residual method.

Weak Form $[(9.5)+(9.6)+(9.7)]$ The variational formulation for the adjoint equation (9.5) is given as

$$\begin{aligned}0 &= \int_{-1}^{+1} \boldsymbol{\varphi}_x{}^T(t) \left\{ \dot{\boldsymbol{\lambda}}(t) + \frac{h}{2}\,\hat{\mathcal{H}}'_x(t, \boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\lambda}(t), \boldsymbol{\mu}(t); t_s, t_f)^T \right\} \mathrm{d}t \\ &= \frac{h}{2} \int_{-1}^{+1} \boldsymbol{\varphi}_x{}^T(t)\, \boldsymbol{\psi}'_x[t]\, \mathrm{d}t + \int_{-1}^{+1} \boldsymbol{\varphi}_x{}^T(t) \left\{ \dot{\boldsymbol{\lambda}}(t) + \frac{h}{2}\,\boldsymbol{f}'_x[t]\,\boldsymbol{\lambda}(t) \right\} \mathrm{d}t \\ &\quad + \frac{h}{2} \int_{-1}^{+1} \boldsymbol{\varphi}_x{}^T(t)\, \boldsymbol{c}'_x[t]\, \boldsymbol{\mu}(t)\, \mathrm{d}t, \qquad \forall \boldsymbol{\varphi}_x \in \mathcal{Y}^1([-1,+1], \mathbb{R}^{n_x}).\end{aligned} \tag{9.15}$$

Using partial integration we reformulate Equation (9.15) as

$$\begin{aligned}
0 =& \frac{h}{2}\int_{-1}^{+1} {\varphi_x}^T(t)\,\psi_x'[t]\,\mathrm{d}t + \int_{-1}^{+1} {\varphi_x}^T(t)\,\dot{\boldsymbol{\lambda}}(t)\,\mathrm{d}t + \frac{h}{2}\int_{-1}^{+1} {\varphi_x}^T(t)\,f_x'[t]\boldsymbol{\lambda}(t)\,\mathrm{d}t \\
&+ \frac{h}{2}\int_{-1}^{+1} {\varphi_x}^T(t)\,c_x'[t]\boldsymbol{\mu}(t)\,\mathrm{d}t \\
=& \frac{h}{2}\int_{-1}^{+1} {\varphi_x}^T(t)\,\psi_x'[t]\,\mathrm{d}t + \int_{-1}^{+1} \boldsymbol{\lambda}^T(t)\left\{\frac{h}{2}f_x'[t]\,\varphi_x(t) - \dot{\varphi}_x(t)\right\}\mathrm{d}t \\
&+ \frac{h}{2}\int_{-1}^{+1} {\varphi_x}^T(t)\,c_x'[t]\boldsymbol{\mu}(t)\,\mathrm{d}t \\
&+ {\varphi_x}^T(+1)\,\boldsymbol{\lambda}(+1) - {\varphi_x}^T(-1)\,\boldsymbol{\lambda}(-1), \qquad \forall \varphi_x \in \mathcal{Y}^1([-1,+1],\mathbb{R}^{n_x}).
\end{aligned}$$

By means of the *identity function* $\mathbf{id} : [-1,+1] \longrightarrow [-1,+1]$, $t \mapsto \mathbf{id}(t) = t$ and the Lebesgue–Stieltjes integral we write this equation as

$$\begin{aligned}
0 =& \frac{h}{2}\int_{-1}^{+1} \psi_x'[t]\,\varphi_x(t)\,\mathrm{d}\,\mathbf{id}(t) + \int_{-1}^{+1} \frac{h}{2}f_x'[t]\,\varphi_x(t) - \dot{\varphi}_x(t)\,\mathrm{d}\,\boldsymbol{\Lambda}(t) \\
&+ \frac{h}{2}\int_{-1}^{+1} c_x'[t]\,\varphi_x(t)\,\mathrm{d}\mathbf{M}(t) \\
&+ {\varphi_x}^T(+1)\,\boldsymbol{\lambda}(+1) - {\varphi_x}^T(-1)\,\boldsymbol{\lambda}(-1), \qquad \forall \varphi_x \in \mathcal{Y}^1([-1,+1],\mathbb{R}^{n_x}).
\end{aligned}$$

Here $\boldsymbol{\Lambda}(\cdot) \in \mathcal{NBV}([-1,+1],\mathbb{R}^{n_x})$ and $\boldsymbol{M}(\cdot) \in \mathcal{NBV}([-1,+1],\mathbb{R}^{n_c})$ are given in terms of the adjoint solutions $\boldsymbol{\lambda}(\cdot)$ and $\boldsymbol{\mu}(\cdot)$ as

$$\boldsymbol{\Lambda}(t) = \int_{-1}^{t} \boldsymbol{\lambda}(\tau)\,\mathrm{d}\tau \qquad \text{and} \qquad \boldsymbol{M}(t) = \int_{-1}^{t} \boldsymbol{\mu}(\tau)\,\mathrm{d}\tau, \tag{9.16}$$

with componentwise integration. We introduce the half–open intervals $\mathcal{I}_n \stackrel{\text{def}}{=} (t_{n-1}, t_n]$ for $1 \leqslant n \leqslant N$ such that we can split the interval $[-1,+1]$ according to the partition encoded in the $\mathcal{Y}^k$, $k = 1,2$, function spaces as

$$[-1,+1] = \{t_s\} \cup \mathcal{I}_1 \cup \ldots \cup \mathcal{I}_N.$$

Hence, we can write the necessary condition as

$$\begin{aligned}
0 =& \sum_{n=1}^{N}\left\{\frac{h}{2}\int_{\mathcal{I}_n} \psi_x'[t]\,\varphi_x(t)\,\mathrm{d}\,\mathbf{id}(t) + \int_{\mathcal{I}_n} \frac{h}{2}f_x'[t]\,\varphi_x(t) - \dot{\varphi}_x(t)\,\mathrm{d}\,\boldsymbol{\Lambda}(t)\right. \\
&\left. + \frac{h}{2}\int_{\mathcal{I}_n} c_x'[t]\,\varphi_x(t)\,\mathrm{d}\mathbf{M}(t)\right\} \\
&+ {\varphi_x}^T(+1)\,\boldsymbol{\lambda}(+1) - {\varphi_x}^T(-1)\,\boldsymbol{\lambda}(-1), \qquad \forall \varphi_x \in \mathcal{Y}^1([-1,+1],\mathbb{R}^{n_x}).
\end{aligned} \tag{9.17}$$

In order to transform the occurring integral boundaries $\mathcal{I}_n$ in (9.17) to the normalized interval $(-1,1]$ we apply the following auxiliary result:

Lemma 9.1 (Substitution Rule for Stieltjes Integrals)
Let $\varphi : [a,b] \longrightarrow \mathbb{R}$ and $f : [\varphi(a),\varphi(b)] \longrightarrow \mathbb{R}$ be continuous and let $g : [\varphi(a),\varphi(b)] \longrightarrow \mathbb{R}$ be of bounded variation. Then it holds

$$\int_a^b f(\varphi(t))\,\mathrm{d}g(\varphi(t)) = \int_{\varphi(a)}^{\varphi(b)} f(t)\,\mathrm{d}g(t). \qquad \triangle$$

Proof See e.g. Falkner and Teschl [155]. □

Let us consider the linear time transformation mappings $\boldsymbol{t}_n : [-1,1] \longrightarrow \overline{\mathcal{I}}_n$, $1 \leqslant n \leqslant N$, which are defined as

$$\boldsymbol{t}_n(t) \stackrel{\text{def}}{=} \frac{t_n + t_{n-1}}{2} + t \cdot \frac{t_n - t_{n-1}}{2}.$$

Integration by substitution for the Lebesgue–Stieltjes integral allows us to normalize the integration boundaries of (9.17). Together with employing (9.6) and (9.7) this yields

$$\begin{aligned} 0 = \sum_{n=1}^{N} \Bigg\{ & \frac{h}{2} \int_{(-1,1]} \psi_x'[\boldsymbol{t}_n(t)]\,\varphi_x(\boldsymbol{t}_n(t))\,\mathrm{d}\,\boldsymbol{t}_n(t) + \int_{(-1,1]} \frac{h}{2} f_x'[\boldsymbol{t}_n(t)]\,\varphi_x(\boldsymbol{t}_n(t)) - \dot{\varphi}_x(\boldsymbol{t}_n(t))\,\mathrm{d}\,\boldsymbol{\Lambda}(\boldsymbol{t}_n(t)) \\ & + \frac{h}{2} \int_{(-1,1]} c_x'[\boldsymbol{t}_n(t)]\varphi_x(\boldsymbol{t}_n(t))\,\mathrm{d}\,\mathbf{M}(\boldsymbol{t}_n(t)) \Bigg\} \\ & + {\varphi_x}^T(+1)\left(r_{x_f}'\ \nu + \varphi_{x_f}'\right) + {\varphi_x}^T(-1)\left(r_{x_s}'\ \nu + \varphi_{x_s}'\right), \quad \forall \varphi_x \in \mathcal{Y}^1([-1,+1],\mathbb{R}^{n_x}). \end{aligned} \tag{9.18}$$

Weak Form $[(9.8)+(9.9)]$ The following auxiliary result provides us with an alternative representation for the transversality conditions (9.8)+(9.9).

Lemma 9.2
Let the augmented Hamilton function $\hat{\mathcal{H}}(\cdot)$ be defined according to (9.4). Then $\hat{\mathcal{H}}[-1]$ and $\hat{\mathcal{H}}[+1]$ can be expressed as

$$\begin{aligned} \hat{\mathcal{H}}[-1] &= -\frac{t_f - t_s}{2} \int_{-1}^{+1} \hat{\mathcal{H}}_{t_s}'[\tau]\,\mathrm{d}\tau + \frac{1}{2}\int_{-1}^{+1} \hat{\mathcal{H}}[\tau]\,\mathrm{d}\tau, \\ \hat{\mathcal{H}}[+1] &= +\frac{t_f - t_s}{2} \int_{-1}^{+1} \hat{\mathcal{H}}_{t_f}'[\tau]\,\mathrm{d}\tau + \frac{1}{2}\int_{-1}^{+1} \hat{\mathcal{H}}[\tau]\,\mathrm{d}\tau. \end{aligned} \qquad \triangle$$

Proof See Appendix A.2. □

Using this lemma the transversality conditions (9.8) and (9.9) can be alternatively written as

$$0 = -\frac{1}{2}\int_{-1}^{+1} \hat{\mathcal{H}}[t]\,\mathrm{d}t + \frac{h}{2}\int_{-1}^{+1} \hat{\mathcal{H}}_{t_s}'[t]\,\mathrm{d}t + \varphi_{t_s}' + r_{t_s}'\ \nu, \tag{9.19}$$

$$0 = -\frac{1}{2}\int_{-1}^{+1} \hat{\mathcal{H}}[t]\,\mathrm{d}t - \frac{h}{2}\int_{-1}^{+1} \hat{\mathcal{H}}_{t_f}'[t]\,\mathrm{d}t - \varphi_{t_f}' - r_{t_f}'\ \nu\,. \tag{9.20}$$

The weak formulations of (9.19)+(9.20) are obtained as

$$0 = \boldsymbol{\varphi}_{t_s}{}^T \left\{ -\frac{1}{2} \int_{-1}^{+1} \hat{\mathcal{H}}[t] \, \mathrm{d}t + \frac{h}{2} \int_{-1}^{+1} \hat{\mathcal{H}}'_{t_s}[t] \, \mathrm{d}t + \boldsymbol{\varphi}'_{t_s} + \boldsymbol{r}'_{t_s} \, \boldsymbol{\nu} \right\}, \quad \forall \, \boldsymbol{\varphi}_{t_s} \in \mathbb{R}, \tag{9.21}$$

$$0 = \boldsymbol{\varphi}_{t_f}{}^T \left\{ -\frac{1}{2} \int_{-1}^{+1} \hat{\mathcal{H}}[t] \, \mathrm{d}t - \frac{h}{2} \int_{-1}^{+1} \hat{\mathcal{H}}'_{t_f}[t] \, \mathrm{d}t - \boldsymbol{\varphi}'_{t_f} - \boldsymbol{r}'_{t_f} \, \boldsymbol{\nu} \right\}, \quad \forall \, \boldsymbol{\varphi}_{t_f} \in \mathbb{R}. \tag{9.22}$$

Similarly to the variational formulation for the adjoint equation we introduce the Lebesgue–Stieltjes integral and apply an interval transformation step to (9.21) which yields

$$\begin{aligned}
0 =& \boldsymbol{\varphi}_{t_s}{}^T \left\{ -\frac{1}{2} \sum_{n=1}^{N} \left[\int_{\mathcal{I}_n} \boldsymbol{\psi}[t] \, \mathrm{d}\,\mathbf{id}(t) + \int_{\mathcal{I}_n} \boldsymbol{f}[t] \, \mathrm{d}\,\boldsymbol{\Lambda}(t) + \int_{\mathcal{I}_n} \boldsymbol{c}[t] \, \mathrm{d}\,\mathbf{M}(t) \right] \right. \\
& + \frac{h}{2} \sum_{n=1}^{N} \left[\int_{\mathcal{I}_n} \boldsymbol{\psi}'_{t_s}[t] \, \mathrm{d}\,\mathbf{id}(t) + \int_{\mathcal{I}_n} \boldsymbol{f}'_{t_s}[t] \, \mathrm{d}\,\boldsymbol{\Lambda}(t) + \int_{\mathcal{I}_n} \boldsymbol{c}'_{t_s}[t] \, \mathrm{d}\,\mathbf{M}(t) \right] \\
& \left. + \boldsymbol{\varphi}'_{t_s} + \boldsymbol{r}'_{t_s} \, \boldsymbol{\nu} \right\} \\
=& \boldsymbol{\varphi}_{t_s}{}^T \left\{ -\frac{1}{2} \sum_{n=1}^{N} \left[\int_{(-1,1]} \boldsymbol{\psi}[\boldsymbol{t}_n(t)] \, \mathrm{d}\,\mathbf{id}(\boldsymbol{t}_n(t)) + \int_{(-1,1]} \boldsymbol{f}[\boldsymbol{t}_n(t)] \, \mathrm{d}\,\boldsymbol{\Lambda}(\boldsymbol{t}_n(t)) \right] \right. \\
& + \frac{h}{2} \sum_{n=1}^{N} \left[\int_{(-1,1]} \boldsymbol{\psi}'_{t_s}[\boldsymbol{t}_n(t)] \, \mathrm{d}\,\mathbf{id}(\boldsymbol{t}_n(t)) + \int_{(-1,1]} \boldsymbol{f}'_{t_s}[\boldsymbol{t}_n(t)] \, \mathrm{d}\,\boldsymbol{\Lambda}(\boldsymbol{t}_n(t)) \right. \\
& \left. \left. + \int_{(-1,1]} \boldsymbol{c}'_{t_s}[\boldsymbol{t}_n(t)] \, \mathrm{d}\,\mathbf{M}(\boldsymbol{t}_n(t)) \right] + \boldsymbol{\varphi}'_{t_s} + \boldsymbol{r}'_{t_s} \, \boldsymbol{\nu} \right\}, \quad \forall \, \boldsymbol{\varphi}_{t_s} \in \mathbb{R}.
\end{aligned} \tag{9.23}$$

Note that we exploited the complementarity condition applied to the term $\int_{\mathcal{I}_n} \boldsymbol{c}[t] \, \mathrm{d}\,\mathbf{M}(t)$ which vanishes. Repeating the same step with (9.22) leads to

$$\begin{aligned}
0 =& \boldsymbol{\varphi}_{t_f}{}^T \left\{ \frac{1}{2} \sum_{n=1}^{N} \left[\int_{\mathcal{I}_n} \boldsymbol{\psi}[t] \, \mathrm{d}\,\mathbf{id}(t) + \int_{\mathcal{I}_n} \boldsymbol{f}[t] \, \mathrm{d}\,\boldsymbol{\Lambda}(t) + \int_{\mathcal{I}_n} \boldsymbol{c}[t] \, \mathrm{d}\,\mathbf{M}(t) \right] \right. \\
& + \frac{h}{2} \sum_{n=1}^{N} \left[\int_{\mathcal{I}_n} \boldsymbol{\psi}'_{t_f}[t] \, \mathrm{d}\,\mathbf{id}(t) + \int_{\mathcal{I}_n} \boldsymbol{f}'_{t_f}[t] \, \mathrm{d}\,\boldsymbol{\Lambda}(t) + \int_{\mathcal{I}_n} \boldsymbol{c}'_{t_f}[t] \, \mathrm{d}\,\mathbf{M}(t) \right] \\
& \left. + \boldsymbol{\varphi}'_{t_f} + \boldsymbol{r}'_{t_f} \, \boldsymbol{\nu} \right\} \\
=& \boldsymbol{\varphi}_{t_f}{}^T \left\{ \frac{1}{2} \sum_{n=1}^{N} \left[\int_{(-1,1]} \boldsymbol{\psi}[\boldsymbol{t}_n(t)] \, \mathrm{d}\,\mathbf{id}(\boldsymbol{t}_n(t)) + \int_{(-1,1]} \boldsymbol{f}[\boldsymbol{t}_n(t)] \, \mathrm{d}\,\boldsymbol{\Lambda}(\boldsymbol{t}_n(t)) \right] \right. \\
& + \frac{h}{2} \sum_{n=1}^{N} \left[\int_{(-1,1]} \boldsymbol{\psi}'_{t_f}[\boldsymbol{t}_n(t)] \, \mathrm{d}\,\mathbf{id}(\boldsymbol{t}_n(t)) + \int_{(-1,1]} \boldsymbol{f}'_{t_f}[\boldsymbol{t}_n(t)] \, \mathrm{d}\,\boldsymbol{\Lambda}(\boldsymbol{t}_n(t)) \right.
\end{aligned}$$

$$+\int_{(-1,1]} \boldsymbol{c}'_{t_\mathrm{f}}[\boldsymbol{t}_n(t)]\,\mathrm{d}\mathbf{M}(\boldsymbol{t}_n(t))\Big] + \boldsymbol{\varphi}'_{t_\mathrm{f}} + \boldsymbol{r}'_{t_\mathrm{f}}\,\nu\Big\}, \quad \forall\,\boldsymbol{\varphi}_{t_\mathrm{f}} \in \mathbb{R}. \tag{9.24}$$

Weak Form [(9.10)] The variational approach for the Hamilton stationarity condition (9.10) reads as

$$\begin{aligned} 0 = &\int_{-1}^{+1} {\boldsymbol{\varphi}_u}^T(t)\,\boldsymbol{\psi}'_u[t]\,\mathrm{d}t + \int_{-1}^{+1} {\boldsymbol{\varphi}_u}^T(t)\,\boldsymbol{f}'_u[t]\,\boldsymbol{\lambda}(t)\,\mathrm{d}t \\ &+ \int_{-1}^{+1} {\boldsymbol{\varphi}_u}^T(t)\,\boldsymbol{c}'_u[t]\,\boldsymbol{\mu}(t)\,\mathrm{d}t, \qquad \forall \boldsymbol{\varphi}_u \in \mathcal{Y}([-1,+1],\mathbb{R}^{n_\mathrm{u}}), \end{aligned}$$

such that introducing the Lebesgue–Stieltjes integral results in the formulation

$$\begin{aligned} 0 = &\int_{-1}^{+1} \boldsymbol{\psi}'_u[t]\,\boldsymbol{\varphi}_u(t)\,\mathrm{d}\,\mathbf{id}(t) + \int_{-1}^{+1} \boldsymbol{f}'_u[t]\,\boldsymbol{\varphi}_u(t)\,\mathrm{d}\,\boldsymbol{\Lambda}(t) \\ &- \int_{-1}^{+1} \boldsymbol{c}'_u[t]\,\boldsymbol{\varphi}_u(t)\,\mathrm{d}\mathbf{M}(t), \qquad \forall \boldsymbol{\varphi}_u \in \mathcal{Y}([-1,+1],\mathbb{R}^{n_\mathrm{u}}). \end{aligned}$$

By splitting the horizon interval according to the $\mathcal{Y}$ function space we can write the equation as

$$\begin{aligned} 0 = &\sum_{n=1}^{N} \Big\{ \int_{\mathcal{I}_n} \boldsymbol{\psi}'_u[t]\,\boldsymbol{\varphi}_u(t)\,\mathrm{d}\,\mathbf{id}(t) + \int_{\mathcal{I}_n} \boldsymbol{f}'_u[t]\,\boldsymbol{\varphi}_u(t)\,\mathrm{d}\,\boldsymbol{\Lambda}(t) \\ &+ \int_{\mathcal{I}_n} \boldsymbol{c}'_u[t]\,\boldsymbol{\varphi}_u(t)\,\mathrm{d}\mathbf{M}(t) \Big\}, \quad \forall \boldsymbol{\varphi}_u \in \mathcal{Y}([-1,+1],\mathbb{R}^{n_\mathrm{u}}). \end{aligned}$$

Finally, by applying a normalization of the interval length we end up with

$$\begin{aligned} 0 = &\sum_{n=1}^{N} \Big\{ \int_{(-1,1]} \boldsymbol{\psi}'_u[\boldsymbol{t}_n(t)]\,\boldsymbol{\varphi}_u(\boldsymbol{t}_n(t))\,\mathrm{d}\,\boldsymbol{t}_n(t) + \int_{(-1,1]} \boldsymbol{f}'_u[\boldsymbol{t}_n(t)]\,\boldsymbol{\varphi}_u(\boldsymbol{t}_n(t))\,\mathrm{d}\,\boldsymbol{\Lambda}(\boldsymbol{t}_n(t)) \\ &+ \int_{(-1,1]} \boldsymbol{c}'_u[\boldsymbol{t}_n(t)]\,\boldsymbol{\varphi}_u(\boldsymbol{t}_n(t))\,\mathrm{d}\mathbf{M}(\boldsymbol{t}_n(t)) \Big\}, \quad \forall \boldsymbol{\varphi}_u \in \mathcal{Y}([-1,+1],\mathbb{R}^{n_\mathrm{u}}). \end{aligned} \tag{9.25}$$

Weak Form [(9.11) + (9.13)] We start with the weak formulation for the inequality in (9.11) and find

$$0 \leqslant \int_{-1}^{+1} {\boldsymbol{\varphi}_\mu}^T(t)\,\boldsymbol{\mu}(t)\,\mathrm{d}t, \quad \forall \boldsymbol{\varphi}_\mu \in \mathcal{Y}([-1,+1],\mathbb{R}^{n_\mathrm{c}}),\ \boldsymbol{\varphi}_\mu \geqslant \mathbf{0}_{n_\mathrm{c}}. \tag{9.26}$$

Thus we can calculate

$$0 \leqslant \int_{-1}^{+1} \varphi_\mu(t)\,\mathrm{d}\mathbf{M}(t) = \sum_{n=1}^{N} \int_{\mathcal{I}_n} \varphi_\mu(t)\,\mathrm{d}\mathbf{M}(t)$$
$$= \sum_{n=1}^{N} \int_{(-1,1]} \varphi_\mu(\boldsymbol{t}_n(t))\,\mathrm{d}\mathbf{M}(\boldsymbol{t}_n(t)), \quad \forall \varphi_\mu \in \mathcal{Y}([-1,+1],\mathbb{R}^{n_c}),\ \varphi_\mu \geqslant \mathbf{0}_{n_c}. \tag{9.27}$$

The weak formulation of the equality in (9.11) can be expressed as

$$0 = \int_{-1}^{+1} \varphi_M(t)\,\boldsymbol{c}(t,\boldsymbol{x}(t),\boldsymbol{u}(t);t_s,t_f)\,\mathrm{d}\mathbf{M}(t), \qquad \forall \varphi_M \in \mathcal{Y}([-1,+1],\mathbb{R}).$$

Splitting the horizon interval and normalizing the single interval leads to

$$0 = \sum_{n=1}^{N} \int_{\mathcal{I}_n} \varphi_M(t)\,\boldsymbol{c}(t,\boldsymbol{x}(t),\boldsymbol{u}(t);t_s,t_f)\,\mathrm{d}\mathbf{M}(t)$$
$$= \sum_{n=1}^{N} \int_{(-1,+1]} \varphi_M(\boldsymbol{t}_n(t))\,\boldsymbol{c}[\boldsymbol{t}_n(t)]\,\mathrm{d}\mathbf{M}(\boldsymbol{t}_n(t)) \qquad \forall \varphi_M \in \mathcal{Y}([-1,+1],\mathbb{R}) \tag{9.28}$$

Likewise we proceed with (9.13):

$$0 \geqslant \int_{-1}^{+1} \boldsymbol{c}(t,\boldsymbol{x}(t),\boldsymbol{u}(t);t_s,t_f)\,\mathrm{d}\,\varphi_c(t), \quad \forall \varphi_c \in \mathcal{NBV}([-1,+1],\mathbb{R}^{n_c}),\ \varphi_c \geqslant \mathbf{0}_{n_c}.$$

Hence, we end up with

$$0 \geqslant \sum_{n=1}^{N} \int_{\mathcal{I}_n} \boldsymbol{c}(t,\boldsymbol{x}(t),\boldsymbol{u}(t);t_s,t_f)\,\mathrm{d}\,\varphi_c(t)$$
$$= \sum_{n=1}^{N} \int_{(-1,1]} \boldsymbol{c}[\boldsymbol{t}_n(t)]\,\mathrm{d}\,\varphi_c(\boldsymbol{t}_n(t)), \quad \forall \varphi_c \in \mathcal{NBV}([-1,+1],\mathbb{R}^{n_c}),\ \varphi_c \geqslant \mathbf{0}_{n_c}. \tag{9.29}$$

Weak Form [(9.12)] The weak formulation of differential equation (9.12) looks as follows:

$$0 = \int_{-1}^{+1} {\varphi_\lambda}^T(t)\left\{\dot{\boldsymbol{x}}(t) - \frac{h}{2}\boldsymbol{f}(t,\boldsymbol{x}(t),\boldsymbol{u}(t);t_s,t_f)\right\}\,\mathrm{d}t, \quad \forall \varphi_\lambda \in \mathcal{NBV}([-1,+1],\mathbb{R}^{n_x}). \tag{9.30}$$

By means of the Lebesgue–Stieltjes integral we write (9.30) as

$$0 = \int_{-1}^{+1} \dot{\boldsymbol{x}}(t) - \frac{h}{2}\boldsymbol{f}(t,\boldsymbol{x}(t),\boldsymbol{u}(t);t_s,t_f)\,\mathrm{d}\,\varphi_\Lambda(t), \quad \forall \varphi_\Lambda \in \mathcal{NBV}([-1,+1],\mathbb{R}^{n_x}).$$

Splitting the integral and normalizing the integral bounds reads as

$$\begin{aligned} 0 &= \sum_{n=1}^{N} \int_{\mathcal{I}_n} \dot{\boldsymbol{x}}(t) - \frac{h}{2} \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) \, \mathrm{d}\, \boldsymbol{\varphi}_\Lambda(t) \\ &= \sum_{n=1}^{N} \int_{(-1,1]} \dot{\boldsymbol{x}}(\boldsymbol{t}_n(t)) - \frac{h}{2} \boldsymbol{f}[\boldsymbol{t}_n(t)] \, \mathrm{d}\, \boldsymbol{\varphi}_\Lambda(\boldsymbol{t}_n(t)), \quad \forall \boldsymbol{\varphi}_\Lambda \in \mathcal{NBV}([-1,+1], \mathbb{R}^{n_x}). \end{aligned} \tag{9.31}$$

Weak Form [(9.14)] The weak formulation of (9.14) is given by

$$0 = {\varphi_r}^T \boldsymbol{r}(t_s, \boldsymbol{x}(-1), t_f, \boldsymbol{x}(+1)), \quad \forall \varphi_r \in \mathbb{R}^{n_r}. \tag{9.32}$$

The Mapping Theorem

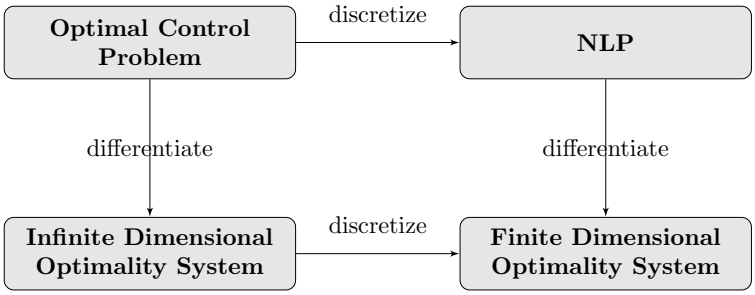


Figure 9.1: In the figure there are the two ways depicted how we discretize the OCP (9.1). In the first way, we employ a collocation method which results in a NLP (arrow to the right) whose KKT form a finite dimensional equation system. For the second way, we determine the weak formulation of the local minimum principle equations (arrow to the bottom) and discretize them with a Petrov–Galerkin method leading to another finite dimensional equation system. Both equation systems are equivalent in the sense described in Theorem 9.3.

We state the main result of this chapter in terms of the following theorem.

Theorem 9.3
Let OCP (9.1) be discretized by means of a local collocation method (see Section 7.4) and let the KKT system of the resulting NLP be given as $\boldsymbol{F}_1(w_1) = \boldsymbol{0}$. Let the weak formulation (Equations (9.18), (9.23), (9.24), (9.25), (9.28), (9.29), (9.31), and (9.32)) of the optimality system (Equations (9.5)–(9.14)) to OCP (9.1) be discretized with the help of a tailored Petrov–Galerkin discretization approach (see Section 9.3) and let the resulting equation system be given as $\boldsymbol{F}_2(w_2) = \boldsymbol{0}$. Then both systems are equivalent in the sense that for a solution w_1^* of the first system (i.e., it holds $\boldsymbol{F}_1(w_1^*) = 0$) there exists a mapping $\boldsymbol{T}_{12}(\cdot)$ such that $\boldsymbol{F}_2(\boldsymbol{T}_{12}(w_1^*)) = 0$, i.e., $\boldsymbol{T}_{12}(\cdot)$ maps w_1^* to a solution of the second system. The same statement holds also for the other direction. △

The proof of Theorem 9.3 follows straight from the calculations of Section 9.2 and Section 9.3. Moreover, we are even able to specify the mappings which transform a solution of one system

to a solution of the other. Figure 9.1 illustrates how the collocation approach and the Petrov–Galerkin approach work starting from the OCP.

9.1.3 Auxiliary Results

In order to be able to prove the mapping theorem we need some supplementary material. To this end let us consider the two Lagrange polynomials

$$\boldsymbol{L}_j(t) \overset{\text{def}}{=} \prod_{\substack{i=0 \\ i\neq j}}^{K} \frac{t-t_i}{t_j-t_i}, \quad \deg\left(\boldsymbol{L}_j\right) = K, \qquad \deg\left(\dot{\boldsymbol{L}}_j\right) = K-1, \quad 0 \leqslant j \leqslant K, \tag{9.33}$$

$$\overline{\boldsymbol{L}}_j(t) \overset{\text{def}}{=} \prod_{\substack{i=1 \\ i\neq j}}^{K} \frac{t-t_i}{t_j-t_i}, \quad \deg\left(\overline{\boldsymbol{L}}_j\right) = K-1, \quad \deg\left(\dot{\overline{\boldsymbol{L}}}_j\right) = K-2, \quad 1 \leqslant j \leqslant K. \tag{9.34}$$

We also introduce the first–order differentiation matrices D and $\overline{D}$ whose entries are determined according to

$$D_{j,i} = \dot{\boldsymbol{L}}_i(t_j), \qquad \overline{D}_{j,i} = \dot{\overline{\boldsymbol{L}}}_i(t_j). \tag{9.35}$$

If the $\{\omega_j\}_{1\leqslant j\leqslant K}$ denote quadrature weights belonging to FLGR quadrature we can prove the following lemma:

Lemma 9.4
Let the Lagrange polynomials $\boldsymbol{L}_j(\cdot)$, $0 \leqslant j \leqslant K$, and $\overline{\boldsymbol{L}}_j(\cdot)$, $1 \leqslant j \leqslant K$, be defined according to (9.33) and (9.34) with $t_0 = -1$ and FLGR points $\{t_j\}_{1\leqslant j\leqslant K}$. Then it holds

$$\begin{aligned} D_{j,i} &= -\frac{\omega_i}{\omega_j}\overline{D}_{i,j}, && i,j \in [K],\ i \neq j, \\ D_{j,j} &= -\overline{D}_{j,j}, && j \in [K-1], \\ D_{K,K} &= -\overline{D}_{K,K} + \frac{1}{\omega_K}, && j = K. \end{aligned}$$

△

Proof Let $\boldsymbol{P} : [-1,+1] \longrightarrow \mathbb{R}$ be a polynomial of degree K and let $\boldsymbol{Q} : [-1,+1] \longrightarrow \mathbb{R}$ be a polynomial of degree $K-1$. Then they have representations

$$\begin{aligned} \boldsymbol{P}(t) &= \sum_{j=0}^{K} \boldsymbol{P}(t_j)\boldsymbol{L}_j(t), & \boldsymbol{Q}(t) &= \sum_{j=1}^{K} \boldsymbol{Q}(t_j)\overline{\boldsymbol{L}}_j(t), \\ \dot{\boldsymbol{P}}(t) &= \sum_{j=0}^{K} \boldsymbol{P}(t_j)\dot{\boldsymbol{L}}_j(t), & \dot{\boldsymbol{Q}}(t) &= \sum_{j=1}^{K} \boldsymbol{Q}(t_j)\dot{\overline{\boldsymbol{L}}}_j(t). \end{aligned}$$

Furthermore it holds

$$\dot{\boldsymbol{P}}(t)\boldsymbol{Q}(t) = \sum_{j=0}^{K}\sum_{i=1}^{K} \boldsymbol{P}(t_j)\boldsymbol{Q}(t_i)\dot{\boldsymbol{L}}_j(t)\overline{\boldsymbol{L}}_i(t),$$

$$\boldsymbol{P}(t)\,\dot{\boldsymbol{Q}}(t) = \sum_{j=0}^{K}\sum_{i=1}^{K} \boldsymbol{P}(t_j)\,\boldsymbol{Q}(t_i)\,\boldsymbol{L}_j(t)\,\dot{\overline{\boldsymbol{L}}}_i(t).$$

Both, $\dot{\boldsymbol{P}}\boldsymbol{Q}$ and $\boldsymbol{P}\dot{\boldsymbol{Q}}$ are polynomials of degree $2K-2$. Hence, the Gauss–Radau quadrature is exact for these polynomials. We calculate

$$\begin{aligned}\int_{-1}^{+1} \dot{\boldsymbol{P}}(t)\,\boldsymbol{Q}(t)\,\mathrm{d}t &= \sum_{k=1}^{K}\omega_k \sum_{l=0}^{K}\sum_{i=1}^{K} \boldsymbol{P}(t_l)\,\boldsymbol{Q}(t_i)\,\dot{\boldsymbol{L}}_l(t_k)\,\overline{\boldsymbol{L}}_i(t_k) = \sum_{k=1}^{K}\sum_{l=0}^{K} \omega_k \boldsymbol{P}(t_l)\,\boldsymbol{Q}(t_k)\,\dot{\boldsymbol{L}}_l(t_k)\\ &= \sum_{k=1}^{K-1}\sum_{l=0}^{K} \omega_k \boldsymbol{P}(t_l)\,\boldsymbol{Q}(t_k)\,D_{k,l} + \omega_K \boldsymbol{Q}(1)\sum_{l=0}^{K} \boldsymbol{P}(t_l) D_{K,l},\end{aligned} \tag{9.36}$$

and similarly

$$\begin{aligned}\int_{-1}^{+1} \boldsymbol{P}(t)\,\dot{\boldsymbol{Q}}(t)\,\mathrm{d}t &= \sum_{k=1}^{K}\omega_k \sum_{j=0}^{K}\sum_{l=1}^{K} \boldsymbol{P}(t_j)\,\boldsymbol{Q}(t_l)\,\boldsymbol{L}_j(t_k)\,\dot{\overline{\boldsymbol{L}}}_l(t_k) = \sum_{k=1}^{K}\sum_{l=1}^{K} \omega_k \boldsymbol{P}(t_k)\,\boldsymbol{Q}(t_l)\,\dot{\overline{\boldsymbol{L}}}_l(t_k)\\ &= \sum_{k=1}^{K-1}\sum_{l=1}^{K} \omega_k \boldsymbol{P}(t_k)\,\boldsymbol{Q}(t_l)\,\overline{D}_{k,l} + \omega_K \boldsymbol{P}(1)\sum_{l=1}^{K} \boldsymbol{Q}(t_l)\,\overline{D}_{K,l}.\end{aligned} \tag{9.37}$$

By means of the partial integration rule we find

$$\boldsymbol{P}(t)\,\boldsymbol{Q}(t)\big|_{-1}^{+1} = \int_{-1}^{+1} \dot{\boldsymbol{P}}(t)\,\boldsymbol{Q}(t)\,\mathrm{d}t + \int_{-1}^{+1} \boldsymbol{P}(t)\,\dot{\boldsymbol{Q}}(t)\,\mathrm{d}t. \tag{9.38}$$

Combining (9.36), (9.37) and (9.38) yields

$$\begin{aligned}\boldsymbol{P}(1)\,\boldsymbol{Q}(1) - \boldsymbol{P}(-1)\,\boldsymbol{Q}(-1) &= \sum_{k=1}^{K-1}\sum_{l=0}^{K} \omega_k \boldsymbol{P}(t_l)\,\boldsymbol{Q}(t_k)\,D_{k,l} + \omega_K \boldsymbol{Q}(1)\sum_{l=0}^{K} \boldsymbol{P}(t_l)\,D_{K,l}\\ &\quad + \sum_{k=1}^{K-1}\sum_{l=1}^{K} \omega_k \boldsymbol{P}(t_k)\,\boldsymbol{Q}(t_l)\,\overline{D}_{k,l} + \omega_K \boldsymbol{P}(1)\sum_{l=1}^{K} \boldsymbol{Q}(t_l)\,\overline{D}_{K,l}.\end{aligned} \tag{9.39}$$

According to our assumptions $\boldsymbol{P}(\cdot)$ is any polynomial of degree K and $\boldsymbol{Q}(\cdot)$ is any polynomial of degree $K-1$. In particular, we can choose $\boldsymbol{P} = \boldsymbol{L}_i$ and $\boldsymbol{Q} = \overline{\boldsymbol{L}}_j$, i.e., (9.39) looks as follows:

$$\begin{aligned}\boldsymbol{L}_i(1)\,\overline{\boldsymbol{L}}_j(1) - \boldsymbol{L}_i(-1)\,\overline{\boldsymbol{L}}_j(-1) &= \sum_{k=1}^{K-1}\sum_{l=0}^{K} \omega_k \boldsymbol{L}_i(t_l)\,\overline{\boldsymbol{L}}_j(t_k) D_{k,l} + \omega_K \overline{\boldsymbol{L}}_j(1)\sum_{l=0}^{K} \boldsymbol{L}_i(t_l)\,D_{K,l}\\ &\quad + \sum_{k=1}^{K-1}\sum_{l=1}^{K} \omega_k \boldsymbol{L}_i(t_k)\,\overline{\boldsymbol{L}}_j(t_l)\,\overline{D}_{k,l} + \omega_K \boldsymbol{L}_i(1)\sum_{l=1}^{K} \overline{\boldsymbol{L}}_j(t_l)\,\overline{D}_{K,l}.\end{aligned}$$

Choosing $j \equiv K$ we have

$$\boldsymbol{L}_i(1) - \boldsymbol{L}_i(-1)\,\overline{\boldsymbol{L}}_K(-1) = \omega_K \sum_{l=0}^{K} \boldsymbol{L}_i(t_l)\,D_{K,l} + \sum_{k=1}^{K-1} \omega_k \boldsymbol{L}_i(t_k)\,\overline{D}_{k,K} + \omega_K \boldsymbol{L}_i(1)\,\overline{D}_{K,K}.$$

For $i \equiv K$ it holds

$$1 = \omega_K D_{K,K} + \omega_K \overline{D}_{K,K}.$$

If we choose $i \in [K-1]$ this yields

$$0 = \omega_K D_{K,i} + \omega_i \overline{D}_{i,K}.$$

With $j \in [K-1]$ and $i \in [K-1]$ we calculate

$$\begin{aligned} 0 &= \sum_{k=1}^{K-1} \sum_{l=0}^{K} \omega_k \boldsymbol{L}_i(t_l) \overline{\boldsymbol{L}}_j(t_k) D_{k,l} + \sum_{k=1}^{K-1} \sum_{l=1}^{K} \omega_k \boldsymbol{L}_i(t_k) \overline{\boldsymbol{L}}_j(t_l) \overline{D}_{k,l} \\ &= \omega_j D_{j,i} + \omega_i \overline{D}_{i,j} \end{aligned}$$

If we summarize the previous results we have for

- $i, j \in [K]$, $i \neq j$:

$$D_{j,i} = -\frac{\omega_i}{\omega_j} \overline{D}_{i,j}.$$

- $j \in [K-1]$:

$$D_{j,j} = -\overline{D}_{j,j}.$$

- $j = K$:

$$D_{K,K} = -\overline{D}_{K,K} + \frac{1}{\omega_K}.$$

This completes the proof. □

The relationship between $D_{j,0}$ and $\overline{D}_{j,i}$ is caught in the subsequent result.

Lemma 9.5
Taking the notations from Lemma 9.4 it holds

$$\begin{aligned} D_{i,0} &= \sum_{j=1}^{K-1} \frac{\omega_j}{\omega_K} \overline{D}_{j,K} + \overline{D}_{K,K} - \frac{1}{\omega_K}, \qquad i \in [K-1], \\ D_{K,0} &= \sum_{j=1}^{K} \frac{\omega_j}{\omega_K} \overline{D}_{j,K} - \frac{1}{\omega_K}. \end{aligned}$$ △

Proof Let $\boldsymbol{P} \equiv C \neq 0$ for a constant C. Then we calculate for $i \in [K]$

$$0 = \dot{\boldsymbol{P}}(t_i) = \sum_{j=0}^{K} \boldsymbol{P}(t_j) \dot{\boldsymbol{L}}_j(t_i) = C \sum_{j=0}^{K} D_{i,j}.$$

We conclude for $i \in [K-1]$

$$D_{i,0} = -\sum_{j=1}^{K} D_{i,j} = \sum_{j=1}^{K} \frac{\omega_j}{\omega_i} \overline{D}_{j,i}.$$

Likewise we find for $i = K$:

$$D_{K,0} = -\sum_{j=1}^{K} D_{K,j} = -\sum_{j=1}^{K-1} D_{K,j} - D_{K,K} = \sum_{j=1}^{K-1} \frac{\omega_j}{\omega_K} \overline{D}_{j,K} + \overline{D}_{K,K} - \frac{1}{\omega_K}$$
$$= \sum_{j=1}^{K} \frac{\omega_j}{\omega_K} \overline{D}_{j,K} - \frac{1}{\omega_K}.$$

□

9.2 First Discretize, Then Optimize: Local Collocation Approach

Recall NLP (7.22), which was derived by a local pseudospectral collocation approach in Section 7.4:

$$\begin{aligned}
\min_{t_s, t_f, x, u} \quad & \varphi(t_s, x_0^{(1)}, t_f, x_{K^{(N)}}^{(N)}) + \frac{h}{2} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \psi_n(\tau_i^{(n)}, x_i^{(n)}, u_i^{(n)}; t_s, t_f) \\
\text{s.t.} \quad & \mathbf{0}_{n_x} = \frac{h}{2} \frac{h_n}{2} \cdot \boldsymbol{f}_n\left(\tau_j^{(n)}, x_j^{(n)}, u_j^{(n)}; t_s, t_f\right) - \sum_{i=0}^{K^{(n)}} x_i^{(n)} D_{j,i}^{(n)}, \quad n \in [N],\ j \in [K^{(n)}], \\
& \mathbf{0}_{n_c} \geqslant \boldsymbol{c}_n\left(\tau_j^{(n)}, x_j^{(n)}, u_j^{(n)}; t_s, t_f\right), \qquad n \in [N],\ j \in [K^{(n)}], \\
& \mathbf{0}_{n_r} = \boldsymbol{r}(t_s, x_0^{(1)}, t_f, x_{K^{(N)}}^{(N)}), \\
& \mathbf{0}_{n_x} = x_0^{(n+1)} - x_{K^{(n)}}^{(n)}, \qquad n \in [N-1].
\end{aligned}$$

Some Notations

We determine the KKT system (see Theorem 3.22) for this NLP. To this end we introduce Lagrange multipliers corresponding to

(i) system dynamics, $\lambda = [\lambda^{(n)^T}]_{1 \leqslant n \leqslant N}^T$, $\lambda^{(n)} = [\lambda_1^{(n)^T}, \ldots, \lambda_{K^{(n)}}^{(n)^T}]^T$,

(ii) path constraints, $\mu = [\mu^{(n)^T}]_{1 \leqslant n \leqslant N}^T$, $\mu^{(n)} = [\mu_1^{(n)^T}, \ldots, \mu_{K^{(n)}}^{(n)^T}]^T$,

(iii) boundary conditions, $\nu^{(0)}$, and

(iv) matching conditions, $[\nu^{(1)^T}, \ldots, \nu^{(N-1)^T}]^T$,

where we use the notation $\nu = [\nu^{(0)^T}, \ldots, \nu^{(N-1)^T}]^T$. The dimensions of λ, μ, and ν are denoted by n_λ, n_μ, and n_ν such that

$$n_\lambda = \sum_{n=1}^{N} K^{(n)} n_x, \qquad n_\mu = \sum_{n=1}^{N} K^{(n)} n_c, \qquad n_\nu = n_r + (N-1) n_x.$$

As in Section 7.4 we keep the terms for state variables, x, and for control variables, u. We also define a variable, η, holding all equality constraint multipliers, i.e., we have $\eta = [\lambda^T, \nu^T]^T$. The dimension of η is denoted by n_η. For convenience we introduce the terms

$$\begin{aligned} \boldsymbol{r}'_{t_s} &\stackrel{\text{def}}{=} \nabla_{t_s} \boldsymbol{r}(t_s, x_0^{(1)}, t_f, x_{K^{(N)}}^{(N)}), \qquad & \boldsymbol{r}'_{x_s} &\stackrel{\text{def}}{=} \nabla_{x_s} \boldsymbol{r}(t_s, x_0^{(1)}, t_f, x_{K^{(N)}}^{(N)}), \\ \boldsymbol{r}'_{t_f} &\stackrel{\text{def}}{=} \nabla_{t_f} \boldsymbol{r}(t_s, x_0^{(1)}, t_f, x_{K^{(N)}}^{(N)}), \qquad & \boldsymbol{r}'_{x_f} &\stackrel{\text{def}}{=} \nabla_{x_f} \boldsymbol{r}(t_s, x_0^{(1)}, t_f, x_{K^{(N)}}^{(N)}). \end{aligned}$$

We do likewise for the function $\boldsymbol{\varphi}(\cdot)$. In a similar way we use the notations

$$\begin{aligned} \boldsymbol{f}\left[\tau_i^{(n)}\right] &\stackrel{\text{def}}{=} \boldsymbol{f}_n(\tau_i^{(n)}, x_i^{(n)}, u_i^{(n)}; t_s, t_f)^T, \quad & \boldsymbol{f}'_t\left[\tau_i^{(n)}\right] &\stackrel{\text{def}}{=} \nabla_t \boldsymbol{f}_n(\tau_i^{(n)}, x_i^{(n)}, u_i^{(n)}; t_s, t_f), \\ \boldsymbol{f}'_x\left[\tau_i^{(n)}\right] &\stackrel{\text{def}}{=} \nabla_x \boldsymbol{f}_n(\tau_i^{(n)}, x_i^{(n)}, u_i^{(n)}; t_s, t_f), \quad & \boldsymbol{f}'_u\left[\tau_i^{(n)}\right] &\stackrel{\text{def}}{=} \nabla_u \boldsymbol{f}_n(\tau_i^{(n)}, x_i^{(n)}, u_i^{(n)}; t_s, t_f). \end{aligned}$$

Analogous terms are employed for the functions $\boldsymbol{\psi}(\cdot)$ and $\boldsymbol{c}(\cdot)$.

Lagrange Function and its Derivatives

The Lagrange function $\mathcal{L} : \mathbb{R}^{n_w} \times \mathbb{R}^{n_\eta} \times \mathbb{R}^{n_\mu} \longrightarrow \mathbb{R}$ for NLP (7.22) is then defined as follows:

$$\begin{aligned} \mathcal{L}(w, \eta, \mu) = \boldsymbol{\Phi}(w) &+ \sum_{n=1}^{N} \sum_{i=1}^{K^{(n)}} \lambda_i^{(n)^T} \boldsymbol{F}_i^{(n)}(w) + \sum_{n=1}^{N} \sum_{i=1}^{K^{(n)}} \mu_i^{(n)^T} \boldsymbol{C}_i^{(n)}(w) \\ &+ \nu^{(0)^T} \boldsymbol{R}(w) + \sum_{n=1}^{N-1} \nu^{(n)^T} \boldsymbol{M}^{(n)}(w). \end{aligned}$$

The multipliers $\lambda_i^{(n)}$ and $\mu_i^{(n)}$ are reparameterized according to the formula

$$\lambda_i^{(n)} = \omega_i^{(n)} \widetilde{\lambda}_i^{(n)}, \qquad \mu_i^{(n)} = \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \widetilde{\mu}_i^{(n)}. \tag{9.40}$$

Forming the Lagrange function by means of $\widetilde{\lambda}$ and $\widetilde{\mu}$ (both built in a canonical way) yields

$$\begin{aligned} \mathcal{L}(w, \widetilde{\eta}, \widetilde{\mu}) = \boldsymbol{\varphi}(t_s, x_0^{(1)}, t_f, x_{K^{(N)}}^{(N)}) &+ \frac{h}{2} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \boldsymbol{\psi}_n(\tau_i^{(n)}, x_i^{(n)}, u_i^{(n)}; t_s, t_f) \\ &+ \sum_{n=1}^{N} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left(\widetilde{\lambda}_i^{(n)}\right)^T \left\{ \frac{h}{2} \frac{h_n}{2} \cdot \boldsymbol{f}_n\left(\tau_i^{(n)}, x_i^{(n)}, u_i^{(n)}; t_s, t_f\right) - \sum_{j=0}^{K^{(n)}} x_j^{(n)} D_{i,j}^{(n)} \right\} \\ &+ \sum_{n=1}^{N} \sum_{i=1}^{K^{(n)}} \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \left(\widetilde{\mu}_i^{(n)}\right)^T \boldsymbol{c}_n\left(\tau_i^{(n)}, x_i^{(n)}, u_i^{(n)}; t_s, t_f\right) \\ &+ \nu^{(0)^T} \boldsymbol{r}(t_s, x_0^{(1)}, t_f, x_{K^{(N)}}^{(N)}) + \sum_{n=1}^{N-1} \nu^{(n)^T} \left\{ x_0^{(n+1)} - x_{K^{(n)}}^{(n)} \right\}. \end{aligned}$$

Let us consider the function $\boldsymbol{\Xi} : \mathbb{R}^{n_w} \times \mathbb{R}^{n_\eta} \times \mathbb{R}^{n_\mu} \longrightarrow \mathbb{R}^{n_s}$ which is defined as

$$\boldsymbol{\Xi}(w, \widetilde{\eta}, \widetilde{\mu}) = \left[\nabla_{x_0^{(1)}} \mathcal{L}^T, \ldots, \nabla_{x_{K^{(1)}}^{(1)}} \mathcal{L}^T, \ldots, \nabla_{x_0^{(N)}} \mathcal{L}^T, \ldots, \nabla_{x_{K^{(N)}}^{(N)}} \mathcal{L}^T \right]^T (w, \widetilde{\eta}, \widetilde{\mu}),$$

where the single components are calculated as

$$\begin{aligned}
\boldsymbol{\Xi}_0^{(1)}(w, \widetilde{\eta}, \widetilde{\mu}) &= -\sum_{i=1}^{K^{(1)}} \omega_i^{(1)} D_{i,0}^{(1)} \widetilde{\lambda}_i^{(1)} + \boldsymbol{r}'_{x_s} \; \nu^{(0)} + \boldsymbol{\varphi}'_{x_s}, \\
\boldsymbol{\Xi}_0^{(n)}(w, \widetilde{\eta}, \widetilde{\mu}) &= -\sum_{i=1}^{K^{(n)}} \omega_i^{(n)} D_{i,0}^{(n)} \widetilde{\lambda}_i^{(n)} + \nu^{(n-1)}, \qquad n \in [N] \setminus \{1\}, \\
\boldsymbol{\Xi}_i^{(n)}(w, \widetilde{\eta}, \widetilde{\mu}) &= -\sum_{j=1}^{K^{(n)}} \omega_j^{(n)} D_{j,i}^{(n)} \widetilde{\lambda}_j^{(n)} + \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \boldsymbol{\psi}'_x \left[\tau_i^{(n)} \right] + \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \boldsymbol{f}'_x \left[\tau_i^{(n)} \right] \widetilde{\lambda}_i^{(n)} \\
&\quad + \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \boldsymbol{c}'_x \left[\tau_i^{(n)} \right] \widetilde{\mu}_i^{(n)}, \qquad n \in [N], \; i \in [K^{(n)} - 1], \\
\boldsymbol{\Xi}_{K^{(n)}}^{(n)}(w, \widetilde{\eta}, \widetilde{\mu}) &= -\sum_{j=1}^{K^{(n)}} \omega_j^{(n)} D_{j,K^{(n)}}^{(n)} \widetilde{\lambda}_j^{(n)} + \frac{h}{2} \frac{h_n}{2} \omega_{K^{(n)}}^{(n)} \boldsymbol{\psi}'_x \left[\tau_{K^{(n)}}^{(n)} \right] + \frac{h}{2} \frac{h_n}{2} \omega_{K^{(n)}}^{(n)} \boldsymbol{f}'_x \left[\tau_{K^{(n)}}^{(n)} \right] \widetilde{\lambda}_{K^{(n)}}^{(n)} \\
&\quad + \frac{h}{2} \frac{h_n}{2} \omega_{K^{(n)}}^{(n)} \boldsymbol{c}'_x \left[\tau_{K^{(n)}}^{(n)} \right] \widetilde{\mu}_{K^{(n)}}^{(n)} - \nu^{(n)}, \qquad n \in [N-1], \\
\boldsymbol{\Xi}_{K^{(N)}}^{(N)}(w, \widetilde{\eta}, \widetilde{\mu}) &= -\sum_{j=1}^{K^{(N)}} \omega_j^{(N)} D_{j,K^{(N)}}^{(N)} \widetilde{\lambda}_j^{(N)} + \frac{h}{2} \frac{h_N}{2} \omega_{K^{(N)}}^{(N)} \boldsymbol{\psi}'_x \left[\tau_{K^{(N)}}^{(N)} \right] + \frac{h}{2} \frac{h_N}{2} \omega_{K^{(N)}}^{(N)} \boldsymbol{f}'_x \left[\tau_{K^{(N)}}^{(N)} \right] \widetilde{\lambda}_{K^{(N)}}^{(N)} \\
&\quad + \frac{h}{2} \frac{h_N}{2} \omega_{K^{(N)}}^{(N)} \boldsymbol{c}'_x \left[\tau_{K^{(N)}}^{(N)} \right] \widetilde{\mu}_{K^{(N)}}^{(N)} + \boldsymbol{r}'_{x_f} \; \nu^{(0)} + \boldsymbol{\varphi}'_{x_f}.
\end{aligned}$$

Likewise we introduce the function $\boldsymbol{Y} : \mathbb{R}^{n_w} \times \mathbb{R}^{n_\eta} \times \mathbb{R}^{n_\mu} \longrightarrow \mathbb{R}^{n_q}$ which is defined as

$$\boldsymbol{Y}(w, \widetilde{\eta}, \widetilde{\mu}) = \left[\nabla_{u_1^{(1)}} \mathcal{L}^T, \ldots, \nabla_{u_{K^{(1)}}^{(1)}} \mathcal{L}^T, \ldots, \nabla_{u_1^{(N)}} \mathcal{L}^T, \ldots, \nabla_{u_{K^{(N)}}^{(N)}} \mathcal{L}^T \right]^T (w, \widetilde{\eta}, \widetilde{\mu}),$$

where the single components are calculated as

$$\begin{aligned}
\boldsymbol{Y}_i^{(n)}(w, \widetilde{\eta}, \widetilde{\mu}) &= + \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \boldsymbol{\psi}'_u \left[\tau_i^{(n)} \right] + \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \boldsymbol{f}'_u \left[\tau_i^{(n)} \right] \widetilde{\lambda}_i^{(n)} \\
&\quad + \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \boldsymbol{c}'_u \left[\tau_i^{(n)} \right] \widetilde{\mu}_i^{(n)}, \qquad n \in [N], \; i \in [K^{(n)}].
\end{aligned}$$

The Lagrange gradient with respect to $\widetilde{\lambda}$ can be expressed by means of the function $\boldsymbol{F}(\cdot)$ as

$$\nabla_{\widetilde{\lambda}_i^{(n)}} \mathcal{L}(w, \widetilde{\eta}, \widetilde{\mu}) = -\omega_i^{(n)} \boldsymbol{F}_i^{(n)}(w) = -\sum_{j=0}^{K^{(n)}} \omega_i^{(n)} D_{i,j}^{(n)} x_j^{(n)} + \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \boldsymbol{f}_n(\tau_i^{(n)}, x_i^{(n)}, u_i^{(n)}),$$

where $n \in [N]$, $i \in [K^{(n)}]$. Employing the function $\boldsymbol{C}(\cdot)$ we can write the Lagrange gradient with respect to $\widetilde{\mu}$ as

$$\nabla_{\widetilde{\mu}_i^{(n)}} \mathcal{L}(w, \widetilde{\eta}, \widetilde{\mu}) = \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \boldsymbol{C}_i^{(n)}(w) = \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \boldsymbol{c}_n(\tau_i^{(n)}, x_i^{(n)}, u_i^{(n)}),\ n \in [N],\ i \in [K^{(n)}].$$

In a similar way the Lagrange gradient with respect ν can be written with the aid of $\boldsymbol{R}(\cdot)$ and $\boldsymbol{M}(\cdot)$ as

$$\nabla_{\nu^{(0)}} \mathcal{L}(w, \widetilde{\eta}, \widetilde{\mu}) = \boldsymbol{R}(w) = \boldsymbol{r}(t_\mathrm{s}, x_0^{(1)}, t_\mathrm{f}, x_{K^{(N)}}^{(N)})$$

and

$$\nabla_{\nu^{(n)}} \mathcal{L}(w, \widetilde{\eta}, \widetilde{\mu}) = \boldsymbol{M}^{(n)}(w) = x_0^{(n+1)} - x_{K^{(n)}}^{(n)},\ n \in [N-1].$$

Finally, we need the gradient of the Lagrangian with respect to t_s and t_f. Let the function $\boldsymbol{T}_{t_\mathrm{s}} : \mathbb{R}^{n_\mathrm{w}} \times \mathbb{R}^{n_\eta} \times \mathbb{R}^{n_\mu} \longrightarrow \mathbb{R}$ be the Lagrange gradient with respect to t_s, i.e., we have

$$\begin{aligned}
\boldsymbol{T}_{t_\mathrm{s}}(w, \widetilde{\eta}, \widetilde{\mu}) = &-\frac{1}{2} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left\{ \boldsymbol{\psi}\left[\tau_i^{(n)}\right] - \frac{h}{2} \left(1 - \boldsymbol{t}_n\left(\tau_i^{(n)}\right)\right) \boldsymbol{\psi}_t'\left[\tau_i^{(n)}\right] \right\} \\
&-\frac{1}{2} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left\{ \boldsymbol{f}\left[\tau_i^{(n)}\right] - \frac{h}{2} \left(1 - \boldsymbol{t}_n\left(\tau_i^{(n)}\right)\right) \boldsymbol{f}_t'\left[\tau_i^{(n)}\right] \right\} \widetilde{\lambda}_i^{(n)} \\
&+\frac{1}{2} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \frac{h}{2} \left(1 - \boldsymbol{t}_n\left(\tau_i^{(n)}\right)\right) \boldsymbol{c}_t'\left[\tau_i^{(n)}\right] \widetilde{\mu}_i^{(n)} + \boldsymbol{r}_{t_\mathrm{s}}'\ \nu^{(0)} + \boldsymbol{\varphi}_{t_\mathrm{s}}' \\
= &-\frac{1}{2} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left\{ \boldsymbol{\psi}\left[\tau_i^{(n)}\right] + \boldsymbol{f}\left[\tau_i^{(n)}\right] \widetilde{\lambda}_i^{(n)} \right\} + \boldsymbol{r}_{t_\mathrm{s}}'\ \nu^{(0)} + \boldsymbol{\varphi}_{t_\mathrm{s}}' \\
&+\frac{1}{2} \frac{h}{2} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left(1 - \boldsymbol{t}_n\left(\tau_i^{(n)}\right)\right) \left\{ \boldsymbol{\psi}_t'\left[\tau_i^{(n)}\right] + \boldsymbol{f}_t'\left[\tau_i^{(n)}\right] \widetilde{\lambda}_i^{(n)} + \boldsymbol{c}_t'\left[\tau_i^{(n)}\right] \widetilde{\mu}_i^{(n)} \right\}.
\end{aligned}$$

Likewise, let $\boldsymbol{T}_{t_\mathrm{f}} : \mathbb{R}^{n_\mathrm{w}} \times \mathbb{R}^{n_\eta} \times \mathbb{R}^{n_\mu} \longrightarrow \mathbb{R}$ be the Lagrange gradient with respect to t_f, i.e., we have

$$\begin{aligned}
\boldsymbol{T}_{t_\mathrm{f}}(w, \widetilde{\eta}, \widetilde{\mu}) = &+\frac{1}{2} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left\{ \boldsymbol{\psi}\left[\tau_i^{(n)}\right] + \frac{h}{2} \left(1 + \boldsymbol{t}_n\left(\tau_i^{(n)}\right)\right) \boldsymbol{\psi}_t'\left[\tau_i^{(n)}\right] \right\} \\
&+\frac{1}{2} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left\{ \boldsymbol{f}\left[\tau_i^{(n)}\right] + \frac{h}{2} \left(1 + \boldsymbol{t}_n\left(\tau_i^{(n)}\right)\right) \boldsymbol{f}_t'\left[\tau_i^{(n)}\right] \right\} \widetilde{\lambda}_i^{(n)} \\
&+\frac{1}{2} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \frac{h}{2} \left(1 + \boldsymbol{t}_n\left(\tau_i^{(n)}\right)\right) \boldsymbol{c}_t'\left[\tau_i^{(n)}\right] \widetilde{\mu}_i^{(n)} + \boldsymbol{r}_{t_\mathrm{f}}'\ \nu^{(0)} + \boldsymbol{\varphi}_{t_\mathrm{f}}' \\
= &+\frac{1}{2} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left\{ \boldsymbol{\psi}\left[\tau_i^{(n)}\right] + \boldsymbol{f}\left[\tau_i^{(n)}\right] \widetilde{\lambda}_i^{(n)} \right\} + \boldsymbol{r}_{t_\mathrm{f}}'\ \nu^{(0)} + \boldsymbol{\varphi}_{t_\mathrm{f}}' \\
&+\frac{1}{2} \frac{h}{2} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left(1 + \boldsymbol{t}_n\left(\tau_i^{(n)}\right)\right) \left\{ \boldsymbol{\psi}_t'\left[\tau_i^{(n)}\right] + \boldsymbol{f}_t'\left[\tau_i^{(n)}\right] \widetilde{\lambda}_i^{(n)} + \boldsymbol{c}_t'\left[\tau_i^{(n)}\right] \widetilde{\mu}_i^{(n)} \right\}.
\end{aligned}$$

The KKT System

In this section we formulate the KKT system for the collocation NLP. With the aid of the notation of the previous section we can write it as follows:

$$\begin{aligned}
&\mathbf{0} = \Xi(w,\widetilde{\eta},\widetilde{\mu}), && 0 = \boldsymbol{T}_{t_s}(w,\widetilde{\eta},\widetilde{\mu}), && \mathbf{0} = \boldsymbol{F}(w), && \mathbf{0} = \boldsymbol{R}(w),\\
&\mathbf{0} = \boldsymbol{Y}(w,\widetilde{\eta},\widetilde{\mu}), && 0 = \boldsymbol{T}_{t_f}(w,\widetilde{\eta},\widetilde{\mu}), && \mathbf{0} = \boldsymbol{M}(w), && \mathbf{0} \geqslant \boldsymbol{C}(w),\\
&\mathbf{0} \leqslant \widetilde{\mu}, && 0 = \widetilde{\mu}^T \boldsymbol{C}(w).
\end{aligned}$$

For later use we reformulate the KKT system in some components. To this end, we define variables

$$\widetilde{\lambda}_0^{(n)} \stackrel{\text{def}}{=} \sum_{j=1}^{K^{(n)}} \omega_j^{(n)} D_{j,0}^{(n)} \widetilde{\lambda}_j^{(n)}, \qquad n \in [N].$$

We calculate

$$\begin{aligned}
\widetilde{\lambda}_0^{(n)} &= \sum_{j=1}^{K^{(n)}} \omega_j^{(n)} D_{j,0}^{(n)} \widetilde{\lambda}_j^{(n)} = \sum_{j=1}^{K^{(n)}-1} \omega_j^{(n)} D_{j,0}^{(n)} \widetilde{\lambda}_j^{(n)} + \omega_{K^{(n)}}^{(n)} D_{K^{(n)},0}^{(n)} \widetilde{\lambda}_{K^{(n)}}^{(n)}\\
&= \sum_{j=1}^{K^{(n)}-1} \omega_j^{(n)} \left[\sum_{i=1}^{K^{(n)}} \frac{\omega_i^{(n)}}{\omega_j^{(n)}} \overline{D}_{i,j}^{(n)} \right] \widetilde{\lambda}_j^{(n)} + \omega_{K^{(n)}}^{(n)} \left[\sum_{i=1}^{K^{(n)}} \frac{\omega_i^{(n)}}{\omega_{K^{(n)}}^{(n)}} \overline{D}_{i,K^{(n)}}^{(n)} - \frac{1}{\omega_{K^{(n)}}^{(n)}} \right] \widetilde{\lambda}_{K^{(n)}}^{(n)}\\
&= \sum_{j=1}^{K^{(n)}} \left[\sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \overline{D}_{i,j}^{(n)} \right] \widetilde{\lambda}_j^{(n)} - \widetilde{\lambda}_{K^{(n)}}^{(n)} = \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \sum_{j=1}^{K^{(n)}} \overline{D}_{i,j}^{(n)} \widetilde{\lambda}_j^{(n)} - \widetilde{\lambda}_{K^{(n)}}^{(n)}.
\end{aligned}$$

Hence, we can write $\Xi_0^{(1)}(w,\widetilde{\eta},\widetilde{\mu}) = \mathbf{0}$ as

$$\begin{aligned}
\Xi_0^{(1)}(w,\widetilde{\eta},\widetilde{\mu}) &= -\sum_{i=1}^{K^{(1)}} \omega_i^{(1)} D_{i,0}^{(1)} \widetilde{\lambda}_i^{(1)} + \boldsymbol{r}'_{x_s}\, \nu^{(0)} + \boldsymbol{\varphi}'_{x_s} = -\widetilde{\lambda}_0^{(1)} + \boldsymbol{r}'_{x_s}\, \nu^{(0)} + \boldsymbol{\varphi}'_{x_s}\\
&= -\sum_{i=1}^{K^{(1)}} \omega_i^{(1)} \sum_{j=1}^{K^{(1)}} \overline{D}_{i,j}^{(1)} \widetilde{\lambda}_j^{(1)} + \widetilde{\lambda}_{K^{(1)}}^{(1)} + \boldsymbol{r}'_{x_s}\, \nu^{(0)} + \boldsymbol{\varphi}'_{x_s} = \mathbf{0}
\end{aligned}$$

and for $n \in [N-1]$ we can write $\Xi_0^{(n+1)}(w,\widetilde{\eta},\widetilde{\mu}) = \mathbf{0}$ as

$$\begin{aligned}
\Xi_0^{(n+1)}(w,\widetilde{\eta},\widetilde{\mu}) &= -\sum_{i=1}^{K^{(n+1)}} \omega_i^{(n+1)} D_{i,0}^{(n+1)} \widetilde{\lambda}_i^{(n+1)} + \nu^{(n)} = -\widetilde{\lambda}_0^{(n+1)} + \nu^{(n)}\\
&= -\sum_{i=1}^{K^{(n+1)}} \omega_i^{(n+1)} \sum_{j=1}^{K^{(n+1)}} \overline{D}_{i,j}^{(n+1)} \widetilde{\lambda}_j^{(n+1)} + \widetilde{\lambda}_{K^{(n+1)}}^{(n+1)} + \nu^{(n)} = \mathbf{0}.
\end{aligned}$$

We can conclude that $\Xi^{(n)}_{K^{(n)}}(w,\widetilde{\eta},\widetilde{\mu})=0$, $n\in[N-1]$, holds if and only if

$$\begin{aligned}
\frac{\Xi^{(n)}_{K^{(n)}}(w,\widetilde{\eta},\widetilde{\mu})}{\omega^{(n)}_{K^{(n)}}} =& \frac{h}{2}\frac{h_n}{2}\left[\psi'_x\left[\tau^{(n)}_{K^{(n)}}\right]+f'_x\left[\tau^{(n)}_{K^{(n)}}\right]\widetilde{\lambda}^{(n)}_{K^{(n)}}+\boldsymbol{c}'_x\left[\tau^{(n)}_{K^{(n)}}\right]\widetilde{\mu}^{(n)}_{K^{(n)}}\right]\\
&-\sum_{j=1}^{K^{(n)}}\frac{\omega^{(n)}_j}{\omega^{(n)}_{K^{(n)}}}D^{(n)}_{j,K^{(n)}}\widetilde{\lambda}^{(n)}_j-\frac{1}{\omega^{(n)}_{K^{(n)}}}\ \nu^{(n)}\\
=&\frac{h}{2}\frac{h_n}{2}\left[\psi'_x\left[\tau^{(n)}_{K^{(n)}}\right]+f'_x\left[\tau^{(n)}_{K^{(n)}}\right]\widetilde{\lambda}^{(n)}_{K^{(n)}}+\boldsymbol{c}'_x\left[\tau^{(n)}_{K^{(n)}}\right]\widetilde{\mu}^{(n)}_{K^{(n)}}\right]+\sum_{j=1}^{K^{(n)}}\overline{D}^{(n)}_{K^{(n)},j}\widetilde{\lambda}^{(n)}_j\\
&-\frac{\widetilde{\lambda}^{(n)}_{K^{(n)}}}{\omega^{(n)}_{K^{(n)}}}-\frac{1}{\omega^{(n)}_{K^{(n)}}}\left[\sum_{i=1}^{K^{(n+1)}}\omega^{(n+1)}_i\sum_{j=1}^{K^{(n+1)}}\overline{D}^{(n+1)}_{i,j}\widetilde{\lambda}^{(n+1)}_j-\widetilde{\lambda}^{(n+1)}_{K^{(n+1)}}\right].\\
=&\frac{h}{2}\frac{h_n}{2}\left[\psi'_x\left[\tau^{(n)}_{K^{(n)}}\right]+f'_x\left[\tau^{(n)}_{K^{(n)}}\right]\widetilde{\lambda}^{(n)}_{K^{(n)}}+\boldsymbol{c}'_x\left[\tau^{(n)}_{K^{(n)}}\right]\widetilde{\mu}^{(n)}_{K^{(n)}}\right]+\sum_{j=1}^{K^{(n)}}\overline{D}^{(n)}_{K^{(n)},j}\widetilde{\lambda}^{(n)}_j\\
&-\frac{1}{\omega^{(n)}_{K^{(n)}}}\left[\widetilde{\lambda}^{(n)}_{K^{(n)}}-\widetilde{\lambda}^{(n+1)}_{K^{(n+1)}}+\sum_{i=1}^{K^{(n+1)}}\omega^{(n+1)}_i\sum_{j=1}^{K^{(n+1)}}\overline{D}^{(n+1)}_{i,j}\widetilde{\lambda}^{(n+1)}_j\right]=0.
\end{aligned}$$

In a similar way we conclude that $\Xi^{(N)}_{K^{(N)}}(w,\widetilde{\eta},\widetilde{\mu})=0$ holds if and only if

$$\begin{aligned}
\frac{\Xi^{(N)}_{K^{(N)}}(w,\widetilde{\eta},\widetilde{\mu})}{\omega^{(N)}_{K^{(N)}}} =& \frac{h}{2}\frac{h_N}{2}\left[\psi'_x\left[\tau^{(N)}_{K^{(N)}}\right]+f'_x\left[\tau^{(N)}_{K^{(N)}}\right]\widetilde{\lambda}^{(N)}_{K^{(N)}}+\boldsymbol{c}'_x\left[\tau^{(N)}_{K^{(N)}}\right]\widetilde{\mu}^{(N)}_{K^{(N)}}\right]\\
&-\sum_{j=1}^{K^{(N)}}\frac{\omega^{(N)}_j}{\omega^{(N)}_{K^{(N)}}}D^{(N)}_{j,K^{(N)}}\widetilde{\lambda}^{(N)}_j+\frac{1}{\omega^{(N)}_{K^{(N)}}}\left[\boldsymbol{r}'_{x_f}\ \nu^{(0)}+\varphi'_{x_f}\right]\\
=&\frac{h}{2}\frac{h_N}{2}\left[\psi'_x\left[\tau^{(N)}_{K^{(N)}}\right]+f'_x\left[\tau^{(N)}_{K^{(N)}}\right]\widetilde{\lambda}^{(N)}_{K^{(N)}}+\boldsymbol{c}'_x\left[\tau^{(N)}_{K^{(N)}}\right]\widetilde{\mu}^{(N)}_{K^{(N)}}\right]\\
&+\sum_{j=1}^{K^{(N)}}\overline{D}^{(N)}_{K^{(N)},j}\widetilde{\lambda}^{(N)}_j-\frac{\widetilde{\lambda}^{(N)}_{K^{(N)}}}{\omega^{(N)}_{K^{(N)}}}+\frac{1}{\omega^{(N)}_{K^{(N)}}}\left[\boldsymbol{r}'_{x_f}\ \nu^{(0)}+\varphi'_{x_f}\right]\\
=&\frac{h}{2}\frac{h_N}{2}\left[\psi'_x\left[\tau^{(N)}_{K^{(N)}}\right]+f'_x\left[\tau^{(N)}_{K^{(N)}}\right]\widetilde{\lambda}^{(N)}_{K^{(N)}}+\boldsymbol{c}'_x\left[\tau^{(N)}_{K^{(N)}}\right]\widetilde{\mu}^{(N)}_{K^{(N)}}\right]\\
&+\sum_{j=1}^{K^{(N)}}\overline{D}^{(N)}_{K^{(N)},j}\widetilde{\lambda}^{(N)}_j-\frac{1}{\omega^{(N)}_{K^{(N)}}}\left[\widetilde{\lambda}^{(N)}_{K^{(N)}}-\boldsymbol{r}'_{x_f}\ \nu-\varphi'_{x_f}\right].
\end{aligned}$$

Repeating the same approach with $\Xi_i^{(n)}(w, \widetilde{\eta}, \widetilde{\mu}) = 0$ for $n \in [N],\ i \in [K^{(n)} - 1]$ yields

$$\begin{aligned}\frac{\Xi_i^{(n)}(w, \widetilde{\eta}, \widetilde{\mu})}{\omega_i^{(n)}} =& \frac{h}{2}\frac{h_n}{2}\left[\psi_x'\left[\tau_i^{(n)}\right] + f_x'\left[\tau_i^{(n)}\right]\widetilde{\lambda}_i^{(n)} + c_x'\left[\tau_i^{(n)}\right]\widetilde{\mu}_i^{(n)}\right] - \sum_{j=1}^{K^{(n)}} \frac{\omega_j^{(n)}}{\omega_i^{(n)}} D_{j,i}^{(n)}\widetilde{\lambda}_j^{(n)} \\ =& \frac{h}{2}\frac{h_n}{2}\left[\psi_x'\left[\tau_i^{(n)}\right] + f_x'\left[\tau_i^{(n)}\right]\widetilde{\lambda}_i^{(n)} + c_x'\left[\tau_i^{(n)}\right]\widetilde{\mu}_i^{(n)}\right] + \sum_{j=1}^{K^{(n)}} \overline{D}_{i,j}^{(n)}\widetilde{\lambda}_j^{(n)}.\end{aligned}$$

Due to the matching conditions we can write the optimality conditions in $\boldsymbol{F}(w) = 0$ also as follows:

$$\begin{aligned}&\sum_{j=0}^{K^{(1)}} x_j^{(1)} D_{i,j}^{(1)} - \frac{h_1}{2}\frac{h}{2} f\left[\tau_i^{(1)}\right] = 0, \\ &\sum_{j=1}^{K^{(n)}} x_j^{(n)} D_{i,j}^{(n)} + x_{K^{(n-1)}}^{(n-1)} D_{i,0}^{(n)} - \frac{h_n}{2}\frac{h}{2} f\left[\tau_i^{(n)}\right] = 0, \qquad 2 \leqslant n \leqslant N,\ i \in [K^{(n)}].\end{aligned}$$

This step makes it easier to compare the equations with respective equations in Section 9.3.

9.3 First Optimize, Then Discretize: Petrov–Galerkin Approach

In order to solve the infinite dimensional optimality conditions, which were derived in Section 9.1.2, numerically, we have to approximate the function spaces $\mathcal{Y}^l([-1,+1],\mathbb{R})$ and $\mathcal{NBV}([-1,+1],\mathbb{R})$ by finite dimensional subspaces. These subspaces are called *Finite Element (FE) spaces* and the method, which transfers the infinite dimensional conditions into a finite dimensional variational system of equalities and inequalities, is called Petrov–Galerkin *approximation.*

We start with a specification of the basis functions to span the finite dimensional subspaces. Afterwards, applying the Petrov–Galerkin approach we derive a system of equalities and inequalities.

9.3.1 Finite Element Spaces

First we discretize $\mathcal{Y}^l([-1,+1],\mathbb{R})$, $l \in \{0,1\}$, as well as $\mathcal{NBV}([-1,+1],\mathbb{R})$ by choosing appropriate sets of basis functions. This procedure is common to FE methods which are usually used to solve PDEs.

It seems reasonable to choose *spline functions* in order to fully discretize the infinite dimensional function spaces $\mathcal{Y}([-1,+1],\mathbb{R})$ and $\mathcal{Y}^1([-1,+1],\mathbb{R})$. Spline functions are functions which are defined piecewise by polynomials and, in some circumstances, are glued together in a smooth way.

Uniform Degree Splines We carry over the notation from the definition of the function spaces $\mathcal{Y}^l([-1,+1],\mathbb{R})$, i.e., we assume a fixed temporal grid $t_s = t_0 < t_1 < \ldots < t_N = t_f$ and

the $\mathcal{I}_n$ denote the single intervals of the grid. We define the set of *break–points* as those points of the grid which separate the (piecewise defined) polynomials, i.e., we introduce the break–point set $\xi \stackrel{\text{def}}{=} \{t_n\}_{n=1}^{N-1}$. If the polynomial degree is denoted by k we can define the function space of *piecewise polynomials* as

$$\mathcal{S}(k,\xi) \stackrel{\text{def}}{=} \left\{ \boldsymbol{x} : \mathcal{T} \longrightarrow \mathbb{R} \,:\, \boldsymbol{x} \restriction_{\mathcal{I}_n} \in \mathcal{P}^{(k)}(\mathcal{I}_n, \mathbb{R}),\ 1 \leqslant n \leqslant N \right\}.$$

We find relevant subspaces of $\mathcal{S}(k,\xi)$ by imposing *homogeneity conditions* additionally. In our case we consider homogeneous conditions which guarantee a certain number of continuous derivatives globally. To this end, let us consider the $N-1$–dimensional vector $\nu \stackrel{\text{def}}{=} (\nu_n)_{n=1}^{N-1}$ of nonnegative integers counting the number of continuity conditions at the break–points. In particular, $\nu_n = 0$ means that no continuity conditions is imposed across the respective break–point. In order to express the homogeneous conditions by mathematical tools we introduce the notation

$$\operatorname{jmp}_t(\boldsymbol{x}) \stackrel{\text{def}}{=} \boldsymbol{x}(t^+) - \boldsymbol{x}(t^-), \qquad t \in \xi,$$

representing the jump of $\boldsymbol{x}(\cdot)$ at any break–point t. Hence, the homogeneous conditions for a function $\boldsymbol{x} \in \mathcal{S}(k,\xi)$ with homogeneous conditions, expressed by ν, can be written in the form

$$\operatorname{jmp}_{t_n}\left(\boldsymbol{x}^{(j-1)}\right) = 0, \quad n \in [N], \quad j \in [\nu_n].$$

Consequently, the function space of *uniform degree splines* (of degree k) is defined as

$$\mathcal{S}(k,\xi,\nu) \stackrel{\text{def}}{=} \left\{ \boldsymbol{x} : \mathcal{T} \to \mathbb{R} \,:\, \boldsymbol{x} \restriction_{\mathcal{I}_n} \in \mathcal{P}^{(k)}(\mathcal{I}_n, \mathbb{R}),\ \operatorname{jmp}_{t_n}\left(\boldsymbol{x}^{(j-1)}\right) = 0,\ n \in [N],\ j \in [\nu_n] \right\}.$$

In order to determine a basis for $\mathcal{S}(k,\xi,\nu)$ one has to construct a linearly independent function sequence $\varphi_1, \varphi_2, \ldots$ with as many entries as there are necessary to satisfy the interval–wise polynomial degree conditions as well as all the homogeneous conditions. Our function is then represented in the form

$$\sum_j x_j \varphi_j.$$

Note that any function $\boldsymbol{x} \in \mathcal{S}(k,\xi,\nu)$ belongs to a certain subspace of $\mathcal{S}(k,\xi)$, namely the one satisfying the homogeneous conditions. In order to derive a basis for the function space $\mathcal{S}(k,\xi,\nu)$ we define

$$(t-\tau)_+ \stackrel{\text{def}}{=} \max\{t-\tau, 0\}. \tag{9.41}$$

With the aid of (9.41) we define the *truncated power function*

$$(t)_+^s \stackrel{\text{def}}{=} (t_+)^s, \qquad s = 0, 1, 2, \ldots.$$

The function $\boldsymbol{x}(t) \stackrel{\text{def}}{=} (t-\tau)_+^s$ is a piecewise polynomial of order $s+1$. It has just one break–point at τ and is continuous at τ for $s > 0$, whereas for $s = 0$ it has a jump at τ of size 1. Since it holds $\frac{\mathrm{d}}{\mathrm{d}t}(t-\tau)_+^s = s \cdot (t-\tau)_+^{s-1}$ one can see that that $(\cdot - \tau)_+^s$ has $s-1$ continuous derivatives and a jump in the s–th derivative of size $s!$.

We define linear functionals $\Lambda_{nj}(\cdot)$ and corresponding functions $\varphi_{nj}(\cdot)$ according to

$$\begin{aligned} \Lambda_{nj}(\boldsymbol{x}) &\stackrel{\text{def}}{=} \begin{cases} \boldsymbol{x}^{(j)}(t_0), & n = 0, \\ \operatorname{jmp}_{t_n}\left(\boldsymbol{x}^{(j)}\right), & n \in [N-1], \end{cases} \\ \varphi_{nj}(t) &\stackrel{\text{def}}{=} \begin{cases} (t-t_0)^j/j!, & n = 0, \\ (t-t_n)_+^j/j!, & n \in [N-1], \end{cases} \end{aligned} \tag{9.42}$$

with $j = 0, 1, \ldots, k-1$. It is obvious that each $\varphi_{nj}(\cdot)$ is in $\mathcal{S}(k, \xi)$. Furthermore, from the previous discussion about derivatives of the truncated power functions we conclude that

$$\Lambda_{nj}(\varphi_{mi}) = \delta_{nm}\,\delta_{ji} = \begin{cases} 1, & \text{if } n = m \text{ and } j = i, \\ 0, & \text{otherwise,} \end{cases} \tag{9.43}$$

showing the linear independence of the double sequence (φ_{nj}). It is a simple exercise to show that $\mathcal{S}(k, \xi)$ has dimension kN and that (φ_{nj}) is a basis of $\mathcal{S}(k, \xi)$. Consequently, every $\boldsymbol{x} \in \mathcal{S}(k, \xi)$ can be uniquely represented in the form

$$\boldsymbol{x} = \sum_{nj} \Lambda_{nj}(\boldsymbol{x})\,\varphi_{nj}.$$

By means of (9.42) and (9.43) we can write this representation explicitly as

$$\boldsymbol{x}(t) = \sum_{j<k} \boldsymbol{x}^{(j)}(t_0)\,(t-t_0)^j/j! + \sum_{n=1}^{N-1} \sum_{j<k} \operatorname{jmp}_{t_n}\left(\boldsymbol{x}^{(j)}\right)(t-t_n)_+^j/j!.$$

This representation is quite beneficial since all jumps of the derivatives of function $\boldsymbol{x}(\cdot)$ across the break–points $t \in \xi$ appear explicitly as coefficients. The enforcement of all homogeneous conditions

$$\operatorname{jmp}_{t_n}\left(\boldsymbol{x}^{(j-1)}\right) = 0, \quad n \in [N-1], \quad j \in [\,\nu_n],$$

can be achieved in a rather straightforward way, namely by a restriction to those functions where respective coefficients vanish. Hence, every function $\boldsymbol{x}(\cdot)$ in $\mathcal{S}(k, \xi, \nu)$ can be written uniquely as

$$\boldsymbol{x} = \sum_{n=0}^{N-1} \sum_{j=\nu_n}^{k-1} x_{nj}\,\varphi_{nj},$$

where we set $\nu_0 = 0$. Even though we have found a basis for the function space of piece-

wise polynomials it suffers from several drawbacks, for our specific purposes as well as for a numerical realization:

(i) Since the basis functions have no local support the function $\boldsymbol{x}(\cdot)$ evaluated at a certain point of the domain interval may involve a considerable amount of coefficients for large N.

(ii) The basis functions (φ_{nj}) may become nearly linear dependent for rather nonuniform chosen grid points resulting in bad conditioning (compare also the discussion about the monomial basis in Section 6.3.2).

(iii) The lack of local support of the (φ_{nj}) implies that it can be just used for piecewise polynomials with the same degree k on every single interval.

The common way to overcome the lack of local support and the bad conditioning is to use so–called *basis splines* or *B–splines*, originally proposed by Schoenberg [390] and Curry and Schoenberg [118]. A good introduction about B–splines including their most relevant properties and efficient implementations is given by de Boor [127]. A B–spline is a spline having minimal support with respect to a certain degree, smoothness, and grid. By a linear combination of B–splines of a certain degree one can express any spline of that degree. B–splines, constructed from truncated power functions in combination with a divided differences approach, suffer from numerical instabilities. A recurrence relation allowing for the evaluation of B–splines in an efficient and numerically stable way was developed independently from each other by de Boor [126] and Cox [116].
For our purposes, we need an extension to the standard concept of splines allowing for piecewise polynomials of different degrees which are glued together with certain smoothness. This is done in the following.

Multi–Degree Splines As we have just pointed out, conventional splines are intended as function spaces where every piece is spanned by polynomials of the same degree. By way of contrast, we need so–called *Multi–Degree Splines (MD–Splines)*, i.e., piecewise polynomial functions comprised of distinct degrees. To construct MD–Splines, we propose the following setting: for the grid $t_s = t_0 < t_1 < \ldots < t_N = t_f$ we define the set of break–points as $\xi \stackrel{\text{def}}{=} \{t_n\}_{n=1}^{N-1}$. The vector $K \stackrel{\text{def}}{=} [K^{(1)}, \ldots, K^{(N)}]^T$ holds the polynomial degrees for the pieces $\mathcal{I}_n$. Two adjacent polynomials defined respectively on $\mathcal{I}_n$ and $\mathcal{I}_{n+1}$ join at the break point t_n with continuity $\mathcal{C}^{\nu_n}$. The ν_n are chosen according to

$$0 \leqslant \nu_n \leqslant \begin{cases} \min(K^{(n)}, K^{(n+1)}), & \text{if } K^{(n)} \neq K^{(n+1)}, \\ K^{(n+1)} - 1, & \text{if } K^{(n)} = K^{(n+1)}. \end{cases}$$

The degree of smoothness is then determined by the vector $\nu \stackrel{\text{def}}{=} [\nu_1, \ldots, \nu_{N-1}]^T$. The set of MD–Splines $\mathcal{S}(K, \xi, \nu)$ is defined as follows:

Definition 9.6 (Multi–Degree Splines)
Let ξ denote the inner grid points of a partition of the compact interval $\mathcal{T} = [t_s, t_f]$, and let ν be the associated degrees of smoothness. Let furthermore K denote the sequence of polynomial degrees associated

with the partition intervals $\mathcal{I}_n$. Then the set of *Multi–Degree Splines (MD–Splines)* is given by

$$\mathcal{S}(K,\xi,\nu) \stackrel{\text{def}}{=} \Big\{ \boldsymbol{x} : \mathcal{T} \longrightarrow \mathbb{R} : \exists\, \boldsymbol{P}_n \in \mathcal{P}_{K^{(n)}},\ n \in [N], \text{such that:}$$
$$\text{i) } \boldsymbol{x}(t) = \boldsymbol{P}_n(t) \text{ for } t \in \mathcal{I}_n,\ n \in [N],$$
$$\text{ii) } \operatorname{jmp}_{t_n}\left(\boldsymbol{x}^{(j-1)}\right) = 0, \quad n \in [N-1], \quad j \in [\nu_n] \Big\}. \qquad \triangle$$

The dimension of the spline space $\mathcal{S}(K,\xi,\nu)$, which we denote with S, can be derived from standard arguments and can be expressed in two different ways:

$$S = \begin{cases} K^{(1)} + 1 + K_s, & K_s \stackrel{\text{def}}{=} \sum_{n=1}^{N-1} \left(K^{(n+1)} - \nu_n\right), \\ K^{(N)} + 1 + K_e, & K_e \stackrel{\text{def}}{=} \sum_{n=1}^{N-1} \left(K^{(n)} - \nu_n\right). \end{cases}$$

In order to construct the B–spline basis for the MD–Spline function space $\mathcal{S}(K,\xi,\nu)$ we introduce two grid point sets $\xi_s = \{s_j\}_{j=1}^{S}$ and $\xi_e = \{e_j\}_{j=1}^{S}$. These two sets are constructed such that the i–th B–spline basis function $N_{i,M}$, where $M \stackrel{\text{def}}{=} \max_n\{K^{(n)}\}$, has support $\operatorname{supp}(N_{i,M}) = [s_i, e_i]$. For this reason, we define:

Definition 9.7 (Extended Partition)
The *left extended partition* associated with $\mathcal{S}(K,\xi,\nu)$ is defined as the set of grid points $\xi_s = \{s_j\}_{j=1}^{S}$, where

(i) $s_1 \leqslant s_2 \leqslant \ldots \leqslant s_S$;

(ii) $s_{K^{(1)}+1} \equiv t_{\mathrm{s}}$;

(iii) $\{s_{K^{(1)}+2}, \ldots, s_S\} \equiv \{\underbrace{t_1, \ldots, t_1}_{K^{(2)}-\nu_1 \text{ times}}, \ldots, \underbrace{t_{N-1}, \ldots, t_{N-1}}_{K^{(N)}-\nu_{N-1} \text{ times}}\}$.

In a similar way the *right extended partition* associated with $\mathcal{S}(K,\xi,\nu)$ is defined as the set of grid points $\xi_e = \{e_j\}_{j=1}^{S}$, where

(i) $e_1 \leqslant e_2 \leqslant \ldots \leqslant e_S$;

(ii) $e_{S-K^{(N)}} \equiv t_{\mathrm{f}}$;

(iii) $\{e_1, \ldots, e_{S-K^{(N)}-1}\} \equiv \{\underbrace{t_1, \ldots, t_1}_{K^{(1)}-\nu_1 \text{ times}}, \ldots, \underbrace{t_{N-1}, \ldots, t_{N-1}}_{K^{(N-1)}-\nu_{N-1} \text{ times}}\}$. $\triangle$

We restrict our discussion to the case of so–called *clamped partitions* where all elements from $\{s_j\}_{j=1}^{K^{(1)}+1}$ are chosen to be t_{s} and all elements from $\{e_j\}_{j=S-K^{(N)}}^{S}$ are chosen to be t_{f}. The integral recurrence relation described in the following definition derives the set $\{N_{i,M}\}_{i=1}^{S}$ of multi–degree B–spline functions.

Definition 9.8 (Multi–Degree B–Spline Functions)
Let $M \stackrel{\text{def}}{=} \max_n\{K^{(n)}\}$. For each $m = 0, \ldots, M$, the function sequence $\{N_{i,m}\}$, $i = M+1-m, \ldots, S$, which is defined on each break–point interval $\mathcal{I}_n \subset [s_i, e_{i-M+m}]$ with $s_i < e_{i-M+m}$, has support $\operatorname{supp}(N_{i,m}) = [s_i, e_{i-M+m}]$, and is recursively generated according to

$$N_{i,m}(t) \stackrel{\text{def}}{=} \begin{cases} 0, & K^{(n)} < M-m, \\ \begin{cases} 1, & t \in \mathcal{I}_n \\ 0, & \text{otherwise} \end{cases}, & K^{(n)} = M-m, \\ \int_{-\infty}^{t} \left\{\delta_{i,m-1} N_{i,m-1}(\tau) - \delta_{i+1,m-1} N_{i+1,m-1}(\tau)\right\} \mathrm{d}\tau, & K^{(n)} > M-m, \end{cases}$$

where

$$\delta_{i,m} \stackrel{\text{def}}{=} \left(\int_{-\infty}^{+\infty} N_{i,m}(t)\,\mathrm{d}t \right)^{-1}. \qquad \triangle$$

In Definition 9.8 we regard undefined functions $N_{i,m}(\cdot)$ as the zero function. Additionally, in case of $N_{i,m} = 0$ we define $\delta_{i,m} N_{i,m} \stackrel{\text{def}}{=} 0$. However, in order to be able to obtain the partition of unity, $\delta_{i,m} N_{i,m}(\cdot)$ should fulfill the formula $\int_{-\infty}^{+\infty} \delta_{i,m} N_{i,m}(t)\,\mathrm{d}t = 1$. For this reason, whenever $N_{i,m} = 0$, we define

$$\int_{-\infty}^{t} \delta_{i,m} N_{i,m}(\tau)\,\mathrm{d}\tau \stackrel{\text{def}}{=} \begin{cases} 0, & t < s_i, \\ 1, & t \geqslant e_i. \end{cases}$$

In the following result we summarize the most important properties of the B–spline functions.

Theorem 9.9 (Properties of the B–spline Functions)
For the B–spline functions $\{N_{i,M}\}_{i=1}^{S}$ of the MD–Spline space $\mathcal{S}(K, \xi, \nu)$ the following properties hold:

(i) Linear Independence: the $\{N_{i,M}\}_i$ are linear independent;

(ii) Local Support: $N_{i,M}(t) = 0$ for $t \notin [s_i, e_i]$;

(iii) Positivity: $N_{i,M}(t) > 0$ for $t \in (s_i, e_i)$;

(iv) Normalization: $\sum_i N_{i,M}(t) = 1$, $\forall\, t \in \mathcal{T}$. $\triangle$

Proof See e.g. Beccari et al. [36] and Shen and Wang [410]. $\square$

Taking into account that the dimension of the spline space is equal to the number of basis functions $\{N_{i,M}\}_i$ we conclude from Theorem 9.9 that they represent a basis of $\mathcal{S}(K, \xi, \nu)$. For this reason, we can write any MD–Spline function $\boldsymbol{x} \in \mathcal{S}(K, \xi, \nu)$ as a linear combination of B–spline basis functions $N_{i,M}$, $1 \leqslant i \leqslant S$, by means of associated coefficients x_i in the following way:

$$\boldsymbol{x}(t) = \sum_{i=1}^{S} x_i N_{i,M}(t), \qquad t \in \mathcal{T}.$$

Discretization of $\mathcal{Y}^k([-1,+1],\mathbb{R})$ The MD–Splines allow us to find reasonable discretizations of the function spaces $\mathcal{Y}^k([-1,+1],\mathbb{R})$, $k = 0, 1$. The space $\mathcal{Y}([-1,+1],\mathbb{R})$ after discretization consists of the splines without homogeneous conditions and polynomial degrees $K^{(n)} - 1$, $n \in [N]$. Accordingly, we define the function space

$$\mathcal{Y}_{\mathcal{P}}(\mathcal{T},\mathbb{R}) \stackrel{\text{def}}{=} \left\{ \boldsymbol{x} : \mathcal{T} \longrightarrow \mathbb{R} \,:\, \boldsymbol{x} \restriction_{\mathcal{I}_n} \in \mathcal{P}_{K^{(n)}-1}(\mathcal{I}_n,\mathbb{R}),\ n \in [N] \right\}.$$

In order to derive a tailored representation for our purposes we introduce the Lagrange polynomials

$$\overline{\boldsymbol{L}}_j^{(n)}(\tau) \stackrel{\text{def}}{=} \prod_{\substack{i=1\\ i\neq j}}^{K^{(n)}} \frac{\tau-\tau_i^{(n)}}{\tau_j^{(n)}-\tau_i^{(n)}}, \quad \deg\left(\overline{\boldsymbol{L}}_j^{(n)}\right) = K^{(n)}-1, \quad n\in[N],\ j\in[K^{(n)}],$$

where the $\{\tau_i\}$ denote FLGR points. Furthermore, we need the time transformation functions $\boldsymbol{t}_n(\cdot)$ as well as the characteristic functions

$$\mathcal{X}^{(n)}(t) = \begin{cases} 1, & \text{if } t\in\mathcal{I}_n,\\ 0, & \text{otherwise.}\end{cases} \tag{9.44}$$

Then any function $\boldsymbol{X}\in\mathcal{Y}_{\mathcal{P}}(\mathcal{T},\mathbb{R})$ can be expressed by means of coefficients $x_i^{(n)}$ as

$$\boldsymbol{X}(t) = \sum_{n=1}^{N} \mathcal{X}^{(n)}(t) \sum_{i=1}^{K^{(n)}} x_i^{(n)} \overline{\boldsymbol{L}}_i^{(n)}(\boldsymbol{t}_n^{-1}(t)).$$

Due to construction it holds

$$\boldsymbol{X}(\boldsymbol{t}_n(\tau_i^{(n)})) = x_i^{(n)}, \quad 1\leqslant n\leqslant N,\ i\in[K^{(n)}].$$

A discretization of the space $\mathcal{Y}^1([-1,+1],\mathbb{R})$ results in the function space of all splines with polynomial degrees $K^{(n)}$, $n\in[N]$. Additionally, continuity over the full horizon is imposed such that we define the respective function space as

$$\mathcal{Y}^1_{\mathcal{P}}(\mathcal{T},\mathbb{R}) \stackrel{\text{def}}{=} \left\{\boldsymbol{x}\in\mathcal{C}(\mathcal{T},\mathbb{R}) : \boldsymbol{x}\restriction_{\mathcal{I}_n}\in\mathcal{P}_{K^{(n)}}(\mathcal{I}_n,\mathbb{R}),\ n\in[N]\right\}.$$

For a tailored representation we introduce the Lagrange polynomials

$$\boldsymbol{L}_j^{(n)}(\tau) \stackrel{\text{def}}{=} \prod_{\substack{i=0\\ i\neq j}}^{K^{(n)}} \frac{\tau-\tau_i^{(n)}}{\tau_j^{(n)}-\tau_i^{(n)}}, \quad \deg\left(\boldsymbol{L}_j^{(n)}\right) = K^{(n)}, \quad n\in[N],\ 0\leqslant j\leqslant K^{(n)},$$

where $\tau_0=-1$ and the $\{\tau_i\}$, $i\in[K^{(n)}]$, denote FLGR points. Using the characteristic functions (9.44) as well as the time transformation functions $\boldsymbol{t}_n(\cdot)$ we can write any function $\boldsymbol{X}\in\mathcal{Y}^1_{\mathcal{P}}(\mathcal{T},\mathbb{R})$ with the aid of coefficients $x_i^{(n)}$ as follows:

$$\boldsymbol{X}(t) = \mathcal{X}^{(1)}(t)\sum_{i=0}^{K^{(n)}} x_i^{(1)}\boldsymbol{L}_i^{(1)}(\boldsymbol{t}_1^{-1}(t)) + \sum_{n=2}^{N}\mathcal{X}^{(n)}(t)\left\{x_{K^{(n-1)}}^{(n-1)}\boldsymbol{L}_0^{(n)}(\boldsymbol{t}_n^{-1}(t)) + \sum_{i=1}^{K^{(n)}} x_i^{(n)}\boldsymbol{L}_i^{(n)}(\boldsymbol{t}_n^{-1}(t))\right\}.$$

For later use we calculate:

$$\boldsymbol{X}(\boldsymbol{t}_n(\tau_i^{(n)})) = x_i^{(n)}, \qquad 1\leqslant n\leqslant N,\ i\in[K^{(n)}],$$

$$\boldsymbol{X}(\boldsymbol{t}_n(\tau_0^{(n)})) = x_{K^{(n-1)}}^{(n-1)}, \quad 2 \leqslant n \leqslant N,$$
$$\boldsymbol{X}(\boldsymbol{t}_1(\tau_0^{(1)})) = x_0^{(1)}.$$

The derivative of $\boldsymbol{X}(\cdot)$ is given as

$$\dot{\boldsymbol{X}}(t) = \mathcal{X}^{(1)}(t) \sum_{i=0}^{K^{(n)}} \frac{2}{h_1} x_i^{(1)} \dot{\boldsymbol{L}}_i^{(1)}(\boldsymbol{t}_1^{-1}(t)) + \sum_{n=2}^{N} \mathcal{X}^{(n)}(t) \frac{2}{h_n} \left\{ x_{K^{(n-1)}}^{(n-1)} \dot{\boldsymbol{L}}_0^{(n)}(\boldsymbol{t}_n^{-1}(t)) + \sum_{i=1}^{K^{(n)}} x_i^{(n)} \dot{\boldsymbol{L}}_i^{(n)}(\boldsymbol{t}_n^{-1}(t)) \right\}.$$

With the usual notation $D_{i,j}^{(n)} \stackrel{\text{def}}{=} \dot{\boldsymbol{L}}_j^{(n)}(\tau_i^{(n)})$, we evaluate $\dot{\boldsymbol{x}}(\cdot)$ at some relevant points:

$$\dot{\boldsymbol{X}}(\boldsymbol{t}_1(t_i^{(1)})) = \sum_{j=0}^{K^{(1)}} \frac{2}{h_1} x_j^{(1)} D_{i,j}^{(1)}, \qquad i \in [K^{(1)}],$$

$$\dot{\boldsymbol{X}}(\boldsymbol{t}_n(t_i^{(n)})) = \frac{2}{h_n} x_{K^{(n-1)}}^{(n-1)} D_{i,0}^{(n)} + \sum_{j=1}^{K^{(n)}} \frac{2}{h_n} x_j^{(n)} D_{i,j}^{(n)}, \quad 2 \leqslant n \leqslant N,\ i \in [K^{(n)}].$$

Note that the extension to the general case $\mathcal{Y}^k(\mathcal{T}, \mathbb{R})$ for $k > 1$ is straightforward, leading to finite dimensional function spaces $\mathcal{Y}_{\mathcal{P}}^k(\mathcal{T}, \mathbb{R})$. The differences in these function spaces is that they have a certain number of continuous derivatives. However, imposing these homogeneous conditions is fully covered by the earlier presented MD–Spline function spaces.

Discretization of $\mathcal{NBV}([-1, +1], \mathbb{R})$ We pursue the concept proposed by Beigel [41] and Beigel et al. [42] in order to discretize the function space $\mathcal{NBV}(\mathcal{T}, \mathbb{R})$ whose discretization we will denote with $\mathcal{Z}_H(\mathcal{T}, \mathbb{R})$. We recall the temporal grid $t_s = t_0 < t_1 < \ldots < t_N = t_f$ and introduce a subgrid on every single interval $\mathcal{I}_n = [t_{n-1}, t_n]$ whose notation looks as follows:

$$t_{n-1} = t_0^{(n)} < t_1^{(n)} < \ldots < t_{K^{(n)}}^{(n)} = t_n, \qquad n \in [N].$$

The $t_0^{(n)}$ and $t_i^{(n)}$, $n \in [N]$, $i \in [K^{(n)}]$ are the images of $\tau_0^{(n)} = -1$ and FLGR points $\tau_i^{(n)}$ under the time transformation function $\boldsymbol{t}_n(\cdot)$. The function space $\mathcal{Z}_H(\mathcal{T}, \mathbb{R})$ consists of functions being constant on intervals $\mathcal{I}_i^{(n)} \stackrel{\text{def}}{=} [t_{i-1}^{(n)}, t_i^{(n)}]$ ($n \in [N]$, $i \in [K^{(n)}]$), i.e., we define

$$\mathcal{Z}_H(\mathcal{T}, \mathbb{R}) \stackrel{\text{def}}{=} \left\{ \boldsymbol{x} : \mathcal{T} \longrightarrow \mathbb{R} : \boldsymbol{x} \restriction_{\mathcal{I}_i^{(n)}} \in \mathcal{P}_0(\mathcal{I}_i^{(n)}, \mathbb{R}),\ n \in [N],\ i \in [K^{(n)}] \right\}.$$

To express a function $\boldsymbol{X} \in \mathcal{Z}_H(\mathcal{T}, \mathbb{R})$ we make use of Heaviside functions

$$\mathcal{H}_i^{(n)}(\tau) = \begin{cases} 0, & \tau < \tau_i^{(n)} \\ 1, & \tau \geqslant \tau_i^{(n)} \end{cases}, \qquad n \in [N],\ i \in [K^{(n)}],$$

as functions which are chosen to be continuous from the right with discontinuities in $\tau_i^{(n)}$. By means of coefficients $x_i^{(n)}$ any $\boldsymbol{X} \in \mathcal{Z}_H(\mathcal{T}, \mathbb{R})$ can be expressed as a linear combination

$$\boldsymbol{X}(t) = \sum_{n=1}^{N} \sum_{i=1}^{K^{(n)}} x_i^{(n)} \, \mathcal{H}_i^{(n)}(\boldsymbol{t}_n^{-1}(t)). \tag{9.45}$$

Note that $\boldsymbol{X}(\cdot)$ is a step function with initial value $\boldsymbol{X}(t_s) = 0$ and jumps of magnitude $x_i^{(n)}$ at $t_i^{(n)}$ for $n \in [N]$ and $i \in [K^{(n)}]$. For this reason we conclude that $\boldsymbol{X}(t_i^{(n)}) = \boldsymbol{X}(t_{i-1}^{(n)}) + x_i^{(n)}$ at the grid points and $\boldsymbol{X}(t) = \boldsymbol{X}(t_i^{(n)})$ for inner points $t \in (t_i^{(n)}, t_{i+1}^{(n)})$, $i \in [K^{(n)} - 1]$. Furthermore, we have $\boldsymbol{X}(t_0^{(n+1)}) = \boldsymbol{X}(t_{K^{(n)}}^{(n)}) + x_{K^{(n)}}^{(n)}$, $n \in [N-1]$. Here we recall that the classical derivative of $\boldsymbol{X}(\cdot)$ does not exist. However, $\boldsymbol{X}(\cdot)$ is differentiable in a weak sense with weak derivatives given by Dirac measures at $\{t_i^{(n)}\}$ with heights $\{x_i^{(n)}\}$ (see Example 2.40).

Discretization of Trial and Test Spaces Let us consider the trial functions first. The discretization of trial function $\boldsymbol{x} \in \mathcal{Y}^1([-1,+1], \mathbb{R}^{n_x})$ is expressed in terms of coefficients $x_i^{(n)}$ as

$$\boldsymbol{x}_h(t) \stackrel{\text{def}}{=} \mathcal{X}^{(1)}(t) \sum_{i=0}^{K^{(n)}} x_i^{(1)} \boldsymbol{L}_i^{(1)}(\boldsymbol{t}_1^{-1}(t)) + \sum_{n=2}^{N} \mathcal{X}^{(n)}(t) \left\{ x_{K^{(n-1)}}^{(n-1)} \boldsymbol{L}_0^{(n)}(\boldsymbol{t}_n^{-1}(t)) + \sum_{i=1}^{K^{(n)}} x_i^{(n)} \boldsymbol{L}_i^{(n)}(\boldsymbol{t}_n^{-1}(t)) \right\},$$

and the discretization of trial function $\boldsymbol{u} \in \mathcal{Y}([-1,+1], \mathbb{R}^{n_x})$ is written with coefficients $u_i^{(n)}$ as

$$\boldsymbol{u}_h(t) \stackrel{\text{def}}{=} \sum_{n=1}^{N} \mathcal{X}^{(n)}(t) \sum_{i=1}^{K^{(n)}} u_i^{(n)} \overline{\boldsymbol{L}}_i^{(n)}(\boldsymbol{t}_n^{-1}(t)).$$

The trial functions associated with dual states $\boldsymbol{\Lambda} \in \mathcal{NBV}([-1,+1], \mathbb{R}^{n_x})$ of the differential equation and with dual states $\boldsymbol{M} \in \mathcal{NBV}([-1,+1], \mathbb{R}^{n_c})$ of the mixed control–state path constraints are represented with coefficients $\Lambda_i^{(n)}$ and $M_i^{(n)}$ as

$$\boldsymbol{\Lambda}_h(t) \stackrel{\text{def}}{=} \sum_{n=1}^{N} \sum_{i=1}^{K^{(n)}} \Lambda_i^{(n)} \, \mathcal{H}_i^{(n)}(\boldsymbol{t}_n^{-1}(t)), \tag{9.46}$$

$$\mathbf{M}_h(t) \stackrel{\text{def}}{=} \sum_{n=1}^{N} \sum_{i=1}^{K^{(n)}} M_i^{(n)} \, \mathcal{H}_i^{(n)}(\boldsymbol{t}_n^{-1}(t)). \tag{9.47}$$

For later convenience and similarly to as we have done in Section 9.2 with coefficients of the Lagrange function we parametrize $\Lambda_i^{(n)}$ and $M_i^{(n)}$ according to the formula

$$\Lambda_i^{(n)} = \frac{h_n}{2} \, \omega_i^{(n)} \, \widetilde{\lambda}_i^{(n)}, \qquad M_i^{(n)} = \frac{h_n}{2} \, \omega_i^{(n)} \, \widetilde{\mu}_i^{(n)}, \tag{9.48}$$

such that we have

$$\mathbf{\Lambda}_{\mathrm{h}}(t) \stackrel{\text{def}}{=} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \widetilde{\lambda}_i^{(n)}\, \omega_i^{(n)}\, \mathcal{H}_i^{(n)}(\boldsymbol{t}_n^{-1}(t)),$$
$$\mathbf{M}_{\mathrm{h}}(t) \stackrel{\text{def}}{=} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \widetilde{\mu}_i^{(n)}\, \omega_i^{(n)}\, \mathcal{H}_i^{(n)}(\boldsymbol{t}_n^{-1}(t)).$$

Gaussian quadrature with FLGR quadrature points suggests the following step function approximation of the identity function $\mathbf{id}(\cdot)$:

$$\boldsymbol{T}^{\mathrm{h}}(t) \stackrel{\text{def}}{=} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)}\, \mathcal{H}_i^{(n)}(\boldsymbol{t}_n^{-1}(t)). \tag{9.49}$$

Now we deal with discretizations of the test functions. We introduce the notation $\boldsymbol{\varphi}_x^h(\cdot)$ for the discretized test functions $\boldsymbol{\varphi}_x \in \mathcal{Y}^1([-1,+1], \mathbb{R}^{n_{\mathrm{x}}})$ and define them as follows:

$$\boldsymbol{\varphi}_x^h(t) \stackrel{\text{def}}{=} \mathcal{X}^{(1)}(t) \sum_{i=0}^{K^{(n)}} \varphi_{x_i}^{(1)}\, \boldsymbol{L}_i^{(1)}(\boldsymbol{t}_1^{-1}(t)) + \sum_{n=2}^{N} \mathcal{X}^{(n)}(t) \left\{ \varphi_{x_{K^{(n-1)}}}^{(n-1)}\, \boldsymbol{L}_0^{(n)}(\boldsymbol{t}_n^{-1}(t)) + \sum_{i=1}^{K^{(n)}} \varphi_{x_i}^{(n)}\, \boldsymbol{L}_i^{(n)}(\boldsymbol{t}_n^{-1}(t)) \right\}.$$

In a similar way we deal with $\boldsymbol{\varphi}_u \in \mathcal{Y}([-1,+1], \mathbb{R}^{n_{\mathrm{x}}})$ and $\boldsymbol{\varphi}_\mu \in \mathcal{Y}([-1,+1], \mathbb{R}^{n_{\mathrm{x}}})$ whose discretizations are given as

$$\boldsymbol{\varphi}_u^h(t) \stackrel{\text{def}}{=} \sum_{n=1}^{N} \mathcal{X}^{(n)}(t) \sum_{i=1}^{K^{(n)}} \varphi_{u_i}^{(n)}\, \overline{\boldsymbol{L}}_i^{(n)}(\boldsymbol{t}_n^{-1}(t))$$

and

$$\boldsymbol{\varphi}_\mu^{\mathrm{h}}(t) \stackrel{\text{def}}{=} \sum_{n=1}^{N} \mathcal{X}^{(n)}(t) \sum_{i=1}^{K^{(n)}} \varphi_{\mu_i}^{(n)}\, \overline{\boldsymbol{L}}_i^{(n)}(\boldsymbol{t}_n^{-1}(t)).$$

9.3.2 Finite Dimensional Optimality Conditions

In the following we approximate the infinite dimensional functions of the weak formulations in Section 9.1.2 with appropriately chosen finite dimensional functions. Evaluating the resulting variational formulations provides us with optimality conditions.

Optimality Conditions $[(9.5)+(9.6)+(9.7)]$ We start by an investigation of the variational formulation (9.18) and replace the trial functions $\boldsymbol{x}(\cdot)$ and $\boldsymbol{u}(\cdot)$ with $\boldsymbol{x}_{\mathrm{h}}(\cdot)$ and $\boldsymbol{u}_{\mathrm{h}}(\cdot)$, respectively. Likewise we replace $\mathbf{id}(\cdot)$ with $\boldsymbol{T}^{\mathrm{h}}(\cdot)$, $\mathbf{\Lambda}(\cdot)$ with $\mathbf{\Lambda}_{\mathrm{h}}(\cdot)$, and $\mathbf{M}(\cdot)$ with $\mathbf{M}_{\mathrm{h}}(\cdot)$. Finally, the test function $\boldsymbol{\varphi}_x(\cdot)$ is substituted with the finite dimensional function $\boldsymbol{\varphi}_x^h(\cdot)$ leading to

$$0 = \sum_{n=1}^{N} \left\{ \frac{h}{2} \int_{(-1,1]} \boldsymbol{\psi}_x'[\boldsymbol{t}_n(t)]\, \boldsymbol{\varphi}_x^h(\boldsymbol{t}_n(t))\, \mathrm{d}\, \boldsymbol{T}^{\mathrm{h}}(\boldsymbol{t}_n(t)) + \int_{(-1,1]} \frac{h}{2}\, \boldsymbol{f}_x'[\boldsymbol{t}_n(t)]\, \boldsymbol{\varphi}_x^h(\boldsymbol{t}_n(t)) - \dot{\boldsymbol{\varphi}}_x^h(\boldsymbol{t}_n(t))\, \mathrm{d}\, \mathbf{\Lambda}_{\mathrm{h}}(\boldsymbol{t}_n(t)) \right.$$

$$+\frac{h}{2}\int_{(-1,1]} \boldsymbol{c}_x'[\boldsymbol{t}_n(t)]\boldsymbol{\varphi}_x^h(\boldsymbol{t}_n(t))\,\mathrm{d}\mathbf{M}_{\mathrm{h}}(\boldsymbol{t}_n(t))\Bigg\}$$
$$+\left(\boldsymbol{\varphi}_x^h\right)^T(+1)\left(\boldsymbol{r}_{x_f}'\ \nu+\boldsymbol{\varphi}_{x_f}'\right)+\left(\boldsymbol{\varphi}_x^h\right)^T(-1)\left(\boldsymbol{r}_{x_s}'\ \nu+\boldsymbol{\varphi}_{x_s}'\right) \tag{9.50}$$

for all $\boldsymbol{\varphi}_x^h \in \mathcal{Y}_{\mathcal{P}}^1([-1,+1],\mathbb{R}^{n_{\mathrm{x}}})$. We evaluate the integrals interval–wise (note the specific evaluation for jump functions in Lemma 2.82) such that we have for

(i) interval $n=1$:

$$\frac{h}{2}\left\{\sum_{i=1}^{K^{(1)}}\frac{h_1}{2}\,\omega_i^{(1)}\,\boldsymbol{\psi}_x'^{\,T}\left[\tau_i^{(1)}\right]\varphi_{x_i}^{(1)}\right\}+\frac{h}{2}\left\{\sum_{i=1}^{K^{(1)}}\frac{h_1}{2}\,\omega_i^{(1)}\left(\widetilde{\mu}_i^{(1)}\right)^T\boldsymbol{c}_x'^{\,T}\left[\tau_i^{(1)}\right]\varphi_{x_i}^{(1)}\right\}$$
$$+\left\{\sum_{i=1}^{K^{(1)}}\frac{h_1}{2}\,\omega_i^{(1)}\left(\widetilde{\lambda}_i^{(1)}\right)^T\left\{\frac{h}{2}\boldsymbol{f}_x'^{\,T}\left[\tau_i^{(1)}\right]\varphi_{x_i}^{(1)}-\sum_{j=0}^{K^{(1)}}\frac{2}{h_1}\,\varphi_{x_j}^{(1)}\,D_{i,j}^{(1)}\right\}\right\}.$$

(ii) intervals $n\in[N]\setminus\{1\}$:

$$\frac{h}{2}\left\{\sum_{i=1}^{K^{(n)}}\frac{h_n}{2}\,\omega_i^{(n)}\boldsymbol{\psi}_x'^{\,T}\left[\tau_i^{(n)}\right]\varphi_{x_i}^{(n)}\right\}+\frac{h}{2}\left\{\sum_{i=1}^{K^{(n)}}\frac{h_n}{2}\,\omega_i^{(n)}\left(\widetilde{\mu}_i^{(n)}\right)^T\boldsymbol{c}_x'^{\,T}\left[\tau_i^{(n)}\right]\varphi_{x_i}^{(n)}\right\}$$
$$+\left\{\sum_{i=1}^{K^{(n)}}\frac{h_n}{2}\,\omega_i^{(n)}\left(\widetilde{\lambda}_i^{(n)}\right)^T\left\{\frac{h}{2}\boldsymbol{f}_x'^{\,T}\left[\tau_i^{(n)}\right]\varphi_{x_i}^{(n)}-\sum_{j=1}^{K^{(n)}}\frac{2}{h_n}\,\varphi_{x_j}^{(n)}\,D_{i,j}^{(n)}-\frac{2}{h_n}\,\varphi_{x_{K^{(n-1)}}}^{(n-1)}\,D_{i,0}^{(n)}\right\}\right\}.$$

(iii) the rest:

$$\varphi_{x_{K^{(N)}}}^{N\ T}\left(\boldsymbol{r}_{x_f}'\ \nu+\boldsymbol{\varphi}_{x_f}'\right)+\varphi_{x_0}^{(1)^T}\left(\boldsymbol{r}_{x_s}'\ \nu+\boldsymbol{\varphi}_{x_s}'\right).$$

Rearranging the sums yields for

(i) interval $n=1$:

$$\sum_{i=1}^{K^{(1)}}\frac{h}{2}\frac{h_1}{2}\,\omega_i^{(1)}\,\boldsymbol{\psi}_x'^{\,T}\left[\tau_i^{(1)}\right]\varphi_{x_i}^{(1)}+\sum_{i=1}^{K^{(1)}}\frac{h}{2}\frac{h_1}{2}\,\omega_i^{(1)}\left(\widetilde{\mu}_i^{(1)}\right)^T\boldsymbol{c}_x'^{\,T}\left[\tau_i^{(1)}\right]\varphi_{x_i}^{(1)}$$
$$-\sum_{i=1}^{K^{(1)}}\sum_{j=0}^{K^{(1)}}\omega_i^{(1)}\left(\widetilde{\lambda}_i^{(1)}\right)^T\varphi_{x_j}^{(1)}\,D_{i,j}^{(1)}+\sum_{i=1}^{K^{(1)}}\frac{h}{2}\frac{h_1}{2}\,\omega_i^{(1)}\left(\widetilde{\lambda}_i^{(1)}\right)^T\boldsymbol{f}_x'^{\,T}\left[\tau_i^{(1)}\right]\varphi_{x_i}^{(1)}.$$

(ii) intervals $n\in[N]\setminus\{1\}$:

$$\sum_{i=1}^{K^{(n)}}\frac{h}{2}\frac{h_n}{2}\,\omega_i^{(n)}\,\boldsymbol{\psi}_x'^{\,T}\left[\tau_i^{(n)}\right]\varphi_{x_i}^{(n)}+\sum_{i=1}^{K^{(n)}}\frac{h}{2}\frac{h_n}{2}\,\omega_i^{(n)}\left(\widetilde{\mu}_i^{(n)}\right)^T\boldsymbol{c}_x'^{\,T}\left[\tau_i^{(n)}\right]\varphi_{x_i}^{(n)}$$

$$
\begin{aligned}
&-\sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left(\widetilde{\lambda}_i^{(n)}\right)^T \varphi_{x_{K^{(n-1)}}}^{(n-1)} D_{i,0}^{(n)} - \sum_{i=1}^{K^{(n)}} \sum_{j=1}^{K^{(n)}} \omega_i^{(n)} \left(\widetilde{\lambda}_i^{(n)}\right)^T \varphi_{x_j}^{(n)} D_{i,j}^{(n)} \\
&+\sum_{i=1}^{K^{(n)}} \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \left(\widetilde{\lambda}_i^{(n)}\right)^T \boldsymbol{f}_x'^T \left[\tau_i^{(n)}\right] \varphi_{x_i}^{(n)}.
\end{aligned}
$$

Another rearrangement step yields for

(i) interval $n = 1$:

$$
\begin{aligned}
&\sum_{i=1}^{K^{(1)}} \frac{h}{2} \frac{h_1}{2} \omega_i^{(1)} \boldsymbol{\psi}_x'^T \left[\tau_i^{(1)}\right] \varphi_{x_i}^{(1)} + \sum_{i=1}^{K^{(1)}} \frac{h}{2} \frac{h_1}{2} \omega_i^{(1)} \left(\widetilde{\mu}_i^{(1)}\right)^T \boldsymbol{c}_x'^T \left[\tau_i^{(1)}\right] \varphi_{x_i}^{(1)} \\
&-\sum_{i=1}^{K^{(1)}} \sum_{j=1}^{K^{(1)}} \omega_j^{(1)} \left(\widetilde{\lambda}_j^{(1)}\right)^T \varphi_{x_i}^{(1)} D_{j,i}^{(1)} - \sum_{i=1}^{K^{(1)}} \omega_i^{(1)} \left(\widetilde{\lambda}_i^{(1)}\right)^T \varphi_{x_0}^{(1)} D_{i,0}^{(1)} \\
&+\sum_{i=1}^{K^{(1)}} \frac{h}{2} \frac{h_1}{2} \omega_i^{(1)} \left(\widetilde{\lambda}_i^{(1)}\right)^T \boldsymbol{f}_x'^T \left[\tau_i^{(1)}\right] \varphi_{x_i}^{(1)}.
\end{aligned}
$$

(ii) intervals $n \in [N] \setminus \{1\}$:

$$
\begin{aligned}
&\sum_{i=1}^{K^{(n)}} \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \boldsymbol{\psi}_x'^T \left[\tau_i^{(n)}\right] \varphi_{x_i}^{(n)} + \sum_{i=1}^{K^{(n)}} \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \left(\widetilde{\mu}_i^{(n)}\right)^T \boldsymbol{c}_x'^T \left[\tau_i^{(n)}\right] \varphi_{x_i}^{(n)} \\
&-\sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left(\widetilde{\lambda}_i^{(n)}\right)^T \varphi_{x_{K^{(n-1)}}}^{(n-1)} D_{i,0}^{(n)} - \sum_{i=1}^{K^{(n)}} \sum_{j=1}^{K^{(n)}} \omega_j^{(n)} \left(\widetilde{\lambda}_j^{(n)}\right)^T \varphi_{x_i}^{(n)} D_{j,i}^{(n)} \\
&+\sum_{i=1}^{K^{(n)}} \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \left(\widetilde{\lambda}_i^{(n)}\right)^T \boldsymbol{f}_x'^T \left[\tau_i^{(n)}\right] \varphi_{x_i}^{(n)}.
\end{aligned}
$$

Gluing everything together we can write (9.50) as

$$
\begin{aligned}
0 = &\sum_{i=1}^{K^{(1)}} \frac{h}{2} \frac{h_1}{2} \omega_i^{(1)} \boldsymbol{\psi}_x'^T \left[\tau_i^{(1)}\right] \varphi_{x_i}^{(1)} + \sum_{i=1}^{K^{(1)}} \frac{h}{2} \frac{h_1}{2} \omega_i^{(1)} \left(\widetilde{\mu}_i^{(1)}\right)^T \boldsymbol{c}_x'^T \left[\tau_i^{(1)}\right] \varphi_{x_i}^{(1)} \\
&-\sum_{i=1}^{K^{(1)}} \sum_{j=1}^{K^{(1)}} \omega_j^{(1)} \left(\widetilde{\lambda}_j^{(1)}\right)^T \varphi_{x_i}^{(1)} D_{j,i}^{(1)} - \sum_{i=1}^{K^{(1)}} \omega_i^{(1)} \left(\widetilde{\lambda}_i^{(1)}\right)^T \varphi_{x_0}^{(1)} D_{i,0}^{(1)} \\
&+\sum_{i=1}^{K^{(1)}} \frac{h}{2} \frac{h_1}{2} \omega_i^{(1)} \left(\widetilde{\lambda}_i^{(1)}\right)^T \boldsymbol{f}_x'^T \left[\tau_i^{(1)}\right] \varphi_{x_i}^{(1)} \\
&+\sum_{n=2}^{N} \sum_{i=1}^{K^{(n)}} \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \boldsymbol{\psi}_x'^T \left[\tau_i^{(n)}\right] \varphi_{x_i}^{(n)} + \sum_{n=2}^{N} \sum_{i=1}^{K^{(n)}} \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \left(\widetilde{\mu}_i^{(n)}\right)^T \boldsymbol{c}_x'^T \left[\tau_i^{(n)}\right] \varphi_{x_i}^{(n)}
\end{aligned}
$$

$$
\begin{aligned}
&- \sum_{n=1}^{N-1} \sum_{i=1}^{K^{(n+1)}} \omega_i^{(n+1)} \left(\widetilde{\lambda}_i^{(n+1)}\right)^T \varphi_{x_{K^{(n)}}}^{(n)} D_{i,0}^{(n+1)} - \sum_{n=2}^{N} \sum_{i=1}^{K^{(n)}} \sum_{j=1}^{K^{(n)}} \omega_j^{(n)} \left(\widetilde{\lambda}_j^{(n)}\right)^T \varphi_{x_i}^{(n)} D_{j,i}^{(n)} \\
&+ \sum_{n=2}^{N} \sum_{i=1}^{K^{(n)}} \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \left(\widetilde{\lambda}_i^{(n)}\right)^T \boldsymbol{f}_x'^T \left[\tau_i^{(n)}\right] \varphi_{x_i}^{(n)} \\
&+ \varphi_{x_{K^{(N)}}}^{N\ T} \left(\boldsymbol{r}'_{x_f}\ \nu + \boldsymbol{\varphi}'_{x_f}\right) + \varphi_{x_0}^{(1)^T} \left(\boldsymbol{r}'_{x_s}\ \nu + \boldsymbol{\varphi}'_{x_s}\right) \qquad \forall \varphi_{x_i}^{(n)}.
\end{aligned}
$$

Rearranging terms yields

$$
\begin{aligned}
0 = &\sum_{n=1}^{N} \sum_{i=1}^{K^{(n)}-1} \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \boldsymbol{\psi}_x'^T \left[\tau_i^{(n)}\right] \varphi_{x_i}^{(n)} + \sum_{n=1}^{N} \sum_{i=1}^{K^{(n)}-1} \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \left(\widetilde{\mu}_i^{(n)}\right)^T \boldsymbol{c}_x'^T \left[\tau_i^{(n)}\right] \varphi_{x_i}^{(n)} \\
&+ \sum_{n=1}^{N} \sum_{i=1}^{K^{(n)}-1} \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \left(\widetilde{\lambda}_i^{(n)}\right)^T \boldsymbol{f}_x'^T \left[\tau_i^{(n)}\right] \varphi_{x_i}^{(n)} - \sum_{n=1}^{N} \sum_{i=1}^{K^{(n)}-1} \sum_{j=1}^{K^{(n)}} \omega_j^{(n)} \left(\widetilde{\lambda}_j^{(n)}\right)^T \varphi_{x_i}^{(n)} D_{j,i}^{(n)} \\
&+ \sum_{n=1}^{N} \frac{h}{2} \frac{h_n}{2} \omega_{K^{(n)}}^{(n)} \boldsymbol{\psi}_x'^T \left[\tau_{K^{(n)}}^{(n)}\right] \varphi_{x_{K^{(n)}}}^{(n)} + \sum_{n=1}^{N} \frac{h}{2} \frac{h_n}{2} \omega_{K^{(n)}}^{(n)} \left(\widetilde{\mu}_{K^{(n)}}^{(n)}\right)^T \boldsymbol{c}_x'^T \left[\tau_{K^{(n)}}^{(n)}\right] \varphi_{x_{K^{(n)}}}^{(n)} \\
&+ \sum_{n=1}^{N} \frac{h}{2} \frac{h_n}{2} \omega_{K^{(n)}}^{(n)} \left(\widetilde{\lambda}_{K^{(n)}}^{(n)}\right)^T \boldsymbol{f}_x'^T \left[\tau_{K^{(n)}}^{(n)}\right] \varphi_{x_{K^{(n)}}}^{(n)} - \sum_{n=1}^{N} \sum_{j=1}^{K^{(n)}} \omega_j^{(n)} \left(\widetilde{\lambda}_j^{(n)}\right)^T \varphi_{x_{K^{(n)}}}^{(n)} D_{j,K^{(n)}}^{(n)} \\
&- \sum_{i=1}^{K^{(1)}} \omega_i^{(1)} \left(\widetilde{\lambda}_i^{(1)}\right)^T \varphi_{x_0}^{(1)} D_{i,0}^{(1)} - \sum_{n=1}^{N-1} \sum_{i=1}^{K^{(n+1)}} \omega_i^{(n+1)} \left(\widetilde{\lambda}_i^{(n+1)}\right)^T \varphi_{x_{K^{(n)}}}^{(n)} D_{i,0}^{(n+1)} \\
&+ \left(\varphi_{x_{K^{(N)}}}^{N}\right)^T \left(\boldsymbol{r}'_{x_f}\ \nu + \boldsymbol{\varphi}'_{x_f}\right) + \left(\varphi_{x_0}^{(1)}\right)^T \left(\boldsymbol{r}'_{x_s}\ \nu + \boldsymbol{\varphi}'_{x_s}\right) \qquad \forall \varphi_{x_i}^{(n)}.
\end{aligned}
$$

We solve the equation for the $\varphi_{x_i}^{(n)}$ and obtain

$$
\begin{aligned}
0 = &\varphi_{x_0}^{(1)^T} \left\{ \boldsymbol{r}'_{x_s}\ \nu + \boldsymbol{\varphi}'_{x_s} - \sum_{j=1}^{K^{(1)}} \omega_j^{(1)} D_{j,0}^{(1)} \widetilde{\lambda}_j^{(1)} \right\} \\
&+ \sum_{n=1}^{N} \sum_{i=1}^{K^{(n)}-1} \varphi_{x_i}^{(n)^T} \left\{ \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \left[\boldsymbol{\psi}'_x \left[\tau_i^{(n)}\right] + \boldsymbol{f}'_x \left[\tau_i^{(n)}\right] \widetilde{\lambda}_i^{(n)} + \boldsymbol{c}'_x \left[\tau_i^{(n)}\right] \widetilde{\mu}_i^{(n)}\right] \right. \\
&\left. - \sum_{j=1}^{K^{(n)}} \omega_j^{(n)} D_{j,i}^{(n)} \widetilde{\lambda}_j^{(n)} \right\} \\
&+ \sum_{n=1}^{N-1} \varphi_{x_{K^{(n)}}}^{(n)\ T} \left\{ \frac{h}{2} \frac{h_n}{2} \omega_{K^{(n)}}^{(n)} \left[\boldsymbol{\psi}'_x \left[\tau_{K^{(n)}}^{(n)}\right] + \boldsymbol{f}'_x \left[\tau_{K^{(n)}}^{(n)}\right] \widetilde{\lambda}_{K^{(n)}}^{(n)} + \boldsymbol{c}'_x \left[\tau_{K^{(n)}}^{(n)}\right] \widetilde{\mu}_{K^{(n)}}^{(n)}\right] \right. \\
&\left. - \sum_{j=1}^{K^{(n)}} \omega_j^{(n)} D_{j,K^{(n)}}^{(n)} \widetilde{\lambda}_j^{(n)} - \sum_{j=1}^{K^{(n+1)}} \omega_j^{(n+1)} D_{j,0}^{(n+1)} \widetilde{\lambda}_j^{(n+1)} \right\}
\end{aligned}
$$

$$+ \varphi_{x_{K^{(N)}}^{(N)}}^{T} \left\{ \frac{h}{2} \frac{h_N}{2} \omega_{K^{(N)}}^{(N)} \left[\psi_x' \left[\tau_{K^{(N)}}^{(N)} \right] + f_x' \left[\tau_{K^{(N)}}^{(N)} \right] \widetilde{\lambda}_{K^{(N)}}^{(N)} + c_x' \left[\tau_{K^{(N)}}^{(N)} \right] \widetilde{\mu}_{K^{(N)}}^{(N)} \right] \right.$$
$$\left. - \sum_{j=1}^{K^{(N)}} \omega_j^{(N)} D_{j,K^{(N)}}^{(N)} \widetilde{\lambda}_j^{(N)} + r_{x_f}' \ \nu + \varphi_{x_f}' \right\} \qquad \forall \varphi_{x_i}^{(n)}. \tag{9.51}$$

The function $\widetilde{\Xi} : \mathbb{R}^{n_w} \times \mathbb{R}^{n_\eta} \times \mathbb{R}^{n_\mu} \longrightarrow \mathbb{R}^{n_s}$ reads as

$$\Xi(w, \widetilde{\eta}, \widetilde{\mu}) = \left[\left(\widetilde{\Xi}_0^{(1)} \right)^T, \ldots, \left(\widetilde{\Xi}_{K^{(1)}}^{(1)} \right)^T, \ldots, \left(\widetilde{\Xi}_{x_1^{(N)}} \right)^T, \ldots, \left(\widetilde{\Xi}_{K^{(N)}}^{(N)} \right)^T \right]^T (w, \widetilde{\eta}, \widetilde{\mu}),$$

where the single components are described in the following. For $n \in [N]$ and $i \in [K^{(n)} - 1]$ we define

$$\Xi_i^{(n)}(w, \widetilde{\eta}, \widetilde{\mu}) \overset{\text{def}}{=} \frac{h}{2} \frac{h_n}{2} \left\{ \psi_x' \left[\tau_i^{(n)} \right] + f_x' \left[\tau_i^{(n)} \right] \widetilde{\lambda}_i^{(n)} + c_x' \left[\tau_i^{(n)} \right] \widetilde{\mu}_i^{(n)} \right\} + \sum_{j=1}^{K^{(n)}} \overline{D}_{i,j}^{(n)} \widetilde{\lambda}_j^{(n)}.$$
$$= \frac{h}{2} \frac{h_n}{2} \left\{ \psi_x' \left[\tau_i^{(n)} \right] + f_x' \left[\tau_i^{(n)} \right] \widetilde{\lambda}_i^{(n)} + c_x' \left[\tau_i^{(n)} \right] \widetilde{\mu}_i^{(n)} \right\} - \sum_{j=1}^{K^{(n)}} \frac{\omega_j^{(n)}}{\omega_i^{(n)}} D_{j,i}^{(n)} \widetilde{\lambda}_j^{(n)}.$$

For this reason we have

$$\omega_i^{(n)} \Xi_i^{(n)}(w, \widetilde{\eta}, \widetilde{\mu}) = \frac{h}{2} \frac{h_n}{2} \omega_i^{(n)} \left\{ \psi_x' \left[\tau_i^{(n)} \right] + f_x' \left[\tau_i^{(n)} \right] \widetilde{\lambda}_i^{(n)} + c_x' \left[\tau_i^{(n)} \right] \widetilde{\mu}_i^{(n)} \right\}$$
$$- \sum_{j=1}^{K^{(n)}} \omega_j^{(n)} D_{j,i}^{(n)} \widetilde{\lambda}_j^{(n)}, \qquad n \in [N], \ i \in [K^{(n)} - 1].$$

For $n = N$ and $i = K^{(N)}$ we define the respective component as

$$\widetilde{\Xi}_{K^{(N)}}^{(N)}(w, \widetilde{\eta}, \widetilde{\mu}) \overset{\text{def}}{=} \frac{h}{2} \frac{h_N}{2} \left[\psi_x' \left[\tau_{K^{(N)}}^{(N)} \right] + f_x' \left[\tau_{K^{(N)}}^{(N)} \right] \widetilde{\lambda}_{K^{(N)}}^{(N)} + c_x' \left[\tau_{K^{(N)}}^{(N)} \right] \widetilde{\mu}_{K^{(N)}}^{(N)} \right]$$
$$+ \sum_{j=1}^{K^{(N)}} \overline{D}_{K^{(N)},j}^{(N)} \widetilde{\lambda}_j^{(N)} - \frac{1}{\omega_{K^{(N)}}^{(N)}} \left[\widetilde{\lambda}_{K^{(N)}}^{(N)} - r_{x_f}' \ \nu - \varphi_{x_f}' \right]$$
$$= \frac{h}{2} \frac{h_N}{2} \left[\psi_x' \left[\tau_{K^{(N)}}^{(N)} \right] + f_x' \left[\tau_{K^{(N)}}^{(N)} \right] \widetilde{\lambda}_{K^{(N)}}^{(N)} + c_x' \left[\tau_{K^{(N)}}^{(N)} \right] \widetilde{\mu}_{K^{(N)}}^{(N)} \right]$$
$$- \sum_{j=1}^{K^{(N)}} \frac{\omega_j^{(N)}}{\omega_{K^{(N)}}^{(N)}} D_{j,K^{(N)}}^{(N)} \widetilde{\lambda}_j^{(N)} + \frac{1}{\omega_{K^{(N)}}^{(N)}} \left[r_{x_f}' \ \nu + \varphi_{x_f}' \right]$$

such that it holds

$$\omega_{K^{(N)}}^{(N)} \widetilde{\Xi}_{K^{(N)}}^{(N)}(w, \widetilde{\eta}, \widetilde{\mu}) = \frac{h}{2} \frac{h_N}{2} \omega_{K^{(N)}}^{(N)} \left[\psi_x' \left[\tau_{K^{(N)}}^{(N)} \right] + f_x' \left[\tau_{K^{(N)}}^{(N)} \right] \widetilde{\lambda}_{K^{(N)}}^{(N)} + c_x' \left[\tau_{K^{(N)}}^{(N)} \right] \widetilde{\mu}_{K^{(N)}}^{(N)} \right]$$

$$-\sum_{j=1}^{K^{(N)}} \omega_j^{(N)} D_{j,K^{(N)}}^{(N)} \widetilde{\lambda}_j^{(N)} + \boldsymbol{r}'_{x_f} \ \nu + \boldsymbol{\varphi}'_{x_f}.$$

For $n = 1$ and $i = 0$ the respective component is defined as

$$\begin{aligned}
\widetilde{\Xi}_0^{(1)}(w, \widetilde{\eta}, \widetilde{\mu}) &\stackrel{\text{def}}{=} -\sum_{i=1}^{K^{(1)}} \omega_i^{(1)} \sum_{j=1}^{K^{(1)}} \overline{D}_{i,j}^{(1)} \widetilde{\lambda}_j^{(1)} + \widetilde{\lambda}_{K^{(1)}}^{(1)} + \boldsymbol{r}'_{x_s} \ \nu + \boldsymbol{\varphi}'_{x_s} \\
&= -\sum_{i=1}^{K^{(1)}-1} \omega_i^{(1)} \left[\sum_{j=1}^{K^{(1)}} \frac{\omega_j^{(1)}}{\omega_i^{(1)}} \overline{D}_{j,i}^{(1)} \right] \widetilde{\lambda}_i^{(1)} - \omega_{K^{(1)}}^{(1)} \left[\sum_{j=1}^{K^{(1)}} \frac{\omega_j^{(1)}}{\omega_{K^{(1)}}^{(1)}} \overline{D}_{j,K^{(1)}}^{(1)} \right] \widetilde{\lambda}_{K^{(1)}}^{(1)} \\
&\quad + \frac{\omega_{K^{(1)}}^{(1)}}{\omega_{K^{(1)}}^{(1)}} \widetilde{\lambda}_{K^{(1)}}^{(1)} + \boldsymbol{r}'_{x_s} \ \nu + \boldsymbol{\varphi}'_{x_s} \\
&= -\sum_{i=1}^{K^{(1)}} \omega_i^{(1)} D_{i,0}^{(1)} \widetilde{\lambda}_i^{(1)} + \boldsymbol{r}'_{x_s} \ \nu + \boldsymbol{\varphi}'_{x_s}.
\end{aligned}$$

Finally, for $n \in [N-1]$ and $i = K^{(n)}$ we define the respective component as

$$\begin{aligned}
\widetilde{\Xi}_{K^{(n)}}^{(n)}(w, \widetilde{\eta}, \widetilde{\mu}) \stackrel{\text{def}}{=} & \frac{h}{2} \frac{h_n}{2} \left[\boldsymbol{\psi}'_x \left[\tau_{K^{(n)}}^{(n)} \right] + \boldsymbol{f}'_x \left[\tau_{K^{(n)}}^{(n)} \right] \widetilde{\lambda}_{K^{(n)}}^{(n)} + \boldsymbol{c}'_x \left[\tau_{K^{(n)}}^{(n)} \right] \widetilde{\mu}_{K^{(n)}}^{(n)} \right] + \sum_{j=1}^{K^{(n)}} \overline{D}_{K^{(n)},j}^{(n)} \widetilde{\lambda}_j^{(n)} \\
& - \frac{1}{\omega_{K^{(n)}}^{(n)}} \left[\widetilde{\lambda}_{K^{(n)}}^{(n)} - \widetilde{\lambda}_{K^{(n+1)}}^{(n+1)} + \sum_{i=1}^{K^{(n+1)}} \omega_i^{(n+1)} \sum_{j=1}^{K^{(n+1)}} \overline{D}_{i,j}^{(n+1)} \widetilde{\lambda}_j^{(n+1)} \right].
\end{aligned}$$

Taking the auxiliary calculation

$$\begin{aligned}
& \sum_{i=1}^{K^{(n+1)}} \omega_i^{(n+1)} \sum_{j=1}^{K^{(n+1)}} \overline{D}_{i,j}^{(n+1)} \widetilde{\lambda}_j^{(n+1)} - \widetilde{\lambda}_{K^{(n+1)}}^{(n+1)} \\
&= \sum_{i=1}^{K^{(n+1)}-1} \omega_i^{(n+1)} \left[\sum_{j=1}^{K^{(n+1)}} \frac{\omega_j^{(n+1)}}{\omega_i^{(n+1)}} \overline{D}_{j,i}^{(n+1)} \right] \widetilde{\lambda}_i^{(n+1)} \\
&\quad + \omega_{K^{(n+1)}}^{(n+1)} \left[\sum_{j=1}^{K^{(n+1)}} \frac{\omega_j^{(n+1)}}{\omega_{K^{(n+1)}}^{(n+1)}} \overline{D}_{j,K^{(n+1)}}^{(n+1)} - \frac{1}{\omega_{K^{(n+1)}}^{(n+1)}} \right] \widetilde{\lambda}_{K^{(n+1)}}^{(n+1)} \\
&= \sum_{i=1}^{K^{(n+1)}-1} \omega_i^{(n+1)} D_{i,0}^{(n+1)} \widetilde{\lambda}_i^{(n+1)} + \omega_{K^{(n+1)}}^{(n+1)} D_{K^{(n+1)},0}^{(n+1)} \widetilde{\lambda}_{K^{(n+1)}}^{(n+1)} = \sum_{i=1}^{K^{(n+1)}} \omega_i^{(n+1)} D_{i,0}^{(n+1)} \widetilde{\lambda}_i^{(n+1)}
\end{aligned}$$

into consideration we deduce

$$\begin{aligned}\widetilde{\Xi}^{(n)}_{K^{(n)}}(w,\widetilde{\eta},\widetilde{\mu}) =& \frac{h}{2}\frac{h_n}{2}\left[\psi'_x\left[\tau^{(n)}_{K^{(n)}}\right]+f'_x\left[\tau^{(n)}_{K^{(n)}}\right]\widetilde{\lambda}^{(n)}_{K^{(n)}}+c'_x\left[\tau^{(n)}_{K^{(n)}}\right]\widetilde{\mu}^{(n)}_{K^{(n)}}\right]\\ &+\sum_{j=1}^{K^{(n)}}\overline{D}^{(n)}_{K^{(n)},j}\,\widetilde{\lambda}^{(n)}_j-\frac{1}{\omega^{(n)}_{K^{(n)}}}\left[\widetilde{\lambda}^{(n)}_{K^{(n)}}+\sum_{i=1}^{K^{(n+1)}}\omega^{(n+1)}_i D^{(n+1)}_{i,0}\,\widetilde{\lambda}^{(n+1)}_i\right]\\ =&\frac{h}{2}\frac{h_n}{2}\left[\psi'_x\left[\tau^{(n)}_{K^{(n)}}\right]+f'_x\left[\tau^{(n)}_{K^{(n)}}\right]\widetilde{\lambda}^{(n)}_{K^{(n)}}+c'_x\left[\tau^{(n)}_{K^{(n)}}\right]\widetilde{\mu}^{(n)}_{K^{(n)}}\right]\\ &-\sum_{j=1}^{K^{(n)}}\frac{\omega^{(n)}_j}{\omega^{(n)}_{K^{(n)}}}D^{(n)}_{j,K^{(n)}}\,\widetilde{\lambda}^{(n)}_j-\sum_{j=1}^{K^{(n+1)}}\frac{\omega^{(n+1)}_j}{\omega^{(n)}_{K^{(n)}}}D^{(n+1)}_{j,0}\,\widetilde{\lambda}^{(n+1)}_j.\end{aligned}$$

Then it holds

$$\begin{aligned}\omega^{(n)}_{K^{(n)}}\widetilde{\Xi}^{(n)}_{K^{(n)}}(w,\widetilde{\eta},\widetilde{\mu}) =& \frac{h}{2}\frac{h_n}{2}\omega^{(n)}_{K^{(n)}}\left[\psi'_x\left[\tau^{(n)}_{K^{(n)}}\right]+f'_x\left[\tau^{(n)}_{K^{(n)}}\right]\widetilde{\lambda}^{(n)}_{K^{(n)}}+c'_x\left[\tau^{(n)}_{K^{(n)}}\right]\widetilde{\mu}^{(n)}_{K^{(n)}}\right]\\ &-\sum_{j=1}^{K^{(n)}}\omega^{(n)}_j D^{(n)}_{j,K^{(n)}}\,\widetilde{\lambda}^{(n)}_j-\sum_{j=1}^{K^{(n+1)}}\omega^{(n+1)}_j D^{(n+1)}_{j,0}\,\widetilde{\lambda}^{(n+1)}_j.\end{aligned}$$

Now we define the vector

$$\varphi_x \overset{\text{def}}{=}\left[\left(\varphi_x{}^{(1)}\right)^T,\ldots,\left(\varphi_x{}^{(N)}\right)^T\right]^T,$$

where the single components are given as

$$\begin{aligned}\varphi_x{}^{(1)} &\overset{\text{def}}{=}\left[\omega^{(1)}_0\left(\varphi_{x^{(1)}_0}\right)^T,\ldots,\omega^{(1)}_{K^{(1)}}\left(\varphi_{x^{(1)}_{K^{(1)}}}\right)^T\right]^T,\quad \omega^{(1)}_0\equiv 1,\\ \varphi_x{}^{(n)} &\overset{\text{def}}{=}\left[\omega^{(n)}_1\left(\varphi_{x^{(n)}_1}\right)^T,\ldots,\omega^{(n)}_{K^{(n)}}\left(\varphi_{x^{(n)}_{K^{(n)}}}\right)^T\right]^T,\quad n\in[N]\setminus\{1\}.\end{aligned}$$

We can write (9.51) as the system of equations

$$\varphi_x{}^T\,\widetilde{\Xi}(w,\widetilde{\eta},\widetilde{\mu})=0\quad\forall\varphi_x\in\mathbb{R}^{n_s}.$$

Hence, the equation system holds if $\widetilde{\Xi}(\cdot)$ vanishes. This is equivalent to vanishing components $\widetilde{\Xi}^{(n)}_i(\cdot)=0$ for all i.

Optimality Conditions [(9.8) + (9.9)] We insert the trial functions into Equation (9.23) leading to

$$\begin{aligned}0=&\varphi_{t_s}{}^T\left\{-\frac{1}{2}\sum_{n=1}^{N}\left[\int_{(-1,1]}\psi[t_n(t)]\,\mathrm{d}\,T^{\mathrm{h}}(t_n(t))+\int_{(-1,1]}f[t_n(t)]\,\mathrm{d}\,\Lambda_{\mathrm{h}}(t_n(t))\right]\right.\\ &\left.+\frac{h}{2}\sum_{n=1}^{N}\left[\int_{(-1,1]}\frac{1}{2}(1-t_n(t))\,\psi'_t[t_n(t)]\,\mathrm{d}\,T^{\mathrm{h}}(t_n(t))+\int_{(-1,1]}\frac{1}{2}(1-t_n(t))\,f'_t[t_n(t)]\,\mathrm{d}\,\Lambda_{\mathrm{h}}(t_n(t))\right.\right.\end{aligned}$$

$$+\int_{(-1,1]} \frac{1}{2}\left(1-\boldsymbol{t}_n(t)\right) \boldsymbol{c}_t'[\boldsymbol{t}_n(t)]\,\mathrm{d}\mathbf{M}_\mathrm{h}(\boldsymbol{t}_n(t))\Big] + \boldsymbol{\varphi}_{t_\mathrm{s}}' + \boldsymbol{r}_{t_\mathrm{s}}'\ \nu\Big\}, \quad \forall\, \boldsymbol{\varphi}_{t_\mathrm{s}} \in \mathbb{R},$$

and evaluate the Lebesgue–Stieltjes integrals yielding for all $\boldsymbol{\varphi}_{t_\mathrm{s}} \in \mathbb{R}$:

$$\begin{aligned}0 =& \boldsymbol{\varphi}_{t_\mathrm{s}}{}^T \Bigg\{ -\frac{1}{2}\sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left[\boldsymbol{\psi}\left[\tau_i^{(n)}\right] + \boldsymbol{f}\left[\tau_i^{(n)}\right] \widetilde{\lambda}_i^{(n)}\right] + \boldsymbol{r}_{t_\mathrm{s}}'\ \nu + \boldsymbol{\varphi}_{t_\mathrm{s}}' \\ &+ \frac{1}{2}\frac{h}{2} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left(1-\boldsymbol{t}_n\left(\tau_i^{(n)}\right)\right) \left[\boldsymbol{\psi}_t'\left[\tau_i^{(n)}\right] + \boldsymbol{f}_t'\left[\tau_i^{(n)}\right] \widetilde{\lambda}_i^{(n)} + \boldsymbol{c}_t'\left[\tau_i^{(n)}\right] \widetilde{\mu}_i^{(n)}\right] \Bigg\}.\end{aligned}$$

We introduce the function $\widetilde{\boldsymbol{T}}_{t_\mathrm{s}} : \mathbb{R}^{n_\mathrm{w}} \times \mathbb{R}^{n_\eta} \times \mathbb{R}^{n_\mu} \longrightarrow \mathbb{R}$ which is defined as

$$\begin{aligned}\widetilde{\boldsymbol{T}}_{t_\mathrm{s}}(w, \widetilde{\eta}, \widetilde{\mu}) =& -\frac{1}{2}\sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left\{\boldsymbol{\psi}\left[\tau_i^{(n)}\right] + \boldsymbol{f}\left[\tau_i^{(n)}\right] \widetilde{\lambda}_i^{(n)}\right\} + \boldsymbol{r}_{t_\mathrm{s}}'\ \nu + \boldsymbol{\varphi}_{t_\mathrm{s}}' \\ &+ \frac{1}{2}\frac{h}{2} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left(1-\boldsymbol{t}_n\left(\tau_i^{(n)}\right)\right) \left\{\boldsymbol{\psi}_t'\left[\tau_i^{(n)}\right] + \boldsymbol{f}_t'\left[\tau_i^{(n)}\right] \widetilde{\lambda}_i^{(n)} + \boldsymbol{c}_t'\left[\tau_i^{(n)}\right] \widetilde{\mu}_i^{(n)}\right\}\end{aligned}$$

such that we can write the equation in terms of $\widetilde{\boldsymbol{T}}_{t_\mathrm{s}}(\cdot)$ as

$$0 = \boldsymbol{\varphi}_{t_\mathrm{s}}{}^T\, \widetilde{\boldsymbol{T}}_{t_\mathrm{s}}(w, \widetilde{\eta}, \widetilde{\mu}) \qquad \forall\, \boldsymbol{\varphi}_{t_\mathrm{s}} \in \mathbb{R}.$$

We proceed in a similar way with Equation (9.24) and find

$$\begin{aligned}0 =& \boldsymbol{\varphi}_{t_\mathrm{f}}{}^T \Bigg\{ \frac{1}{2}\sum_{n=1}^{N} \Bigg[\int_{(-1,1]} \boldsymbol{\psi}[\boldsymbol{t}_n(t)]\,\mathrm{d}\,\boldsymbol{T}^\mathrm{h}(\boldsymbol{t}_n(t)) + \int_{(-1,1]} \boldsymbol{f}[\boldsymbol{t}_n(t)]\,\mathrm{d}\,\boldsymbol{\Lambda}_\mathrm{h}(\boldsymbol{t}_n(t)) \Bigg] \\ &+ \frac{h}{2}\sum_{n=1}^{N} \Bigg[\int_{(-1,1]} \frac{1}{2}\left(1+\boldsymbol{t}_n(t)\right) \boldsymbol{\psi}_t'[\boldsymbol{t}_n(t)]\,\mathrm{d}\,\boldsymbol{T}^\mathrm{h}(\boldsymbol{t}_n(t)) + \int_{(-1,1]} \frac{1}{2}\left(1+\boldsymbol{t}_n(t)\right) \boldsymbol{f}_t'[\boldsymbol{t}_n(t)]\,\mathrm{d}\,\boldsymbol{\Lambda}_\mathrm{h}(\boldsymbol{t}_n(t)) \\ &+ \int_{(-1,1]} \frac{1}{2}\left(1+\boldsymbol{t}_n(t)\right) \boldsymbol{c}_t'[\boldsymbol{t}_n(t)]\,\mathrm{d}\mathbf{M}_\mathrm{h}(\boldsymbol{t}_n(t)) \Bigg] + \boldsymbol{\varphi}_{t_\mathrm{f}}' + \boldsymbol{r}_{t_\mathrm{f}}'\ \nu \Bigg\}, \quad \forall\, \boldsymbol{\varphi}_{t_\mathrm{f}} \in \mathbb{R}.\end{aligned}$$

Evaluating the Lebesgue–Stieltjes integrals yields for all $\boldsymbol{\varphi}_{t_\mathrm{f}} \in \mathbb{R}$:

$$\begin{aligned}0 =& \boldsymbol{\varphi}_{t_\mathrm{f}}{}^T \Bigg\{ \frac{1}{2}\sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left[\boldsymbol{\psi}\left[\tau_i^{(n)}\right] + \boldsymbol{f}\left[\tau_i^{(n)}\right] \widetilde{\lambda}_i^{(n)}\right] + \boldsymbol{r}_{t_f}'\ \nu + \boldsymbol{\varphi}_{t_f}' \\ &+ \frac{1}{2}\frac{h}{2} \sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left(1+\boldsymbol{t}_n\left(\tau_i^{(n)}\right)\right) \left[\boldsymbol{\psi}_t'\left[\tau_i^{(n)}\right] + \boldsymbol{f}_t'\left[\tau_i^{(n)}\right] \widetilde{\lambda}_i^{(n)} + \boldsymbol{c}_t'\left[\tau_i^{(n)}\right] \widetilde{\mu}_i^{(n)}\right] \Bigg\}.\end{aligned}$$

By means of the function $\widetilde{\boldsymbol{T}}_{t_\mathrm{f}} : \mathbb{R}^{n_\mathrm{w}} \times \mathbb{R}^{n_\eta} \times \mathbb{R}^{n_\mu} \longrightarrow \mathbb{R}$ which is defined as

$$\boldsymbol{T}_{t_\mathrm{f}}(w, \widetilde{\eta}, \widetilde{\mu}) = +\frac{1}{2}\sum_{n=1}^{N} \frac{h_n}{2} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \left\{\boldsymbol{\psi}\left[\tau_i^{(n)}\right] + \boldsymbol{f}\left[\tau_i^{(n)}\right] \widetilde{\lambda}_i^{(n)}\right\} + \boldsymbol{r}_{t_\mathrm{f}}'\ \nu + \boldsymbol{\varphi}_{t_\mathrm{f}}'$$

$$+\frac{1}{2}\frac{h}{2}\sum_{n=1}^{N}\frac{h_n}{2}\sum_{i=1}^{K^{(n)}}\omega_i^{(n)}\left(1+t_n\left(\tau_i^{(n)}\right)\right)\left\{\psi_t'\left[\tau_i^{(n)}\right]+f_t'\left[\tau_i^{(n)}\right]\widetilde{\lambda}_i^{(n)}+c_t'\left[\tau_i^{(n)}\right]\widetilde{\mu}_i^{(n)}\right\}$$

we write the equation as

$$0=\varphi_{t_f}{}^T\,\widetilde{T}_{t_f}(w,\widetilde{\eta},\widetilde{\mu})\qquad\forall\,\varphi_{t_f}\in\mathbb{R}.$$

Optimality Conditions [(9.10)] We replace trial and test functions in (9.25) and use the fact that the Lebesgue–Stieltjes integral can be easily evaluated for step functions. The resulting equation reads as

$$\begin{aligned}
0=&\sum_{n=1}^{N}\left\{\int_{(-1,1]}\psi_u'[t_n(t)]\,\varphi_u^h(t_n(t))\,\mathrm{d}\,T^{\mathrm{h}}(t_n(t))+\int_{(-1,1]}f_u'[t_n(t)]\,\varphi_u^h(t_n(t))\,\mathrm{d}\,\Lambda_{\mathrm{h}}(t_n(t))\right.\\
&\left.+\int_{(-1,1]}c_u'[t_n(t)]\,\varphi_u^h(t_n(t))\,\mathrm{d}\,\mathbf{M}_{\mathrm{h}}(t_n(t))\right\}\\
=&\sum_{n=1}^{N}\left\{\sum_{i=1}^{K^{(n)}}\frac{h_n}{2}\,\omega_i^{(n)}\,\psi_u'\left[\tau_i^{(n)}\right]\varphi_{u_i}{}^{(n)}+\sum_{i=1}^{K^{(n)}}\frac{h_n}{2}\,\omega_i^{(n)}\left(\widetilde{\lambda}_i^{(n)}\right)^T f_u'\left[\tau_i^{(n)}\right]\varphi_{u_i}{}^{(n)}\right.\\
&\left.+\sum_{i=1}^{K^{(n)}}\frac{h_n}{2}\,\omega_i^{(n)}\left(\widetilde{\mu}_i^{(n)}\right)^T c_u'\left[\tau_i^{(n)}\right]\varphi_{u_i}{}^{(n)}\right\}\\
=&\sum_{n=1}^{N}\sum_{i=1}^{K^{(n)}}\frac{h_n}{2}\,\omega_i^{(n)}\left(\varphi_{u_i}{}^{(n)}\right)^T\left\{\psi_u'{}^T\left[\tau_i^{(n)}\right]+f_u'{}^T\left[\tau_i^{(n)}\right]\widetilde{\lambda}_i^{(n)}+c_u'{}^T\left[\tau_i^{(n)}\right]\widetilde{\mu}_i^{(n)}\right\}.
\end{aligned}\tag{9.52}$$

By introducing the vectors

$$\varphi_u\overset{\text{def}}{=}\left[\left(\varphi_u{}^{(1)}\right)^T,\ldots,\left(\varphi_u{}^{(N)}\right)^T\right]^T,\qquad\varphi_u{}^{(n)}\overset{\text{def}}{=}\left[\left(\varphi_{u1}{}^{(n)}\right)^T,\ldots,\left(\varphi_{uK^{(n)}}{}^{(n)}\right)^T\right]^T,\quad n\in[N],$$

and the vector–valued function $\widetilde{Y}:\mathbb{R}^{n_{\mathrm{w}}}\times\mathbb{R}^{n_\eta}\times\mathbb{R}^{n_\mu}\longrightarrow\mathbb{R}^{n_{\mathrm{q}}}$

$$\widetilde{Y}(w,\widetilde{\eta},\widetilde{\mu})=\left[\left(\widetilde{Y}_1^{(1)}\right)^T,\ldots,\left(\widetilde{Y}_{K^{(1)}}^{(1)}\right)^T,\ldots,\left(\widetilde{Y}_1^{(N)}\right)^T,\ldots,\left(\widetilde{Y}_{K^{(N)}}^{(N)}\right)^T\right]^T(w,\widetilde{\eta},\widetilde{\mu}),$$

with

$$\widetilde{Y}_i^{(n)}\overset{\text{def}}{=}\frac{h_n}{2}\,\omega_i^{(n)}\left\{\psi_u'{}^T\left[\tau_i^{(n)}\right]+f_u'{}^T\left[\tau_i^{(n)}\right]\widetilde{\lambda}_i^{(n)}+c_u'{}^T\left[\tau_i^{(n)}\right]\widetilde{\mu}_i^{(n)}\right\},\quad n\in[N],\ i\in[K^{(n)}],$$

we can write (9.52) as the system of equations

$$\varphi_u{}^T\,\widetilde{Y}(w,\widetilde{\eta},\widetilde{\mu})=0\quad\forall\varphi_u\in\mathbb{R}^{n_{\mathrm{q}}}.$$

Since we have $\frac{h_n}{2}\omega_i^{(n)} \neq 0$ the equation system holds if and only if

$$\boldsymbol{\psi}_u'^{T}\left[\tau_i^{(n)}\right] + \boldsymbol{f}_u'^{T}\left[\tau_i^{(n)}\right]\widetilde{\lambda}_i^{(n)} + \boldsymbol{c}_u'^{T}\left[\tau_i^{(n)}\right]\widetilde{\mu}_i^{(n)} = \mathbf{0}, \quad n \in [N],\ i \in [K^{(n)}].$$

Optimality Conditions [(9.11) + (9.13)] Replacing trial and test functions in (9.27) yields

$$\begin{aligned} 0 &\leqslant \sum_{n=1}^{N} \int_{(-1,1]} \boldsymbol{\varphi}_\mu^{\mathrm{h}}(\boldsymbol{t}_n(t))\, \mathrm{d}\mathbf{M}_{\mathrm{h}}(\boldsymbol{t}_n(t)), \\ &= \sum_{n=1}^{N} \sum_{i=1}^{K^{(n)}} \frac{h_n}{2}\, \omega_i^{(n)}\, {\varphi_\mu}_i^{(n)T}\, \widetilde{\mu}_i^{(n)} \quad \forall {\varphi_\mu}_i^{(n)} \geqslant \mathbf{0}. \end{aligned}$$

The vector definitions

$${\varphi_\mu}^{(n)} \stackrel{\text{def}}{=} \left[{\varphi_\mu}_1^{(n)T}, \ldots, {\varphi_\mu}_{K^{(n)}}^{(n)T}\right]^T, \qquad \varphi_\mu \stackrel{\text{def}}{=} \left[{\varphi_\mu}^{(1)T}, \ldots, {\varphi_\mu}^{(N)T}\right]^T$$

and

$$\widetilde{\mu}^{(n)} \stackrel{\text{def}}{=} \left[\frac{h_n}{2}\,\omega_1^{(n)}\left(\widetilde{\mu}_1^{(n)}\right)^T, \ldots, \frac{h_n}{2}\,\omega_{K^{(n)}}^{(n)}\left(\widetilde{\mu}_{K^{(n)}}^{(n)}\right)^T\right]^T, \qquad \widetilde{\mu} \stackrel{\text{def}}{=} \left[\left(\widetilde{\mu}^{(1)}\right)^T, \ldots, \left(\widetilde{\mu}^{(N)}\right)^T\right]^T$$

enable us to write the variational inequality as

$$0 \leqslant {\varphi_\mu}^T \widetilde{\mu} \qquad \forall \varphi_\mu \geqslant \mathbf{0}.$$

This formulation is equivalent to $\widetilde{\mu} \geqslant \mathbf{0}$ and since $\frac{h_n}{2}\,\omega_i^{(n)} \neq 0$ it is equivalent to $\widetilde{\mu}_i^{(n)} \geqslant 0$ for all $n \in [N]$ and $i \in [K^{(n)}]$. Now we continue with (9.28) and replace trial and test functions with their finite dimensional counterparts such that we obtain

$$\begin{aligned} 0 &= \sum_{n=1}^{N} \int_{(-1,+1]} \boldsymbol{\varphi}_M^{\mathrm{h}}(\boldsymbol{t}_n(t))\, \boldsymbol{c}[\boldsymbol{t}_n(t)]\, \mathrm{d}\mathbf{M}_{\mathrm{h}}(\boldsymbol{t}_n(t)) \\ &= \sum_{n=1}^{N} \sum_{i=1}^{K^{(n)}} \frac{h_n}{2}\, \omega_i^{(n)}\, {\varphi_M}_i^{(n)} \left(\widetilde{\mu}_i^{(n)}\right)^T \boldsymbol{c}[t_i^{(n)}] \end{aligned}$$

for all ${\varphi_M}_i^{(n)} \in \mathcal{Z}_H([-1,+1],\mathbb{R})$. We define the vectors

$${\varphi_M}^{(n)} \stackrel{\text{def}}{=} \left[{\varphi_M}_1^{(n)}, \ldots, {\varphi_M}_{K^{(n)}}^{(n)}\right]^T \qquad \text{and} \qquad \varphi_M \stackrel{\text{def}}{=} \left[{\varphi_M}^{(1)T}, \ldots, {\varphi_M}^{(N)T}\right]^T,$$

as well as the vector–valued function $\widetilde{\boldsymbol{C}} : \mathbb{R}^{n_{\mathrm{w}}} \times \mathbb{R}^{n_\eta} \times \mathbb{R}^{n_\mu} \longrightarrow \mathbb{R}^{n_{\mathrm{q}}}$

$$\widetilde{\boldsymbol{C}}(w,\widetilde{\eta},\widetilde{\mu}) = \left[\left(\widetilde{\boldsymbol{C}}_1^{(1)}\right)^T, \ldots, \left(\widetilde{\boldsymbol{C}}_{K^{(1)}}^{(1)}\right)^T, \ldots, \left(\widetilde{\boldsymbol{C}}_1^{(N)}\right)^T, \ldots, \left(\widetilde{\boldsymbol{C}}_{K^{(N)}}^{(N)}\right)^T\right]^T (w,\widetilde{\eta},\widetilde{\mu}),$$

with

$$\widetilde{C}_i^{(n)} \overset{\text{def}}{=} \frac{h_n}{2}\,\omega_i^{(n)} \left(\widetilde{\mu}_i^{(n)}\right)^T \boldsymbol{c}[t_i^{(n)}], \quad n \in [N],\ i \in [K^{(n)}].$$

Hence, we can write the variational equality equivalently as

$$0 = {\varphi_M}^T\, \widetilde{\boldsymbol{C}}(w, \widetilde{\eta}, \widetilde{\mu}) \qquad \forall \varphi_M,$$

which is equivalent to $\left(\widetilde{\mu}_i^{(n)}\right)^T \boldsymbol{c}[t_i^{(n)}] = 0$ for all $n \in [N]$ and $i \in [K^{(n)}]$ since $\frac{h_n}{2}\,\omega_i^{(n)} \neq 0$. Finally, we proceed with (9.29) in a similar way and obtain

$$0 \geqslant \sum_{n=1}^{N} \int_{(-1,1]} \boldsymbol{c}[\boldsymbol{t}_n(t)]\, \mathrm{d}\, \boldsymbol{\varphi}_c(\boldsymbol{t}_n(t)) = \sum_{n=1}^{N} \sum_{i=1}^{K^{(n)}} {\varphi_c}_i^{(n)^T} \boldsymbol{c}[t_i^{(n)}] \qquad \forall {\varphi_c}_i^{(n)} \geqslant \mathbf{0}.$$

By means of the vector

$${\varphi_c}^{(n)} \overset{\text{def}}{=} \left[{\varphi_c}_1^{(n)^T}, \ldots, {\varphi_c}_{K^{(n)}}^{(n)^T}\right]^T \qquad \text{and} \qquad \varphi_c \overset{\text{def}}{=} \left[{\varphi_c}^{(1)^T}, \ldots, {\varphi_c}^{(N)^T}\right]^T$$

we can write the variational inequality as

$${\varphi_c}^T\, \boldsymbol{C}(w) \qquad \forall \varphi_c \geqslant \mathbf{0}$$

which is equivalent to $\boldsymbol{C}(w) \leqslant \mathbf{0}$.

Optimality Conditions [(9.12)] We replace trial and test functions in (9.31) and evaluate the Lebesgue–Stieltjes integral resulting in

$$\begin{aligned}
0 &= \sum_{n=1}^{N} \int_{(-1,1]} \dot{\boldsymbol{x}}_{\mathrm{h}}(\boldsymbol{t}_n(t)) - \frac{h}{2}\boldsymbol{f}[\boldsymbol{t}_n(t)]\, \mathrm{d}\, \boldsymbol{\varphi}_\Lambda^{\mathrm{h}}(\boldsymbol{t}_n(t)) \\
&= \left\{ \sum_{i=1}^{K^{(1)}} \frac{h_1}{2}\omega_i^{(1)} {\varphi_\lambda}_i^{(1)^T} \left\{ \sum_{j=0}^{K^{(1)}} \frac{2}{h_1} x_j^{(1)} D_{i,j}^{(1)} - \frac{h}{2}\boldsymbol{f}\left[\tau_i^{(1)}\right] \right\} \right\} \\
&\quad + \sum_{n=2}^{N} \left\{ \sum_{i=1}^{K^{(n)}} \frac{h_n}{2}\omega_i^{(n)} {\varphi_\lambda}_i^{(n)^T} \left\{ \frac{2}{h_n} x_{K^{(n-1)}}^{(n-1)} D_{i,0}^{(n)} + \sum_{j=1}^{K^{(n)}} \frac{2}{h_n} x_j^{(n)} D_{i,j}^{(n)} - \frac{h}{2}\boldsymbol{f}\left[\tau_i^{(n)}\right] \right\} \right\} \\
&= \sum_{i=1}^{K^{(1)}} \sum_{j=0}^{K^{(1)}} \omega_i^{(1)} x_j^{(1)^T} D_{i,j}^{(1)} {\varphi_\lambda}_i^{(1)} - \sum_{i=1}^{K^{(1)}} \omega_i^{(1)} \frac{h_1}{2}\frac{h}{2}\boldsymbol{f}\left[\tau_i^{(1)}\right]^T {\varphi_\lambda}_i^{(1)} + \sum_{n=2}^{N} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} x_{K^{(n-1)}}^{(n-1)^T} D_{i,0}^{(n)} {\varphi_\lambda}_i^{(n)} \\
&\quad + \sum_{n=2}^{N} \sum_{i=1}^{K^{(n)}} \sum_{j=1}^{K^{(n)}} \omega_i^{(n)} x_j^{(n)^T} D_{i,j}^{(n)} {\varphi_\lambda}_i^{(n)} - \sum_{n=2}^{N} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} \frac{h_n}{2}\frac{h}{2}\boldsymbol{f}\left[\tau_i^{(n)}\right]^T {\varphi_\lambda}_i^{(n)}
\end{aligned}$$

$$= \sum_{i=1}^{K^{(1)}} \omega_i^{(1)} {\varphi_{\lambda_i}^{(1)}}^T \left\{ \sum_{j=0}^{K^{(1)}} x_j^{(1)} D_{i,j}^{(1)} - \frac{h_1}{2} \frac{h}{2} \boldsymbol{f}\left[\tau_i^{(1)}\right] \right\}$$
$$+ \sum_{n=2}^{N} \sum_{i=1}^{K^{(n)}} \omega_i^{(n)} {\varphi_{\lambda_i}^{(n)}}^T \left\{ \sum_{j=1}^{K^{(n)}} x_j^{(n)} D_{i,j}^{(n)} + x_{K^{(n-1)}}^{(n-1)} D_{i,0}^{(n)} - \frac{h_n}{2} \frac{h}{2} \boldsymbol{f}\left[\tau_i^{(n)}\right] \right\}. \tag{9.53}$$

We then introduce the vectors

$$\varphi_\lambda \stackrel{\text{def}}{=} \left[\left(\varphi_\lambda{}^{(1)}\right)^T, \ldots, \left(\varphi_\lambda{}^{(N)}\right)^T \right]^T, \qquad \varphi_\lambda{}^{(n)} \stackrel{\text{def}}{=} \left[\left(\varphi_{\lambda_1}^{(n)}\right)^T, \ldots, \left(\varphi_{\lambda_{K^{(n)}}}^{(n)}\right)^T \right]^T, \quad n \in [N],$$

and the vector valued function $\widetilde{\boldsymbol{F}} : \mathbb{R}^{n_\mathrm{w}} \longrightarrow \mathbb{R}^{n_\mathrm{F}}$ defined as

$$\widetilde{\boldsymbol{F}}(w) = \left[\left(\widetilde{\boldsymbol{F}}_1^{(1)}\right)^T, \ldots, \left(\widetilde{\boldsymbol{F}}_{K^{(1)}}^{(1)}\right)^T, \ldots, \left(\widetilde{\boldsymbol{F}}_1^{(N)}\right)^T, \ldots, \left(\widetilde{\boldsymbol{F}}_{K^{(N)}}^{(N)}\right)^T \right]^T (w),$$

where

$$\widetilde{\boldsymbol{F}}_i^{(1)}(w) \stackrel{\text{def}}{=} \omega_i^{(1)} \left\{ \sum_{j=0}^{K^{(1)}} x_j^{(1)} D_{i,j}^{(1)} - \frac{h_1}{2} \frac{h}{2} \boldsymbol{f}\left[\tau_i^{(1)}\right] \right\}, \quad i \in [K^{(1)}],$$

and

$$\widetilde{\boldsymbol{F}}_i^{(n)}(w) \stackrel{\text{def}}{=} \omega_i^{(n)} \left\{ \sum_{j=1}^{K^{(n)}} x_j^{(n)} D_{i,j}^{(n)} + x_{K^{(n-1)}}^{(n-1)} D_{i,0}^{(n)} - \frac{h_n}{2} \frac{h}{2} \boldsymbol{f}\left[\tau_i^{(n)}\right] \right\}, \quad 2 \leqslant n \leqslant N,\ i \in [K^{(n)}],$$

such that we can write (9.53) as equation system

$${\varphi_\lambda}^T \widetilde{\boldsymbol{F}}(w) = 0 \quad \forall \varphi_\lambda \in \mathbb{R}^{n_\mathrm{F}}.$$

Since it is $\omega_i^{(n)} \neq 0$ the system of equations holds if and only if

$$\sum_{j=0}^{K^{(1)}} x_j^{(1)} D_{i,j}^{(1)} - \frac{h_1}{2} \frac{h}{2} \boldsymbol{f}\left[\tau_i^{(1)}\right] = 0,$$
$$\sum_{j=1}^{K^{(n)}} x_j^{(n)} D_{i,j}^{(n)} + x_{K^{(n-1)}}^{(n-1)} D_{i,0}^{(n)} - \frac{h_n}{2} \frac{h}{2} \boldsymbol{f}\left[\tau_i^{(n)}\right] = 0, \qquad 2 \leqslant n \leqslant N,\ i \in [K^{(n)}].$$

Optimality Conditions [(9.14)] Replacing the trial function $\boldsymbol{x}(\cdot)$ in (9.32) with $\boldsymbol{x}_\mathrm{h}(\cdot)$ and a subsequent evaluation yields

$$0 = {\varphi_r}^T \boldsymbol{r}(t_\mathrm{s}, x_0^{(1)}, t_\mathrm{f}, x_{K^{(N)}}^{(N)}), \quad \forall \varphi_r \in \mathbb{R}^{n_\mathrm{r}},$$

which is equivalent to $\boldsymbol{r}(t_\mathrm{s}, x_0^{(1)}, t_\mathrm{f}, x_{K^{(N)}}^{(N)}) = 0$.

9.4 Synthesis

It can be easily checked by investigating the single equation components that the equation systems from Sections 9.2 and 9.3 in the variables $x_i^{(n)}$, $u_i^{(n)}$, $\nu = \nu^{(0)}$, $\widetilde{\lambda}_i^{(n)}$, and $\widetilde{\mu}_i^{(n)}$ are equivalent. This shows the statement of Theorem 9.3. The transformation from one system to the other can be determined as follows: we can identify the $x_i^{(n)}$ and $u_i^{(n)}$ variables. In order to map the $\lambda_i^{(n)}$ and the $\Lambda_i^{(n)}$ resp. the $\mu_i^{(n)}$ and the $M_i^{(n)}$ we use (9.40) and (9.48) and identify the algebraic mapping formulas

$$\Lambda_i^{(n)} = \frac{h_n}{2}\,\lambda_i^{(n)} \qquad \text{and} \qquad M_i^{(n)} = \frac{2}{h}\,\mu_i^{(n)}. \tag{9.54}$$

These algebraic relations enable us to determine costate estimates from a solution of the collocation system. The costate approximate solutions $\mathbf{\Lambda}_{\mathrm{h}}(\cdot)$ and $\mathbf{M}_{\mathrm{h}}(\cdot)$ are calculated by means of the multipliers $\lambda_i^{(n)}$ and $\mu_i^{(n)}$ coming from the NLP solver and (9.54).
In order to come up with approximations for the "derivatives" $\dot{\mathbf{\Lambda}}_{\mathrm{h}}(\cdot)$ and $\dot{\boldsymbol{M}}_{\mathrm{h}}(\cdot)$, evaluated at the collocation points $t_i^{(n)}$, we propose the following approach: since the step function (9.49) is an approximation of the **id**–function we expect the slopes of the steps to be approximately equal to one. This leads to the relation

$$\frac{\frac{h_n}{2}\,\omega_i^{(n)}}{h_i^{(n)}} = 1,$$

where $h_i^{(n)} \overset{\text{def}}{=} t_i^{(n)} - t_{i-1}^{(n)}$. We use this formula to find the approximations

$$\boldsymbol{\lambda}_{\mathrm{h}}(t_i^{(n)}) = \dot{\mathbf{\Lambda}}_{\mathrm{h}}(t_i^{(n)}) \approx \frac{\Lambda_i^{(n)}}{h_i^{(n)}} = \frac{\frac{h_n}{2}\,\lambda_i^{(n)}}{\frac{h_n}{2}\,\omega_i^{(n)}} = \frac{\lambda_i^{(n)}}{\omega_i^{(n)}} \quad \text{and}$$

$$\boldsymbol{\mu}_{\mathrm{h}}(t_i^{(n)}) = \dot{\boldsymbol{M}}_{\mathrm{h}}(t_i^{(n)}) \approx \frac{M_i^{(n)}}{h_i^{(n)}} = \frac{\frac{2}{h}\,\mu_i^{(n)}}{\frac{h_n}{2}\,\omega_i^{(n)}} = \frac{2}{h}\,\frac{2}{h_n}\,\frac{\mu_i^{(n)}}{\omega_i^{(n)}}.$$

We use these approximations for our numerical experiments in Chapter 13 and Chapter 14.

Chapter 10

A Goal–Oriented Global Error Estimation for Collocation Methods

The introduction of the local pseudospectral method from Section 7.4 along with the novel interpretation for its discrete adjoints in the previous Chapter 9 enables us now to derive novel goal–oriented global error estimators for the aforesaid method. The error estimation approach derived is based on concepts that are used for a posteriori error estimation in Galerkin–type FE methods.

In this thesis we derive for the first time a posteriori global error estimators that use information computed from adjoint NLP variables which arise from a local pseudospectral discretization approach. For a criterion of interest J we estimate the difference

$$J(\boldsymbol{x},\boldsymbol{u}) - J(\boldsymbol{x}_{\mathrm{h}},\boldsymbol{u}_{\mathrm{h}}), \tag{10.1}$$

where $(\boldsymbol{x},\boldsymbol{u})$ is the unknown exact OCP solution and $(\boldsymbol{x}_{\mathrm{h}},\boldsymbol{u}_{\mathrm{h}})$ the computed approximation. This difference in J is called the *goal–oriented global error* of $(\boldsymbol{x}_{\mathrm{h}},\boldsymbol{u}_{\mathrm{h}})$. We point out that we consider discretizations of rather generally formulated OCPs, i.e., in particular OCPs with mixed control–state constraints. The functional J is supposed to be sufficiently smooth throughout this chapter.

In general, we distinguish between so–called *error representations* and the aforementioned *error estimators*, which become relevant when we have to put error representations into practice. Error representations still hold unknown exact quantities while error estimators only rely on quantities that are available in practical implementations. Hence, error representations are especially of relevance in terms of theoretical investigations such as the determination of the asymptotic behavior of discretizations for decreasing step sizes. However, we need error estimators in order to be able to choose step sizes as large as possible while pushing the error under a certain threshold.

Well–established counterparts in Galerkin–type FE methods for PDEs as well their transfer to error estimations for multistep BDF methods in the thesis of Beigel [41] inspired our novel goal–oriented error estimators. Section 10.1 is devoted to a literature review about commonly used error estimators in the ODE and OCP context.

In Section 10.2 we derive an error representation for (10.1). It still includes the unknown adjoint solutions for both systems dynamics, $\boldsymbol{\Lambda}(\cdot)$, and mixed control–state path constraints, $\boldsymbol{M}(\cdot)$.

Section 10.3 describes ways how the goal–oriented error representation can be approximated.

Finally, in Section 10.4 we derive goal–oriented error estimators.

10.1 Literature Review of Global Error Estimation

Prior to analyzing error control in an OCP oriented context, we give a short literature review of advanced error control techniques in the fields of ordinary and partial differential equations. Carrying over some fundamental ideas to OCP error control enables us to derive a novel goal–oriented global error estimator and associated mesh refinement strategies. As Section 10.1.2 reveals, our approach differs fundamentally from current approaches.

10.1.1 Error Estimation for Differential Equations

Before we deal with the PDE case we investigate the case of ODE error control.

ODE Error Estimation Error estimation of OCPs subject to ODEs is related to error estimation of IVPs for ODEs in the sense that OCP solution methods such as direct multiple shooting (see Section 6.2.3) rely on good IVP solution approximations. IVP error estimation represents a research field in applied mathematics that has been investigated from the 1960s on.
Zadunaisky [462] proposes an error estimation approach in which one first determines a continuous approximation of the solution obtained by a numerical integration scheme. This continuous approximation is then used to construct a "neighboring" IVP with a known exact solution. After this the new IVP is solved with the same discretization scheme that was used for the original IVP and one obtains a global error by comparing the solution with the exact one. Another approach similar in fashion as it relies on a related IVP was proposed by Henrici [229] and involves the unknown local truncation error (see Zadunaisky [463]). Stetter [417] uses Zadunaisky's global error estimator in order to iteratively improve the nominal approximation of IVPs. Skeel [411] provides an excellent survey about early IVP error estimation approaches. Those approaches were later applied to BDF methods as well, cf. Skeel [412]. They all have in common that they suffer from severe drawbacks. In particular, they are computationally expensive and require small and constant step sizes.
Later on, error control based on local techniques such as order and step size adaption according to local error quantities became center of interest. For a comprehensive overview, we recommend reading the textbooks of Hairer et al. [219] and Hairer and Wanner [217], or the article of Shampine [406]. Even though most of these techniques work satisfactorily, better performance can be expected by equipping them with adjoint information: in fact, one is interested in determining the impact factors of the local error contributions on the target quantity. Similarly to optimal control theory one seeks a sensitivity analysis with respect to local disturbances of the model which naturally leads to the concept of an 'adjoint' (or 'dual') problem.
Incorporating adjoint information into a posteriori global error estimation has been intensively analyzed since the 2000s. Here, the global error with respect to a certain criterion of interest is determined by solving the adjoint variational IVP and using it as weights for local error quantities. Detailed information can be found in the articles of Moon et al. [330], Cao and Petzold [101], and Lang and Verwer [284]. These approaches have in common that they require an additional adaptive numerical integration in order to estimate the adjoint IVP solution along a nominal solution approximation.

PDE Error Estimation Modern PDE error control techniques rely on *residual–based* a posteriori error estimation. In case of OCPs we have already encountered the term 'residual' in the context of the weighted residual method, cf. Section 6.3.2. The ODE *defect* or *residual*, evaluated at an approximate solution $\boldsymbol{x}_{\mathrm{h}}(\cdot)$, is calculated as

$$\varrho(\boldsymbol{x}_{\mathrm{h}})(\cdot) = \dot{\boldsymbol{x}}_{\mathrm{h}}(\cdot) - \boldsymbol{f}(\cdot, \boldsymbol{x}_{\mathrm{h}}(\cdot)).$$

A similar concept can be carried over to the PDEs case: to this end let us consider a continuous and a related discrete model. The continuous and discrete model equations are given as

$$A(\boldsymbol{y}) = \boldsymbol{f} \quad \text{and} \quad A_h(\boldsymbol{y}_h) = \boldsymbol{f}_h.$$

Here, the continuous model is characterized by the functional A, representing a linear differential operator, and the function $\boldsymbol{f}(\cdot)$ that acts as a force term. The discretization parameter $h \in \mathbb{R}_{>}$ of the discrete model components A_h and $\boldsymbol{f}_h(\cdot)$ indicates the approximation quality of the associated mapping. The residual term is then defined as

$$\varrho(\boldsymbol{y}_h) = \boldsymbol{f} - A(\boldsymbol{y}_h).$$

In the late 1970s residual–based error control was established in Galerkin FE methods for PDEs. Traditional approaches such as the pioneering work of Babuška and Rheinboldt [23, 24] result in estimates of the form

$$\|\boldsymbol{y} - \boldsymbol{y}_h\|_E \leqslant C \, \|\varrho(\boldsymbol{y}_h)\|_E^* ,$$

where $\|\cdot\|_E$ usually denotes a natural 'energy norm' in a PDE context and $\|\cdot\|_E^*$ a suitable dual norm. Energy error estimation directly involves the variational formulation of the problem and allows for exploiting its natural coercivity properties which make it rather generic.

However, it does not provide error estimation with respect to physically motivated quantities of interest. For this reason the approach was later extended in the sense that it can incorporate certain duality information: for some quantity of interest, which is expressed by applying a functional $J(\cdot)$ to the solution $\boldsymbol{y}(\cdot)$, one wants to express the error $J(\boldsymbol{y}) - J(\boldsymbol{y}_h)$ in terms of local residuals $\varrho_{\mathcal{I}_n}(\boldsymbol{y}_h)$. The approach was first applied to elliptic model problems, cf. Babuška and Miller [20, 21, 22]. In a similar vein, error control is addressed by Eriksson et al. [148, 149] for more general situations. The same authors transferred residual–based error control into an ODE context, cf. Johnson [258], Estep [150], and Eriksson et al. [148]. In these methods the stability of the nominal problem is expressed by a single (global) stability constant which is derived from the associated adjoint problem. Those stability constants are usually derived by analytical arguments. For more details we refer the reader to the textbooks of Ainsworth and Oden [7], Babuška and Strouboulis [25] and Verfürth [434].

In general, it is quite cumbersome or even impossible to determine the aforementioned stability constants. Rather, one would like to have a computation–based feedback method. The *DWR method*, developed by Becker and Rannacher [37, 38], represents such a method, and is briefly described in the following: if we consider $J(\boldsymbol{y})$ as quantity of physical interest, then it is our goal to control the discretization error with respect to this functional output, i.e., we want

to express the error $E(\boldsymbol{y}_h) \stackrel{\text{def}}{=} J(\boldsymbol{y}) - J(\boldsymbol{y}_h)$ in terms of the computable residuals $\varrho_K(\boldsymbol{y}_h)$. As an example we consider the case of minimizing the local total error $e_K = (\boldsymbol{y} - \boldsymbol{y}_h) \restriction_K$ on a finite element K. In case of a linear problem $A\boldsymbol{y} = \boldsymbol{f}$, the error e_K splits into two components by superposition, namely the local truncation error e_K^{loc} and the globally transported pollution error e_K^{pol} such that it holds that $e_K = e_K^{loc} + e_K^{pol}$. The impact of the residual $\varrho_K(\boldsymbol{y}_h) = A(\boldsymbol{y} - \boldsymbol{y}_h)$ on the local error $e_{K'}$ of another element K' is basically controlled by a global Green function of the continuous problem. In general, it is impossible to determine the complex error interaction analytically. Thus, the DWR methodology is about how those dependences can be captured by numerical computations. By employing an auxiliary adjoint problem $A^* \boldsymbol{z} = \boldsymbol{j}$, which is driven by the target functional $J(\cdot)$ in terms of a density function $\boldsymbol{j}(\cdot)$, one ends up with a posteriori error information of the form

$$|J(\boldsymbol{y}) - J(\boldsymbol{y}_h)| = \sum\nolimits_K \varrho_K(\boldsymbol{y}_h)\, \omega_h(\boldsymbol{z}).$$

The adjoint solution $\boldsymbol{z}(\cdot)$ can be seen as a generalized global Green function with respect to the functional $J(\cdot)$. The weights $\omega_h(\boldsymbol{z})$ quantify the impact of local variations (sensitivity analysis) of the residuals $\varrho_K(\boldsymbol{y}_h)$ on the error quantity $E(\boldsymbol{y}_h)$.

10.1.2 Error Estimation for Optimal Control Problems

Once again we start by investigating the ODE case followed by the PDE case.

ODE Error Estimation There exist several approaches of ODE error estimation and mesh refinement. In an collocation method context, one generally distinguishes h, p and hp direct collocation methods. A p collocation approach is usually applied to global collocation methods (see Section 7.2), i.e., one increases the polynomial degree of the approximating polynomials to achieve convergence. Contrary, one often applies a h collocation approach to local collocation methods (see Section 7.4), i.e., one increases the number of finite elements to achieve convergence. However, in order to be able to exploit spectral convergence properties of a p collocation approach it is highly recommended to combine p and h approach for local collocation methods which is then called a hp approach. The general strategy should be to make the grid finer in regions of discontinuities and to increase the polynomial degree in smooth regions.
The methodology of our first mesh refinement approach, which is explained in detail e.g. in the textbook of Betts [62], can be employed for Shooting Methods (see Section 6.2.2+6.2.3) as well as local approach methods (see Section 6.3.1). Note that it is applied to Shooting Methods in every NLP solver iteration. In contrast, local approach methods use it in every iteration of the SNLP routine. The idea behind can be summarized as follows: let $\widetilde{\boldsymbol{x}}(\cdot)$ and $\widetilde{\boldsymbol{u}}(\cdot)$ denote an approximation of state and control trajectories obtained in an NLP/SNLP iteration. Then it is assumed that $\widetilde{\boldsymbol{u}}(\cdot)$ is correct and optimal such that it remains to estimate the error between $\widetilde{\boldsymbol{x}}(\cdot)$ and the correct solution $\boldsymbol{x}^*(\cdot)$. This reduces the error estimation approach to an ODE error estimation which was described in the previous section. The subsequent mesh refinement process is then driven by equilibrating and reducing the calculated element–wise errors.
A typical p pseudospectral method is described by Gong et al. [201]: the general idea is to

increase the number of collocation points near the points "at which control undergoes a sudden change". This is realized by means of the pseudospectral differentiation matrix and by detecting maximum derivatives of the control approximations.
Munos and Moore [332] and Grüne [211] describe refinement strategies for indirect discretization methods based on the Hamilton–Jacobi–Bellman equation. The refinement process is based on detecting some specified local irregularities. In case an irregularity is detected the grid is refined by splitting selected intervals.
Refinement strategies based on density functions can be found in Zhao and Tsiotras [471]: for a grid containing N_i grid points in the i–th iteration one calculates a mesh size increment $\Delta N_i \geqslant 0$ according to the approach of Betts and Huffman [65] such that the $i+1$–th grid has $N_{i+1} = N_i + \Delta N_i$ grid points. These grid points are distributed according to a density function which tries to regulate the integration error. For instance, one could consider a piecewise constant density function whose value on each interval is equal to the corresponding principle local truncation error (see Schwartz [397]).
Darby et al. [122] carry out an hp pseudospectral method where errors are measured according to residual violations at points between collocation points. If these errors have roughly the same order of magnitude they are called 'uniform–type' errors and the polynomial degree is increased. Otherwise they call the errors to be of 'nonuniform–type'. In this case the respective finite element is split.
A similar refinement strategy is described by Darby et al. [124] where they use the same criterion to measure the error. However, they decide according to guesses of trajectory curvatures if the polynomial degree is increased or if a finite element is split. This strategy is in accordance with a rapid convergence behavior of p methods for smooth solutions.
Another refinement technique can be found in the article of Patterson et al. [352]. Here, the error estimation rests on estimating the error in the solution of the differential equation. This is done by constructing an additional higher–order state approximation. Based on a–priori error estimations of the used integration method the necessary polynomial degree N can be determined for every finite element. If N exceeds a certain threshold $N_{\max}$ the associated finite element is divided into subintervals.
A combination of the aforementioned refinement strategies, which also allows for merging existing finite elements, is described by Liu et al. [299].

PDE Error Estimation State of the art a–posteriori error estimation for PDE optimal control is built upon the previously presented DWR methodology (see Section 10.1.1). To show its principles in an optimal control context let us consider a PDE constrained OCP of the form

$$J(\boldsymbol{y},\boldsymbol{q}) \to \min!, \qquad A(\boldsymbol{y},\boldsymbol{q}) = 0, \tag{10.2}$$

with state space V, control space Q, cost functional $J(\cdot,\cdot) : V \times Q \longrightarrow \mathbb{R}$, continuous model $A(\cdot,\cdot)$ on $V \times Q$, states $\boldsymbol{y}$, and controls $\boldsymbol{q}$. Since the DWR methodology provides error estimation techniques for finite element Galerkin approximations of general variational problems, we consider the optimization problem

$$J(\boldsymbol{y},\boldsymbol{q}) \to \min!, \qquad a(\boldsymbol{y},\boldsymbol{q})(\boldsymbol{\psi}) = 0 \quad \forall \boldsymbol{\psi} \in V, \tag{10.3}$$

with the semi–linear form $a(\cdot,\cdot)(\cdot)$ on $V \times Q \times V$, coming from a variational formulation of the model equation in (10.2). Assuming the existence of a unique local minimum $(\boldsymbol{y},\boldsymbol{q}) \in V \times Q$, it can be characterized by means of the Lagrangian framework as a saddle point $(\boldsymbol{y},\boldsymbol{q},\boldsymbol{\lambda}) \in V \times Q \times V$ of the Lagrangian functional

$$\mathcal{L}(\boldsymbol{y},\boldsymbol{q},\boldsymbol{\lambda}) \stackrel{\text{def}}{=} J(\boldsymbol{y},\boldsymbol{q}) + a(\boldsymbol{y},\boldsymbol{q})(\boldsymbol{\lambda})$$

with the costate variable $\boldsymbol{\lambda}$. The saddle point is given as solution of the saddle point problem

$$\begin{aligned} a'_y(\boldsymbol{y},\boldsymbol{q})(\boldsymbol{\varphi},\boldsymbol{\lambda}) &= J'_y(\boldsymbol{y},\boldsymbol{q})(\boldsymbol{\varphi}) \quad \forall \boldsymbol{\varphi} \in V, \\ a'_q(\boldsymbol{y},\boldsymbol{q})(\boldsymbol{\chi},\boldsymbol{\lambda}) &= J'_q(\boldsymbol{y},\boldsymbol{q})(\boldsymbol{\chi}) \quad \forall \boldsymbol{\chi} \in Q, \\ a(\boldsymbol{y},\boldsymbol{q})(\boldsymbol{\psi}) &= \Theta \qquad\qquad\quad \forall \boldsymbol{\psi} \in V. \end{aligned}$$

A discretization of the variational problem (10.3) by a standard Galerkin approach employing the finite dimensional spaces $V_\mathrm{h} \times Q_\mathrm{h} \subset V \times Q$ yields the problem

$$J(\boldsymbol{y}_\mathrm{h},\boldsymbol{q}_\mathrm{h}) \to \min!, \qquad a(\boldsymbol{y}_\mathrm{h},\boldsymbol{q}_\mathrm{h})(\boldsymbol{\psi}_\mathrm{h}) = 0 \quad \forall \boldsymbol{\psi}_\mathrm{h} \in V_\mathrm{h}.$$

Under certain assumptions we can find solutions $(\boldsymbol{y}_\mathrm{h},\boldsymbol{q}_\mathrm{h}) \in V_\mathrm{h} \times Q_\mathrm{h}$ to this finite dimensional optimization problem by means of determining a saddle point $(\boldsymbol{y}_\mathrm{h},\boldsymbol{q}_\mathrm{h},\boldsymbol{\lambda}_\mathrm{h}) \in V_\mathrm{h} \times Q_\mathrm{h} \times V_\mathrm{h}$ of the Lagrangian $\mathcal{L}(\cdot,\cdot,\cdot)$. That is, we solve the discrete saddle–point problem

$$\begin{aligned} a'_y(\boldsymbol{y}_\mathrm{h},\boldsymbol{q}_\mathrm{h})(\boldsymbol{\varphi}_\mathrm{h},\boldsymbol{\lambda}_\mathrm{h}) &= J'_y(\boldsymbol{y}_\mathrm{h},\boldsymbol{q}_\mathrm{h})(\boldsymbol{\varphi}_\mathrm{h}) \quad \forall \boldsymbol{\varphi}_\mathrm{h} \in V_\mathrm{h}, \\ a'_q(\boldsymbol{y}_\mathrm{h},\boldsymbol{q}_\mathrm{h})(\boldsymbol{\chi}_\mathrm{h},\boldsymbol{\lambda}_\mathrm{h}) &= J'_q(\boldsymbol{y}_\mathrm{h},\boldsymbol{q}_\mathrm{h})(\boldsymbol{\chi}_\mathrm{h}) \quad \forall \boldsymbol{\chi}_\mathrm{h} \in Q_\mathrm{h}, \\ a(\boldsymbol{y}_\mathrm{h},\boldsymbol{q}_\mathrm{h})(\boldsymbol{\psi}_\mathrm{h}) &= \Theta \qquad\qquad\quad \forall \boldsymbol{\psi}_\mathrm{h} \in V_\mathrm{h}. \end{aligned}$$

It is the goal to control the discretization error that corresponds to the cost functional $J(\cdot,\cdot)$. To do so, one can employ the DWR methodology. By means of the 'primal', 'dual', and 'control' residuals given by

$$\begin{aligned} \varrho^y(\cdot) &\stackrel{\text{def}}{=} J'_y(\boldsymbol{y}_\mathrm{h},\boldsymbol{q}_\mathrm{h})(\cdot) - a'_y(\boldsymbol{y}_\mathrm{h},\boldsymbol{q}_\mathrm{h})(\cdot,\boldsymbol{\lambda}_\mathrm{h}) && \text{(dual residual)} \\ \varrho^q(\cdot) &\stackrel{\text{def}}{=} J'_q(\boldsymbol{y}_\mathrm{h},\boldsymbol{q}_\mathrm{h})(\cdot) - a'_q(\boldsymbol{y}_\mathrm{h},\boldsymbol{q}_\mathrm{h})(\cdot,\boldsymbol{\lambda}_\mathrm{h}) && \text{(control residual)} \\ \varrho(\cdot) &\stackrel{\text{def}}{=} -a(\boldsymbol{y}_\mathrm{h},\boldsymbol{q}_\mathrm{h})(\cdot) && \text{(primal residual)} \end{aligned}$$

one can show (see Becker et al. [40, Theorem 1]) the a posteriori error representation

$$J(\boldsymbol{y},\boldsymbol{q}) - J(\boldsymbol{y}_\mathrm{h},\boldsymbol{q}_\mathrm{h}) = \tfrac{1}{2}\left\{\varrho^y(\boldsymbol{y} - i_\mathrm{h}\boldsymbol{y}) + \varrho^q(\boldsymbol{q} - i_\mathrm{h}\boldsymbol{q}) + \varrho(\boldsymbol{\lambda} - i_\mathrm{h}\boldsymbol{\lambda})\right\} + \mathcal{R}_h, \tag{10.4}$$

where $i_\mathrm{h}\boldsymbol{y}$, $i_\mathrm{h}\boldsymbol{\lambda} \in V_\mathrm{h}$, and $i_\mathrm{h}\boldsymbol{q} \in Q_\mathrm{h}$ are arbitrary approximations with suitable interpolation operators i_h. One finds the remainder term $\mathcal{R}_h$ to be cubic in the error $\boldsymbol{e} = (\boldsymbol{e}^y, \boldsymbol{e}^q, \boldsymbol{e}^\lambda)$ with $\boldsymbol{e}^y \stackrel{\text{def}}{=} \boldsymbol{y} - \boldsymbol{y}_\mathrm{h}$, $\boldsymbol{e}^q \stackrel{\text{def}}{=} \boldsymbol{q} - \boldsymbol{q}_\mathrm{h}$, and $\boldsymbol{e}^\lambda \stackrel{\text{def}}{=} \boldsymbol{\lambda} - \boldsymbol{\lambda}_\mathrm{h}$. The terms $\varrho^y(\boldsymbol{y} - i_\mathrm{h}\boldsymbol{y})$, $\varrho^q(\boldsymbol{q} - i_\mathrm{h}\boldsymbol{q})$, and $\varrho(\boldsymbol{\lambda} - i_\mathrm{h}\boldsymbol{\lambda})$ in (10.4) represent the discretization error. They need to be approximated by numerically evaluating the interpolation errors $\boldsymbol{y} - i_\mathrm{h}\boldsymbol{y}$, $\boldsymbol{q} - i_\mathrm{h}\boldsymbol{q}$, and $\boldsymbol{\lambda} - i_\mathrm{h}\boldsymbol{\lambda}$. This can be done,

for instance, by means of a local higher–order interpolation. By doing so and by ignoring the cubic remainder term $\mathcal{R}_h$, one can derive local cell–wise error indicators $\eta_{\mathrm{h},K}$ that are associated with the mesh cells K and depend exclusively on the available values $\boldsymbol{y}_\mathrm{h}$, $\boldsymbol{q}_\mathrm{h}$, and $\boldsymbol{\lambda}_\mathrm{h}$, i.e., one has an error estimate of the form

$$|J(\boldsymbol{y},\boldsymbol{q}) - J(\boldsymbol{y}_\mathrm{h},\boldsymbol{q}_\mathrm{h})| \leqslant \sum\nolimits_K \eta_{\mathrm{h},K}(\boldsymbol{y}_\mathrm{h},\boldsymbol{q}_\mathrm{h},\boldsymbol{\lambda}_\mathrm{h}).$$

Based on this estimate one can apply several mesh adaption strategies, cf. Becker and Rannacher [39].

10.2 Goal–Oriented Error Representation

In this section, we derive an error representation for the performance criterion

$$J(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)) \stackrel{\text{def}}{=} \boldsymbol{\varphi}(\boldsymbol{x}(t_\mathrm{f})) + \int_{t_\mathrm{s}}^{t_\mathrm{f}} \boldsymbol{\psi}(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\mathrm{d}t$$

for the Petrov–Galerkin FE discretization developed in Chapter 9. By means of a Taylor expansion of the performance criterion $J(\cdot)$ at the current solution approximation $(\boldsymbol{x}_\mathrm{h}(\cdot),\boldsymbol{u}_\mathrm{h}(\cdot))$, we express the error $J(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)) - J(\boldsymbol{x}_\mathrm{h}(\cdot),\boldsymbol{u}_\mathrm{h}(\cdot))$ as a sum of local error contributions. Our contribution extends Beigel's theory (see Beigel [41]) from CVP type problems to OCPs with mixed control–state constraints.

Theorem 10.1 (Error Representation)
Let $(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),\boldsymbol{\Lambda}(\cdot),\mathbf{M}(\cdot),\lambda_{t_\mathrm{s}}) \in \mathcal{Y}^1(\mathcal{T},\mathbb{R}^{n_\mathrm{x}}) \times \mathcal{Y}^0(\mathcal{T},\mathbb{R}^{n_\mathrm{u}}) \times \mathcal{NBV}(\mathcal{T},\mathbb{R}^{n_\mathrm{x}}) \times \mathcal{NBV}(\mathcal{T},\mathbb{R}^{n_\mathrm{c}}) \times \mathbb{R}^{n_\mathrm{x}}$ be the solution of the weak formulation of the local minimum principle equations (9.5)–(9.14) and $(\boldsymbol{x}_\mathrm{h}(\cdot),\boldsymbol{u}_\mathrm{h}(\cdot)) \in \mathcal{Y}^1_{\mathcal{P}}(\mathcal{T},\mathbb{R}^{n_\mathrm{x}}) \times \mathcal{Y}^0_{\mathcal{P}}(\mathcal{T},\mathbb{R}^{n_\mathrm{u}})$ the solution of the nominal Petrov–Galerkin FE discretization from Section 9.3.2. Then, the global error in the criterion of interest has the following form:

$$\begin{aligned} J(\boldsymbol{x}_\mathrm{h}(\cdot),\boldsymbol{u}_\mathrm{h}(\cdot)) - J(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)) = &\sum_{n=1}^{N} \int_{\mathcal{I}_n} \dot{\boldsymbol{x}}_\mathrm{h}(t) - \boldsymbol{f}(\boldsymbol{x}_\mathrm{h}(t),\boldsymbol{u}_\mathrm{h}(t))\,\mathrm{d}\,[\boldsymbol{\Lambda} - \boldsymbol{i}_\mathrm{h}\boldsymbol{\Lambda}](t) \\ &+ \sum_{n=1}^{N} \int_{\mathcal{I}_n} \boldsymbol{c}(\boldsymbol{x}_\mathrm{h}(t),\boldsymbol{u}_\mathrm{h}(t))\,\mathrm{d}\,[\mathbf{M} - \boldsymbol{i}_\mathrm{h}\mathbf{M}](t) \\ &+ \left[\lambda_{t_\mathrm{s}} - \boldsymbol{i}_\mathrm{h}\lambda_{t_\mathrm{s}}\right]^T \{\boldsymbol{x}_\mathrm{h}(t_\mathrm{s}) - x_\mathrm{s}\} + \mathcal{R}_h. \end{aligned} \tag{10.5}$$

Here, $\boldsymbol{i}_\mathrm{h} : \mathcal{NBV}(\mathcal{T},\mathbb{R}^{n_\mathrm{x}}) \times \mathcal{NBV}(\mathcal{T},\mathbb{R}^{n_\mathrm{c}}) \times \mathbb{R}^{n_\mathrm{x}} \longrightarrow \mathcal{Z}_H(\mathcal{T},\mathbb{R}^{n_\mathrm{x}}) \times \mathcal{Z}_H(\mathcal{T},\mathbb{R}^{n_\mathrm{c}}) \times \mathbb{R}^{n_\mathrm{x}}$ denotes an interpolation operator and the remainder $\mathcal{R}_h$ is quadratic in the global error function $\boldsymbol{e}(\cdot) = [\boldsymbol{e}_x(\cdot)^T, \boldsymbol{e}_u(\cdot)^T]^T$ with $\boldsymbol{e}_x(\cdot) = \boldsymbol{x}(\cdot) - \boldsymbol{x}_\mathrm{h}(\cdot)$ and $\boldsymbol{e}_u(\cdot) = \boldsymbol{u}(\cdot) - \boldsymbol{u}_\mathrm{h}(\cdot)$, and consists of three summands $\mathcal{R}_h = \mathcal{R}_J^h + \mathcal{R}_f^h + \mathcal{R}_c^h$ with

$$\begin{aligned} \mathcal{R}_J^h \stackrel{\text{def}}{=} &- \int_0^1 \boldsymbol{e}_x(\cdot)^T J''_{xx}(\boldsymbol{x}_\mathrm{h}(\cdot) + s\,\boldsymbol{e}_x(\cdot), \boldsymbol{u}_\mathrm{h}(\cdot) + s\,\boldsymbol{e}_u(\cdot))\,\boldsymbol{e}_x(\cdot)\,s\,\mathrm{d}s \\ &- \int_0^1 \boldsymbol{e}_u(\cdot)^T J''_{uu}(\boldsymbol{x}_\mathrm{h}(\cdot) + s\,\boldsymbol{e}_x(t), \boldsymbol{u}_\mathrm{h}(\cdot) + s\,\boldsymbol{e}_u(\cdot))\,\boldsymbol{e}_u(\cdot)\,s\,\mathrm{d}s \end{aligned}$$

$$-2\int_0^1 \boldsymbol{e}_x(\cdot)^T J''_{xu}(\boldsymbol{x}_\mathrm{h}(\cdot)+s\,\boldsymbol{e}_x(\cdot),\boldsymbol{u}_\mathrm{h}(\cdot)+s\,\boldsymbol{e}_u(\cdot))\,\boldsymbol{e}_u(\cdot)s\;\mathrm{d}s,$$

$$\begin{aligned}\mathcal{R}_f^h \overset{\text{def}}{=} &-\int_0^1 \sum_{n=1}^N \int_{\mathcal{I}_n} \boldsymbol{e}_x(t)^T \boldsymbol{f}''_{xx}(\boldsymbol{x}_\mathrm{h}(t)+s\,\boldsymbol{e}_x(t),\boldsymbol{u}_\mathrm{h}(t)+s\,\boldsymbol{e}_u(t))\,\boldsymbol{e}_x(t)\;\mathrm{d}\boldsymbol{\Lambda}(t)s\;\mathrm{d}s\\ &-\int_0^1 \sum_{n=1}^N \int_{\mathcal{I}_n} \boldsymbol{e}_u(t)^T \boldsymbol{f}''_{uu}(\boldsymbol{x}_\mathrm{h}(t)+s\,\boldsymbol{e}_x(t),\boldsymbol{u}_\mathrm{h}(t)+s\,\boldsymbol{e}_u(t))\,\boldsymbol{e}_u(t)\;\mathrm{d}\boldsymbol{\Lambda}(t)s\;\mathrm{d}s\\ &-2\int_0^1 \sum_{n=1}^N \int_{\mathcal{I}_n} \boldsymbol{e}_x(t)^T \boldsymbol{f}''_{xu}(\boldsymbol{x}_\mathrm{h}(t)+s\,\boldsymbol{e}_x(t),\boldsymbol{u}_\mathrm{h}(t)+s\,\boldsymbol{e}_u(t))\,\boldsymbol{e}_u(t)\;\mathrm{d}\boldsymbol{\Lambda}(t)s\;\mathrm{d}s,\end{aligned} \tag{10.6}$$

and

$$\begin{aligned}\mathcal{R}_c^h \overset{\text{def}}{=} &\int_0^1 \sum_{n=1}^N \int_{\mathcal{I}_n} \boldsymbol{e}_x(t)^T \boldsymbol{c}''_{xx}(\boldsymbol{x}_\mathrm{h}(t)+s\,\boldsymbol{e}_x(t),\boldsymbol{u}_\mathrm{h}(t)+s\,\boldsymbol{e}_u(t))\,\boldsymbol{e}_x(t)\;\mathrm{d}\mathbf{M}(t)s\;\mathrm{d}s\\ &+\int_0^1 \sum_{n=1}^N \int_{\mathcal{I}_n} \boldsymbol{e}_u(t)^T \boldsymbol{c}''_{uu}(\boldsymbol{x}_\mathrm{h}(t)+s\,\boldsymbol{e}_x(t),\boldsymbol{u}_\mathrm{h}(t)+s\,\boldsymbol{e}_u(t))\,\boldsymbol{e}_u(t)\;\mathrm{d}\mathbf{M}(t)s\;\mathrm{d}s\\ &+2\int_0^1 \sum_{n=1}^N \int_{\mathcal{I}_n} \boldsymbol{e}_x(t)^T \boldsymbol{c}''_{xu}(\boldsymbol{x}_\mathrm{h}(t)+s\,\boldsymbol{e}_x(t),\boldsymbol{u}_\mathrm{h}(t)+s\,\boldsymbol{e}_u(t))\,\boldsymbol{e}_u(t)\;\mathrm{d}\mathbf{M}(t)s\;\mathrm{d}s.\end{aligned} \tag{10.7}$$

△

Proof The proof of the theorem employs some results on the generalized Taylor's Theorem which can be found with our common notation for exact and approximate solutions in Appendix A.3 and Appendix A.4. Differentiating the performance criterion with respect to $\boldsymbol{x}$ and $\boldsymbol{u}$ yields

$$J'_x(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot))\,\boldsymbol{e}_x(\cdot) = \boldsymbol{\varphi}'_x(\boldsymbol{x}(t_f))\,\boldsymbol{e}_x(\cdot) + \sum_{n=1}^N \int_{\mathcal{I}_n} \boldsymbol{\psi}'_x(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{e}_x(t)\;\mathrm{d}\,\mathbf{id}(t), \tag{10.8}$$

$$J'_u(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot))\,\boldsymbol{e}_u(\cdot) = \sum_{n=1}^N \int_{\mathcal{I}_n} \boldsymbol{\psi}'_u(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{e}_u(t)\;\mathrm{d}\,\mathbf{id}(t). \tag{10.9}$$

With the result from Appendix A.4 we find

$$J'_x(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot))\,\boldsymbol{e}_x(\cdot) + J'_u(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot))\,\boldsymbol{e}_u(\cdot) - \{J(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)) - J(\boldsymbol{x}_\mathrm{h}(\cdot),\boldsymbol{u}_\mathrm{h}(\cdot))\} = -\mathcal{R}_J^h(\cdot), \tag{10.10}$$

where

$$\begin{aligned}\mathcal{R}_J^h(\cdot) = &-\int_0^1 \boldsymbol{e}_x(\cdot)^T J''_{xx}(\boldsymbol{x}_\mathrm{h}(\cdot)+s\,\boldsymbol{e}_x(\cdot),\boldsymbol{u}_\mathrm{h}(\cdot)+s\,\boldsymbol{e}_u(\cdot))\,\boldsymbol{e}_x(\cdot)s\;\mathrm{d}s\\ &-\int_0^1 \boldsymbol{e}_u(\cdot)^T J''_{uu}(\boldsymbol{x}_\mathrm{h}(\cdot)+s\,\boldsymbol{e}_x(\cdot),\boldsymbol{u}_\mathrm{h}(\cdot)+s\,\boldsymbol{e}_u(\cdot))\,\boldsymbol{e}_u(\cdot)s\;\mathrm{d}s\\ &-2\int_0^1 \boldsymbol{e}_x(\cdot)^T J''_{xu}(\boldsymbol{x}_\mathrm{h}(\cdot)+s\,\boldsymbol{e}_x(\cdot),\boldsymbol{u}_\mathrm{h}(\cdot)+s\,\boldsymbol{e}_u(\cdot))\,\boldsymbol{e}_u(\cdot)s\;\mathrm{d}s.\end{aligned}$$

A substitution of (10.8) and (10.9) into (10.10) yields

$$-\mathcal{R}_J^h(\cdot) = \varphi_x'(\boldsymbol{x}(t_\mathrm{f}))\,\boldsymbol{e}_x(\cdot) + \sum_{n=1}^{N}\int_{\mathcal{I}_n} \psi_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{e}_x(t)\,\mathrm{d}\,\mathbf{id}(t) + \sum_{n=1}^{N}\int_{\mathcal{I}_n} \psi_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{e}_u(t)\,\mathrm{d}\,\mathbf{id}(t) - \{J(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)) - J(\boldsymbol{x}_\mathrm{h}(\cdot),\boldsymbol{u}_\mathrm{h}(\cdot))\}\,.$$

We use again Appendix A.4 to write

$$\sum_{n=1}^{N}\int_{\mathcal{I}_n} \boldsymbol{f}_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{e}_x(t)\,\mathrm{d}\,\boldsymbol{\Lambda}(t) + \sum_{n=1}^{N}\int_{\mathcal{I}_n} \boldsymbol{f}_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{e}_u(t)\,\mathrm{d}\,\boldsymbol{\Lambda}(t) - \sum_{n=1}^{N}\int_{\mathcal{I}_n} \{\boldsymbol{f}(\boldsymbol{x}(t),\boldsymbol{u}(t)) - \boldsymbol{f}(\boldsymbol{x}_\mathrm{h}(t),\boldsymbol{u}_\mathrm{h}(t))\}\;\mathrm{d}\,\boldsymbol{\Lambda}(t) = -\mathcal{R}_f^h$$

with the $\mathcal{R}_f^h$ from (10.6). We use the same approach for the mixed control–state constraint such that we have

$$\sum_{n=1}^{N}\int_{\mathcal{I}_n} \boldsymbol{c}_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{e}_x(t)\,\mathrm{d}\,\mathbf{M}(t) + \sum_{n=1}^{N}\int_{\mathcal{I}_n} \boldsymbol{c}_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{e}_u(t)\,\mathrm{d}\,\mathbf{M}(t) - \sum_{n=1}^{N}\int_{\mathcal{I}_n} \{\boldsymbol{c}(\boldsymbol{x}(t),\boldsymbol{u}(t)) - \boldsymbol{c}(\boldsymbol{x}_\mathrm{h}(t),\boldsymbol{u}_\mathrm{h}(t))\}\;\mathrm{d}\,\mathbf{M}(t) = \mathcal{R}_c^h$$

with the $\mathcal{R}_c^h$ from (10.7). A summation of the remainder terms $\mathcal{R}_J^h$, $\mathcal{R}_f^h$ and $\mathcal{R}_c^h$ yields

$$\begin{aligned}
\mathcal{R}_J^h + \mathcal{R}_f^h + \mathcal{R}_c^h = {} & -\varphi_x'(\boldsymbol{x}(t_\mathrm{f}))\,\boldsymbol{e}_x(\cdot) - \sum_{n=1}^{N}\int_{\mathcal{I}_n} \psi_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{e}_x(t)\,\mathrm{d}\,\mathbf{id}(t) \\
& - \sum_{n=1}^{N}\int_{\mathcal{I}_n} \psi_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{e}_u(t)\,\mathrm{d}\,\mathbf{id}(t) + \{J(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)) - J(\boldsymbol{x}_\mathrm{h}(\cdot),\boldsymbol{u}_\mathrm{h}(\cdot))\} \\
& - \sum_{n=1}^{N}\int_{\mathcal{I}_n} \boldsymbol{f}_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{e}_x(t)\,\mathrm{d}\,\boldsymbol{\Lambda}(t) - \sum_{n=1}^{N}\int_{\mathcal{I}_n} \boldsymbol{f}_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{e}_u(t)\,\mathrm{d}\,\boldsymbol{\Lambda}(t) \\
& + \sum_{n=1}^{N}\int_{\mathcal{I}_n} \{\boldsymbol{f}(\boldsymbol{x}(t),\boldsymbol{u}(t)) - \boldsymbol{f}(\boldsymbol{x}_\mathrm{h}(t),\boldsymbol{u}_\mathrm{h}(t))\}\;\mathrm{d}\,\boldsymbol{\Lambda}(t) \\
& + \sum_{n=1}^{N}\int_{\mathcal{I}_n} \boldsymbol{c}_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{e}_x(t)\,\mathrm{d}\,\mathbf{M}(t) + \sum_{n=1}^{N}\int_{\mathcal{I}_n} \boldsymbol{c}_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{e}_u(t)\,\mathrm{d}\,\mathbf{M}(t) \\
& - \sum_{n=1}^{N}\int_{\mathcal{I}_n} \{\boldsymbol{c}(\boldsymbol{x}(t),\boldsymbol{u}(t)) - \boldsymbol{c}(\boldsymbol{x}_\mathrm{h}(t),\boldsymbol{u}_\mathrm{h}(t))\}\;\mathrm{d}\,\mathbf{M}(t). \qquad (10.11)
\end{aligned}$$

Now, we investigate $(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot))$ and $(\boldsymbol{x}_\mathrm{h}(\cdot),\boldsymbol{u}_\mathrm{h}(\cdot))$ in $\boldsymbol{e}_x(\cdot)$ and $\boldsymbol{e}_u(\cdot)$ separately from each other on the right side of (10.11). We start with $(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot))$:

$$-\varphi_x'(\boldsymbol{x}(t_\mathrm{f}))\,\boldsymbol{x}(t_\mathrm{f}) - \sum_{n=1}^{N}\int_{\mathcal{I}_n} \psi_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{x}(t)\,\mathrm{d}\,\mathbf{id}(t)$$

$$
\begin{aligned}
&-\sum_{n=1}^{N}\int_{\mathcal{I}_n}\psi_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{u}(t)\,\mathrm{d}\,\mathbf{id}(t)-\sum_{n=1}^{N}\int_{\mathcal{I}_n}\boldsymbol{f}_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{x}(t)\,\mathrm{d}\,\boldsymbol{\Lambda}(t)\\
&-\sum_{n=1}^{N}\int_{\mathcal{I}_n}\boldsymbol{f}_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{u}(t)\,\mathrm{d}\,\boldsymbol{\Lambda}(t)+\sum_{n=1}^{N}\int_{\mathcal{I}_n}\boldsymbol{c}_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{x}(t)\,\mathrm{d}\,\mathbf{M}(t)\\
&+\sum_{n=1}^{N}\int_{\mathcal{I}_n}\boldsymbol{c}_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{u}(t)\,\mathrm{d}\,\mathbf{M}(t).
\end{aligned}
$$

A reformulation of the terms and an application of the local minimum principle equations (adjoint equation, stationarity of augmented Hamilton) and integration by parts yields

$$
\begin{aligned}
&-\boldsymbol{\varphi}_x'(\boldsymbol{x}(t_\mathrm{f}))\,\boldsymbol{x}(t_\mathrm{f})-\sum_{n=1}^{N}\left\{\int_{\mathcal{I}_n}\boldsymbol{\psi}_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{x}(t)\,\mathrm{d}\,\mathbf{id}(t)+\int_{\mathcal{I}_n}\boldsymbol{f}_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{x}(t)\,\mathrm{d}\,\boldsymbol{\Lambda}(t)\right.\\
&\left.-\int_{\mathcal{I}_n}\boldsymbol{c}_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{x}(t)\,\mathrm{d}\,\mathbf{M}(t)\right\}-\sum_{n=1}^{N}\left\{\int_{\mathcal{I}_n}\boldsymbol{\psi}_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{u}(t)\,\mathrm{d}\,\mathbf{id}(t)\right.\\
&\left.+\int_{\mathcal{I}_n}\boldsymbol{f}_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{u}(t)\,\mathrm{d}\,\boldsymbol{\Lambda}(t)-\int_{\mathcal{I}_n}\boldsymbol{c}_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{u}(t)\,\mathrm{d}\,\mathbf{M}(t)\right\}\\
&\quad=-\sum_{n=1}^{N}\int_{\mathcal{I}_n}\dot{\boldsymbol{x}}(t)\,\mathrm{d}\,\boldsymbol{\Lambda}(t)-{\lambda_{t_s}}^{T}x_s.
\end{aligned}
$$

Now, we focus on the terms in (10.11) containing $\boldsymbol{e}_x(\cdot)$ and $\boldsymbol{e}_u(\cdot)$, and pick the approximate solution parts $(\boldsymbol{x}_\mathrm{h}(\cdot),\boldsymbol{u}_\mathrm{h}(\cdot))$:

$$
\begin{aligned}
&\boldsymbol{\varphi}_x'(\boldsymbol{x}(t_\mathrm{f}))\,\boldsymbol{x}_\mathrm{h}(t_\mathrm{f})+\sum_{n=1}^{N}\int_{\mathcal{I}_n}\boldsymbol{\psi}_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{x}_\mathrm{h}(t)\,\mathrm{d}\,\mathbf{id}(t)\\
&+\sum_{n=1}^{N}\int_{\mathcal{I}_n}\boldsymbol{\psi}_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{u}_\mathrm{h}(t)\,\mathrm{d}\,\mathbf{id}(t)+\sum_{n=1}^{N}\int_{\mathcal{I}_n}\boldsymbol{f}_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{x}_\mathrm{h}(t)\,\mathrm{d}\,\boldsymbol{\Lambda}(t)\\
&+\sum_{n=1}^{N}\int_{\mathcal{I}_n}\boldsymbol{f}_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{u}_\mathrm{h}(t)\,\mathrm{d}\,\boldsymbol{\Lambda}(t)-\sum_{n=1}^{N}\int_{\mathcal{I}_n}\boldsymbol{c}_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{x}_\mathrm{h}(t)\,\mathrm{d}\,\mathbf{M}(t)\\
&-\sum_{n=1}^{N}\int_{\mathcal{I}_n}\boldsymbol{c}_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{u}_\mathrm{h}(t)\,\mathrm{d}\,\mathbf{M}(t).
\end{aligned}
$$

We use again the equations of the local minimum principle and integrate by parts such that we end up with

$$
\begin{aligned}
&\boldsymbol{\varphi}_x'(\boldsymbol{x}(t_\mathrm{f}))\,\boldsymbol{x}_\mathrm{h}(t_\mathrm{f})+\sum_{n=1}^{N}\left\{\int_{\mathcal{I}_n}\boldsymbol{\psi}_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{x}_\mathrm{h}(t)\,\mathrm{d}\,\mathbf{id}(t)+\int_{\mathcal{I}_n}\boldsymbol{f}_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{x}_\mathrm{h}(t)\,\mathrm{d}\,\boldsymbol{\Lambda}(t)\right.\\
&\left.-\int_{\mathcal{I}_n}\boldsymbol{c}_x'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{x}_\mathrm{h}(t)\,\mathrm{d}\,\mathbf{M}(t)\right\}+\sum_{n=1}^{N}\left\{\int_{\mathcal{I}_n}\boldsymbol{\psi}_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{u}_\mathrm{h}(t)\,\mathrm{d}\,\mathbf{id}(t)\right.\\
&\left.+\int_{\mathcal{I}_n}\boldsymbol{f}_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{u}_\mathrm{h}(t)\,\mathrm{d}\,\boldsymbol{\Lambda}(t)-\int_{\mathcal{I}_n}\boldsymbol{c}_u'(\boldsymbol{x}(t),\boldsymbol{u}(t))\,\boldsymbol{u}_\mathrm{h}(t)\,\mathrm{d}\,\mathbf{M}(t)\right\}
\end{aligned}
$$

$$= \sum_{n=1}^{N} \int_{\mathcal{I}_n} \dot{\boldsymbol{x}}_{\mathrm{h}}(t)\, \mathrm{d}\boldsymbol{\Lambda}(t) + {\lambda_{t_{\mathrm{s}}}}^{T} \boldsymbol{x}_{\mathrm{h}}(t_{\mathrm{s}}).$$

By putting everything together, we find

$$\begin{aligned}
&J(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)) - J(\boldsymbol{x}_{\mathrm{h}}(\cdot),\boldsymbol{u}_{\mathrm{h}}(\cdot)) - \sum_{n=1}^{N} \int_{\mathcal{I}_n} \dot{\boldsymbol{x}}(t)\, \mathrm{d}\boldsymbol{\Lambda}(t) - {\lambda_{t_{\mathrm{s}}}}^{T} x_{\mathrm{s}} \\
&+ \sum_{n=1}^{N} \int_{\mathcal{I}_n} \dot{\boldsymbol{x}}_{\mathrm{h}}(t)\, \mathrm{d}\boldsymbol{\Lambda}(t) + {\lambda_{t_{\mathrm{s}}}}^{T} \boldsymbol{x}_{\mathrm{h}}(t_{\mathrm{s}}) + \sum_{n=1}^{N} \int_{\mathcal{I}_n} \{\boldsymbol{f}(\boldsymbol{x}(t),\boldsymbol{u}(t)) - \boldsymbol{f}(\boldsymbol{x}_{\mathrm{h}}(t),\boldsymbol{u}_{\mathrm{h}}(t))\}\ \mathrm{d}\boldsymbol{\Lambda}(t) \\
&- \sum_{n=1}^{N} \int_{\mathcal{I}_n} \{\boldsymbol{c}(\boldsymbol{x}(t),\boldsymbol{u}(t)) - \boldsymbol{c}(\boldsymbol{x}_{\mathrm{h}}(t),\boldsymbol{u}_{\mathrm{h}}(t))\}\ \mathrm{d}\mathbf{M}(t) = \mathcal{R}_h.
\end{aligned}$$

Rearranging terms yields

$$\begin{aligned}
&J(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)) - J(\boldsymbol{x}_{\mathrm{h}}(\cdot),\boldsymbol{u}_{\mathrm{h}}(\cdot)) - \sum_{n=1}^{N} \int_{\mathcal{I}_n} \dot{\boldsymbol{x}}(t) - \boldsymbol{f}(\boldsymbol{x}(t),\boldsymbol{u}(t))\, \mathrm{d}\boldsymbol{\Lambda}(t) \\
&+ \sum_{n=1}^{N} \int_{\mathcal{I}_n} \dot{\boldsymbol{x}}_{\mathrm{h}}(t) - \boldsymbol{f}(\boldsymbol{x}_{\mathrm{h}}(t),\boldsymbol{u}_{\mathrm{h}}(t))\, \mathrm{d}\boldsymbol{\Lambda}(t) - \sum_{n=1}^{N} \int_{\mathcal{I}_n} \boldsymbol{c}(\boldsymbol{x}(t),\boldsymbol{u}(t))\, \mathrm{d}\mathbf{M}(t) \\
&+ \sum_{n=1}^{N} \int_{\mathcal{I}_n} \boldsymbol{c}(\boldsymbol{x}_{\mathrm{h}}(t),\boldsymbol{u}_{\mathrm{h}}(t))\, \mathrm{d}\mathbf{M}(t) + {\lambda_{t_{\mathrm{s}}}}^{T} \{\boldsymbol{x}_{\mathrm{h}}(t_{\mathrm{s}}) - x_{\mathrm{s}}\} = \mathcal{R}_h.
\end{aligned}$$

The terms $\sum_{n=1}^{N} \int_{\mathcal{I}_n} \dot{\boldsymbol{x}}(t) - \boldsymbol{f}(\boldsymbol{x}(t),\boldsymbol{u}(t))\, \mathrm{d}\boldsymbol{\Lambda}(t)$ and $\sum_{n=1}^{N} \int_{\mathcal{I}_n} \boldsymbol{c}(\boldsymbol{x}(t),\boldsymbol{u}(t))\, \mathrm{d}\mathbf{M}(t)$ vanish since the differential equation and the complementarity condition of the local minimum principle holds for the exact solution such that we end up with the error representation

$$\begin{aligned}
J(\boldsymbol{x}_{\mathrm{h}}(\cdot),\boldsymbol{u}_{\mathrm{h}}(\cdot)) - J(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)) = &\sum_{n=1}^{N} \int_{\mathcal{I}_n} \dot{\boldsymbol{x}}_{\mathrm{h}}(t) - \boldsymbol{f}(\boldsymbol{x}_{\mathrm{h}}(t),\boldsymbol{u}_{\mathrm{h}}(t))\, \mathrm{d}\boldsymbol{\Lambda}(t) \\
&+ \sum_{n=1}^{N} \int_{\mathcal{I}_n} \boldsymbol{c}(\boldsymbol{x}_{\mathrm{h}}(t),\boldsymbol{u}_{\mathrm{h}}(t))\, \mathrm{d}\mathbf{M}(t) + {\lambda_{t_{\mathrm{s}}}}^{T} \{\boldsymbol{x}_{\mathrm{h}}(t_{\mathrm{s}}) - x_{\mathrm{s}}\} + \mathcal{R}_h.
\end{aligned}$$

Since $(\boldsymbol{x}_{\mathrm{h}}(\cdot),\boldsymbol{u}_{\mathrm{h}}(\cdot))$ solves the FE Petrov–Galerkin equations and since $\boldsymbol{i}_{\mathrm{h}}\boldsymbol{\Lambda} \in \mathcal{Z}_H(\mathcal{T},\mathbb{R}^{n_{\mathrm{x}}})$ and $\boldsymbol{i}_{\mathrm{h}}\mathbf{M} \in \mathcal{Z}_H(\mathcal{T},\mathbb{R}^{n_{\mathrm{c}}})$ the error representation (10.5) holds true. □

Note that the stability of the continuous OCP in the error representation of Theorem 10.1 is expressed by means of the weak adjoint solutions $\boldsymbol{\Lambda}(\cdot)$ and $\mathbf{M}(\cdot)$ but not by the approximate adjoint functions $\boldsymbol{\Lambda}_{\mathrm{h}}(\cdot)$ and $\mathbf{M}_{\mathrm{h}}(\cdot)$ of the FE Petrov–Galerkin discretization. The weights $\boldsymbol{\Lambda} - \boldsymbol{i}_{\mathrm{h}}\boldsymbol{\Lambda}$ and $\mathbf{M} - \boldsymbol{i}_{\mathrm{h}}\mathbf{M}$ in the error representation of Theorem 10.1 particularly include the local interpolation error of the exact weak adjoints in $\mathcal{NBV}(\mathcal{T},\mathbb{R}^{n_{\mathrm{x}}})$ resp. $\mathcal{NBV}(\mathcal{T},\mathbb{R}^{n_{\mathrm{c}}})$ by their interpolants in $\mathcal{Z}_H(\mathcal{T},\mathbb{R}^{n_{\mathrm{x}}})$ resp. $\mathcal{Z}_H(\mathcal{T},\mathbb{R}^{n_{\mathrm{c}}})$. The problem how to find reasonable guesses for the unknown functions $\boldsymbol{\Lambda}(\cdot)$, $\mathbf{M}(\cdot)$ and for $\lambda_{t_{\mathrm{s}}}$ is investigated in the following Section 10.3.

10.3 Approximation of the Error Representation

In this section, we use the error representation (10.5) of Theorem 10.1 to derive an approximation of the error $J(\boldsymbol{x}_\mathrm{h}(\cdot),\boldsymbol{u}_\mathrm{h}(\cdot))-J(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot))$. Due to the particular element–wise representation of the global error, the approximation also provides information on the local error contribution. This can be exploited for the development of adaptive mesh refinement algorithms. We will illustrate this in the next section, where we will present one particular *hp* approach.

In the evaluation of (10.5), we neglect the terms $\left[\lambda_{t_\mathrm{s}}-\boldsymbol{i}_\mathrm{h}\,\lambda_{t_\mathrm{s}}\right]^T\{\boldsymbol{x}_\mathrm{h}(t_\mathrm{s})-x_\mathrm{s}\}$ and the remainder $\mathcal{R}_h$. This is reasonable because $\boldsymbol{x}_\mathrm{h}(t_\mathrm{s})=x_\mathrm{s}$ is part of the FE Petrov–Galerkin formulation and because $\mathcal{R}_h$ is quadratic in the error $\boldsymbol{e}(\cdot)$. In the literature, especially in the PDE DWR literature, there exist several approaches, how the weights $\boldsymbol{\Lambda}-\boldsymbol{i}_\mathrm{h}\,\boldsymbol{\Lambda}$ resp. $\mathbf{M}-\boldsymbol{i}_\mathrm{h}\,\mathbf{M}$ can be approximated. Usually, one uses a higher order interpolation based on the approximate adjoint solutions $\boldsymbol{\Lambda}_\mathrm{h}(\cdot)$ and $\mathbf{M}_\mathrm{h}(\cdot)$. In this contribution, we present an approach that provides an easy implementation and shows promising numerical results (see Chapter 14). However, there is still much research to do in order to end up with more suitable results.

The first step towards an approximation of (10.5) is to pass over from $\boldsymbol{\Lambda}_\mathrm{h}(\cdot)$ and $\mathbf{M}_\mathrm{h}(\cdot)$, where we take their representations (9.46) and (9.47), to their "derivatives". Even though $\boldsymbol{\Lambda}_\mathrm{h}(\cdot)$ and $\mathbf{M}_\mathrm{h}(\cdot)$ are not differentiable in the classical sense there still exist weak derivatives, which we denote with $\boldsymbol{\lambda}_\mathrm{h}(\cdot)$ and $\boldsymbol{\mu}_\mathrm{h}(\cdot)$. The weak derivatives are given by the Dirac measures at the discretization points $\left\{t_i^{(n)}\right\}$ with heights $\left\{\Lambda_i^{(n)}\right\}$ resp. $\left\{M_i^{(n)}\right\}$. Therefore, we set up $\boldsymbol{\lambda}_\mathrm{h}(\cdot)$ and $\boldsymbol{\mu}_\mathrm{h}(\cdot)$ as the interpolating polynomials with the interpolation points $\left\{\left(t_i^{(n)},\Lambda_i^{(n)}\right)\right\}$ and $\left\{\left(t_i^{(n)},M_i^{(n)}\right)\right\}$. We obtain the higher order approximations for $\boldsymbol{\lambda}_\mathrm{h}(\cdot)$ and $\boldsymbol{\mu}_\mathrm{h}(\cdot)$ on FEs by employing neighboring FEs. More specifically, for the m–th FE we use the data set $\left\{\left(t_i^{(n)},\Lambda_i^{(n)}\right)\right\}_{m-1\leqslant n\leqslant m+1}$ to construct an interpolating polynomial which acts as an approximation for $\boldsymbol{\lambda}(\cdot)$. We can do the same with $\boldsymbol{\mu}(\cdot)$. We denote the higher order polynomials with $\boldsymbol{\lambda}_\mathrm{h}^\mathrm{ho}(\cdot)$ and $\boldsymbol{\mu}_\mathrm{h}^\mathrm{ho}(\cdot)$. The first and last FE is handled by just taking the single neighboring FE into account. With these considerations we can approximate the first two summands of (10.5) by integrals of the form

$$\int_{\mathcal{I}_n}\left\{\boldsymbol{\lambda}_\mathrm{h}^\mathrm{ho}(t)-\boldsymbol{\lambda}_\mathrm{h}(t)\right\}^T\left\{\dot{\boldsymbol{x}}_\mathrm{h}(t)-\boldsymbol{f}(\boldsymbol{x}_\mathrm{h}(t),\boldsymbol{u}_\mathrm{h}(t))\right\}\,\mathrm{d}t\quad\text{and}\tag{10.12}$$

$$\int_{\mathcal{I}_n}\left\{\boldsymbol{\mu}_\mathrm{h}^\mathrm{ho}(t)-\boldsymbol{\mu}_\mathrm{h}(t)\right\}^T\boldsymbol{c}(\boldsymbol{x}_\mathrm{h}(t),\boldsymbol{u}_\mathrm{h}(t))\,\mathrm{d}t.\tag{10.13}$$

All the information necessary to compute these two integrals ($\boldsymbol{\lambda}_\mathrm{h}^\mathrm{ho}$, $\boldsymbol{\mu}_\mathrm{h}^\mathrm{ho}$, $\boldsymbol{\lambda}_\mathrm{h}$, $\boldsymbol{\mu}_\mathrm{h}$, $\boldsymbol{x}_\mathrm{h}$, $\boldsymbol{u}_\mathrm{h}$) is available, the only thing remaining is to evaluate these integrals. If we would use a LGR quadrature rule of the same order as the polynomial degree of $\boldsymbol{\lambda}_\mathrm{h}(\cdot)$ and $\boldsymbol{\mu}_\mathrm{h}(\cdot)$ on the respective FE the integrals would always be evaluated to zero since $\boldsymbol{\lambda}_\mathrm{h}(\cdot)$ interpolates $\boldsymbol{\lambda}_\mathrm{h}^\mathrm{ho}(\cdot)$. The same holds for $\boldsymbol{\mu}_\mathrm{h}(\cdot)$ and $\boldsymbol{\mu}_\mathrm{h}^\mathrm{ho}(\cdot)$. Therefore, we increase the order for the LGR quadrature.

10.4 Towards an Adaptive Mesh Refinement

In this section, we distill an adaptive mesh refinement algorithm based on the error representation of Theorem 10.1 and its approximation which was described in the previous Section 10.3. The presented method is an *hp* approach and makes use of local error contributions to the global error with respect to the performance criterion as well as a curvature value to measure the smoothness of approximate solution trajectories on FEs and to decide if the polynomial degree should be increased on certain FEs or if an FE should be divided into at least two FEs.
Starting with an initial discretization scheme, we determine an approximate solution to the OCP we actually want to solve. Next, we determine the local element–wise error contributions to the error $J(\boldsymbol{x}_{\mathrm{h}}(\cdot),\boldsymbol{u}_{\mathrm{h}}(\cdot))-J(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot))$. FEs whose estimated errors exceed the predefined termination tolerance are either subdivided into FEs of smaller size or the degree of the approximating polynomials is increased on the respective FE.
For the decision if either the FE grid or the polynomial degree should be adapted we rely on the contributions of Liu et al. [299]. The smoothness of the solution on a FE mainly determines their adaptations of the discretization scheme. In accordance with our considerations about spectral methods in Section 6.3.2 we know that they are mainly suitable for problems with smooth solutions. In case of existing discontinuities one would prefer local methods (Gibbs phenomenon). Thus, the FE grid is refined in case of non–smooth solutions while the polynomial degree is increased for sufficiently smooth solutions.
The criterion if a solution is considered to be smooth or not is based on whether the ratio between the maximum second derivative of the solution in the current iteration and the one in the previous iteration exceeds a certain predefined parameter value. This means that the algorithm needs solution information of the current and the previous iteration. In our numerical experiments, we just equally bisected all FEs after the first iteration.

Non–smoothness Detection In order to describe the procedure let $\boldsymbol{X}^{(K)}=[\boldsymbol{X}_1^{(K)},\dots,\boldsymbol{X}_{n_{\mathrm{x}}}^{(K)}]^T$ denote the approximate solution in iteration K. Let furthermore $t_{i,j}$ denote the time instants in the interior of FEs where $|\ddot{\boldsymbol{X}}_i^{(K)}(\cdot)|$ has its local maxima. We introduce the notation $P_{i,j}^{(K)}\overset{\text{def}}{=}|\ddot{\boldsymbol{X}}_i^{(K)}(t_{i,j})|$. In the same way, let $P_{i,j}^{(K-1)}$ denote the maximum value of the function $|\ddot{\boldsymbol{X}}_i^{(K-1)}(\cdot)|$ in the interior of the FE ($(K-1)$–th FE grid) that contains $t_{i,j}$. For a given predefined real parameter $\overline{R}>0$ we call a solution non–smooth on the FE $\mathcal{I}_n$ if

$$R_{i,j}=\frac{P_{i,j}^{(K)}}{P_{i,j}^{(K-1)}}\geqslant\overline{R} \tag{10.14}$$

holds for any $t_{i,j}\in\mathcal{I}_n$.

An A–Priori Error Estimation Result If the discretization on a FE must be updated according to condition (10.14) there has to be determined the new number of FEs or the new polynomial degree for the approximating polynomials. Similarly to standard refinement techniques in the ODE numerics, we rely on a a–priori error estimation result of Hou [243] that guarantees for the OCP solution $(\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot))$ and the approximate solution $(\boldsymbol{X}(\cdot),\boldsymbol{U}(\cdot))$ that the

estimate

$$\|\boldsymbol{X}-\boldsymbol{x}\|_\infty + \|\boldsymbol{U}-\boldsymbol{u}\|_\infty \leqslant \frac{c\,h^q}{N^{q-2.5}} \tag{10.15}$$

holds under suitable assumptions. In (10.15), c denotes a constant, N the number of the approximating polynomial degree, h the FE length, q the minimum of N and the number of continuous derivatives in the solution, and $\|\cdot\|_\infty$ the uniform norm over the FE grid points.

Dividing a FE For dividing a FE $\mathcal{I}_n$ where the non–smoothness condition (10.14) indicates a non–smooth solution, we use (10.15) with equality such that the error $e_n^{(K)}$ in iteration K is estimated as

$$e_n^{(K)} = \frac{c\left[h_n^{(K)}\right]^q}{\left[N_n^{(K)}\right]^{q-2.5}}. \tag{10.16}$$

The error on the respective FE grid in iteration $(K+1)$ should satisfy the predefined termination tolerance ε such that we require

$$\varepsilon = \frac{c\left[h_n^{(K+1)}\right]^q}{\left[N_n^{(K+1)}\right]^{q-2.5}}. \tag{10.17}$$

Since the approximating polynomial degree is not changed from iteration K to $(K+1)$, we assume $N_n^{(K)} = N_n^{(K+1)}$. The ratio $h_n^{(K)}/h_n^{(K+1)}$ indicating how many FEs must be created in FE $\mathcal{I}_n$ from iteration K to $(K+1)$ follows straight from combining (10.16) and (10.17):

$$H \overset{\text{def}}{=} \frac{h_n^{(K)}}{h_n^{(K+1)}} = \left[\frac{e_n^{(K)}}{\varepsilon}\right]^{1/q}.$$

Under the assumption that q does not change from iteration $(K-1)$ to K, we obtain a guess for q by exploiting the a–priori error estimation for iteration $(K-1)$:

$$e_n^{(K-1)} = \frac{c\left[h_n^{(K-1)}\right]^q}{\left[N_n^{(K-1)}\right]^{q-2.5}}. \tag{10.18}$$

We can solve for q by combining (10.16) and (10.18) since c is eliminated. It may happen that the number of newly created sub–FEs from one iteration to the other is disproportionately high. Hence, there should exist an upper limit for that number. The limit should be reduced as the actual error $e_n^{(K)}$ approaches the termination tolerance ε. Liu et al. [299] propose the limit

$$H_{\max} = \left\lceil \log_{N_n^{(K)}}(e_n^{(K)}/\varepsilon)\right\rceil$$

such that the number of sub–FEs of FE $\mathcal{I}_n$ becomes

$$S = \min(\lceil H \rceil, H_{\max}).$$

Increasing the Approximating Polynomial Degree Updating the approximating polynomial degree works pretty similar to the previous case. In case the condition (10.14) proposes a smooth FE, we assume $h_n^{(K)} = h_n^{(K+1)}$ and calculate $N_n^{(K+1)}$ from (10.16) and (10.17) as

$$N_n^{(K+1)} = N_n^{(K)} \left[\frac{e_n^{(K)}}{\varepsilon} \right]^{1/(q-2.5)},$$

where we use the same approach as before to find a guess for q. To end up with a natural number for the new polynomial degree, we use

$$N_n^{(K+1)} = \left\lceil N_n^{(K)} \left[\frac{e_n^{(K)}}{\varepsilon} \right]^{1/(q-2.5)} \right\rceil.$$

In order to avoid extraordinary high polynomial degrees, we can set an upper limit $N_{\max}$. If this upper limit would be exceeded one would keep the previous polynomial degree but equally divide the FE into sub–FEs.

Concluding Remarks A major drawback of the described mesh refinement method is that it does not allow for coarsening the FE grid and to reduce the degree of the approximating polynomial. Hence, the algorithms may produce discretization schemes of unnecessarily high resolution. Furthermore, there may be an accumulation of FE grid points in a vicinity of solution discontinuities. For those reasons, it is highly recommended to augment the algorithm with respective functionality. Liu et al. [299] propose some approaches to do so. One may also think of other extensions such as an equilibration of the local error contributions over the optimization horizon. In summary, one can say that there is still plenty of potential for further research on our novel DWR approach to ODE OCP.

Chapter 11

A Unified Framework for Optimal Control Problems with Switches

This chapter is devoted to present a novel method to switched optimal control that we published in Bock et al. [78] and which serves as basis for this chapter. The method solves explicitly and implicitly switched OCPs in a unified way by setting up an equivalent counterpart problem with additional binary control functions. By means of techniques from generalized disjunctive programming, mixed–integer optimal control, and a direct simultaneous approach to optimal control the problem is processed further resulting in a MPVC. Aiming to determine approximations to the original switched OCP we construct a sequence of NLPs, whose instances are based on gradually solving the MPVC instance and on a simultaneous adaption of the discretization grid towards an identification of the OCP switching structure.

Section 11.1 reviews relevant literature about solution approaches to explicitly and implicitly switched OCPs. In Section 11.2 we establish the problem which is investigated in this chapter. In Sections 11.3, 11.4, and 11.5 we employ techniques from hybrid control, generalized disjunctive programming and MIOCP to transform the switched OCP into a new problem with additional constraints and boolean control functions. Recent results justify the additional step to drop the integrality conditions. The approach to discretize the resulting continuous OCP is outlined in Section 11.6 while the ideas to handle the MPVC that arises from the discretization step are described in Section 11.7. Finally, in Section 11.8 we sketch the SNLP approach to solve the MPVC and to detect the switching structure simultaneously.

11.1 Literature Review

A majority of algorithms that solve EFS systems can be assigned to two method types: *two-stage optimization* and *embedding transformation.* Two-stage optimization algorithms work as follows: the switching sequence σ is assumed to be fixed in stage 1. The switching times τ and the optimal control input $\boldsymbol{u}$ is optimized. The switching sequence is updated in level stage 2. This process is repeated until it convergences. Several authors (see [313, 302, 202, 203, 456]) independently use a bi–level hierarchical algorithm to solve stage 1 and stage 2. In [327] a direct simultaneous method is used to solve the stage 1 problem. A MINLP master problem updates the switching sequence. Allgor and Barton [10] and Bansal et al. [28] employ a direct single shooting approach for the stage 1 problem. Algorithms that use gradient projection as well as constrained Newton's method can be found in the publications of Xu and Antsaklis [458, 459].

Embedding transformation methods do not use a switched dynamic system in the form of (1.5)

but work with the relaxed formulation of (1.6), i.e., they consider continuous systems

$$\dot{\boldsymbol{x}}(t) = \sum_{i=1}^{n_\omega} \boldsymbol{\alpha}_i(t)\, \boldsymbol{f}_i(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), \quad \boldsymbol{x}(t_s) = x_s, \tag{11.1}$$

where $\boldsymbol{\alpha}_i(t) \in [0,1]$ and $\sum_{i=1}^{n_\omega} \boldsymbol{\alpha}_i(t) = 1$. There is no assumption required about the number of switches at the beginning of the optimization. The same holds for the switching sequence σ and the switching times τ. The approach has been developed independently by Sager [380] and Bengea and DeCarlo [49] for EFS systems. Standard OCP algorithms such as direct multiple shooting (see [381]) and direct collocation methods (see [447]) can be used to solve the resulting continuous OCPs. Our approach rests on the idea of complementarity based formulations for IFS systems developed by Baumrucker and Biegler [33] and combines them with the idea of embedding transformation. As a result thereof we develop a unified approach for systems undergoing explicit and implicit switches. Moreover, the discretized problems of our approach belong to the subclass of MPVCs. As we had seen in Chapter 4 MPVCs allow for tailored first–order optimality systems compared to the larger class of MPECs.
The literature, which deals with IFS problems, mostly focuses on Piecewise Affine (PWA) systems, cf. [46, 249, 270, 374]. A PWA system

$$\boldsymbol{x}(t+1) = A_i\, \boldsymbol{x}(t) + B_i\, \boldsymbol{u}(t) + f_i, \text{ if} \tag{11.2}$$

$$\begin{bmatrix} \boldsymbol{x}(t) \\ \boldsymbol{u}(t) \end{bmatrix} \in \mathcal{X}_i \triangleq \left\{ \begin{bmatrix} x \\ u \end{bmatrix} \,:\, G_i\, x + H_i\, u \leqslant K_i \right\}, \tag{11.3}$$

partitions the state space into polyhedral regions $\{\mathcal{X}_i\}_{i=1}^{M}$. Each region $\mathcal{X}_i$ is associated with a linear difference equation. The system can be augmented with constraints

$$I\, \boldsymbol{x}(t) + J\, \boldsymbol{u}(t) \leqslant L, \tag{11.4}$$

which are independent of the mode. Considered objectives are of type

$$\boldsymbol{J}\left(\boldsymbol{u}(0), \ldots, \boldsymbol{u}(T-1), \boldsymbol{x}(0)\right) \triangleq \|P\boldsymbol{x}(T)\|_p \; + \; \sum_{j=0}^{T-1} \left(\|Q\boldsymbol{x}(j)\|_p + \|R\boldsymbol{u}(j)\|_p \right). \tag{11.5}$$

We can distinguish between two different types of approaches to solve optimization problems subject to PWA systems. For the first approach, the problem is considered within a Mixed Logical Dynamic (MLD) framework. The resulting problems are difference equations incorporating continuous as well as boolean variables. The interested reader can find more information on MLD in the article of [45]. Embedding PWA problems into the MLD framework results in either Mixed Integer Quadratic Programs (MIQPs) for the choice $p = 2$ (see [45]) in (11.5) or (if $p \in \{1, \infty\}$) in MILPs (see [47]). Borrelli et al. [79] and Borrelli et al. [80] proposed a second approach to solve PWA problems . It is based on a combination of dynamic programming and multi–parametric programming techniques.
More general IFS OCPs can be solved by a combination of a simultaneous optimization method such as direct multiple shooting (see [75]) with a tailored switch detecting ODE solver (see

[73, 83, 271]). The identification of switching times and the implementation of a sensitivity update mechanism at state discontinuities is crucial for those switch detecting solvers, cf. [83, 271, 379, 237, 289]. Multi–phase OCPs can be considered as the method of choice for models with a priori known number of switching points as well as their sequence, cf. [328, 443]). Implicit discontinuities do not have to be treated explicitly under these conditions.

11.2 Problem Formulation

We consider the OCP with explicit and implicit switches from Definition 1.24. In order to shorten the notation we employ an autonomous OCP formulation and drop the $\boldsymbol{d}(\cdot)$ constraints. Thus, we consider the OCP

$$
\begin{aligned}
\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),\boldsymbol{v}(\cdot)} \quad & \boldsymbol{\varphi}(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})) && && \text{(11.6a)}\\
\text{s.t.} \quad & \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t),\boldsymbol{u}(t),\boldsymbol{v}(t),\operatorname{sgn}(\boldsymbol{\sigma}(\boldsymbol{x}(t)))), && t\in\mathcal{T}, && \text{(11.6b)}\\
& \boldsymbol{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(\boldsymbol{x}(t),\boldsymbol{u}(t),\boldsymbol{v}(t)), && t\in\mathcal{T}, && \text{(11.6c)}\\
& \boldsymbol{0}_{n_\mathrm{r}} = \boldsymbol{r}(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})), && && \text{(11.6d)}\\
& \boldsymbol{v}(t)\in\Omega\subset\mathbb{R}^{n_\mathrm{v}}, && t\in\mathcal{T},\ |\Omega| = n_\omega < \infty. && \text{(11.6e)}
\end{aligned}
$$

The meaning of all arising functions and sets does not change compared to Definition 1.24.

11.3 Optimal Control of Hybrid Systems

Similarly to Example 1.6 we reformulate OCP (11.6) in the sense of hybrid optimal control by means of an indexed set of differential equations. We distinguish two separate cases, namely the one where the transversality conditions holds (consistent switching) and the one of inconsistent switching. Bifurcation are not investigated in this thesis.

Consistent Switching

In the consistent case we can equivalently rewrite OCP (11.6) as

$$
\begin{aligned}
\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),\boldsymbol{v}(\cdot)} \quad & \boldsymbol{\varphi}(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})) && && \text{(11.7a)}\\
\text{s.t.} \quad & \dot{\boldsymbol{x}}(t) = \begin{cases} \boldsymbol{f}^-(\boldsymbol{x}(t),\boldsymbol{u}(t),\boldsymbol{v}(t)) & \text{if } \boldsymbol{\sigma}(\boldsymbol{x}(t)) < 0,\\ \boldsymbol{f}^+(\boldsymbol{x}(t),\boldsymbol{u}(t),\boldsymbol{v}(t)) & \text{if } \boldsymbol{\sigma}(\boldsymbol{x}(t)) > 0, \end{cases} && t\in\mathcal{T}, && \text{(11.7b)}\\
& \boldsymbol{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(\boldsymbol{x}(t),\boldsymbol{u}(t),\boldsymbol{v}(t)), && t\in\mathcal{T}, && \text{(11.7c)}\\
& \boldsymbol{0}_{n_\mathrm{r}} = \boldsymbol{r}(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})), && && \text{(11.7d)}\\
& \boldsymbol{v}(t)\in\Omega, && t\in\mathcal{T}, && \text{(11.7e)}
\end{aligned}
$$

where the functions $\boldsymbol{f}^- : \mathcal{X} \times \mathcal{U} \times \Omega \longrightarrow \mathbb{R}^{n_x}$ and $\boldsymbol{f}^+ : \mathcal{X} \times \mathcal{U} \times \Omega \longrightarrow \mathbb{R}^{n_x}$ are chosen appropriately as

$$\begin{aligned}\boldsymbol{f}^-(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t)) &= \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t), -1),\\ \boldsymbol{f}^+(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t)) &= \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t), +1).\end{aligned}$$

Inconsistent Switching

If the transversality assumption is violated (inconsistent switching), we additionally have to consider the Filippov (see Section 1.3) case of sliding on the zero manifold Σ:

$$\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),\boldsymbol{v}(\cdot)} \quad \boldsymbol{\varphi}(\boldsymbol{x}(t_s), \boldsymbol{x}(t_f)) \tag{11.8a}$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}}(t) = \begin{cases} \boldsymbol{f}^-(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t)) & \text{if } \boldsymbol{\sigma}(\boldsymbol{x}(t)) < 0, \\ \boldsymbol{f}^0(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t)) & \text{if } \boldsymbol{\sigma}(\boldsymbol{x}(t)) = 0, \\ \boldsymbol{f}^+(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t)) & \text{if } \boldsymbol{\sigma}(\boldsymbol{x}(t)) > 0, \end{cases} \quad t \in \mathcal{T}, \tag{11.8b}$$

$$\mathbf{0}_{n_c} \geqslant \boldsymbol{c}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t)), \qquad t \in \mathcal{T}, \tag{11.8c}$$

$$\mathbf{0}_{n_r} = \boldsymbol{r}(\boldsymbol{x}(t_s), \boldsymbol{x}(t_f)), \tag{11.8d}$$

$$\boldsymbol{v}(t) \in \Omega, \qquad t \in \mathcal{T}, \tag{11.8e}$$

where the functions $\boldsymbol{f}^- : \mathcal{X}\times\mathcal{U}\times\Omega \to \mathbb{R}^{n_x}$, $\boldsymbol{f}^0 : \mathcal{X}\times\mathcal{U}\times\Omega \to \mathbb{R}^{n_x}$ as well as $\boldsymbol{f}^+ : \mathcal{X}\times\mathcal{U}\times\Omega \to \mathbb{R}^{n_x}$ are chosen respectively as

$$\begin{aligned}\boldsymbol{f}^-(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t)) &= \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t), -1),\\ \boldsymbol{f}^0(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t)) &= \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t), 0),\\ \boldsymbol{f}^+(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t)) &= \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t), +1).\end{aligned}$$

The objective function $\boldsymbol{\varphi}$ and the constraint functions $\boldsymbol{c}$, $\boldsymbol{r}$ do not change in (11.7) and (11.8). For $n_\sigma > 1$ we would obtain 2^{n_σ} cases for consistent switches in the worst case. If the transversality assumption does not hold for any switch, there could be as many as 3^{n_σ}. Usually, however, not all components of $\boldsymbol{f}$ depend on all switches, and a much better complexity is observed during reformulation (see Example 1.28).
An approach very close in spirit was described by Baumrucker and Biegler [33]. Note, however, that our approach does not reformulate the original ordinary differential equation with discontinuous right hand side into a differential algebraic equation on intervals with $\boldsymbol{\sigma} \equiv 0$, but rather leaves the resolution of the sliding mode right hand side to the choice of the convex multipliers within the optimizer.

11.4 Generalized Disjunctive Programming

Optimization problems involving both continuous and discrete variables are often solved with techniques from mixed–integer optimization. Disjunctive Programming (DP) is an alterna-

tive approach to solve this kind of problems, see e.g. [27]. DP models consist of algebraic constraints, logic disjunctions and logic propositions. A particular case of disjunctive programming named Generalized Disjunctive Programming (GDP) was developed by Raman and Grossmann [367]. A GDP problem reads as

$$\min_{x\in\mathbb{R}^n,\omega_{ik}\in\{0,1\}} \quad \psi(x)+\sum_{k\in K} c_k \tag{11.9a}$$

$$\text{s.t.} \quad \mathbf{0}\geqslant \boldsymbol{g}(x), \tag{11.9b}$$

$$\bigoplus_{i\in\mathcal{D}_k}\begin{bmatrix}\omega_{ik}=1\\ \boldsymbol{s}_{ik}(x)\leqslant 0\\ c_k=\gamma_{ik}\end{bmatrix}, \quad k\in\mathcal{K}, \tag{11.9c}$$

$$1=\Omega(\omega), \tag{11.9d}$$

$$x\in[x^{\mathrm{l}},x^{\mathrm{u}}], \tag{11.9e}$$

where $\mathcal{K}\overset{\text{def}}{=}\{1,\dots,K\}$ and $\mathcal{D}_k\overset{\text{def}}{=}\{1,\dots,D_k\}$. The problem involves continuous variables $x\in\mathbb{R}^n$ in the bounds $[x^{\mathrm{l}},x^{\mathrm{u}}]$ and binary variables $\omega\overset{\text{def}}{=}\{\omega_{ik}\}_{i,k}$, $\omega_{ik}\in\{0,1\}$. The objective function $\psi:\mathbb{R}^n\longrightarrow\mathbb{R}$ and the global constraint function $\boldsymbol{s}:\mathbb{R}^n\longrightarrow\mathbb{R}^m$ are supposed to be sufficiently smooth. K logical expression must hold. Each of these expressions is composed of D_k terms that are connected by the EX-OR operator $\oplus$, indicating that exactly one of the boolean variables ω_{ik} must be set to one. If that is the case for a particular variable ω_{ik}, the associated constraint $\boldsymbol{s}_{ik}(x)\leqslant 0$ and the objective weight c_k are enforced. They are ignored for all $\omega_{ik}=0$. The constraint $\Omega(\omega)=1$ summarizes further constraints on the boolean variables ω_{ik}.

By reformulating the different ODE equation branches together with the $\boldsymbol{c}$–constraints of Problem (11.8) in the GDP framework we can define

$$Y_\nu^+(t)\overset{\text{def}}{=}\begin{bmatrix}\boldsymbol{\omega}_\nu^+(t)=1\\ \dot{\boldsymbol{x}}(t)=\boldsymbol{f}^+(\boldsymbol{x}(t),\boldsymbol{u}(t),\nu)\\ \mathbf{0}_{n_{\mathrm{c}}}\geqslant\boldsymbol{c}(\boldsymbol{x}(t),\boldsymbol{u}(t),\nu)\\ \boldsymbol{\sigma}(\boldsymbol{x}(t))>0\end{bmatrix},\quad Y_\nu^-(t)\overset{\text{def}}{=}\begin{bmatrix}\boldsymbol{\omega}_\nu^-(t)=1\\ \dot{\boldsymbol{x}}(t)=\boldsymbol{f}^-(\boldsymbol{x}(t),\boldsymbol{u}(t),\nu)\\ \mathbf{0}_{n_{\mathrm{c}}}\geqslant\boldsymbol{c}(\boldsymbol{x}(t),\boldsymbol{u}(t),\nu)\\ \boldsymbol{\sigma}(\boldsymbol{x}(t))<0\end{bmatrix},$$

and

$$Y_\nu^0(t)\overset{\text{def}}{=}\begin{bmatrix}\boldsymbol{\omega}_\nu^0(t)=1\\ \dot{\boldsymbol{x}}(t)=\boldsymbol{f}^0(\boldsymbol{x}(t),\boldsymbol{u}(t),\nu)\\ \mathbf{0}_{n_{\mathrm{c}}}\geqslant\boldsymbol{c}(\boldsymbol{x}(t),\boldsymbol{u}(t),\nu)\\ \boldsymbol{\sigma}(\boldsymbol{x}(t))=0\end{bmatrix}.$$

Consistent Switching

The full OCP with a consistent switch then reads as

$$\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),\boldsymbol{\omega}(\cdot)} \quad \boldsymbol{\varphi}(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})) \tag{11.10a}$$

$$\text{s.t.} \quad \bigoplus_{\nu\in\Omega}\left[Y_\nu^-(t)\oplus Y_\nu^+(t)\right], \quad t\in\mathcal{T}, \tag{11.10b}$$

$$\mathbf{0}_{n_\mathrm{r}} = \boldsymbol{r}(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})). \tag{11.10c}$$

Inconsistent Switching

The full OCP with an inconsistent switch reads as

$$\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),\boldsymbol{\omega}(\cdot)} \quad \boldsymbol{\varphi}(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})) \tag{11.11a}$$

$$\text{s.t.} \quad \bigoplus_{\nu\in\Omega}\left[Y_\nu^-(t)\oplus Y_\nu^0(t)\oplus Y_\nu^+(t)\right], \quad t\in\mathcal{T} \tag{11.11b}$$

$$\mathbf{0}_{n_\mathrm{r}} = \boldsymbol{r}(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})). \tag{11.11c}$$

For both the consistently and the inconsistently switched problem the disjunction over all clauses in the brackets and all explicitly switchable modes in Ω must hold at any point $t \in \mathcal{T}$ and the objective function $\boldsymbol{\varphi}$ as well as the global constraints $\boldsymbol{c}$ and $\boldsymbol{r}$ do not depend on the explicitly and implicitly switched mode. The equivalent reformulation of Problem (11.8) to Problem (11.10) resp. (11.11) is crucial since an implicitly switched system is transformed into an explicit one by introducing additional boolean control variables and constraints. As a consequence thereof the treatment of explicit and implicit switches is unified.

Formulation (11.11) is ill-posed in a computational setting, as it is numerically difficult to distinguish between the equality and the two inequality cases. For consistently switched systems we can set $\varepsilon \equiv 0$, introduce the tractable constraints $\boldsymbol{\sigma}(\boldsymbol{x}(t)) \geqslant 0$ and $\boldsymbol{\sigma}(\boldsymbol{x}(t)) \leqslant 0$ for the first two modes, respectively, and dispose of the third mode. For inconsistently switched systems, one avenue to handling the arising numerical issues is to introduce an ε-tube for the case $\boldsymbol{\omega}_\nu^0(t) = 1$, as follows:

$$Y_\nu^+(t) \stackrel{\text{def}}{=} \begin{bmatrix} \boldsymbol{\omega}_\nu^+(t) = 1 \\ \dot{\boldsymbol{x}}(t) = \boldsymbol{f}^+(\boldsymbol{x}(t),\boldsymbol{u}(t),\nu) \\ \mathbf{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(\boldsymbol{x}(t),\boldsymbol{u}(t),\nu) \\ \boldsymbol{\sigma}(\boldsymbol{x}(t)) \geqslant +\varepsilon \end{bmatrix}, \quad Y_\nu^-(t) \stackrel{\text{def}}{=} \begin{bmatrix} \boldsymbol{\omega}_\nu^-(t) = 1 \\ \dot{\boldsymbol{x}}(t) = \boldsymbol{f}^-(\boldsymbol{x}(t),\boldsymbol{u}(t),\nu) \\ \mathbf{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(\boldsymbol{x}(t),\boldsymbol{u}(t),\nu) \\ \boldsymbol{\sigma}(\boldsymbol{x}(t)) \leqslant -\varepsilon \end{bmatrix},$$

and

$$Y_\nu^0(t) \stackrel{\text{def}}{=} \begin{bmatrix} \boldsymbol{\omega}_\nu^0(t) = 1 \\ \dot{\boldsymbol{x}}(t) = \boldsymbol{f}^0(\boldsymbol{x}(t),\boldsymbol{u}(t),\nu) \\ \mathbf{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(\boldsymbol{x}(t),\boldsymbol{u}(t),\nu) \\ -\varepsilon \leqslant \boldsymbol{\sigma}(\boldsymbol{x}(t)) \leqslant +\varepsilon \end{bmatrix},$$

$$\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),\boldsymbol{\omega}(\cdot)} \quad \varphi(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})) \tag{11.12a}$$

$$\text{s.t.} \quad \bigoplus_{v\in\Omega}\left[Y_v^-(t)\oplus Y_v^0(t)\oplus Y_v^+(t)\right], \quad t\in\mathcal{T}, \tag{11.12b}$$

$$\mathbf{0}_{n_\mathrm{r}} = \boldsymbol{r}(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})). \tag{11.12c}$$

In some practical problems, function $\boldsymbol{f}^0$ may be a linear combination of $\boldsymbol{f}^+$ and $\boldsymbol{f}^-$, leading to redundancies in the constraint set formulation. Moreover, such a linear combination may be state dependent and of Filippov type, and it will be desirable for a computational approach to identify it automatically. To rid the formulation of constraint redundancy, we may then wish to solve the following GDP problem instead of Problem (11.12):

$$Y_v^+(t) \overset{\text{def}}{=} \begin{bmatrix} \boldsymbol{\omega}_v^+(t) = 1 \\ \dot{\boldsymbol{x}}(t) = \boldsymbol{f}^+(\boldsymbol{x}(t),\boldsymbol{u}(t),v) \\ \mathbf{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(\boldsymbol{x}(t),\boldsymbol{u}(t),v) \\ \boldsymbol{\sigma}(\boldsymbol{x}(t)) \geqslant -\varepsilon \end{bmatrix}, \quad Y_v^-(t) \overset{\text{def}}{=} \begin{bmatrix} \boldsymbol{\omega}_v^-(t) = 1 \\ \dot{\boldsymbol{x}}(t) = \boldsymbol{f}^-(\boldsymbol{x}(t),\boldsymbol{u}(t),v) \\ \mathbf{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(\boldsymbol{x}(t),\boldsymbol{u}(t),v) \\ \boldsymbol{\sigma}(\boldsymbol{x}(t)) \leqslant +\varepsilon \end{bmatrix},$$

$$\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),\boldsymbol{\omega}(\cdot)} \quad \varphi(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})) \tag{11.13a}$$

$$\text{s.t.} \quad \bigoplus_{v\in\Omega}\left[Y_v^-(t)\oplus Y_v^0(t)\oplus Y_v^+(t)\right], \quad t\in\mathcal{T} \tag{11.13b}$$

$$\mathbf{0}_{n_\mathrm{r}} = \boldsymbol{r}(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})), \tag{11.13c}$$

Here, the ε-tubes of either mode overlap, and both modes are feasible on subarcs showing inconsistent switching behavior. We return to the advantages of this formulation when discussing a particular relaxation in the next section.

11.5 Mixed Integer Optimal Control Problems

In this section, we apply a technique of Kirches et al. [275], Jung [259], and Lenders [292] that makes use of MPVCs to obtain a constraint formulation for the disjunction in the GDP Problems (11.10), (11.11), (11.12) and (11.13) that is amenable to the direct approach to optimal control. Problem (11.11) can be equivalently reformulated as a MIOCP

$$\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),\boldsymbol{\omega}(\cdot)} \quad \varphi(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})) \tag{11.14a}$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}}(t) = \sum_{\substack{v\in\Omega,\\ w\in\{-,0,+\}}} \boldsymbol{\omega}_v^w(t)\cdot\boldsymbol{f}^w(\boldsymbol{x}(t),\boldsymbol{u}(t),v), \qquad t\in\mathcal{T}, \tag{11.14b}$$

$$\mathbf{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(\boldsymbol{x}(t),\boldsymbol{u}(t),\boldsymbol{v}(t)), \qquad t\in\mathcal{T}, \tag{11.14c}$$

$$\mathbf{0}_{n_\mathrm{r}} = \boldsymbol{r}(\boldsymbol{x}(t_\mathrm{s}),\boldsymbol{x}(t_\mathrm{f})), \tag{11.14d}$$

$$0 \geqslant -\boldsymbol{\omega}_\nu^+(t) \cdot (\boldsymbol{\sigma}(\boldsymbol{x}(t)) - \varepsilon), \qquad \nu \in \Omega, \quad t \in \mathcal{T}, \tag{11.14e}$$
$$0 \geqslant +\boldsymbol{\omega}_\nu^-(t) \cdot (\boldsymbol{\sigma}(\boldsymbol{x}(t)) + \varepsilon), \qquad \nu \in \Omega, \quad t \in \mathcal{T}, \tag{11.14f}$$
$$0 \geqslant +\boldsymbol{\omega}_\nu^0(t) \cdot (\boldsymbol{\sigma}(\boldsymbol{x}(t)) - \varepsilon), \qquad \nu \in \Omega, \quad t \in \mathcal{T}, \tag{11.14g}$$
$$0 \geqslant -\boldsymbol{\omega}_\nu^0(t) \cdot (\boldsymbol{\sigma}(\boldsymbol{x}(t)) + \varepsilon), \qquad \nu \in \Omega, \quad t \in \mathcal{T}, \tag{11.14h}$$
$$\boldsymbol{\omega}(t) \in \mathbb{S}^{3^{n_\omega}}. \tag{11.14i}$$

To guarantee that exactly one term in the disjunction is active, we define the set

$$\mathbb{S}^{3^{n_\omega}} \stackrel{\text{def}}{=} \left\{ \omega_\nu^w \in \{0,1\},\ \nu \in \Omega, w \in \{-,0,+\} \ : \ \sum\nolimits_{\nu \in \Omega, w \in \{-,0,+\}} \omega_\nu^w = 1 \right\}$$

of all vectors $\omega \stackrel{\text{def}}{=} [\omega_1^-, \omega_1^0, \omega_1^+, \ldots, \omega_{n_\omega}^-, \omega_{n_\omega}^0, \omega_{n_\omega}^+]^T$ with the SOS–1 property. We obtain a relaxed counterpart problem for (11.14) by relaxing the binary constraint $\boldsymbol{\omega}(t) \in \mathbb{S}^{3^{n_\omega}}$ to the convex hull $\boldsymbol{\alpha}(t) \in \operatorname{conv}(\mathbb{S}^{3^{n_\omega}})$:

$$\min_{\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot), \boldsymbol{\alpha}(\cdot)} \quad \boldsymbol{\varphi}(\boldsymbol{x}(t_\mathrm{s}), \boldsymbol{x}(t_\mathrm{f})) \tag{11.15a}$$
$$\text{s.t.} \quad \dot{\boldsymbol{x}}(t) = \sum_{\substack{\nu \in \Omega, \\ w \in \{-,0,+\}}} \boldsymbol{\alpha}_\nu^w(t) \cdot \boldsymbol{f}^w(\boldsymbol{x}(t), \boldsymbol{u}(t), \nu), \qquad t \in \mathcal{T}, \tag{11.15b}$$
$$\mathbf{0}_{n_\mathrm{c}} \geqslant \boldsymbol{c}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{v}(t)), \qquad t \in \mathcal{T}, \tag{11.15c}$$
$$\mathbf{0}_{n_\mathrm{r}} = \boldsymbol{r}(\boldsymbol{x}(t_\mathrm{s}), \boldsymbol{x}(t_\mathrm{f})), \tag{11.15d}$$
$$0 \geqslant -\boldsymbol{\alpha}_\nu^+(t) \cdot (\boldsymbol{\sigma}(\boldsymbol{x}(t)) - \varepsilon), \qquad \nu \in \Omega, \quad t \in \mathcal{T}, \tag{11.15e}$$
$$0 \geqslant +\boldsymbol{\alpha}_\nu^-(t) \cdot (\boldsymbol{\sigma}(\boldsymbol{x}(t)) + \varepsilon), \qquad \nu \in \Omega, \quad t \in \mathcal{T}, \tag{11.15f}$$
$$0 \geqslant +\boldsymbol{\alpha}_\nu^0(t) \cdot (\boldsymbol{\sigma}(\boldsymbol{x}(t)) - \varepsilon), \qquad \nu \in \Omega, \quad t \in \mathcal{T}, \tag{11.15g}$$
$$0 \geqslant -\boldsymbol{\alpha}_\nu^0(t) \cdot (\boldsymbol{\sigma}(\boldsymbol{x}(t)) + \varepsilon), \qquad \nu \in \Omega, \quad t \in \mathcal{T}, \tag{11.15h}$$
$$\boldsymbol{\alpha}(t) \in \operatorname{conv}(\mathbb{S}^{3^{n_\omega}}). \tag{11.15i}$$

Problem (11.15) is an OCP with *vanishing constraints*. The terminology is due to the fact that the implied constraint $\boldsymbol{\sigma}(\boldsymbol{x}(t)) \geqslant \varepsilon$ in (11.15e) *vanishes* as soon as $\boldsymbol{\alpha}_+(t) = 0$. Similar arguments apply to constraints (11.15f)–11.15h.

Problem (11.15) has a larger feasible set than the original MIOCP (11.14) that we actually aim to solve. Hence, lower optimal objective function values may be attained for Problem (11.15), and the relation of both objective function values is of interest. An answer to this question was already given in Theorem 1.35 which justifies the relaxation step. When applied to our specific MIOCP (11.15) we can show the following result:

Theorem 11.1
Let $\hat{\boldsymbol{x}} : \mathcal{T} \longrightarrow \mathbb{R}^{n_\mathrm{x}}$, $\hat{\boldsymbol{u}} : \mathcal{T} \longrightarrow \mathbb{R}^{n_\mathrm{u}}$ and $\hat{\boldsymbol{\alpha}} : \mathcal{T} \longrightarrow \mathbb{R}^{3^{n_\omega}}$ be feasible for Problem (11.15). Then, for every $\delta > 0$ there is $\boldsymbol{x}^\delta : \mathcal{T} \longrightarrow \mathbb{R}^{n_\mathrm{x}}$ and $\boldsymbol{\omega}^\delta : \mathcal{T} \longrightarrow \mathbb{R}^{3^{n_\omega}}$ such that

$$\left| \boldsymbol{\varphi}(\boldsymbol{x}^\delta(t_\mathrm{s}), \boldsymbol{x}^\delta(t_\mathrm{f})) - \boldsymbol{\varphi}(\hat{\boldsymbol{x}}(t_\mathrm{s}), \hat{\boldsymbol{x}}(t_\mathrm{f})) \right| < \delta$$

and

$$
\begin{aligned}
&\dot{\boldsymbol{x}}^{\delta}(t) = \boldsymbol{f}(\boldsymbol{x}^{\delta}(t), \hat{\boldsymbol{u}}(t), \boldsymbol{\omega}^{\delta}(t)), && && t \in \mathcal{T},\\
&\delta L_{\mathrm{c}} \mathbf{1} \geqslant \boldsymbol{c}(\boldsymbol{x}^{\delta}(t), \hat{\boldsymbol{u}}(t)), && && t \in \mathcal{T},\\
&\delta L_{\mathrm{r}} \mathbf{1} = \boldsymbol{r}(\boldsymbol{x}^{\delta}(t_{\mathrm{s}}), \boldsymbol{x}^{\delta}(t_{\mathrm{f}})), &&\\
&\delta L_{\sigma} \geqslant -(\boldsymbol{\omega}^{\delta})_{\nu}^{+}(t)(\boldsymbol{\sigma}(\boldsymbol{x}(t)) - \varepsilon), && \nu \in \Omega, && t \in \mathcal{T},\\
&\delta L_{\sigma} \geqslant +(\boldsymbol{\omega}^{\delta})_{\nu}^{-}(t)(\boldsymbol{\sigma}(\boldsymbol{x}(t)) + \varepsilon), && \nu \in \Omega, && t \in \mathcal{T},\\
&\delta L_{\sigma} \geqslant +(\boldsymbol{\omega}^{\delta})_{\nu}^{0}(t)(\boldsymbol{\sigma}(\boldsymbol{x}(t)) - \varepsilon), && \nu \in \Omega, && t \in \mathcal{T},\\
&\delta L_{\sigma} \geqslant -(\boldsymbol{\omega}^{\delta})_{\nu}^{0}(t)(\boldsymbol{\sigma}(\boldsymbol{x}(t)) + \varepsilon), && \nu \in \Omega, && t \in \mathcal{T},\\
&\boldsymbol{\omega}^{\delta}(t) \in \mathbb{S}^{3^{n_\omega}}, && && t \in \mathcal{T},
\end{aligned}
$$

where L_{c}, L_{r}, and L_{σ} are δ-independent Lipschitz constants of the corresponding functions with respect to the state $\boldsymbol{x}$.

That is, $(\boldsymbol{x}^{\delta}, \boldsymbol{\omega}^{\delta})$ is feasible for Problem (11.14) with the exception of the (vanishing) constraints, which are violated by less than δ times a constant. △

According to Theorem 11.1, every feasible point $\boldsymbol{\alpha}$ of the relaxed MIOCP (11.15) can be approximated arbitrarily well by a binary feasible point $\boldsymbol{\omega}$. In particular, this approximation result also applies to optimal solutions of the relaxed counterpart problem. Note that the binary feasible point $(\boldsymbol{x}^{\delta}, \boldsymbol{\omega}^{\delta})$ obtained by Theorem 11.1 in general depends on the chosen tolerance $\delta > 0$. A constructive algorithm for retrieving binary feasible controls $\boldsymbol{\omega}^{\delta}$ with guaranteed approximation properties is given by the VC–SOS–SUR algorithm (see Section 1.5).

11.6 Discretization

For solving the infinite dimensional OCP (11.15) we apply a tailored pseudospectral collocation method which is also covered by our multi–degree pseudospectral method, cf. Chapter 7. For the reader's convenience, we describe the single discretization steps.

We split up the horizon $\mathcal{T} = [t_{\mathrm{s}}, t_{\mathrm{f}}]$ into $N \in \mathbb{N}$ finite elements by choosing a time grid

$$t_{\mathrm{s}} = t_0 < t_1 < \cdots < t_N = t_{\mathrm{f}}.$$

For each finite element we choose Lagrange basis polynomials $\left\{\boldsymbol{L}_j^{(n)}\right\}_{j=0}^{K_n}$ and $\left\{\overline{\boldsymbol{L}}_j^{(n)}\right\}_{j=1}^{\overline{K}_n}$ given by

$$\boldsymbol{L}_j^{(n)}(t) \stackrel{\text{def}}{=} \prod_{\substack{i=0\\ i\neq j}}^{K_n} \frac{t - t_i^{(n)}}{t_j^{(n)} - t_i^{(n)}}, \quad \overline{\boldsymbol{L}}_j^{(n)}(t) \stackrel{\text{def}}{=} \prod_{\substack{i=1\\ i\neq j}}^{\overline{K}_n} \frac{t - \overline{t}_i^{(n)}}{\overline{t}_j^{(n)} - \overline{t}_i^{(n)}}, \quad n \in [N].$$

We use FLGR points for the $t_i^{(n)}$, $\overline{t}_j^{(n)} \in \mathbb{R}$ ($i = 0, \ldots, K_n$, $j \in [\overline{K}_n]$, $n \in [N]$). The affine

transformations

$$t^{(n)}(\tau) \stackrel{\text{def}}{=} \frac{t_n + t_{n-1}}{2} + \tau \, \frac{t_n - t_{n-1}}{2}, \quad n \in [N],$$

map FLGR points to the finite element intervals $\mathcal{T}_n \stackrel{\text{def}}{=} [t_{n-1}, t_n]$ and yield the collocation points $t_i^{(n)}, \overline{t}_j^{(n)}$. In addition we set $t_0^{(n)} = t_{n-1}$. The differential states are approximated element–wise as

$$\boldsymbol{X}(t) \stackrel{\text{def}}{=} \sum_{j=0}^{K_n} x_j^{(n)} \boldsymbol{L}_j^{(n)}(t), \quad t \in \mathcal{T}_n, \quad n \in [N],$$

where K_n is the number of collocation points and $x_j^{(n)} \in \mathbb{R}^{n_x}$ the nodal values. The derivative with respect to the time of the differential state approximation is given by

$$\dot{\boldsymbol{X}}(t) = \sum_{j=0}^{K_n} x_j^{(n)} \dot{\boldsymbol{L}}_j^{(n)}(t), \quad t \in \mathcal{T}_n, \quad n \in [N].$$

Analogously to the state approximations the controls $\boldsymbol{u}$ are approximated by

$$\boldsymbol{U}(t) \stackrel{\text{def}}{=} \sum_{j=1}^{\overline{K}_n} u_j^{(n)} \overline{\boldsymbol{L}}_j^{(n)}(t), \quad t \in \mathcal{T}_n, \quad n \in [N].$$

Here we have the nodal values $u_j^{(n)} \in \mathbb{R}^{n_u}$. We just consider the case with no explicit switches ($n_\omega = 1$) in order to avoid notational clutter. The extension to the case with explicit and implicit switches is straight-forward. The controls $\boldsymbol{\alpha}_+, \boldsymbol{\alpha}_0, \boldsymbol{\alpha}_-$ are approximated by piecewise constant functions

$$\hat{\boldsymbol{\alpha}}(t) \stackrel{\text{def}}{=} \alpha^{(n)} = \left[\alpha_+^{(n)}, \alpha_0^{(n)}, \alpha_-^{(n)}\right]^T, \quad t \in \mathcal{T}_n, \quad n \in [N]. \tag{11.16}$$

To end up with an NLP we discretize the Mayer type objective as $\boldsymbol{\varphi}(x_{K_N}^{(N)})$ and the differential equations by means of finite element wise collocation

$$\begin{aligned} 0 &= \dot{\boldsymbol{X}}\left(t_i^{(n)}\right) - \boldsymbol{f}\left(\boldsymbol{X}\left(t_i^{(n)}\right), \boldsymbol{U}\left(t_i^{(n)}\right), \hat{\boldsymbol{\alpha}}\left(t_i^{(n)}\right)\right), \quad i \in [K_n], \quad n \in [N], \\ \Leftrightarrow \quad 0 &= \sum_{j=0}^{K_n} x_j^{(n)} \dot{\boldsymbol{L}}_j^{(n)}\left(t_i^{(n)}\right) - \boldsymbol{f}\left(x_i^{(n)}, \boldsymbol{U}\left(t_i^{(n)}\right), \alpha^{(n)}\right), \quad i \in [K_n], \quad n \in [N]. \end{aligned}$$

We augment the system with matching conditions to enforce continuity of the differential states:

$$x_{K_n}^{(n)} = x_0^{(n+1)}, \quad n \in [N-1].$$

Alternatively one could simply identify the variables $x_{K_n}^{(n)}$ and $x_0^{(n+1)}$. But this would couple the variables over the finite element boundaries and this is usually not desired. The discretization of the boundary constraints leads to the NLP constraints

$$0 \geqslant \boldsymbol{r}\left(x_0^{(1)}, x_{K_N}^{(N)}\right).$$

Path constraints are enforced to hold at collocation points and vanishing constraints are enforced to hold at finite element grid points

$$\begin{aligned}
&0 \geqslant \boldsymbol{c}(x_i^{(n)}, \boldsymbol{U}(t_i^{(n)})), && i \in [K_n], \quad n \in [N], \\
&0 \geqslant -\alpha_+^{(n)}(\boldsymbol{\sigma}(x_0^{(n)}) - \varepsilon), && n \in [N], && \text{(11.17a)} \\
&0 \geqslant +\alpha_-^{(n)}(\boldsymbol{\sigma}(x_0^{(n)}) + \varepsilon), && n \in [N], && \text{(11.17b)} \\
&0 \geqslant +\alpha_0^{(n)}(\boldsymbol{\sigma}(x_0^{(n)}) - \varepsilon), && n \in [N], && \text{(11.17c)} \\
&0 \geqslant -\alpha_0^{(n)}(\boldsymbol{\sigma}(x_0^{(n)}) + \varepsilon), && n \in [N]. && \text{(11.17d)}
\end{aligned}$$

The relaxed SOS–1 constraint $\boldsymbol{\alpha}(t) \in \operatorname{conv}(\mathbb{S}^3)$ leads to the NLP constraints

$$\sum_{j \in \{-,0,+\}} \alpha_j^{(n)} = 1, \quad \alpha_i^{(n)} \in [0, +\infty), \quad i \in \{-,0,+\}, \quad n \in [N].$$

11.7 MPVC Handling

A common approach for solving MPVCs (see Section 4.4) using standard nonlinear programming software originated in the field of MPECs by Scholtes [391] and is in particular advocated for MPVCs by Hoheisel [238, Chapter 10]. The approach pursues the idea of embedding (4.2) into a family of perturbed problems parameterized by a scalar perturbation $\tau > 0$. Problem (4.2) may be embedded into the problem family

$$\begin{aligned}
\min_{x \in \mathbb{R}^n} \quad & \varphi(x) && && \text{(11.18)} \\
\text{s.t.} \quad & 0 = \boldsymbol{s}_i(x), && i \in \mathcal{E}, \\
& 0 \geqslant \boldsymbol{s}_i(x), && i \in \mathcal{I}, \\
& \tau \geqslant \boldsymbol{g}_i(x) \cdot \boldsymbol{h}_i(x), && i \in [n_{\mathrm{s}}] \\
& \boldsymbol{0} \geqslant \boldsymbol{h}(x), &&
\end{aligned}$$

where $\tau > 0$ is the regularization parameter. The feasible set of (11.18) relaxes the one of (4.2). For $\tau \to 0^+$, the feasible set approaches the one of (4.2). Other relaxation formulations are possible (see Section 4.4) but have not performed better than (11.18) on the numerical examples provided in Chapter 15.

11.8 Sequential Nonlinear Programming

In this section we describe in detail how we apply the method from the previous section to the MPVCs arising from the OCP discretization described in Section 11.6. Our numerical experiments have shown that it is important to couple the discretization accuracy with the homotopy parameter to avoid infeasible NLPs along the homotopy path.
We denote the value of the relaxation parameter in the k-th iteration by τ_k. Starting with an initial assignment for the relaxation parameter τ_0 we solve a sequence of NLPs. The relaxation parameter is driven to zero according to the rule

$$\tau_k = \gamma\,\tau_{k-1}, \quad k \geqslant 1, \quad \gamma \in (0,1). \tag{11.19}$$

The NLP solver is initialized in iteration k with the NLP solution of iteration $k-1$. If the NLP in iteration k is infeasible we refine the finite element grid adaptively and solve the NLP again with the current relaxation parameter τ_k. In case the new NLP is feasible we continue applying rule (11.19). Otherwise the finite element grid is refined again. After a refinement step we initialize the NLP variables as follows: we take the NLP solver result from the previous iteration even if the NLP was infeasible. Compared to other strategies this was the one that worked the best in our experiments. Then we initialize all states and controls on the refined grid by interpolation.
Due to lack of a priori knowledge about the switching structure we usually start with an equidistant finite element grid. If NLP infeasibilities arise it is often because a switching point is not well resolved by the grid points. Hence our refinement algorithm has to take this into account.
We propose the following simple heuristic to refine the grid in order to resolve switching points: To this end, we construct a cubic spline interpolant $\boldsymbol{s}(t)$ of each component of $\boldsymbol{\alpha}(t)$ in the element interface nodes $t_j, j = 0,\ldots,N$. We adaptively bisect a grid cell $[t_i, t_{i+1}]$ into two equally sized cells if either

$$\left|\boldsymbol{s}'\left(\frac{t_{i-1}+t_i}{2}\right)\right| > \frac{1}{4}\max_{j\in[N]}\left\{\left|\boldsymbol{s}'\left(\frac{t_{j-1}+t_j}{2}\right)\right|\right\} \tag{11.20}$$

$$\text{or} \quad |\boldsymbol{s}''(t_i)| > \frac{1}{4}\max_{j\in[N]}\{|\boldsymbol{s}''(t_j)|\}. \tag{11.21}$$

This heuristic detects high slopes of $\boldsymbol{\alpha}$ through (11.20), which indicates that a switch should happen at some place within the interval, and high curvature of $\boldsymbol{\alpha}$ through (11.21), which indicates that a Filippov arc should begin or end at some place within the interval.
Note that we implemented the full SNLP approach within our software package `grc`. The software was used to make the numerical experiments in Chapter 15.

Part III

Applications and Numerical Results

Chapter 12

Multi–Degree Pseudospectral Collocation Numerics

In this chapter, we present numerical results on the multi–degree pseudospectral collocation method that was introduced in Chapter 7. To this end we developed a software package in `Matlab` implementing the method. The software employs several thirdparty software packages. Models are implemented with the help of the ODE/DAE solver suite `SolvIND` [9]. Evaluations of functions and respective derivative information are provided by the automatic differentiation software tool ADOL-C [445] via an interface of `SolvIND`. Arising NLP instances can either be solved by the interior–point solver `Ipopt` [444] in version 3.12.8 or by the SQP solver SNOPT [197] in version 7.2-7.

We showed in Chapter 7 that a collocation approach with a uniform number of collocation points results in well–structured and sparse NLP constraint Jacobians, cf. Figure 7.1. In contrast, sparsity might get lost when using multiple collocation point numbers, cf. Figure 7.4. We aim to show that there exist cases where it is favorable to give up some structure but obtaining faster results while maintaining the approximation quality. The two academic examples studied in this chapter show possible application scenarios. However, more research has to be done in the future.

The first problem, which is investigated in Section 12.1, is an academic ODE example with known analytic solution. All four state components of this example are oscillating. While the oscillation of two components has a low frequency the frequency of the other two components is high. We observe in our numerical experiments that there is a lower bound for the number of collocation points such that the oscillations are approximated sufficiently well. Moreover, the higher the frequency is the more collocation points are required. We find out in our experiments that it is faster to use an adaptive number of collocation points for each component depending on its oscillation frequency, rather than to use the same high–resolution grid for all components that is necessary to resolve the component with the fastest oscillations.

Section 12.2 deals with an OCP which is related with the ODE problem from Section 12.1. It has two controls and a tracking objective functional where one control component tracks a state having a low frequency oscillation while the other control component tracks a state with a high frequency oscillation. Our experiments indicate – similar to the ODE case – lower solution times if we use the "one–size–fits–all" approach instead of suitable collocation points numbers.

12.1 An Academic ODE Example

We consider the following academic feasibility problem:

$$
\begin{aligned}
\text{find} \quad & \boldsymbol{x}(\cdot) = [\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4]^T(\cdot) \\
\text{s.t.} \quad & \dot{\boldsymbol{x}}_1(t) = +\boldsymbol{x}_2(t), \\
& \dot{\boldsymbol{x}}_2(t) = -\boldsymbol{x}_1(t), \\
& \dot{\boldsymbol{x}}_3(t) = +\lambda \cdot \boldsymbol{x}_4(t), \\
& \dot{\boldsymbol{x}}_4(t) = -\lambda \cdot \boldsymbol{x}_3(t), \\
& \boldsymbol{x}(0) = [0, 1, 0, 1]^T, \ \lambda = 10.
\end{aligned}
\tag{12.1}
$$

The solution to this ODE problem is given as

$$\boldsymbol{x}_1(t) = \sin(t), \qquad \boldsymbol{x}_3(t) = \sin(10\,t), \tag{12.2}$$

$$\boldsymbol{x}_2(t) = \cos(t), \qquad \boldsymbol{x}_4(t) = \cos(10\,t). \tag{12.3}$$

The example was implemented in our OCP software `grc` employing a LGR collocation method as it was introduced in Chapter 7. The arising NLP instances were solved using the NLP solver `Ipopt`. The Hessian of the Lagrangian was approximated by a limited–memory quasi–Newton method (L–BFGS updates) and the (relative) convergence tolerance was chosen to be equal to 1×10^{-10}. The initial guess for nodal values at collocation points was chosen to be equal to $[0, 1, 0, 1]^T$, i.e., it coincides with $\boldsymbol{x}(0)$.

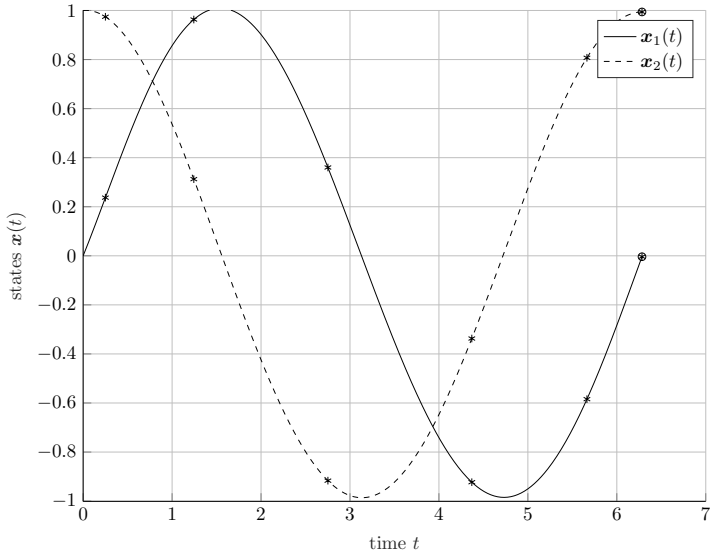


Figure 12.1: The figure depicts the state approximations $\boldsymbol{x}_1(\cdot)$ and $\boldsymbol{x}_2(\cdot)$ for Problem (12.1) on the interval $[0, 2\,\pi]$. The number of finite elements and number of collocation points were chosen to be equal to one and six, respectively. The approximations are in accordance with the exact solutions given by $\sin(t)$ and $\cos(t)$.

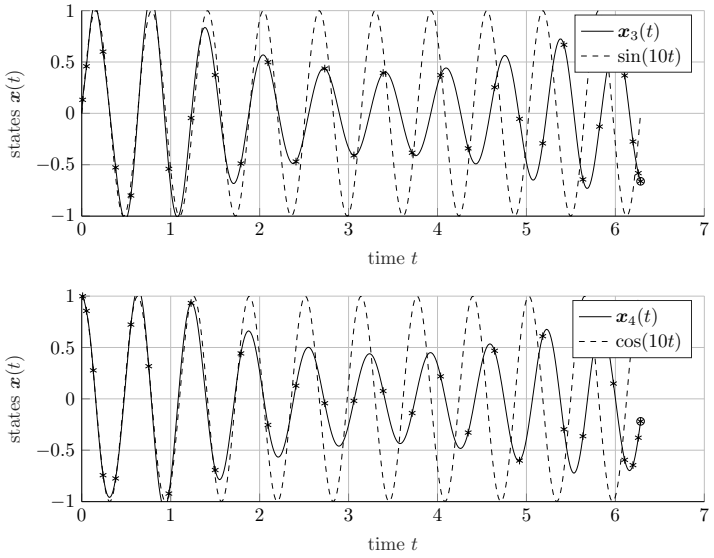
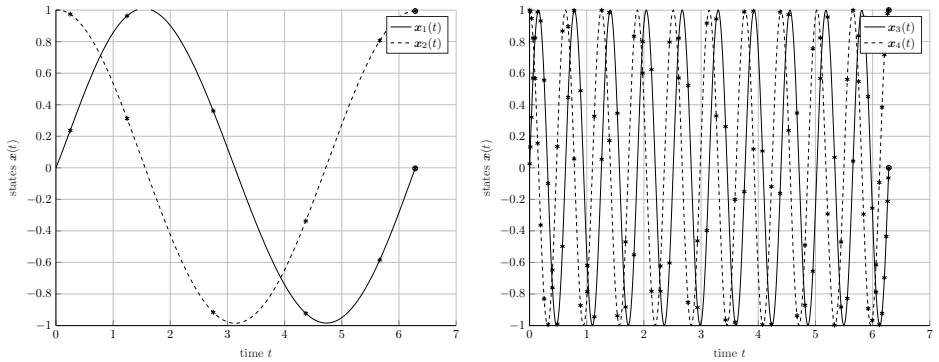


Figure 12.2: The figure shows the state approximations $x_3(t)$ and $x_4(t)$ for Problem 12.1 with one finite element and 30 collocation points on the interval $[0, 2\pi]$. Furthermore, the analytic solutions $\sin(10\,t)$ and $\cos(10\,t)$ are depicted in order to highlight the poor accordance between the exact and the approximated trajectories.

We solved Problem 12.1 with different discretization configurations. The first configuration under consideration solves the feasibility problem with one finite element and six collocation points for the components $x_1(\cdot)$ and $x_2(\cdot)$. The number of collocation points for components $x_3(\cdot)$ and $x_4(\cdot)$ were chosen to be equal to 30. Figures 12.1 and 12.2 show the associated trajectories. While components $x_1(\cdot)$ and $x_2(\cdot)$ approximate the real solutions $\sin(t)$ and $\cos(t)$ rather well, components $x_3(\cdot)$ and $x_4(\cdot)$ are just poorly approximated. It took just one iteration to solve the discretization NLP for the prescribed convergence tolerance. The total number of NLP variables was 78 and the number of nonzeros in the constraint Jacobian was 3750.

In order to overcome the poor approximation properties for the components $x_3(\cdot)$ and $x_4(\cdot)$ we used 60 collocation points for those components in our second configuration. The number of collocation points was left unchanged for the components $x_1(\cdot)$ and $x_2(\cdot)$. Likewise, we used one finite element. The resulting trajectories can be found in Figure 12.3. One can easily see that the analytic solutions $\sin(10\,t)$ and $\cos(10\,t)$ for components $x_3(\cdot)$ and $x_4(\cdot)$ are approximated quite well now. As expected, the trajectories for components $x_1(\cdot)$ and $x_2(\cdot)$ are not affected compared to the first configuration. The convergence tolerance was achieved within one iteration again. The number of NLP variables and nonzeros in the constraint Jacobian increased to 138 and 10950, respectively.

In order to stress the benefits of our software over others we also considered the case where all components are solved with the same number of collocation points. Obviously, we had to choose 60 collocation points in order to be able to fulfill the approximation quality in the last two components. Even though the resulting NLP has a nicer structure in terms of its constraint



a) States $\boldsymbol{x}_1(t)$ and $\boldsymbol{x}_2(t)$. b) States $\boldsymbol{x}_3(t)$ and $\boldsymbol{x}_4(t)$.

Figure 12.3: The figure depicts the state approximations of Problem 12.1 where the discretization was chosen such that there is one finite element, six collocation points for the first two components as well as 60 collocation points for components three and four. The approximations agree well with the analytic solutions from (12.2)+(12.3).

Jacobian it has 246 NLP variables and 15846 nonzeros in the constraint Jacobian. The NLP solver converges within one iteration. However, there is a speedup between the second and the third configuration of 45 %.

12.2 An Academic OCP Example

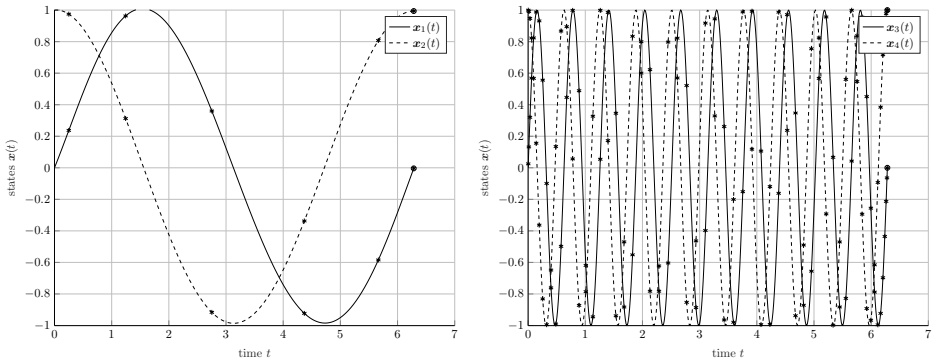
We consider the following OCP:

$$\begin{aligned}
\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)} \quad & \int_0^{2\pi} (\boldsymbol{x}_1(t)-\boldsymbol{u}_1(t))^2 + (\boldsymbol{x}_3(t)-\boldsymbol{u}_2(t))^2 \, \mathrm{d}t \\
\text{s.t.} \quad & \dot{\boldsymbol{x}}_1(t) = +\boldsymbol{x}_2(t), \\
& \dot{\boldsymbol{x}}_2(t) = -\boldsymbol{x}_1(t), \\
& \dot{\boldsymbol{x}}_3(t) = +\lambda \cdot \boldsymbol{x}_4(t), \\
& \dot{\boldsymbol{x}}_4(t) = -\lambda \cdot \boldsymbol{x}_3(t), \\
& \boldsymbol{x}(0) = [0,1,0,1]^T, \ \lambda = 10.
\end{aligned} \tag{12.4}$$

It is obvious that this OCP is strongly related with Problem 12.1. Consequently, the analytic solutions for the differential states $\boldsymbol{x}(\cdot) = [\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4]^T(\cdot)$ coincide with the ones given in (12.2)+(12.3). The optimal control $\boldsymbol{u}(\cdot) = [\boldsymbol{u}_1, \boldsymbol{u}_2](\cdot)^T$ is given as follows:

$$\boldsymbol{u}_1(t) = \sin(t), \qquad \boldsymbol{u}_2(t) = \sin(10\,t). \tag{12.5}$$

The implementation of OCP 12.4 was done by means of `grc`. As opposed to Section 12.1, we



a) Differential states $\boldsymbol{x}_1(t)$ and $\boldsymbol{x}_2(t)$.

b) Differential states $\boldsymbol{x}_3(t)$ and $\boldsymbol{x}_4(t)$.

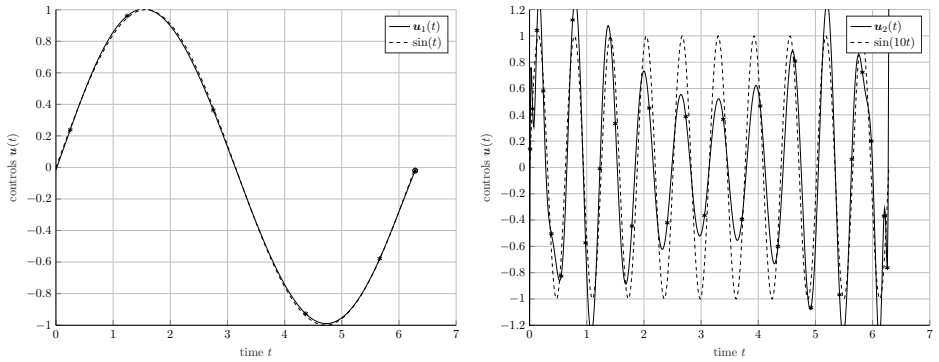
Figure 12.4: The figure depicts the differential states of OCP 12.4. The problem is discretized with one finite element, six collocation points for state components $\boldsymbol{x}_1(t)$ and $\boldsymbol{x}_2(t)$, and 60 collocation points for components $\boldsymbol{x}_3(t)$ and $\boldsymbol{x}_4(t)$.

used SNOPT with an optimality and feasibility convergence tolerance of 1×10^{-6}. The initial guess at collocation points was chosen as in Section 12.1 for the differential states and $[0, 0]^T$ for the controls.

The discretization of the differential states is based on the numerical experiments in the previous section, i.e., we chose six collocation points for components $\boldsymbol{x}_1(\cdot)$ and $\boldsymbol{x}_2(\cdot)$ and 60 collocation points for components $\boldsymbol{x}_3(\cdot)$ and $\boldsymbol{x}_4(\cdot)$. Choosing six collocation points for control component $\boldsymbol{u}_1(\cdot)$ and 30 collocation points for $\boldsymbol{u}_2(\cdot)$ results in a NLP with 235 variables and 23393 nonzeros in the constraint Jacobian. It takes 54 iterations until convergence. The state approximations are depicted in Figure 12.4 and the control approximations in Figure 12.5. Expectedly, Figure 12.4 and Figure 12.3 apparently coincide. While $\boldsymbol{u}_1(\cdot)$ shows a good tracking behavior of state $\boldsymbol{x}_1(\cdot)$ this does not hold for control $\boldsymbol{u}_2(\cdot)$ and state $\boldsymbol{x}_3(\cdot)$.

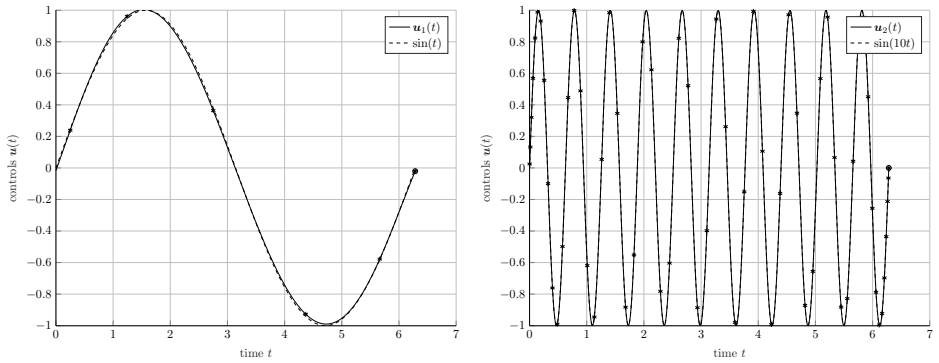
In order to improve the quality of the tracking we changed the number of collocation points for $\boldsymbol{u}_2(\cdot)$ to 60 in our second experiment. The state solution trajectories remain unchanged compared to the first experiment. The control solution trajectories can be found in Figure 12.6. Now $\boldsymbol{u}_2(\cdot)$ tracks $\boldsymbol{x}_3(\cdot)$ quite well. The associated NLP takes 41 iterations until convergence. It has 256 variables but just 18533 nonzeros in the constraint Jacobian. This is due to the improved structure compared to the first experiment.

In order to compare the previous results with common collocation implementations we also considered the case with 60 collocation points for all state and control components. With 50 iterations it takes slightly more iterations to converge than in the previous experiment. The arising NLP also comprises more variables and nonzeros in the constraint Jacobian, namely 427 and 20705. This is reflected in the speedup of 49 % of the second compared to the third experiment.



a) Control $u_1(t)$ and exact solution $\sin(t)$. b) Control $u_2(t)$ and exact solution $\sin(10\,t)$.

Figure 12.5: The figure contains control trajectories (solid lines) for OCP 12.4 where the discretization was chosen such that there is one finite element, six collocation points for $u_1(\cdot)$ and 30 collocation points for $u_2(\cdot)$. Dashed lines show the associated analytic solutions from (12.5).



a) Control $u_1(t)$ and exact solution $\sin(t)$. b) Control $u_2(t)$ and exact solution $\sin(10\,t)$.

Figure 12.6: The figure depicts control trajectories (solid lines) for OCP 12.4 with a discretization having six collocation points for $u_1(\cdot)$ and 60 for $u_2(\cdot)$. The associated analytic solutions (dashed lines) indicate a good approximation quality for both control components.

Chapter 13

Petrov–Galerkin Costate Estimation Numerics

In this chapter, we illustrate the results of Chapter 9 by reference to several OCP models, i.e., we investigate the correlation of NLP multipliers coming from an OCP discretization and adjoint states of the PMP. The models under consideration have different characteristics. First, we study an example that has no path constraints at all. The second example is purely control constrained while the third example has mixed control–state path constraints. The underlying ODE model for the aforementioned three problems is a variant of the well–known van der Pol oscillator. All variants under consideration were described and analyzed by Maurer [317] who solved them with an indirect approach. Those solutions serve as a reference solution for our analysis. The chapter is concluded with a purely state constrained OCP whose analytic solution is known. We use this example to demonstrate the need for an efficient mesh refinement strategy which will be the topic of the following Chapter 14.

Regarding the practical realization, we rely on our software `grc` which was mentioned earlier in Chapter 12. In order to come up with a costate estimation according to the theory of Chapter 9, we use the NLP multipliers that are provided by the NLP solvers `Ipopt` or SNOPT. According to (9.16) the function $\boldsymbol{\Lambda}(\cdot)$ is given in terms of the adjoint solution $\boldsymbol{\lambda}(\cdot)$. The discretization of $\boldsymbol{\Lambda}(\cdot)$ by means of the function space $\mathcal{Z}_H(\mathcal{T}, \mathbb{R})$ leads to representations $\boldsymbol{\Lambda}_{\mathrm{h}}(\cdot)$ of the form (9.45) where the coefficients are algebraically related to the NLP multipliers. In order to compare the results of our costate estimation with results from other publications we need to come up with adjoint state estimations, i.e., we need to determine derivatives of $\boldsymbol{\Lambda}_{\mathrm{h}}(\cdot)$. As we had pointed out already, the classic derivative of $\boldsymbol{\Lambda}_{\mathrm{h}}(\cdot)$ does not exist but at least it is differentiable in a weak form. We exploit the weak differentiability to find adjoint state estimations. In case of purely state constrained problems there may occur discontinuous differential state costates. However, those discontinuities can be avoided by employing the "indirect adjoining approach with continuous adjoint functions" as described by Hartl et al. [224, Section 7] and investigated by Francolin et al. [173] and Francolin [172] in a LGR collocation environment. Hartl et al. [224, Remark 7.2] show how costates in the PMP and the indirect adjoining approach are interrelated. We use the representation of the indirect adjoining approach in order to be able to apply the error estimation of the mixed control–state constrained case. Costate jumps only appear for constraint costates. They have to be detected for an efficient error estimation. Ideas about practical realizations are described in the following chapter.

In Section 13.1, we introduce the Rayleigh equation, which is a variant of the van der Pol oscillator, in an optimal control context without path constraints. We determine numerical solutions with the aid of `grc` and present the resulting trajectories. In particular, we present costate estimates and compare them with the results found by Maurer [317].

Section 13.2 extends the model of Section 13.1 with additional control constraints. The trajectories provided by `grc` are compared with the ones from Maurer [317].

In Section 13.3, we add mixed control–state constraints to the Rayleigh equation of Section 13.1. We compute state and control trajectories as well as state and constraint costates by means of `grc`. Then we check if they match with the respective trajectories of Maurer [317].

The final Section 13.4 of this chapter deals with a minimum–energy OCP that has a pure state constraint and whose analytic solution is available. We solve the problem with `grc` where we make use of an implemented pure state constraint detection routine. The resulting adjoint trajectories coincide with the ones coming from the aforementioned PMP with continuous differential costates. Based on two different discretization we outline issues that must be addressed by efficient error estimation routines.

13.1 Rayleigh Problem Without Constraints

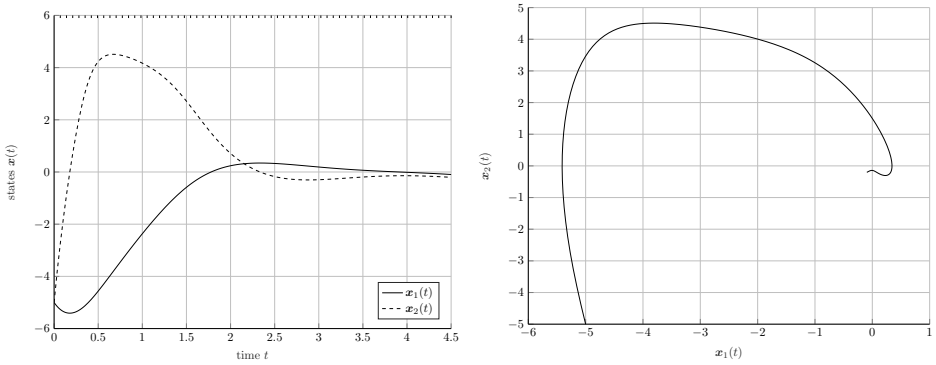
By means of the Rayleigh equation one can model oscillations of the electric current in an electric circuit. Augmenting the Rayleigh equation with a control–quadratic objective results in the following OCP:

$$
\begin{aligned}
\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)} \quad & \int_0^{t_\mathrm{f}} \left(\boldsymbol{x}_1(t)^2 + \boldsymbol{u}(t)^2\right) \mathrm{d}t \\
\text{s.t.} \quad & \dot{\boldsymbol{x}}_1(t) = +\boldsymbol{x}_2(t), \\
& \dot{\boldsymbol{x}}_2(t) = -\boldsymbol{x}_1(t) + \boldsymbol{x}_2(t)\left(1.4 - 0.14 \cdot \boldsymbol{x}_2(t)^2\right) + 4\,\boldsymbol{u}(t), \\
& \boldsymbol{x}(0) = [-5,-5]^T,\ t_\mathrm{f} = 4.5.
\end{aligned}
\tag{13.1}
$$

The electric current at a time instant t in OCP (13.1) is denoted with $\boldsymbol{x}_1(t)$. The voltage at the generator acts as a control function and it enters the system in the form of the scalar control function $\boldsymbol{u}(\cdot)$ after a suitable transformation. The optimal control of a control–quadratic objective subject to the Rayleigh equation has been investigated several times in different variations, cf. e.g. Maurer and Augustin [318], Chen and Gerdts [108], Osmolovskii and Maurer [345], and Maurer and Osmolovskii [319]. The majority of those publications analyze the problem with indirect solution methods. In particular, adjoint information is gained in this way.

We compare our numerical results with the ones of Maurer [317]. We implemented OCP (13.1) in our software `grc`. The problem discretization is chosen as follows: we have an equidistant FE–grid with 64 FEs. The number of collocation points on each FE and for all state and control components is three. As NLP solver we used SNOPT where the optimality and feasibility convergence tolerance was chosen to be 1×10^{-6}.

Figures 13.1 and 13.2 show the resulting trajectories for (adjoint) differential states and controls. All of them are consistent with the respective results of Maurer [317].



a) Differential states b) Phase portrait

Figure 13.1: The figure depicts the differential states $\boldsymbol{x}(\cdot) = [\boldsymbol{x}_1, \boldsymbol{x}_2]^T(\cdot)$ of OCP (13.1) and their phase portrait. The state plot illustrates the 64 equidistant FEs as ticks at the top.

13.2 Rayleigh Problem With Control Bounds

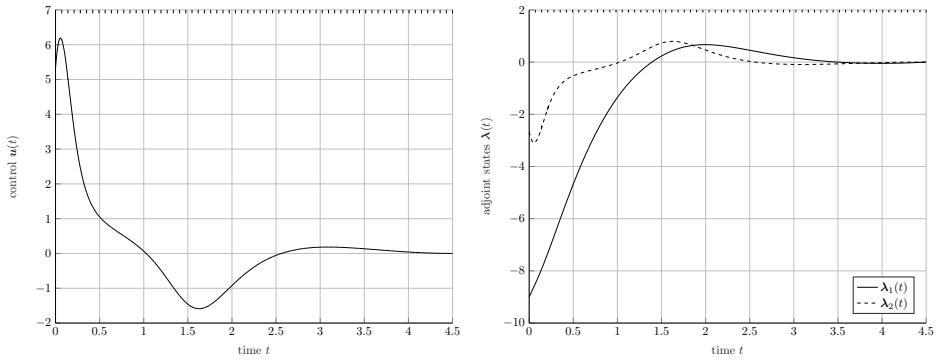
In this section, we augment OCP (13.1) with pure control constraints such that we consider the following problem:

$$
\begin{aligned}
\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)} \quad & \int_0^{t_\mathrm{f}} \left(\boldsymbol{x}_1(t)^2 + \boldsymbol{u}(t)^2\right) \mathrm{d}t \\
\text{s.t.} \quad & \dot{\boldsymbol{x}}_1(t) = +\boldsymbol{x}_2(t), \\
& \dot{\boldsymbol{x}}_2(t) = -\boldsymbol{x}_1(t) + \boldsymbol{x}_2(t)\left(1.4 - 0.14 \cdot \boldsymbol{x}_2(t)^2\right) + 4\boldsymbol{u}(t), \\
& \quad -1 \leqslant \boldsymbol{u}(t) \leqslant +1, \\
& \boldsymbol{x}(0) = [-5, -5]^T,\ t_\mathrm{f} = 4.5.
\end{aligned}
\tag{13.2}
$$

The meanings of the variables remain the same as for OCP (13.1). The implementation was done with `grc` where we used `Ipopt` with exact second–order information to solve the problem after the discretization process. The convergence tolerance for `Ipopt` was chosen to be equal to 1×10^{-10}. The discretization was done such that there are 64 FEs and the unified number of collocation points for states and controls is three.
Figures 13.3 and 13.4 show trajectories for differential states, control and differential costates. If one compares those trajectories with the respective ones of Maurer [317] one can see that they are in accordance with each other.

13.3 Rayleigh Problem With Mixed Control–State Constraint I

In this section, we augment the Rayleigh problem from Section 13.1 with a mixed control–state constraint. The resulting full problem then reads as:



a) Control b) Adjoint differential states

Figure 13.2: The control as well as the adjoint differential states for OCP (13.1) are portrayed in the figure. Furthermore, the 64 FEs of the employed discretization are shown at the top of both plots.

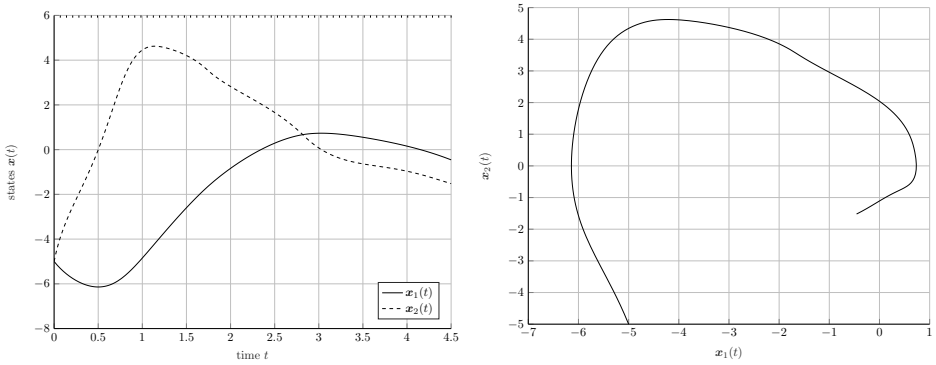
$$\begin{aligned}
\min_{x(\cdot),u(\cdot)} \quad & \int_0^{t_\mathrm{f}} \left(x_1(t)^2 + u(t)^2\right) \mathrm{d}t \\
\text{s.t.} \quad & \dot{x}_1(t) = +x_2(t), \\
& \dot{x}_2(t) = -x_1(t) + x_2(t)\left(1.4 - 0.14 \cdot x_2(t)^2\right) + 4u(t), \\
& -1 \leqslant u(t) + \frac{x_1(t)}{6} \leqslant 0, \\
& x(0) = [-5, -5]^T, \; t_\mathrm{f} = 4.5.
\end{aligned} \tag{13.3}$$

The notation of all variables is the same as in Sections 13.1 and 13.2. We solved OCP (13.3) by means of a discretization having 64 equidistant FEs and three collocation points per state and control component. The solution was computed with `grc` and `Ipopt` as NLP solver. The convergence tolerance for `Ipopt` was chosen to be 1×10^{-10}.
Figures 13.5 and 13.6 show the resulting trajectories for differential states, control, mixed control–state constraint and its costate. The results coincide with the ones of Maurer [317].

13.4 A Minimum Energy Double Integrator

In this section, we consider an energy minimization OCP subject to pure state constraint (see Bryson and Ho [88, p. 120–123]). The full system reads as

$$\min_{x(\cdot),u(\cdot)} \quad \frac{1}{2} \int_0^1 u(t)^2 \, \mathrm{d}t \tag{13.4}$$



a) Differential states b) Phase portrait

Figure 13.3: The figure depicts both differential state trajectories for OCP (13.2) in the left plot and the respective phase portrait in the right plot. The associated discretization uses 64 FEs and three collocation points for states and control. The FEs are illustrated in the left plot by means of ticks at the top.

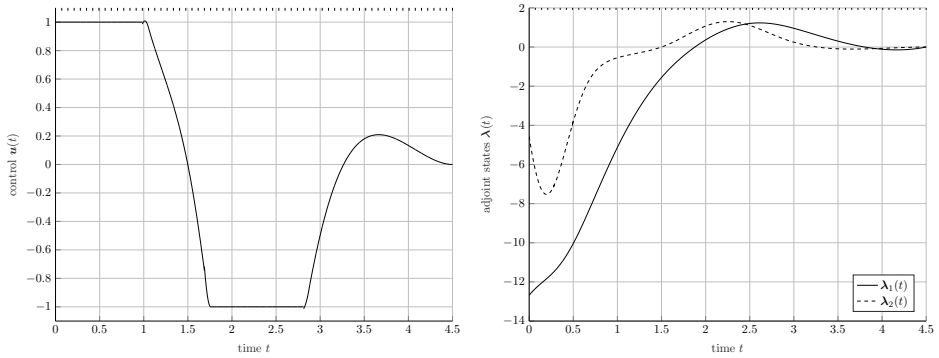
$$
\begin{aligned}
\text{s.t.} \qquad & \dot{\boldsymbol{x}}_1(t) = \boldsymbol{x}_2(t), \\
& \dot{\boldsymbol{x}}_2(t) = \boldsymbol{u}(t), \\
& \quad l \geqslant \boldsymbol{x}_1(t), \\
& \boldsymbol{x}(0) = [0,+1]^T,\ \boldsymbol{x}(1) = [0,-1]^T,\ l = \tfrac{1}{12}.
\end{aligned}
$$

Bryson and Ho [88] point out that OCP (13.4) is a system with a second–order state variable inequality constraint, since the control variable $\boldsymbol{u}(\cdot)$ does not enter the constraint $\boldsymbol{c}(t) \stackrel{\text{def}}{=} \boldsymbol{x}_1(t) - l$ and its derivative $\dot{\boldsymbol{c}}(t) = \boldsymbol{x}_2(t)$ explicitly. In contrast, the second derivative $\ddot{\boldsymbol{c}}(t) = \boldsymbol{u}(t)$ explicitly contains the control variable.

The analytical solution to OCP (13.4) for the differential states are given as

$$
\boldsymbol{x}_1^*(t) = \begin{cases} l\left(1 - \left(1 - \dfrac{t}{3l}\right)^3\right), & t \in [0, 3l], \\ l, & t \in [3l, 1-3l], \\ l\left(1 - \left(1 - \dfrac{1-t}{3l}\right)^3\right), & t \in [1-3l, 1], \end{cases}
$$

$$
\boldsymbol{x}_2^*(t) = \begin{cases} \left(1 - \dfrac{t}{3l}\right)^2, & t \in [0, 3l], \\ 0, & t \in [3l, 1-3l], \\ -\left(1 - \dfrac{1-t}{3l}\right)^2, & t \in [1-3l, 1], \end{cases}
$$



a) Control b) Adjoint differential states

Figure 13.4: The control trajectory for OCP (13.2) is shown in the left plot and the adjoint differential states are shown in the right plot. Both plots indicate the 64 equidistant FEs with ticks at the top. The number of collocation points for all components is three in the realized numerical experiment.

for the control as

$$\boldsymbol{u}(t)^* = \begin{cases} -\frac{2}{3l}\left(1-\frac{t}{3l}\right), & t \in [0, 3l], \\ 0, & t \in [3l, 1-3l], \\ -\frac{2}{3l}\left(1-\frac{1-t}{3l}\right), & t \in [1-3l, 1], \end{cases}$$

for the differential state multipliers as

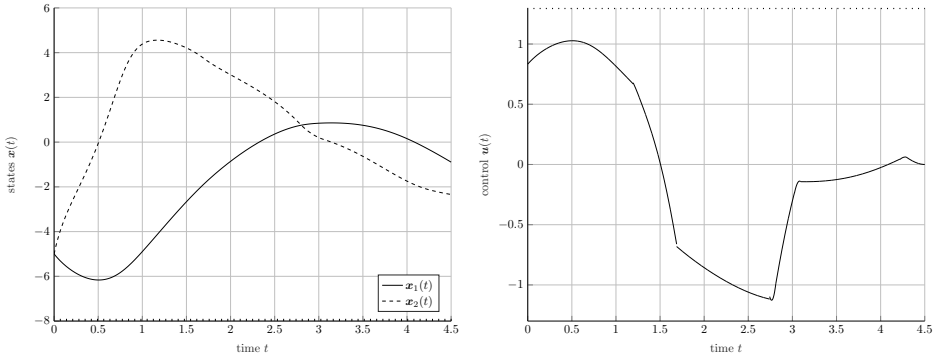
$$\boldsymbol{\lambda}_1^*(t) = -\frac{2}{9l^2}, \qquad t \in [0, 1],$$

$$\boldsymbol{\lambda}_2^*(t) = \begin{cases} \frac{2}{3l}\left(1-\frac{t}{3l}\right), & t \in [0, 3l], \\ 0, & t \in [3l, 1-3l], \\ \frac{2}{3l}\left(1-\frac{1-t}{3l}\right), & t \in [1-3l, 1], \end{cases}$$

and finally for the constraint multiplier as

$$\boldsymbol{\nu}^*(t) = \begin{cases} -\frac{4}{9l^2}, & t \in [0, 3l], \\ -\frac{2}{9l^2}, & t \in [3l, 1-3l], \\ 0, & t \in [1-3l, 1]. \end{cases}$$

Note that the optimal costates $\boldsymbol{\lambda}_1^*(\cdot)$, $\boldsymbol{\lambda}_2^*(\cdot)$, and $\boldsymbol{\nu}^*(\cdot)$ denote costates in the sense of the indirect adjoining approach with continuous adjoints according to Hartl et al. [224], i.e.,



a) Differential states b) Control

Figure 13.5: The figure depicts the differential state trajectories and the control trajectory for OCP (13.3). The discretization was done with 64 FEs and three collocation points per component.

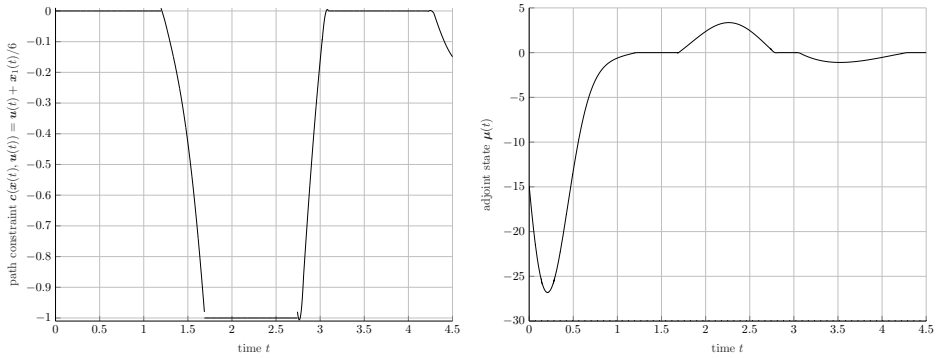
the differential costates are continuous. This seems desirable for us since we can handle the differential costates in a unified way within systems subject to mixed control–state constraints and pure state constraints in the context of our error control routines. It allows us to shift the discontinuity treatment to the pure state constraint costates.

As Hartl et al. [224] point out, the necessary optimality conditions of the PMP and the respective conditions of the indirect adjoining approach with continuous adjoints can be related under suitable assumptions.

We implemented the model within our software `grc` and solved the problem with two different discretization schemes. The coarser discretization employs 10 equidistant FEs and three collocation points per FE and component. While the number of collocation points remains unchanged for the finer discretization schemes, we increased the number of FEs to 20. The FE grid is chosen equidistantly again for the fine discretization. We solved the resulting NLPs with `Ipopt` where second–order derivative information was calculated with AD. The convergence tolerance of `Ipopt` was chosen to be equal to 1×10^{-10}.

Figures 13.7–13.9 show the primal and dual solution approximations for the coarse discretization scheme. Likewise, Figures 13.10–13.12 depict the primal and dual solution approximations for the fine discretization scheme. The analytical solutions for the control and both differential costates in Figure 13.7 and 13.8 indicate a clear deviation of the associated approximations. The pure state constraint costate approximation in Figure 13.9 shows that the jumps at time instants $t = 0.25$ and $t = 0.75$ could not be resolved. Those jump time instants are not part of the FE grid which could be an explanation for the unresolved jumps. In contrast, the fine grid covers the jump time instants. The resulting costate approximation in Figure 13.12 shows well resolved jumps. Control and differential costate approximations in Figure 13.10 and 13.11 also show a proper accordance with the analytical solutions.

Our numerical experiments show that the FE grid distribution has a crucial impact on the fact if costates involving jumps can be resolved properly or not. In order to end up with well

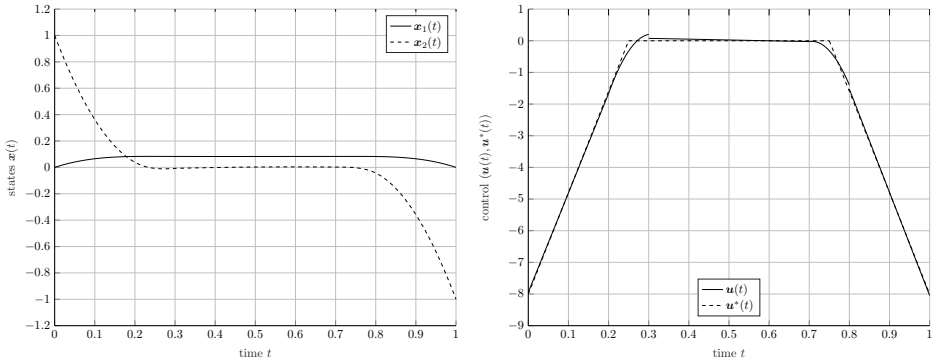


a) Mixed control–state constraint

b) Adjoint for mixed control–state constraint

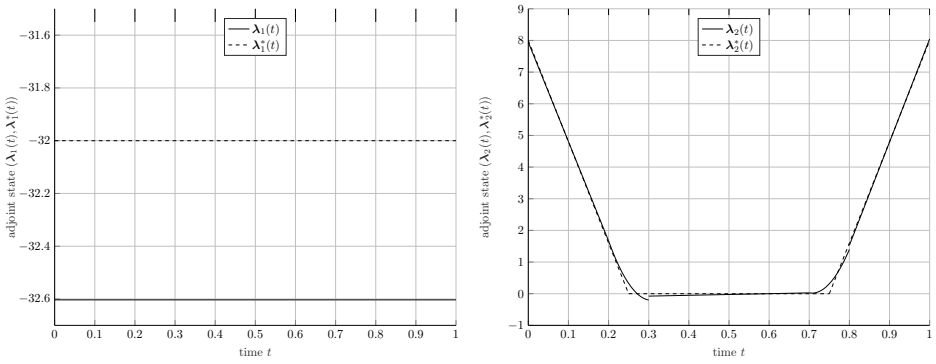
Figure 13.6: The mixed control–state constraint $\boldsymbol{u}(t) + \frac{\boldsymbol{x}_1(t)}{6}$ for OCP (13.3) is portrayed in the left plot. The right plot shows the associated adjoint. The discretization was done with 64 equidistant FEs and three collocation points.

resolved jumps one needs to detect the jump time instants. An efficient a posteriori error estimation needs to consider how the pure state constraint contribution involving a Stieltjes integral should be approximated on the one hand and how jumps can be detected properly on the other hand. We are far from a final and satisfactory solution but in the following Chapter 14 we present first considerations how a goal–oriented error estimation can be realized for OCPs with pure state constraints.



a) Differential states b) Control

Figure 13.7: The figure depicts the differential state trajectories and the control trajectory for OCP (13.4). The discretization was done with 10 FEs (indicated by ticks at the top of both plots) and three collocation points per FE. The control plot also includes the analytical solution $\boldsymbol{u}^*(\cdot)$.



a) Differential costate for state $\boldsymbol{x}_1(\cdot)$ b) Differential costate for state $\boldsymbol{x}_2(\cdot)$

Figure 13.8: Differential costate trajectories for OCP (13.4) are depicted in this figure. The number of FEs was chosen to be 20 and the number of collocation points is three. Both plots depict the FEs marked by ticks at the top and the analytical solutions with dashed lines. Both trajectories deviate significantly from the analytical solutions.

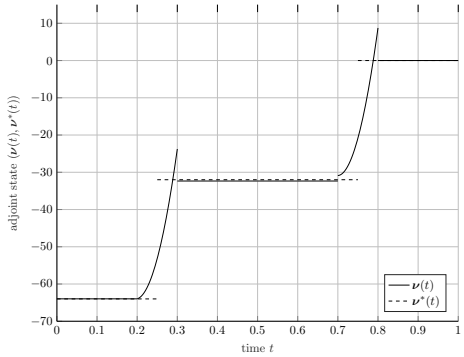
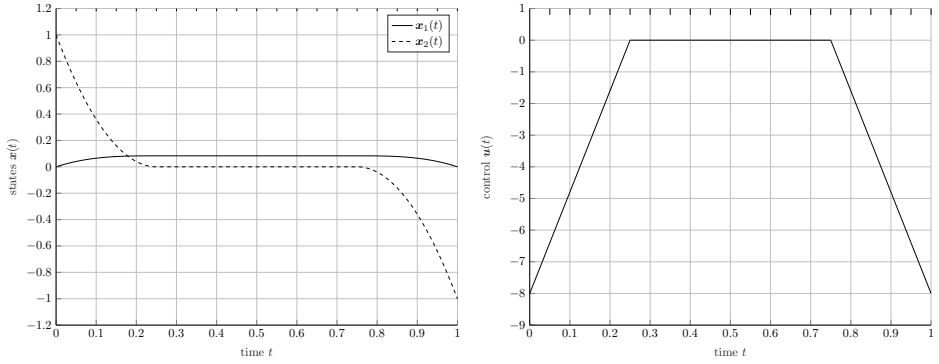
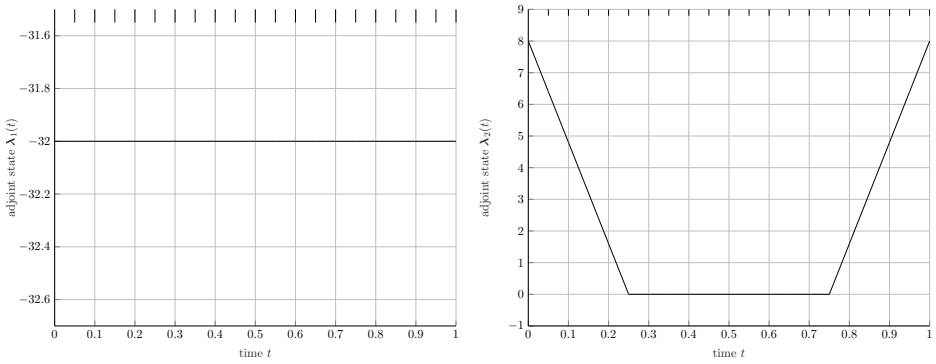


Figure 13.9: The figure shows the costate trajectory for the pure state constraint of OCP (13.4). The discretization has 10 FEs (indicated by ticks at the top) and three collocation points per FE. The figure also includes the analytical solution $\nu^*(\cdot)$. The jumps at the jump points $t = 0.25$ and $t = 0.75$ are not resolved properly. The jump points are not part of the FE discretization.



a) Differential states b) Control

Figure 13.10: The figure depicts the differential state trajectories and the control trajectory for OCP (13.4). The discretization was done with 20 FEs (indicated by ticks at the top of both plots) and three collocation points per FE. Compared to the coarser discretization with 10 FEs (see Figure 13.7) there is significantly less deviation from the analytical solution.



a) Differential costate for state $\boldsymbol{x}_1(\cdot)$ b) Differential costate for state $\boldsymbol{x}_2(\cdot)$

Figure 13.11: The figure depicts differential costate trajectories for OCP (13.4). The number of FEs is 20 and the number of collocation points is three. Both plots depict the FEs marked by ticks at the top. If we compare both trajectories with their respective counterparts in Figure 13.8 we can see that, regarding the deviation from the analytical solution, the discretization with 20 FEs is significantly better than the one with 10 FEs.

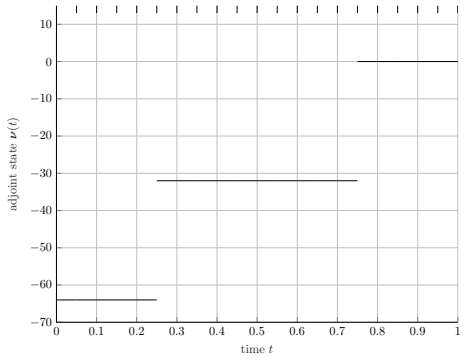


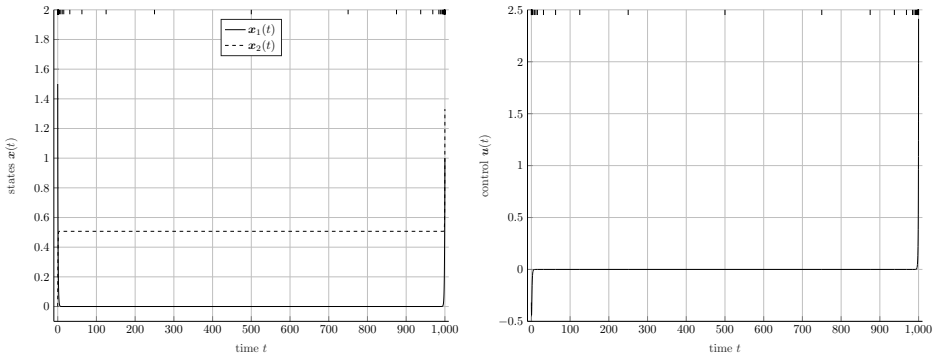
Figure 13.12: The figure shows the costate trajectory for the pure state constraint of OCP (13.4). The discretization has 20 equidistant FEs (indicated by ticks at the top) and three collocation points per FE. In contrast to the discretization with 10 FEs (see Figure 13.9), the jump points $t = 0.25$ and $t = 0.75$ are part of the FE discretization. The jumps are resolved properly now.

Chapter 14

Goal–Oriented Error Estimation Numerics

In this chapter, we investigate an *hp* refinement strategy, which is based on a non–smoothness detection step (see Liu et al. [299]) to control the polynomial degree for the discretization scheme and on our novel goal–oriented error estimation (see Chapter 10) to update the FE grid. The *hp* refinement strategy is then applied to several benchmark problems in order to illustrate its excellent performance and practical applicability.

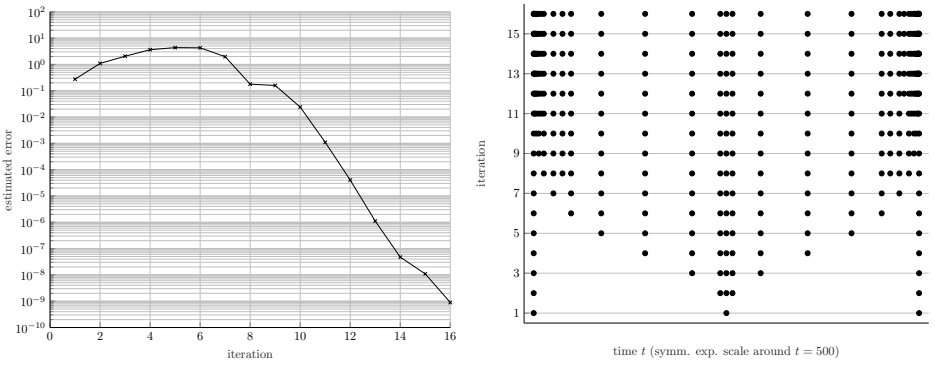
14.1 Hyper–Sensitive Problem I



a) Differential states b) Control

Figure 14.1: The figure depicts the differential state trajectories as well as the control trajectory for OCP (14.1) after employing our adaptive mesh refinement routine based on the goal–oriented error estimation of Chapter 10. The initial discretization was done with two equidistant FEs and three collocation points per FE. It took 16 iterations until termination where the termination tolerance for the estimated error was set to 1×10^{-8}. The ticks at the top of the plots indicate the final FE grid. One can see a distinct accumulation of FE points at the horizon borders. That is the grid is finer where the trajectories undergo abrupt changes of the curvature. There is no fill–up of FE grid points where the trajectories show a nearly constant behavior.

In this section, we investigate an OCP, which was originally introduced by Rao and Mease [368], and whose adjoint equation of the PMP is completely hyper–sensitive for a sufficiently large optimization horizon. The problem also acts as a benchmark problem of the well–established OCP software GPOPS-II [351]. It has the specific characteristic that a state tra-



a) Estimated error development b) FE grid development

Figure 14.2: The left plot shows the development of the estimated error corresponding to the iteration for the numerical experiment which is described in Figure 14.1. On the right, the development of the FE grid is depicted. The error estimation plot is a semi–log plot with respect to the y–axis and the FE grid plot is an exponential plot with respect to the x–axis in order to disperse the accumulation of grid points at the horizon borders.

jectory is divided into three phases, namely a "take–off" phase, followed by a "cruise" phase, and finally a "landing" phase. While the trajectory is nearly constant in the "cruise" phase, all the action takes place during the "take–off" and "landing" phase. Moreover, the longer the optimization horizon is chosen the more time is spent in the "cruise" phase and the more extreme is the slope of the trajectory close to the horizon borders. The trajectory has an exponential like behavior during the "take–off" and "landing" phase. In the following Section 14.2, we will investigate a slightly modified version of this problem whose analytical solution is known.

The model equations together with the performance criterion and a boundary value constraint look as follows:

$$\begin{aligned} \min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)} \quad & \boldsymbol{x}_2(t_\mathrm{f}) \\ \text{s.t.} \quad & \dot{\boldsymbol{x}}_1(t) = -\boldsymbol{x}_1(t)^3 + \boldsymbol{u}(t), \\ & \dot{\boldsymbol{x}}_2(t) = \frac{1}{2}\left(\boldsymbol{x}_1(t)^2 + \boldsymbol{u}(t)^2\right), \\ & \boldsymbol{x}(0) = [1,0]^T,\ \boldsymbol{x}_1(t_\mathrm{f}) = 1.5,\ t_\mathrm{f} = 1000. \end{aligned} \tag{14.1}$$

As Rao and Mease [368] could show by means of an indirect approach, there appear rapid changes in the solution close to the borders of the optimization horizon $[0, t_\mathrm{f}]$. However, most parts of the solution trajectories are nearly constant. We conducted a numerical experiment with OCP (14.1) where we started with a rather coarse FE grid and employed a mesh refinement strategy in order to reduce the error with respect to the performance criterion. To this end we used our goal–oriented error estimation which provides us with an element–wise error

contribution. Depending on whether the required tolerance is fulfilled on a FE or not, the discretization is updated. If the tolerance is satisfied, the discretization remains unchanged otherwise it is checked if the trajectories are smooth or not with the smoothness check of Liu et al. [299]. If the trajectories are smooth we increase the polynomial degree. If the trajectories are not smooth we split the FE into several new FEs depending on non–smoothness of the trajectories measured by a smoothness value.
We initiated the numerical experiment with two FEs and three collocation points. We used `Ipopt` with exact second–order derivative information and its standard convergence tolerance to solve the arising NLPs. The termination tolerance for the adaptive mesh refinement approach was set to 1×10^{-8}. The maximal refinement parameter, i.e., the number of FEs that can be created during the mesh refinement per iteration and FE, was set to two. Our algorithm took 16 iterations until termination.
The differential state trajectories and the control trajectory after termination together with the final FE grid are depicted in Figure 14.1. The left plot of Figure 14.2 shows the development of the estimated error over the iteration process. The development of the FE grid over the iteration process in an exponential scale around the midpoint of the optimization horizon $t = 500$ is depicted in the right plot of Figure 14.2. Near the initial time and near the final time one can observe an accumulation of grid points while the rest of the horizon does not undergo a fill–up of grid points, i.e., the grid points are set where the interesting behavior takes place.

14.2 Hyper–Sensitive Problem II

In this section, we consider a slight modification of the OCP from the previous Section 14.1. However, its analytical solution is known such that we can compare the estimated and the exact error. This OCP has already been used to study other mesh refinement techniques, cf. e.g. Patterson et al. [352] and Liu et al. [299]. The full OCP reads as

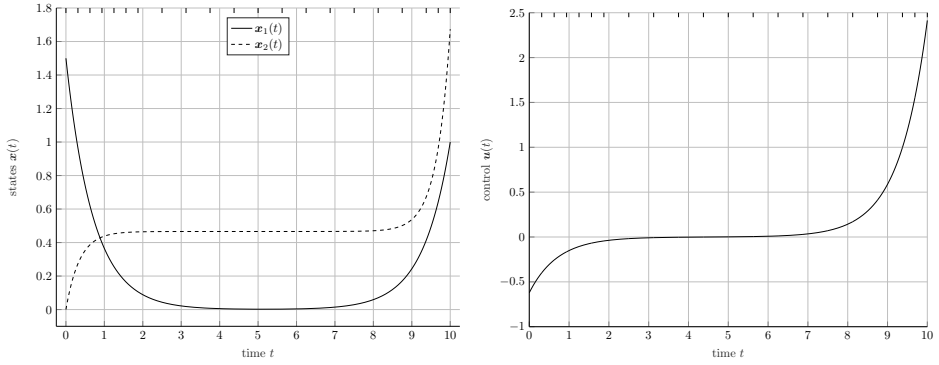
$$\begin{aligned} \min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)} \quad & \boldsymbol{x}_2(t_\mathrm{f}) \\ \text{s.t.} \quad & \dot{\boldsymbol{x}}_1(t) = -\boldsymbol{x}_1(t) + \boldsymbol{u}(t), \\ & \dot{\boldsymbol{x}}_2(t) = \frac{1}{2}\left(\boldsymbol{x}_1(t)^2 + \boldsymbol{u}(t)^2\right), \\ & \boldsymbol{x}(0) = [1.5, 0]^T,\ \boldsymbol{x}_1(t_\mathrm{f}) = 1,\ t_\mathrm{f} = 10. \end{aligned} \tag{14.2}$$

The analytical solution to this OCP is given as

$$\begin{bmatrix} \boldsymbol{x}_1^*(t) \\ \boldsymbol{u}(t) \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1+\sqrt{2} & 1-\sqrt{2} \end{bmatrix} \begin{bmatrix} c_1 \exp(+t\sqrt{2}) \\ c_2 \exp(-t\sqrt{2}) \end{bmatrix},$$

where the constants c_1 and c_2 are given as

$$\begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \frac{1}{\exp(-t_\mathrm{f}\sqrt{2}) - \exp(+t_\mathrm{f}\sqrt{2})} \begin{bmatrix} +1.5\exp(-t_\mathrm{f}\sqrt{2}) - 1 \\ -1.5\exp(+t_\mathrm{f}\sqrt{2}) + 1 \end{bmatrix}.$$

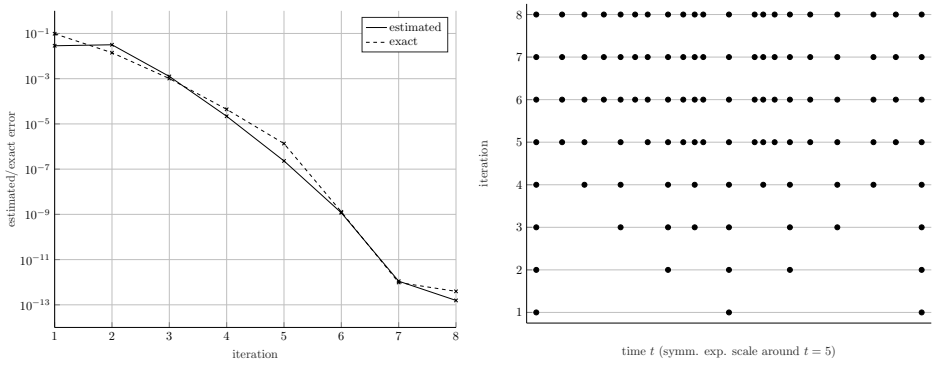


a) Differential states b) Control

Figure 14.3: The figure depicts the differential state trajectories as well as the control trajectory for OCP (14.1) after employing our adaptive mesh refinement routine based on the goal–oriented error estimation of Chapter 10. The initial discretization was done with two equidistant FEs and three collocation points per FE. It took eight iterations until termination where the termination tolerance for the estimated error was set to 1×10^{-12}. The ticks at the top of the plots indicate the final FE grid. One can see a distinct accumulation of FE points at the horizon borders. That is the grid is finer where the trajectories undergo abrupt changes of the curvature. There is no fill–up of FE grid points where the trajectories show a nearly constant behavior.

We used the same mesh refinement approach as described in the previous Section 14.1. The SNLP type algorithm was initiated with a coarse grid just having two FEs and a polynomial degree for differential states and control of three. Arising NLP instances were solved with the help of `Ipopt` where we used the exact second–order derivative information option of the software. The maximal refinement parameter was set to two. The SNLP mesh refinement approach took eight iterations to fulfill the termination tolerance of 1×10^{-12}.

Figure 14.3 depicts the differential state trajectories and the control trajectory after eight iterations. Furthermore, the top of both plots in the figure show the FE grid marked by ticks. One can see that the exponential decay and growth during the "take–off" and "landing" phase are less distinct compared to OCP (14.1). However, this changes for larger values of t_f. The right plot of Figure 14.4 shows the FE grids for all SNLP iterations. The left plot of Figure 14.4 contains the estimated as well as the exact error with respect to the performance criterion of OCP (14.2). One can see that both graphs show a very similar convergence behavior. An illustration of the SNLP grids in combination with changes in the polynomial degrees is given by the plot in Figure 14.5. Roughly speaking, the polynomial degree is gradually increased from the middle of the horizon ("cruise" phase) to its borders ("take–off"/"landing" phase).



a) Estimated/exact error development

b) FE grid development

Figure 14.4: The development of both the exact (dashed line) and the estimated (solid line) error is depicted in the left plot of this figure for the numerical experiment described in Figure 14.3. The error is illustrated with an logarithmic scale (y–axis). Exact and estimated error show a very similar behavior and show a convergence behavior as expected by our theoretical investigations. The grid development for the experiment is shown in the right plot of the figure. We use an exponential scale for the time horizon (x–axis) to disperse the accumulation of grid points at the horizon borders.

14.3 Rayleigh Problem With Mixed Control–State Constraint II

In this section, we slightly modify the Rayleigh problem that has been considered in Section 13.3 such that the control appears only linearly in the objective functional, the differential equation, and in the mixed control–state constraint. We consider the OCP

$$
\begin{aligned}
\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)} \quad & \int_0^{t_\mathrm{f}} \left(\boldsymbol{x}_1(t)^2 + \boldsymbol{x}_2(t)^2\right) \mathrm{d}t \\
\text{s.t.} \quad & \dot{\boldsymbol{x}}_1(t) = +\boldsymbol{x}_2(t), \\
& \dot{\boldsymbol{x}}_2(t) = -\boldsymbol{x}_1(t) + \boldsymbol{x}_2(t)\left(1.4 - 0.14 \cdot \boldsymbol{x}_2(t)^2\right) + 4\,\boldsymbol{u}(t), \\
& -2 \leqslant \boldsymbol{u}(t) + \frac{\boldsymbol{x}_1(t)}{6} \leqslant 0, \\
& \boldsymbol{x}(0) = [-5,-5]^T,\ t_\mathrm{f} = 4.5.
\end{aligned}
\tag{14.3}
$$

The notation in OCP (14.3) remains the same as in Section 13.1–13.3. Maurer [317] solved the problem with an indirect approach. We use his results as a reference solution. He detected the mixed control–state constraint as "bang–singular–bang–singular" type constraint which makes the problem rather challenging to solve.

We solved the problem with the mesh refinement approach that we developed in Chapter 10 and that was also applied in the previous sections. We initialized the discretization scheme

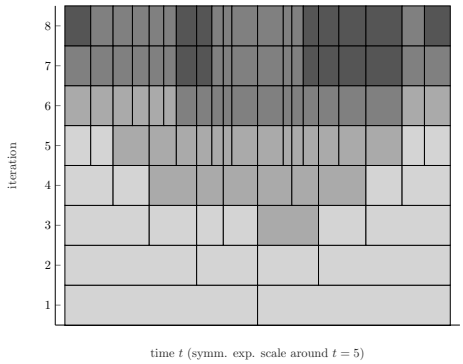


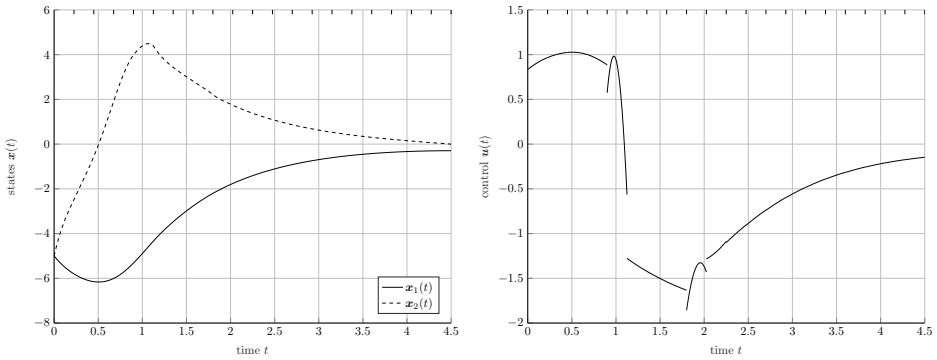
Figure 14.5: The figure depicts the FE grid development and the polynomial degree development for the numerical experiment described in Figure 14.3. Note that an exponential scale for the time horizon axis (x–axis) around the horizon midpoint was used. One can see that the algorithm works in accordance with the trajectories depicted in Figure 14.3, i.e., the polynomial degree is increased where the trajectories show a nearly constant behavior. Near the horizon borders the number of FEs is gradually increased. The darker shaded a "FE rectangle" is, the higher is the polynomial degree.

with 10 FEs and three collocation points for both differential states and control approximating polynomials. The arising NLP instances were solved with `Ipopt` where we used exact second–order information and the standard convergence tolerance. The termination tolerance for the SNLP algorithm was set to 1×10^{-8} and it terminated after seven iterations successfully. The parameter defining maximum allowed refinement of a FE was set to three.

Figure 14.6 shows the differential state trajectories and the control trajectory after two iterations. The mixed control–state and the scaled switching function (see Maurer [317]) are depicted in the left plot of Figure 14.7. The right plot of Figure 14.7 contains the constraint costate. One can observe that the transitions from the first "bang" to the first "singular arc" phase and from the first "singular arc" to the second "bang" phase are not properly resolved after the second iteration. This is different from the approximate solution after seven iterations as one can see in Figures 14.8 and 14.9. The FE grid indicated by ticks at the top of the plots is rather fine where transitions from "bang" to "singular arc" phases take place and vice versa. In contrast, the grid is not refined compared to the grid of the second iteration in regions where the solution is smoother. The final trajectories coincide with the ones of Maurer [317]. The decay of the estimated error over the iteration process is depicted in Figure 14.10 alongside with the FE grid.

14.4 Problem with Tangential Path Constraint Exit

In this section, we investigate an OCP subject to a control box constraint and a pure state constraint. The full problem reads as



a) Differential states b) Control

Figure 14.6: The figure shows the differential state trajectories as well as the control trajectory for OCP (14.3) after the second iteration of the adaptive mesh refinement approach described in Chapter 10. Ticks at the top of the plots indicate the actual (equidistant) FE grid. The arcs of the control trajectory are not well resolved.

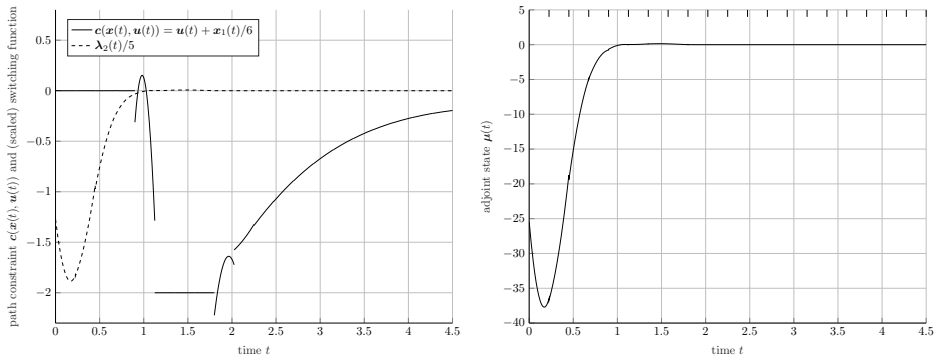
$$\begin{aligned} \min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)} \quad & \int_0^3 e^{-r\,t}\,\boldsymbol{u}(t)\,\mathrm{d}t \\ \text{s.t.} \quad & \dot{\boldsymbol{x}}(t) = \boldsymbol{u}(t), \\ & 0 \leqslant \boldsymbol{u}(t) \leqslant 3, \\ & 0 \leqslant \boldsymbol{x}(t) - 1 + (t-2)^2, \\ & \boldsymbol{x}(0) = 0,\ r = 1. \end{aligned} \tag{14.4}$$

Hartl et al. [224, Example 9.2.] calculate the analytical solution to OCP (14.4): differential state and control solutions $\boldsymbol{x}^*(\cdot)$ and $\boldsymbol{u}^*(\cdot)$ are given as

$$\boldsymbol{x}^*(t) = \begin{cases} 0, \\ 1-(t-2)^2, \\ 1, \end{cases} \qquad \boldsymbol{u}^*(t) = \begin{cases} 0, \\ 2\,(2-t), \\ 0, \end{cases} \qquad t \in \begin{cases} [0,1), \\ [1,2], \\ (2,3]. \end{cases}$$

For the differential costates $\boldsymbol{\lambda}^*(\cdot)$ and the pure state constraint costates $\boldsymbol{\nu}^*(\cdot)$ (indirect adjoining with continuous multipliers) they derived the solutions

$$\boldsymbol{\lambda}^*(t) \equiv 0, \quad t \in [0,3], \qquad \boldsymbol{\nu}^*(t) = \begin{cases} -e^{-1}, & t \in [0,1), \\ -e^{-t}, & t \in [1,2], \\ 0, & t \in (2,3]. \end{cases}$$



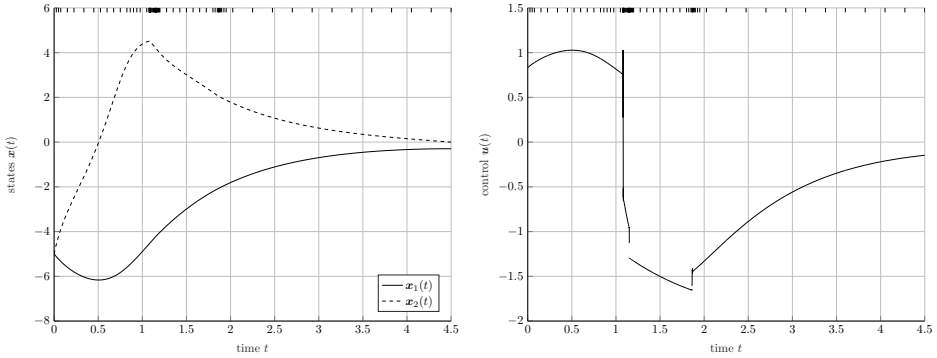
a) Mixed control–state constraint b) Adjoint for mixed control–state constraint

Figure 14.7: The left plot shows the mixed control–state constraint and the scaled switching function for OCP (14.3) after the second iteration of the adaptive mesh refinement approach described in Chapter 10. One can see that the "bang–singular–bang–singular" structure is not properly resolved after this iteration. The right plot shows the constraint costate for the same problem and the same iteration. Furthermore, the equidistant FE grid is illustrated by means of ticks at the top.

We briefly analyze the pure state constraint structure in the solution. To this end, let the function $\boldsymbol{c}^*(\cdot)$ be defined as the constraint function in the solution $\boldsymbol{c}^*(t) \stackrel{\text{def}}{=} \boldsymbol{c}(t, \boldsymbol{x}^*(t)) \stackrel{\text{def}}{=} \boldsymbol{x}^*(t) - 1 + (t-2)^2$. The derivative $\dot{\boldsymbol{c}}^*(\cdot)$ is given as $\dot{\boldsymbol{c}}^*(t) = \boldsymbol{u}(t)^* + 2(t-2)$ since $\dot{\boldsymbol{x}}^* = \boldsymbol{u}^*$. Since $\boldsymbol{u}^*(\cdot)$ is discontinuous at time instant $t = 1$ this also holds for $\dot{\boldsymbol{c}}^*(\cdot)$ which makes the entry point non–tangential. In contrast, $\boldsymbol{u}^*(\cdot)$ and therefore also $\dot{\boldsymbol{c}}^*(\cdot)$ is continuous at time instant $t = 2$ which makes the exit point a tangential point.

We solved OCP (14.4) with our software package `grc` where we used the NLP solver SNOPT with its standard configuration to determine solutions for the discretized problem instances. We solved the problem with two different discretization schemes. The first configuration has 30 equidistant FEs and two collocation points for the approximating polynomials. For the second configuration we used 90 equidistant FEs and three collocation points. The approximate trajectories for the first configuration are depicted in Figure 14.11 and Figure 14.12. They match very well with the exact solutions $\boldsymbol{x}^*$, $\boldsymbol{u}^*$, $\boldsymbol{\lambda}^*$ and $\boldsymbol{\nu}^*$. The same holds for the second configuration which is not depicted for this reason.

Note that for both configurations the FE grid contains both entry ($t = 2$) and exit point ($t = 3$). For this reason, we can avoid the unresolved jumps that we investigated in Section 13.4. Nevertheless, an efficient mesh adaption requires a reliable jump detection algorithm. For our numerical experiments we achieved already good results by implementing some first– and second–order checking heuristics based on derivative information that we obtained from the software package `chebfun` [141]. Recently, Miller et al. [326] proposed another approach to jump function detection by means of a Fourier series approach approximation approach in an OCP context. However, jump detection and efficient adaptive mesh refinement for pure state constrained OCPs with our new goal–oriented error estimation is still up to future research.



a) Differential states b) Control

Figure 14.8: The figure shows the differential state trajectories and the control trajectory for OCP (14.3) after seven iterations of the adaptive mesh refinement approach described in Chapter 10. Ticks at the top of the plots indicate the actual FE grid. The arcs of the control trajectory are well resolved.

As we had already done in Section 13.4, we employ the indirect adjoining approach with continuous adjoints to express the costates. Thus, we can handle the differential equation part in the same way as (10.12) since no jumps appear in the differential state. However, it makes no sense to use (10.13) in order to evaluate the path constraint part of the error estimation as jumps may occur. Instead, we use the linearity of the Lebesgue–Stieltjes integral in the measure function and split the error representation integral $\int_{\mathcal{I}_n} \boldsymbol{c}(t, \boldsymbol{x}(t)) \,\mathrm{d}[\boldsymbol{\nu} - \boldsymbol{\nu}_\mathrm{h}](t)$ element–wise into

$$\int_{\mathcal{I}_n} \boldsymbol{c}(t, \boldsymbol{x}(t)) \,\mathrm{d}\, \boldsymbol{\nu}(t) - \int_{\mathcal{I}_n} \boldsymbol{c}(t, \boldsymbol{x}(t)) \,\mathrm{d}\, \boldsymbol{\nu}_\mathrm{h}(t). \tag{14.5}$$

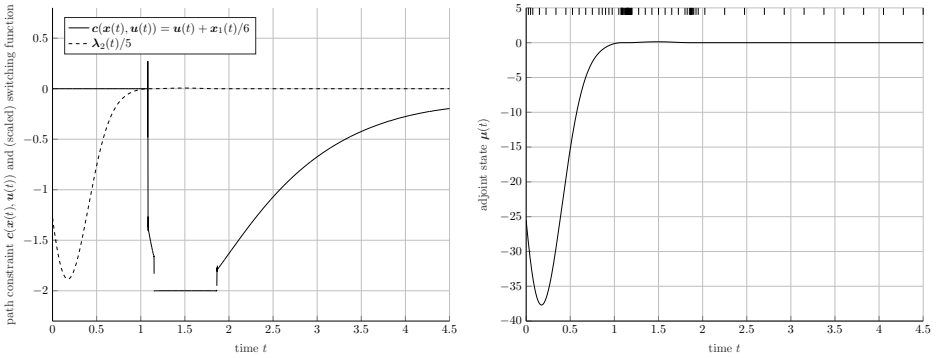
Both integrals in (14.5) must be approximated. Cerone and Dragomir [102] proposed the following rule to approximate a Stieltjes integral on an interval $\mathcal{I} = [a, b]$:

$$\int_{\mathcal{I}} \boldsymbol{f}(t) \,\mathrm{d}\boldsymbol{\mu}(t) = \frac{\mu(b) - \mu(a)}{b - a} \int_{\mathcal{I}} \boldsymbol{f}(t) \,\mathrm{d}t.$$

We employed the formula to approximate the integrals in (14.5) where we used the LGR quadrature rule to approximate the Lebesgue integrals. For the jump handling in both integrals we made use of Theorem 2.79. We approximate the second integral in (14.5) as

$$\boldsymbol{c}(t_{n-1}, \boldsymbol{x}_\mathrm{h}(t_{n-1}))\,(\boldsymbol{\nu}_\mathrm{h}(t_{n-1}^+) - \boldsymbol{\nu}_\mathrm{h}(t_{n-1}^-)) + \frac{\boldsymbol{\nu}_\mathrm{h}(t_n^-) - \boldsymbol{\nu}_\mathrm{h}(t_{n-1}^+)}{|\mathcal{I}_n|} \int_{\mathcal{I}_n} \boldsymbol{c}(t, \boldsymbol{x}_\mathrm{h}(t)) \,\mathrm{d}t,$$

where the Lebesgue integral is approximated with a high–order LGR quadrature rule. It is slightly more difficult to find a good approximation for the first integral in (14.5) since a high–



a) Mixed control–state constraint b) Adjoint for mixed control–state constraint

Figure 14.9: The left plot shows the mixed control–state constraint and the scaled switching function for OCP (14.3) after seven iterations of the adaptive mesh refinement approach described in Chapter 10. It is easy to see that the "bang–singular–bang–singular" structure is well resolved after this iteration. The right plot shows the constraint costate for the same problem and the same iteration. The FE grid, indicated by ticks at the top of the plot, is rather fine near the "bang" and "singular arc" phase transitions.

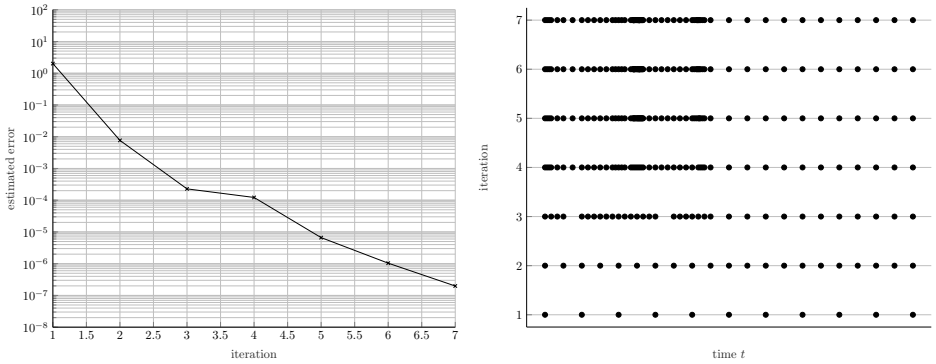
order polynomial approximation involving jumps is of little use. Thus, we used our `chebfun` jump detection algorithm and used just neighboring FEs for the high–order approximation where no jump is involved. As a guess for the time instant of the jump we took the midpoint of the interval $[t_0^{(n)}, t_1^{(n)}]$, i.e., the midpoint of the interval between $t_{n-1} \equiv t_0^{(n)}$ and the first collocation point $t_1^{(n)}$ of the FE $\mathcal{I}_n$. If we denote this point with $t_{m,n}$, we can write the approximation of the first integral in (14.5) as

$$\boldsymbol{c}(t_{m,n}, \boldsymbol{x}_\mathrm{h}(t_{m,n}))(\boldsymbol{\nu}(t_{m,n}^+) - \boldsymbol{\nu}(t_{m,n}^-)) + \frac{\boldsymbol{\nu}(t_{m,n}^-) - \boldsymbol{\nu}(t_{n-1}^+)}{t_{m,n} - t_{n-1}} \int_{[t_{n-1}, t_{m,n}]} \boldsymbol{c}(t, \boldsymbol{x}_\mathrm{h}(t))\,\mathrm{d}t$$
$$+ \frac{\boldsymbol{\nu}(t_n^-) - \boldsymbol{\nu}(t_{m,n}^+)}{t_n - t_{m,n}} \int_{[t_{m,n}, t_n]} \boldsymbol{c}(t, \boldsymbol{x}_\mathrm{h}(t))\,\mathrm{d}t,$$

where we approximate again the arising Lebesgue integrals by means of high–order LGR quadrature rules.

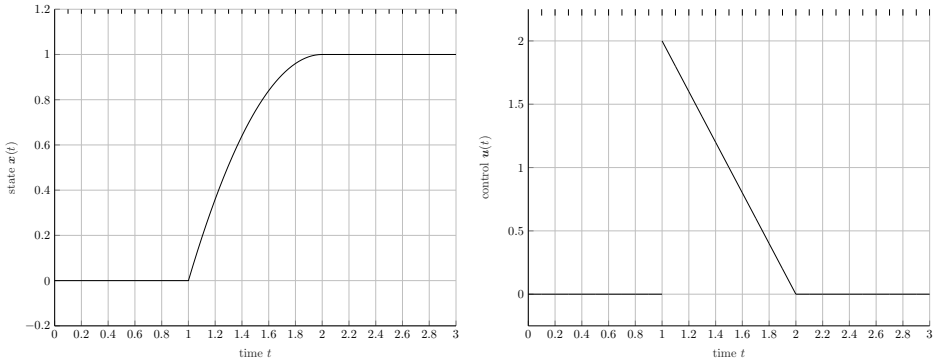
We applied the error estimation just described to the two previously mentioned discretizations. Since we have the analytical solution to OCP (14.4) we can compare the estimated and the exact errors in the performance criterion. For the first configuration we find an estimated error 6.18×10^{-5} compared to an exact error of 7.68×10^{-6}. For the second configurations estimated and exact error have the values 1.5×10^{-6} and 1.62×10^{-6}, respectively.

The proposed error approximation may serve as a good starting point to intensify the research on error approximation for OCPs involving pure state constrains and to develop tailored grid adaption routines.



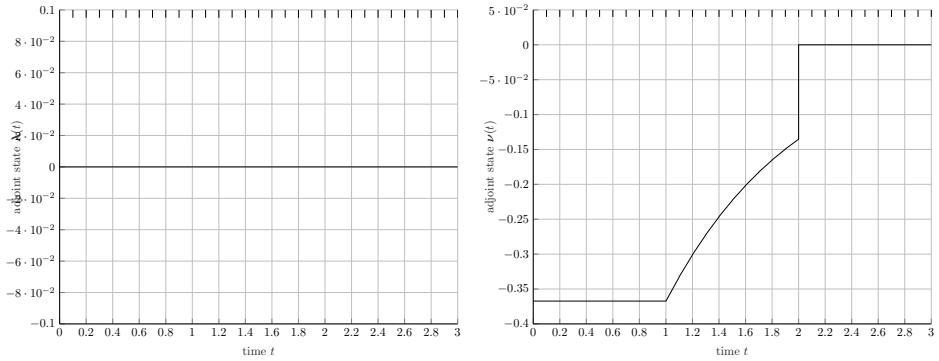
a) Estimated error development b) FE grid development

Figure 14.10: Depicted are the development of the estimated error and the FE grid of the adaptive mesh refinement approach described in Chapter 10 when applied to OCP (14.3).



a) Differential state b) Control

Figure 14.11: The figure depicts the differential state as wells as the control trajectory for OCP (14.4) where the discretization scheme has 30 FEs and an approximating polynomial degree of two. Both trajectories coincide very well with the analytical solution.



a) Differential state costate

b) Pure state constraint costate

Figure 14.12: Depicted are the costates for the differential state and the pure state constraint of OCP (14.4) where the discretization scheme has 30 FEs and the approximating polynomials are of degree two. Both trajectories are in accordance with the analytical solutions for the costates of OCP (14.4).

Chapter 15

Switched Optimal Control Problem Numerics

In this section we demonstrate the applicability of our novel approach to switched OCPs (see Chapter 11) on three different benchmark problems. The first problem is a time optimal control problem and exhibits consistent switching behavior. Problem two and three are initial value problems with inconsistent switches. The results of this chapter are also part of our publication Bock et al. [78].

15.1 A Coulombic Friction Model

We consider the model with consistent switches in Christiansen et al. [110] where a copper coil is guided in the air gap on a slider. The coil and slider mass is denoted by m_1. The Coulombic friction force F_R, which acts in the direction opposite to the velocity is produced by the linear guide. A load mass m_2 is mounted on the slider with a spring k that has negligible damping. A coil current I induces the actuating force $\boldsymbol{F}(t) = K_F\, \boldsymbol{I}(t)$. The moving coil with the velocity $\boldsymbol{v}_1$ generates a voltage $\boldsymbol{U}(t) = K_S\, \boldsymbol{v}_1(t)$.
The system states are the motor mass position $\boldsymbol{x}_1(t)$, the motor mass velocity $\boldsymbol{v}_1(t)$, the load mass position $\boldsymbol{x}_2(t)$, the load mass velocity $\boldsymbol{v}_2(t)$ and the electric current $\boldsymbol{I}(t)$. The control variable of the motor is the voltage $\boldsymbol{U}(t)$. The piecewise linear model equation reads as

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{U}(t)),$$

where the state vector is denoted by $\boldsymbol{x}(t) = [\boldsymbol{x}_1(t), \boldsymbol{v}_1(t), \boldsymbol{x}_2(t), \boldsymbol{v}_2(t), \boldsymbol{I}(t)]^T$. The right hand side function $\boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{U}(t)) = [\boldsymbol{f}_1, \boldsymbol{f}_2, \boldsymbol{f}_3, \boldsymbol{f}_4, \boldsymbol{f}_5]^T(\boldsymbol{x}(t), \boldsymbol{U}(t))$ is given as

$$\begin{aligned}
\boldsymbol{f}_1(\boldsymbol{x}(t), \boldsymbol{U}(t)) &= \boldsymbol{v}_1(t),\\
\boldsymbol{f}_2(\boldsymbol{x}(t), \boldsymbol{U}(t)) &= (K_F\, \boldsymbol{I}(t) - k\,(\boldsymbol{x}_1(t) - \boldsymbol{x}_2(t)) - F_R\, \operatorname{sgn}(\boldsymbol{v}_1(t)))\,/m_1,\\
\boldsymbol{f}_3(\boldsymbol{x}(t), \boldsymbol{U}(t)) &= \boldsymbol{v}_2(t),\\
\boldsymbol{f}_4(\boldsymbol{x}(t), \boldsymbol{U}(t)) &= k\,(\boldsymbol{x}_1(t) - \boldsymbol{x}_2(t))/m_2,\\
\boldsymbol{f}_5(\boldsymbol{x}(t), \boldsymbol{U}(t)) &= (\boldsymbol{U}(t) - R\,\boldsymbol{I}(t) - K_S\, \boldsymbol{v}_1(t))\,/L.
\end{aligned}$$

The time horizon of the system is given by the interval $[0, t_f]$. We want to investigate the system on the unified time horizon $[0, 1]$. This can be achieved by the linear time transformation $t(\tau) = \tau\, t_f$, $\tau \in [0, 1]$, cf. Section 5.1. We choose t_f as an additional variable to achieve a system with free final time. Define

$$\overline{\boldsymbol{x}}(\tau) = \boldsymbol{x}(t(\tau)), \quad \overline{\boldsymbol{U}}(\tau) = \boldsymbol{U}(t(\tau)).$$

Then we obtain the equivalent transformed problem

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\overline{\boldsymbol{x}}(\tau) = t_{\mathrm{f}}\boldsymbol{f}(\overline{\boldsymbol{x}}(\tau), \overline{\boldsymbol{U}}(\tau)).$$

For the sake of clarity we use $\boldsymbol{x}(t)$ and $\boldsymbol{U}(t)$ instead of $\overline{\boldsymbol{x}}(\tau)$ and $\overline{\boldsymbol{U}}(\tau)$ in the remainder of this section. The model parameters are given in Table 15.1. There are control box constraints

$$-U_{\max} \leqslant \boldsymbol{U}(t) \leqslant +U_{\max}$$

and initial as well as terminal state constraints

$$\boldsymbol{x}(0) = [0, 0, 0, 0, 0]^T, \qquad \boldsymbol{x}(1) = [0.01, 0, 0.01, 0, 0]^T.$$

We consider the minimal time cost functional

$$\min t_{\mathrm{f}},$$

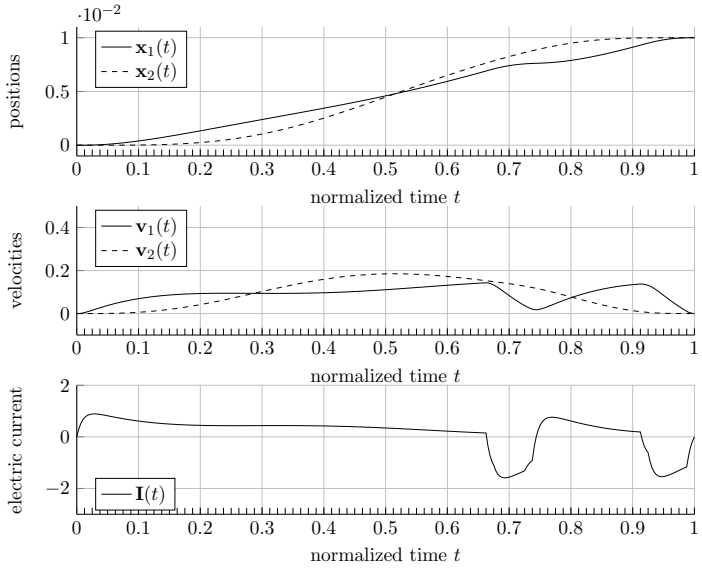
and apply our approach to the resulting OCP. A switch is induced by the Coulombic friction force $-F_R\ \mathrm{sgn}(\boldsymbol{v}_1(t))$. The right hand side discontinuity is a consistent switch. Therefore we distinguish the two cases $-F_R$, if $\boldsymbol{v}_1(t) \geqslant 0$ and $+F_R$, if $\boldsymbol{v}_1(t) \leqslant 0$ for the Coulombic friction. We model this by introducing additional controls $\boldsymbol{\omega}(t) = [\boldsymbol{\omega}_{\geqslant}(t), \boldsymbol{\omega}_{\leqslant}(t)]^T$, $\boldsymbol{\omega}_{\geqslant}(t), \boldsymbol{\omega}_{\leqslant}(t) \in \{0, 1\}$. Then the Coulombic friction force can be written as $-F_R(\boldsymbol{\omega}_{\geqslant}(t) - \boldsymbol{\omega}_{\leqslant}(t))$ if the two implications

$$\begin{aligned}
[\boldsymbol{\omega}_{\geqslant}(t) = 1 \implies \boldsymbol{v}_1(t) \geqslant 0] \quad &\iff \quad [-\boldsymbol{v}_1(t)\,\boldsymbol{\omega}_{\geqslant}(t) \leqslant 0], \\
[\boldsymbol{\omega}_{\leqslant}(t) = 1 \implies \boldsymbol{v}_1(t) \leqslant 0] \quad &\iff \quad [+\boldsymbol{v}_1(t)\,\boldsymbol{\omega}_{\leqslant}(t) \leqslant 0],
\end{aligned}$$

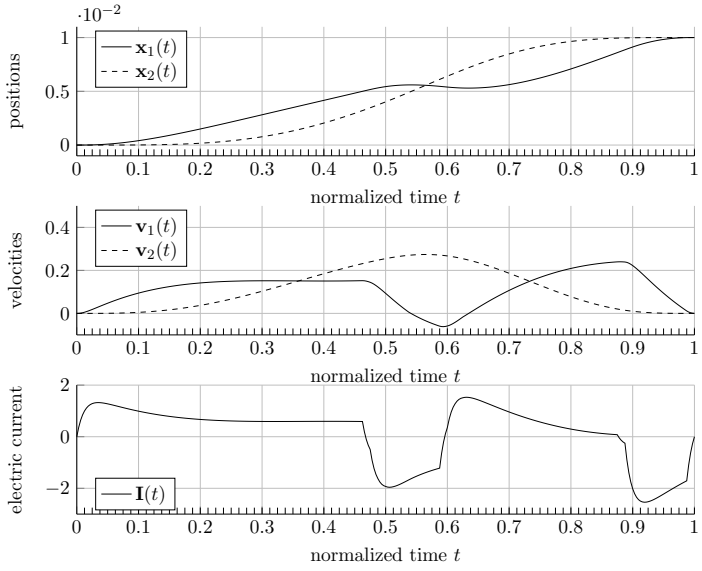
and the SOS–1 constraint $\boldsymbol{\omega}_{\leqslant}(t) + \boldsymbol{\omega}_{\geqslant}(t) = 1$ hold. In the next step we replace the binary variables $\boldsymbol{\omega}_{\geqslant}(t), \boldsymbol{\omega}_{\leqslant}(t)$ by convexified control variables $\boldsymbol{\alpha}_{\geqslant}(t), \boldsymbol{\alpha}_{\leqslant}(t) \in [0, 1]$. The resulting

Table 15.1: Parameters of the coulombic friction model

Physical quantity	Identifier	Value	Unit
Coil resistance	R	2	Ω
Coil inductivity	L	2	mH
Force constant	K_F	12	N/A
Voltage constant	K_S	12	Vs/m
Motor mass (slider, guide, coil)	m_1	1.03	kg
Load mass	m_2	0.56	kg
Spring constant	k	2.4	kN/m
Guide friction force	F_R	2.1	N



a) Case: $U_{\max} = 2$.



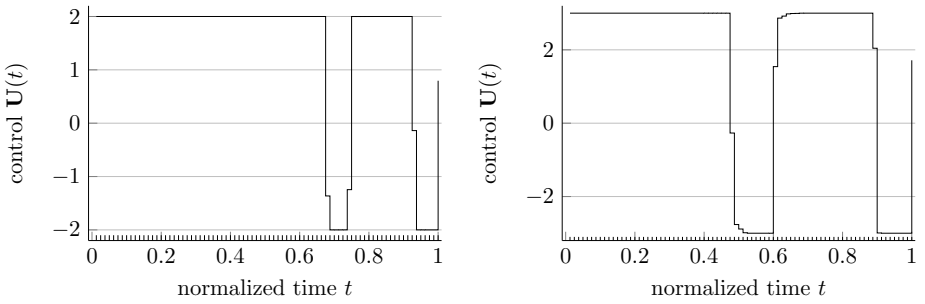
b) Case: $U_{\max} = 3$.

Figure 15.1: State trajectories of the coulombic friction model from Christiansen et al. [110] with two different values for control bounds $U_{\max}$ and minimal time objective functional. Compared to Christiansen et al. [110] all trajectories show structurally the same behavior.

OCP now reads as

$$
\begin{aligned}
\min_{t_f, \boldsymbol{x}(\cdot), \boldsymbol{U}(\cdot), \boldsymbol{\alpha}_{\geqslant}(\cdot), \boldsymbol{\alpha}_{\leqslant}(\cdot)} \quad & t_f \\
\text{s.t.} \quad & \dot{\boldsymbol{x}}_1(t) = t_f \boldsymbol{v}_1(t), \\
& \dot{\boldsymbol{v}}_1(t) = t_f \left(K_F \boldsymbol{I}(t) - k(\boldsymbol{x}_1(t) - \boldsymbol{x}_2(t)) - F_R(\boldsymbol{\alpha}_{\geqslant}(t) - \boldsymbol{\alpha}_{\leqslant}(t)) \right) / m_1, \\
& \dot{\boldsymbol{x}}_2(t) = t_f \boldsymbol{v}_2(t), \\
& \dot{\boldsymbol{v}}_2(t) = t_f k(\boldsymbol{x}_1(t) - \boldsymbol{x}_2(t)) / m_2, \\
& \dot{\boldsymbol{I}}(t) = t_f \left(\boldsymbol{U}(t) - R \boldsymbol{I}(t) - K_S \boldsymbol{v}_1(t) \right) / L, \\
& \boldsymbol{x}(0) = [0,0,0,0,0]^T, \ \boldsymbol{x}(1) = [0.01,0,0.01,0,0]^T, \\
& 0 \geqslant -\boldsymbol{v}_1(t) \boldsymbol{\alpha}_{\geqslant}(t), \\
& 0 \geqslant +\boldsymbol{v}_1(t) \boldsymbol{\alpha}_{\leqslant}(t), \\
& 1 = \boldsymbol{\alpha}_{\geqslant}(t) + \boldsymbol{\alpha}_{\leqslant}(t), \quad \boldsymbol{\alpha}_{\geqslant}(t), \boldsymbol{\alpha}_{\leqslant}(t) \in [0,1].
\end{aligned} \tag{15.1a}
$$

Finally we eliminate $\boldsymbol{\alpha}_{\leqslant}(t)$: due to equation (15.1a) this can easily be done by replacing $\boldsymbol{\alpha}_{\leqslant}(t)$ with $1 - \boldsymbol{\alpha}_{\geqslant}(t)$ in all equations of problem (15.1). Equation (15.1a) then reduces to $\boldsymbol{\alpha}_{\geqslant}(t) \in [0,1]$.



a) Case: $U_{\max} = 2$. b) Case: $U_{\max} = 3$.

Figure 15.2: Control profiles for two different values of $U_{\max}$ of the coulombic friction model from Christiansen et al. [110]. Compared to Christiansen et al. [110] we see the same structural behavior. Due to lack of refinement steps the bang-bang control in both plots is not fully pronounced.

Then we discretize the OCP as described in Section 11.6 with 80 equidistant finite elements, polynomial order 2 for states and polynomial order 0 for control $\boldsymbol{U}$. The technique described in Section 11.8 is applied to solve the resulting MPVC. We choose $\tau_0 = 10^{-3}$ as the initial regularization parameter. The regularization parameter is reduced in each iteration according to the rule $\tau_k = \sqrt{0.9}\, \tau_{k-1}$, $k \geqslant 1$. The arising NLPs are solved with the sparse SQP solver SNOPT [197] without adaptive mesh refinement. We apply the default settings of SNOPT.
Figure 15.1 shows the resulting state trajectories for chosen control bounds $U_{\max} = 2$ and $U_{\max} = 3$ returned by SNOPT. One can see that the state trajectories show structurally the

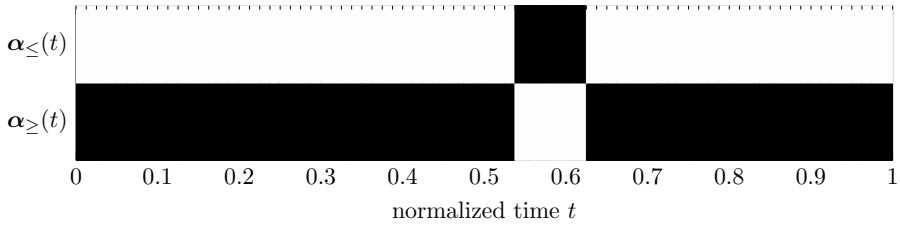


Figure 15.3: Mode switching profile for control bound choice $U_{\max} = 3$ of the coulombic friction model from Christiansen et al. [110]. One can see the chosen finite element grid indicated by ticks on the upper axis of the plot. The grayscale colors associated with each mode range from white (mode is not active) to black (mode is active). The plot shows that in the beginning and in the end mode $\boldsymbol{\alpha}_{\geqslant}$ is active. In between mode $\boldsymbol{\alpha}_{\leqslant}$ is active. This result coincides with the plot in Figure 15.1b where the sign of $\boldsymbol{v}_1$ matches with chosen modes in this plot.

same behavior compared to the results of Christiansen et al. [110]. We cannot compare the results in detail because the value for parameter K_S is not specified in the article of Christiansen et al. [110]. In Figure 15.2 the control trajectory of $\boldsymbol{U}$ is depicted for both control bound choices. Figure 15.3 provides for control bound $U_{\max} = 3$ a view of the chosen modes represented by the control trajectories of $\boldsymbol{\alpha}_{\leqslant}$ and $\boldsymbol{\alpha}_{\geqslant}$. For case $U_{\max} = 2$ we are permanently in mode $\boldsymbol{\alpha}_{\geqslant} = 1$ and $\boldsymbol{\alpha}_{\geqslant} = 0$. Therefore we do not show the corresponding plot.

15.2 A Stick-Slip Model with Known Switching Point

We consider the following inconsistently switched model

$$\dot{\boldsymbol{x}}(t) = \begin{bmatrix} \boldsymbol{x}_2(t) \\ -\dfrac{k}{m}\boldsymbol{x}_1(t) + \dfrac{\boldsymbol{F}(\boldsymbol{x}_1(t), \boldsymbol{v}_{\text{rel}}(t))}{m} \end{bmatrix}, \tag{15.2}$$

where $\boldsymbol{x}(t) = [\boldsymbol{x}_1(t), \boldsymbol{x}_2(t)]^T$ and $\boldsymbol{v}_{\text{rel}}(t) = \boldsymbol{x}_2(t) - v_b$. The friction force $\boldsymbol{F}$ is a function of the relative velocity $\boldsymbol{v}_{\text{rel}}$ in the slip phase and a function of the spring force kx in the stick phase:

$$\boldsymbol{F}(x, v_{\text{rel}}) = \begin{cases} \min(|kx|, F_s)\,\operatorname{sgn}(kx), & v_{\text{rel}} = 0 \qquad \text{stick}, \\ -F_s\, v_{\text{rel}}, & v_{\text{rel}} \neq 0 \qquad \text{slip} \end{cases} \tag{15.3}$$

A simple calculation shows that starting from the initial value $\boldsymbol{x}(0) = [0, 1]^T$, the first switch from the sticking mode to the slipping mode occurs at time $t = F_s/k$.
We apply our approach to the initial value problem: first we replace the stick mode triggering constraint $\boldsymbol{v}_{\text{rel}}(t) = 0$ by a relaxed constraint $-\varepsilon \leqslant \boldsymbol{v}_{\text{rel}}(t) \leqslant \varepsilon$. The stick-slip branches, the $\operatorname{sgn}(\cdot)$, $\min(\cdot,\cdot)$ and the $|\cdot|$ functions in (15.3) induce state dependent switches. One identifies five modes in the model if all switches are taken into account. Therefore we introduce controls

$\boldsymbol{\alpha}(t) = [\boldsymbol{\alpha}_1(t), \ldots, \boldsymbol{\alpha}_5(t)]^T$, $\boldsymbol{\alpha}_i(t) \in [0,1]$, $i = 1, \ldots, 5$. The five modes are characterized by the following implications:

$$\boldsymbol{\alpha}_1(t) = 1 \quad \Longrightarrow \quad -\varepsilon \leqslant \boldsymbol{v}_{\text{rel}}(t) \leqslant +\varepsilon \quad \wedge \quad |k\boldsymbol{x}_1(t)| \leqslant F_s \tag{15.4}$$

$$\boldsymbol{\alpha}_2(t) = 1 \quad \Longrightarrow \quad -\varepsilon \leqslant \boldsymbol{v}_{\text{rel}}(t) \leqslant +\varepsilon \quad \wedge \quad k\boldsymbol{x}_1(t) \leqslant -F_s \tag{15.5}$$

$$\boldsymbol{\alpha}_3(t) = 1 \quad \Longrightarrow \quad -\varepsilon \leqslant \boldsymbol{v}_{\text{rel}}(t) \leqslant +\varepsilon \quad \wedge \quad k\boldsymbol{x}_1(t) \geqslant +F_s \tag{15.6}$$

$$\boldsymbol{\alpha}_4(t) = 1 \quad \Longrightarrow \quad \boldsymbol{v}_{\text{rel}}(t) \geqslant +\varepsilon \tag{15.7}$$

$$\boldsymbol{\alpha}_5(t) = 1 \quad \Longrightarrow \quad \boldsymbol{v}_{\text{rel}}(t) \leqslant -\varepsilon \tag{15.8}$$

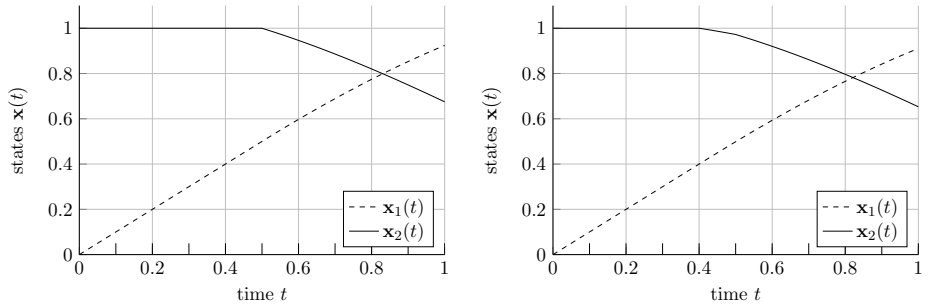
The reformulated model equations (15.2)-(15.3), the vanishing constraint formulations of implications (15.4)-(15.8) and the convexified SOS–1 constraint $\boldsymbol{\alpha}(t) \in \text{conv}(\mathbb{S}^5)$ yield the following feasibility problem:

$$\begin{aligned}
&\text{find} && \boldsymbol{x}(\cdot),\ \boldsymbol{\alpha}(\cdot) \\
&\text{s.t.} && \dot{\boldsymbol{x}}_1(t) = \boldsymbol{x}_2(t), \\
& && \dot{\boldsymbol{x}}_2(t) = \big((\boldsymbol{\alpha}_1(t) - 1)\, k\, \boldsymbol{x}_1(t) + F_s\, \boldsymbol{v}_{\text{rel}}(t)\, (\boldsymbol{\alpha}_3(t) - \boldsymbol{\alpha}_2(t))\big)/m, \\
& && \boldsymbol{x}(0) = [0,1]^T, \\
&(15.4) \Leftrightarrow && \mathbf{0} \geqslant [-\varepsilon + \boldsymbol{v}_{\text{rel}}(t), -\varepsilon - \boldsymbol{v}_{\text{rel}}(t), -F_s + k\,\boldsymbol{x}_1(t), -F_s - k\,\boldsymbol{x}_1(t)]^T\, \boldsymbol{\alpha}_1(t), \\
&(15.5) \Leftrightarrow && \mathbf{0} \geqslant [-\varepsilon + \boldsymbol{v}_{\text{rel}}(t), -\varepsilon - \boldsymbol{v}_{\text{rel}}(t), +F_s + k\,\boldsymbol{x}_1(t)]^T\, \boldsymbol{\alpha}_2(t), \\
&(15.6) \Leftrightarrow && \mathbf{0} \geqslant [-\varepsilon + \boldsymbol{v}_{\text{rel}}(t), -\varepsilon - \boldsymbol{v}_{\text{rel}}(t), +F_s - k\,\boldsymbol{x}_1(t)]^T\, \boldsymbol{\alpha}_3(t), \\
&(15.7) \Leftrightarrow && 0 \geqslant [+\varepsilon - \boldsymbol{v}_{\text{rel}}(t)]\, \boldsymbol{\alpha}_4(t), \\
&(15.8) \Leftrightarrow && 0 \geqslant [+\varepsilon + \boldsymbol{v}_{\text{rel}}(t)]\, \boldsymbol{\alpha}_5(t), \\
& && 1 = \sum\nolimits_{j=1}^{5} \boldsymbol{\alpha}_j(t), \quad \boldsymbol{\alpha}_i(t) \in [0,1], \quad i = 1, \ldots, 5.
\end{aligned}$$

The feasibility problem is discretized with the techniques from Section 11.6. We choose an equidistant grid with 10 finite elements and a polynomial order 2 for the state approximation polynomials. Model parameters and ε assignment are given in Table 15.2. One can see that we choose two different values for parameter F_s. For $F_s = 0.5$ the switching point of the system is a finite element grid point whereas for $F_s = 0.45$ the switching point is in the middle of the

Table 15.2: Parameters of the stick-slip model

Physical quantity	Identifier	Value	Unit
Spring constant	k	1.0	Nm
Mass	m	1.0	kg
Relative belt vel.	v_b	1.0	m/s
Max. stat. friction force	F_s	0.45,0.5	N
Relaxation parameter	ε	10^{-15}	-



a) Case: $F_s = 0.5$. The exact switching point at time $t = 0.5$ is part of the finite element grid. Our approach detects the switch exactly, as indicated by the kink of the $\boldsymbol{x}_2$ trajectory.

b) Case: $F_s = 0.45$. The exact switching point at time $t = 0.45$ is not part of the finite element grid. Our approach detects the switch too early indicated by the kink of the $\boldsymbol{x}_2$ trajectory at time $t = 0.4$.

Figure 15.4: State trajectories of the stick-slip model for two different assignments for parameter F_s. In both cases we choose 10 equidistant finite elements.

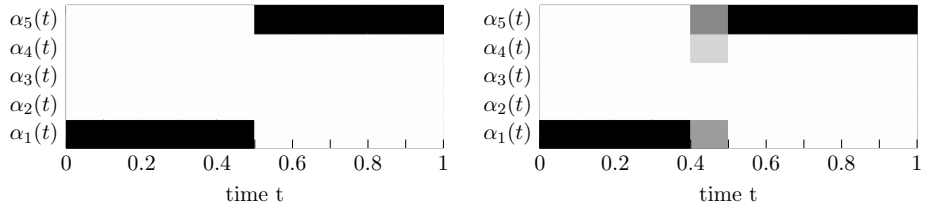
finite element interval $[0.4, 0.5]$. The two discretized systems are then solved with SNOPT. In both cases default options for SNOPT are chosen. SNOPT can solve the system without the regularization approach to handle the vanishing constraints.

Figure 15.4 shows the resulting state trajectories for $F_s = 0.45$ and $F_s = 0.5$. Analogously Figure 15.5 depicts the chosen modes for both cases.

As can be seen in Figure 15.4b, in the case $F_s = 0.45$ our approach detects a switch from the stick to the slip mode at time $t = 0.4$ indicated by the kink of the $\boldsymbol{x}_2$ trajectory at time $t = 0.4$. According to the analytical solution the switch actually occurs at time $t = F_s/k = 0.45$. In this case we cannot expect the switching time to be detected at the analytically correct time, because the time $t = 0.45$ is not part of the finite element grid. Figure 15.5b illustrates this: apart from the switch point including finite element interval $[0.4, 0.5]$ there is either mode 1 ($\boldsymbol{\alpha}_1(t) = 1$) or mode 5 ($\boldsymbol{\alpha}_5(t) = 1$) active. In $[0.4, 0.5]$ fractions of mode 1, 4 and 5 are active.

In the case of $F_s = 0.5$ our approach detects the first switch at time $t = 0.5$. This can either be seen in Figure 15.4a indicated by the kink of the $\boldsymbol{x}_2$ trajectory at time $t = 0.5$ or by the mode switching event from finite element interval $[0.4, 0.5]$ to $[0.5, 0.6]$ illustrated in Figure 15.5a. The formula $t = F_s/k = 0.5$ confirms the switching point for the analytical solution. The accordance of predicted and analytical switching time could be explained by the fact that the switching time is part of the finite element grid.

This example makes clear that it is indispensable for our approach to develop reliable refinement strategies. Only if one can detect switching points up to a certain level of accuracy reliable solutions can be expected. But our experiments also give rise to hopes that the values of the mode controls $\boldsymbol{\alpha}$ give good indications in which finite element intervals the switches are located.



a) Case: $F_s = 0.5$. The analytical switching point from mode 1 ($\boldsymbol{\alpha}_1(t) = 1$) to mode 5 ($\boldsymbol{\alpha}_1(t) = 1$) is at time $t = 0.5$. This can also be seen in the plot.

b) Case: $F_s = 0.45$. The analytical switching point from mode 1 ($\boldsymbol{\alpha}_1(t) = 1$) to mode 5 ($\boldsymbol{\alpha}_1(t) = 1$) is at time $t = 0.45$. The plot shows that in the finite element interval [0.4,0.5] where the switch takes place, there are three partially active modes.

Figure 15.5: Element-wise partitions indicating the active modes of the stick-slip model for two different assignments of parameter F_s. In both cases we choose 10 equidistant finite elements indicated by ticks in both plots. The grayscale colors associated with each mode range from white (mode is not active) to black (mode is active).

15.3 An Alternate Friction Model

We consider the inconsistently switched model in Leine et al. [288] and Leine et al. [289] where a mass m is attached to an inertial space by a spring with spring constant k. The mass is sliding on a driving belt. The belt is moving at constant velocity v_b. Dry friction with a friction force F occurs between mass and belt. The model equations read as

$$\dot{\boldsymbol{x}}(t) = \begin{bmatrix} \boldsymbol{x}_2(t) \\ -\frac{k}{m}\boldsymbol{x}_1(t) + \frac{\boldsymbol{F}(\boldsymbol{x}_1(t), v_{\text{rel}}(t))}{m} \end{bmatrix}, \tag{15.9}$$

where $\boldsymbol{x}(t) = [\boldsymbol{x}_1(t), \boldsymbol{x}_2(t)]^T$ and $\boldsymbol{v}_{\text{rel}}(t) = \boldsymbol{x}_2(t) - v_b$. The friction force $\boldsymbol{F}$ is a function of the relative velocity $\boldsymbol{v}_{\text{rel}}$ in the slip phase and a function of the spring force kx in the stick phase

$$\boldsymbol{F}(x, v_{\text{rel}}) = \begin{cases} \min(|kx|, F_s)\,\operatorname{sgn}(kx), & v_{\text{rel}} = 0 \quad \text{stick}, \\ -\frac{F_s \operatorname{sgn}(v_{\text{rel}})}{1+\delta\,|v_{\text{rel}}|}, & v_{\text{rel}} \neq 0 \quad \text{slip}. \end{cases} \tag{15.10}$$

The model parameters are depicted in Table 15.3. We solve the initial value problem with initial values $\boldsymbol{x}(0) = [1.133944669704, 0]^T$ on the horizon $\mathcal{T} = [0, 12]$.

State dependent switches are induced by the stick-slip branches, $\operatorname{sgn}(\cdot)$, $\min(\cdot,\cdot)$ and $|\cdot|$ functions in (15.10). In the following, we apply our approach: to this end we replace the stick mode triggering constraint $\boldsymbol{v}_{\text{rel}}(t) = 0$ by a relaxed constraint $-\varepsilon \leqslant \boldsymbol{v}_{\text{rel}}(t) \leqslant \varepsilon$. We then identify five modes in the model and introduce controls $\boldsymbol{\alpha}(t) = [\boldsymbol{\alpha}_1(t), \ldots, \boldsymbol{\alpha}_5(t)]^T$, $\boldsymbol{\alpha}_i(t) \in [0, 1]$. The five modes are characterized by the implications

$$\boldsymbol{\alpha}_1(t) = 1 \quad \Longrightarrow \quad -\varepsilon \leqslant \boldsymbol{v}_{\text{rel}}(t) \leqslant +\varepsilon \quad \wedge \quad |k\boldsymbol{x}_1(t)| \leqslant F_s, \tag{15.11}$$

$$\boldsymbol{\alpha}_2(t) = 1 \quad \Longrightarrow \quad -\varepsilon \leqslant \boldsymbol{v}_{\text{rel}}(t) \leqslant +\varepsilon \quad \wedge \quad k\boldsymbol{x}_1(t) \leqslant -F_s, \tag{15.12}$$

$$\boldsymbol{\alpha}_3(t) = 1 \quad \Longrightarrow \quad -\varepsilon \leqslant \boldsymbol{v}_{\text{rel}}(t) \leqslant +\varepsilon \quad \wedge \quad k\boldsymbol{x}_1(t) \geqslant +F_s, \tag{15.13}$$

$$\boldsymbol{\alpha}_4(t) = 1 \quad \Longrightarrow \quad \boldsymbol{v}_{\text{rel}}(t) \geqslant +\varepsilon, \tag{15.14}$$

$$\boldsymbol{\alpha}_5(t) = 1 \quad \Longrightarrow \quad \boldsymbol{v}_{\text{rel}}(t) \leqslant -\varepsilon. \tag{15.15}$$

The new formulated model equations (15.9)-(15.10), the mode characterizing implications

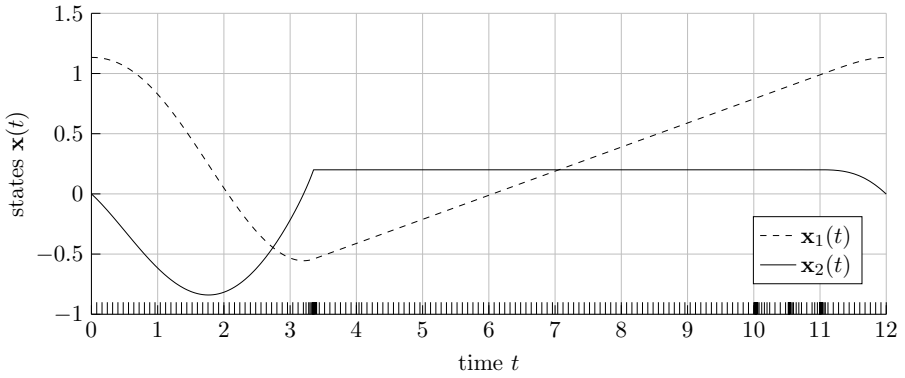


Figure 15.6: State trajectories of the switched alternate friction model (see Leine et al. [288] and Leine et al. [289]). The black tick marks on the lower axis indicate the element boundaries, which accumulate at the switch points to accurately resolve the switching times.

(15.11)-(15.15) and the convexified SOS–1 constraint $\boldsymbol{\alpha}(t) \in \operatorname{conv}(\mathbb{S}^5)$ result in the following feasibility problem:

$$\begin{aligned}
&\text{find} && \boldsymbol{x}(\cdot),\ \boldsymbol{\alpha}(\cdot) \\
&\text{s.t.} && \dot{\boldsymbol{x}}_1(t) = \boldsymbol{x}_2(t), \\
& && \dot{\boldsymbol{x}}_2(t) = \boldsymbol{f}_2(\boldsymbol{x}(t), \boldsymbol{\alpha}(t)), \\
& && \boldsymbol{x}(0) = [1.133944669704, 0]^T,
\end{aligned}$$

Table 15.3: Parameters of the alternate friction model

Physical quantity	Identifier	Value	Unit
Spring constant	k	1.0	Nm
Mass	m	1.0	kg
Relative belt vel.	v_b	0.2	m/s
Max. stat. friction force	F_s	1.0	N
Phys. constant	δ	3.0	s/m
Relaxation parameter	ε	10^{-15}	-

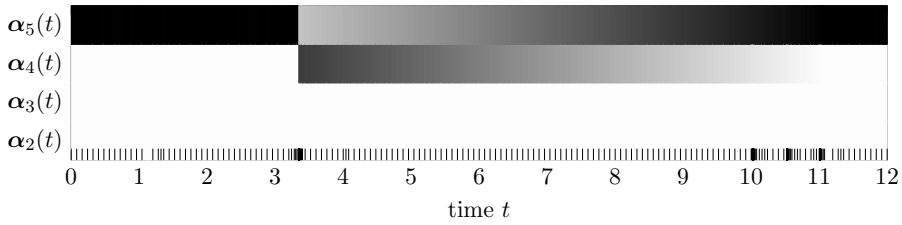


Figure 15.7: Element-wise partitions indicating the active modes in the alternate friction model (see Leine et al. [288] and Leine et al. [289]). The grayscale colors associated with each mode range from white (mode is not active) to black (mode is active). The plot shows that in the beginning and in the end mode 5 is active. In between there is a Filippov solution realized by convex combinations of modes 4 and 5. The black tick marks on the lower axis indicate the element boundaries, which accumulate at the switch points to accurately resolve the switching times.

$$
\begin{aligned}
(15.11) \Leftrightarrow \quad & \mathbf{0} \geqslant [-\varepsilon + \boldsymbol{v}_{\mathrm{rel}}(t), -\varepsilon - \boldsymbol{v}_{\mathrm{rel}}(t), -F_s + k\,\boldsymbol{x}_1(t), -F_s - k\,\boldsymbol{x}_1(t)]^T\, \boldsymbol{\alpha}_1(t), \\
(15.12) \Leftrightarrow \quad & \mathbf{0} \geqslant [-\varepsilon + \boldsymbol{v}_{\mathrm{rel}}(t), -\varepsilon - \boldsymbol{v}_{\mathrm{rel}}(t), +F_s + k\,\boldsymbol{x}_1(t)]^T\, \boldsymbol{\alpha}_2(t), \\
(15.13) \Leftrightarrow \quad & \mathbf{0} \geqslant [-\varepsilon + \boldsymbol{v}_{\mathrm{rel}}(t), -\varepsilon - \boldsymbol{v}_{\mathrm{rel}}(t), +F_s - k\,\boldsymbol{x}_1(t)]^T\, \boldsymbol{\alpha}_3(t), \\
(15.14) \Leftrightarrow \quad & 0 \geqslant [+\varepsilon - \boldsymbol{v}_{\mathrm{rel}}(t)]\, \boldsymbol{\alpha}_4(t), \\
(15.15) \Leftrightarrow \quad & 0 \geqslant [+\varepsilon + \boldsymbol{v}_{\mathrm{rel}}(t)]\, \boldsymbol{\alpha}_5(t), \\
& 1 = \sum\nolimits_{j=1}^{5} \boldsymbol{\alpha}_j(t), \quad \boldsymbol{\alpha}_i(t) \in [0,1], \quad i = 1, \ldots, 5,
\end{aligned}
\tag{15.16a}
$$

where

$$
f_2(x, \alpha) = \frac{(\alpha_1 - 1)\, k\, x_1 + F_s \left(\alpha_3 - \alpha_2 + \dfrac{\alpha_5}{1 - \delta\, v_{\mathrm{rel}}} - \dfrac{\alpha_4}{1 + \delta\, v_{\mathrm{rel}}} \right)}{m}. \tag{15.17}
$$

We apply the discretization strategy described in Section 11.6, where we choose an equidistant finite element grid with 150 elements and polynomial order 3 to approximate the states. Before solving the arising MPVCs, we investigate the problem further for the case $\varepsilon = 0$: based on physical insight, we expect that there is an interval with $\boldsymbol{v}_{\mathrm{rel}}(t) = 0$. If we set $\boldsymbol{v}_{\mathrm{rel}}(t) = 0$ in (15.17) and assume that mode 1 is active ($\boldsymbol{\alpha}_1(t) = 1,\ \boldsymbol{\alpha}_2(t), \ldots, \boldsymbol{\alpha}_5(t) = 0$) then $\boldsymbol{f}_2(\boldsymbol{x}(t), \boldsymbol{\alpha}(t))$ is equal to zero. On the other hand if we set $\boldsymbol{v}_{\mathrm{rel}}(t) = 0$ and assume $|k\boldsymbol{x}_1(t)| \leqslant F_s$, we can use (15.16a) to eliminate the explicit appearance of $\boldsymbol{\alpha}_1$ to obtain

$$
\begin{aligned}
0 &= \frac{-k\,\boldsymbol{x}_1(t)\,(\boldsymbol{\alpha}_3(t) + \boldsymbol{\alpha}_2(t) + \boldsymbol{\alpha}_5(t) + \boldsymbol{\alpha}_4(t)) + F_s \left(\boldsymbol{\alpha}_3(t) - \boldsymbol{\alpha}_2(t) + \boldsymbol{\alpha}_5(t) - \boldsymbol{\alpha}_4(t)\right)}{m} \\
&= \frac{+F_s - k\,\boldsymbol{x}_1(t)}{m}(\boldsymbol{\alpha}_3(t) + \boldsymbol{\alpha}_5(t)) + \frac{-F_s - k\,\boldsymbol{x}_1(t)}{m}(\boldsymbol{\alpha}_2(t) + \boldsymbol{\alpha}_4(t)),
\end{aligned}
\tag{15.18}
$$

which is a linear combination of the right hand side terms corresponding to modes 4 and 5.

This observation yields the link to Filippov solutions of the switched system (15.9)–(15.10). To this end, we reformulate the system in the sense of Filippov by only describing the dynamics of the two slip modes ($v_{\text{rel}} > 0$ and $v_{\text{rel}} < 0$). This can be achieved by replacing $\boldsymbol{F}$ with

$$\hat{\boldsymbol{F}}(x, v_{\text{rel}}) = \begin{cases} -\dfrac{F_s}{1+\delta v_{\text{rel}}}, & v_{\text{rel}} \geq 0, \\ +\dfrac{F_s}{1-\delta v_{\text{rel}}}, & v_{\text{rel}} \leq 0, \end{cases} \tag{15.19}$$

which is multi-valued in $v_{\text{rel}} = 0$. We then construct Filippov solutions for the stick phase ($v_{\text{rel}} = 0$) by a convex combination of the right hand side traces $-F_s$ and $+F_s$ on the switch manifold $v_{\text{rel}} = 0$. Apparently, the Filippov approach coincides with our approach only if the linear combination coefficients $\boldsymbol{\alpha}_3 + \boldsymbol{\alpha}_5$ and $\boldsymbol{\alpha}_2 + \boldsymbol{\alpha}_4$ in (15.18) add up to one. This implies $\boldsymbol{\alpha}_1 \equiv 0$. In fact, we can explicitly enforce Filippov solutions by bounding $\boldsymbol{\alpha}_1 \leq \tau \to 0$ within the homotopy approach to obtain satisfactory numerical solutions.

After discretization we apply the MPVC relaxation algorithm from Section 11.7 and the techniques described in Section 11.8 to the resulting finite dimensional feasibility problem with vanishing constraints. We choose $\tau_0 = 10^{-3}$ as initial regularization parameter and update it according to the rule $\tau_k = \sqrt{0.9}\,\tau_{k-1}$, $k \geqslant 1$. The arising NLPs are solved with the SQP solver SNOPT.

Figure 15.6 depicts the state trajectories resulting from our calculations. As can be seen in Figure 15.7 our approach detects a switch from the slipping mode to the sticking mode and back. The refinement scheme is important to accurately determine the switching points.

Conclusion and Outlook

In the following, we give an overview of the broad range of topics which have been covered in this thesis. We discuss our achievements and shed some light on various aspects of our work that offer the potential for further research.

Multi–Degree Collocation

In Chapter 7, we proposed a pseudospectral collocation method which, in contrast to previous methods of this type, allows to set distinct polynomial degrees for the approximating polynomials of all differential state and control components. We illustrated how to exploit the specific structure of the resulting Jacobians and Hessians. We discussed the trade–off and valuation between a mostly uniform or a mostly distinct polynomial degree environment and the consequences on the resulting problem size of the discretized problem as well as the running time to solve the discretized problem.
The utility of the proposed approach could be demonstrated in two ways. First, we applied it to tailored benchmark problems in Chapter 12, where the setting with distinct polynomial degrees could outperform the uniform setting in terms of run–time while maintaining the quality of the solution. Second, we could use the approach to realize our novel approach to the numerical solution of explicitly and implicitly switched OCPs, cf. Chapter 11.
Opportunities for further research are to realize an efficient implementation of the approach, in particular with respect to the potential to boost its performance by a parallelization of the code. Furthermore, the approach has to prove its potential when it is applied to real–world problem instances. One might also think of extending the theory for costate estimation (Chapter 9) and error estimation (Chapter 10) to the multi–degree case. It would also be quite appealing to develop heuristics for an efficient automatic choice of the polynomial degrees.

Costate Estimation

In Chapter 9, we have found that a specific pseudospectral collocation method, which is based on Legendre–Gauss–Radau points as collocation points, can be interpreted as a tailored Finite Element Petrov–Galerkin approach when applied to the weak formulation of the equations of the local minimum principle. Demonstrating the equivalence of the "first discretize, then optimize" approach on the one hand and the *'first optimize, then discretize'* approach on the other hand also enabled us to derive interpretations for the NLP multipliers coming from the collocation approach as approximate solutions to the costates of the local minimum principle. In particular, this includes estimates for the costates of the differential equations, the boundary constraints, and mixed control–state constraints. The validity of our calculations was confirmed by executing suitable numerical experiments as described in Chapter 13.

Pseudospectral methods are generally considered to be very efficient and robust. Their equivalence with a FE Petrov–Galerkin approach might introduce an untrodden way to solve OCPs in the future. As Kunkel and Gerdts [282] and Gerdts and Kunkel [191] have done in the past for a local one–step method (see Section 6.3.1) we could do with the Petrov–Galerkin method. Here, the complementarity condition of the local minimum principle is replaced with a NCP function formulation leading to a nonlinear and non–smooth equation. The research on semismooth Newton methods has been strongly evolving recently and provides a broad arsenal of powerful algorithms to solve this equation.
The same NCP function reformulation of the necessary optimality conditions might even be helpful to show certain convergence results for LGR pseudospectral methods as Gerdts and Kunkel [192] could analyze convergence properties of the Euler discretization when applied to discretize OCPs with mixed control–state constraints.

Goal–Oriented Error Estimation

In Chapter 10, we derived a formula which allows to express the error between an approximate OCP solution coming from a FE Petrov–Galerkin approach and the exact solution with respect to the performance criterion. Based on this error estimation, we developed an adaptive mesh refinement strategy where the discretization scheme can be updated with regard to both FE grid and the degree of the approximating polynomials. We demonstrated its good performance for several OCPs of different type in Chapter 14.
In the future, there is some need to extend the described mesh adaption algorithm for coarsening the FE grid. Furthermore, we should implement strategies to decrease the approximating polynomial degrees. One might also think of fundamentally different ideas for grid refinement such as strategies that are based on an equilibration of the error. This is possible due to the error representation as element–wise contributions.
The error estimation should allow for general goals not just the error in the performance criterion. There exist several strategies for that in the PDE literature which could act as a good starting point. Furthermore, one has to take account for the difficulties to estimate the error of pure state constrained OCP and develop suitable strategies.

Switched Optimal Control

We proposed an approach how explicitly and implicitly switched OCPs can be solved numerically in a unified way in Chapter 11. In a first step, we described a technique based on generalized disjunctive programming how to equivalently reformulate an implicitly switched OCP into a counterpart problem wherein those switches loose their implicit character. Instead, discrete decision variables and vanishing constraints enter the new problem. Recent results justify to omit the integrality constraints which finally leads to the task to solve a continuous OCP with vanishing constraints. We described some strategies to efficiently solve those problems.
We have several ideas how our approach to switched optimal control can be enhanced or enriched. One could develop better heuristics to detect the different switching types (consistent

switches, sliding mode). In case there is a sliding mode over a certain time interval one could use the multi–degree availability of our collocation method (see Chapter 7) to switch from piecewise constant to piecewise linear control approximations. From a theoretical perspective, it would be interesting to find out if the convex multiplier functions $\hat{\boldsymbol{\alpha}}(\cdot)$ (see (11.16)) are related to the respective functions of the Filippov theory (see Section 1.3). Currently, there is some ongoing research to augment the approach such that it can handle OCPs with state jumps, cf. Kirches et al. [276, 277]. Applying the approach within a Multiple Shooting environment is an obvious idea. We would also like to combine the approach with our goal–oriented error estimation since both SNLP approaches can be easily combined. A potential extension of our error estimation to arbitrary goals could allow us not to adapt the grid with respect to the performance criterion but to the switching function. This could improve the detection of the implicitly given switching structure. Finally, the homotopy parameter to control the embedded MPVC algorithm should be driven to zero problem specifically and not just with a constant decrease rate. Since the homotopy in our approach reminds of following the central path in interior–point algorithms, one could think of borrowing some successful techniques such as Mehrotra's probing procedure or the Fiacco–McCormick approach.

Nonlinear Model Predictive Control

In this contribution, we have considered OCPs with explicit and implicit switches as offline problems. This type of optimal control is called *open–loop* optimal control. In contrast to this, there exists the *closed–loop* optimal control approach, where a sequence of related OCPs is solved. For our future research we are planning to extend the theory developed in Chapter 11 to closed–loop problems, and in particular applied in an NMPC context.
The principles of NMPC are described in Appendix C. A promising approach to realize a concrete NMPC algorithm can be found in the RTI approach (see Appendix C.3), or more general the MLI approach (see Appendix C.4). In Appendix D of this contribution we introduce the theoretical foundations of a new level for MLI. Especially, this level is suitable for problems whose right hand side derivatives have a sparse structure. On top of that, our unified framework for switched OCPs can be linked to MLI in two different ways: either one could consider to solve the resulting OCPs in a Multiple Shooting context, or one could apply the MLI idea to collocation methods.

Appendices

Appendix A

Auxiliary Results

A.1 Proof of Lemma 8.5

We prove Lemma 8.5, i.e., we have to show that it holds

$$\int_{t_s}^{t_f} \boldsymbol{f}(t) \left(\int_{t_s}^{t} \boldsymbol{g}(\tau)\, \mathrm{d}\tau \right) \mathrm{d}t = \int_{t_s}^{t_f} \left(\int_{t}^{t_f} \boldsymbol{f}(\tau)\, \mathrm{d}\tau \right) \boldsymbol{g}(t)\, \mathrm{d}t.$$

We define the functions $\boldsymbol{F}(\cdot)$ and $\boldsymbol{G}(\cdot)$ as

$$\boldsymbol{F}(t) \stackrel{\text{def}}{=} \int_{t_f}^{t} \boldsymbol{f}(\tau)\, \mathrm{d}\tau \quad \text{and} \quad \boldsymbol{G}(t) \stackrel{\text{def}}{=} \int_{t_s}^{t} \boldsymbol{g}(\tau)\, \mathrm{d}\tau$$

and calculate (integration by parts) then

$$\begin{aligned} &\int_{t_s}^{t_f} \boldsymbol{f}(t) \left(\int_{t_s}^{t} \boldsymbol{g}(\tau)\, \mathrm{d}\tau \right) \mathrm{d}t + \int_{t_s}^{t_f} \left(\int_{t_f}^{t} \boldsymbol{f}(\tau)\, \mathrm{d}\tau \right) \boldsymbol{g}(t)\, \mathrm{d}t \\ &= \int_{t_s}^{t_f} \boldsymbol{f}(t) \boldsymbol{G}(t)\, \mathrm{d}t + \int_{t_s}^{t_f} \boldsymbol{F}(t) \boldsymbol{g}(t)\, \mathrm{d}t = \boldsymbol{F}(t) \boldsymbol{G}(t) |_{t_s}^{t_f} = 0. \end{aligned}$$

This concludes the proof.

A.2 Proof of Lemma 9.2

We prove Lemma 9.2, i.e., with the augmented Hamilton function

$$\hat{\mathcal{H}}(t, x, u, \lambda, \mu; t_s, t_f) \stackrel{\text{def}}{=} \boldsymbol{\psi}(t, x, u; t_s, t_f) + \lambda^T \boldsymbol{f}(t, x, u; t_s, t_f) + \mu^T \boldsymbol{c}(t, x, u; t_s, t_f).$$

we have to show that

$$\hat{\mathcal{H}}[-1] = -\frac{t_f - t_s}{2} \int_{-1}^{+1} \hat{\mathcal{H}}'_{t_s}[\tau]\, \mathrm{d}\tau + \frac{1}{2} \int_{-1}^{+1} \hat{\mathcal{H}}[\tau]\, \mathrm{d}\tau, \tag{A.1}$$

$$\hat{\mathcal{H}}[+1] = +\frac{t_f - t_s}{2} \int_{-1}^{+1} \hat{\mathcal{H}}'_{t_f}[\tau]\, \mathrm{d}\tau + \frac{1}{2} \int_{-1}^{+1} \hat{\mathcal{H}}[\tau]\, \mathrm{d}\tau. \tag{A.2}$$

We introduce the time transformation function $\boldsymbol{t} : [-1;+1] \longrightarrow [t_\mathrm{s}, t_\mathrm{f}]$ as

$$\boldsymbol{t}(\tau; t_\mathrm{s}, t_\mathrm{f}) \stackrel{\text{def}}{=} \frac{t_\mathrm{f}+t_\mathrm{s}}{2} + \tau \cdot \frac{t_\mathrm{f}-t_\mathrm{s}}{2},$$

and find

$$\begin{aligned}
\boldsymbol{t}'(\tau; t_\mathrm{s}, t_\mathrm{f}) &= \frac{t_\mathrm{f}-t_\mathrm{s}}{2},\\
\frac{\partial \boldsymbol{t}}{\partial t_\mathrm{s}}(\tau; t_\mathrm{s}, t_\mathrm{f}) &= \frac{1}{2} - \frac{1}{2}\cdot\tau,\\
\frac{\partial \boldsymbol{t}}{\partial t_\mathrm{f}}(\tau; t_\mathrm{s}, t_\mathrm{f}) &= \frac{1}{2} + \frac{1}{2}\cdot\tau.
\end{aligned}$$

Then (A.1) follows from

$$\begin{aligned}
&-\frac{t_\mathrm{f}-t_\mathrm{s}}{2}\int_{-1}^{+1} \hat{\mathcal{H}}'_{t_\mathrm{s}}[\tau]\,\mathrm{d}\tau + \frac{1}{2}\int_{-1}^{+1}\hat{\mathcal{H}}[\tau]\,\mathrm{d}\tau\\
=&\int_{-1}^{+1} -\boldsymbol{t}'(\tau; t_\mathrm{s}, t_\mathrm{f})\cdot\hat{\mathcal{H}}'_t[\tau]\left(\frac{1}{2}-\frac{1}{2}\cdot\tau\right)\mathrm{d}\tau + \frac{1}{2}\int_{-1}^{+1}\hat{\mathcal{H}}[\tau]\,\mathrm{d}\tau\\
=&\frac{1}{2}\int_{-1}^{+1}\frac{\mathrm{d}}{\mathrm{d}\tau}\hat{\mathcal{H}}[\tau]\tau\,\mathrm{d}\tau - \frac{1}{2}\int_{-1}^{+1}\frac{\mathrm{d}}{\mathrm{d}\tau}\hat{\mathcal{H}}[\tau]\,\mathrm{d}\tau + \frac{1}{2}\int_{-1}^{+1}\hat{\mathcal{H}}[\tau]\,\mathrm{d}\tau\\
=&\frac{1}{2}\,\hat{\mathcal{H}}[\tau]\tau\big|_{-1}^{+1} - \frac{1}{2}\int_{-1}^{+1}\hat{\mathcal{H}}[\tau]\,\mathrm{d}\tau - \frac{1}{2}\,\hat{\mathcal{H}}[\tau]\big|_{-1}^{+1} + \frac{1}{2}\int_{-1}^{+1}\hat{\mathcal{H}}[\tau]\,\mathrm{d}\tau\\
=&\frac{1}{2}\hat{\mathcal{H}}[1] + \frac{1}{2}\hat{\mathcal{H}}[-1] - \frac{1}{2}\hat{\mathcal{H}}[1] + \frac{1}{2}\hat{\mathcal{H}}[-1] = \hat{\mathcal{H}}[-1],
\end{aligned}$$

and (A.2) from

$$\begin{aligned}
&+\frac{t_\mathrm{f}-t_\mathrm{s}}{2}\int_{-1}^{+1} \hat{\mathcal{H}}'_{t_\mathrm{s}}[\tau]\,\mathrm{d}\tau + \frac{1}{2}\int_{-1}^{+1}\hat{\mathcal{H}}[\tau]\,\mathrm{d}\tau\\
=&\int_{-1}^{+1} \boldsymbol{t}'(\tau; t_\mathrm{s}, t_\mathrm{f})\cdot\hat{\mathcal{H}}'_t[\tau]\left(\frac{1}{2}+\frac{1}{2}\cdot\tau\right)\mathrm{d}\tau + \frac{1}{2}\int_{-1}^{+1}\hat{\mathcal{H}}[\tau]\,\mathrm{d}\tau\\
=&\frac{1}{2}\int_{-1}^{+1}\frac{\mathrm{d}}{\mathrm{d}\tau}\hat{\mathcal{H}}[\tau]\tau\,\mathrm{d}\tau + \frac{1}{2}\int_{-1}^{+1}\frac{\mathrm{d}}{\mathrm{d}\tau}\hat{\mathcal{H}}[\tau]\,\mathrm{d}\tau + \frac{1}{2}\int_{-1}^{+1}\hat{\mathcal{H}}[\tau]\,\mathrm{d}\tau\\
=&\frac{1}{2}\,\hat{\mathcal{H}}[\tau]\tau\big|_{-1}^{+1} - \frac{1}{2}\int_{-1}^{+1}\hat{\mathcal{H}}[\tau]\,\mathrm{d}\tau + \frac{1}{2}\,\hat{\mathcal{H}}[\tau]\big|_{-1}^{+1} + \frac{1}{2}\int_{-1}^{+1}\hat{\mathcal{H}}[\tau]\,\mathrm{d}\tau\\
=&\frac{1}{2}\hat{\mathcal{H}}[1] + \frac{1}{2}\hat{\mathcal{H}}[-1] + \frac{1}{2}\hat{\mathcal{H}}[1] - \frac{1}{2}\hat{\mathcal{H}}[-1] = \hat{\mathcal{H}}[+1].
\end{aligned}$$

A.3 Towards the Generalized Taylor's Theorem I

Taylor's Theorem in the calculus with integral type remainder reads as:

$$f(x_0) = f(x) + f'(x)(x_0 - x) + \int_x^{x_0} f''(t)(x_0 - t)\,\mathrm{d}t.$$

The remainder term can be rewritten by means of the substitution $t(s) = x_0 + s(x - x_0)$, $t(0) = x_0$, $t(1) = x$, $\mathrm{d}t = (x - x_0)\mathrm{d}s$ as

$$\begin{aligned}\int_x^{x_0} f''(t)(x_0 - t)\,\mathrm{d}t &= -\int_1^0 f''(x_0 + s(x - x_0))\,s\,(x - x_0)^2\,\mathrm{d}s \\ &= \int_0^1 f''(x_0 + s(x - x_0))\,(x - x_0)^2\,s\,\mathrm{d}s.\end{aligned}$$

It is a well–known fact (see e.g. Zeidler [467, Theorem 4.A]) that Taylor's Theorem also holds in a Banach space setting. We make use of this fact and write for appropriate functions $x(\cdot)$ and $x_0(\cdot)$ and a mapping $f(\cdot)$ the Taylor series expansion as

$$\begin{aligned}&f(x_0(\cdot)) - f(x(\cdot)) + f'(x(\cdot))(x(\cdot) - x_0(\cdot)) \\ &= \int_0^1 f''(x_0(\cdot) + s(x(\cdot) - x_0(\cdot)))\,(x(\cdot) - x_0(\cdot))^2\,s\,\mathrm{d}s.\end{aligned} \tag{A.3}$$

In order to put this formula into practice in Chapter 10 we use its notation for the exact solution function $x(\cdot)$, the approximate solution function $x_\mathrm{h}(\cdot)$ and the error function $e(t) = x(t) - x_\mathrm{h}(t)$ and substitute in (A.3) $x_0(\cdot) = x_\mathrm{h}(\cdot)$ which results in

$$\begin{aligned}&f(x_\mathrm{h}(\cdot)) - f(x(\cdot)) + f'(x(\cdot))(x(\cdot) - x_\mathrm{h}(\cdot)) \\ &= \int_0^1 (x(\cdot) - x_\mathrm{h}(\cdot))^T f''(x_\mathrm{h}(\cdot) + s(x(\cdot) - x_\mathrm{h}(\cdot)))\,(x(\cdot) - x_\mathrm{h}(\cdot))s\,\mathrm{d}s.\end{aligned}$$

An integration with a costate like function $\Lambda(\cdot)$ and integration over the horizon interval, which is split into the intervals $\mathcal{I}_n$, yields

$$\begin{aligned}&\sum_n \int_{\mathcal{I}_n} f'(x(t))\,e(t)\,\mathrm{d}\Lambda(t) - \sum_n \int_{\mathcal{I}_n} f(x(t)) - f(x_\mathrm{h}(t))\,\mathrm{d}\Lambda(t) \\ &= \sum_n \int_{\mathcal{I}_n} \int_0^1 e(t)^T f''(x_\mathrm{h}(t) + s\,e(t))\,e(t)s\,\mathrm{d}s\,\mathrm{d}\Lambda(t),\end{aligned}$$

which is exploited in Section 10.2.

A.4 Towards the Generalized Taylor's Theorem II

Similar to the calculations from Appendix A.3, we find a second result which is used in Section 10.2. If we use again the notation $(\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot))$ for the exact OCP solution and $(\boldsymbol{x}_{\mathrm{h}}(\cdot), \boldsymbol{u}_{\mathrm{h}}(\cdot))$ for the approximate OCP solution and if we define the error terms $\boldsymbol{e}_x(\cdot)$, $\boldsymbol{e}_u(\cdot)$, and $\boldsymbol{e}(\cdot)$ as

$$\boldsymbol{e}_x(\cdot) \stackrel{\text{def}}{=} \boldsymbol{x}(\cdot) - \boldsymbol{x}_{\mathrm{h}}(\cdot), \quad \boldsymbol{e}_u(\cdot) \stackrel{\text{def}}{=} \boldsymbol{u}(\cdot) - \boldsymbol{u}_{\mathrm{h}}(\cdot), \quad \boldsymbol{e}(\cdot) \stackrel{\text{def}}{=} [\boldsymbol{e}_x(\cdot)^T, \boldsymbol{e}_u(\cdot)^T]^T,$$

then we find the Taylor series expansion

$$\begin{aligned}
\boldsymbol{f}(\boldsymbol{x}_{\mathrm{h}}(\cdot), \boldsymbol{u}_{\mathrm{h}}(\cdot)) = \boldsymbol{f}(\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot)) &- \boldsymbol{f}'_x(\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot))\, \boldsymbol{e}_x(\cdot) - \boldsymbol{f}'_u(\boldsymbol{x}(\cdot), \boldsymbol{u}(\cdot))\, \boldsymbol{e}_u(\cdot) \\
&+ \int_0^1 \boldsymbol{e}_x(\cdot)^T \boldsymbol{f}''_{xx}(\boldsymbol{x}_{\mathrm{h}}(\cdot) + s\,\boldsymbol{e}_x(\cdot), \boldsymbol{u}_{\mathrm{h}}(\cdot) + s\,\boldsymbol{e}_u(\cdot))\, \boldsymbol{e}_x(\cdot)\, s \,\mathrm{d}s \\
&+ \int_0^1 \boldsymbol{e}_u(\cdot)^T \boldsymbol{f}''_{uu}(\boldsymbol{x}_{\mathrm{h}}(\cdot) + s\,\boldsymbol{e}_x(\cdot), \boldsymbol{u}_{\mathrm{h}}(\cdot) + s\,\boldsymbol{e}_u(\cdot))\, \boldsymbol{e}_u(\cdot)\, s \,\mathrm{d}s \\
&+ 2 \int_0^1 \boldsymbol{e}_x(\cdot)^T \boldsymbol{f}''_{xu}(\boldsymbol{x}_{\mathrm{h}}(\cdot) + s\,\boldsymbol{e}_x(\cdot), \boldsymbol{u}_{\mathrm{h}}(\cdot) + s\,\boldsymbol{e}_u(\cdot))\, \boldsymbol{e}_u(\cdot)\, s \,\mathrm{d}s.
\end{aligned}$$

An integration with a costate like function $\boldsymbol{\Lambda}(\cdot)$ and integration over the horizon interval, which is split into the intervals $\mathcal{I}_n$, yields

$$\begin{aligned}
&\sum_n \int_{\mathcal{I}_n} \boldsymbol{f}'_x(\boldsymbol{x}(t), \boldsymbol{u}(t))\, \boldsymbol{e}_x(t) \,\mathrm{d}\boldsymbol{\Lambda}(t) + \sum_n \int_{\mathcal{I}_n} \boldsymbol{f}'_u(\boldsymbol{x}(t), \boldsymbol{u}(t))\, \boldsymbol{e}_u(t) \,\mathrm{d}\boldsymbol{\Lambda}(t) \\
-&\sum_n \int_{\mathcal{I}_n} \{\boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)) - \boldsymbol{f}(\boldsymbol{x}_{\mathrm{h}}(t), \boldsymbol{u}_{\mathrm{h}}(t))\} \,\mathrm{d}\boldsymbol{\Lambda}(t) \\
=&\sum_n \int_{\mathcal{I}_n} \int_0^1 \boldsymbol{e}_x(t)^T \boldsymbol{f}''_{xx}(\boldsymbol{x}_{\mathrm{h}}(t) + s\,\boldsymbol{e}_x(t), \boldsymbol{u}_{\mathrm{h}}(t) + s\,\boldsymbol{e}_u(t))\, \boldsymbol{e}_x(t)\, s \,\mathrm{d}s \,\mathrm{d}\boldsymbol{\Lambda}(t) \\
+&\sum_n \int_{\mathcal{I}_n} \int_0^1 \boldsymbol{e}_u(t)^T \boldsymbol{f}''_{uu}(\boldsymbol{x}_{\mathrm{h}}(t) + s\,\boldsymbol{e}_x(t), \boldsymbol{u}_{\mathrm{h}}(t) + s\,\boldsymbol{e}_u(t))\, \boldsymbol{e}_u(t)\, s \,\mathrm{d}s \,\mathrm{d}\boldsymbol{\Lambda}(t) \\
+2&\sum_n \int_{\mathcal{I}_n} \int_0^1 \boldsymbol{e}_x(t)^T \boldsymbol{f}''_{xu}(\boldsymbol{x}_{\mathrm{h}}(t) + s\,\boldsymbol{e}_x(t), \boldsymbol{u}_{\mathrm{h}}(t) + s\,\boldsymbol{e}_u(t))\, \boldsymbol{e}_u(t)\, s \,\mathrm{d}s \,\mathrm{d}\boldsymbol{\Lambda}(t).
\end{aligned}$$

Appendix B

Numerical Analysis

B.1 Orthogonal Polynomials

Orthogonal polynomials play a crucial role in *polynomial interpolation* as well as *numerical integration* which will be introduced in subsequent sections. A system of polynomials $\{\boldsymbol{Q}_n\}$ is called to be orthogonal if it satisfies the *condition of orthogonality*, i.e.,

$$\int_{-1}^{+1} \boldsymbol{w}(t)\boldsymbol{Q}_n(t)\boldsymbol{Q}_m(t)\,\mathrm{d}t = 0, \quad n \neq m, \tag{B.1}$$

where the degree of every polynomial $\boldsymbol{Q}_n(\cdot)$ is equal to n, and the weight function $\boldsymbol{w}(t) \geqslant 0$ on the interval $[-1,+1]$. If we define the scalar product

$$\langle \boldsymbol{f}, \boldsymbol{g} \rangle \stackrel{\text{def}}{=} \int_{-1}^{+1} \boldsymbol{w}(t)\boldsymbol{f}(t)\boldsymbol{g}(t)\,\mathrm{d}t$$

on the linear space $L^2([-1,+1],\mathbb{R})$ the condition of orthogonality can be formulated as $\langle \boldsymbol{Q}_n, \boldsymbol{Q}_m \rangle = 0$, $n \neq m$. The following result answers the question if such systems of orthogonal polynomials exist.

Theorem B.1
There exist uniquely defined polynomials $\boldsymbol{Q}_n(\cdot)$ of degree n, $n = 0, 1, \ldots$, such that

$$\langle \boldsymbol{Q}_n, \boldsymbol{Q}_m \rangle = 0, \quad n \neq m. \qquad \triangle$$

Proof See Stoer et al. [419]. □

Since every polynomial $\boldsymbol{P}(\cdot)$ of degree n can be represented as a linear combination of the orthogonal polynomials $\boldsymbol{Q}_m(\cdot)$, $m \leqslant n$ we have $\langle \boldsymbol{P}, \boldsymbol{Q}_n \rangle = 0$ for all $\boldsymbol{P}(\cdot)$ up to degree $n-1$. It can be shown that the roots t_i, $i = 1, \ldots, n$ of $\boldsymbol{Q}_n(\cdot)$ are real, simple and lie in the open interval $(-1,+1)$.
The class of Jacobi polynomials $\boldsymbol{P}_n(x;\alpha,\beta)$ forms an important class of orthogonal polynomials. They are the eigenfunctions of a singular Sturm–Liouville problem on the interval $[-1,+1]$ and fulfill the condition of orthogonality for the weights $\boldsymbol{w}(t) = (1-t)^\alpha(1+t)^\beta$, $\alpha > -1$, $\beta > -1$.
Well known representatives of the Jacobi polynomial class are given by Chebyshev polynomials of first kind $\{\boldsymbol{T}_n\}$ (for which $\alpha = \beta = -\frac{1}{2}$ holds) and Legendre polynomials $\{\boldsymbol{P}_n\}$ (for which $\alpha = \beta = 0$ holds).

Legendre polynomials

The Legendre polynomials $\{\boldsymbol{P}_n\}$ were introduced by Legendre [287] and are eigenfunctions of the singular Sturm–Liouville problem

$$\frac{\mathrm{d}}{\mathrm{d}t}\left((1-t^2)\cdot\frac{\mathrm{d}\boldsymbol{P}_n}{\mathrm{d}t}(t)\right)+n(n+1)\cdot\boldsymbol{P}_n(t)=0,$$

with the normalization $\boldsymbol{P}_n(1)=1$. Legendre polynomials are orthogonal with respect to the scalar product

$$\langle \boldsymbol{f},\boldsymbol{g}\rangle=\int_{-1}^{+1}\boldsymbol{f}(t)\cdot\boldsymbol{g}(t)\,\mathrm{d}t,$$

which means that $\boldsymbol{w}(t)=1$ in (B.1). They are defined by the formula

$$\boldsymbol{P}_n(t)=\frac{1}{n!2^n}\frac{\mathrm{d}^n}{\mathrm{d}t^n}(t^2-1)^n,\quad n=0,1,\ldots,$$

and satisfy the three term recursion

$$(n+1)\cdot\boldsymbol{P}_{n+1}(t)=(2n+1)\cdot t\cdot\boldsymbol{P}_n(t)-n\cdot\boldsymbol{P}_{n-1}(t),\quad \boldsymbol{P}_0(t)=1,\quad \boldsymbol{P}_1(t)=t.$$

For fixed $\theta\neq 0$ it holds

$$\boldsymbol{P}_n(\cos\theta)=\left[\frac{2}{n\pi\sin\theta}\right]^{\frac{1}{2}}\sin\left[\left(n+\frac{1}{2}\right)\theta+\frac{\pi}{4}\right]+\mathcal{O}\left(n^{-\frac{3}{2}}\right),\qquad n\to\infty. \tag{B.2}$$

Figure B.1 depicts the three Legendre polynomials $\boldsymbol{P}_1(\cdot)$, $\boldsymbol{P}_3(\cdot)$ and $\boldsymbol{P}_5(\cdot)$.

Chebyshev polynomials

Chebyshev polynomials of the first kind $\{\boldsymbol{T}_n\}$ were introduced by Chebyshev [106] in 1854 and are the eigenfunctions of the singular Sturm–Liouville problem

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\sqrt{1-t^2}\cdot\frac{\mathrm{d}\boldsymbol{T}_n}{\mathrm{d}t}(t)\right)+\frac{n^2}{\sqrt{1-t^2}}\cdot\boldsymbol{T}_n(t)=0,$$

with the normalization $\boldsymbol{P}_n(1)=1$. They are orthogonal with respect to the scalar product

$$\langle \boldsymbol{f},\boldsymbol{g}\rangle=\int_{-1}^{+1}\frac{\boldsymbol{f}(t)\cdot\boldsymbol{g}(t)}{\sqrt{1-t^2}}\,\mathrm{d}t,$$

which means that

$$\boldsymbol{w}(t)=\frac{1}{\sqrt{1-t^2}},\quad t\in(-1,+1),$$

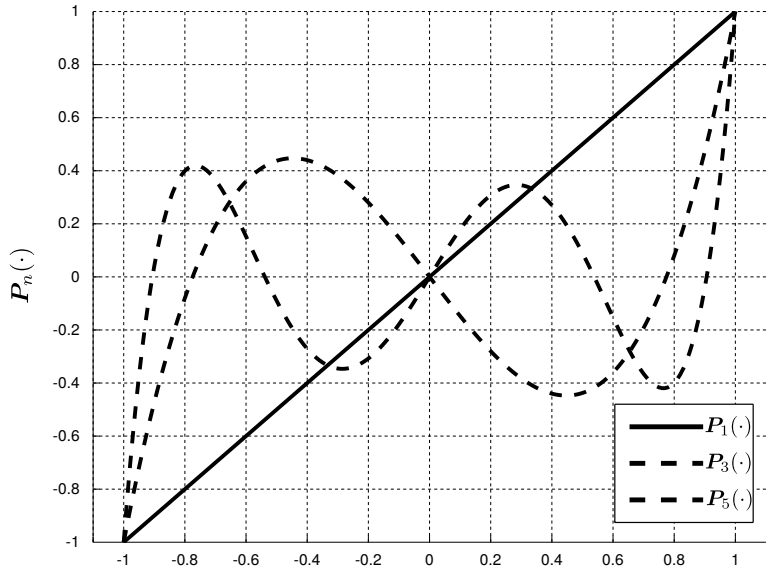


Figure B.1: The plot shows three different Legendre polynomials.

in (B.1). Chebyshev polynomials can be easily obtained by the formula

$$\boldsymbol{T}_n(t) = \cos(n \cdot \arccos(t)), \quad t \in [-1, +1],$$

and satisfy the three term recursion

$$\boldsymbol{T}_{n+1}(t) = 2t \cdot \boldsymbol{T}_n(t) - \boldsymbol{T}_{n-1}(t), \quad \boldsymbol{T}_0(t) = 1, \quad \boldsymbol{T}_1(t) = t.$$

The roots of Chebyshev polynomial $\boldsymbol{T}_n(t)$ are often called Chebyshev *nodes* and can be found to be

$$t_i^{(n)} = \cos\left(\frac{2i-1}{2n}\pi\right), \quad i \in [n].$$

B.2 Polynomial Interpolation

Lagrange Interpolation Given an arbitrary dataset *polynomial interpolation* deals with the task to interpolate the dataset by a polynomial of lowest possible degree. Often the dataset is given by some abscissa values and the associated function values for a given function.

We consider a compact interval $\mathcal{I} \stackrel{\text{def}}{=} [a, b]$ with $a < b$, a function $\boldsymbol{f} : \mathcal{I} \longrightarrow \mathbb{R}$ and $n \in \mathbb{N}$ distinct abscissa values t_i as well as the associated function values f_i, i.e.,

$$f_i \stackrel{\text{def}}{=} \boldsymbol{f}(t_i), \quad i \in [n].$$

It is the goal to interpolate the dataset $\{(t_i, f_i)\}$. The points t_i are called *support points*. One can show that there exists a unique polynomial $\boldsymbol{P} : \mathcal{I} \longrightarrow \mathbb{R}$ of degree $n-1$ such that

$$\boldsymbol{P}(t_i) = f_i, \quad i \in [n].$$

The unique polynomial can be constructed by means of the so called Lagrange interpolation formula, which was first discovered by Waring [446], as

$$\boldsymbol{P}(t) = \sum_{i=1}^{n} f_i \cdot \boldsymbol{L}_i(t), \tag{B.3}$$

where the $\boldsymbol{L}_i(\cdot)$ denote the Lagrange interpolation polynomials. These polynomials are given as

$$\boldsymbol{L}_i(t) \stackrel{\text{def}}{=} \prod_{\substack{j=1 \\ j \neq i}}^{n} \frac{t - t_j}{t_i - t_j}. \tag{B.4}$$

The i–th interpolation polynomial has the property that it is equal to one at the i–th support point t_i and vanishes at the remaining ones t_j, $j \neq i$, i.e.,

$$\boldsymbol{L}_i(t_j) = \delta_{ij} = \begin{cases} 1, & j = i, \\ 0, & j \neq i. \end{cases}$$

Figure B.2 depicts three interpolation polynomials for six support points on the interval $[-1, +1]$.

Choice of Support Points It is a crucial point how to choose the support points t_i. Intuitively one could choose equidistant support points on the interval $\mathcal{I}$. However, for polynomial approximation, an equidistant grid has some undesirable properties. One would expect that the error between the polynomial approximation and the true function continuously decreases for an increasing number of support points. This is not the case for an equidistant grid. For a finer equidistant grid one observes the so called Runge *phenomenon* (see e.g. Davis [125]), meaning that the approximation error near the interval boundaries increases when the grid becomes finer. This implies that polynomial interpolation using equally distributed points is an extremely ill–conditioned problem, i.e., small changes in the input data might cause huge changes in the interpolant.
In order to make polynomial interpolation to be a well–posed problem we have to dispense with equally distributed support points. Approximation theory suggests to use support points that are cumulated at the interval boundaries and having an asymptotic density which is proportional to $(1-t^2)^{-1/2}$ as $n \to \infty$.

Towards Interpolation Error By a proper choice of the support points one can overcome the Runge phenomenon and guarantee that the polynomial approximation error monotoni-

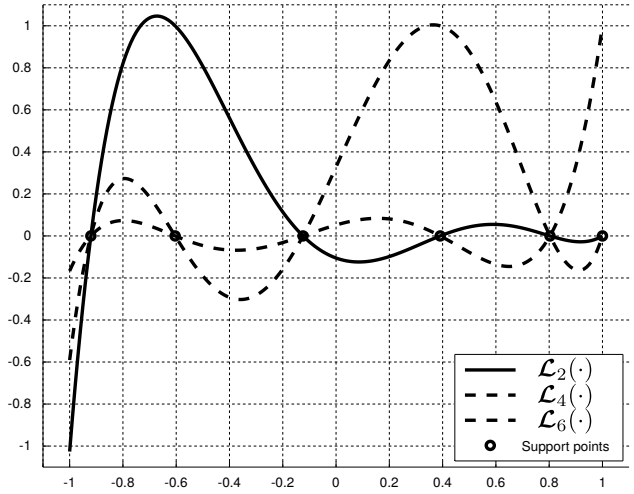


Figure B.2: Three Lagrange interpolating polynomials for six arbitrary support points on the compact interval $[-1, +1]$.

cally decreases as the number of support points increases. Support points based on the roots of Legendre and Chebyshev polynomials have this property. The roots of these polynomials have the characteristic that they are cumulated at the interval boundaries.
The interpolation error $\boldsymbol{f}(t) - \boldsymbol{P}(t)$ for functions $\boldsymbol{f}$ in which n derivatives exist is known to be

$$\boldsymbol{f}(t) - \boldsymbol{P}(t) = \frac{1}{n!} \frac{\mathrm{d}^n \boldsymbol{f}}{\mathrm{d}t^n}(\xi) \prod_{i=1}^{n} (t - t_i) \tag{B.5}$$

for some $\xi \in \mathcal{I}$. This equation shows, as the number of support points is increased, a rapid convergence for functions whose derivatives are bounded.
According to the approximation error (B.5) it is obvious to choose support points that minimize

$$\max_{t \in [-1,+1]} \left| \prod_{i=1}^{n} (t - t_i) \right| .$$

One can show that the maximum norm of any such polynomial is bounded from below by 2^{1-n}. This bound is attained by the scaled Chebyshev polynomials $2^{1-n} \cdot \boldsymbol{T}_n$. Since it holds $|\boldsymbol{T}_n(t)| \leqslant 1$ for $t \in [-1, +1]$ we get an approximation error

$$|\boldsymbol{f}(t) - \boldsymbol{P}(t)| = \frac{1}{2^{n-1}\, n!} \max_{\xi \in [-1,+1]} \left| \frac{\mathrm{d}^n \boldsymbol{f}}{\mathrm{d}t^n}(\xi) \right|$$

if we choose the Chebyshev nodes as support points. Other point sets leading to well–conditioned polynomial approximations are given by Legendre points that come from roots or extrema of the Legendre polynomials.

An Improved Lagrange Formula Representation (B.3)+(B.4) of the Lagrange interpolation suffers from several drawbacks in particular in terms of numerical aspects. Relevant shortcomings for our purposes are the following:

(i) An evaluation of $\boldsymbol{P}(t)$ costs $\mathcal{O}\left(n^2\right)$ additions and multiplications.

(ii) The evaluation is numerically unstable.

For this reason the Lagrange form of $\boldsymbol{P}(\cdot)$ is mainly of theoretical relevance to prove theorems but of less importance for numerical computations.
However, there is a way to rewrite the Lagrange formula such that it can be evaluated and updated in $\mathcal{O}(n)$ operations. To this end, we remark that the numerator of $\boldsymbol{L}_i(\cdot)$ in (B.3) can be written as

$$\boldsymbol{L}(t) = \prod_{j=1}^{n} (t - t_j) = (t - t_1) \cdot (t - t_2) \cdot \ldots \cdot (t - t_n)$$

divided by $t - t_i$. Recalling L'Hospital's rule (see Taylor [423, p. 456]) we calculate the quantity

$$\dot{\boldsymbol{L}}(t_i) = \lim_{t \to t_i} \frac{\dot{\boldsymbol{L}}(t)}{1} = \lim_{t \to t_i} \frac{\boldsymbol{L}(t)}{t - t_i} = \lim_{t \to t_i} \prod_{\substack{j=1 \\ j \neq i}}^{n} (t - t_j) = \prod_{\substack{j=1 \\ j \neq i}}^{n} (t_i - t_j),$$

which can be identified as the denominator of $\boldsymbol{L}_i(\cdot)$ such that we have shown

$$\boldsymbol{L}_i(t) \stackrel{\text{def}}{=} \frac{\boldsymbol{L}(t)}{\dot{\boldsymbol{L}}(t_i) \cdot (t - t_i)}. \tag{B.6}$$

The Lagrange interpolation formula then reads as

$$\boldsymbol{P}(t) = \sum_{i=1}^{n} f_i \cdot \boldsymbol{L}_i(t) = \boldsymbol{L}(t) \sum_{i=1}^{n} \frac{f_i}{\dot{\boldsymbol{L}}(t_i) \cdot (t - t_i)}.$$

Differentiation of Polynomial Interpolants For pseudospectral methods, which approximate OCP states and controls by global polynomials, it is required to evaluate at least first–order derivatives of interpolation polynomials, i.e., one needs to calculate

$$\dot{\boldsymbol{P}}(t) = \sum_{i=1}^{n} f_i \cdot \dot{\boldsymbol{L}}_i(t),$$

evaluated at the support points t_j, $j \in [n]$.

Representation (B.6) of the i–th Lagrange polynomial helps us to derive handy expressions: multiplying both sides of (B.6) by $t-t_i$ to render them differentiable at $t = t_i$, and a subsequent differentiation yields

$$\dot{\boldsymbol{L}}_i(t) \cdot (t - t_i) + \boldsymbol{L}_i(t) = \frac{\dot{\boldsymbol{L}}(t)}{\dot{\boldsymbol{L}}(t_i)}. \tag{B.7}$$

Differentiating one more time yields

$$\ddot{\boldsymbol{L}}_i(t) \cdot (t - t_i) + 2\,\dot{\boldsymbol{L}}_i(t) = \frac{\ddot{\boldsymbol{L}}(t)}{\dot{\boldsymbol{L}}(t_i)}. \tag{B.8}$$

By inserting $t = t_j$, $i \neq j$, in (B.7) and $t = t_i$ in (B.8) we get

$$D_{j,i} = \dot{\boldsymbol{L}}_i(t_j) = \begin{cases} \dfrac{\dot{\boldsymbol{L}}(t_j)}{\dot{\boldsymbol{L}}(t_i) \cdot (t_j - t_i)}, & \text{if } i \neq j, \\ \dfrac{\ddot{\boldsymbol{L}}(t_i)}{2\,\dot{\boldsymbol{L}}(t_i)}, & \text{if } i = j. \end{cases} \tag{B.9}$$

The matrix $D = \big(D_{j,i}\big)_{i,j=1,\ldots,n}$ is commonly known as (first–order) *differentiation matrix*. For a grid $\{t_i\}$ of support points and an associated function value vector $f = [f_1, \ldots, f_n]^T$ the vector $D\,f$ can be obtained by interpolating the data $\{(t_i, f_i)\}$ and differentiating the interpolating polynomial at the grid points.
Note that the representation (B.9) of the differentiation matrix in terms of $\boldsymbol{L}(\cdot)$ and its first– and second–order derivatives is rather convenient: as we have emphasized already the support points are preferably chosen to be roots of specific orthogonal polynomials whose derivatives (evaluated at the support points) can often be obtained in a numerically stable and computationally efficient way.

B.3 Numerical Integration

Given a function $\boldsymbol{f} : [-1, +1] \longrightarrow \mathbb{R}$ *numerical integration* deals with methods that approximate the definite integral

$$I(\boldsymbol{f}) \stackrel{\text{def}}{=} \int_{-1}^{+1} \boldsymbol{f}(t)\,\mathrm{d}t \tag{B.10}$$

numerically. Such techniques are typically referred to as *quadrature* formulas.
More concrete the approximation of definite integrals usually involves formulas of the type

$$I(\boldsymbol{f}) = I_n(\boldsymbol{f}) + e_n = \sum_{i=1}^{n} \omega_i \cdot \boldsymbol{f}(t_i) + e_n, \tag{B.11}$$

where the ω_i are called *quadrature weights* and the t_i are called *quadrature points* or *nodes*. The

approximation error between the exact integral $I(\boldsymbol{f})$ and its approximation $I_n(\boldsymbol{f})$ is denoted by e_n.

B.3.1 Quadrature Using Interpolating Polynomials

Quadrature formulas $I_n(\boldsymbol{f})$ can be constructed by means of the polynomial interpolation which was introduced in the previous section. Given n distinct and arbitrary support points the integrand $\boldsymbol{f}(\cdot)$ is approximated by Lagrange polynomials as

$$\int_{-1}^{+1} \boldsymbol{f}(t)\,\mathrm{d}t \simeq \int_{-1}^{+1} \sum_{i=1}^{n} \boldsymbol{L}_i(t) \cdot \boldsymbol{f}(t_i)\,\mathrm{d}t. \tag{B.12}$$

Hence, the quadrature weights ω_i in (B.11) can be easily determined as

$$\omega_i = \int_{-1}^{+1} \boldsymbol{L}_i(t)\,\mathrm{d}t, \quad i \in [n].$$

Quadrature rules of this type are exact for polynomials of degree $n-1$ or less. It is important to note that the ω_i are constants, independent of $\boldsymbol{f}(\cdot)$.
We restrict our discussions about quadrature using interpolating polynomials to the linear case since it is the only one employed in this thesis. Let $t_1 = -1$ and $t_2 = +1$. From equation (B.12) it follows that

$$\begin{aligned} I_2(\boldsymbol{f}) &= \boldsymbol{f}(-1) \int_{-1}^{+1} \frac{t+1}{2}\,\mathrm{d}t + \boldsymbol{f}(+1) \int_{-1}^{+1} \frac{t-1}{2}\,\mathrm{d}t \\ &= \boldsymbol{f}(-1) + \boldsymbol{f}(+1). \end{aligned}$$

The case of linear interpolation is typically referred to as the *trapezoidal rule*.

B.3.2 Gauss Quadrature

Compared to (B.10) we extend our investigations by considering a broader class of definite integrals. They have the form

$$I(\boldsymbol{f}) \stackrel{\text{def}}{=} \int_{-1}^{+1} \boldsymbol{w}(t)\boldsymbol{f}(t)\,\mathrm{d}t,$$

where $\boldsymbol{w}(\cdot)$ is a given nonnegative *weight function* that has to satisfy some mild requirements, cf. Stoer et al. [419]. As in (B.11) we want to retrieve integration rules of type

$$I_n(\boldsymbol{f}) = \sum_{i=1}^{n} \omega_i \cdot \boldsymbol{f}(t_i).$$

In the previous section, we assumed the quadrature nodes to be fixed in advance. If we leave

their location up to optimization we can gain more degrees of freedom and can therefore expect a higher degree of precision for the integral approximation. This idea leads us to an algorithm class which is called *Gaussian quadrature*.
The Gaussian quadrature is concerned to choose the quadrature nodes in an optimal manner. In other words it determines for an arbitrary function $\boldsymbol{f}(\cdot)$ quadrature nodes $\{t_i\}$ all in $[-1,+1]$ and quadrature weights $\{\omega_i\}$ which would be expected to minimize the approximation error $e_n \overset{\text{def}}{=} I(\boldsymbol{f}) - I_n(\boldsymbol{f})$.
There are at most $2n$ parameters involved since the quadrature weights $\{\omega_i\}$ are completely arbitrary and those of the $\{t_i\}$ are restricted just in this sense that they have to be in the interval $[-1,+1]$ and that $\boldsymbol{f}(\cdot)$ has to be defined at these points.
Considering the coefficients of a polynomial also as free parameters, the class of polynomials up to degree $2n-1$ contains at most $2n$ parameters. Hence, this would be the polynomial class for which equation (B.11) can be expected to hold with approximation error e_n equal to zero.
By means of orthogonal polynomials the following result shows that the Gaussian integration rules are unique and indeed of order $2n-1$. Furthermore, it is shown that the weights ω_i are positive and the integration nodes x_i are all in the interval $[-1,+1]$.

Theorem B.2

(i) Let $\{t_i\}$ be the roots of the n–th orthogonal polynomial $\boldsymbol{Q}_n(\cdot)$ and let $\{\omega_i\}$ be the solutions of the equation system

$$\sum_{i=1}^{n} \omega_i \cdot \boldsymbol{Q}_j(t_i) = \begin{cases} \langle \boldsymbol{Q}_0, \boldsymbol{Q}_0 \rangle, & \text{if } j = 0, \\ 0, & \text{if } j \in [n-1]. \end{cases} \tag{B.13}$$

Then $\omega_i > 0$ for $i \in [n]$, and

$$\int_{-1}^{+1} \boldsymbol{w}(t)\boldsymbol{P}(t)\,\mathrm{d}t = \sum_{i=1}^{n} \omega_i \cdot \boldsymbol{P}(t_i) \tag{B.14}$$

holds for all polynomials $\boldsymbol{P}(\cdot)$ up to degree $2n-1$.

(ii) Conversely, if (B.14) holds for all polynomials with degree equal to $2n-1$ or less, then the t_i are the roots of $\boldsymbol{Q}_n(\cdot)$ and the weights ω_i satisfy the equation system (B.13). △

Proof See Stoer et al. [419]. □

The approximation error of Gaussian integration is estimated in the following theorem.

Theorem B.3
If $\boldsymbol{f} \in \mathcal{C}^{2n}([-1,+1],\mathbb{R})$, then

$$\int_{-1}^{+1} \boldsymbol{w}(t)\boldsymbol{f}(t)\,\mathrm{d}t - \sum_{i=1}^{n} \omega_i \cdot \boldsymbol{f}(t_i) = \frac{1}{(2n)!} \frac{\mathrm{d}^{2n}\boldsymbol{f}}{\mathrm{d}t^{2n}}(\xi)\,\langle \boldsymbol{Q}_n, \boldsymbol{Q}_n \rangle$$

for some $\xi \in (-1,+1)$. △

Proof See Stoer et al. [419]. □

In the following subsections we investigate the special case of the weighting function $w(t) = 1$ since it is the only one used in this thesis. We have seen in Section B.1 that the Legendre polynomials are the associated orthogonal polynomials. Moreover, we examine the implications of fixing either one or two integration nodes to interval boundaries.

Legendre–Gauss Quadrature

The quadrature rule with the maximum degree of precision for weighting function $w(t) = 1$ is the LG quadrature rule. It is exact for polynomials up to degree $2n - 1$. The quadrature points and associated weights are determined such that

$$\int_{-1}^{+1} f(t)\,\mathrm{d}t = \sum_{i=1}^{n} \omega_i \cdot f(t_i) + e_n,$$

and the error e_n is zero for polynomials $f(\cdot)$ of degree $2n-1$. The Gauss points t_i are determined as the roots of the n–th degree Legendre polynomial and the weights are the integrals of the associated Lagrange interpolation polynomials, i.e.,

$$\omega_j = \int_{-1}^{+1} \prod_{\substack{i=0 \\ i\neq j}}^{n} \frac{t - t_i}{t_j - t_i}\,\mathrm{d}t = \frac{2}{(1-t_i^2)\left[\dot{P}_n(t_i)\right]^2}, \quad j \in [n],$$

where $\dot{P}_n(\cdot)$ denotes the derivative of the n–th Legendre polynomial. The Gauss points are all interior to the interval $[-1, +1]$ and tend to be more piled close to the interval boundaries. The approximation error e_n is proportional to the $(2n)$–th derivative of the integrand, so that we can state for $\xi \in [-1, +1]$:

$$e_n = \frac{1}{(2n)!}\frac{\mathrm{d}^{2n} f}{\mathrm{d}t^{2n}}(\xi) \int_{-1}^{+1} \left(\prod_{i=1}^{n}(t - t_i)\right)^2 \mathrm{d}t = \frac{2^{2n+1}(n!)^4}{(2n+1)[(2n)!]^3}\frac{\mathrm{d}^{2n} f}{\mathrm{d}t^{2n}}(\xi).$$

Legendre–Gauss–Radau Quadrature

The second quadrature rule presented here is the LGR quadrature rule whose nodes lie on the half–open interval $[-1, +1)$. This means that we fix one integration node to start of the integration interval. The quadrature nodes and the associated weights are then determined such that

$$\int_{-1}^{+1} f(t)\,\mathrm{d}t = \omega_1 \cdot f(-1) + \sum_{i=2}^{n} \omega_i \cdot f(t_i) + e_n,$$

and the error e_n is zero for polynomials of highest possible degree.

Since one of the points is forced to lie at the interval boundary we loose one degree of freedom. Hence, it is exact for polynomials up to at most degree $2n-2$. The LGR nodes t_i are determined

as the roots of the polynomial

$$\boldsymbol{P}_n(t) + \boldsymbol{P}_{n-1}(t),$$

where $\boldsymbol{P}_n(\cdot)$ denotes the n–th degree Legendre polynomial. The inner quadrature weights are given as

$$\omega_i = \frac{1-t_i}{n^2[\boldsymbol{P}_{n-1}(t_i)]^2} = \frac{1}{(1-t_i)\left[\dot{\boldsymbol{P}}_{n-1}(t_i)\right]^2}, \quad i = 2, \dots, n,$$

and the endpoint weight is

$$\omega_1 = \frac{2}{n^2}.$$

The approximation error e_n for the LGR quadrature nodes can be found to be

$$e_n = \frac{n2^{2n-1}[(n-1)!]^4}{[(2n-1)!]^3} \frac{\mathrm{d}^{2n-1}\boldsymbol{f}}{\mathrm{d}t^{2n-1}}(\xi), \quad \xi \in (-1, +1).$$

The standard variant of LGR quadrature includes the initial point but not the final point. Alternatively one could also imagine a quadrature formula including the final point but not the initial point of the interval. The resulting quadrature rule is called *Flipped Legendre–Gauss–Radau (FLGR) quadrature*. The corresponding quadrature nodes are called *FLGR nodes* and can be found from the roots of $\boldsymbol{P}_n(t) - \boldsymbol{P}_{n-1}(t)$.

Legendre–Gauss–Lobatto Quadrature

Compared to LGR quadrature, in LGL quadrature also the missing interval endpoint is involved, i.e., the interval boundaries -1 and $+1$ act as quadrature nodes. Fixing two quadrature nodes reduces the degree of freedom by two degrees as opposed to LG quadrature. Hence, this quadrature scheme can be accurate up to degree at most $2n-3$.
The formula is constructed by choosing weights ω_i and $n-2$ additional nodes $t_i \in (-1, +1)$ such that polynomials with the highest possible degree are integrated without approximation error, i.e.,

$$\int_{-1}^{+1} \boldsymbol{f}(t)\,\mathrm{d}t = \omega_1 \cdot \boldsymbol{f}(-1) + \sum_{i=2}^{n-1} \omega_i \cdot \boldsymbol{f}(t_i) + \omega_n \cdot \boldsymbol{f}(+1) + e_n.$$

The LGL quadrature rule is exact for polynomials up to degree $2n-3$. The LGL nodes are determined to be the roots of the derivative of the Legendre polynomial of degree $n-1$ together with the two boundary points ± 1, i.e., the roots of the polynomial $(1-t^2) \cdot \dot{\boldsymbol{P}}_{n-1}(t)$. The inner quadrature weights are given as

$$\omega_i = \frac{2}{n(n-1)[\boldsymbol{P}_{n-1}(t_i)]^2}, \quad i = 2, \dots, n-1,$$

and the endpoint weights are

$$\omega_i = \frac{2}{n(n-1)}, \quad i = 1, n.$$

The approximation error e_n for the LGL quadrature nodes can be found to be

$$e_n = -\frac{n(n-1)^3 2^{2n+1}[(n-2)!]^4}{(2n-1)[(2n-2)!]^3} \frac{\mathrm{d}^{2n-2} \boldsymbol{f}}{\mathrm{d}t^{2n-2}}(\xi), \quad \xi \in (-1, +1).$$

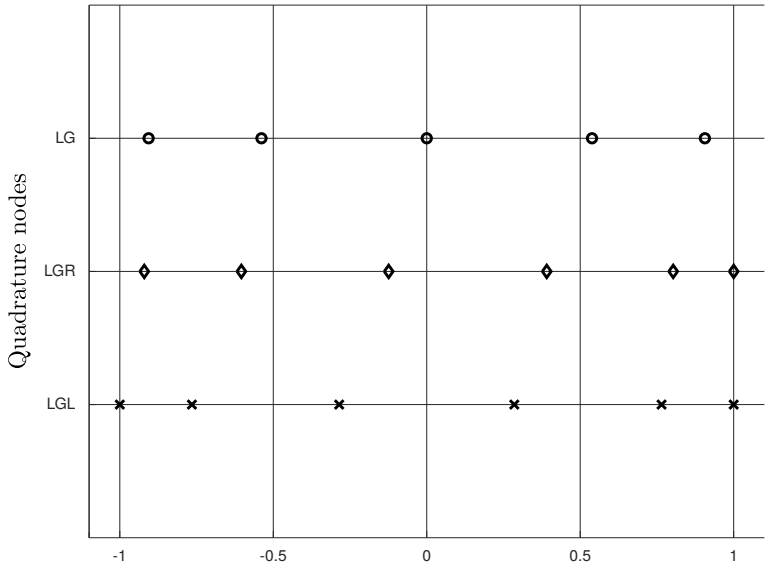


Figure B.3: Comparison of LG, LGR and LGL quadrature nodes.

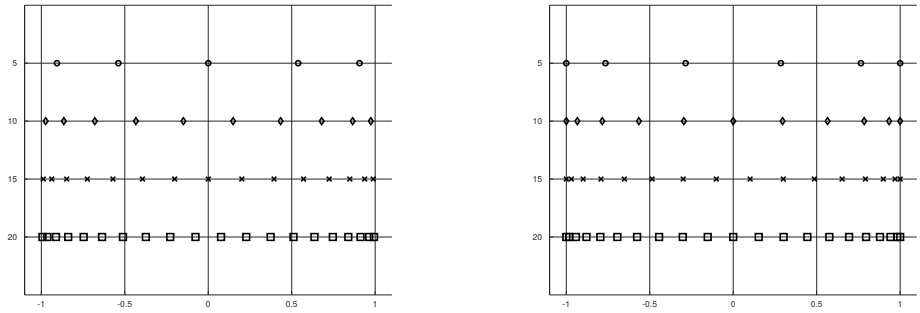


Figure B.4: The figure depicts LG and LGL quadrature node distributions for several polynomial degrees.

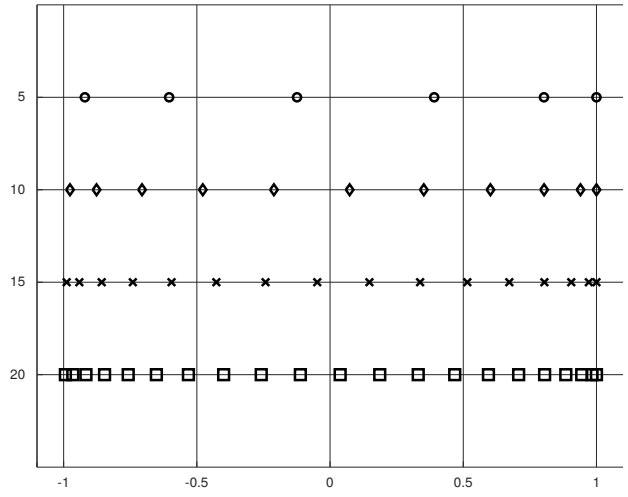


Figure B.5: The figure depicts the distribution of LGR quadrature nodes for several polynomial degrees.

Appendix C

Nonlinear Model Predictive Control

C.1 Feedback Control

For the most part of this thesis we covered the topic of solving OCPs. Regarding real world processes it is by no means judicious to apply the (numerically) computed optimal control $\boldsymbol{u}^*(\cdot)$ of an OCP to the full time horizon. The modeling of real processes may underlie several deficiencies: restricted by the demand for computational power or due to the lack of knowledge certain aspects of the process may be unmodeled. Moreover, the process environment may be disturbed by peripheral influences or disturbances are caused by inexactly determined parameters. In *open–loop* optimal control, after the solution of the OCP, the process is no longer monitored anymore and the obtained solution is applied without further feedback of the actual process. This makes *open–loop* control susceptible to disturbances which invalidate the previously optimal solution.
In contrast to this, there exists the so–called *closed–loop* or *feedback* control approach where the actual system behavior is taken into account and the controller is continuously fed with the updated system state.

C.2 The Principle of NMPC

In this section we introduce the concept of *MPC*, a quite versatile and powerful state–of–the–art feedback control approach. The MPC approach can be summarized as follows: one repeatedly solves open–loop OCPs on a finite prediction horizon. At sampling times t_s^k, where k denotes the sample index, we retrieve the current real process state x_s^k. Using x_s^k as initial state condition we solve an OCP on a prediction horizon $[t_\mathrm{s}^k, t_\mathrm{s}^k + H]$ and obtain an optimal control $\boldsymbol{u}_k^*(\cdot)$. The choice of the initial state conditions provides us with a coupling of the state prediction and the real state. We apply the optimal control $\boldsymbol{u}_k^*(\cdot)$ only for the sampling time period δ. At the subsequent sampling time $t_\mathrm{s}^{k+1} = t_\mathrm{s}^k + \delta$ we solve a new OCP on the horizon $[t_\mathrm{s}^{k+1}, t_\mathrm{s}^{k+1} + H]$ with updated initial state conditions x_s^{k+1} and apply the obtained optimal control $\boldsymbol{u}_{k+1}^*(\cdot)$.
Applied to OCP (6.14) the MPC feedback approach solves the following OCP for each sampling time t_s^k

$$\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot)} \quad \varphi(t_\mathrm{s}^k + H, \boldsymbol{x}(t_\mathrm{s}^k + H)) + \int_{t_\mathrm{s}^k}^{t_\mathrm{s}^k+H} \psi(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) \,\mathrm{d}t \tag{C.1a}$$

$$\text{s.t.} \quad \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), \qquad t \in \mathcal{T}, \tag{C.1b}$$

$$\boldsymbol{x}(t_s^k) = x_s^k, \tag{C.1c}$$
$$\mathbf{0}_{n_c} \geqslant \boldsymbol{c}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), \qquad t \in \mathcal{T}, \tag{C.1d}$$
$$\mathbf{0}_{n_r} \geqslant \boldsymbol{r}(t_s^k + H, \boldsymbol{x}(t_s^k + H)). \tag{C.1e}$$

MPC subject to OCPs with quadratic objective function and affine–linear dynamic equations and inequality constraints are referred to as *Linear Model Predictive Control (LMPC)*. In case of nonlinear dynamic equations or inequality constraints, or if the objective function is nonlinear but not quadratic we call it *NMPC*.

As opposed to PID controllers, which consider the deviation to a desired reference value exclusively, MPC is able to predict the process behavior by means of a (nonlinear) model and a suitable chosen objective function. This allows for a notable improved feedback performance. MPC also has a wider range of applications as it is not restricted to linear models and quadratic objective functions like LQR controllers. Compared to other approaches the problem formulation in the MPC approach can be augmented with equality and inequality constraints. Furthermore, it allows for a broader spectrum of objective functions such as time–optimal and economic feedback control.

For real world applications it is impossible to obtain the real process state x_s^k exactly. Instead one has to rely on measurements of some components or of quantities which can be used to determine the remaining components. These measurements underlie measurement errors which is why only an estimate of the real process state can be obtained. Algorithms dealing with this kind of issues are subsumed as *state estimation* algorithms. Approaches to state estimation are beyond the scope of this contribution and we refer the reader to the excellent introduction presented by Wirsching [451] or the standard textbook by Rawlings et al. [370, Chapter 4].

C.3 Real–Time Iteration Scheme for NMPC

State–of–the–art approaches applied to NMPC problems solve an OCP in each sample. Black box OCP solvers based on discretization strategies as introduced in Chapter 6 are used to do this task. Real–world applications such as processes from chemical engineering result very often in large–scale models and as a consequence thereof in large–scale NLPs. The calculation of NLP solutions can be quite time–consuming. Therefore, especially for fast process dynamics, the feedback control obtained from the NLP solver will most likely be outdated or sometimes even infeasible.

To overcome the aforementioned issues it is crucial to reduce the time that is spent to compute the feedback control. The *RTI* approach which is sketched in this and extended in the following section avoids computing OCP solutions numerically up to a certain accuracy but uses the fact that consecutive OCPs of a NMPC loop are closely related. Hence, it reutilizes gained information from a NMPC step to the next one.

C.3.1 Tangential Predictors

We have seen in Section 6.2 that a Multiple Shooting discretization of the k–th NMPC sample OCP (C.1) yields the NLP (6.20) where x_s is equal to x_s^k. For now we suppress the term $[\boldsymbol{I}_{n_x}\mathbf{0}\dots\mathbf{0}]^T x_s^k$, and express the fact that the equality constraints depend on the initial value by adding x_s^k as additional function argument:

$$\begin{aligned} \min_{w^k} \quad & \boldsymbol{F}(w^k) \\ \text{s.t.} \quad & \mathbf{0} = \overline{\boldsymbol{G}}(w^k, x_s^k), \\ & \mathbf{0} \geqslant \boldsymbol{H}(w^k). \end{aligned} \tag{C.2}$$

Hence, these NLPs differ from sample k to sample $k+1$ only by an updated process state x_s^{k+1} compared to x_s^k. To express the fact that the solution of NLP (C.2) depends on the choice of the parameter x_s^k we call it *parametric programming*.

In order to facilitate the subsequent considerations let us suppose that there are no inequality constraints present in NLP (C.2). With Lagrange multipliers λ^k for the equality constraints we can write the first–order KKT conditions of the resulting NLP as

$$\boldsymbol{K}(w^k, \lambda^k, x_s^k) \overset{\text{def}}{=} \begin{bmatrix} \nabla \boldsymbol{F}(w^k) - \nabla \overline{\boldsymbol{G}}(w^k, x_s^k)\lambda^k \\ \overline{\boldsymbol{G}}(w^k, x_s^k) \end{bmatrix} = \mathbf{0}.$$

The Implicit Function Theorem states under certain smoothness assumptions that in a neighborhood of x_s^k there exist functions $\boldsymbol{w}(\cdot)$ and $\boldsymbol{\lambda}(\cdot)$ such that $\boldsymbol{w}(x_s^k) = w^k$ and $\boldsymbol{\lambda}(x_s^k) = \lambda^k$, and for all x_s in this neighborhood of x_s^k it holds $\boldsymbol{K}(\boldsymbol{w}(x_s), \boldsymbol{\lambda}(x_s), x_s) = \mathbf{0}$. Now consider the process state x_s^k in sample k and another point x_s. Then Taylor's theorem states for sufficiently smooth $\boldsymbol{w}(\cdot)$ that

$$\boldsymbol{w}(x_s) = \boldsymbol{w}\left(x_s^k\right) + \frac{\mathrm{d}}{\mathrm{d}x}\boldsymbol{w}\left(x_s^k\right)\left(x_s - x_s^k\right) + \mathcal{O}\left(\left\|x_s - x_s^k\right\|^2\right).$$

For this reason a good first–order approximation of $\boldsymbol{w}(x_s)$ for x_s close to x_s^k can be realized by the *tangential predictor* $\boldsymbol{t}(x_s) \overset{\text{def}}{=} \boldsymbol{w}\left(x_s^k\right) + \boldsymbol{w}'\left(x_s^k\right)\left(x_s - x_s^k\right)$. One of our goals is it to reduce the feedback time. For fast feedback we can assume process state x_s^{k+1} to be close to the preceding process state x_s^k. Applying the tangential predictor results in

$$\boldsymbol{w}\left(x_s^{k+1}\right) \simeq \boldsymbol{t}\left(x_s^{k+1}\right) = \boldsymbol{w}\left(x_s^k\right) + \frac{\mathrm{d}}{\mathrm{d}x}\boldsymbol{w}\left(x_s^k\right)\left(x_s^{k+1} - x_s^k\right).$$

By construction the value $\boldsymbol{w}\left(x_s^k\right)$ is equal to w^k, and $\boldsymbol{w}'\left(x_s^k\right)$ does not depend on x_s^{k+1}. Thus it can be computed before the new process state x_s^{k+1} is available. Since $\boldsymbol{w}(\cdot)$ is implicitly defined we still have to answer the open question how $\boldsymbol{w}'\left(x_s^k\right)$ can be calculated. According

to the Implicit Function Theorem the derivative of $\boldsymbol{w}(\cdot)$ is given as

$$\frac{\mathrm{d}}{\mathrm{d}x}\left[\boldsymbol{w}\left(x_{\mathrm{s}}^{k}\right), \boldsymbol{\lambda}\left(x_{\mathrm{s}}^{k}\right)\right] = -\frac{\partial}{\partial(w,\lambda)}\boldsymbol{K}\left(w^{k}, \lambda^{k}, x_{\mathrm{s}}^{k}\right)^{-1}\frac{\partial}{\partial x_{\mathrm{s}}}\boldsymbol{K}\left(w^{k}, \lambda^{k}, x_{\mathrm{s}}^{k}\right),$$

where we assume $\frac{\partial}{\partial(w,\lambda)}\boldsymbol{K}\left(w^{k}, \lambda^{k}, x_{\mathrm{s}}^{k}\right)$ to be invertible.
Till now we were dealing with equality constrained problems exclusively, but in case there are inequality constraints $\boldsymbol{H}(\cdot)$ present the first–order optimality conditions are no longer a set of equations only. However, it was shown in Diehl [131] that it is also possible to determine a tangential predictor for this problem type, but it is not given by solving a linear equation but by solving a QP. In the presence of inequality constraints it arises the question how active set changes are handled. In case of very close samples the active set of the solution does not change and the determination of a tangential predictor is made in the same way as in the equality constrained case. But even in case of active set changes these are taken into account since QPs incorporate linearized inequality constraints.

C.3.2 Initial Value Embedding

By augmenting the variable y^{k} with the initial value constraint, NLP (C.2) can be written in the following form:

$$\begin{aligned}
\min_{y^{k}, w^{k}} \quad & \boldsymbol{F}(w^{k}) \\
\text{s.t.} \quad & \mathbf{0} = y^{k} - x_{\mathrm{s}}^{k}, \\
& \mathbf{0} = \overline{\boldsymbol{G}}(w^{k}, y^{k}), \\
& \mathbf{0} \geqslant \boldsymbol{H}(w^{k}).
\end{aligned} \tag{C.3}$$

Let y^{k}, w^{k}, λ^{k} and μ^{k} be a primal–dual solution of the corresponding KKT–conditions for the initial value x_{s}^{k}. One can show that the tangential predictor for the $(k+1)$–th sample x_{s}^{k+1} is provided by a QP of the form

$$\begin{aligned}
\min_{\Delta y, \Delta w} \quad & \frac{1}{2}\begin{bmatrix}\Delta w \\ \Delta y\end{bmatrix}^{T}\mathcal{H}(w^{k}, y^{k}, \lambda^{k}, \mu^{k})\begin{bmatrix}\Delta w \\ \Delta y\end{bmatrix} + \nabla \boldsymbol{F}(w^{k})^{T}\Delta w \\
\text{s.t.} \quad & \mathbf{0} = \Delta y + (y^{k} - x_{\mathrm{s}}^{k+1}), \\
& \mathbf{0} = \frac{\partial}{\partial y}\overline{\boldsymbol{G}}(w^{k}, y^{k})\,\Delta y + \frac{\partial}{\partial w}\overline{\boldsymbol{G}}(w^{k}, y^{k})\,\Delta w + \overline{\boldsymbol{G}}(w^{k}, y^{k}), \\
& \mathbf{0} \geqslant \frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{H}(w^{k})\,\Delta w + \boldsymbol{H}(w^{k}),
\end{aligned} \tag{C.4}$$

where $\mathcal{H}(\cdot)$ denotes the Hessian of the Lagrangian with respect to w and y. The primal and dual steps calculated from QP (C.4) are used to update the solution y^{k}, w^{k}, λ^{k} and μ^{k} from sample k, and this is the tangential predictor step for sample $k+1$. If QP (C.4) is not initialized in the

primal–dual solution but with approximations for the Hessian $\mathcal{H}(\cdot)$, the constraint Jacobians or constraint residuals, then its solution becomes an approximated tangential predictor step.
Augmenting NLP (C.2) with the constraint $\mathbf{0} = y^k - x_s^k$ is called *Initial Value Embedding (IVE)*. The idea of IVE is to introduce the parameter x_s^k as additional NLP variable, and in this way derivative information with respect to the parameter can be gained for the QP.
Note that due to the initial value constraint (C.1c) NLPs arising from a Multiple Shooting discretization of the NMPC feedback generating OCPs (C.1) automatically have the tailored structure of NLP (C.3), see NLP (6.19) or NLP (6.20). Thus it is not necessary to introduce an additional variable y and QP (C.4) is replaced with

$$\begin{aligned} \min_{\Delta w} \quad & \frac{1}{2}\Delta w^T B^k \Delta w + \nabla \boldsymbol{F}(w^k)^T \Delta w \\ \text{s.t.} \quad & \mathbf{0} = \frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w^k)\,\Delta w + \boldsymbol{G}(w^k) + [\boldsymbol{I}_{n_x}\,\mathbf{0}\ldots\mathbf{0}]^T x_s^{k+1}, \\ & \mathbf{0} \geqslant \frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{H}(w^k)\,\Delta w + \boldsymbol{H}(w^k), \end{aligned} \tag{C.5}$$

where B^k is an appropriately chosen approximation of the Hessian e.g. using BFGS updates.

C.3.3 RTI for MPC

Classic NMPC approaches are realized by waiting for the actual process state and solving NLP (C.2) afterwards. It is impossible to find bounds for the number of iterations of the NLP solver which guarantees a certain accuracy of the solution. Therefore one usually has to work with worst–case guesses of the solution time. Moreover, while solving an NLP possibly outdated or even constraint violating controls have to be applied to the process.
In this section we use the results about Tangential Predictors (TPs) and IVE to adapt classic NMPC feedback strategies in two different ways: on the one hand the sampling intervals are reduced, and thus the controller does not need to work on outdated data. On the other hand the delay between obtaining the system state and updating the process control, which was calculated and is fed back, is diminished. An increased feedback frequency allows for operating the system closer to its constraint bounds and improves the handling time–critical processes.
Our considerations regarding TPs suggest for the k–th NMPC iteration to solve QP (C.5). Respective remarks on IVE show that the initial value x_s^k enters the QP only linearly. Consequently, all derivatives and almost all constraint evaluations that are required to set up QP (C.5) can be calculated without knowledge of the $(k+1)$–th sample x_s^{k+1}. These considerations motivate the following three steps for a real–time optimization approach:

(i) **Preparation Phase**: Set up QP (C.5) as far as possible without knowledge of the next sample x_s^{k+1}, i.e., if w^k, λ^k and μ^k denotes the primal–dual solution for current sample x_s^k, then the Hessian $\mathcal{H}(w^k, y^k, \lambda^k, \mu^k)$ or its approximation B^k, the constraint Jacobians $\frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w^k)$ and $\frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{H}(w^k)$, the objective function gradient $\nabla \boldsymbol{F}(w^k)$, and the constraint

residuals $\boldsymbol{G}(w^k)$ and $\boldsymbol{H}(w^k)$ can be evaluated. This step is computationally expensive since dynamic equations and their sensitivities with respect to w have to be evaluated.

(ii) **Feedback Phase**: As soon as x_s^{k+1} is available QP (C.5) is solved and one obtains the primal solution step $\Delta w = [(\Delta s)^T, (\Delta q)^T]^T$ as well as new multipliers. The control feedback $q^{k+1} = q^k + \Delta q$ is immediately fed back to the process. Compared to solving an NLP in classic NMPC approaches the feedback delay is reduced to solving one QP. The fact that the parameter x_s^{k+1} enters the QP affine–linearly can be exploited by tailored parametric QP solvers, cf. Ferreau et al. [159], Ferreau [158].

(iii) **Transition Phase**: By applying the step Δw we obtain the new set of NLP variables $(w^{k+1}, \lambda^{k+1}, \mu^{k+1})$. Next, in case there is some time available we execute more SQP steps in order to converge towards a local solution. Otherwise the preparation phase for the next sample is performed.

We call this approach RTI and it has been proposed and investigated by Diehl [131] and Diehl et al. [132]. Diehl et al. [133, 134] could also show the stability of RTI. Compared to classic NMPC approaches, where one waits for the sample x_s^{k+1} and cannot use free computational capacities otherwise, the RTI approach decouples the computation step (Preparation Phase) and the sampling step (Feedback Phase).

C.4 Multi–Level Iteration Scheme

In this section we present the *Multi–Level Iteration Schemes (MLIs)* as an extension to RTI introduced in the previous section. We have stressed the need for fast feedback in NMPC and have taken this fact into account in the RTI approach. The idea of MLI is to reduce the time spent in the computationally expensive preparation phase of RTI, where functions and derivatives have to be evaluated. The evaluation time in this phase depends among others on the size of the discretization grid, the size and nonlinearity of the system, and on the chosen horizon length. The MLI approach reduces the time spent in the preparation phase by introducing several controller levels. The different controller levels differ in their information updated in succeeding NMPC iterations. The levels range from a full RTI step on the highest level, i.e., all information are updated, to no information update on the lowest level. In consecutive NMPC iterations the chosen levels may vary. However, in this work we consider schemes with a fixed level and refer the interested reader to the dissertation of Wirsching [451, Chapter 6]. MLI was first introduced by Bock et al. [77], and its applicability to problems from mechanical and chemical engineering was shown in several publications, cf. Wirsching et al. [452, 453, 454]. An application of MLI to NMPC with long horizons can be found in the article of Kirches et al. [274].

C.4.1 Description of the MLI Levels

Independent of the level MLI splits its computational effort into a preparation phase, a feedback phase and a transition phase. The meaning of each phase is comparable to the one from RTI.

The feedback phase performs a single iteration per sample for feedback calculation. In the feedback phase of sample k a QP of the form

$$\begin{aligned} \min_{\Delta w} \quad & \frac{1}{2}\Delta w^T B^k \Delta w + \left(f^k\right)^T \Delta w \\ \text{s.t.} \quad & \mathbf{0} = G^k \Delta w + g^k + [I_{n_x}\mathbf{0}\dots\mathbf{0}]^T x_s^{k+1}, \\ & \mathbf{0} \geqslant H^k \Delta w + h^k, \end{aligned} \tag{C.6}$$

is solved. The different levels differ in how they update the values for B^k, G^k, H^k, f^k, g^k and h^k. Consequently, the computation time spent in the preparation phase varies for the different levels. The choice of the aforementioned matrices and vectors for the different levels is described in the following.

RTI (Level–D) Level–D iterations are RTI iterations, i.e., for the set of primal–dual variables $(w^{k,\mathrm{D}}, \lambda^{k,\mathrm{D}}, \mu^{k,\mathrm{D}})$ in each iteration, the objective gradient $f^k = \nabla \boldsymbol{F}(w^{k,\mathrm{D}})$, the constraints $g^k = \boldsymbol{G}(w^{k,\mathrm{D}})$, $h^k = \boldsymbol{H}(w^{k,\mathrm{D}})$, and the constraint Jacobians $G^k = \frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w^{k,\mathrm{D}})$, $H^k = \frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{H}(w^{k,\mathrm{D}})$ are evaluated. Furthermore, a Hessian (approximation) B^k must be provided.
After solving QP (C.6) in the feedback phase (with primal–dual QP solution $\Delta w^{k,\mathrm{D}}$, $\lambda^{k,\mathrm{QP}}$, $\mu^{k,\mathrm{QP}}$), the control feedback $q^{k+1,\mathrm{D}} = q^{k,\mathrm{D}} + \Delta q^{k,\mathrm{D}}$ is returned to the process. In the transition phase the primal and dual variables are updated as

$$w^{k+1,\mathrm{D}} = w^{k,\mathrm{D}} + \Delta w^{k,\mathrm{D}}, \quad \lambda^{k+1,\mathrm{D}} = \lambda^{k,\mathrm{QP}}, \quad \mu^{k+1,\mathrm{D}} = \mu^{k,\mathrm{QP}}.$$

The computationally most expensive steps in level–D iterations are the calculation of the Hessian (approximation) as well as the constraint Jacobians, in particular the sensitivities of the system dynamics.

Optimality Iterations (Level–C) Compared to level–D iterations *optimality iterations* omit the evaluation of the constraint Jacobians, which avoids especially computations of the system dynamics sensitivities. Level–C holds its own NLP variables $(w^{k,\mathrm{C}}, \lambda^{k,\mathrm{C}}, \mu^{k,\mathrm{C}})$, and additionally matrices B^k, G^k and H^k filled with Hessian and constraint Jacobian approximations. They are often provided by a previously executed level–D iteration. Each level–C iteration evaluates the constraints $g^k = \boldsymbol{G}(w^{k,\mathrm{C}})$ and $h^k = \boldsymbol{H}(w^{k,\mathrm{C}})$. Instead of the standard objective gradient there is a *modified gradient* which is given as

$$\begin{aligned} f^k &= \nabla \boldsymbol{F}(w^{k,\mathrm{C}}) + \left(G^{k\,T} - \frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w^{k,\mathrm{C}})\right)\lambda^{k,\mathrm{C}} + \left(H^{k\,T} - \frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{H}(w^{k,\mathrm{C}})\right)\mu^{k,\mathrm{C}} \\ &= \nabla \mathcal{L}(w^{k,\mathrm{C}}, \lambda^{k,\mathrm{C}}, \mu^{k,\mathrm{C}}) + G^{k\,T}\lambda^{k,\mathrm{C}} + H^{k\,T}\mu^{k,\mathrm{C}}. \end{aligned}$$

Even though the exact constraint Jacobians enter the modified gradient it can be evaluated efficiently by adjoint IND and the forward/reverse mode of automatic differentiation because they are just involved by a matrix–vector product, cf. Griewank and Walther [210] and Section 6.1. In level–C iterations the evaluation of the modified gradient or more specifically

the Lagrange gradient $\nabla\mathcal{L}(w^{k,\mathrm{C}}, \lambda^{k,\mathrm{C}}, \mu^{k,\mathrm{C}})$ takes the bulk of computational effort per sample with not more than five times the cost of evaluating both the objective function and the constraints.
After solving QP (C.6) in the feedback phase (with primal–dual QP solution $\Delta w^{k,\mathrm{C}}$, $\lambda^{k,\mathrm{QP}}$, $\mu^{k,\mathrm{QP}}$), the control feedback $q^{k+1,\mathrm{C}} = q^{k,\mathrm{C}} + \Delta q^{k,\mathrm{C}}$ is returned to the process. In the transition phase the primal and dual variables are updated according to

$$w^{k+1,\mathrm{C}} = w^{k,\mathrm{C}} + \Delta w^{k,\mathrm{C}}, \quad \lambda^{k+1,\mathrm{C}} = \lambda^{k,\mathrm{QP}}, \quad \mu^{k+1,\mathrm{C}} = \mu^{k,\mathrm{QP}}.$$

Feasibility Iterations (Level–B) *Feasibility iterations* omit any updates of derivative information. Analogously to level–C iterations level–B iterations hold their own NLP variables $(w^{k,\mathrm{B}}, \lambda^{k,\mathrm{B}}, \mu^{k,\mathrm{B}})$, and matrices B^k, G^k and H^k. Furthermore, level–B holds a fixed reference objective gradient $\overline{f}$ and a fixed reference primal variable $\overline{w}$. Both of them are usually provided by previously level–D or level–C iterations. The constraints $g^k = \boldsymbol{G}(w^{k,\mathrm{B}})$ and $h^k = \boldsymbol{H}(w^{k,\mathrm{B}})$ are evaluated in each level–B iteration. An objective gradient approximation is given by

$$f^k = \overline{f} + B^k\left(w^{k,\mathrm{B}} - \overline{w}\right).$$

After solving QP (C.6) in the feedback phase (with primal–dual QP solution $\Delta w^{k,\mathrm{B}}$, $\lambda^{k,\mathrm{QP}}$, $\mu^{k,\mathrm{QP}}$), the control feedback $q^{k+1,\mathrm{B}} = q^{k,\mathrm{B}} + \Delta q^{k,\mathrm{B}}$ is returned to the process. In the transition phase the primal and dual variables are updated as

$$w^{k+1,\mathrm{B}} = w^{k,\mathrm{B}} + \Delta w^{k,\mathrm{B}}, \quad \lambda^{k+1,\mathrm{B}} = \lambda^{k,\mathrm{QP}}, \quad \mu^{k+1,\mathrm{B}} = \mu^{k,\mathrm{QP}}.$$

The computational effort in level–B iterations is mostly spent in evaluating the constraints, in particular the integration of the system dynamics. It can be shown that primal–dual iterates of level–B iterations are driven towards a feasible but in general not to an optimal point.

Feedback Iterations (Level–A) In *feedback iterations* no QP data is updated at all. If a parametric QP solver is used for solving the NMPC QPs then there are no new matrix decompositions required, but it can be reused the one from a previous iteration. Level–A does not hold its own variables, and requires approximations B^k, G^k and H^k of Hessian and constraint Jacobians, as well as approximations f^k, g^k and h^k of the objective gradient and constraint residuals. There is also a fixed reference variable $\overline{w}$ necessary which is usually provided by previously executed higher level iterations (level–B – level–D).
After solving QP (C.6) in the feedback phase (with primal QP solution $\Delta w^{k,\mathrm{A}}$), the control feedback $\overline{q} + \Delta q^{k,\mathrm{A}}$ is returned to the process.

C.4.2 Convergence Analysis

Convergence for the presented MLI levels is usually analyzed under the assumption that the problem does not change ($x_\mathrm{s} = x_\mathrm{s}^k$, $k \geqslant 1$) during the NMPC loop. In this case it is obvious that level–D iterations are standard full SQP steps and the corresponding local convergence theory can be applied. Similarly we will see that level–B and level–C QPs can be interpreted

as steps of inexact SQP methods. Local convergence can then be derived from relevant theory about inexact SQP methods. This analysis will also explain the alternative notation of level–C and level–B as optimality iterations and feasibility iterations, respectively.

Stability of the QP Active Set Now we compare the active set of the QP (C.6) in a vicinity of the solution of NLP (C.2). Under rather mild assumptions the following result states that they coincide.

Theorem C.1 (Stability of the QP Active Set near NLP Solution)
Let (w^*, λ^*, μ^*) be a KKT point of NLP (C.2), and let assume that w^* is a regular point and the strict complementarity conditions holds at (w^*, λ^*, μ^*). Then for any constants $\alpha, \beta > 0$ there exists a neighborhood $\mathcal{N} = \mathcal{N}(\alpha, \beta)$ of (w^*, λ^*, μ^*) and a constant $\gamma > 0$ such that for all $(w, \lambda, \mu) \in \mathcal{N}$ the following statement holds: for any matrices $B \in \mathbb{R}^{n_w \times n_w}$, $G \in \mathbb{R}^{n_g \times n_w}$ and $H \in \mathbb{R}^{n_h \times n_w}$, where B is positive semidefinite, $\|G\|_F$, $\|H\|_F \leqslant \alpha$, and such that the matrix

$$J = \boldsymbol{J}(B, G, H) \stackrel{\text{def}}{=} \begin{bmatrix} B & -G^T & -\widetilde{H}^T \\ G & \mathbf{0} & \mathbf{0} \\ \widetilde{H} & \mathbf{0} & \mathbf{0} \end{bmatrix}, \qquad \widetilde{H} \stackrel{\text{def}}{=} \left(H_{j,\cdot}\right)_{j \in \mathcal{A}(w^*)}, \tag{C.7}$$

is invertible and it holds $\left\|J^{-1}\right\|_F \leqslant \beta$, the QP

$$\begin{aligned} \min_{\Delta w} \quad & \frac{1}{2} \Delta w^T B \Delta w + f^T \Delta w \\ \text{s.t.} \quad & \mathbf{0} = G \Delta w + g, \\ & \mathbf{0} \geqslant H \Delta w + h, \end{aligned} \tag{C.8}$$

with the modified objective gradient $f = \nabla_w \mathcal{L}(w, \lambda, \mu) + G^T \lambda + H^T \mu$ has a unique solution $(\Delta w, \lambda^{\text{QP}}, \mu^{\text{QP}})$ that satisfies

$$\left\|(\Delta w, \lambda^{\text{QP}}, \mu^{\text{QP}}) - (\mathbf{0}, \lambda^*, \mu^*)\right\| \leqslant \gamma \cdot \|(w^*, \lambda^*, \mu^*) - (w, \lambda, \mu)\| .$$

Furthermore, the QP solution has the same active set $\mathcal{A}$ as the NLP solution w^*. △

Proof See Wirsching [450, Theorem 4.1]. □

Local Convergence

Lemma C.2 (Local Convergence for Inexact Newton Methods)
Let $\mathcal{D} \subset \mathbb{R}^n$ be open and $\boldsymbol{\Phi} : \mathcal{D} \longrightarrow \mathbb{R}^n$ continuously differentiable. Let the sequence $\{x_n\}_{n \geqslant 0}$ be defined as

$$x_{n+1} = x_n + \Delta x_n, \quad \Delta x_n = -{J_n}^{-1} \boldsymbol{\Phi}(x_n),$$

where x_0 is chosen such that $x_0 \in \mathcal{D}$. Let us suppose that the following assumptions hold:

(i) The sequence of invertible matrices $\{J_n\}_{n \geqslant 0}$ is uniformly bounded with uniformly bounded inverse.

(ii) There exists a $\kappa < 1$ such that for all $n \in \mathbb{N}$ it holds

$$\left\| J_{n+1}^{-1} \left(J_n - \frac{\mathrm{d}}{\mathrm{d}x} \boldsymbol{\Phi}(x_n + t \Delta x_n) \right) \Delta x_n \right\| \leqslant \kappa \|\Delta x_n\|, \quad \forall t \in [0, 1].$$

(iii) The set

$$\mathcal{U}_{\frac{\|\Delta x_0\|}{1-\kappa}}(x_0) \stackrel{\text{def}}{=} \left\{ x \in \mathbb{R}^n \,:\, \|x - x_0\| \leqslant \frac{\|\Delta x_0\|}{1-\kappa} \right\}$$

is contained in $\mathcal{D}$.

Then the sequence $(x_n)_{n\in\mathbb{N}}$ remains in $\mathcal{D}$ and converges towards a point $x^* \in \mathcal{U}_{\frac{\|\Delta x_0\|}{1-\kappa}}(x_0)$ that satisfies $\boldsymbol{\Phi}(x^*) = 0$. In case it holds

$$\lim_{n\to\infty} \frac{\left\| J_{n+1}^{-1} \left(J_n - \frac{\mathrm{d}}{\mathrm{d}x} \boldsymbol{\Phi}(x^*) \right) \Delta x_n \right\|}{\|\Delta x_n\|} = 0,$$

the convergence rate is q–superlinear. △

Theorem C.3 (Local Convergence of Level–C Iterations)
Let (w^*, λ^*, μ^*) be a KKT point of NLP (C.2), and let

$$\boldsymbol{\Phi}(x) \stackrel{\text{def}}{=} \boldsymbol{\Phi}(w, \lambda, \mu) = \begin{bmatrix} \nabla_w \mathcal{L}(w, \lambda, \mu) \\ \boldsymbol{G}(w) \\ \widetilde{\boldsymbol{H}}(w) \end{bmatrix}$$

be the function that consists of the Lagrange gradient, the equality constraints, and the inequality constraints being active in the primal solution w^*. Let us suppose that the following assumptions hold:

(i) The primal solution w^* is a regular point and the strict complementarity condition holds at (w^*, λ^*, μ^*).

(ii) Let (B^k), (G^k) and (H^k) be uniformly bounded matrix sequences, where the B^k are positive semidefinite for all $k \in \mathbb{N}$. The sequence $\left(\left(J^k \right)^{-1} \right)$ is uniformly bounded with J^k being defined as $J^k \stackrel{\text{def}}{=} \boldsymbol{J}(B^k, G^k, H^k)$ (see (C.7)).

(iii) The sequence $x^k \stackrel{\text{def}}{=} (w^k, \lambda^k, \mu^k)$ is generated as

$$x^{k+1} = x^k + \Delta x^k, \qquad \Delta x^k \stackrel{\text{def}}{=} (\Delta w^k, \lambda^{k,\mathrm{QP}} - \lambda^k, \mu^{k,\mathrm{QP}} - \mu^k),$$

where Δw^k is the primal solution, and $\lambda^{k,\mathrm{QP}}$, $\mu^{k,\mathrm{QP}}$ is the dual solution of the QP

$$\begin{aligned} \min_{\Delta w} \quad & \frac{1}{2} \Delta w^T B^k \Delta w + \left(\nabla_w \mathcal{L}(w^k, \lambda^k, \mu^k) + \left(G^k \right)^T \lambda^k + \left(H^k \right)^T \mu^k \right)^T \Delta w \\ \text{s.t.} \quad & \mathbf{0} = G^k \Delta w + \boldsymbol{G}(w^k) + [\boldsymbol{I}_{n_x} \mathbf{0} \dots \mathbf{0}]^T x_{\mathrm{s}}^{k+1}, \\ & \mathbf{0} \geqslant H^k \Delta w + \boldsymbol{H}(w^k). \end{aligned} \tag{C.9}$$

(iv) There exists a $\kappa < 1$ such that for all $k \in \mathbb{N}$ it holds

$$\left\| \left(J^{k+1} \right)^{-1} \left(J^k - \frac{\mathrm{d}}{\mathrm{d}x} \boldsymbol{\Phi}(x^k + t \Delta x^k) \right) \Delta x^k \right\| \leqslant \kappa \left\| \Delta x^k \right\|, \quad \forall t \in [0, 1].$$

Then there exists a neighborhood $\mathcal{N}$ of (w^*, λ^*, μ^*) such that for any initial guess $(w^0, \lambda^0, \mu^0) \in \mathcal{N}$ the sequence (w^k, λ^k, μ^k) converges q–superlinearly towards (w^*, λ^*, μ^*) with convergence rate κ. The active sets of the primal solutions of QPs (C.9) and w^* coincide. △

Proof See Wirsching [450, Theorem 4.3]. □

Theorem C.3 shows the local convergence of level–C iterations if they are applied to the same problem in each iteration, i.e., if the initial values are chosen to be $x_s^k = x_s$ for all k. It is obvious that applying level–D iterations in each iteration results in standard full step SQP iterations. For this reason local convergence can either be concluded from respective convergence theory about SQP methods, or by applying Theorem C.3 with the appropriate choice for the constraints Jacobians. Note that the modified objective gradient becomes the objective gradient if the exact constraint Jacobians are chosen as constraint Jacobian approximations G^k and H^k.

Next we consider the case of level–B iterations: it has been shown by Bock et al. [76] that if level–B iterations are applied to the same problem ($x_s^k = x_s$) in each iteration and if they converge to a limit w^* (let (λ^*, μ^*) denote the associate level–B QP multipliers) then (w^*, λ^*, μ^*) is a KKT point of the problem

$$\begin{aligned} \min_{w^k} \quad & \frac{1}{2}(w^k - \overline{w})^T B^k (w^k - \overline{w}) + (\overline{f} + e^k)^T w^k \\ \text{s.t.} \quad & \mathbf{0} = \boldsymbol{G}(w^k), \\ & \mathbf{0} \geqslant \boldsymbol{H}(w^k), \end{aligned} \tag{C.10}$$

where $e^k \overset{\text{def}}{=} \left(\frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w^*) - G^{k^T}\right)\lambda^* + \left(\frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{H}(w^*) - H^{k^T}\right)\mu^*$, and B^k, G^k, H^k, $\overline{w}$, $\overline{f}$ are the Hessian approximation, the equality as well as inequality Jacobian approximations, the reference trajectory and the reference objective gradient, respectively (see description of level–B iterations).

Under the same conditions as and analogously to Theorem C.1 the stability of the active set can be proven in a neighborhood of the KKT points. In a similar way as in Theorem C.3 with $\boldsymbol{\Phi}(\cdot)$ being defined as

$$\boldsymbol{\Phi}(x) \overset{\text{def}}{=} \boldsymbol{\Phi}(w, \lambda, \mu) = \begin{bmatrix} \overline{f} + B^k(w^k - \overline{w}) - (G^k)^T\lambda - (\widetilde{H}^k)^T\mu \\ \boldsymbol{G}(w) \\ \widetilde{\boldsymbol{H}}(w) \end{bmatrix}$$

local convergence of level–B iterations can be shown. It is obvious that level–B iterations are feasible with respect to constraints of the original NLP, but in general they do no fulfill its KKT conditions. Hence, a convergence to an optimal point cannot be expected, and compared to the notation "optimality iterations" for level–C iterations the term "feasibility iterations" for level–B iterations is used.

Appendix D

Extensions to Multi–Level Iteration Schemes

In this chapter of the appendix, we develop a new level for the MLI approach (see Appendix C.4), which can be considered to be in between level–C and level–D iterations. This level is described in the remainder of this chapter.
Let us consider a sequence of NMPC samples $\left(x_{\mathrm{s}}^{k}\right)_{k\geqslant 1}$. Then classic NMPC means to solve a sequence of parametric NLPs that have the form

$$\begin{aligned} \mathrm{NLP}\left(x_{\mathrm{s}}^{k+1}\right): \quad & \min_{w} && \boldsymbol{F}(w) \\ & \text{s.t.} && \mathbf{0} = \boldsymbol{G}(w) + [\boldsymbol{I}_{n_{\mathrm{x}}}\, \mathbf{0} \ldots \mathbf{0}]^{T} x_{\mathrm{s}}^{k+1}, \\ & && \mathbf{0} \geqslant \boldsymbol{H}(w). \end{aligned}$$

These NLPs arise from from a Multiple Shooting discretization (compare NLP (6.20)). For an initial guess (w^0, λ^0, μ^0) the MLI approach generates iterates

$$w^{k+1} = w^k + \Delta w^k, \quad \lambda^{k+1} = \lambda^{k,\mathrm{QP}}, \quad \mu^{k+1} = \mu^{k,\mathrm{QP}},$$

where $(\Delta w^k, \lambda^{k,\mathrm{QP}}, \mu^{k,\mathrm{QP}})$ is the solution of the parametric quadratic program

$$\begin{aligned} \mathrm{QP}\left(x_{\mathrm{s}}^{k+1}\right): \quad & \min_{\Delta w} && \frac{1}{2} \Delta w^{T} B^{k} \Delta w + \left(f^{k}\right)^{T} \Delta w \\ & \text{s.t.} && \mathbf{0} = G^{k} \Delta w + \boldsymbol{G}(w^{k}) + [\boldsymbol{I}_{n_{\mathrm{x}}}\, \mathbf{0} \ldots \mathbf{0}]^{T} x_{\mathrm{s}}^{k+1}, \\ & && \mathbf{0} \geqslant H^{k} \Delta w + \boldsymbol{H}(w^{k}). \end{aligned} \tag{D.1}$$

We use the same notation as in Appendix C, i.e., B^k denotes (an approximation of) the Hessian of the Lagrangian, and f^k, G^k, H^k denote the (modified) objective gradient, the equality and inequality constraint Jacobians, respectively. In the preparation phase the following evaluations are necessary:

- Evaluations of Hessian approximation B^k, e.g. BFGS, L-BFGS, cf. Section 3.6.2.
- Evaluations of constraint residuals $\boldsymbol{G}(w^k)$, $\boldsymbol{H}(w^k)$ and (modified) objective gradient f^k.
  → Requires function evaluations and numerical integration.
- Evaluation of constraint Jacobians G^k and H^k.

Our new level focuses on the last point where we evaluate equality and inequality constraint Jacobians G^k and H^k. While the G^k are evaluated in each level–D iteration ($G^k = \frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w^k)$) they are not updated at all in level–C iterations.

Approximation of Constraint Jacobian

The idea of our new level is to calculate approximate equality constraint Jacobians in each iteration which can be calculate with low computational cost. To make things more clearly we suppress the NMPC index k in the following. Let us recall the constraint Jacobian

$$\frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w) = \begin{bmatrix} -\boldsymbol{I}_{n_\mathrm{x}} & & & & & & \\ X_0^s & X_0^q & -\boldsymbol{I}_{n_\mathrm{x}} & & & & \\ & & X_1^s & X_1^q & -\boldsymbol{I}_{n_\mathrm{x}} & & \\ & & & \ddots & & & \\ & & & & X_{N-1}^s & X_{N-1}^q & -\boldsymbol{I}_{n_\mathrm{x}} \end{bmatrix}$$

with X_n^s and X_n^q being defined as

$$X_n^s \overset{\text{def}}{=} \nabla_{s_n}\boldsymbol{x}_n(t_{n+1};s_n,q_n)^T \in \mathbb{R}^{n_\mathrm{x}\times n_\mathrm{x}}, \quad X_n^q \overset{\text{def}}{=} \nabla_{q_n}\boldsymbol{x}_n(t_{n+1};s_n,q_n,p)^T \in \mathbb{R}^{n_\mathrm{x}\times n_\mathrm{u}},$$

cf. (6.21). Using the notations $\boldsymbol{X}_n^s(t) \overset{\text{def}}{=} \nabla_{s_n}\boldsymbol{x}_n(t;s_n,q_n)^T$ and $\boldsymbol{X}_n^q(t) \overset{\text{def}}{=} \nabla_{q_n}\boldsymbol{x}_n(t;s_n,q_n,p)^T$ we define

$$\boldsymbol{X}_n(t) \overset{\text{def}}{=} \begin{bmatrix} \boldsymbol{X}_n^s(t) & \boldsymbol{X}_n^q(t) \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix}, \tag{D.2}$$

cf. (6.11). As we have shown in (6.12)+(6.13) $\boldsymbol{X}_n(\cdot)$ is the solution of the VDE

$$\dot{\boldsymbol{X}}_n(t) = \begin{bmatrix} f_x'(t,\boldsymbol{x}_n(t;s_n,q_n),q_n) & f_u'(t,\boldsymbol{x}_n(t;s_n,q_n),q_n) \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \boldsymbol{X}_n(t), \tag{D.3}$$

together with initial conditions

$$\boldsymbol{X}_n(t_n) = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix}. \tag{D.4}$$

Now we apply a 2–stage Runge–Kutta method to the IVP (D.3)+(D.4) for $n = 0,\dots,N-1$. Let us briefly recall the *trapezoidal rule* to solve the general IVP

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{g}(t,\boldsymbol{x}(t)), \quad t \in [t_\mathrm{s}, t_\mathrm{f}], \tag{D.5}$$

$$\boldsymbol{x}(t_\mathrm{s}) = x_s. \tag{D.6}$$

The trapezoidal method applied to IVP (D.5)+(D.6) with discretization grid $\{\tau^k\}$ generates the sequence of approximations $\{\eta^k\}$ by means of the evaluation rule

$$\eta^{k+1} = \eta^k + \frac{1}{2}h^k\left(\boldsymbol{g}(\tau^k,\eta^k) + \boldsymbol{g}(\tau^{k+1},\eta^{k+1})\right), \quad h^k = \tau^{k+1} - \tau^k.$$

Next we do the following:

- Apply trapezoidal rule to VDE on each Multiple Shooting interval $h_n \overset{\text{def}}{=} [t_n, t_{n+1}]$.
- Transform the implicit discretization approach to an explicit time stepping rule.

With definitions $\boldsymbol{A}_{11}[t] \overset{\text{def}}{=} \boldsymbol{f}_x'(t, \boldsymbol{x}_n(t;s_n,q_n),q_n)$ and $\boldsymbol{A}_{12}[t] \overset{\text{def}}{=} \boldsymbol{f}_u'(t, \boldsymbol{x}_n(t;s_n,q_n),q_n)$ and an application of the trapezoidal rule we write (D.3)+(D.4) as

$$\begin{aligned}\boldsymbol{X}_n(t_{n+1}) &= \boldsymbol{I} + \int_{t_n}^{t_{n+1}} \begin{bmatrix}\boldsymbol{A}_{11}[t] & \boldsymbol{A}_{12}[t]\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\boldsymbol{X}_n(t)\,\mathrm{d}t\\ &\simeq \boldsymbol{I} + \frac{1}{2}h_n\left(\begin{bmatrix}\boldsymbol{A}_{11}[t_n] & \boldsymbol{A}_{12}[t_n]\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\boldsymbol{X}_n(t_n) + \begin{bmatrix}\boldsymbol{A}_{11}[t_{n+1}] & \boldsymbol{A}_{12}[t_{n+1}]\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\boldsymbol{X}_n(t_{n+1})\right).\end{aligned}$$

Using the initial value condition (D.4) and collecting the $\boldsymbol{X}_n(t_{n+1})$ terms yields

$$\begin{aligned}&\left(\boldsymbol{I} - \frac{1}{2}h_n\begin{bmatrix}\boldsymbol{A}_{11}[t_{n+1}] & \boldsymbol{A}_{12}[t_{n+1}]\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\right)\boldsymbol{X}_n(t_{n+1}) = \boldsymbol{I} + \frac{1}{2}h_n\begin{bmatrix}\boldsymbol{A}_{11}[t_n] & \boldsymbol{A}_{12}[t_n]\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\\ \Leftrightarrow\quad &\boldsymbol{X}_n(t_{n+1}) = \left(\boldsymbol{I} - \frac{h_n}{2}\begin{bmatrix}\boldsymbol{A}_{11}[t_{n+1}] & \boldsymbol{A}_{12}[t_{n+1}]\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\right)^{-1}\left(\boldsymbol{I} + \frac{h_n}{2}\begin{bmatrix}\boldsymbol{A}_{11}[t_n] & \boldsymbol{A}_{12}[t_n]\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\right).\end{aligned}$$

Note that invertibility of the matrix on the right side of the equation holds under rather mild assumptions. One can think of bounded functions $\boldsymbol{f}_x'(\cdot)$ and $\boldsymbol{f}_u'(\cdot)$ such that the matrix can be inverted for sufficiently small values of h_n. Now we reformulate the inverse matrix term as

$$\begin{aligned}&\left(\boldsymbol{I} - \frac{1}{2}h_n\begin{bmatrix}\boldsymbol{A}_{11}[t_{n+1}] & \boldsymbol{A}_{12}[t_{n+1}]\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\right)^{-1} = \begin{bmatrix}\boldsymbol{I} - \frac{1}{2}h_n\boldsymbol{A}_{11}[t_{n+1}] & -\frac{1}{2}h_n\boldsymbol{A}_{12}[t_{n+1}]\\ \boldsymbol{0} & \boldsymbol{I}\end{bmatrix}^{-1}\\ &= \begin{bmatrix}\left(\boldsymbol{I} - \frac{1}{2}h_n\boldsymbol{A}_{11}[t_{n+1}]\right)^{-1} & \left(\boldsymbol{I} - \frac{1}{2}h_n\boldsymbol{A}_{11}[t_{n+1}]\right)^{-1}\left(\frac{1}{2}h_n\boldsymbol{A}_{12}[t_{n+1}]\right)\\ \boldsymbol{0} & \boldsymbol{I}\end{bmatrix}.\end{aligned}$$

We therefore arrive at

$$\boldsymbol{X}_n(t_{n+1}) = \begin{bmatrix}\left(\boldsymbol{I} - \frac{1}{2}h_n\boldsymbol{A}_{11}[t_{n+1}]\right)^{-1} & \left(\boldsymbol{I} - \frac{1}{2}h_n\boldsymbol{A}_{11}[t_{n+1}]\right)^{-1}\left(\frac{1}{2}h_n\boldsymbol{A}_{12}[t_{n+1}]\right)\\ \boldsymbol{0} & \boldsymbol{I}\end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{I} + \frac{1}{2} h_n \boldsymbol{A}_{11}[t_n] & \frac{1}{2} h_n \boldsymbol{A}_{12}[t_n] \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix}.$$

Using (D.2), we have approximations of $\boldsymbol{X}_n^s(t_{n+1})$ and $\boldsymbol{X}_n^q(t_{n+1})$ given by

$$\begin{aligned} \boldsymbol{X}_n^s(t_{n+1}) &\simeq \left(\boldsymbol{I} - \frac{1}{2} h_n \boldsymbol{A}_{11}[t_{n+1}]\right)^{-1} \left(\boldsymbol{I} + \frac{1}{2} h_n \boldsymbol{A}_{11}[t_n]\right) \quad \text{and} \\ \boldsymbol{X}_n^q(t_{n+1}) &\simeq \left(\boldsymbol{I} - \frac{1}{2} h_n \boldsymbol{A}_{11}[t_{n+1}]\right)^{-1} \frac{1}{2} h_n \left(\boldsymbol{A}_{12}[t_n] + \boldsymbol{A}_{12}[t_{n+1}]\right). \end{aligned}$$

Now we introduce the notation

$$\begin{aligned} N_n^s &= \boldsymbol{I}_{n_x} + \frac{1}{2} h_n \boldsymbol{f}_x'(t_n, s_n, q_n), \\ N_n^q &= \frac{1}{2} h_n \left(\boldsymbol{f}_u'(t_n, s_n, q_n) + \boldsymbol{f}_u'(t_{n+1}, \boldsymbol{x}_n(t_{n+1}; s_n, q_n), q_n)\right), \\ D_n &= \boldsymbol{I}_{n_x} - \frac{1}{2} h_n \boldsymbol{f}_x'(t_{n+1}, \boldsymbol{x}_n(t_{n+1}; s_n, q_n), q_n), \end{aligned}$$

and use it to find approximations of X_n^s and X_n^q by

$$X_n^s \simeq D_n^{-1} N_n^s, \quad X_n^q \simeq D_n^{-1} N_n^q.$$

Therefore an approximation of $\frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w)$ is given by

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w) &\simeq \begin{bmatrix} -I_{n_x} & & & & & & \\ D_0^{-1}N_0^s & D_0^{-1}N_0^q & -I_{n_x} & & & & \\ & & D_1^{-1}N_1^s & D_1^{-1}N_1^q & -I_{n_x} & & \\ & & & \ddots & & & \\ & & & & D_{N-1}^{-1}N_{N-1}^s & D_{N-1}^{-1}N_{N-1}^q & -I_{n_x} \end{bmatrix} \\ &= \begin{bmatrix} I_{n_x} & & & \\ & D_0 & & \\ & & D_1 & \\ & & & \ddots \\ & & & & D_{N-1} \end{bmatrix}^{-1} \begin{bmatrix} -I_{n_x} & & & & & \\ N_0^s & N_0^q & -D_0 & & & \\ & & N_1^s & N_1^q & -D_1 & \\ & & & \ddots & & \\ & & & & N_{N-1}^s & N_{N-1}^q & -D_{N-1} \end{bmatrix} = D^{-1}N = \overline{G}. \end{aligned}$$

Newton–Type Method Approach

Next we deal with the question how the equality constraint Jacobian G^k and the modified objective gradient f^k in QP (D.1) have to be chosen. To avoid cluttering in our notations, we omit inequality constraints as well as the linear initial value constraint and the NMPC index k in the following considerations. The RTI QP (see (C.5)) therefore becomes

$$\begin{aligned}\min_{\Delta w} \quad & \frac{1}{2}\Delta w^T B \Delta w + \nabla \boldsymbol{F}(w)^T \Delta w \\ \text{s.t.} \quad & \mathbf{0} = \frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w)\,\Delta w + \boldsymbol{G}(w).\end{aligned}$$

Now we use the equivalence of SQP steps and Newton steps as we have seen in Sections 3.6.1 and 3.6.2. The corresponding primal–dual Newton–step $(\Delta w, \Delta\lambda)$ is calculated by solving the linear system

$$\begin{bmatrix} B & -\frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w)^T \\ \frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w) & \mathbf{0} \end{bmatrix} \cdot \begin{bmatrix} \Delta w \\ \Delta\lambda \end{bmatrix} = - \begin{bmatrix} \nabla \boldsymbol{F}(w) - \frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w)^T \lambda \\ \boldsymbol{G}(w) \end{bmatrix}.$$

We want to use the fact that the Jacobian in a Newton step may be chosen rather arbitrarily, cf. Nocedal and Wright [341]. We replace the exact equality constraint Jacobian expressions $\frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w)$ with our approximation $\overline{G}$ and solve the nonlinear system

$$\begin{bmatrix} B & -\overline{G}^T \\ \overline{G} & \mathbf{0} \end{bmatrix} \cdot \begin{bmatrix} \Delta w \\ \Delta\lambda \end{bmatrix} = - \begin{bmatrix} \nabla \boldsymbol{F}(w) - \frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w)^T \lambda \\ \boldsymbol{G}(w) \end{bmatrix}.$$

In order to express the system in the primal step Δw and the updated dual variable $\lambda + \Delta\lambda$ we reformulate the first row as

$$\begin{aligned} & B\Delta w - \overline{G}^T \Delta\lambda = -\nabla \boldsymbol{F}(w) + \frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w)^T \lambda \\ \Longleftrightarrow \quad & B\Delta w - \overline{G}^T (\lambda + \Delta\lambda) = -\nabla \boldsymbol{F}(w) + \left(\frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w)^T - \overline{G}^T\right)\lambda. \end{aligned}$$

This results in the system

$$\begin{bmatrix} B & -\overline{G}^T \\ \overline{G} & \mathbf{0} \end{bmatrix} \cdot \begin{bmatrix} \Delta w \\ \lambda + \Delta\lambda \end{bmatrix} = - \begin{bmatrix} \nabla \boldsymbol{F}(w) - \left(\frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w)^T - \overline{G}^T\right)\lambda \\ \boldsymbol{G}(w) \end{bmatrix}. \tag{D.7}$$

Again we use the equivalence of SQP and Newton steps which results in the QP

$$\begin{aligned}\min_{\Delta w} \quad & \frac{1}{2}\Delta w^T B \Delta w + \left(\nabla \boldsymbol{F}(w) - \left(\frac{\mathrm{d}}{\mathrm{d}w}\boldsymbol{G}(w)^T - \overline{G}^T\right)\lambda\right)^T \Delta w \\ \text{s.t.} \quad & \mathbf{0} = \overline{G}\,\Delta w + \boldsymbol{G}(w).\end{aligned}$$

Putting everything together, the parametric QP (D.1) for our new MLI level is given by

$$\begin{aligned}\mathrm{QP}\left(x_{\mathrm{s}}^{k+1}\right): \quad \min_{\Delta w} \quad & \frac{1}{2}\Delta w^T B^k \Delta w + \boldsymbol{M}(w^k, \lambda^k)^T \Delta w \\ \text{s.t.} \quad & \mathbf{0} = \overline{G}^k\,\Delta w + \boldsymbol{G}(w^k) + [\boldsymbol{I}_{n_x}\,\mathbf{0}\ldots\mathbf{0}]^T x_{\mathrm{s}}^{k+1},\end{aligned}$$

$$\mathbf{0} \geqslant H^k \, \Delta w + \boldsymbol{H}(w^k),$$

where

$$\boldsymbol{M}(w^k, \lambda^k) \stackrel{\text{def}}{=} \nabla \boldsymbol{F}(w^k) + \overline{G}^T \lambda^k - \frac{\mathrm{d}}{\mathrm{d}w} \boldsymbol{G}(w^k)^T \lambda^k.$$

So far, we have shown which QP has to be solved in each level–T iteration. In order to evaluate the $\overline{G}^k = \left(D^k\right)^{-1} N^k$, it is necessary to calculate a matrix inverse. Due to numerical instabilities and the computational effort it is usually recommended to avoid the calculation of a matrix inverse. As we will show in the following this is not required in our case if we transform the constraint space of the QP to solve.

Transformation of Constraint Space

We show the transformation of the constraint space by reference to the equality constrained system (D.7), i.e., we investigate

$$\begin{bmatrix} B^k & -\left(\overline{G}^k\right)^T \\ \overline{G}^k & \mathbf{0} \end{bmatrix} \cdot \begin{bmatrix} \Delta w \\ \lambda^{k+1} \end{bmatrix} = - \begin{bmatrix} \boldsymbol{M}(w^k, \lambda^k) \\ \boldsymbol{G}(w^k) \end{bmatrix}. \tag{D.8}$$

Substituting $\overline{G}^k = \left(D^k\right)^{-1} N^k$ in the first row yields

$$\begin{aligned} & B^k \Delta w - \left(\overline{G}^k\right)^T \lambda^{k+1} = -\boldsymbol{M}(w^k, \lambda^k) \\ \Longleftrightarrow \quad & B^k \Delta w - \left(N^k\right)^T \left(\left(D^k\right)^{-T} \lambda^{k+1}\right) = -\boldsymbol{M}(w^k, \lambda^k). \end{aligned}$$

Likewise we substitute $\overline{G}^k = \left(D^k\right)^{-1} N^k$ in the second row and get

$$\begin{aligned} & \overline{G}^k \Delta w = -\boldsymbol{G}(w^k) \\ \Longleftrightarrow \quad & N^k \Delta w = -D^k \boldsymbol{G}(w^k). \end{aligned}$$

By introducing the auxiliary Lagrange multiplier $\tilde{\lambda}^{k+1} \stackrel{\text{def}}{=} \left(D^k\right)^{-T} \lambda^{k+1}$ we can write the nonlinear system (D.8) as

$$\begin{bmatrix} B^k & -\left(N^k\right)^T \\ N^k & \mathbf{0} \end{bmatrix} \cdot \begin{bmatrix} \Delta w \\ \tilde{\lambda}^{k+1} \end{bmatrix} = - \begin{bmatrix} \boldsymbol{M}(w^k, \lambda^k) \\ D^k \boldsymbol{G}(w^k) \end{bmatrix}.$$

Using this result we obtain the level–T feedback QP

$$\begin{aligned} \mathrm{QP}\left(x_{\mathrm{s}}^{k+1}\right): \qquad & \min_{\Delta w} \quad \frac{1}{2} \Delta w^T B^k \Delta w + \boldsymbol{M}(w^k, \lambda^k)^T \Delta w \\ & \text{s.t.} \quad \mathbf{0} = N^k \Delta w + D^k \boldsymbol{G}(w^k) + D^k \left[\boldsymbol{I}_{n_{\mathrm{x}}} \, \mathbf{0} \ldots \mathbf{0}\right]^T x_{\mathrm{s}}^{k+1}, \end{aligned}$$

$$\mathbf{0} \geqslant H^k \, \Delta w + \boldsymbol{H}(w^k).$$

Since the level–T iterations can be interpreted as a special case of level–C iterations, the convergence results for level–C (see Theorem C.3) also apply to the level–T iterates.

Danksagung

Zu allererst geht mein tief empfundener Dank an Hans Georg Bock, Johannes Schlöder und Andreas Potschka, die mir durch die Anstellung am IWR die einzigartige Möglichkeit gegeben haben, nahzu uneingeschränkt meinen Forscherdrang auf dem äußerst vielschichtigen und spannenden wissenschaftlichen Gebiet der optimalen Steuerung ausleben zu dürfen.
Ein weiterer Dank geht an all diejenigen Personen, welche durch rege wissenschaftliche Konversationen für neue Denkanstöße in meiner Forschung gesorgt haben und dadurch maßgeblich am erfolgreichen Fertigstellen dieser Arbeit beteiligt waren. Bei einer namentlichen Nennung würde ich mit an Sicherheit grenzender Wahrscheinlichkeit wesentliche Personen vergessen. Ich beschränke mich daher lediglich auf die Nennung der relevanten Arbeitsgruppen. Im einzelnen handelt es sich um die Mitglieder der Gruppen “Simulation und Optimierung”, “Numerische Optimierung”, “MOBOCON”, “OPTIMUS” und “Optimale Versuchsplanung”.
Den Co-Autoren meiner im Rahmen dieser Arbeit enstandenen Publikationen möchte ich für die wissenschaftlich fruchtvolle und inspirierende Zusammenarbeit danken. Namentlich handelt es sich um Nahid Azadfallah, Hans Georg Bock, Sebastian Engell, Daniel Haßkerl, Christian Kirches, Ekaterina A. Kostina, Clemens Lindscheid, Andreas Potschka, Matthias Schlöder und Leonard Wirsching.
In Anbetracht der vielen administrativen Dinge, welche im Laufe der Jahre bewältigt werden mussten, möchte ich mich bei Jeannette Walsch, Anastasia Walter, Abir Al-Laham, Anja Vogel, Ramona Ludwig, Dorothea Heukäufer sowie Rebecca Paimann für ihre tatkräftige Unterstützung bedanken. Dem in Fragen der IT-Administration jederzeit hilfsbereiten Thomas Klöpfer gebührt mein besonderer Dank.
Herzlich bedanken möchte ich mich für das Kommentieren an kleineren oder größeren Teilen dieses Manuskripts in alphabetischer Reihenfolge bei Jürgen Gutekunst, Johannes Herold, Andreas Potschka und Matthias Schlöder.
Ohne die finanzielle Unterstützung durch das Bundesministerium für Bildung und Forschung (BMBF) im Rahmen des Projekts GOSSIP (grant n o 05M2013-GOSSIP) und durch den Europäischen Forschungsrat im Rahmen des Projekts MOBOCON (Adv. Inv. Grant MOBOCON 291 458) wäre ein Gelingen dieser Arbeit unvorstellbar gewesen. Dafür ein herzliches Vergelt’s Gott!

Heidelberg, im August 2019

Bibliography

[1] J. M. Abadie. On the Kuhn-Tucker Theorem. *Nonlinear Programming, J. M. Abadie, ed., John Wiley, New York, NY*, pages 21–36, 1967.

[2] T. Achterberg. *Constraint Integer Programming*. PhD Thesis, Berlin Institute of Technology, 2007.

[3] W. Achtziger and C. Kanzow. Mathematical programs with vanishing constraints: Optimality conditions and constraint qualifications. *Mathematical Programming*, 114(1): 69–99, 2007. doi: 10.1007/s10107-006-0083-3.

[4] W. Achtziger, T. Hoheisel, and C. Kanzow. A smoothing-regularization approach to mathematical programs with vanishing constraints. *Computational Optimization and Applications*, 55(3):733–767, 2013. doi: 10.1007/s10589-013-9539-6.

[5] C. R. Adams. The Space of Functions of Bounded Variation and Certain General Spaces. *Transactions of the American Mathematical Society*, 40(3):421–438, 1936. doi: 10.1090/s0002-9947-1936-1501882-8.

[6] R. A. Adams and J. J. F. Fournier. *Sobolev Spaces*, volume 140 of *Pure and Applied Mathematics*. Elsevier, 2003. ISBN 978-0-12-044143-3.

[7] M. Ainsworth and J. T. Oden. *A Posterori Error Estimation in Finite Element Analysis*. Wiley-Interscience, 2000. ISBN 0-471-29411-X.

[8] J. Albersmeyer. *Adjoint Based Algorithms and Numerical Methods for Sensitivity Generation and Optimization of Large Scale Dynamic Systems*. PhD Thesis, University Heidelberg, 2010.

[9] J. Albersmeyer and C. Kirches. The SolvIND webpage. url: http://www.solvind.org, 2007.

[10] R. Allgor and P. Barton. Mixed-integer dynamic optimization. I - Problem formulation. *Computers & Chemical Engineering*, 23(4):567–584, 1999. doi: 10.1016/S0098-1354(98)00294-4.

[11] F. Allgöwer and A. Zheng, editors. *Nonlinear Model Predictive Control*. Birkhäuser, Basel, 2000. ISBN 978-3-7643-6297-3.

[12] F. Allgöwer, R. Findeisen, and Z. Nagy. Nonlinear model predictive control: From theory to application. *J. Chin. Inst. Chem. Engrs*, 35:299–315, 2004.

[13] W. Alt. *Nichtlineare Optimierung: Eine Einführung in Theorie, Verfahren Und Anwendungen*. Aufbaukurs Mathematik (German Edition). Vieweg+Teubner Verlag, 2002. ISBN 978-3-528-03193-0.

[14] A. A. Andronov, A. A. Vitt, and S. Khaikin. *Theory of Oscillators*, volume 4 of *International Series of Monographs in Physics*. Pergamon Press, Oxford, 1966. ISBN 978-1-4831-6724-4.

[15] M. Anitescu, P. Tseng, and S. J. Wright. Elastic-mode algorithms for mathematical programs with equilibrium constraints: Global convergence and stationarity properties. *Mathematical Programming*, 110(2):337–371, 2007. doi: 10.1007/s10107-006-0005-4.

[16] U. M. Ascher and L. R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. Society for Industrial and Applied Mathematics, Philadelphia, 1998. ISBN 978-0-89871-412-8.

[17] U. M. Ascher, R. M. M. Mattheij, and R. D. Russell. *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*. Prentice-Hall Series in Computational Mathematics. Prentice-Hall, Englewood Cliffs, 1st edition, 1988. ISBN 0-13-627266-5.

[18] S. A. Attia, M. Alamir, and C. C. de Wit. Sub optimal control of switched nonlinear systems under location and switching constraints. *IFAC Proceedings Volumes*, 38(1):133–138, 2005. doi: 10.3182/20050703-6-cz-1902.00307.

[19] J. P. Aubin and A. Cellina. *Differential Inclusions: Set-Valued Maps and Viability Theory*. Springer-Verlag, Berlin Heidelberg, 1984. ISBN 978-3-642-69514-8.

[20] I. Babuška and A. Miller. The post-processing approach in the finite element method—part 1: Calculation of displacements, stresses and other higher derivatives of the displacements. *International Journal for Numerical Methods in Engineering*, 20(6):1085–1109, 1984. doi: 10.1002/nme.1620200610.

[21] I. Babuška and A. Miller. The post-processing approach in the finite element method—Part 2: The calculation of stress intensity factors. *International Journal for Numerical Methods in Engineering*, 20(6):1111–1129, 1984. doi: 10.1002/nme.1620200611.

[22] I. Babuška and A. Miller. The post-processing approach in the finite element method—Part 3:A posteriori error estimates and adaptive mesh selection. *International Journal for Numerical Methods in Engineering*, 20(12):2311–2324, 1984. doi: 10.1002/nme.1620201211.

[23] I. Babuška and W. C. Rheinboldt. A-posteriori error estimates for the finite element method. *International Journal for Numerical Methods in Engineering*, 12(10):1597–1615, 1978. doi: 10.1002/nme.1620121010.

[24] I. Babuška and W. C. Rheinboldt. Error Estimates for Adaptive Finite Element Computations. *SIAM Journal on Numerical Analysis*, 15(4):736–754, 1978. doi: 10.1137/0715049.

[25] I. Babuška and T. Strouboulis. *The Finite Element Method and Its Reliability*. Numerical Mathematics and Scientific Computation. Clarendon Press, New York, 2001. ISBN 978-0-19-850276-0.

[26] G. Bal, Y. Maday, L. F. Pavarino, and A. Toselli. A "Parareal" Time Discretization for Non-Linear PDE's with Application to the Pricing of an American Put. In *Lecture Notes in Computational Science and Engineering*, volume 23, pages 189–202. Springer, Berlin Heidelberg, 2002. ISBN 978-3-540-43413-9.

[27] E. Balas. Disjunctive Programming and a Hierarchy of Relaxations for Discrete Optimization Problems. *SIAM Journal on Algebraic Discrete Methods*, 6(3):466–486, 1985. doi: 10.1137/0606047.

[28] V. Bansal, V. Sakizlis, R. Ross, J. Perkins, and E. Pistikopoulos. New algorithms for mixed-integer dynamic optimization. *Computers & Chemical Engineering*, 27:647–668, 2003. doi: 10.1016/S0098-1354(02)00261-2.

[29] V. Bär. *Ein Kollokationsverfahren Zur Numerischen Lösung Allgemeiner Mehrpunktrandwertaufgaben Mit Schalt– Und Sprungbedingungen Mit Anwendungen in Der Optimalen Steuerung Und Der Parameteridentifizierung*. Master thesis, University of Bonn, 1983.

[30] J. F. Bard. Convex two-level optimization. *Mathematical Programming*, 40-40(1-3):15–27, 1988. doi: 10.1007/bf01580720.

[31] F. Bashforth and J. Adams. *An Attempt to Test the Theories of Capillary Action: By Comparing the Theoretical and Measured Forms of Drops of Fluid.* Cambridge University Press, 1883.

[32] I. Bauer. *Numerische Verfahren Zur Lösung von Anfangswertaufgaben Und Zur Generierung von Ersten Und Zweiten Ableitungen Mit Anwendungen Bei Optimierungsaufgaben in Chemie Und Verfahrenstechnik*. PhD Thesis, University Heidelberg, 1999.

[33] B. Baumrucker and L. Biegler. MPEC strategies for optimization of a class of hybrid dynamic systems. *Journal of Process Control*, 19(8):1248–1256, 2009. doi: 10.1016/j.jprocont.2009.02.006.

[34] B. Baumrucker, J. Renfro, and L. Biegler. MPEC problem formulations and solution strategies with chemical engineering applications. *Computers & Chemical Engineering*, 32(12):2903–2913, 2008. doi: 10.1016/j.compchemeng.2008.02.010.

[35] J. Bausa and G. Tsatsaronis. Dynamic Optimization of Startup and Load-Increasing Processes in Power Plants—Part I: Method. *Journal of Engineering for Gas Turbines and Power*, 123(1):246, 2001. doi: 10.1115/1.1286728.

[36] C. V. Beccari, G. Casciola, and S. Morigi. On multi-degree splines. *Computer Aided Geometric Design*, 58:8–23, 2017. doi: 10.1016/j.cagd.2017.10.003.

[37] R. Becker and R. Rannacher. A Feed-Back Approach to Error Control in Finite Element Methods: Basic Analysis and Examples. *East-West J. Numer. Math*, 4:237–264, 1996.

[38] R. Becker and R. Rannacher. Weighted a posteriori error control in FE methods. In *Lecture ENUMATH-95, Paris, Sept. 18-22, 1995, in: Proc. ENUMATH-97, Heidelberg, Sept. 28 - Oct.3, 1997 (H.G. Bock, et al., Eds.)*, pages 621–637, World Scientific Publ., Singapore, 1998.

[39] R. Becker and R. Rannacher. An optimal control approach to a posteriori error estimation in finite element methods. *Acta Numerica*, 10, 2001. doi: 10.1017/s0962492901000010.

[40] R. Becker, M. Braack, D. Meidner, R. Rannacher, and B. Vexler. Adaptive Finite Element Methods for PDE-Constrained Optimal Control Problems. In W. Jäger, R. Rannacher, and J. Warnatz, editors, *Reactive Flows, Diffusion and Transport*, pages 177–205, Berlin, Heidelberg, 2007. Springer. ISBN 978-3-540-28396-6.

[41] D. Beigel. *Efficient Goal-Oriented Global Error Estimation for BDF-Type Methods Using Discrete Adjoints*. PhD Thesis, University Heidelberg, 2012.

[42] D. Beigel, M. S. Mommer, L. Wirsching, and H. G. Bock. Approximation of weak adjoints by reverse automatic differentiation of BDF methods. *Numerische Mathematik*, 126(3):383–412, 2014. doi: 10.1007/s00211-013-0570-4.

[43] P. Belotti, C. Kirches, S. Leyffer, J. Linderoth, J. Luedtke, and A. Mahajan. Mixed-integer nonlinear optimization. *Acta Numerica*, 22:1–131, 2013. doi: 10.1017/s0962492913000032.

[44] P. Belotti, P. Bonami, M. Fischetti, A. Lodi, M. Monaci, A. Nogales-Gómez, and D. Salvagnin. On Handling Indicator Constraints in Mixed Integer Programming. *Computational Optimization and Applications*, 65(3):545–566, 2016. doi: 10.1007/s10589-016-9847-8.

[45] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, 1999. doi: 10.1016/S0005-1098(98)00178-2.

[46] A. Bemporad, G. F. Trecate, and M. Morari. Observability and Controllability of Piecewise Affine and Hybrid Systems. *IEEE Transactions on Automatic Control*, 45:1864–1876, 1999. doi: 10.1109/TAC.2000.880987.

[47] A. Bemporad, F. Borrelli, and M. Morari. Piecewise linear optimal controllers for hybrid systems. *Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No.00CH36334)*, 2:1190–1194, 2000. doi: 10.1109/ACC.2000.876688.

[48] A. Bemporad, S. D. Cairano, E. Henriksson, and K. H. Johansson. Hybrid model predictive control based on wireless sensor feedback: An experimental study. In *2007 46th IEEE Conference on Decision and Control*, pages 5062–5067, 2007. doi: 10.1109/CDC.2007.4434918.

[49] S. C. Bengea and R. A. DeCarlo. Optimal control of switching systems. *Automatica*, 41(1):11–27, 2005. doi: 10.1016/j.automatica.2004.08.003.

[50] M. Benko and H. Gfrerer. An SQP method for mathematical programs with complementarity constraints with strong convergence properties. *Kybernetika*, pages 169–208, 2016. doi: 10.14736/kyb-2016-2-0169.

[51] M. Benko and H. Gfrerer. An SQP method for mathematical programs with vanishing constraints with strong convergence properties. *Computational Optimization and Applications*, 67(2):361–399, 2017. doi: 10.1007/s10589-017-9894-9.

[52] D. A. Benson. *A Gauss Pseudospectral Transcription for Optimal Control*. PhD Thesis, Massachusetts Institute of Technology, 2005.

[53] D. A. Benson, G. T. Huntington, T. P. Thorvaldsen, and A. V. Rao. Direct Trajectory Optimization and Costate Estimation via an Orthogonal Collocation Method. *Journal of Guidance, Control, and Dynamics*, 29(6):1435–1440, 2006. doi: 10.2514/1.20478.

[54] H. Y. Benson, D. F. Shanno, and R. J. Vanderbei. Interior-Point Methods for Nonconvex Nonlinear Programming: Complementarity Constraints. Technical report, Operations Research and Financial Engineering Department, Princeton University, 2002.

[55] H. Y. Benson, A. Sen, D. F. Shanno, and R. J. Vanderbei. Interior-Point Algorithms, Penalty Methods and Equilibrium Problems. *Computational Optimization and Applications*, 34(2):155–182, 2006. doi: 10.1007/s10589-005-3908-8.

[56] Y. M. Berezansky, Z. G. Sheftel, and G. F. Us. *Functional Analysis: Vol. I*, volume 85 of *Operator Theory: Advances and Applications*. Birkhauser Verlag, Basel, 2011. ISBN 978-3-0348-9939-0.

[57] L. D. Berkovitz. *Optimal Control Theory*, volume 12 of *Applied Mathematical Sciences*. Springer-Verlag, New York, 1974. ISBN 978-1-4757-6097-2.

[58] L. D. Berkovitz and N. G. Medhin. *Nonlinear Optimal Control Theory*. Chapman & Hall/CRC Applied Mathematics & Nonlinear Science. CRC Press, Boca Raton, 2012. ISBN 978-1-4665-6026-0.

[59] C. Bernardi and Y. Maday. Properties of Some Weighted Sobolev Spaces and Application to Spectral Approximations. *SIAM Journal on Numerical Analysis*, 26(4):769–829, 1989. doi: 10.1137/0726045.

[60] C. Bernardi and Y. Maday. Polynomial interpolation results in Sobolev spaces. *Journal of Computational and Applied Mathematics*, 43(1-2):53–80, 1992. doi: 10.1016/0377-0427(92)90259-z.

[61] C. Bernardi and Y. Maday. Spectral methods. In P. G. Ciarlet and J. L. Lions, editors, *Techniques of Scientific Computing (Part 2)*, volume 5 of *Handbook of Numerical Analysis*, pages 209–485. Elsevier, 1997. ISBN 978-0-444-82278-9.

[62] J. T. Betts. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. Advances in Design and Control. Society for Industrial & Applied Mathematics, 2nd edition, 2010. ISBN 978-0-89871-688-7.

[63] J. T. Betts and W. P. Huffman. Application of sparse nonlinear programming to trajectory optimization. *Journal of Guidance, Control, and Dynamics*, 15(1):198–206, 1992. doi: 10.2514/3.20819.

[64] J. T. Betts and W. P. Huffman. Sparse optimal control software SOCS. Technical Report Mathematics and Engineering Analysis Technical Document MEA-LR-085, Boeing Information and Support Services, 1997.

[65] J. T. Betts and W. P. Huffman. Mesh refinement in direct transcription methods for optimal control. *Optimal Control Applications and Methods*, 19(1):1–21, 1998. doi: 10.1002/(sici)1099-1514(199801/02)19:1<1::aid-oca616>3.0.co;2-q.

[66] J. T. Betts and W. P. Huffman. Exploiting Sparsity in the Direct Transcription Method for Optimal Control. *Computational Optimization and Applications*, 14(2):179–201, 1999. doi: 10.1023/A:1008739131724.

[67] J. T. Betts, N. Biehn, and S. L. Campbell. Convergence of Nonconvergent IRK Discretizations of Optimal Control Problems with State Inequality Constraints. *SIAM Journal on Scientific Computing*, 23(6):1981–2007, 2002. doi: 10.1137/s1064827500383044.

[68] L. T. Biegler. Solution of dynamic optimization problems by successive quadratic programming and orthogonal collocation. *Computers & Chemical Engineering*, 8(3-4):243–247, 1984. doi: 10.1016/0098-1354(84)87012-x.

[69] H. G. Bock. *Numerische Berechnung Zustandsbeschränkter Optimaler Steuerungen Mit Der Mehrzielmethode*. Carl-Cranz-Gesellschaft, Heidelberg, 1978.

[70] 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.

[71] H. G. Bock. Numerical Treatment of Inverse Problems in Chemical Reaction Kinetics. In *Springer Series in Chemical Physics*, pages 102–125. Springer Berlin Heidelberg, 1981. doi: 10.1007/978-3-642-68220-9_8.

[72] H. G. Bock. Recent Advances in Parameteridentification Techniques for O.D.E. In *Numerical Treatment of Inverse Problems in Differential and Integral Equations*, pages 95–121. Birkhäuser Boston, 1983. doi: 10.1007/978-1-4684-7324-7_7.

[73] H. G. Bock. *Randwertproblemmethoden Zur Parameteridentifizierung in Systemen Nichtlinearer Differentialgleichungen*, volume 183 of *Bonner Mathematische Schriften*. Universität Bonn, Bonn, 1987.

[74] H. G. Bock and R. Longman. Computation of optimal controls on disjoint control sets for minimum energy subway operation. In *Proceedings of the American Astronomical Society. Symposium on Engineering Science and Mechanics*, Taiwan, 1982.

[75] H. G. Bock and K. J. Plitt. A Multiple Shooting Algorithm for Direct Solution of Optimal Control Problems. *IFAC Proceedings Volumes*, 17(2):1603–1608, 1984. doi: 10.1016/s1474-6670(17)61205-9.

[76] H. G. Bock, M. Diehl, E. Kostina, and J. P. Schlöder. Constrained Optimal Feedback Control of Systems Governed by Large Differential Algebraic Equations. In L. T. Biegler, O. Ghattas, M. Heinkenschloss, D. Keyes, and B. Bloemen Waanders, editors, *Real-Time PDE-Constrained Optimization*, pages 3–24. SIAM, 2007. ISBN 978-0-89871-621-4.

[77] H. G. Bock, M. Diehl, P. Kühl, E. Kostina, J. P. Schlöder, and L. Wirsching. Numerical Methods for Efficient and Fast Nonlinear Model Predictive Control. In R. Findeisen, F. Allgöwer, and L. T. Biegler, editors, *Assessment and Future Directions of Nonlinear Model Predictive Control*, pages 163–179. Springer, Berlin Heidelberg, 2007. ISBN 978-3-540-72699-9.

[78] H. G. Bock, C. Kirches, A. Meyer, and A. Potschka. Numerical solution of optimal control problems with explicit and implicit switches. *Optimization Methods and Software*, 33(3):450–474, 2018. doi: 10.1080/10556788.2018.1449843.

[79] F. Borrelli, M. Baotic, A. Bemporad, and M. Morari. An efficient algorithm for computing the state feedback optimal control law for discrete time hybrid systems. *Proceedings of the 2003 American Control Conference, 2003.*, 6:4717–4722, 2003. doi: 10.1109/ACC.2003.1242468.

[80] F. Borrelli, M. Baotic, A. Bemporad, and M. Morari. Dynamic programming for constrained optimal control of discrete-time linear hybrid systems. *Automatica*, 41(10): 1709–1721, 2005. doi: 10.1016/j.automatica.2005.04.017.

[81] J. P. Boyd. The optimization of convergence for chebyshev polynomial methods in an unbounded domain. *Journal of Computational Physics*, 45(1):43–79, 1982. doi: 10.1016/0021-9991(82)90102-4.

[82] J. P. Boyd. *Chebyshev and Fourier Spectral Methods*. Dover Books on Mathematics. Dover Publications, Mineola, N.Y., 2nd revised edition, 2001. ISBN 978-0-486-41183-5.

[83] U. Brandt-Pollmann. *Numerical Solution of Optimal Control Problems with Implicitly Defined Discontinuities with Applications in Engineering*. PhD Thesis, University Heidelberg, 2004.

[84] M. Branicky, V. Borkar, and S. Mitter. A unified framework for hybrid control: Model and optimal control theory. *IEEE Transactions on Automatic Control*, 43(1):31–45, 1998. doi: 10.1109/9.654885.

[85] K. E. Brenan and B. E. Engquist. Backward Differentiation Approximations of Nonlinear Differential/Algebraic Systems. *Mathematics of Computation*, 51(184):659–676, 1988. doi: 10.2307/2008768.

[86] K. E. Brenan and L. R. Petzold. The Numerical Solution of Higher Index Differential/Algebraic Equations by Implicit Methods. *SIAM Journal on Numerical Analysis*, 26 (4):976–996, 1989. doi: 10.1137/0726054.

[87] C. G. Broyden. The Convergence of a Class of Double-Rank Minimization Algorithms 1. General Considerations. *IMA Journal of Applied Mathematics*, 6(1):76–90, 1970. doi: 10.1093/imamat/6.1.76.

[88] A. E. Bryson and Y. C. Ho. *Applied Optimal Control: Optimization, Estimation, and Control.* A Halsted Press Book. Taylor & Francis, New York, London, 1975. ISBN 978-0-89116-228-5.

[89] A. Buchner. *Auf Dynamischer Programmierung Basierende Nichtlineare Modellprädiktive Regelung Für LKW.* Master thesis, University Heidelberg, 2010.

[90] R. Bulirsch. Die Mehrzielmethode zur numerischen Lösung von nichtlinearen Randwertproblemen und Aufgaben der optimalen Steuerung. Technical report, Carl-Cranz-Gesellschaft, Deutsches Zentrum für Luft- und Raumfahrt (DLR), Oberpfaffenhofen, Germany, 1971.

[91] J. Burgschweiger, B. Gnädig, and M. C. Steinbach. Optimization models for operative planning in drinking water networks. *Optimization and Engineering*, 10(1):43–73, 2009. doi: 10.1007/s11081-008-9040-8.

[92] C. Büskens and H. Maurer. SQP-methods for solving optimal control problems with control and state constraints: Adjoint variables, sensitivity analysis and real-time control. *Journal of Computational and Applied Mathematics*, 120(1-2):85–108, 2000. doi: 10.1016/s0377-0427(00)00305-8.

[93] M. Buss, M. Glocker, M. Hardt, O. von Stryk, R. Bulirsch, and G. Schmidt. Nonlinear Hybrid Dynamical Systems: Modeling, Optimal Control, and Applications. In S. Engell, G. Frehse, and E. Schnieder, editors, *Modelling, Analysis, and Design of Hybrid Systems*, volume 279 of *Lecture Notes in Control and Information Science*, pages 311–335. Springer, Berlin, Heidelberg, 2002. doi: 10.1007/3-540-45426-8_18.

[94] R. H. Byrd, N. I. Gould, J. Nocedal, and R. A. Waltz. An algorithm for nonlinear optimization using linear programming and equality constrained subproblems. *Mathematical Programming*, 100(1), 2003. doi: 10.1007/s10107-003-0485-4.

[95] R. H. Byrd, N. I. M. Gould, J. Nocedal, and R. A. Waltz. On the Convergence of Successive Linear-Quadratic Programming Algorithms. *SIAM Journal on Optimization*, 16(2):471–489, 2005. doi: 10.1137/s1052623403426532.

[96] C. Canuto. Boundary Conditions in Chebyshev and Legendre Methods. *SIAM Journal on Numerical Analysis*, 23(4):815–831, 1986. doi: 10.1137/0723052.

[97] C. Canuto and A. Quarteroni. Spectral and pseudo-spectral methods for parabolic problems with non periodic boundary conditions. *Calcolo*, 18(3):197–217, 1981. doi: 10.1007/bf02576357.

[98] C. Canuto and A. Quarteroni. Approximation Results for Orthogonal Polynomials in Sobolev Spaces. *Mathematics of Computation*, 38(157):67, 1982. doi: 10.2307/2007465.

[99] C. Canuto, M. Hussaini, A. Quarteroni, and J. Z. Thomas A. *Spectral Methods in Fluid Dynamics*. Scientific Computation. Springer, Berlin, Heidelberg, 1988. ISBN 978-3-540-52205-8.

[100] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang. *Spectral Methods: Fundamentals in Single Domains*. Scientific Computation. Springer, Berlin, Heidelberg, 2006. ISBN 3-540-30725-7.

[101] Y. Cao and L. Petzold. A Posteriori Error Estimation and Global Error Control for Ordinary Differential Equations by the Adjoint Method. *SIAM Journal on Scientific Computing*, 26(2):359–374, 2004. doi: 10.1137/s1064827503420969.

[102] P. Cerone and S. S. Dragomir. Approximation of the Stieltjes Integral and Applications in Numerical Integration. *Applications of Mathematics*, 51(1):37–47, 2006. doi: 10.1007/s10492-006-0003-0.

[103] A. Cervantes and L. Biegler. Large-scale DAE optimization using a simultaneous NLP formulation. *AIChE Journal*, 44(5):1038–1050, 1998. doi: 10.1002/aic.690440505.

[104] A. M. Cervantes-Peredo. *Stable Large-Scale Dae Optimization Using Simultaneous Approaches*. PhD Thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 2000.

[105] L. Cesari. *Optimization—Theory and Applications*, volume 17 of *Applications of Mathematics*. Springer, New York, 1983. ISBN 978-1-4613-8167-9. doi: 10.1007/978-1-4613-8165-5.

[106] P. Chebyshev. Complete collected works. *Vol. II, Moscov-Leningrad*, pages 23–51, 1947.

[107] H. Chen, A. Kremling, and F. Allgöwer. Nonlinear Predictive Control of a Benchmark CSTR. In *Proceedings of 3rd European Control Conference*, pages 3247–3252, 1995.

[108] J. Chen and M. Gerdts. Smoothing Technique of Nonsmooth Newton Methods for Control-State Constrained Optimal Control Problems. *SIAM Journal on Numerical Analysis*, 50(4):1982–2011, 2012. doi: 10.1137/110822177.

[109] Y. Chen and M. Florian. The nonlinear bilevel programming problem:formulations,regularity and optimality conditions. *Optimization*, 32(3):193–209, 1995. doi: 10.1080/02331939508844048.

[110] B. Christiansen, H. Maurer, and O. Zirn. Optimal control of a voice-coil-motor with Coulombic friction. In *47th IEEE Conference on Decision and Control*, pages 1557–1562, 2008. doi: 10.1109/CDC.2008.4739025.

[111] F. Clarke. *Functional Analysis, Calculus of Variations and Optimal Control*, volume 264 of *Graduate Texts in Mathematics*. Springer, London, 2013. ISBN 978-1-4471-4820-3.

[112] F. H. Clarke. *Optimization and Nonsmooth Analysis*. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, 1990. ISBN 978-0-89871-256-8.

[113] F. H. Clarke, Y. S. Ledyaev, R. J. Stern, and P. R. Wolenski. *Nonsmooth Analysis and Control Theory*, volume 178. Springer Science & Business Media, 2008.

[114] E. A. Coddington and N. Levinson. *Theory of Ordinary Differential Equations*. McGraw-Hill, New Delhi, 1984. ISBN 978-0-07-099256-6.

[115] T. F. Coleman, B. S. Garbow, and J. J. Moré. Software for Estimating Sparse Hessian Matrices. *ACM Trans. Math. Softw.*, 11(4):363–377, 1985. ISSN 0098-3500. doi: 10.1145/6187.6190.

[116] M. G. Cox. The Numerical Evaluation of B-Splines. *IMA Journal of Applied Mathematics*, 10(2):134–149, 1972. doi: 10.1093/imamat/10.2.134.

[117] A. Curnier. Unilateral Contact. In *New Developments in Contact Problems*, pages 1–54. Springer Vienna, 1999. doi: 10.1007/978-3-7091-2496-3_1.

[118] H. B. Curry and I. J. Schoenberg. On spline distributions and their limits: The Polya distribution functions. *Bull. Amer. Math. Soc.*, 53:1114, 1947.

[119] C. F. Curtiss and J. O. Hirschfelder. Integration of Stiff Equations. *Proceedings of the National Academy of Sciences of the United States of America*, 38(3):235–243, 1952. ISSN 00278424.

[120] J. Cuthrell and L. Biegler. Simultaneous optimization and solution methods for batch reactor control profiles. *Computers & Chemical Engineering*, 13(1-2):49–62, 1989. doi: 10.1016/0098-1354(89)89006-4.

[121] J. E. Cuthrell and L. T. Biegler. On the optimization of differential-algebraic process systems. *AIChE Journal*, 33(8):1257–1270, 1987. doi: 10.1002/aic.690330804.

[122] C. L. Darby, W. W. Hager, and A. V. Rao. An hp-adaptive pseudospectral method for solving optimal control problems. *Optimal Control Applications and Methods*, 32(4):476–502, 2010. doi: 10.1002/oca.957.

[123] C. L. Darby, D. Garg, and A. V. Rao. Costate Estimation using Multiple-Interval Pseudospectral Methods. *Journal of Spacecraft and Rockets*, 48(5):856–866, 2011. doi: 10.2514/1.a32040.

[124] C. L. Darby, W. W. Hager, and A. V. Rao. Direct Trajectory Optimization Using a Variable Low-Order Adaptive Pseudospectral Method. *Journal of Spacecraft and Rockets*, 48(3):433–445, 2011. doi: 10.2514/1.52136.

[125] P. Davis. *Interpolation and Approximation*. Dover Books on Mathematics. Dover Publications, New York, 1975. ISBN 978-0-486-62495-2.

[126] C. de Boor. On calculating with B-splines. *Journal of Approximation Theory*, 6(1):50–62, 1972. doi: 10.1016/0021-9045(72)90080-9.

[127] C. de Boor. *A Practical Guide to Splines*, volume 27 of *Applied Mathematical Sciences*. Springer, New York, 2001. ISBN 0-387-95366-3.

[128] 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(2):587–609, 2005. doi: 10.1137/04060754x.

[129] S. Dempe. Annotated Bibliography on Bilevel Programming and Mathematical Programs with Equilibrium Constraints. *Optimization*, 52(3):333–359, 2003. doi: 10.1080/02331930310001498 94.

[130] L. Dieci and L. Lopez. Sliding Motion in Filippov Differential Systems: Theoretical Results and a Computational Approach. *SIAM Journal on Numerical Analysis*, 47(3): 2023–2051, 2009. doi: 10.1137/080724599.

[131] M. Diehl. *Real-Time Optimization for Large Scale Nonlinear Processes*. PhD Thesis, University Heidelberg, 2001.

[132] M. Diehl, H. Bock, J. P. Schlöder, R. Findeisen, Z. Nagy, and F. Allgöwer. Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. *Journal of Process Control*, 12(4):577–585, 2002. doi: 10.1016/s0959-1524(01)00023-3.

[133] M. Diehl, R. Findeisen, H. Bock, F. Allgöwer, and J. Schlöder. Nominal stability of real-time iteration scheme for nonlinear model predictive control. *IEE Proceedings - Control Theory and Applications*, 152(3):296–308, 2005. doi: 10.1049/ip-cta:20040008.

[134] M. Diehl, R. Findeisen, and F. Allgöwer. A Stabilizing Real-Time Implementation of Nonlinear Model Predictive Control. In L. T. Biegler, O. Ghattas, M. Heinkenschloss, D. Keyes, and B. van Bloemen Waanders, editors, *Real-Time PDE-Constrained Optimization*, pages 25–52. Society for Industrial and Applied Mathematics, 2007. doi: 10.1137/1.9780898718935.ch2.

[135] S. P. Dirkse. MPEC world. Webpage, GAMS Development Corporation, 2001.

[136] S. P. Dirkse and M. C. Ferris. The path solver: A nommonotone stabilization scheme for mixed complementarity problems. *Optimization Methods and Software*, 5(2):123–156, 1995. doi: 10.1080/10556789508805606.

[137] A. L. Dontchev, W. W. Hager, and K. Malanowski. Error bounds for euler approximation of a state and control constrained optimal control problem1. *Numerical Functional Analysis and Optimization*, 21(5-6):653–682, 2000. doi: 10.1080/01630560008816979.

[138] J. Dormand and P. Prince. A reconsideration of some embedded Runge—Kutta formulae. *Journal of Computational and Applied Mathematics*, 15(2):203–211, 1986. doi: 10.1016/0377-0427(86)90027-0.

[139] H. L. Dret and B. Lucquin. *Partial Differential Equations: Modeling, Analysis and Numerical Approximation*, volume 168 of *International Series of Numerical Mathematics*. Birkhäuser, Basel, 2016. ISBN 978-3-319-27065-4.

[140] S. Dreyfus. *Dynamic Programming and the Calculus of Variations*. Mathematics in Science and Engineering. Rand Corporation, Santa Monica, 1965. ISBN 978-0-08-095527-8.

[141] T. A. Driscoll, N. Hale, and L. N. Trefethen. *Chebfun Guide*. Pafnuty Publications, Oxford, 2014.

[142] A. Dubovitskii and A. Milyutin. Extremum problems with constraints. *Sov. Math., Dokl.*, 4:452–455, 1963.

[143] A. Dubovitskii and A. Milyutin. Extremum problems in the presence of restrictions. *USSR Computational Mathematics and Mathematical Physics*, 5(3):1–80, 1965. doi: 10.1016/0041-5553(65)90148-5.

[144] M. A. Duran and I. E. Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming*, 36(3):307–339, 1986. ISSN 1436-4646. doi: 10.1007/BF02592064.

[145] G. Elnagar, M. Kazemi, and M. Razzaghi. The pseudospectral Legendre method for discretizing optimal control problems. *IEEE Transactions on Automatic Control*, 40(10):1793–1796, 1995. doi: 10.1109/9.467672.

[146] P. J. Enright and B. A. Conway. Discrete approximations to optimal trajectories using direct transcription and nonlinear programming. *Journal of Guidance, Control, and Dynamics*, 15(4):994–1002, 1992. doi: 10.2514/3.20934.

[147] W. Enright. Continuous numerical methods for ODEs with defect control. *Journal of Computational and Applied Mathematics*, 125(1):159–170, 2000. ISSN 0377-0427. doi: 10.1016/S0377-0427(00)00466-0.

[148] K. Eriksson, D. Estep, P. Hansbo, and C. Johnson. Introduction to Adaptive Methods for Differential Equations. *Acta Numerica*, 4:105–158, 1995. doi: 10.1017/S0962492900002531.

[149] K. Eriksson, D. Estep, P. Hansbo, and C. Johnson. *Computational Differential Equations*. Cambridge University Press, New York, 1996. ISBN 0-521-56738-6.

[150] D. Estep. A Posteriori Error Bounds and Global Error Control for Approximation of Ordinary Differential Equations. *SIAM Journal on Numerical Analysis*, 32(1):1–48, Feb. 1995. doi: 10.1137/0732001.

[151] F. Facchinei, H. Jiang, and L. Qi. A smoothing method for mathematical programs with equilibrium constraints. *Mathematical Programming*, 85(1):107–134, May 1999. doi: 10.1007/s101079900015a.

[152] F. Fahroo and I. Ross. Direct trajectory optimization by a Chebyshev pseudospectral method. In *Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No.00CH36334)*, Chicago, 2000. IEEE. doi: 10.1109/acc.2000.876945.

[153] F. Fahroo and I. Ross. Pseudospectral Methods for Infinite-Horizon Nonlinear Optimal Control Problems. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*. American Institute of Aeronautics and Astronautics, 2005. doi: 10.2514/6.2005-6076.

[154] F. Fahroo and I. M. Ross. Costate Estimation by a Legendre Pseudospectral Method. *Journal of Guidance, Control, and Dynamics*, 24(2):270–277, 2001. doi: 10.2514/2.4709.

[155] N. Falkner and G. Teschl. On the substitution rule for Lebesgue–Stieltjes integrals. *Expositiones Mathematicae*, 30(4):412–418, 2012. doi: 10.1016/j.exmath.2012.09.002.

[156] E. Fehlberg. Klassische Runge-Kutta-Formeln fünfter und siebenter Ordnung mit Schrittweiten-Kontrolle. *Computing*, 4(2):93–106, 1969. ISSN 1436-5057. doi: 10.1007/BF02234758.

[157] E. Fehlberg. Klassische Runge-Kutta-Formeln vierter und niedrigerer Ordnung mit Schrittweiten-Kontrolle und ihre Anwendung auf Wärmeleitungsprobleme. *Computing*, 6(1):61–71, 1970. ISSN 1436-5057. doi: 10.1007/BF02241732.

[158] H. Ferreau. *Model Predictive Control Algorithms for Applications with Millisecond Timescales*. PhD Thesis, K.U. Leuven, 2011.

[159] H. Ferreau, H. Bock, and M. Diehl. An online active set strategy to overcome the limitations of explicit MPC. *International Journal of Robust and Nonlinear Control*, 18(8):816–830, 2008. doi: 10.1002/rnc.1251.

[160] H. Ferreau, C. Kirches, A. Potschka, H. Bock, and M. Diehl. qpOASES: A parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation*, 6(4):327–363, 2014.

[161] A. Fiacco and G. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, 1990. ISBN 978-0-89871-254-4.

[162] A. F. Filippov. Differential Equations with Discontinuous Right Hand Side. *AMS Transl.*, 42:199–231, 1964.

[163] A. F. Filippov. *Differential Equations with Discontinuous Right Hand Side*. Kluwer Academic Publishers, Dordrecht, Boston, London, 1988. ISBN 978-90-481-8449-1.

[164] B. A. Finlayson. *The Method of Weighted Residuals and Variational Principles*. Society for Industrial and Applied Mathematics, 2013. doi: 10.1137/1.9781611973242.

[165] R. Fletcher. A New Approach to Variable Metric Algorithms. *The computer journal*, 13 (3):317–322, 1970. doi: 10.1093/comjnl/13.3.317.

[166] R. Fletcher. *Practical Methods of Optimization*. Wiley-Interscience, New York, NY, USA, 2 edition, 1987. ISBN 978-0-471-49463-8.

[167] R. Fletcher and S. Leyffer. Solving mathematical programs with complementarity constraints as nonlinear programs. *Optimization Methods and Software*, 19(1):15–40, 2004. doi: 10.1080/10556780410001654241.

[168] 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. doi: 10.1137/s1052623402407382.

[169] B. Fornberg. *A Practical Guide to Pseudospectral Methods*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 1998. ISBN 978-0-521-64564-5.

[170] J. Fourier. *Théorie Analytique de La Chaleur*. Chez Firmin Didot, père et fils, 1822.

[171] E. C. Francis. The Lebesgue-Stieltjes Integral. *Mathematical Proceedings of the Cambridge Philosophical Society*, 22(6):935–950, 1925. doi: 10.1017/S0305004100014535.

[172] C. Francolin. *Costate Estimation in Optimal Control Problems Using Orthogonal Collocation at Gaussian Quadrature Points*. PhD Thesis, University of Florida, 2013.

[173] C. C. Francolin, H. Hou, W. W. Hager, and A. V. Rao. Costate estimation of state-inequality path constrained optimal control problems using collocation at Legendre-Gauss-Radau points. In *52nd IEEE Conference on Decision and Control*. IEEE, Dec. 2013. doi: 10.1109/cdc.2013.6760913.

[174] M. Fukushima and P. Tseng. An Implementable Active-Set Algorithm for Computing a B-Stationary Point of a Mathematical Program with Linear Complementarity Constraints. *SIAM Journal on Optimization*, 12(3):724–739, 2002. doi: 10.1137/s1052623499363232.

[175] M. Fukushima, J.-S. Pang, M. Théra, and R. Tichatschke. Convergence of a Smoothing Continuation Method for Mathematical Progams with Complementarity Constraints. In *Ill-Posed Variational Problems and Regularization Techniques*, volume 477 of *Lecture Notes in Economics and Mathematical Systems*, pages 99–110. Springer, Berlin Heidelberg, 1999. ISBN 978-3-540-66323-2.

[176] H. Gajewski, K. Gröger, and K. Zacharias. *Nichtlineare Operatorgleichungen Und Operatordifferential-Gleichungen*, volume 38 of *Mathematische Lehrbücher Und Monographien*. Akad.-Verl., Berlin, 1974.

[177] M. J. Gander and S. Vandewalle. Analysis of the Parareal Time-Parallel Time-Integration Method. *SIAM Journal on Scientific Computing*, 29(2):556–578, 2007. doi: 10.1137/05064607x.

[178] C. E. Garcia and M. Morari. Internal model control. A unifying review and some new results. *Industrial & Engineering Chemistry Process Design and Development*, 21(2): 308–323, 1982. doi: 10.1021/i200017a016.

[179] D. Garg. *Advances in Global Pseudospectral Methods for Optimal Control*. PhD Thesis, University of Florida, 2011.

[180] D. Garg, M. Patterson, W. Hager, A. Rao, D. R. Benson, and G. T. Huntington. An Overview of Three Pseudospectral Methods for the Numerical Solution of Optimal Control Problems. *Adv. Astronaut. Sci.*, 135:1–17, 2009. http://vdol.mae.ufl.edu/ConferencePublications/unifiedFrameworkAAS.pdf. (Cited on pp. 893, 894).

[181] D. Garg, M. Patterson, W. W. Hager, A. V. Rao, D. A. Benson, and G. T. Huntington. A unified framework for the numerical solution of optimal control problems using pseudospectral methods. *Automatica*, 46(11):1843–1851, 2010. doi: 10.1016/j.automatica.2010.06.048.

[182] D. Garg, M. A. Patterson, C. Francolin, C. L. Darby, G. T. Huntington, W. W. Hager, and A. V. Rao. Direct trajectory optimization and costate estimation of finite-horizon and infinite-horizon optimal control problems using a Radau pseudospectral method. *Computational Optimization and Applications*, 49(2):335–358, 2011.

[183] C. W. Gear. *Numerical Initial Value Problems in Ordinary Differential Equations*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1971. ISBN 0-13-626606-1.

[184] C. W. Gear and L. R. Petzold. ODE Methods for the Solution of Differential/Algebraic Systems. *SIAM Journal on Numerical Analysis*, 21(4):716–728, 1984. doi: 10.1137/0721048.

[185] C. W. Geart and D. S. Watanabe. Stability and Convergence of Variable Order Multistep Methods. *SIAM Journal on Numerical Analysis*, 11(5):1044–1058, 1974. doi: 10.1137/0711080.

[186] C. Geiger and C. Kanzow. *Theorie Und Numerik Restringierter Optimierungsaufgaben*. Springer-Lehrbuch Masterclass. Springer Berlin Heidelberg, 2002. ISBN 978-3-540-42790-2.

[187] M. Gerdts. Solving mixed-integer optimal control problems by branch&bound: A case study from automobile test-driving with gear shift. *Optimal Control Applications and Methods*, 26(1):1–18, 2005. doi: 10.1002/oca.751.

[188] M. Gerdts. A variable time transformation method for mixed-integer optimal control problems. *Optimal Control Applications and Methods*, 27(3):169–182, 2006. doi: 10.1002/oca.778.

[189] M. Gerdts. *Optimal Control of Ordinary Differential Equations and Differential-Algebraic Equations*. Habilitation, Department of Mathematics, University of Bayreuth, 2006.

[190] M. Gerdts. *Optimal Control of ODEs and DAEs*. De Gruyter Textbook. De Gruyter, 2012. ISBN 978-3-11-024999-6.

[191] M. Gerdts and M. Kunkel. A globally convergent semi-smooth Newton method for control-state constrained DAE optimal control problems. *Computational Optimization and Applications*, 48(3):601–633, 2009. doi: 10.1007/s10589-009-9275-0.

[192] M. Gerdts and M. Kunkel. Convergence analysis of Euler discretization of control-state constrained optimal control problems with controls of bounded variation. *Journal of Industrial and Management Optimization*, 10(1):311–336, 2013. doi: 10.3934/jimo.2014.10.311.

[193] S. Gerschgorin. Über die Abgrenzung der Eigenwerte einer Matrix. *Izvestija Akademii Nauk SSSR, Serija Matematika*, 7(3):749–754, 1931.

[194] H. Gfrerer. Optimality Conditions for Disjunctive Programs Based on Generalized Differentiation with Application to Mathematical Programs with Equilibrium Constraints. *SIAM Journal on Optimization*, 24(2):898–931, 2014.

[195] J. W. Gibbs. Fourier's Series. *Nature*, 59(1539):606–606, Apr. 1899. doi: 10.1038/059606a0.

[196] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization*. Academic Press, London, 1981. ISBN 978-0-12-283950-4.

[197] P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization. *SIAM review*, 47(1):99–131, 2005.

[198] R. Goebel, R. G. Sanfelice, and A. R. Teel. Hybrid dynamical systems. *IEEE Control Systems Magazine*, 29(2):28–93, 2009. ISSN 1066-033X. doi: 10.1109/MCS.2008.931718.

[199] D. Goldfarb. A Family of Variable-Metric Methods Derived by Variational Means. *Mathematics of computation*, 24(109):23–26, 1970. doi: 10.1090/S0025-5718-1970-0258249-6.

[200] Q. Gong, I. M. Ross, W. Kang, and F. Fahroo. On the Pseudospectral Covector Mapping Theorem for Nonlinear Optimal Control. In *Proceedings of the 45th IEEE Conference on Decision and Control*. IEEE, 2006. doi: 10.1109/cdc.2006.377729.

[201] Q. Gong, F. Fahroo, and I. M. Ross. Spectral Algorithm for Pseudospectral Methods in Optimal Control. *Journal of Guidance, Control, and Dynamics*, 31(3):460–471, 2008. doi: 10.2514/1.32908.

[202] H. Gonzalez, R. Vasudevan, M. Kamgarpour, S. Sastry, R. Bajcsy, and C. Tomlin. A Numerical Method for the Optimal Control of Switched Systems. In *Proceedings of the 49th IEEE Conference on Decision and Control*, pages 7519–7526, 2010. doi: 10.1109/CDC.2010.5717469.

[203] H. Gonzalez, R. Vasudevan, M. Kamgarpour, S. Sastry, R. Bajcsy, and C. Tomlin. A Descent Algorithm for the Optimal Control of Constrained Nonlinear Switched Dynamical Systems. In *Proceedings of the 13th International Conference on Hybrid Systems: Computation and Control*, pages 51–60, 2010. doi: 10.1145/1755952.1755961.

[204] S. Göttlich, A. Potschka, and C. Teuber. A partial outer convexification approach to control transmission lines. *Optimization Online*, 2017.

[205] S. Göttlich, A. Potschka, and U. Ziegler. Partial Outer Convexification for Traffic Light Optimization in Road Networks. *SIAM Journal on Scientific Computing*, 39(1):B53–B75, 2017. doi: 10.1137/15m1048197.

[206] D. Gottlieb and S. A. Orszag. *Numerical Analysis of Spectral Methods*. Society for Industrial and Applied Mathematics, 1977. ISBN 978-0-89871-023-6. doi: 10.1137/1.9781611970425.

[207] F. Gould and J. W. Tolle. A Necessary and Sufficient Qualification for Constrained Optimization. *SIAM Journal on Applied Mathematics*, 20(2):164–172, 1971. doi: 10.1137/0120021.

[208] N. Gould, D. Orban, and P. Toint. Numerical methods for large-scale nonlinear optimization. *Acta Numerica*, 14:299–361, 2005. doi: 10.1017/s0962492904000248.

[209] M. Gräber, C. Kirches, H. G. Bock, J. P. Schlöder, W. Tegethoff, and J. Köhler. Determining the optimum cyclic operation of adsorption chillers by a direct method for periodic optimal control. *International Journal of Refrigeration*, 34(4):902–913, 2011. doi: 10.1016/j.ijrefrig.2010.12.021.

[210] A. Griewank and A. Walther. *Evaluating Derivatives*. Society for Industrial and Applied Mathematics, Philadelphia, 2 edition, 2008. ISBN 978-0-89871-659-7. doi: 10.1137/1.9780898717761.

[211] L. Grüne. An adaptive grid scheme for the discrete Hamilton-Jacobi-Bellman equation. *Numerische Mathematik*, 75(3):319–337, 1997. doi: 10.1007/s002110050241.

[212] M. Guignard. Generalized Kuhn–Tucker Conditions for Mathematical Programming Problems in a Banach Space. *SIAM Journal on Control*, 7(2):232–241, 1969.

[213] L. Gurobi Optimization. Gurobi Optimizer Reference Manual, 2018.

[214] W. W. Hager. Rates of Convergence for Discrete Approximations to Unconstrained Control Problems. *SIAM Journal on Numerical Analysis*, 13(4):449–472, 1976. doi: 10.1137/0713040.

[215] W. W. Hager. Runge-Kutta methods in optimal control and the transformed adjoint system. *Numerische Mathematik*, 87(2):247–282, 2000. doi: 10.1007/s002110000178.

[216] W. W. Hager, H. Hou, and A. V. Rao. Lebesgue constants arising in a class of collocation methods. *IMA Journal of Numerical Analysis*, 37(4):1884–1901, 2016. doi: 10.1093/imanum/drw060.

[217] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II.* Number 14 in Springer Series in Computational Mathematics. Springer-Verlag, Berlin Heidelberg, 2 edition, 1996. ISBN 978-3-540-60452-5.

[218] E. Hairer, M. Roche, and C. Lubich. *The Numerical Solution of Differential-Algebraic Systems by Runge-Kutta Methods.* Number 1409 in Lecture Notes in Mathematics. Springer-Verlag, Berlin Heidelberg, 1989. ISBN 978-3-540-51860-0.

[219] E. Hairer, S. P. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations I.* Number 8 in Springer Series in Computational Mathematics. Springer-Verlag, Berlin, Heidelberg, 2 edition, 1993. ISBN 978-3-540-56670-0.

[220] J. Hale. *Ordinary Differential Equations.* R.E. Krieger Pub. Co, Malabar, FL, 1980. ISBN 978-0-89874-011-0.

[221] S.-P. Han. Superlinearly convergent variable metric algorithms for general nonlinear programming problems. *Mathematical Programming*, 11(1):263–282, 1976. doi: 10.1007/bf01580395.

[222] S.-P. Han. A Globally Convergent Method for Nonlinear Programming. *Journal of optimization theory and applications*, 22(3):297–309, 1977.

[223] C. R. Hargraves and S. W. Paris. Direct trajectory optimization using nonlinear programming and collocation. *Journal of Guidance, Control, and Dynamics*, 10(4):338–342, 1987. doi: 10.2514/3.20223.

[224] 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.

[225] J. Haslinger and P. Neittaanmäki. *Finite Element Approximation for Optimal Shape Design: Theory and Applications.* Wiley, Chichester, 1988. ISBN 978-0-471-92079-3.

[226] D. Haßkerl, A. Meyer, N. Azadfallah, S. Engell, A. Potschka, L. Wirsching, and H. G. Bock. Study of the performance of the multi-level iteration scheme for dynamic online optimization for a fed-batch reactor example. In *2016 European Control Conference (ECC)*. IEEE, 2016. doi: 10.1109/ecc.2016.7810327.

[227] K. Hatz. *Efficient Numerical Methods for Hierarchical Dynamic Optimization with Application to Cerebral Palsy Gait Modeling.* PhD Thesis, University Heidelberg, 2014.

[228] W. Heemels. *Linear Complementarity Systems: A Study in Hybrid Dynamics.* PhD Thesis, Eindhoven Univ. of Technol., 1999.

[229] P. Henrici. *Discrete Variable Methods in Ordinary Differential Equations*. Wiley, New York, 1962. ISBN 978-0-471-37224-0.

[230] A. L. Herman and B. A. Conway. Direct optimization using collocation based on high-order Gauss-Lobatto quadrature rules. *Journal of Guidance, Control, and Dynamics*, 19 (3):592–599, 1996. doi: 10.2514/3.21662.

[231] M. R. Hestenes. *Calculus of Variations and Optimal Control Theory*. John Wiley & Sons, New York, 1966.

[232] J. S. Hesthaven. From Electrostatics to Almost Optimal Nodal Sets for Polynomial Interpolation in a Simplex. *SIAM Journal on Numerical Analysis*, 35(2):655–676, 1998. doi: 10.1137/s003614299630587x.

[233] J. S. Hesthaven. *Nodal Discontinuous Galerkin Methods*. Number 54 in Texts in Applied Mathematics. Springer-Verlag, New York, 2008. ISBN 978-0-387-72065-4.

[234] J. S. Hesthaven, P. S. Gottlieb, and D. Gottlieb. *Spectral Methods for Time-Dependent Problems*. Number 21 in Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, Cambridge, 2009. ISBN 978-0-511-61835-2.

[235] E. Hewitt and R. E. Hewitt. The Gibbs-Wilbraham phenomenon: An episode in fourier analysis. *Archive for History of Exact Sciences*, 21(2):129–160, 1979. doi: 10.1007/bf00330404.

[236] G. Hicks and W. Ray. Approximation Methods for Optimal Control Synthesis. *The Canadian Journal of Chemical Engineering*, 49(4):522–528, 1971. doi: 10.1002/cjce.5450490416.

[237] I. Hiskens and M. Pai. Trajectory sensitivity analysis of hybrid systems. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 47(2):204–220, 2000. doi: 10.1109/81.828574.

[238] T. Hoheisel. *Mathematical Programs with Vanishing Constraints*. PhD Thesis, University of Würzburg, 2009.

[239] T. Hoheisel and C. Kanzow. First-and Second-Order Optimality Conditions for Mathematical Programs with Vanishing Constraints. *Applications of Mathematics*, 52(6): 495–514, 2007. doi: 10.1007/s10492-007-0029-y.

[240] T. Hoheisel and C. Kanzow. Stationary Conditions for Mathematical Programs with Vanishing Constraints Using Weak Constraint Qualifications. *Journal of Mathematical Analysis and Applications*, 337(1):292–310, 2008. doi: 10.1016/j.jmaa.2007.03.087.

[241] T. Hoheisel and C. Kanzow. On the Abadie and Guignard Constraint Qualifications for Mathematical Programmes with Vanishing Constraints. *Optimization*, 58(4):431–448, 2009. doi: 10.1080/02331930701763405.

[242] 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. doi: 10.1007/s10107-011-0488-5.

[243] H. Hou. *Convergence Analysis of Orthogonal Collocation Methods for Unconstrained Optimal Control*. PhD Thesis, University of Florida, 2013.

[244] X. M. Hu and D. Ralph. Convergence of a Penalty Method for Mathematical Programming with Complementarity Constraints. *Journal of Optimization Theory and Applications*, 123(2):365–390, 2004. doi: 10.1007/s10957-004-5154-0.

[245] G. Huntington. *Advancement and Analysis of Gauss Pseudospectral Transcription for Optimal Control Problems*. PhD Thesis, Massachusetts Institute of Technology, 2007.

[246] G. Huntington, D. Benson, and A. Rao. A Comparison of Accuracy and Computational Efficiency of Three Pseudospectral Methods. In *AIAA Guidance, Navigation and Control Conference and Exhibit*. American Institute of Aeronautics and Astronautics, 2007. doi: 10.2514/6.2007-6405.

[247] H. T. Huynh. Collocation and Galerkin Time-Stepping Methods. In *19th AIAA Computational Fluid Dynamics*. American Institute of Aeronautics and Astronautics, 2009. doi: 10.2514/6.2009-4323.

[248] Ibm. IBM ILOG CPLEX Optimization Studio CPLEX User's Manual, 2011.

[249] J. Imura and A. Van Der Schaft. Characterization of well-posedness of piecewise-linear systems. *IEEE Transactions on Automatic Control*, 45(9):1600–1619, 2000. doi: 10.1109/9.880612.

[250] W. R. Inc. Mathematica, Version 11.3, 2018. Champaign, IL.

[251] A. D. Ioffe and V. M. Tikhomirov. *Theory of Extremal Problems*. North-Holland Pub. Co., Amsterdam, 1979. ISBN 0-444-85167-4.

[252] A. F. Izmailov and M. V. Solodov. Mathematical Programs with Vanishing Constraints: Optimality Conditions, Sensitivity, and a Relaxation Method. *Journal of Optimization Theory and Applications*, 142(3):501–532, 2009. doi: 10.1007/s10957-009-9517-4.

[253] A. F. Izmailov, A. L. Pogosyan, and M. V. Solodov. Semismooth Newton method for the lifted reformulation of mathematical programs with complementarity constraints. *Computational Optimization and Applications*, 51(1):199–221, 2010. doi: 10.1007/s10589-010-9341-7.

[254] D. Jackson. *The Theory of Approximation*, volume 11 of *Colloquium Publications*. American Mathematical Society, New York, 1930.

[255] R. H. F. Jackson. Factorable programming. In *Encyclopedia of Operations Research and Management Science*. Springer, Boston, MA, 2001. ISBN 978-1-4020-0611-1.

[256] D. Jacobson, M. Lele, and J. Speyer. New necessary conditions of optimality for control problems with state-variable inequality constraints. *Journal of Mathematical Analysis and Applications*, 35(2):255–284, 1971. doi: 10.1016/0022-247x(71)90219-8.

[257] S. Jain and P. Tsiotras. Trajectory Optimization Using Multiresolution Techniques. *Journal of Guidance, Control, and Dynamics*, 31(5):1424–1436, 2008. doi: 10.2514/1.32220.

[258] C. Johnson. Error Estimates and Adaptive Time-Step Control for a Class of One-Step Methods for Stiff Ordinary Differential Equations. *SIAM Journal on Numerical Analysis*, 25(4):908–926, 1988. doi: 10.1137/0725051.

[259] M. Jung. *Relaxations and Approximations for Mixed-Integer Optimal Control*. PhD Thesis, University Heidelberg, 2013.

[260] M. Jung, C. Kirches, and S. Sager. On Perspective Functions and Vanishing Constraints in Mixed-Integer Nonlinear Optimal Control. In *Facets of Combinatorial Optimization*, pages 387–417. Springer Berlin Heidelberg, 2013. doi: 10.1007/978-3-642-38189-8_16.

[261] S. Kameswaran and L. Biegler. Convergence rates for direct transcription of optimal control problems with final-time equality constraints using collocation at Radau points. In *2006 American Control Conference*. IEEE, 2006. doi: 10.1109/acc.2006.1655348.

[262] S. Kameswaran and L. Biegler. Convergence rates for direct transcription of optimal control problems using collocation at Radau points. *Computational Optimization and Applications*, 41(1):81–126, 2007. doi: 10.1007/s10589-007-9098-9.

[263] M. Kamgarpour and C. Tomlin. On optimal control of non-autonomous switched systems with a fixed mode sequence. *Automatica*, 48(6):1177–1181, 2012. doi: 10.1016/j.automatica.2012.03.019.

[264] R. Kannan and C. L. Monma. On the Computational Complexity of Integer Programming Problems. In *Lecture Notes in Economics and Mathematical Systems*, pages 161–172. Springer Berlin Heidelberg, 1978. doi: 10.1007/978-3-642-95322-4_17.

[265] L. V. Kantorovich. On a new method of approximate solution for equations in partial derivatives. *Dokl. Akad. Nauk. SSSR*, 4:532–536, 1934. (in Russian).

[266] C. Kanzow and A. Schwartz. A New Regularization Method for Mathematical Programs with Complementarity Constraints with Strong Convergence Properties. *SIAM Journal on Optimization*, 23(2):770–798, 2013. doi: 10.1137/100802487.

[267] N. Karmarkar. A New Polynomial Time Algorithm for Linear Programming. *Combinatorica*, 4(4):373–395, 1984. doi: https://doi.org/10.1007/BF02579150.

[268] W. Karush. *Minima of Functions of Several Variables with Inequalities as Side Conditions*. Master thesis, University of Chicago, 1939.

[269] G. Kedem. Automatic Differentiation of Computer Programs. *ACM Transactions on Mathematical Software*, 6(2):150–165, 1980. doi: 10.1145/355887.355890.

[270] E. Kerrigan and D. Mayne. Optimal control of constrained, piecewise affine systems with bounded disturbances. In *Proc. 41th IEEE Conf. on Decision and Control*, pages 1552–1557, 2002. doi: 10.1109/CDC.2002.1184740.

[271] C. Kirches. *A Numerical Method for Nonlinear Robust Optimal Control with Implicit Discontinuities and an Application to Powertrain Oscillations.* Diploma Thesis, Heidelberg University, 2006.

[272] C. Kirches. *Fast Numerical Methods for Mixed-Integer Nonlinear Model-Predictive Control.* PhD Thesis, Heidelberg University, 2010.

[273] C. Kirches, S. Sager, H. G. Bock, and J. P. Schlöder. Time-optimal control of automobile test drives with gear shifts. *Optimal Control Applications and Methods*, 31(2):137–153, 2010. doi: 10.1002/oca.892.

[274] C. Kirches, L. Wirsching, H. Bock, and J. Schlöder. Efficient direct multiple shooting for nonlinear model predictive control on long horizons. *Journal of Process Control*, 22 (3):540–550, 2012. doi: 10.1016/j.jprocont.2012.01.008.

[275] C. Kirches, A. Potschka, H. Bock, and S. Sager. A Parametric Active Set Method for a Subclass of Quadratic Programs with Vanishing Constraints. *Pacific Journal of Optimization*, 9(2):275–299, 2013.

[276] C. Kirches, E. A. Kostina, A. Meyer, and M. Schlöder. Numerical Solution of Optimal Control Problems with Switches, Switching Costs and Jumps. *Optimization Online Preprint 6888*, 2018.

[277] C. Kirches, E. A. Kostina, A. Meyer, and M. Schlöder. Generation of Optimal Walking-Like Motions Using Dynamic Models with Switches, Switch Costs, and State Jumps. *Optimization Online Preprint 7124*, 2019. (submitted to International Conference on Decision and Control 2019).

[278] A. N. Kolmogorov and S. V. Fomin. *Introductory Real Analysis.* Dover Books on Mathematics. Dover Publications, New York, 1975. ISBN 978-0-486-61226-3.

[279] D. Kraft. On Converting Optimal Control Problems into Nonlinear Programming Problems. In *Computational Mathematical Programming*, pages 261–280. Springer Berlin Heidelberg, 1985. doi: 10.1007/978-3-642-82450-0_9.

[280] A. Kufner, O. John, and S. Fucik. *Function Spaces.* Number 3 in Mechanics: Analysis. Springer Netherlands, 1977. ISBN 978-90-286-0015-7.

[281] H. W. Kuhn and A. W. Tucker. Nonlinear Programming. In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492, Berkeley, Calif., 1951. University of California Press.

[282] M. Kunkel and M. Gerdts. A nonsmooth Newton's method for discretized optimal control problems with state and control constraints. *Journal of Industrial and Management Optimization*, 4(2):247–270, 2008. doi: 10.3934/jimo.2008.4.247.

[283] C. Lanczos. Trigonometric Interpolation of Empirical and Analytical Functions. *Journal of Mathematics and Physics*, 17(1-4):123–199, 1938. doi: 10.1002/sapm1938171123.

[284] J. Lang and J. G. Verwer. On Global Error Estimation and Control for Initial Value Problems. *SIAM Journal on Scientific Computing*, 29(4):1460–1475, 2007. doi: 10.1137/050646950.

[285] D. Lebiedz, S. Sager, H. G. Bock, and P. Lebiedz. Annihilation of Limit-Cycle Oscillations by Identification of Critical Perturbing Stimuli via Mixed-Integer Optimal Control. *Physical Review Letters*, 95(10), 2005. doi: 10.1103/physrevlett.95.108303.

[286] J. H. Lee, M. Morari, and C. E. Garcia. State-space interpretation of model predictive control. *Automatica*, 30(4):707–717, 1994. doi: 10.1016/0005-1098(94)90159-7.

[287] A. M. Legendre. Recherches sur l'attraction des sphéroides homogénes. *Mém. math. phys. prés. á l'Acad. Aci. par. divers savants 10*, pages 411–434, 1785.

[288] R. Leine, D. van Campen, A. de Kraker, and L. van den Steen. Stick-Slip Vibrations Induced by Alternate Friction Models. *Nonlinear Dynamics*, 16(1):41–54, 1998. ISSN 1573-269X. doi: 10.1023/A:1008289604683.

[289] R. I. Leine, D. H. van Campen, and B. L. van de Vrande. Bifurcations in Nonlinear Discontinuous Systems. *Nonlinear Dynamics*, 23(2):105–164, 2000. doi: 10.1023/A:1008384928636.

[290] D. Leineweber. *Analyse Und Restrukturierung Eines Verfahrens Zur Direkten Lösung von Optimal-Steuerungsproblemen.* Master thesis, University Heidelberg, 1995.

[291] D. Leineweber. *Efficient Reduced SQP Methods for the Optimization of Chemical Processes Described by Large Sparse DAE Models.* PhD Thesis, University Heidelberg, 1999.

[292] F. Lenders. *Numerical Methods for Mixed-Integer Optimal Control with Combinatorial Constraints.* PhD Thesis, Heidelberg University, 2018.

[293] S. Leyffer. MacMPEC: AMPL collection of MPECs, 2000.

[294] S. Leyffer. The penalty interior-point method fails to converge. *Optimization Methods and Software*, 20(4-5):559–568, 2005. doi: 10.1080/10556780500140078.

[295] S. Leyffer and T. S. Munson. A globally convergent filter method for MPECs. Preprint ANL/MCS-P1457-0907, Argonne National Laboratory, Mathematics and Computer Science Division, Argonne, IL, USA, 2007.

[296] S. Leyffer, G. López-Calva, and J. Nocedal. Interior Methods for Mathematical Programs with Complementarity Constraints. *SIAM Journal on Optimization*, 17(1):52–77, 2006. doi: 10.1137/040621065.

[297] D. Liberzon. *Switching in Systems and Control.* Systems & Control: Foundations & Applications. Birkhäuser, Boston, MA, 2012. ISBN 978-1-4612-6574-0.

[298] C. Lindscheid, D. Haßkerl, A. Meyer, A. Potschka, H. G. Bock, and S. Engell. Parallelization of modes of the multi-level iteration scheme for nonlinear model-predictive control of an industrial process. In *2016 IEEE Conference on Control Applications (CCA)*. IEEE, 2016. doi: 10.1109/cca.2016.7588014.

[299] F. Liu, W. W. Hager, and A. V. Rao. Adaptive mesh refinement method for optimal control using nonsmoothness detection and mesh size reduction. *Journal of the Franklin Institute*, 352(10):4081–4106, 2015. doi: 10.1016/j.jfranklin.2015.05.028.

[300] H. Logemann and E. P. Ryan. *Ordinary Differential Equations: Analysis, Qualitative Theory and Control*. Springer Undergraduate Mathematics Series. Springer, London, 2014. ISBN 978-1-4471-6397-8.

[301] P. Lötstedt and L. Petzold. Numerical Solution of Nonlinear Differential Equations with Algebraic Constraints I: Convergence Results for Backward Differentiation Formulas. *Mathematics of Computation*, 46(174):491–516, 1986. doi: 10.2307/2007989.

[302] R. Loxton, K. Teo, and V. Rehbock. Optimal control problems with multiple characteristic time points in the objective and constraints. *Automatica*, 44(11):2923–2929, 2008. doi: 10.1016/j.automatica.2008.04.011.

[303] D. G. Luenberger. *Optimization by Vector Space Methods*. John Wiley, New York, 1969.

[304] J. Lunze and F. Lamnabhi-Lagarrigue. *Handbook of Hybrid Systems Control: Theory, Tools, Applications*. Cambridge University Press, 2009. ISBN 978-0-521-76505-3.

[305] Z.-Q. Luo, J.-S. Pang, and D. Ralph. *Mathematical Programs with Equilibrium Constraints*. Cambridge University Press, Cambridge, 1996. ISBN 978-0-511-98365-8.

[306] Z.-Q. Luo, J.-S. Pang, D. Ralph, and S.-Q. Wu. Exact penalization and stationarity conditions of mathematical programs with equilibrium constraints. *Mathematical Programming*, 75(1):19–76, 1996. doi: 10.1007/bf02592205.

[307] Z.-Q. Luo, J.-S. Pang, and D. Ralph. Piecewise sequential quadratic programming for mathematical programs with nonlinear complementarity constraints. In *Multilevel Optimization: Algorithms and Applications*, volume 20 of *Nonconvex Optim. Appl.*, pages 209–229. Kluwer Acad. Publ., Dordrecht, 1998. doi: 10.1007/978-1-4613-0307-7_9.

[308] K. Makowski and L. W. Neustadt. Optimal Control Problems with Mixed Control-Phase Variable Equality and Inequality Constraints. *SIAM Journal on Control*, 12(2): 184–228, 1974. doi: 10.1137/0312016.

[309] K. Malanowski. Sufficient Optimality Conditions for Optimal Control Subject to State Constraints. *SIAM Journal on Control and Optimization*, 35(1):205–227, 1997. doi: 10.1137/s0363012994267637.

[310] K. Malanowski and H. Maurer. Sensitivity analysis for parametric control problems with control-state constraints. *Computational Optimization and Applications*, 5(3):253–283, 1996. doi: 10.1007/BF00248267.

[311] O. L. Mangasarian and S. Fromovitz. The Fritz John Necessary Optimality Conditions in the Presence of Equality and Inequality Constraints. *Journal of Mathematical Analysis and applications*, 17(1):37–47, 1967. doi: 10.1016/0022-247X(67)90163-1.

[312] C. A. Martell and J. A. Lawton. Adjoint variable solutions via an auxiliary optimization problem. *Journal of Guidance, Control, and Dynamics*, 18(6):1267–1272, 1995. doi: 10.2514/3.21540.

[313] R. Martin. Optimal control drug scheduling of cancer chemotherapy. *Automatica*, 28 (6):1113–1123, 1992. doi: 10.1016/0005-1098(92)90054-J.

[314] H. Maurer. On Optimal Control Problems with Bounded State Variables and Control Appearing Linearly. *SIAM Journal on Control and Optimization*, 15(3):345–362, 1977. doi: 10.1137/0315023.

[315] H. Maurer. On the Minimum Principle for Optimal Control Problems with State Constraints. *Schriftenreihe des Rechenzentrums der Universität Münster*, 41, 1979.

[316] H. Maurer. First and second order sufficient optimality conditions in mathematical programming and optimal control. In H. König, B. Korte, and K. Ritter, editors, *Mathematical Programming at Oberwolfach*, pages 163–177. Springer, Berlin, Heidelberg, 1981. ISBN 978-3-642-00806-1.

[317] H. Maurer. Tutorial on Control and State Constrained Optimal Control Problems. In *SADCO Summer School 2011 - Optimal Control*, London, United Kingdom, 2011. ffinria-00629518f.

[318] H. Maurer and D. Augustin. Sensitivity Analysis and Real-Time Control of Parametric Optimal Control Problems Using Boundary Value Methods. In M. Grötschel, S. O. Krumke, and J. Rambau, editors, *Online Optimization of Large Scale Systems*, pages 17–55. Springer, Berlin, Heidelberg, 2001. ISBN 978-3-662-04331-8. doi: 10.1007/978-3-662-04331-8_2.

[319] H. Maurer and N. Osmolovskii. Second-order conditions for optimal control problems with mixed control-state constraints and control appearing linearly. In *52nd IEEE Conference on Decision and Control*, Florence, Italy, 2013. IEEE. doi: 10.1109/cdc.2013.6759933.

[320] H. Maurer and H. J. Pesch. Solution differentiability for parametric nonlinear control problems with control-state constraints. *Journal of Optimization Theory and Applications*, 86(2):285–309, 1995. doi: 10.1007/BF02192081.

[321] D. Mayne. Nonlinear model predictive control: Challenges and opportunities. In *Nonlinear Model Predictive Control (Ascona, 1998)*, volume 26 of *Progr. Systems Control Theory*, pages 23–44. Birkhäuser, Basel, 2000. doi: 10.1007/978-3-0348-8407-5_2.

[322] G. P. McCormick. A minimanual for use of the SUMT computer program and the factorable programming language. Technical Report SOL 74–15, Department of Operations Research, Stanford University, Stanford, California, 1974.

[323] S. Mehrotra. On the Implementation of a Primal-Dual Interior Point Method. *SIAM Journal on Optimization*, 2(4):575–601, 1992. doi: 10.1137/0802028.

[324] N. G. Meyers and J. Serrin. H = W. *Proceedings of the National Academy of Sciences*, 51 (6):1055–1056, 1964. doi: 10.1073/pnas.51.6.1055.

[325] A. A. Michelson and S. W. Stratton. A new harmonic analyzer. *American Journal of Science*, s4-5(25):1–13, 1898. doi: 10.2475/ajs.s4-5.25.1.

[326] A. T. Miller, W. W. Hager, and A. V. Rao. A Preliminary Analysis of Mesh Refinement for Optimal Control Using Discontinuity Detection via Jump Function Approximations. In *2018 AIAA Guidance, Navigation, and Control Conference*. American Institute of Aeronautics and Astronautics, 2018. doi: 10.2514/6.2018-0852.

[327] M. Mohideen, J. Perkins, and E. Pistikopoulos. Towards an efficient numerical procedure for mixed integer optimal control. *Computers & Chemical Engineering*, 21:S457–S462, 1997. doi: 10.1016/S0098-1354(97)87544-8.

[328] K. D. Mombaur. *Stability Optimization of Open-Loop Controlled Walking Robots*. PhD Thesis, Heidelberg University, 2001.

[329] M. B. Monagan, K. O. Geddes, K. M. Heal, G. Labahn, S. M. Vorkoetter, J. McCarron, and P. DeMarco. *Maple 10 Programming Guide*. Maplesoft, Waterloo ON, Canada, 2005.

[330] K.-S. Moon, A. Szepessy, R. Tempone, and G. E. Zouraris. A variational principle for adaptive approximation of ordinary differential equations. *Numerische Mathematik*, 96 (1):131–152, 2003. doi: 10.1007/s00211-003-0467-8.

[331] F. R. Moulton. *New Methods in Exterior Ballistics*. Univ. of Chicago, Chicago, IL, 1926.

[332] R. Munos and A. Moore. Variable Resolution Discretization in Optimal Control. *Machine Learning*, 49(2/3):291–323, 2002. doi: 10.1023/a:1017992615625.

[333] T. S. Munson, F. Facchinei, M. C. Ferris, A. Fischer, and C. Kanzow. The Semismooth Algorithm for Large Scale Complementarity Problems. *INFORMS Journal on Computing*, 13(4):294–311, 2001. doi: 10.1287/ijoc.13.4.294.9734.

[334] F. H. Murphy, H. D. Sherali, and A. L. Soyster. A mathematical programming approach for determining oligopolistic market equilibrium. *Mathematical Programming*, 24(1):92–106, 1982. doi: 10.1007/bf01585096.

[335] K. G. Murty and S. N. Kabadi. Some NP-complete problems in quadratic and nonlinear programming. *Mathematical Programming*, 39(2):117–129, June 1987. doi: 10.1007/bf02592948.

[336] L. D. Muu and W. Oettli. Optimization over equilibrium sets. *Optimization*, 49(1-2): 179–189, Jan. 2001. doi: 10.1080/02331930108844527.

[337] I. P. Natanson. *Theorie Der Funktionen Einer Reellen Veränderlichen*. Akademie-Verlag, Berlin, 1975. Übersetzung nach der zweiten russischen Auflage von 1957.

[338] L. W. Neustadt. *Optimization - a Theory of Necessary Conditions*. Princeton Univ. Pr., Princeton, NJ, 1976. ISBN 978-0-691-08141-0.

[339] S. M. Nikol'skii. *Approximation of Functions of Several Variables and Imbedding Theorems*. Springer-Verlag, Berlin Heidelberg, 1975. ISBN 978-3-642-65713-9.

[340] J. Nocedal. Updating Quasi-Newton Matrices with Limited Storage. *Mathematics of computation*, 35(151):773–782, 1980. doi: 10.2307/2006193.

[341] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, 2nd edition, 2006. ISBN 978-0-387-30303-1.

[342] R. Oberdieck and E. N. Pistikopoulos. Explicit hybrid model-predictive control: The exact solution. *Automatica*, 58:152–159, 2015. doi: 10.1016/j.automatica.2015.05.021.

[343] J. Oldenburg, W. Marquardt, D. Heinz, and D. B. Leineweber. Mixed-logic dynamic optimization applied to batch distillation process design. *AIChE Journal*, 49(11):2900–2917, 2003. doi: 10.1002/aic.690491120.

[344] M. Osborne. On shooting methods for boundary value problems. *Journal of Mathematical Analysis and Applications*, 27(2):417–433, 1969. doi: 10.1016/0022-247x(69)90059-6.

[345] N. P. Osmolovskii and H. Maurer. *Applications to Regular and Bang-Bang Control: Second-Order Necessary and Sufficient Optimality Conditions in Calculus of Variations and Optimal Control*. Advances in Design and Control. Society for Industrial and Applied Mathematics, Philadelphia, 2012. ISBN 978-1-61197-235-1.

[346] J. Outrata, M. Kočvara, and J. Zowe. *Nonsmooth Approach to Optimization Problems with Equilibrium Constraints: Theory, Applications and Numerical Results*. Number 28 in Nonconvex Optimization and Its Applications. Springer US, 1998. ISBN 978-0-7923-5170-2.

[347] J. V. Outrata. Optimality Conditions for a Class of Mathematical Programs with Equilibrium Constraints. *Mathematics of operations research*, 24(3):627–644, 1999. doi: 10.1287/moor.24.3.627.

[348] J. V. Outrata. A Generalized Mathematical Program with Equilibrium Constraints. *SIAM Journal on Control and Optimization*, 38(5):1623–1638, 2000. doi: 10.1137/S0363012999352911.

[349] B. Owren and M. Zennaro. Derivation of Efficient, Continuous, Explicit Runge–Kutta Methods. *SIAM Journal on Scientific and Statistical Computing*, 13(6):1488–1501, 1992. doi: 10.1137/0913084.

[350] J.-S. Pang and M. Fukushima. Complementarity Constraint Qualifications and Simplified B-Stationarity Conditions for Mathematical Programs with Equilibrium Constraints. *Computational Optimization and Applications*, 13(1):111–136, 1999. doi: 10.1023/A:1008656806889.

[351] M. A. Patterson and A. V. Rao. GPOPS-II. *ACM Transactions on Mathematical Software*, 41(1):1–37, 2014. doi: 10.1145/2558904.

[352] M. A. Patterson, W. W. Hager, and A. V. Rao. A ph mesh refinement method for optimal control. *Optimal Control Applications and Methods*, 36(4):398–421, 2014. doi: 10.1002/oca.2114.

[353] H. J. Pesch. Real-time computation of feedback controls for constrained optimal control problems. part 1: Neighbouring extremals. *Optimal Control Applications and Methods*, 10(2):129–145, 1989. doi: 10.1002/oca.4660100205.

[354] D. W. Peterson. A Review of Constraint Qualifications in Finite-Dimensional Spaces. *SIAM Review*, 15(3):639–654, 1973. doi: 10.1137/1015075.

[355] E. Philip, W. Murray, M. A. Saunders, and M. H. Wright. User's guide for NPSOL 5.0: A FORTRAN package for nonlinear programming. Technical Report SOL 86-6, Stanford University, 2001.

[356] K. Plitt. *Ein Superlinear Konvergentes Mehrzielverfahren Zur Direkten Berechnung Beschränkter Optimaler Steuerungen.* Master thesis, University of Bonn, 1981.

[357] L. Pontryagin. *Mathematische Theorie Optimaler Prozesse.* Oldenbourg Verlag, Wien, 1964.

[358] A. Potschka. *Handling Path Constraints in a Direct Multiple Shooting Method for Optimal Control Problems.* Master thesis, University Heidelberg, 2006.

[359] A. Potschka, H. Bock, and J. 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. doi: 10.1080/10556780902753098.

[360] M. J. D. Powell. A Fast Algorithm for Nonlinearly Constrained Optimization Calculations. In G. Watson, editor, *Numerical Analysis*, volume 630 of *Lecture Notes in Mathematics*, pages 144–157. Springer, Berlin Heidelberg, 1978. ISBN 978-3-540-08538-6.

[361] M. J. D. Powell. Algorithms for nonlinear constraints that use lagrangian functions. *Mathematical Programming*, 14(1):224–248, 1978. doi: 10.1007/bf01588967.

[362] S. J. Qin and T. A. Badgwell. An Overview of Nonlinear Model Predictive Control Applications. In *Nonlinear Model Predictive Control*, pages 369–392. Birkhäuser Basel, 2000. doi: 10.1007/978-3-0348-8407-5_21.

[363] S. J. Qin and T. A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11(7):733–764, 2003. doi: 10.1016/s0967-0661(02)00186-7.

[364] A. Quarteroni. Some results of bernstein and jackson type for polynomial approximation in L p-spaces. *Japan Journal of Applied Mathematics*, 1(1):173–181, 1984. doi: 10.1007/bf03167866.

[365] A. U. Raghunathan and L. T. Biegler. Mathematical programs with equilibrium constraints (MPECs) in process engineering. *Computers & Chemical Engineering*, 27(10):1381–1392, 2003. doi: 10.1016/s0098-1354(03)00092-9.

[366] A. U. Raghunathan, V. Gopal, D. Subramanian, L. T. Biegler, and T. Samad. Dynamic Optimization Strategies for Three-Dimensional Conflict Resolution of Multiple Aircraft. *Journal of Guidance, Control, and Dynamics*, 27(4):586–594, 2004. doi: 10.2514/1.11168.

[367] R. Raman and I. Grossmann. Modelling and computational techniques for logic based integer programming. *Computers & Chemical Engineering*, 18(7):563–578, 1994. doi: 10.1016/0098-1354(93)e0010-7.

[368] A. V. Rao and K. D. Mease. Eigenvector approximate dichotomic basis method for solving hyper-sensitive optimal control problems. *Optimal Control Applications and Methods*, 21(1):1–19, 2000. doi: 10.1002/(SICI)1099-1514(200001/02)21:1<1::AID-OCA646>3.0.CO;2-V.

[369] J. B. Rawlings, E. S. Meadows, and K. R. Muske. Nonlinear Model Predictive Control: A Tutorial and Survey. *IFAC Proceedings Volumes*, 27(2):185–197, 1994. doi: 10.1016/s1474-6670(17)48151-1.

[370] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl. *Model Predictive Control: Theory, Computation, and Design*. Nob Hill Publishing, LLC, 2nd edition, 2017. ISBN 978-0-9759377-3-0.

[371] T. J. Rivlin. *An Introduction to the Approximation of Functions*. Blaisdell Publishing Company, Walham, Massachusetts, 1969.

[372] S. M. Robinson. Perturbed Kuhn-Tucker points and rates of convergence for a class of nonlinear-programming algorithms. *Mathematical Programming*, 7(1):1–16, Dec. 1974. doi: 10.1007/bf01585500.

[373] S. M. Robinson. Stability Theory for Systems of Inequalities, Part II: Differentiable Nonlinear Systems. *SIAM Journal on Numerical Analysis*, 13(4):497–513, 1976. doi: 10.1137/0713043.

[374] J. Roll. Identification of piecewise affine systems via mixed-integer programming. *Automatica*, 40(1):37–50, 2004. doi: 10.1016/j.automatica.2003.08.006.

[375] I. M. Ross and F. Fahroo. A Pseudospectral Transformation of the Convectors of Optimal Control Systems. *IFAC Proceedings Volumes*, 34(13):543–548, 2001. doi: 10.1016/s1474-6670(17)39048-1.

[376] I. M. Ross and F. Fahroo. A Direct Method for Solving Nonsmooth Optimal Control Problems. *IFAC Proceedings Volumes*, 35(1):479–484, 2002. doi: 10.3182/20020721-6-es-1901.00329.

[377] I. M. Ross and F. Fahroo. Pseudospectral Knotting Methods for Solving Nonsmooth Optimal Control Problems. *Journal of Guidance, Control, and Dynamics*, 27(3):397–405, 2004. doi: 10.2514/1.3426.

[378] W. Rudin. *Real and Complex Analysis*. McGraw-Hill Series in Higher Mathematics. McGraw-Hill, New York, 3. edition, 1987. ISBN 0-07-100276-6.

[379] A. Saccon, N. Wouw, and H. Nijmeijer. Sensitivity analysis of hybrid systems with state jumps with application to trajectory tracking. *IEEE Conference on Decision and Control*, 53(27):3065–3070, 2014. doi: 10.1109/CDC.2014.7039861.

[380] S. Sager. *Numerical Methods for Mixed–Integer Optimal Control Problems*. PhD Thesis, University Heidelberg, 2006.

[381] S. Sager, H. G. Bock, and G. Reinelt. Direct methods with maximal lower bound for mixed-integer optimal control problems. *Mathematical Programming*, 118(1):109–149, 2007. doi: 10.1007/s10107-007-0185-6.

[382] S. Sager, M. Diehl, G. Singh, A. Küpper, and S. Engell. Determining SMB superstructures by mixed-integer control. In K.-H. Waldmann and U. Stocker, editors, *Operations Research Proceedings 2006*, pages 37–44. Springer, Berlin, Heidelberg, 2007. ISBN 978-3-540-69995-8.

[383] S. Sager, C. Kirches, and H. G. Bock. Fast solution of periodic optimal control problems in automobile test-driving with gear shifts. In *2008 47th IEEE Conference on Decision and Control*. IEEE, 2008. doi: 10.1109/cdc.2008.4739014.

[384] S. Sager, H. G. Bock, and M. Diehl. The integer approximation error in mixed-integer optimal control. *Mathematical Programming*, 133(1-2):1–23, 2010. doi: 10.1007/s10107-010-0405-3.

[385] R. W. H. Sargent and G. R. Sullivan. The Development of an Efficient Optimal Control Package. In J. Stoer, editor, *Optimization Techniques*, volume 7 of *Lecture Notes in Control and Information Sciences*, pages 158–168. Springer, Berlin, Heidelberg, 1978. ISBN 978-3-540-08708-3.

[386] H. Scheel and S. Scholtes. Mathematical Programs with Complementarity Constraints: Stationarity, Optimality, and Sensitivity. *Mathematics of Operations Research*, 25(1):1–22, 2000. doi: 10.1287/moor.25.1.1.15213.

[387] K. Schittkowski. The Nonlinear Programming Method of Wilson, Han, and Powell with an Augmented Lagrangian Type Line Search Function. *Numerische Mathematik*, 38(1):83–114, 1982. doi: 10.1007/BF01395810.

[388] K. Schittkowski. On the Convergence of a Sequential Quadratic Programming Method with an Augmented Lagrangian Line Search Function. *Mathematische Operationsforschung und Statistik. Series Optimization*, 14(2):197–216, 1983. doi: 10.1080/02331938308842847.

[389] K. Schittkowski. NLPQL: A FORTRAN Subroutine Solving Constrained Nonlinear Programming Problems. *Annals of operations research*, 5(1):485–500, 1986. doi: 10.1007/BF02022087.

[390] I. J. Schoenberg. Contributions to the problem of approximation of equidistant data by analytic functions. Part A. On the problem of smoothing or graduation. A first class of analytic approximation formulae. *Quarterly of Applied Mathematics*, 4(1):45–99, 1946. doi: 10.1090/qam/15914.

[391] S. Scholtes. Convergence Properties of a Regularization Scheme for Mathematical Programs with Complementarity Constraints. *SIAM Journal on Optimization*, 11(4):918–936, 2001. doi: 10.1137/S1052623499361233.

[392] S. Scholtes. Combinatorial structures in nonlinear programming. Technical report, The Judge Institute of Management Studies, University of Cambridge, Cambridge, England, 2002.

[393] S. Scholtes. Nonconvex Structures in Nonlinear Programming. *Operations Research*, 52(3):368–383, 2004. doi: 10.1287/opre.1030.0102.

[394] S. Scholtes and M. Stöhr. Exact Penalization of Mathematical Programs with Equilibrium Constraints. *SIAM Journal on Control and Optimization*, 37(2):617–652, 1999. doi: 10.1137/s0363012996306121.

[395] A. Schönhage. Fehlerfortpflanzung bei Interpolation. *Numerische Mathematik*, 3(1):62–71, Dec. 1961. doi: 10.1007/bf01386001.

[396] C. Schwab. *P- and Hp- Finite Element Methods: Theory and Applications to Solid and Fluid Mechanics*. Numerical Mathematics and Scientific Computation. Clarendon Press, 1999. ISBN 978-0-19-850390-3.

[397] A. L. Schwartz. *Theory and Implementation of Numerical Methods Based on Runge-Kutta Integration for Solving Optimal Control Problems*. PhD Thesis, EECS Department, University of California, Berkeley, 1996.

[398] L. Schwartz. Théorie des distributions à valeurs vectorielles. I. *Annales de l'institut Fourier*, 7:1–141, 1957. doi: 10.5802/aif.68.

[399] L. Schwartz. Théorie des distributions à valeurs vectorielles. II. *Annales de l'institut Fourier*, 8:1–209, 1958. doi: 10.5802/aif.77.

[400] C. A. Schweiger and C. A. Floudas. Interaction of Design and Control: Optimization with Dynamic Models. In *Optimal Control*, volume 15 of *Applied Optimization*, pages 388–435. Springer, Boston, MA, 1998. ISBN 978-1-4419-4796-3.

[401] H. Seywald and R. Kumar. A method for automatic costate calculation. In *Guidance, Navigation, and Control Conference*. American Institute of Aeronautics and Astronautics, 1996. doi: 10.2514/6.1996-3699.

[402] O. Shaik, S. Sager, O. Slaby, and D. Lebiedz. Phase tracking and restoration of circadian rhythms by model-based optimal control. *IET Systems Biology*, 2(1):16–23, 2008. doi: 10.1049/iet-syb:20070016.

[403] M. S. Shaikh and P. E. Caines. On the Hybrid Optimal Control Problem: Theory and Algorithms. *IEEE Transactions on Automatic Control*, 52(9):1587–1603, 2007. doi: 10.1109/tac.2007.904451.

[404] L. F. Shampine. Interpolation for Runge-Kutta Methods. *SIAM Journal on Numerical Analysis*, 22(5):1014–1027, 1985. doi: 10.1137/0722060.

[405] L. F. Shampine. *Numerical Solution of Ordinary Differential Equations*. Number 4 in Chapman & Hall Mathematics. Chapman & Hall, New York, London, 1994. ISBN 978-0-412-05151-7.

[406] L. F. Shampine. Error Estimation and Control for ODEs. *Journal of Scientific Computing*, 25(1):3–16, 2005. doi: 10.1007/s10915-004-4629-3.

[407] L. F. Shampine and M. K. Gordon. *Computer Solution of Ordinary Differential Equations: The Initial Value Problem*. W. H. Freeman, San Francisco, 1975. ISBN 978-0-7167-0461-4.

[408] D. F. Shanno. Conditioning of Quasi-Newton Methods for Function Minimization. *Mathematics of computation*, 24(111):647–656, 1970. doi: 10.2307/2004840.

[409] M. E. Shayan. *A Methodology for Comparing Algorithms and a Method of Computing Mth Order Directional Derivatives Based on Factorable Programming*. PhD Thesis, The George Washington University, 1980.

[410] W. Shen and G. Wang. Changeable degree spline basis functions. *Journal of Computational and Applied Mathematics*, 234(8):2516–2529, 2010. doi: 10.1016/j.cam.2010.03.015.

[411] R. D. Skeel. Thirteen ways to estimate global error. *Numerische Mathematik*, 48(1):1–20, 1986. doi: 10.1007/bf01389440.

[412] R. D. Skeel. Global error estimation and the backward differentiation formulas. *Applied Mathematics and Computation*, 31:197–208, 1989. doi: 10.1016/0096-3003(89)90119-7.

[413] J. C. Slater. Electronic Energy Bands in Metals. *Physical Review*, 45(11):794–801, 1934. doi: 10.1103/physrev.45.794.

[414] B. Speelpenning. *Compiling Fast Partial Derivatives of Functions Given by Algorithms*. PhD Thesis, University of Illinois Urbana-Champaign, 1980.

[415] B. Srinivasan, S. Palanki, and D. Bonvin. Dynamic optimization of batch processes: I. Characterization of the nominal solution. *Computers & Chemical Engineering*, 27(1):1–26, 2003. doi: 10.1016/S0098-1354(02)00116-3.

[416] O. Stein. Lifting mathematical programs with complementarity constraints. *Mathematical Programming*, 131(1-2):71–94, 2012. doi: 10.1007/s10107-010-0345-y.

[417] H. J. Stetter. Economical Global Error Estimation. In R. A. Willoughby, editor, *Stiff Differential Systems*, The IBM Research Symposia Series, pages 245–258. Springer, Boston, MA, 1974. ISBN 978-1-4684-2102-6.

[418] J. Stoer. Principles of Sequential Quadratic Programming Methods for Solving Nonlinear Programs. In K. Schittkowski, editor, *Computational Mathematical Programming*, volume 15 of *NATO ASI Series (Series F: Computer and Systems Sciences)*, pages 165–207. Springer, Berlin, Heidelberg, 1985. ISBN 978-3-642-82452-4.

[419] J. Stoer, R. Bartels, W. Gautschi, R. Bulirsch, and C. Witzgall. *Introduction to Numerical Analysis*. Number 12 in Texts in Applied Mathematics. Springer-Verlag, New York, 2002. ISBN 978-0-387-95452-3.

[420] B. Sündermann. Lebesgue constants in Lagrangian interpolation at the Fekete points. *Mitt. Math. Ges. Hamburg*, 11(2):204–211, 1983. ISSN 0340-4358.

[421] G. Szegö. *Orthogonal Polynomials*, volume 23 of *AMS Colloquium Publications*. American Mathematical Society, Providence, R. I., revised edition, 1959. ISBN 978-0-8218-1023-1.

[422] E. Tadmor. The Exponential Accuracy of Fourier and Chebyshev Differencing Methods. *SIAM Journal on Numerical Analysis*, 23(1):1–10, 1986. doi: 10.1137/0723001.

[423] A. E. Taylor. *Calculus with Analytic Geometry*. Prentice-Hall Mathematics Series. Prentice-Hall Inc., Englewood Cliffs, N.J., 1959. ISBN 978-1-114-26443-4.

[424] S. Terwen, M. Back, and V. Krebs. Predictive Powertrain Control for Heavy Duty Trucks. *IFAC Proceedings Volumes*, 37(22):105–110, 2004. doi: 10.1016/s1474-6670(17)30329-4.

[425] G. Teschl. *Ordinary Differential Equations and Dynamical Systems*, volume 140 of *Graduate Studies in Mathematics*. American Mathematical Society, 2012. ISBN 978-0-8218-8328-0.

[426] A. F. Timan. *Theory of Approximation of Functions of A Real Variable*. International Series of Monographs on Pure and Applied Mathematics. Pergamon Press, 1963. ISBN 978-0-08-009929-3.

[427] L. Tran and M. Berzins. Defect Sampling in Global Error Estimation for ODEs and Method-Of-Lines PDEs Using Adjoint Methods. SCI Technical Report UUSCI-2011-006, SCI Institute, University of Utah, 2011.

[428] L. N. Trefethen. *Spectral Methods in MATLAB*. Software, Environments, Tools. SIAM, Philadelphia, Pa, 3rd. repr. edition, 2000. ISBN 978-0-89871-465-4.

[429] A. H. Turetskii. The bounding of polynomials prescribed at equally distributed points. In *Proc. Pedag. Inst. Vitebsk*, volume 3, pages 117–127, 1940.

[430] V. I. Utkin. *Sliding Modes in Control and Optimization*. Communications and Control Engineering. Springer-Verlag, Berlin Heidelberg, 1992. ISBN 978-3-642-84381-5.

[431] B. van Brunt and M. Carter. *The Lebesgue-Stieltjes Integral: A Practical Introduction*. Undergraduate Texts in Mathematics. Springer-Verlag, New York, 2000. ISBN 978-0-387-95012-9.

[432] A. J. van der Schaft and H. Schumacher. *An Introduction to Hybrid Dynamical Systems*, volume 251 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag, London, 2000. ISBN 978-1-85233-233-4.

[433] S. A. Vavasis. Quadratic Programming is in NP. *Inf. Process. Lett.*, 36(2):73–77, 1990. doi: 10.1016/0020-0190(90)90100-C.

[434] R. Verfürth. *A Review of a Posteriori Error Estimation and Adaptive Mesh-Refinement Techniques*. Wiley-Teubner Series in Advances in Numerical Mathematics. Wiley-Teubner, 1996. ISBN 978-0-471-96795-8.

[435] J. H. Verner. Explicit Runge–Kutta Methods with Estimates of the Local Truncation Error. *SIAM Journal on Numerical Analysis*, 15(4):772–790, 1978. doi: 10.1137/0715051.

[436] P. Vértesi. On lagrange interpolation. *Periodica Mathematica Hungarica*, 12(2):103–112, 1981.

[437] P. Vértesi. Optimal Lebesgue Constant for Lagrange Interpolation. *SIAM Journal on Numerical Analysis*, 27(5):1322–1331, 1990. doi: 10.1137/0727075.

[438] J. Vlassenbroeck. A chebyshev polynomial method for optimal control with state constraints. *Automatica*, 24(4):499–506, 1988. doi: 10.1016/0005-1098(88)90094-5.

[439] J. Vlassenbroeck and R. V. Dooren. A Chebyshev technique for solving nonlinear optimal control problems. *IEEE Transactions on Automatic Control*, 33(4):333–340, 1988. ISSN 0018-9286. doi: 10.1109/9.192187.

[440] O. von Stryk. Numerical Solution of Optimal Control Problems by Direct Collocation. In R. Bulirsch, A. Miele, J. Stoer, and K. Well, editors, *Optimal Control*, volume 111 of *ISNM International Series of Numerical Mathematics*, pages 129–143. Birkhäuser, Basel, 1993. ISBN 978-3-0348-7541-7.

[441] O. von Stryk. User's Guide for DIRCOL (Version 2.1): A direct collocation method for the numerical solution of optimal control problems. Technical report, Fachgebiet Simulation und Systemoptimierung, Technische Universität Darmstadt, 1999.

[442] O. von Stryk and R. Bulirsch. Direct and indirect methods for trajectory optimization. *Annals of Operations Research*, 37(1):357–373, 1992. doi: 10.1007/bf02071065.

[443] O. von Stryk and M. Glocker. Decomposition of Mixed-Integer Optimal Control Problems Using Branch and Bound and Sparse Direct Collocation. In *Proc. ADPM 2000 – The 4th International Conference on Automatisation of Mixed Processes: Hybrid Dynamical Systems*, pages 99–104, 2000.

[444] 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, 2005. doi: 10.1007/s10107-004-0559-y.

[445] A. Walther and A. Griewank. Getting Started with ADOL-C. In U. Naumann and O. Schenk, editors, *Combinatorial Scientific Computing*, Chapman & Hall/CRC Computational Science. Chapman and Hall/CRC, 2012. ISBN 978-1-4398-2735-2.

[446] E. Waring. Problems concerning Interpolations. *Philosophical Transactions of the Royal Society of London*, 69(0):59–67, 1779. doi: 10.1098/rstl.1779.0008.

[447] S. Wei, K. Uthaichana, M. Žefran, R. DeCarlo, and S. Bengea. Applications of numerical optimal control to nonlinear hybrid systems. *Nonlinear Analysis: Hybrid Systems*, 1(2):264–279, 2007. doi: 10.1016/j.nahs.2006.10.007.

[448] R. E. Wengert. A simple automatic derivative evaluation program. *Communications of the ACM*, 7(8):463–464, 1964. doi: 10.1145/355586.364791.

[449] H. Wilbraham. On a certain periodic function. *Cambridge and Dublin Math. J.*, 3: 198–201, 1848.

[450] L. Wirsching. *An SQP Algorithm with Inexact Derivatives for a Direct Multiple Shooting Method for Optimal Control Problems*. Master thesis, University Heidelberg, 2006.

[451] L. Wirsching. *Multi-Level Iteration Schemes with Adaptive Level Choice for Nonlinear Model Predictive Control*. PhD Thesis, Heidelberg University, 2018.

[452] L. Wirsching, H. Bock, and M. Diehl. Fast NMPC of a chain of masses connected by springs. In *2006 IEEE International Conference on Control Applications*. IEEE, 2006. doi: 10.1109/cca.2006.285936.

[453] L. Wirsching, H. J. Ferreau, H. G. Bock, and M. Diehl. An Online Active Set Strategy for Fast Adjoint Based Nonlinear Model Predictive Control. *IFAC Proceedings Volumes*, 40(12):234–239, 2007. doi: 10.3182/20070822-3-za-2920.00039.

[454] L. Wirsching, J. Albersmeyer, P. Kühl, M. Diehl, and H. Bock. An Adjoint-based Numerical Method for Fast Nonlinear Model Predictive Control. *IFAC Proceedings Volumes*, 41(2):1934–1939, 2008. doi: 10.3182/20080706-5-kr-1001.00329.

[455] J. Wloka. *Funktionalanalysis Und Anwendungen*. De Gruyter Lehrbuch. Walter de Gruyter, 1971. ISBN 978-3-11-001989-6.

[456] S. Woon, V. Rehbock, and R. Loxton. Towards global solutions of optimal discrete-valued control problems. *Optimal Control Applications and Methods*, 33(5):576–594, 2012. doi: 10.1002/oca.1015.

[457] X. Xu and P. Antsaklis. Optimal control of switched autonomous systems. In *Proceedings of the 41st IEEE Conference on Decision and Control*. IEEE, 2002. doi: 10.1109/cdc.2002.1185065.

[458] X. Xu and P. Antsaklis. Optimal control of switched systems based on parameterization of the switching instants. *IEEE Transactions on Automatic Control*, 49(1):2–16, 2004. doi: 10.1109/TAC.2003.821417.

[459] X. Xu and P. J. Antsaklis. Results and Perspectives on Computational Methods for Optimal Control of Switched Systems. In O. Maler and A. Pnueli, editors, *Hybrid Systems: Computation and Control*, volume 2623 of *Lecture Notes in Computer Science*, pages 540–555. Springer, Berlin Heidelberg, 2003. ISBN 978-3-540-00913-9.

[460] K. Yosida. *Functional Analysis*. Springer Classics in Mathematics. Springer-Verlag, Berlin Heidelberg, 1995. ISBN 978-3-642-61859-8.

[461] L. Young. Generalized curves and the existence of an attained absolute minimum in the calculus of variations. *C.R. Soc. Sci. Lett. Varsovie, CL III*, 30:212–234, 1937.

[462] P. E. Zadunaisky. A method for the estimation of errors propagated in the numerical solution of a system of ordinary differential equations. In G. I. Kontopoulos, editor, *The Theory of Orbits in the Solar System and in Stellar Systems*, volume 25 of *IAU Symposium*, pages 281–287, Thessaloniki, 1966.

[463] P. E. Zadunaisky. On the estimation of errors propagated in the numerical integration of ordinary differential equations. *Numerische Mathematik*, 27(1):21–39, 1976. doi: 10.1007/bf01399082.

[464] A. Zanelli, R. Quirynen, J. Jerez, and M. Diehl. A Homotopy-based Nonlinear Interior-Point Method for NMPC. *IFAC-PapersOnLine*, 50(1):13188–13193, 2017. doi: 10.1016/j.ifacol.2017.08.2175.

[465] V. M. Zavala and L. T. Biegler. The advanced-step NMPC controller: Optimality, stability and robustness. *Automatica*, 45(1):86–93, 2009. doi: 10.1016/j.automatica.2008.06.011.

[466] V. Zeidan. The Riccati Equation for Optimal Control Problems with Mixed State-Control Constraints: Necessity and Sufficiency. *SIAM Journal on Control and Optimization*, 32 (5):1297–1321, 1994. doi: 10.1137/S0363012992233640.

[467] E. Zeidler. *Nonlinear Functional Analysis and Its Applications: Fixed Point Theorems*. Nonlinear Functional Analysis and Its Applications. Springer-Verlag, New York, 1986. ISBN 978-0-387-90914-1.

[468] E. Zeidler. *Nonlinear Functional Analysis and Its Applications: II/B: Nonlinear Monotone Operators*. Springer-Verlag, New York, 1990. ISBN 978-0-387-97167-4.

[469] E. Zeidler. *Nonlinear Functional Analysis and Its Applications: II/A: Linear Monotone Operators*. Springer-Verlag, New York, 1990. ISBN 978-0-387-96802-5.

[470] J. Zhang and G. Liu. A New Extreme Point Algorithm and Its Application in PSQP Algorithms for Solving Mathematical Programs with Linear Complementarity Constraints. *Journal of Global Optimization*, 19(4):345–361, 2001. ISSN 1573-2916. doi: 10.1023/A:1011226232107.

[471] Y. Zhao and P. Tsiotras. Density Functions for Mesh Refinement in Numerical Optimal Control. *Journal of Guidance, Control, and Dynamics*, 34(1):271–277, 2011. doi: 10.2514/1.45852.

[472] W. P. Ziemer. *Weakly Differentiable Functions: Sobolev Spaces and Functions of Bounded Variation*. Graduate Texts in Mathematics. Springer-Verlag, New York, 1989. ISBN 978-0-387-97017-2.

Nomenclature

List of Symbols

$\triangle$	End of a definition, lemma, or theorem
$\square$	End of a proof
$\stackrel{\text{def}}{=}$	Defined to be equal
$\dot\cup$, $\cup$	(Disjoint) set–theoretic union ("unified with")
$\cap$	Set–theoretic intersection ("intersected with")
$\wedge$	Logical conjunction ("AND")
$\oplus$, $\vee$	Logical exclusive/inclusive disjunction ("EX–OR/OR")
$\supseteq, \supset$	Superset of a set ("is a (proper) superset of")
$\subseteq, \subset$	Subset of a set ("is a (proper) subset of")
$\in, \notin$	Set membership ("is (not) an element of")
$\setminus$	Set difference
$\mathrm{cl}(\cdot)$	Closure of a set
$\mathrm{int}(\cdot)$	Interior of a set
$\times$	Cartesian product of sets, multiplication in literal numbers
$\emptyset$	The empty set
$\forall$	Universal quantification ("for all")
$\exists$	Existential quantification ("exists")
$A_{i,\cdot}$, $A_{\cdot,i}$	i–th row/column of matrix A, row/column vector
A^T	Transpose of matrix A
A^{-1}	Inverse of matrix A
A^{-T}	Transpose of the inverse of matrix A
x_i	i–th entry of vector x
$\boldsymbol{f}_i$	i–th entry of vector–valued function $\boldsymbol{f}$
$\lceil x \rceil$	Least integer greater than or equal to x
Θ_X	Zero element of some vector space X
$\mathbf{0}_n, \mathbf{0}$	n–by–1 vector of zeros
$\mathbf{1}_n, \mathbf{1}$	n–by–1 vector of ones
$\boldsymbol{f}(t^-)$	Limit from the left $\boldsymbol{f}(t^-) = \lim_{s \nearrow t} \boldsymbol{f}(s)$
$\boldsymbol{f}(t^+)$	Limit from the right $\boldsymbol{f}(t^+) = \lim_{s \searrow t} \boldsymbol{f}(s)$
σ	Switching sequence (see Definition 1.8)
σ_E, σ_I	EFS/IFS induced switching sequence
$\cdot \perp \cdot$	$s, t \in \mathbb{R}^n$: $0 \geqslant s \perp t \leqslant 0 \Leftrightarrow 0 \geqslant s, 0 \geqslant t,\ s^T t = 0$
D	First–order differentiation matrix (see Eq (B.9))
$F'(x; d)$	Directional derivative of F at x in direction d
$\delta F(x)(d)$	Gateaux derivative of F at x in direction d
$F'(x)(d)$	Fréchet derivative of F at x in direction d
$F'_x(x, y)$	Partial Fréchet derivative of F at (x, y)
$\nabla F(x)$	Gradient of $F : \mathbb{R}^n \longrightarrow \mathbb{R}$ at x

Black Board Symbols

$\mathbb{N}$	Set of natural numbers excluding zero
$\mathbb{Z}$	Set of integer numbers
$\mathbb{R}$	Set of real numbers
$\mathbb{R}^n$	Space of n–vectors with elements from the set $\mathbb{R}$
$\mathbb{R}^{m\times n}$	Space of $m \times n$–matrices with elements from the set $\mathbb{R}$

Function Space and Norm Symbols

$\mathcal{C}$	Space of continuous functions
$\mathcal{Y}^k$	Tailored semi–discrete function spaces (see Section 2.4.5)
$\mathcal{Y}^k_{\mathcal{P}}$	Discretized function space $\mathcal{Y}^k$ (see Section 9.3)
$\mathcal{AC}$	Space of absolutely continuous functions
L^p	Space of all mappings that are bounded in the norm $\|\cdot\|_p$
$W^{q,p}$	Sobolev space of mappings in $\mathcal{AC}$ that are bounded in the norm $\|\cdot\|_{q,p}$
$\mathcal{BV}$	Space of functions of bounded variation
$\mathcal{Z}_H$	Discretized function space $\mathcal{BV}$ (see Section 9.3)
$\mathcal{NBV}$	Space of normalized functions of bounded variation
$\mathscr{L}(X,Y)$	Set of all linear and continuous mappings $L : X \longrightarrow Y$
X^*	Topological dual space of a normed vector space X
$\|\cdot\|$	Component-wise mapping of a real number to the absolute value
$\|\cdot\|$	The (euclidean) norm of a matrix or vector
$\|\cdot\|_F$	Frobenius norm of a matrix
$\|\cdot\|_{\mathcal{Y}^k}$	$\mathcal{Y}^k$ norm
$\|\cdot\|_p$	L^p norm
$\|\cdot\|_{q,p}$	$W^{q,p}$ norm

Interval Symbols

$\mathcal{I}$	Some time interval (open, closed, half–open) with endpoints a and b
$\mathcal{T}$	Time horizon $\mathcal{T} = [t_s, t_f] \subset \mathbb{R}$ for an ODE or OCP
t	Model or process time $t \in \mathcal{T}$
$t_s,\ t_f$	Initial/Final model or process time, start/end of time horizon $\mathcal{T}$
t_σ	Activation time of an implicit switch

Interval Symbols

$\lambda,\ \boldsymbol{\lambda}(\cdot)$	Multiplier/Costate of the equality constraints
$\mu,\ \boldsymbol{\mu}(\cdot)$	Multiplier/Costate of the inequality constraints
ν	Multiplier of the boundary constraint (and matching conditions)

Sets

$[N]$	Set $\{1, \dots, N\}$ for $N \in \mathbb{N}$
$\lvert\mathcal{X}\rvert$	Cardinality of a set $\mathcal{X}$
$\mathcal{X}$	Set of all differential state trajectories
$\mathcal{U}$	Set of all continuous control functions
Ω	Set of admissible choices for discrete control function $\boldsymbol{v}$
K^+	Positive dual cone of cone K (see Definition 2.21)
K^-	Negative dual cone of cone K (see Definition 2.21)
$\mathcal{U}_\varepsilon(x)$	$\mathcal{U}_\varepsilon(x) = \{y \in M \,:\, d(x,y) < \varepsilon\}$ for a metric space (M, d)
$\operatorname{conv}(\mathcal{X})$	Convex hull of a set $\mathcal{X}$
$\mathcal{A}(x)$	Active set at x (see Definition 3.13)
$\mathcal{T}(\Sigma, x)$	Tangent cone (see Definition 2.14)
$\mathcal{F}(\Sigma, x)$	Linearized feasibility cone (see Definition 3.15)

Functions

$\boldsymbol{\varphi}(\cdot)$	MAYER cost function $[\boldsymbol{\varphi}(z) \in \mathbb{R}]$
$\boldsymbol{\psi}(\cdot)$	Lagrangian $[\boldsymbol{\psi}(z) \in \mathbb{R}]$
$\boldsymbol{f}(\cdot)$	ODE system right hand side $[\boldsymbol{f}(z) \in \mathbb{R}^{n_\mathrm{x}}]$
$\boldsymbol{c}(\cdot)$	Path constraint function $[\boldsymbol{c}(z) \in \mathbb{R}^{n_\mathrm{c}}]$
$\boldsymbol{r}(\cdot)$	Endpoint constraint function $[\boldsymbol{r}(z) \in \mathbb{R}^{n_\mathrm{r}}]$
$\boldsymbol{x}(\cdot)$	Trajectory of ODE system states $[\boldsymbol{x}(t) \in \mathbb{R}^{n_\mathrm{x}},\, t \in \mathcal{T}]$
$\boldsymbol{u}(\cdot)$	Trajectory of continuous process controls $[\boldsymbol{u}(t) \in \mathbb{R}^{n_\mathrm{u}},\, t \in \mathcal{T}]$
$\boldsymbol{v}(\cdot)$	Trajectory of discrete process controls $[\boldsymbol{v}(t) \in \mathbb{R}^{n_\mathrm{v}},\, t \in \mathcal{T}]$
$\boldsymbol{\omega}(\cdot)$	Trajectory of binary convex multipliers
$\boldsymbol{\alpha}(\cdot)$	Trajectory of relaxed convex multipliers
$\boldsymbol{\sigma}(\cdot)$	Switching function
$\mathcal{L}(\cdot)$	LAGRANGE function
$\mathcal{H}(\cdot)$	HAMILTON function
$\hat{\mathcal{H}}(\cdot)$	Augmented HAMILTON function
$\{\boldsymbol{P}_n(\cdot;\cdot,\cdot)\}$	Set of JACOBI polynomials
$\{\boldsymbol{P}_n(\cdot)\}$	Set of LEGENDRE polynomials
$\{\boldsymbol{T}_n(\cdot)\}$	Set of CHEBYSHEV polynomials of first kind
$\{\boldsymbol{L}_n(\cdot)\}$	Set of LAGRANGE fundamental polynomials
$\mathcal{X}_\mathcal{I}(\cdot)$	Characteristic function
$\mathcal{H}_s(\cdot)$	(Translated) HEAVISIDE function
$\boldsymbol{\delta}(\cdot)$	DIRAC delta function
$\operatorname{sgn}(\cdot)$	Sign function

Dimensions

n_c	Number of path constraints $\boldsymbol{c}(\cdot)$
n_r	Number of boundary constraints $\boldsymbol{r}(\cdot)$
n_x	Number of differential states $\boldsymbol{x}(\cdot)$
n_u	Number of controls $\boldsymbol{u}(\cdot)$

List of Figures

1.1 Trajectories towards an accumulation point. 40
1.2 Attracting sliding mode. 41
1.3 Repulsive sliding mode. 42
1.4 IC and OC for the example $\boldsymbol{f}(x, v) = -(x - v)^2 + 16$. 51
1.5 IC of $\boldsymbol{c}(x, v)$ and its feasible set. 53
1.6 OC of $\boldsymbol{c}(x, v)$ and its feasible set. 54
1.7 VC of $\boldsymbol{c}(x, v)$ and its feasible set. 56

6.1 Direct Methods Overview 145
6.2 Overview: Direct Methods based on control discretization. 156
6.3 Illustration of the direct single shooting discretization. 158
6.4 Illustration of the direct multiple shooting discretization. 160
6.5 Direct Methods with state and control discretization Overview 164
6.6 Plot of several partial sums of the LEGENDRE series expansion for sgn(·). 184

7.1 Sparsity pattern of FRPM constraint Jacobian. 198
7.2 Discretization point distribution for the multi–degree global collocation example. 201
7.3 NLP constraint Jacobian of the multi–degree global collocation example without variable and constraint permutation. 203
7.4 NLP constraint Jacobian of the multi–degree global collocation example with variable and constraint permutation. 204

9.1 Commutativity diagram “FDTO” vs. “FOTD” 237

12.1 States $\boldsymbol{x}_1(\cdot)$ and $\boldsymbol{x}_2(\cdot)$ of Problem (12.1) with 6 collocation points. 300
12.2 States $\boldsymbol{x}_3(\cdot)$ and $\boldsymbol{x}_4(\cdot)$ of Problem (12.1) with 30 collocation points. 301
12.3 States of Problem (12.1) with six collocation points for the first two components and 60 collocation points for the third and fourth component. 302
12.4 Differential states $\boldsymbol{x}(\cdot)$ of OCP (12.4) 303
12.5 Control approximations of OCP 12.4 with six collocation points for the first control component and 30 collocation points for the second control component. 304
12.6 Controls of OCP 12.4 with six collocation points for $\boldsymbol{u}_1(t)$ and 60 for $\boldsymbol{u}_2(t)$. . . 304

13.1 Differential states and phase portrait for OCP (13.1). 307
13.2 Control and adjoint differential states for OCP (13.1). 308
13.3 Differential states and phase portrait for OCP (13.2). 309
13.4 Control and adjoint differential states for OCP (13.2). 310
13.5 Differential states and control for OCP (13.3). 311
13.6 Mixed control–state constraint and its adjoint for OCP (13.3). 312
13.7 Differential states and control for OCP (13.4) (coarse discretization). 313
13.8 Differential costates for OCP (13.4) (coarse discretization). 313

13.9 Constraint costate for OCP (13.4) (coarse discretization). 314
13.10 Differential states and control for OCP (13.4) (fine discretization). 314
13.11 Differential costates for OCP (13.4) (fine discretization). 315
13.12 Constraint costate for OCP (13.4) (fine discretization). 315

14.1 Differential states and control of OCP (14.1) 317
14.2 Estimated error development and FE grid development for OCP (14.1). 318
14.3 Differential states and control of OCP (14.2) 320
14.4 Estimated/exact error and FE grid development for OCP (14.2). 321
14.5 FE grid and polynomial order development for OCP (14.2). 322
14.6 Differential states and control for OCP (14.3) after second iteration. 323
14.7 Mixed control–state constraint and its costate for OCP (14.3) after second iteration. 324
14.8 Differential states and control for OCP (14.3) after seventh iteration. 325
14.9 Mixed control–state constraint and its costate for OCP (14.3) after seventh iteration. 326
14.10 Estimated error development and FE grid development for OCP (14.3). 327
14.11 Differential state and control for OCP (14.4). 327
14.12 Differential state and pure state constraint costates for OCP (14.4) 328

15.1 Coulombic friction model state trajectories. 331
15.2 Coulombic friction model control profiles. 332
15.3 Coulombic friction model mode switching profile. 333
15.4 Stick–slip model state trajectories. 335
15.5 Stick-slip model active modes. 336
15.6 Alternate friction model state trajectories. 337
15.7 Alternate friction model active modes. 338

B.1 Legendre polynomials 353
B.2 Lagrange interpolating polynomials 355
B.3 Comparison of LG, LGR and LGL quadrature nodes. 362
B.4 Distribution of LG and LGL quadrature nodes. 363
B.5 Distribution of LGR quadrature nodes. 363

List of Tables

6.1 Condition numbers of mass matrices. 172
6.2 Order of max–norm errors for Gibbs phenomenon. 185

15.1 Parameters of the coulombic friction model 330
15.2 Parameters of the stick-slip model 334
15.3 Parameters of the alternate friction model 337