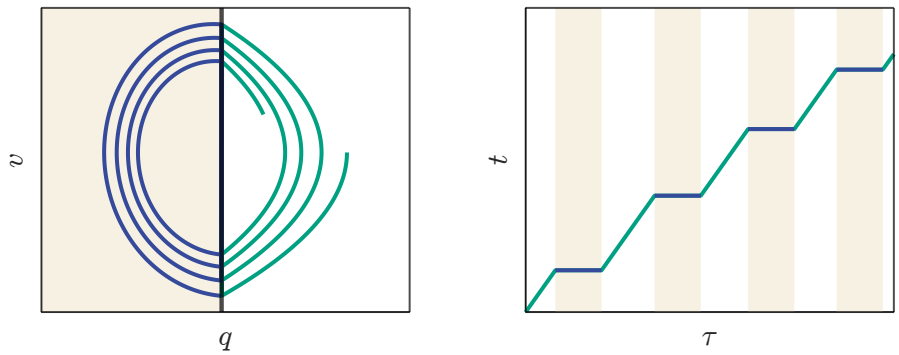
universität freiburg

Numerical Methods for Optimal Control of Nonsmooth Dynamical Systems

Armin Nurkanović



University of Freiburg, Faculty of Engineering
Department of Microsystems Engineering
Systems Control and Optimization Laboratory

November 30, 2023

Numerical Methods for Optimal Control of Nonsmooth Dynamical Systems

Armin NURKANOVIĆ

Dean: Prof. Dr. Frank Balle

Examination committee:

First reviewer: Prof. Dr. Moritz Diehl
Second reviewer: Dr. Bernard Brogliato
Observer: Prof. Dr. Lars Pastewka
Chair of committee: Prof. Dr. Oliver Paul

Dissertation zur Erlangung des Doktorgrades der Technischen Fakultät der Albert-Ludwigs-Universität Freiburg im Breisgau

November 30, 2023

Self-published, Armin Nurkanović, Georges-Köhler-Allee 102, 79110 Freiburg in Br. (Germany)

Acknowledgments

My PhD years were an exciting and very happy time of my life. This is due to some good fortune and many great people who supported me during this time, to whom I would like to express my deepest gratitude.

First and foremost, I would like to thank Moritz Diehl for his exceptional supervision, and many great discussions about science and beyond. I thank Moritz for supporting me in my work, for constantly pushing me to do my best, which improved my skills, but also all of our joint papers. With his mathematical ingenuity, scientific integrity, and optimism, but also his exceptionally friendly and warm character, he is a true role model for me. Our collaboration has been a true privilege and I have enjoyed every moment of it. Moritz also created a very friendly, healthy, productive, and stimulating research environment at the Systems Control and Optimization Laboratory (syscop). I am very grateful to him for this.

Before moving to the University of Freiburg in October 2021, I was employed as an external PhD student at Siemens Research in Munich. This was only possible because Sebastian Albrecht supported me and believed in me. I thank him for introducing me to optimal control in the first place and defining the topic of my PhD thesis, about which I have become obsessed in the past few years. I also thank him for the wonderful supervision during the Siemens years.

I wish to express my gratitude to Bernard Brogliato for reading and reviewing my thesis. Moreover, I thank him for teaching me a lot about nonsmooth mechanics while working on our Automatica paper.

At this point, I would like to thank all my dear Siemens colleagues with whom I had great discussions and many beautiful moments. I thank Georg von Wichert for creating a healthy environment at Siemens T-FOA-ASY-DE. I would like to thank my PhD peers at that time (in alphabetical order): Amer Mešanović, Bernd Kast, Florian Wirnshofer, Philipp Sebastian Schmitt, Sarah Braun, Simon Ackermann, Sissi Bazan Santos, Stefan Löw and Vincent Dietrich. I enjoyed

our scientific discussions, coffee and lunch breaks, and table soccer games. I thank Andrei Szabo for the collaboration in the SynErgie project. I thank Amer Mešanović and the MoreNet team from IWR Heidelberg: Hans Georg Bock, Ekaterina Kostina, Andreas Potschka, Robert Scholz, and Jürgen Gutekunst, for the fruitful collaboration on NMPC for microgrids.

Working at syscop is a true privilege. It is mainly because of all the kind and smart people I have had the opportunity to work with and become friends with. I would like to thank our past and present team members (in alphabetical order): Adrian Bürger, Andrea Ghezzi, Andrea Zanelli, Anton Pozharskiy, Arne Groß, Benjamin Stickan, David Kiessling, Dimitris Kouzoupis, Florian Messerer, Gianluca Frison, Jakob Harzer, Jochem De Schutter, Jonathan Frey, Katrin Baumgärtner, Léo Simpson, Manuel Kollmar, Mikhail Katliar, Per Rutquist, Rachel Leuthold, Robin Verschueren, Rudolf Reiter, Tobias Schöls, and Tommaso Sartor. I thank Andrea Zanelli for his great support in writing my first papers and for teaching me a lot about NMPC and optimization in my early PhD days. I thank Jonathan Frey and Jochem De Schutter for proofreading my thesis. Their comments and feedback (also beyond this thesis) are always very valuable and constructive. I thank Jonathan Frey and Anton Pozharskiy for significantly improving `nosnoc`, and creating `nosnoc_py`. Thank you for being valuable and supportive co-authors on all `nosnoc`-related papers, and for helping to prepare our summer school on nonsmooth optimal control. I owe my sincere gratitude to our team assistant, Christine Paasch, for always quickly solving any problems that arose alongside our research, and for her patience and pragmatism in preparing our summer school. Moreover, I wish to thank Christian Kirches for giving three great lectures in the summer school and for the numerous pleasant interactions we had over the years.

One of the most rewarding parts of the PhD work is mentoring bright young students. I thank all my brilliant master students from whom I have learned a lot (in alphabetical order): Anton Pozharskiy, Christian Dietz, Jonas Hall, Léo Simpson, and Mario Sperl.

I thank my friends Jakob Ungerland, Miloš Katanić, Mirco Theile, and Smajil Halilović for countless fruitful discussions about PhD life and beyond. I would like to thank my dear family for their love and support. I thank my dear parents, Suada and Mirzet, for their support throughout my life and for teaching me how to be persistent in important life matters. As well as my dear brothers Semin and Admir, for their support and love. Last but not least, I would like to thank my wonderful wife Elmira for her love and patience. Thank you for your unconditional love and support in my academic journey. Thank you for learning German and moving to Freiburg for me! *Hvala vam od srca, volim vas.*

Freiburg, Armin Nurkanović

Abstract

This thesis regards algorithmic, theoretical, and software aspects of numerically solving Optimal Control Problems (OCPs) subject to nonsmooth dynamical systems. When solving an optimal control problem, one selects the optimal control action while explicitly considering constraints and system dynamics. The ability to consider constraints and system dynamics increases the expressiveness in the controller design process. This expressiveness is further improved by considering *hybrid systems*. Hybrid systems are a class of nonsmooth dynamical systems characterized by closely coupled continuous and discrete behavior. This coupling results in dynamical systems with continuous but nondifferentiable vector fields, systems with discontinuous vector fields, and systems with state jumps. This allows, for example, the formulation of OCPs for complex robotic tasks or to incorporate Boolean logic relations between parts of the system. However, from a numerical point of view, such OCPs are difficult to solve.

Standard direct methods for solving smooth OCPs applied to nonsmooth OCPs suffer from some fundamental limitations. We develop a toolchain of algorithms and reformulation methods that overcome these limitations and allow one to solve all mentioned classes of nonsmooth OCPs in a unified way. All methods in this toolchain are implemented in the open-source software package `nosnoc`.

Some fundamental limitations of standard direct methods, including time-stepping discretizations, mixed-integer reformulations, and smoothing are highlighted. It is shown that all these approaches suffer from the same limitations unless the discontinuities are explicitly treated by switch detection. They will achieve only first-order accuracy, and the discrete-time numerical sensitivities do not converge to the correct values. As a consequence, the algorithms may converge to spurious local solutions or make almost no progress from a given initial guess. Furthermore, it is discussed why standard direct methods sometimes lead to feasible or seemingly reasonable solutions. In general, obtaining a highly accurate solution with some of these approaches may require a prohibitive computational effort.

The application of direct optimal control methods, based on Newton-type optimization, requires the accurate numerical simulation of nonsmooth systems and the computation of numerical sensitivities. This thesis presents the Finite Elements with Switch Detection (FESD) method for Filippov systems, which achieves these two goals and thereby overcomes the fundamental limitation of standard methods. The focus is on Filippov systems, as they provide a sound solution concept for ODEs with a discontinuous right-hand side. The proposed approach reformulates these systems into equivalent Dynamic Complementarity Systems (DCSs). After the time discretization, mathematical programs with complementarity constraints are obtained, which can be solved efficiently with a homotopy approach using standard nonlinear programming solvers. We provide a detailed theoretical analysis of the FESD method and show that it is superior to time-stepping methods in terms of computation time and accuracy. For example, we achieve in an OCP benchmark up to one million times more accurate solutions for the same computational time.

Systems with state jumps are not Filippov systems. Therefore, they cannot be treated with the methods developed for this class of nonsmooth systems. We introduce the time-freezing reformulation, which transforms systems with state jumps into equivalent Filippov systems. The main idea of time-freezing is to define a clock state and an auxiliary ODE in the infeasible region of the state space of the original system. The endpoints of the trajectory of the auxiliary ODE satisfy the state jump law of the original system. Moreover, the evolution of the clock state is *frozen* during the runtime of the auxiliary ODE. By considering only the parts of the trajectory where the clock state evolves, one can reconstruct the solution of the original system with state jumps. This allows one to seamlessly apply the FESD method and the theory of Filippov systems to systems with state jumps.

As a somewhat isolated contribution from the topics above, we present several new real-time algorithms for nonlinear model predictive control for smooth dynamical systems. We extend the well-known Real-Time Iteration (RTI), by combining the algorithmic ideas from the Multi-Level Iteration (MLI) and the Advanced Step Controller (ASC). We call this method the Advanced-Step Real-Time Iteration (AS-RTI). The main idea is to improve the current linearization point between two samples by iterating with some MLI variant on an advanced problem with a predicted state while waiting for the next state estimate. The AS-RTI scheme bridges the gap between the two well-established algorithmic paradigms: the ASC which solves the OCP to convergence at every sampling time and the RTI which performs only one Newton-type iteration.

Kurze Zusammenfassung

Diese Arbeit befasst sich mit algorithmischen, theoretischen und softwaretechnischen Aspekten der numerischen Lösung von Optimalsteuerungsproblemen (Englisch: Optimal Control Problem (OCP)), mit nicht-glatten dynamischen Systemen. Bei der Lösung eines OCPs wird der optimale Steuerungseingang unter expliziter Berücksichtigung von Nebenbedingungen und der Systemdynamik ausgewählt. Der Entwurfsprozess eines Reglers auf der Grundlage von OCPs ist durch die explizite Formulierung der Systemdynamik und der Nebenbedingungen besonders ausdrucksstark. Diese Ausdrucksfähigkeit wird weiter verbessert, wenn sogenannte "hybride Systeme" berücksichtigt werden. Hybride Systeme sind eine Klasse nicht-glatter dynamischer Systeme, die sich durch eng gekoppeltes kontinuierliches und diskretes Verhalten auszeichnen. Dies führt zu dynamischen Systemen mit kontinuierlichen, aber nicht differenzierbaren Vektorfeldern, Systemen mit diskontinuierlichen Vektorfeldern und Systemen mit Zustandssprüngen. Dadurch wird beispielsweise die Formulierung von OCPs für komplexe Robotikaufgaben oder die Abbildung logischer Beziehungen zwischen Systemteilen ermöglicht. Aus numerischer Sicht sind solche Probleme jedoch schwierig zu lösen.

Herkömmliche direkte Methoden zur Lösung glatter OCPs, angewendet auf nicht-glatte OCPs, haben grundlegende Einschränkungen. In dieser Arbeit wird eine Werkzeugkette von Algorithmen und Umformulierungsmethoden entwickelt, um diese Einschränkungen zu überwinden und die einheitliche Lösung aller zuvor genannten Arten von nicht-glatten OCPs zu ermöglichen. Alle Methoden dieser Werkzeugkette sind in dem Open Source Softwarepaket `nosnoc` implementiert. Es werden einige grundlegende Einschränkungen konventioneller direkter Methoden aufgezeigt, einschließlich der Diskretisierung mit Zeitschrittmethoden, gemischt-ganzzahliger Umformulierung und Glättung, um nur einige Beispiele zu nennen. Es wird gezeigt, dass alle diese Ansätze unter den gleichen Einschränkungen leiden, es sei denn, die Unstetigkeiten werden explizit durch Schalterkennung behandelt.

Herkömmliche direkte Methoden erreichen nur eine Genauigkeit erster Ordnung, und die zeitdiskreten numerischen Sensitivitäten konvergieren nicht zu den richtigen Werten. Infolgedessen können die Algorithmen zu lokalen Scheinlösungen konvergieren oder von einer gegebenen Lösungsschätzung kaum Fortschritte machen. Es wird auch diskutiert, warum konventionelle direkte Methoden manchmal zu scheinbar vernünftigen Lösungen konvergieren. Im Allgemeinen kann das Erreichen einer hochgenauen Lösung mit einigen dieser Ansätze einen inakzeptablen Rechenaufwand erfordern.

Direkte Methoden zur Lösung von OCPs, die auf Newton-basierten Optimierungsalgorithmen basieren, erfordern die genaue numerische Simulation von nicht-glatten Systemen und die Berechnung numerischer Sensitivitäten. In dieser Arbeit wird die Finite Elemente Methode mit Schalterkennung (FESD) für Filippov Systeme entwickelt, die beide Ziele erreicht und damit die grundlegenden Einschränkungen der Standardmethoden überwindet. Filippov-Systeme sind von besonderem Interesse, da sie ein solides Lösungskonzept für gewöhnliche Differentialgleichungen mit diskontinuierlicher rechter Seite liefern. Unser Ansatz besteht darin, diese Systeme in äquivalente dynamische Komplementaritätssysteme (Englisch: Dynamic Complementarity System (DCS)) umzuformulieren. Nach der zeitlichen Diskretisierung erhält man mathematische Programme mit Komplementaritätsbeschränkungen (Englisch: Mathematical Program with Complementarity Constraints (MPCC)), die mit einem Homotopie-Ansatz unter Verwendung von Standardlösern für nichtlineare Optimierung effizient gelöst werden können. Eine detaillierte theoretische Analyse der FESD-Methode wird vorgestellt und es wird gezeigt, dass diese den Zeitschrittmethoden in Bezug auf Rechenzeit und Genauigkeit überlegen ist. Beispielsweise werden in einem OCP-Benchmark bis zu einer Million Mal genauere Lösungen bei gleicher Rechenzeit erreicht.

Systeme mit Zustandssprüngen sind keine Filippov-Systeme. Daher können sie nicht mit den Methoden behandelt werden, die für diese Klasse von nicht-glatten Systemen entwickelt wurden. Um diese Beschränkung zu überwinden, wird die Time-Freezing-Reformulation eingeführt, um Systeme mit Zustandssprüngen in äquivalente Filippov-Systeme umzuformulieren. Die Hauptidee des Time-Freezing besteht darin, einen Uhrenzustand und eine Hilfsdifferentialgleichung im unzulässigen Bereich des Zustandsraums des ursprünglichen Systems zu definieren. Die Endpunkte der Trajektorie der Hilfsdifferentialgleichung erfüllen das Zustandssprunggesetz des ursprünglichen Systems. Außerdem ist die Evolution des Uhrzustands während der Laufzeit der Hilfsdifferentialgleichung *eingefroren*. Betrachtet man nur die Teile der Trajektorie, in denen sich der Uhrenzustand entwickelt hat, so kann man die Lösung des ursprünglichen Systems mit Zustandssprüngen rekonstruieren. Dies ermöglicht die Anwendung der FESD-Methode und der Theorie der Filippov-Systeme auf Systeme mit

Zustandssprüngen.

Als etwas isolierter Beitrag zu den oben genannten Themen werden mehrere neue Echtzeit-Algorithmen für nichtlineare modellprädiktive Regelung für glatte dynamische Systeme vorgestellt. Die bekannte Real-Time Iteration (RTI) wird erweitert, indem die algorithmischen Ideen der Multi-Level Iteration (MLI) und des Advanced Step Controllers (ASC) kombiniert werden. Diese neue Methode wird Advanced-Step Real-Time Iteration (AS-RTI) genannt. Die Hauptidee besteht darin, den aktuellen Linearisierungspunkt zwischen zwei Samples zu verbessern, indem eine MLI-Variante auf einem fortgeschrittenen Problem mit einem vorhergesagten Zustand iteriert wird, während auf die nächste Zustandsschätzung gewartet wird. Der AS-RTI-Algorithmus schließt die Lücke zwischen zwei etablierten Algorithmen: dem ASC-Algorithmus, der das OCP in jedem Sample bis zur Konvergenz löst, und dem RTI-Algorithmus, der nur eine Newton-Iteration durchführt.

Contents

Chapter 1

Introduction

Mathematical optimization is an indispensable tool in almost all fields of science and engineering. It enables us to naturally express what we almost always want to achieve: find the best solution to our problem, given an objective and constraints. To write an optimization problem in terms of equations, we need a *model*, which is a simplified description of the real world that quantifies some of its aspects that are relevant for the considered purpose. Moreover, it helps one to make more or less accurate predictions of the future, and to study some non-obvious properties of the phenomenon that one is modeling. A rich family of mathematical models are dynamical systems, which describe the time evolution of a system. They are often based on underlying physical laws, and they can be very accurate.

Having a (physical) system and its mathematical model at hand, we may want to behave in a certain way by influencing what we can influence. Moreover, we want it to work reliably without our frequent intervention. This gives rise to the field of *automatic control*, where based on our current knowledge (the system's state), we decide what actions to take (the control inputs) to achieve our goal. Determining the system's state and deciding on the control input can be arbitrarily difficult. Inevitably, we want to do this in the best possible way. This gives rise to *Optimal Control Problems* (OCPs).

Formally, OCPs are infinite-dimensional problems that we solve *open loop* starting from a fixed initial state. However, our model might be inaccurate, unforeseen changes may happen in the state or in external factors that we cannot model accurately (the weather, stock prices, states of other systems, etc.). To some extent, this can be alleviated by Model Predictive Control (MPC), where at every sampling instant a new OCP with the most recent state estimate

as initial value, is solved. MPC is becoming a standard control technique in academia and industry. Its main advantages are: that the control goal can be directly formulated as the objective of an optimization problem, constraints on the system and control are part of the problem formulation, and a model for predicting the future is explicitly considered in the search for a control input. In the case that our models are sufficiently regular and smooth functions, MPC is a mature and reliable control technique. In practice, direct methods have proven highly effective. They discretize the infinite-dimensional OCP and then solve a finite-dimensional problem. They offer real-time capable algorithms, a mature theory, and sophisticated software implementations.

To improve real-world performance or to automate more complex tasks, we must use increasingly sophisticated models. For example, we may want to have if-then or either-or conditions within our model, or we may want to approximate complex microscopic physical phenomena with nonsmooth macroscopic empirical laws as in Coulomb's friction law. This gives rise to so-called *hybrid systems*. In this case, the discrete and continuous dynamics do not just coexist but are closely interconnected. Mathematically, this leads to nonsmooth dynamical systems. In simple terms, they arise whenever the functions defining the underlying differential equations are not differentiable anymore. This opens the possibility to have more sophisticated models or to simplify the formulation of the problem. However, a general rule seems to be: the more expressive the model, the more difficult the optimization problems become. Even the simplest nonsmooth dynamical system is extremely nonlinear, and the resulting OCPs are necessarily difficult nonconvex optimization problems.

Traditional mixed-integer optimal control selects control inputs from a finite set of values. These problems are also discrete-continuous. The optimizer *explicitly* chooses which discrete value the control input should take. In contrast to that, in nonsmooth dynamical systems, the discrete decisions are *implicit* and *state dependent*. This highlights that these systems are different, and we will see in this thesis that it might not be beneficial to treat nonsmooth OCPs with mixed-integer reformulations. An alternative and obvious approach to solving OCPs subject to nonsmooth systems is to smooth the problem and apply standard, well-established methods. However, accurate smooth approximations lead to at least equally difficult problems. We will show that standard smoothing methods fail in surprising and non-obvious ways. We argue that attempting to smooth out all nonsmoothness is not the way to go, as it often seems from a numerical perspective that smooth approximations behave the same as nonsmooth systems. Moreover, smoothing hides the nonsmoothness from the problem, which usually appears in a very structured way and should be exploited.

Complementarity conditions frequently appear as a form of structured nonsmoothness. Given two scalars $x, y \in \mathbb{R}$, a complementarity condition

reads as

$$0 \leq x \perp y \geq 0,$$

which means that both x and y have to be nonnegative but they cannot be both positive simultaneously, i.e., $xy = 0$. They appear, for example, in the well-known Karush-Kuhn-Tucker (KKT) conditions, which provide necessary conditions for a local minimizer of an optimization problem. Beyond that, they enable us to model numerous combinatorial structures. From a numerical point of view, what makes them so practical is that they can be very efficiently treated with Newton-type methods without any enumeration. In this thesis, we will often end up solving complementarity problems.

1.1 Classification of nonsmooth dynamical systems

Not all nonsmooth optimal control problems are equally difficult. Depending on where and how the nonsmoothness appears, the systems allow for different algorithms and theoretical conclusions. In this thesis, we classify the nonsmooth dynamical systems depending on the smoothness of the vector field and the solution trajectory. Given an ODE $\dot{x}(t) = f(x)$, with the right-hand side (r.h.s.) $f(x)$, we distinguish between Nonsmooth Dynamics (NSD) of three levels:

(NSD1) Continuous, but nondifferentiable r.h.s.- continuously differentiable solutions.

(NSD2) Discontinuous r.h.s. - continuous solutions.

(NSD3) Jump discontinuity in the solutions.

We illustrate these classes with a few examples to highlight some of their main qualitative properties. Figure 1.1 shows sample solutions of our examples. An NSD1 system with a continuous r.h.s. is:

$$\dot{x} = 1 + |x|.$$

This system already poses difficulties to standard methods, as the derivatives of the solution map w.r.t. the initial values are nonsmooth, and the integration methods might lose their accuracy properties.

NSD2 systems have a discontinuous r.h.s., which makes already most nonlinear analysis tools not applicable. The discontinuities allow for rich behavior, e.g., the ODE:

$$\dot{x} = 2 - \operatorname{sign}(x),$$

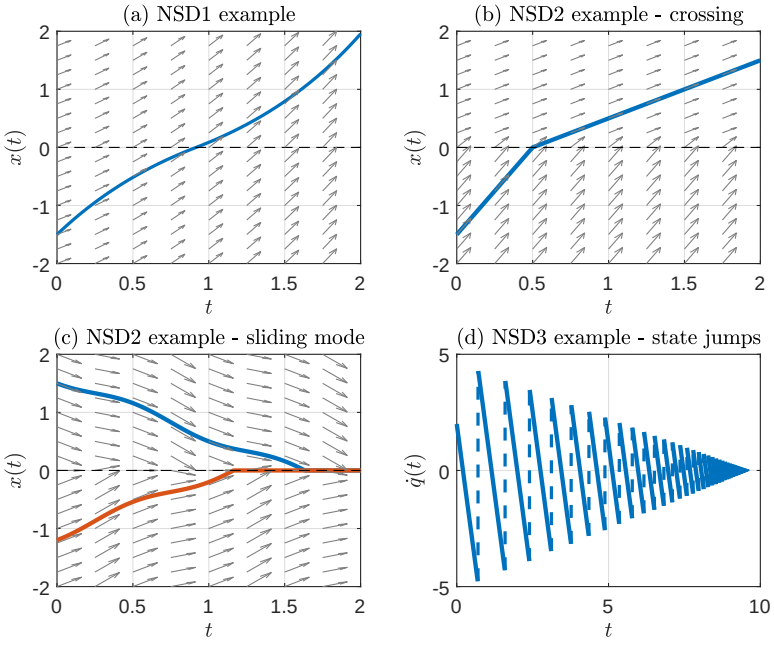


Figure 1.1: Example trajectories for NSD1 to NSD3 systems.

results in a trajectory that crosses the surface of discontinuity defined by $x = 0$, cf. Figure 1.1(b). On the other hand, in examples such as

$$\dot{x} \in -\text{sign}(x) + 0.5\sin(t),$$

the trajectories end up in a so-called *sliding mode*. The trajectories reach the surface of discontinuity $x = 0$ and stay there since the vector fields from both sides push toward $x = 0$, cf. Figure 1.1 (c). Moreover, standard notions of solutions are not applicable anymore, and we need to use a set-valued generalization of ODE. In our example we define, $\text{sign}(0) = [-1, 1]$, such that at $x = 0$ we have that $0 \in 0.5\sin(t) - \text{sign}(x)$ becomes meaningful. This generalization is an instance of Filippov systems, which we study in great detail in this thesis. Remarkably, higher-order Runge-Kutta methods applied to NSD2 systems, if they converge, have only first-order accuracy.

Probably the most famous example of an NSD3 system is the bouncing ball. If q is the ball's position, g the gravitational acceleration, $m = 1$ the mass, the equations of motions are:

$$\ddot{q} = -g,\ \dot{q}(t_{\text{s}}^{+}) = -\epsilon_{\text{r}}\dot{q}(t_{\text{s}}^{-}),\ \text{if } q(t_{\text{s}}) = 0 \text{ and } q(t_{\text{s}}^{-}) < 0.$$

The second part of the equations is the state jump law, which says: if the ball hits the ground ($q(t_s) = 0$), then the post-impact velocity is given by $\dot{q}(t_s^+) = -\epsilon_r \dot{q}(t_s^-)$, with $\epsilon_r \in [0, 1]$, cf. Figure 1.1 (d).

The nonsmoothness in models creates difficulties in all aspects of optimal control problems involving such systems. This thesis aims to address these difficulties by understanding the limitations of existing methods, by developing new tailored methods that overcome these limitations, by establishing a sound theory for the proposed methods, and by providing open-source software implementations of them. To advance the state-of-the-art, we present novel ideas and improvements to existing approaches for solving OCPs subject to nonsmooth dynamical systems.

Figure 1.2 depicts the toolchain developed in this thesis. We will briefly explain its main parts here. Given a dynamical system of NSD1 to NSD3, the objective function, and constraints, we want to formulate and numerically solve an OCP. Mathematically, NSD1 systems can be treated as a special case and a more regular case of NSD2 systems. In both cases, the state space is partitioned into some regions equipped with different vector fields. In NSD1 the vector fields are continuous across the region boundaries, and in NSD2 they are not. Moreover, we introduce the time-freezing reformulation, which enables us to transform NSD3 into equivalent NSD2 systems. Therefore, we can focus only on NSD2 systems and treat all classes in a unified way. We use Filippov's notion of solution as a meaningful and theoretically sound concept for NSD2 systems. Next, we transform the Filippov systems into dynamic complementarity systems, which are a coupling of an ODE and a complementarity problem. Complementarity conditions enable us to encode all nonsmooth and combinatorial structures in the dynamical system.

The next step is to formulate an OCP and to discretize it. For the time discretization of the OCP, we develop the Finite Elements with Switch Detection (FESD) method. This method recovers the properties that Runge-Kutta methods have for smooth ODEs: high accuracy and correct sensitivities. Leaving the FESD step out and using a standard time-stepping method leads to some variants of existing direct methods in the literature. However, we show that they are doomed to fail and generally cannot lead even to locally optimal solutions. After discretization, we obtain a Mathematical Program with Complementarity Constraints (MPCC). This is a degenerate Nonlinear Program (NLP) that cannot be solved directly with standard NLP solvers. Fortunately, we can solve a sequence of related and more regular problems with standard optimization methods in a homotopy loop, and obtain a solution for the initial MPCC. Finally, this provides a very accurate solution approximation to an OCP subject to an NSD1, NSD2, or NSD3 system.

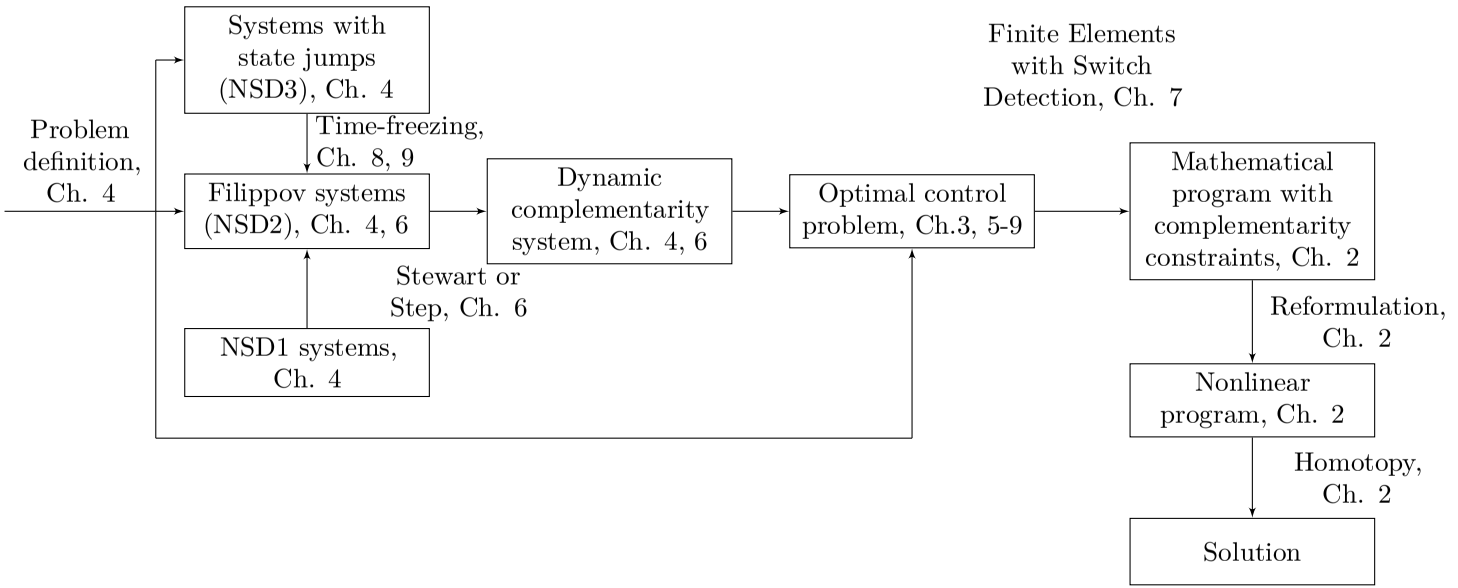


Figure 1.2: Overview of the toolchain for numerically solving nonsmooth optimal control problems developed in this thesis.

1.2 Contributions and outline

We give an overview of the content in this thesis and list the most important contributions. Figure 1.2 provides a graphical illustration of the toolchain developed in this thesis, with references to chapters where specific parts are introduced or discussed in more detail.

1.2.1 Software contributions - `nosnoc`

Open-source implementations make algorithms available, deployable, and useful. The software contribution of this thesis is the implementation of the whole toolchain from Figure 1.2 in the open source software package `nosnoc` (NOnSmooth Numerical Optimal Control)[1]. `nosnoc` for `MATLAB` is based on the symbolic framework of `CasADi` [9]. Recently, the author's colleagues have also implemented a version in `python`, which they have named `nosnoc_py`[2].

We mention some key features. The model functions, constraints, objective, and switching functions are expressed via `CasADi` symbolic variables and functions. The software supports an automatic reformulation of systems with state jumps into NSD2 systems via the time-freezing reformulation. Additionally, we have implemented automatic reformulation into dynamic complementarity systems, and time discretization using FESD or standard time-stepping methods. After the discretization, we obtain an MPCC. In `nosnoc`, they are automatically formulated and solved in a homotopy loop via some standard regularization method. Both numerical simulation and optimal control problems are supported. Moreover, automatic reformulation of time-optimal control problems, path complementarity constraints, various relaxations of terminal constraints, control of sparsity in complementarity constraints, and different options for the homotopy loop are supported. Numerous other expert and non-expert options are available as well.

The methods and software developed in this thesis facilitate the implementation, prototyping, and solution of optimal control problems involving nonsmooth dynamical systems. Furthermore, we aim to implement new developments or additional components in the same software toolchain.

For brevity, we have not included an additional chapter on the implementation details. However, all mathematical developments are explained in detail in the subsequent chapters. Moreover, all examples shown in this thesis, and several

[1] `https://github.com/nurkanovic/nosnoc`
[2] `https://github.com/FreyJo/nosnoc_py`

other examples from the literature, are available in `nosnoc`'s repositories. They allow us to efficiently explore all possible options and details.

1.2.2 Outline and specific contributions

The main contributions of this thesis are novel numerical methods for solving optimal control problems subject to NSD1 to NSD3-type nonsmooth dynamical systems. These contributions, along with introductory material, are discussed in Chapters 2-9. A secondary and less central contribution of this thesis involves the development of real-time and accurate numerical methods for nonlinear Model Predictive Control (MPC) applied to smooth dynamical systems. This aspect is addressed in Chapter 10. We proceed with a detailed outline and listing of the specific contributions.

Chapter 2 - Nonlinear Optimization with Complementarity Constraints In Section 2.1, we discuss the fundamentals of nonlinear programming. Section 2.2 discusses variational inequalities, complementarity problems, and generalized equations and shows when they are equivalent. It is important to be familiar with them, as in a dynamic setting they are the sources of nonsmoothness. NLPs subject to complementarity constraints lead to MPCCs. All numerical methods for nonsmooth OCPs developed in the thesis require the solution of an MPCC at some point. They are nonsmooth optimization problems that violate standard constraint qualifications at all feasible points, which leads to theoretical and algorithmic difficulties. We review the theory and standard numerical methods for solving this class of problems in Sections 2.3 and 2.4, respectively. Relaxation, smoothing, and penalty methods solve a (finite) sequence of related more regular problems and obtain in the limit a solution of the initial MPCC. They are our methods of choice as they are easy to implement if one has a robust nonlinear programming solver implementation. In our case, we use `IPOPT` [281].

Chapter 3 - Direct Optimal Control Methods Direct optimal control methods first discretize an infinite-dimensional OCP subject to a smooth dynamical system and then solve a finite-dimensional NLP. For the discretization of the dynamics we require numerical integration methods, and for optimization the computation of the derivatives. We discuss this in Section 3.2. The chapter finishes with Section 3.3, where we discuss optimal control problems and solution approaches, with a focus on direct optimal control. These methods are the basis for the tailored methods which we develop in this thesis. For example, we review Runge-Kutta and collocation methods for smooth ODEs and DAEs in Section 3.2.1, which we generalize to NSD2 systems in Chapter 7.

Chapter 4 - Nonsmooth Dynamical Systems To numerically solve nonsmooth optimal control problems, we require the tools from several different fields, cf. Figure 1.2. Together with Chapters 2 and 3, this chapter completes a self-contained introduction to the methods developed in this thesis for researchers with a background either in numerical optimal control or in nonsmooth dynamical systems. Therefore, we focus on numerical difficulties that arise with nonsmooth systems and their smooth approximations. In Section 4.1, we provide a high-level introduction to nonsmooth dynamical systems and answer some natural questions that arise. Section 4.2 illustrates on tutorial examples several phenomena encountered only within nonsmooth systems and their time discretization. In particular, we show, via extensive numerical experiments, the surprising fact that standard integration methods behave the same on accurate smooth approximations as on the original nonsmooth systems.

The same set of tools can be used to study both the theoretical and numerical aspects of sufficiently regular, smooth nonlinear dynamical systems. In the nonsmooth cases, this is, unfortunately, much more complicated. There are many ways to model nonsmoothness, resulting in various mathematical frameworks with different degrees of difficulty and their own theory and numerical methods. In Section 4.3, we review some basics from nonsmooth analysis and standard modeling frameworks for nonsmooth systems. We recall some existence and uniqueness results with a focus on systems studied in this thesis.

Chapter 5 - Limitations in Nonsmooth Direct Optimal Control In this chapter, we highlight some of the fundamental limitations of standard methods, which refer to the application of time-stepping methods in direct transcription, direct multiple, or single shooting. In this case, after discretization one obtains a nonsmooth NLP. The nonsmoothness can be smoothed explicitly or implicitly, before or after the discretization. In Section 5.1, we survey direct methods for nonsmooth OCPs from the literature and highlight their strengths and weaknesses. Section 5.2 is inspired by the seminal paper of Stewart and Anitescu [259]. It shows that sensitivities obtained in a direct method paired with a time-stepping discretization are wrong, no matter how small the step size is, and that smoothing works only under very restrictive assumptions. Consequently, the optimizer may make almost no progress from the initial guess. Based on [203], we show that equivalent formulations of the problem, which lead to different algorithms, suffer from the same limitations. Section 5.3 shows that similar limitations exist for direct methods for NSD3-type systems. We also discuss why these methods still converge to feasible but suboptimal solutions, which explains their occasional success in practice.

The main contribution of this chapter is to show why standard methods fail

and why they sometimes deliver feasible solutions. This shows the necessity for tailored methods, which we develop in the subsequent chapters, and which resolve the fundamental limitations of standard methods.

Chapter 6 - Reformulation of Filippov Systems into Dynamic Complementarity Systems In a Piecewise Smooth System (PSS) the state space is partitioned into regions R_i, and each region has a different vector field f_i. In this chapter, we regard PSS, their Filippov convexification, and their reformulation into equivalent Dynamic Complementarity Systems (DCSs). These systems allow us to cover a broad class of practical problems and they have a sound theory. Sections 6.2 and 6.3 regard Stewart's [250] and the set-valued Heaviside step function approach [6, 74] to compute the Filippov convex multiplier as the solution of a Linear Program (LP), respectively. Using the KKT conditions of these LPs, Filippov systems can be reformulated into equivalent DCSs. We study in great detail the favorable properties of these DCS, e.g., the uniqueness of solutions for a fixed active set and the continuity of the Lagrange multipliers. The specific contributions are:

- In Section 6.2.1, we provide a generic way to pass from the definition of regions R_i via standard switching functions $\psi(x)$ to the less intuitive representation of Stewart given via indicator functions $g(x)$.
- Compared to [250], we provide in Section 6.2.2 a simpler study of the well-posedness of the ordinary and algebraic differential algebraic equations obtained from fixing the active set in the DCS.
- In Section 6.2.3, we study the continuity properties of the algebraic variables in the DCS and recall some results from [250].
- In Section 6.3 we provide a compact representation of regions via switching functions $\psi(x)$ and prove that we can construct a Filippov set from this representation via set-valued Heaviside step functions.
- Analogously to Stewart's case, in Section 6.3.5, we derive well-posedness results for the step reformulation DCS with a fixed active set.
- We discuss how to compactly derive expressions for Filippov multipliers via step functions for standard geometries appearing in practice, i.e., when the regions are defined as unions, intersections, and differences of two or more sets.
- In the Heaviside step reformulation, the Filippov multipliers θ_i are expressed via multi-affine terms, which may become very nonlinear. In

Section 6.3.8, we introduce a lifting algorithm which automatically defines intermediate variables that reduce the nonlinearity.

This chapter is based on the articles [207, 208, 213].

Chapter 7 - Finite Elements with Switch Detection In this chapter, we develop the Finite Elements with Switch Detection (FESD) method, which can be used for numerical simulation and the discretization of optimal control problems. In Sections 7.2 and 7.4, we introduce the three main ingredients of FESD. First, we start with a standard Runge-Kutta (RK) discretization for the DCS obtained in the previous chapter. Furthermore, inspired by Baumrucker and Biegler [28], we let the integration step sizes be degrees of freedom. Second, we introduce additional complementarity conditions (called *cross complementarity* conditions) to enable implicit and exact switch detection. Third, to remove possible spurious degrees of freedom if no switches occur, we introduce the *step equilibration* conditions. Notably, FESD is an event-based integration method that is, unlike other methods from the literature, entirely equations-based. In Section 7.3, we provide a detailed theoretical analysis of the method. Section 7.5 discusses how to use FESD in direct optimal control. More specifically, the main contributions of this chapter are:

- In Section 7.2, we start from a standard RK discretization and derive the FESD method step-by-step for Stewart's reformulation from Section 6.2.
- Furthermore, we extend FESD to the step reformulation from Section 6.3. This makes FESD applicable to differential inclusions obtained via step functions, which can be more general than Filippov systems.
- In Theorem 7.7, we prove that the FESD subproblems, despite being always over-determined, have locally isolated solutions.
- In Theorem 7.10, we prove that the FESD method has the same order of accuracy as the underlying Runge-Kutta (RK) method. Thus, we recover the high-accuracy properties that RK methods have for smooth ODEs.
- In Theorem 7.12, we prove that the numerical sensitivities converge to their correct values. This resolves the fundamental issues of standard methods discussed in Chapter 5.
- All theoretical findings are illustrated on numerical examples. Remarkably, compared to a standard RK discretization, we show that with FESD, we can obtain up to one million times more accurate solutions for the same computational time.

This chapter is based on the articles [206, 207, 208, 213].

Chapter 8 - The Time-Freezing Reformulation for Nonsmooth Mechanical Systems The time-freezing reformulation enables one to transform an NSD3-type system into an equivalent PSS. This enables the seamless application of FESD to systems with state jumps and a unified treatment of many systems from NSD1 to NSD3. In this chapter, we regard Complementary Lagrangian Systems (CLSs) both with elastic and inelastic impacts with friction. These systems are ubiquitous in modern robotic applications. Frictional impacts lead to state jumps in the velocity, which complicates the numerical treatment of CLS. To transform CLS into an equivalent PSS system, we propose the following. In the infeasible parts of the CLS, we introduce auxiliary ODEs whose trajectory endpoints satisfy the point-wise state jump laws. Moreover, we introduce a clock state that *freezes* its evolution during the runtime of the auxiliary ODE. This new system has only discontinuities in the right-hand side of the ODE, but not in the solution itself anymore. By taking the pieces of the trajectory when the clock state was running, we recover the solution of the initial system. In Section 8.1, we provide more background on CLS and discuss related work. Sections 8.3 and 8.4 develop the time-freezing reformulation for elastic and inelastic impacts, respectively. Next, in Section 8.5, we formulate an OCP subject to time-freezing systems and relate it to the OCP subject to the initial CLS. More specifically, the contributions of this chapter are as follows:

- In Proposition 8.4, we provide a constructive way to derive auxiliary ODEs for affine constraints and elastic impacts. We extend this to general nonlinear constraints and prove solution equivalence in Theorem 8.5.
- We extend time-freezing to the inelastic impact case in Section 8.4. Proposition 8.8 provides a constructive way to find auxiliary ODEs for general nonlinear constraints.
- Theorem 8.10 shows that the sliding mode of the time-freezing systems is unique and equivalent to the dynamics of the CLS in persistent contact mode. Furthermore, we prove in Theorem 8.12 that the initial CLS and time-freezing system are equivalent.
- Section 8.4.3 extends the developments to inelastic impacts with friction. We show how to construct auxiliary ODEs in this case.
- Theorem 8.14 generalizes Theorem 8.12 for the case of stick and slip friction. Moreover, Theorem 8.16 established the solution equivalence in the frictional cases.

- Section 8.5 describes how time-transformations can be used in OCPs to achieve equidistant control grids in physical time, a desirable property in feedback control. Additionally, we establish the relationship between the solution of the time-freezing OCP and the solution of an OCP subject to a CLS.
- Section 8.6 provides several OCP examples, including a hopping robot and a manipulation task, with time-freezing and FESD. The methods find surprisingly creative solutions without any hints or sophisticated initial guesses.

This chapter is mainly based on the articles [212] and [202].

Chapter 9 - The Time-Freezing Reformulation for Nonsmooth Systems with Hysteresis In this chapter, we extend the time-freezing reformulation to a class of hybrid automata, namely hybrid systems with a hysteresis. This class is not closely related to CLS. However, we build upon the same core ideas: introducing auxiliary dynamics in infeasible regions to mimic state jumps and freezing the time during their evolution. From a practical perspective, it allows for the use of FESD, which facilitates highly accurate numerical optimal control of hybrid systems with hysteresis. The main contributions are as follows:

- In Section 9.2, we develop the time-freezing reformulation for a class of hybrid systems with hysteresis. We provide constructive ways to derive auxiliary ODEs and so-called DAE-forming ODEs.
- In Section 9.3, we prove the equivalence between the time-freezing and the hysteresis system.
- In Section 9.4, we show on a numerical example that time-freezing, together with FESD delivers faster and more accurate solutions than mixed-integer formulations.

This chapter is mainly based on the article [205].

Chapter 10 - The Advanced Step Real-Time Iteration for Nonlinear Model Predictive Control This chapter is isolated from the previous chapters and treats a different topic. It introduces a family of new algorithms for real-time nonlinear Model Predictive Control (MPC). We introduce the Advanced Step Real-Time Iteration (AS-RTI) scheme, which is an extension to the well-known Real-Time Iteration (RTI) scheme [75, 76]. We combine algorithmic ideas of the

RTI, Advanced Step Controller (ASC) [290] and Multi-Level Iteration (MLI) [42]. This approach allows for flexible trade-offs between control performance and computational efficiency. It provides a way to balance these two factors according to the needs of a particular application. The main idea is to improve the linearization point for a new iteration by making cheap iterations with a new initial parameter prediction.

- The AS-RTI bridges the gap between two well-established algorithmic paradigms: the ASC that solves the OCP to convergence at every sampling time, and the RTI that performs only one Newton-type iteration. This allows flexibility in algorithmic design.
- In Section 10.4, we study the convergence properties and error bounds of the proposed methods in a quite general setting, i.e., with the presence of inequality constraints and active-set changes.
- Section 10.5 illustrates the method's performance on numerical example.

This chapter is based on [214, 215, 204], which introduce the AS-RTI and extensions to the MLI, and [211, 210], which applies the new methods in simulations of islanded microgrid control.

Chapter 11 - Conclusions and Future Research This chapter concludes this thesis and summarizes its contributions. We discuss the main advantages and limitations of the methods and the theory that has been developed. Furthermore, we provide several ideas for the improvements of the current methods and discuss future research directions.

1.3 List of publications

Journal publications (as the main author)

1. Nurkanović, A., Sartor, T., Albrecht, S., Diehl, M. (2020). A time-freezing approach for numerical optimal control of nonsmooth differential equations with state jumps. IEEE Control Systems Letters, 5(2), 439-444.
2. Nurkanović, A., Diehl, M. (2022). Continuous optimization for control of hybrid systems with hysteresis via time-freezing. IEEE Control Systems Letters, 6, 3182-3187.

3. Nurkanović, A., Diehl, M. (2022). NOSNOC: A software package for numerical optimal control of nonsmooth systems. IEEE Control Systems Letters, 6, 3110-3115.

4. Nurkanović, A., Albrecht, S., Brogliato, B., Diehl, M. (2023). The time-Freezing reformulation for numerical optimal control of complementarity lagrangian systems with state jumps. Automatica, 158, 111295.

5. Nurkanović, A., Frey, J., Pozharskiy, A., Diehl, M. (2024). FESD-J: Finite Elements with Switch Detection for Numerical Optimal Control of Rigid Bodies with Impacts and Coulomb Friction. Nonlinear Analysis: Hybrid Systems, 52, 101460.

6. Nurkanović, A., Mešanović, A., Sperl, M., Albrecht, S., Münz, U., Findeisen, R., Diehl, M. (2020). Optimization-based primary and secondary control of microgrids. at-Automatisierungstechnik, 68(12), 1044-1058.

Journal publications (as co-author)

7. Hall, J., Nurkanović, A., Messerer, F., Diehl, M. (2021). A sequential convex programming approach to solving quadratic programs and optimal control problems with linear complementarity constraints. IEEE Control Systems Letters, 6, 536-541.

8. Gutekunst, J., Scholz, R., Nurkanović, A., Mešanović, A., Bock, H. G., Kostina, E. (2020). Fast moving horizon estimation using multi-level iterations for microgrid control. at-Automatisierungstechnik, 68(12), 1059-1076.

9. Hall, J., Nurkanović, A., Messerer, F., Diehl, M. (2022). LCQPow–A Solver for Linear Complementarity Quadratic Programs. Mathematical Programming Computation (accepted for publication).

10. Reiter R., Nurkanović A., Frey J., Diehl M,. (2023). Frenet-Cartesian Model Representations for Automotive Obstacle Avoidance within Nonlinear MPC. European Journal of Control, 100847.

Submitted journal articles (as the main author)

1. Nurkanović, A., Sperl, M., Albrecht, S., Diehl, M. (2022). Finite elements with switch detection for direct optimal control of nonsmooth systems. arXiv preprint - arXiv:2205.05337 (under review in Numerische Mathematik).

2. Nurkanović, A., Pozharskiy A., Frey, J., Diehl, M. (2023). Finite Elements with Switch Detection for Numerical Optimal Control of Nonsmooth Dynamical Systems with Set-Valued Heaviside Step Functions. arXiv preprint - arXiv:2307.03482 (under review in Nonlinear Analysis: Hybrid systems, special issue on Nonsmooth Dynamical Systems: Analysis, Control and Optimization).

3. Nurkanović, A., Pozharskiy, A., Diehl, M. (2023). Solving mathematical programs with complementarity constraints arising in nonsmooth optimal control. arXiv preprint - arXiv:2312.11022 (Submitted for a special issue in the Vietnam Journal of Mathematics dedicated to Tamás Terlaky's 70th birthday).

Submitted journal articles (as co-author)

4. Messerer F., Baumgärtner K., Nurkanović A., Diehl M. (2023). Approximate propagation of normal distributions for stochastic optimal control of nonsmooth systems. arXiv preprint arXiv:2308.03431 (under review to Nonlinear Analysis: Hybrid systems, special issue on Nonsmooth Dynamical Systems: Analysis, Control and Optimization).

Conference publications (as the main author)

1. Nurkanović, A., Zanelli, A., Albrecht, S., Diehl, M. (2019). The advanced step real time iteration for NMPC. In 2019 IEEE 58th Conference on Decision and Control (CDC) (pp. 5298-5305).

2. Nurkanović, A., Albrecht, S., Diehl, M. (2020). Limits of MPCC formulations in direct optimal control with nonsmooth differential equations. In 2020 European Control Conference (ECC) (pp. 2015-2020).

3. Nurkanović, A., Mešanović, A., Zanelli, A., Frison, G., Frey, J., Albrecht, S., Diehl, M. (2020). Real-time nonlinear model predictive control for microgrid operation. In 2020 American Control Conference (ACC) (pp. 4989-4995). IEEE.

4. Nurkanović, A., Zanelli, A., Albrecht, S., Frison, G., Diehl, M. (2020). Contraction properties of the advanced step real-time iteration for NMPC. IFAC World Congress, IFAC-PapersOnLine, 53(2), 7041-7048.

5. Nurkanović, A., Frey, J., Pozharskiy A., Diehl, M. (2023). Finite Elements with Switch Detection for Direct Optimal Control of Nonsmooth Systems with Set-Valued Step Functions. In 2023 IEEE 62nd Conference on Decision and Control (CDC).

Conference publications (as co-author)

6. Scholz, R., Nurkanović, A., Mešanović, A., Gutekunst, J., Potschka, A., Bock, H. G., Kostina, E. (2020). Model-based optimal feedback control for microgrids with multi-level iterations. In Operations Research Proceedings 2019: Selected Papers of the Annual International Conference of the German Operations Research Society (GOR), 2019 (pp. 73-79). Springer International Publishing.

7. Scholz, R., Nurkanović, A., Mešanović, A., Gutekunst, J., Potschka, A., Bock, H. G., Kostina, E. (2021). Multi-level iterations for microgrid control with automatic level choice. In Scientific Computing in Electrical Engineering: SCEE 2020 (pp. 293-301). Springer International Publishing.

8. Kiessling, D., Zanelli, A., Nurkanović, A., Gillis, J., Diehl, M., Zeilinger, M., Pipeleers, G., Swevers, J. (2022). A Feasible Sequential Linear Programming Algorithm with Application to Time-Optimal Path Planning Problems. In 2022 IEEE 61st Conference on Decision and Control (CDC) (pp. 1196-1203). IEEE.

9. Simpson, L., Nurkanović, A., Diehl, M. (2022). Direct Collocation for Numerical Optimal Control of Second-Order ODE. In 2023 European Control Conference (ECC).

10. Dietz C., Albrecht, S., Nurkanović, A., Diehl M. Efficient Collision Modeling for Numerical Optimal Control. In 2023 European Control Conference (ECC).

11. Van Roy, W., Nurkanović, A., Abbasi-Esfeden, R., Frey, J., Pozharskiy, A., Swevers, J. , Diehl, M. (2023). Continuous Optimization for Control of Finite-State Machines with Cascaded Hysteresis via Time-Freezing. In 2023 IEEE 62nd Conference on Decision and Control (CDC).

Book chapters

1. Nurkanović, A., Albrecht, S., Diehl, M. (2021). Multi-level iterations for economic nonlinear model predictive control. Recent Advances in Model Predictive Control: Theory, Algorithms, and Applications, 65-105.

Patents

1. Albrecht, S., Kast, B., Nurkanović, A., (2022). Method and control device for controlling a technical system. US Patent 11,378,924, 2022.

1.4 Notation

In this section, we review the notation used in this thesis. Sometimes, more specific notation will be introduced and highlighted in the chapters.

The sets $\mathbb{R}$ and $\mathbb{Z}$ are the set of real numbers and the set of integers, respectively. Entries of a vector x are scalar and denoted by x_i. The concatenation of two column vectors $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$ is denoted by $(x, y) \coloneqq [x^\top, y^\top]^\top$, the concatenation of several column vectors is defined analogously. For a given vector $x \in \mathbb{R}^n$ and set $\mathcal{I} \subseteq \{1, \dots, n\}$, we define the projection matrix $P_\mathcal{I} \in \mathbb{R}^{|\mathcal{I}| \times n}$, which has zeros or ones as entries. It selects all components $x_i, i \in \mathcal{I}$ from the vector x, i.e., $x_\mathcal{I} = P_\mathcal{I} x \in \mathbb{R}^{|\mathcal{I}|}$.

Entries of a matrix A are denoted by $a_{i,j}$. For some matrix $A \in \mathbb{R}^{n \times m}$, its i-th row is denoted by $A_{i,\bullet}$ and its j-th column is denoted by $A_{\bullet,j}$. Given a matrix $A \in \mathbb{R}^{n \times m}$ and as set $\mathcal{I} \subseteq \{1, \dots, n\}$, the matrix $A_{\mathcal{I},\bullet} \in \mathbb{R}^{|\mathcal{I}| \times m}$ is a submatrix of A consisting of the rows $A_{i,\bullet}$ for $i \in \mathcal{I}$. The submatrix $A_{\bullet,\mathcal{J}} \in \mathbb{R}^{n \times |\mathcal{J}|}$ is defined accordingly. The transpose of a matrix is denoted by $A^\top$. The inverse of an invertible square matrix is denoted by A^{-1}.

A column vector with all ones is denoted by $e = (1, 1, \dots, 1) \in \mathbb{R}^n$, and its dimension is clear from the context. The identity matrix is denoted by $I_n \in \mathbb{R}^{n \times n}$. Subscripts may be omitted when clear from the context. A matrix from $\mathbb{R}^{n \times m}$ with all zeroes is denoted by $\mathbf{0}_{n,m}$.

For a vector x, $\|x\|_p \coloneqq (\sum_{i=1}^n |x_i|^p)^{\frac{1}{p}}$ denotes the p-norm. If the index is omitted, then the Euclidean 2-norm is meant. Given a set $\mathcal{I}$, its cardinality is denoted by $|\mathcal{I}|$. The ϵ-ball centered at $x \in \mathbb{R}^n$ is the set $\mathcal{B}_\epsilon(x) = \{y \in \mathbb{R}^n \mid \|x - y\| \leq \epsilon\}$.

All vector inequalities are to be understood element-wise. The complementary conditions for two vectors $x, y \in \mathbb{R}^n$ read as $0 \leq x \perp y \geq 0$, where $x \perp y$ means $x^\top y = 0$. The function $\operatorname{diag}(x)$, $\operatorname{diag} : \mathbb{R}^n \to \mathbb{R}^{n \times n}$, returns a diagonal matrix with $x \in \mathbb{R}^n$ containing the diagonal entries.

We make use of asymptotic notation. We say that $f(h) = O(g(h))$ as $h \downarrow 0$, whenever there exists a constant $C > 0$ and $h_0 > 0$ such that $|f(h)| \leq Cg(h)$, provided that $0 < h < h_0$. Similarly, $f(h) = o(g(h))$ means that $\lim_{h \to 0} \frac{f(h)}{g(h)} = 0$.

For a function $f : \mathbb{R}^n \to \mathbb{R}^m$ we denote by $\mathrm{D}f(x) = \frac{\partial f}{\partial x}(x) \in \mathbb{R}^{m \times n}$ the Jacobian matrix and by $\nabla f(x) \coloneqq \frac{\partial f}{\partial x}(x)^\top$ its transpose.

Given two sets $\mathcal{A}, \mathcal{B}$, then the Minkowski set addition is defined as $\mathcal{A} + \mathcal{B} = \{c \mid c = a + b, a \in \mathcal{A}, b \in \mathcal{B}\}$. The closure of a set $\mathcal{C}$ is denoted by $\overline{\mathcal{C}}$, its boundary

as $\partial\mathcal{C}$, and $\text{conv}(\mathcal{C})$ is its convex hull.

We often use the same symbol for different functions, e.g., $f(\cdot)$ is the objective function in general optimization problems, the right-hand side (r.h.s.) of ordinary differential equations, the r.h.s. in the differential part of a semi-explicit ODE. However, which functions are meant by the used symbol is always clear from the context.

Following the American Mathematical Society Style Guide [176, p. 92], and the Society for Industrial and Applied Mathematics Style Manual [1, p. 40], we do not hyphenate mathematical words with the prefix "non" - e.g., *nonnegative, nonlinear, nonconvex, nonsmooth*.

Chapter 2

Nonlinear Optimization with Complementarity Constraints

The main goal of this thesis is to efficiently solve optimal control problems that involve nonsmooth dynamical systems. Combinatorial structures and nonsmoothness in such systems can often be expressed through complementarity constraints. Such constraints are ubiquitous in the time-discretizations of nonsmooth dynamical systems. Additionally, when a optimal control problem is discretized, it leads to a special class of degenerate nonlinear programs known as Mathematical Programs with Complementarity Constraints (MPCCs). Fortunately, they can often be solved efficiently by solving a sequence of standard smooth and more regular optimization problems.

Therefore, solving discretized nonsmooth optimal control problems requires a background in nonlinear optimization, complementarity problems, and their combination, i.e., MPCCs.

Outline. In this chapter, we study these three ingredients. Section 2.1 introduces some of the basic concepts and algorithms in constrained optimization with smooth functions. In Section 2.2 we discuss variational inequalities, complementarity problems and generalized equations. They are all somewhat nonsmooth generalizations of standard root-finding problems. Section 2.3 is devoted to reviewing in detail the theory of MPCCs. The chapter finishes with Section 2.4, which reviews standard solution methods for MPCCs. We focus on the algorithms that are implemented in `nosnoc`.

2.1 Smooth nonlinear optimization: theory and algorithms

In this section, we provide some basic background on finite-dimensional constrained optimization problems, which we call from now on nonlinear programming problems (NLP). We review the first and second-order optimality conditions and some of the most popular iterative solution methods. Ultimately, all methods in this thesis require solving NLPs at some point. NLP theory is also the basis for studying mathematical programs with complementarity constraints (MPCC), which are also a central topic in this thesis.

For this purpose, we regard an NLP in standard form:

$$\min_{x \in \mathbb{R}^n} \quad f(x) \tag{2.1a}$$

$$\text{s.t.} \quad g(x) = 0, \tag{2.1b}$$

$$h(x) \geq 0, \tag{2.1c}$$

with the vector $x \in \mathbb{R}^n$ as optimization (decision) variables and the objective (cost) function $f : \mathbb{R}^n \to \mathbb{R}$. The functions $g : \mathbb{R}^n \to \mathbb{R}^{n_g}$ and $h : \mathbb{R}^n \to \mathbb{R}^{n_h}$ are the equality and inequality constraint functions, respectively. We assume that these functions are at least twice continuously differentiable with respect to x.

2.1.1 Optimality conditions

We are interested in geometric and algebraic conditions that characterize an optimal solution to the problem (2.1). To be able to state these conditions we state some basic definitions.

Definition 2.1 (Feasible set)**.** *The feasible set of the NLP* (2.1) *is defined as the set* $\Omega = \{x \in \mathbb{R}^n \mid g(x) = 0,\ h(x) \geq 0\} \subseteq \mathbb{R}^n$*. A point* $x \in \Omega$ *is called a feasible point.*

Definition 2.2 (Local minimizer)**.** *A feasible point* $x^* \in \Omega$ *is called a local minimizer of the NLP* (2.1) *if there exists an open ball* $\mathcal{B}_\epsilon(x^*)$ *with* $\epsilon > 0$ *such that for all* $x \in \mathcal{B}_\epsilon(x^*) \cap \Omega$ *it holds that* $f(x) \geq f(x^*)$*. Furthermore, if* $f(x) > f(x^*)$ *for* $x \neq x^*$*, then* x^* *it is called a strict local minimizer. The value* $f(x^*)$ *at a local minimizer* x^* *is called the (strict) local minimum.*

Definition 2.3 (Global minimizer)**.** *A feasible point* $x^* \in \Omega$ *is called a global minimizer of the NLP* (2.1) *if for all* $x \in \Omega$ *it holds that* $f(x) \geq f(x^*)$*. The value* $f(x^*)$ *at a global minimizer* x^* *is called the global minimum.*

Alternatively, we call a minimizer x^* a local (global) solution or optimizer to the NLP (2.1). If the function g is affine and h convex, then the feasible set is a convex set. Moreover, if the objective f is a convex function, then the NLP is a convex optimization problem and every local minimizer is a global minimizer. An NLP is nonconvex if it is not convex. In this thesis, we deal with nonconvex problems and we are interested in computing a local minimizer.

First-order optimality conditions

We first provide some useful definitions and first-order optimality conditions based on the geometric characterization of the set Ω. The subsection finishes by providing a purely algebraic characterization of a local minimizer.

Definition 2.4 (Tangent Cone)**.** *For a closed set Ω and a point $x \in \Omega$, a vector d is said to be tangent to Ω at x if there exists a sequence $x_k \in \Omega$ with $x_k \to x$, along with a sequence of positive scalar $t_k \to 0$ such that $d_k = \frac{x_k - x}{t_k} \to d$. The set of all vectors d that are tangent to Ω at a point $x \in \Omega$ is called the tangent cone to Ω at x and denoted as $\mathcal{T}_\Omega(x)$.*

If the set $\Omega = \{x\}$ is a singleton, then $\mathcal{T}_\Omega(x) = \{0\}$. If x is in the interior of Ω, then $\mathcal{T}_\Omega(x) = \mathbb{R}^n$. At points x at the boundary of Ω it is a more complicated conic set and as we see below it is not necessarily convex. We now have the necessary tools to state a first variant of first-order necessary conditions.

Theorem 2.5 (First-order necessary conditions (FONC))**.** *If x^* is a local solution of the NLP* (2.1)*, then it holds that*

$$\nabla f(x^*)^\top d \geq 0, \text{ for all } d \in \mathcal{T}_\Omega(x^*). \tag{2.2}$$

A point x^* satisfying the condition (2.2) is called a *stationary point*. Next, we define some cones related to the tangent cone. As it will be useful, we define some additional cones that can be related to the tangent cone. We can express FONC conditions with their help as well.

Definition 2.6 (Normal cone)**.** *The normal cone to the set Ω at a point $x \in \Omega$ is defined as*

$$\mathcal{N}_\Omega(x) = \{v \in \mathbb{R}^n \mid v^\top d \leq 0 \text{ for all } d \in \mathcal{T}_\Omega(x)\}. \tag{2.3}$$

Each vector $v \in \mathcal{N}_\Omega(x)$ is said to be a normal vector. Furthermore, if $x \notin \Omega$, the normal cone is defined as $\mathcal{N}_\Omega(x) = \emptyset$.

Example tangent and normal cones at several points of a set Ω are depicted in Figure 2.1.

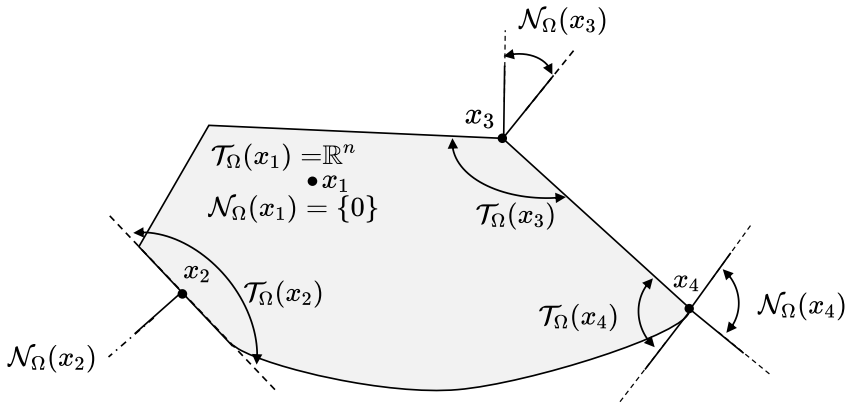


Figure 2.1: The normal and tangent cones at several points from a convex set Ω.

Definition 2.7 (Polar cone)**.** *The polar cone of a cone $C \subseteq \mathbb{R}^n$ is defined as $C^\circ = \{v \in \mathbb{R}^n \mid v^\top d \leq 0, \text{ for all } d \in C\}$.*

Definition 2.8 (Dual cone)**.** *The dual cone of a cone $C \subseteq \mathbb{R}^n$ is defined as $C^* = \{v \in \mathbb{R}^n \mid v^\top d \geq 0, \text{ for all } d \in C\}$.*

From the last two definitions, it follows that $C^* = -C^\circ$.

Observe that the definitions and theorems above are only based on the geometry of the feasible set Ω, but not its algebraic representation. In fact, for checking the necessary conditions we must verify infinitely many inequalities, cf. Theorem 2.5. To check some properties of the set Ω and to compute a solution to the NLP, it is more convenient to work with the algebraic characterization of the set Ω as it consists of a finite set of equations.

Due to continuity, it is clear that in a neighborhood of a point x, strictly satisfied inequalities $h_i(x) > 0$ will stay strictly satisfied, hence they do not influence the local algebraic characterization of the feasible set. In this sense the following definitions are useful.

Definition 2.9 (Active set)**.** *An inequality constraint $h_i : \mathbb{R}^n \to \mathbb{R}$ is called active at a feasible point $x \in \Omega$ if $h_i(x) = 0$. The index set $\mathcal{A}(x) = \{i \in \{1, \ldots, n_h\} \mid h_i(x) = 0\}$ of active inequality constraints is called the active set.*

Next, we look at a local algebraic specification of the feasible set at a point x based on the first-order derivatives of the constraint functions g and h. Moving

away from x in some of these directions we stay feasible up to the first order. The set of such directions is defined as follows.

Definition 2.10 (Linearized Feasible Cone)**.** *The linearized feasible cone of* Ω *at a feasible point* x *is defined as*

$$\mathcal{F}_\Omega(x) = \{d \in \mathbb{R}^n \mid \nabla g(x)^\top d = 0, \nabla h_{\mathcal{A}(x)}(x)^\top d \geq 0\}. \tag{2.4}$$

The set $\mathcal{F}_\Omega(x)$ is a closed convex polyhedral cone. It should somewhat replace the tangent cone $\mathcal{T}_\Omega(x)$ in the theorems above. However, these cones are not always the same, i.e., $\mathcal{T}_\Omega(x)$ might even be a nonconvex set. The so-called constraint qualifications (CQ) give sufficient (and sometimes necessary) conditions when these two cones are equal or how are they are related, if at all. In other words, they specify conditions under which the algebraically defined set $\mathcal{F}_\Omega(x)$ captures the local geometry of Ω. Some of the most widely used CQs are listed in the next definition.

Definition 2.11 (Constraint Qualifications)**.** *Let* $x \in \Omega$ *be a feasible point to the NLP* (2.1)*.*

- *The Linear Independence CQ (LICQ) holds at* x *if the vectors* $\nabla g_i(x), i = 1, \ldots, n_h$ *and* $\nabla h_i(x),\ i \in \mathcal{A}(x)$ *are linearly independent.*
- *The Mangasarian-Fromovitz (MFCQ) holds at* x *if the vectors* $\nabla g_i(x), i = 1, \ldots, n_h$ *are linearly independent and there exist a vector* $d \in \mathbb{R}^n$ *such that* $\nabla g(x)^\top d = 0$ *and* $\nabla h_i(x)^\top d > 0$ *for all* $i \in \mathcal{A}(x)$*.*
- *The Abadie CQ (ACQ) holds at* x *if* $\mathcal{T}_\Omega(x) = \mathcal{F}_\Omega(x)$*.*
- *The Guignard CQ (GCQ) holds at* x *if* $\mathcal{T}_\Omega(x)^\circ = \mathcal{F}_\Omega(x)^\circ$*.*

The LICQ implies the MFCQ, and the MFCQ implies the ACQ. The ACQ implies the GCQ. The converses are in general not true. The LICQ and MFCQ are sufficient for the equality of the linearized feasible cone and tangent cone, but they are more practical than the ACQ and the GCQ since they can be verified by using the algebraic specification of the feasible set Ω. We illustrate the different CQs in a simple example.

Example 2.12. *(Constraint Qualifications) Regard the NLP*

$$\begin{aligned} \min_{x \in \mathbb{R}^2} \quad & x_1^2 + x_2^2 \\ \text{s.t.} \quad & x_1 \geq 0, \\ & x_2 \geq 0, \\ & x_1 x_2 \leq 0. \end{aligned}$$

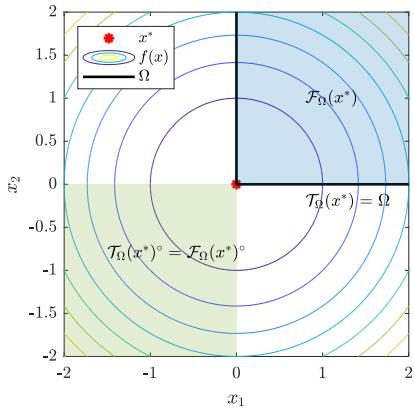


Figure 2.2: Illustration of the feasible set, objective function contour lines, linearized feasible cone, tangent cone, and their polar cones for the NLP of Example 2.12

The global minimizer of this NLP is $x^* = (0, 0)$. *The cones* $\mathcal{F}_\Omega(x^*)$ *and* $\mathcal{T}_\Omega(x^*)$ *are illustrated in Figure 2.2. It can be verified that* $\mathcal{T}_\Omega(x^*) = \Omega$ *and that* $\mathcal{F}_\Omega(x^*) = \{d \in \mathbb{R}^2 \mid d \geq 0\}$. *Clearly,* $\mathcal{T}_\Omega(x^*)$ *is nonconvex and we have only the GCQ to hold at* x^*. *This example belongs to the class of mathematical programs with complementarity constraints and it is characteristic for these problems that only the GCQ has a chance to hold, cf. Section 2.3.*

A key function in the theory of constrained optimization is the Lagrange function.

Definition 2.13 (Lagrange Function/Lagrangian)**.** *The function* $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^{n_g} \times \mathbb{R}^{n_h} \to \mathbb{R}$*:*

$$\mathcal{L}(x, \lambda, \mu) \coloneqq f(x) - \lambda^\top g(x) - \mu^\top h(x), \tag{2.6}$$

with Lagrange multipliers $\lambda \in \mathbb{R}^{n_g}$ *and* $\mu \in \mathbb{R}^{n_h}$ *is called the Lagrangian of the NLP* (2.1).

The next theorem is probably the most famous result in constrained optimization theory and was independently discovered by Karush [160] and Kuhn and Tucker [172]. It provides conditions based on the first-order derivatives of the NLP functions that characterize a local minimizer. Under appropriate CQs it holds that $\mathcal{T}_\Omega(x) = \mathcal{F}_\Omega(x)$ and from Theorem 2.5 for a local minimizer x^* it holds that $-\nabla f(x^*) \in \mathcal{F}_\Omega(x)^\circ$. Using now Farka's lemma, one can derive the KKT conditions. For a proof and further details see e.g., [201, Chapter 12].

Theorem 2.14 (Karush-Kuhn-Tucker (KKT) conditions)**.** *Let x^* be a local minimizer of the NLP* (2.1) *and assume that LICQ holds at x^*. Then there exist Lagrange multipliers λ^* and μ^* such that the following conditions are satisfied:*

$$\nabla_x \mathcal{L}(x^*, \lambda^*, \mu^*) = \nabla f(x^*) - \nabla g(x^*)^\top \lambda^* - \nabla h(x^*)^\top \mu^* = 0, \tag{2.7a}$$

$$g(x^*) = 0, \tag{2.7b}$$

$$h(x^*) \geq 0, \tag{2.7c}$$

$$\mu^* \geq 0, \tag{2.7d}$$

$$\mu_i^* h_i(x^*) = 0, \ i = 1, \ldots, n_h. \tag{2.7e}$$

A vector (x^*, λ^*, μ^*) is called a KKT point if LICQ and Eq. (2.7) *hold.*

The set of equations (2.7) is called the KKT conditions. The last condition $\mu_i^* h_i(x^*) = 0$ states that both terms cannot be nonzero at the same time, which implies that inactive constraints must have zero multipliers. If there is no i such that both terms are zero, then we say that *strict complementarity holds.* A constraint $h_i(x^*)$ for which it holds that $\mu_i^* = h_i(x^*) = 0$ is said to be *weakly active.*

Second-order optimality conditions If the NLP (2.1) is convex and LICQ holds at x^*, then the KKT conditions are necessary and sufficient for x^* to be a global minimizer. However, for nonconvex problems they are only necessary and we usually have feasible directions $d \in \mathcal{F}_\Omega(x^*)$ of zero slope, i.e., $\nabla f(x^*)^\top d = 0$. To make further conclusions, we must look at second-order derivatives of the NLP functions at x^*.

Definition 2.15 (Critical cone)**.** *Let (x^*, λ^*) be a KKT point. The set is defined as*

$$\begin{aligned} \mathcal{C}_\Omega(x^*, \mu^*) = \{ d \in \mathbb{R}^n \mid & \nabla g_i(x)^\top d = 0, \ \textit{for all } i \in \{1, \ldots, n_g\}, \\ & \nabla h_i(x)^\top d = 0, \ \textit{for all } i \in \mathcal{A}(x^*) \ \textit{with } \mu_i^* > 0, \\ & \nabla h_i(x)^\top d \geq 0, \ \textit{for all } i \in \mathcal{A}(x^*) \ \textit{with } \mu_i^* = 0 \} \end{aligned} \tag{2.8}$$

is called the critical cone.

Note that it holds in general that $\mathcal{C}_\Omega(x^*, \mu^*) \subseteq \mathcal{F}_\Omega(x^*)$ since the *increasing* directions $\{d \in \mathcal{F}_\Omega(x^*) \mid h_i(x)^\top d > 0, \text{ for all } i \in \mathcal{A}(x^*) \text{ with } \mu_i^* > 0\}$ are excluded in the critical cone. If there were no weakly active constraints, then

the critical cone would be a subspace of $\mathbb{R}^n$. This simplifies the verification of second-order conditions. Moreover, if the matrix formed by the constraint normal defining the critical cone has rank n, then the critical cone is the zero set. The next theorem gives second-order optimality conditions.

Theorem 2.16 (Second-order conditions)**.** *Let $x^* \in \Omega$ be a feasible point of the NLP* (2.1)*.*

- *(Second-Order Necessary Conditions (SONC)) Suppose that x^* is a local minimizer of the NLP* (2.1) *and that the LICQ condition holds at x^*. Then there exist Lagrange multipliers (λ^*, μ^*) such that the KKT conditions* (2.7) *are satisfied, and it holds that*

$$d^\top \nabla^2_{xx} L(x^*, \lambda^*, \mu^*) d \geq 0, \text{ for all } d \in \mathcal{C}_\Omega(x^*, \mu^*). \tag{2.9}$$

- *(Second-Order Sufficient Conditions (SOSC)) Suppose that for x^* there exist Lagrange multipliers (λ^*, μ^*) such that the KKT conditions* (2.7) *are satisfied and that the following inequality holds*

$$d^\top \nabla^2_{xx} L(x^*, \lambda^*, \mu^*) d > 0, \text{ for all } d \in \mathcal{C}_\Omega(x^*, \mu^*) \setminus \{0\}. \tag{2.10}$$

  Then x^* is a strict local minimizer of the NLP (2.1)*.*

A short and elegant proof for this theorem can be found in [100, Chapter 9.1]. A longer but more instructive proof is available in [232, Section 8.12].

It might be tempting to assume that we have to look only at the curvature of the objective $\nabla^2 f(x)$ in the critical directions. However, this consideration does not capture the curvature of the constraints, and we have to indeed look at the Hessian of the Lagrange function as we illustrate in the next example.

Example 2.17 (Fiacco-McCormic)**.** *Regard the NLP parametrized by the scalar β:*

$$\begin{aligned} \min_{x \in \mathbb{R}^2} \quad & \frac{1}{2}((x_1 - 1)^2 + x_2^2) \\ \text{s.t.} \quad & -x_1 + \beta x_2^2 = 0. \end{aligned}$$

Since we can explicitly express $x_1 = \beta x_2^2$, our NLP has the same solutions as the unconstrained reduced problem $\min_{x_2} \tilde{f}(x_2)$ with $\tilde{f}(x_2) = (\beta x_2^2 - 1)^2 + x_2^2$. Let us regard the point $x^* = 0$ for $\beta = 0.25$ (for which x^* is a local minimizer) and $\beta = 2$ (for which x^* is not a local minimizer), cf. the left plot in Figure 2.3. Since $x^* = 0$ is feasible and $\nabla f(x^*) = \nabla g(x^*) = (-1, 0)$ the KKT conditions are satisfied for $\lambda^* = 1$. The critical cone is the set $C_\Omega(x^*, \lambda^*) = \{d \in \mathbb{R}^2 \mid$

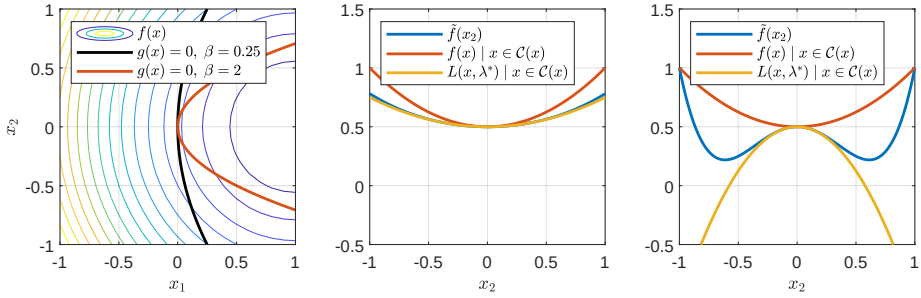


Figure 2.3: The feasible set and objective function contour lines (left plot), the functions $f(x)$, $L(x, \lambda^*)$ and $\tilde{f}(x_2)$ for $\beta = 0.25$ (middle plot) and the functions $f(x)$, $L(x, \lambda^*)$ and $\tilde{f}(x_2)$ for $\beta = 2$ (right plot).

$d_1 = 0\}$. *The Hessian of the objective function is* $\nabla^2_{xx} f(x) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ *is always positive definite, but obviously, we can draw no conclusions about optimality using this information. On the other hand, the Hessian of the Lagrangian is* $\nabla^2_{xx} L(x^*, \lambda^*) = \begin{bmatrix} 1 & 0 \\ 0 & 1 - 2\beta \end{bmatrix}$, *i.e.* $d^\top \nabla^2_{xx} L(x^*, \lambda^*) d = d_2^2(1 - 2\beta)$. *For* $\beta < 0.5$, *the SOSC are satisfied and for* $\beta > 0.5$, *the SONC are violated. The second-order derivative of the reduced objective at* x^* *is* $\nabla^2_{xx} f_2(x_2^*) = (1 - 2\beta)$ *and has the same sign as* $d^\top \nabla^2_{xx} L(x^*, \lambda^*) d$ *in the critical directions. The functions* $f(x)$, $L(x, \lambda^*)$ *and* $\tilde{f}(x_2)$ *are illustrated for different points along the critical directions in Figure 2.3.*

2.1.2 Nonlinear programming algorithms

Regard the root-finding problem $F(w) = 0$, where $F : \mathbb{R}^{n_w} \to \mathbb{R}^{n_w}$. Given some initial guess w^0, Newton's method generates a sequence of solution approximations w^k by solving a linearized version of the initial problem:

$$F(w^k) + \nabla F(w^k)^\top (w^{k+1} - w^k) = 0.$$

If the Jacobian $\nabla F(w)^\top$ is invertible and if w^0 is sufficiently close to a solution w^*, then the generated sequence of solutions approximations w^k converges to w^*. In Newton-type methods, instead of the exact Jacobian, an approximation $A^k \approx \nabla F(w^k)^\top$ is used. To promote global convergence, i.e., to ensure convergence from any initial point, a globalization strategy must be used. That is, the step $\delta w^k = (w^{k+1} - w^k)$ is adapted in some way such that convergence is ensured.

Most NLP solvers try to find a solution candidate (x^*, λ^*, μ^*) by solving the KKT conditions (2.7). Starting from an initial guess (x^0, λ^0, μ^0), they generate a sequence $(x^k, \lambda^k, \mu^k) \to (x^*, \lambda^*, \mu^*)$ as $k \to \infty$. If no inequalities are present, the KKT system reduces to a standard root-finding problem. In this thesis, we focus on Newton-type algorithms for iteratively finding a solution to the KKT system (2.7). However, things get more complicated with the presence of inequality constraints. One of the main distinguishing features of NLP algorithms is how they treat these constraints. The two most widely used Newton-type NLP solution methods are Sequential Quadratic Programming (SQP) and Interior-Point (IP) methods. Several great textbooks cover the broad topic of Newton-type NLP solution methods [38, 100, 148, 201].

Sequential quadratic programming

The main idea in SQP is, given the current iterate (x^k, λ^k, μ^k), to construct the following Quadratic Program (QP), which approximates the NLP (2.1):

$$\min_{d \in \mathbb{R}^n} \quad \frac{1}{2} d^\top A^k d + \nabla f(x^k)^\top d \tag{2.12a}$$

$$\text{s.t.} \quad g(x^k) + \nabla g(x^k)^\top d = 0, \tag{2.12b}$$

$$h(x^k) + \nabla h(x^k)^\top d \geq 0. \tag{2.12c}$$

The solution of this QP provides a new search direction $d^k \in \mathbb{R}^n$. The Lagrange multipliers of the QP equality and inequality constraints are denoted by λ_{QP} and μ_{QP}, respectively. The QP is obtained from the first-order Taylor approximations of the equality and inequality constraints and a quadratic approximation for the objective function, where A^k is the Hessian of the Lagrangian or an approximation of it. The KKT conditions of this QP correspond to a linearization of the KKT conditions of the NLP, but where the complementarity conditions are *structurally linearized* such that the two terms entering the bilinear expression (2.7) are linearized.

The next step is computed as

$$x^{k+1} = x^k + d^k, \ \lambda^{k+1} = \lambda^k_{\text{QP}}, \ \mu^{k+1} = \mu^k_{\text{QP}}. \tag{2.13}$$

Under suitable conditions, the generated sequence converges to a local minimizer of the NLP (2.1).

To implement an SQP method, one requires a QP solver at hand. An overview of QP solvers tailored to optimal control problems is given in [170]. In practice, there are many variations to the basic SQP algorithm. Globalization strategies

adapt the search direction d^k to promote convergence from any initial guess. In line search methods, the step d^k is replaced by $\alpha d^k,\ \alpha \in (0, 1]$ such that a sufficient decrease in the objective and/or constraint infeasibility is achieved. Trust region methods add constraints to bound d in the QP (2.12), such that both the direction and length of d^k are adapted simultaneously. SQP methods are known for good warm starting properties, that is, given a good initial guess, the solution is expected to be found in a few iterations.

In practice, instead of the exact Hessian $A^k = \nabla^2_{xx}\mathcal{L}(x^*, \lambda^*, \mu^*)$, a suitable approximation is often used. Some approximations as the BFGS and SR1 update formulae rely on previous iterates, whereas the Gauss-Newton approximation (suitable only for nonlinear least squares objective functions) depends only on the current iterate. There exist several variants of inexact SQP methods where even the first-order derivatives in the QP (2.12b) are approximated. In general, the approximations slow down the convergence rate but lead to cheaper iterations. Widely used and robust SQP implementations are `SNOPT` [111] and `filterSQP` [101], and in the field of optimal control `MUSCOD-II` [81], `ACADO Toolkit` [140, 226], and `acados` [278].

Interior-point methods

Similar to SQP methods, IP methods also solve a sequence of approximate problems. The main obstacle to directly applying Newton's method to the KKT system (2.7) are the nonsmooth complementarity conditions (2.7e). A possible remedy is to solve a sequence of smoothed KKT systems:

$$\nabla_x \mathcal{L}(x, \lambda, \mu) = 0, \tag{2.14a}$$

$$g(x) = 0, \tag{2.14b}$$

$$\mu_i h_i(x) = \tau^{[k]},\ i = 1, \ldots, n_h, \tag{2.14c}$$

where $\tau^{[k]} \to 0$. Note that here k is the index of the subproblem in the sequence of smoothed systems, but not the index of the Newton iteration. For every fixed smoothing parameter $\tau^{[k]}$, one finds an (approximate) solution to (2.14) with a Newton-type method and updates afterward the $\tau^{[k]}$. The Newton steps are adapted with a *fraction-to-boundary* line search such that $\mu, h(x) > 0$ is maintained. The smoothing parameter updated strategy (which does not have to be monotonic), the used line-search and type of Hessian approximation $A \approx \nabla^2_{xx}\mathcal{L}(x, \lambda, \mu)$ are the key to the practical success of interior-point methods.

Eq. (2.14) are the KKT conditions of the following equality-constrained problem:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) - \tau^{[k]} \sum_{i=1}^{n_h} \log(h(x)) \\ \text{s.t.} \quad & g(x) = 0. \end{aligned}$$

A solution to this problem lies in the interior of the feasible set Ω, hence the name. Some widely used interior-point method implementations are `LOQO` [276], `KNITRO` [58], and `IPOPT` [281].

2.2 Variational inequalities and complementarity problems

Variational inequalities (VI) are a generalization of root-finding problems. They are a powerful modeling tool that provides an efficient way to write down many problems in science and engineering that have some combinatorial and nonsmooth structure. For example, we have already used VIs to express necessary optimality conditions for constrained optimization problems, cf. Theorem 2.5. Moreover, they provide a convenient way to model constrained systems of equations, Nash equilibrium problems, frictional contact problems in mechanics (cf. Chapter 8), traffic equilibrium models, economic equilibrium problems, and many more [89]. Solution sets of parametric VIs are under appropriate conditions piecewise smooth functions of the parameter. We exploit this on several occasions in this thesis for modeling of piecewise smooth dynamical systems, in particular in Chapter 6. Under mild assumptions, VIs can be brought into several equivalent forms, namely complementarity problems, generalized equations, and nonsmooth equations. This facilitates the theoretical analysis and development of computational methods. In this section we define the aforementioned concepts, relate them and list some basic theoretical results. All results given below can be found in [89].

2.2.1 Variational inequalities and generalized equations

We follow the exposition of [270] to motivate the definition of VIs, a generalization of root-finding problems. Regard the problem of finding $x \in \mathbb{R}^n$ with $F : \mathbb{R}^n \to \mathbb{R}^n$ such that:

$$F(x) = 0.$$

This is equivalent to the *variational problem* of finding $x \in \mathbb{R}^n$ such that

$$F(x)^\top d = 0, \quad \text{for all } d \in \mathbb{R}^n,$$

which is further equivalent to finding $x \in \mathbb{R}^n$ such that

$$F(x)^\top (y - x) = 0, \quad \text{for all } y \in \mathbb{R}^n,$$

A general variational inequality has the same form, but now with restricting x, y to a set $K \subseteq \mathbb{R}^n$.

Definition 2.18. *Let $K \subseteq \mathbb{R}^n$ and $F : \mathbb{R}^n \to \mathbb{R}^n$. A variational inequality, denoted compactly by $\mathrm{VI}(K, F)$, is the problem of finding $x \in \mathbb{R}^n$ such that*

$$x \in K, \; F(x)^\top (y - x) \geq 0, \quad \textit{for all } y \in K. \tag{2.15}$$

The set of solutions to this problem is denoted by $\mathrm{SOL}(K, F)$.

In most of what follows, we assume that K is a closed convex set and that $F(x)$ is continuous on K. This is sufficient for our needs and still allows great generality. For results on variants where these conditions are relaxed, cf. [89].

Geometrically, a vector $x \in K$ is a solution of $\mathrm{VI}(K, F)$ if and only if either $F(x) = 0$ or $F(x)$ forms a nonobtuse angle with every vector $y - x$ for all $y \in K$. This also reveals that feasible solutions of $F(x) = 0$ are solutions to the VI, i.e., $F^{-1}(0) \cap K \subseteq \mathrm{SOL}(K, F)$. However, the cases where it is interesting to analyze VIs are when the zeros of $F(x) = 0$ (if any exist at all) are not in K, i.e., $F^{-1}(0) \cap K = \emptyset$.

Comparing Definition 2.18 with the definition of the normal cone in Definition 2.6, i.e., $\mathcal{N}_K(x) = \{v \in \mathbb{R}^n \mid v^\top (y - x) \leq 0, \text{ for all } y \in K\}$, it can be seen that $\mathrm{VI}(K, F)$ is equivalent to finding $x \in K$ such that:

$$0 \in F(x) + \mathcal{N}_K(x). \tag{2.16}$$

This problem is called a generalized equation and was introduced by Robinson in [231]. Of course, in more general forms, the set-valued normal cone map can be replaced by some more general set-valued map. For a detailed treatment of generalized equations, cf. [84]. Conceptually, the problem of satisfying infinitely many inequality conditions is now replaced by finding the zeros of a set-valued equation. A geometrical interpretation of the VI (2.15) and the generalized equation (2.16) is given in Figure 2.4.

We restate a fundamental existence and uniqueness result for VIs.

Theorem 2.19 (Corollaries 2.2.5, 2.2.6 and Theorem 2.2.3 in [89])**.** *Let $K \subseteq \mathbb{R}^n$ be a closed convex set and suppose that $F : K \to \mathbb{R}^n$ is a continuous function. Then the following statements hold.*

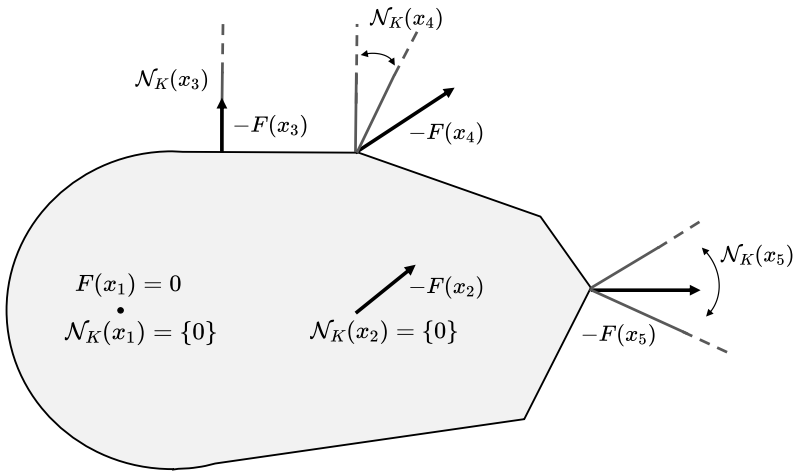


Figure 2.4: Geometric illustration of the VI(K, F). The vector x_1 is a solution since $F_1(x) = 0$, x_2 is not a solution. Vectors x_3 and x_5 are solutions since $-F(x) \in \mathcal{N}_K(x)$, thus the GE (2.16) is satisfied. The VI is not satisfied at x_4.

a) If there exist $x_{\text{ref}} \in K$ *such that* $F(x)^\top(x - x_{\text{ref}}) \geq 0$ *for all* $x \in K$*, then the set* $\text{SOL}(K, F)$ *is nonempty.*

b) If K *is bounded, then the set* $\text{SOL}(K, F)$ *is nonempty and compact.*

c) If $F(\cdot)$ *is strictly monotone on* K*, that is* $(F(x_1) - F(x_2))^\top(x_1 - x_2) > 0$ *for all* $x_1, x_2 \in K$ *and* $x_1 \neq x_2$*, then* $\text{SOL}(K, F)$ *has at least one element.*

d) If $F(\cdot)$ *is* $\xi-$*monotone on* K *with* $\xi > 1$*, i.e., there exist a* $c > 0$ *such that* $(F(x_1) - F(x_2))^\top(x_1 - x_2) > c\|x_1 - x_2\|^\xi$ *for all* $x_1, x_2 \in K$ *and* $x_1 \neq x_2$*, then* $\text{SOL}(K, F)$ *is a singleton.*

Note that only the second assertion needs a compact K. The other assumptions are applicable for unbounded sets, e.g., when K is a cone. Monotonicity is a ubiquitous assumption in the study of VIs. For example, gradients of convex functions are monotone.

2.2.2 Complementarity problems

An ample number of practical problems are VIs with K being a closed convex cone. In this case, the VI is equivalent to a *cone complementarity problem*. Recall that $K^* = \{v \in \mathbb{R}^n \mid d^\top v \geq 0 \text{ for all } d \in K\}$ is the dual cone of K, cf. Definition 2.8.

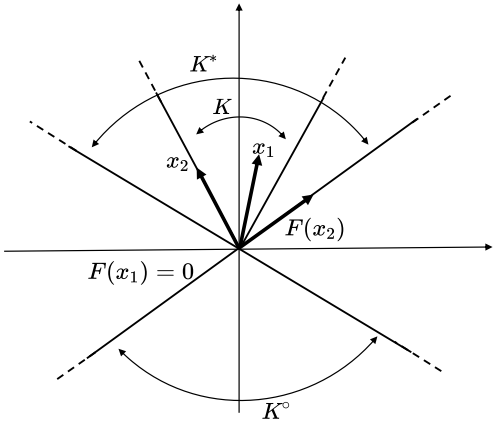


Figure 2.5: Geometric illustration of the cone complementarity problem from Definition 2.20 with two solution examples.

Definition 2.20. *Given a closed convex cone K and a mapping $F : K \to \mathbb{R}^n$, the cone complementarity problem is to find $x \in K$ such that*

$$K \ni x \perp F(x) \in K^*, \tag{2.17}$$

where this compact notation means that $x \in K, F(x) \in K^*$ and $F(x)^\top x = 0$.

Figure 2.5 provides a geometrical interpretation of this problem. The VI and the cone complementarity problem (2.17) are equivalent as shown in the next result.

Proposition 2.21 (Proposition 1.1.3. in [89])**.** *Let K be a closed convex cone. A vector $x \in \mathbb{R}^n$ is a solution to* VI(K, F) *in* (2.15) *if and only if it is a solution to the cone complementarity problem* (2.17)*.*

Proof. Let x be a solution to the VI(K, F). On the one hand, since K is a cone, setting $y = 0 \in K$ we have from (2.15) that $F(x)^\top x \leq 0$. On the other hand, from the definition of a cone $x \in K$ it follows that $2x \in K$. Again, from (2.15) and setting $y = 2x$ we obtain that $F(x)^\top x \geq 0$. Therefore, $F(x)^\top x = 0$. We further exploit that $F(x)^\top x \geq 0$, i.e., we can see that $F(x)^\top (y - x) \geq 0$ implies that $F(x)^\top y \geq 0$ for all $y \in K$, which is equivalent to $F(x) \in K^*$. Thus we have proven that x solves also (2.17). Conversely, if x solves (2.17), we have from the definition that $F(x)^\top y \geq 0$ for all $y \in K$ and $F(x)^\top x = 0$. Subtracting these relations we obtain that the VI (2.15) holds. □

We mention some important special cases related to (2.17). The problem with $K = \mathbb{R}^n_{\geq 0}$ is called a Nonlinear Complementarity Problem (NCP). This cone is self-dual, i.e., $\mathbb{R}^n_{\geq 0} = (\mathbb{R}^n_{\geq 0})^*$.

Definition 2.22. *Given a function $F : \mathbb{R}^n_{\geq 0} \to \mathbb{R}^n$, a nonlinear complementarity problem is to find a vector $x \in \mathbb{R}^n$ such that*

$$0 \leq x \perp F(x) \geq 0. \tag{2.18}$$

The nonnegativity of x and $F(x)$ allows us to write the orthogonality condition component-wise, i.e., the NCP is equivalent to solving:

$$x \geq 0,\ F(x) \geq 0,$$

$$F_i(x)x_i = 0, \quad \text{for all } i \in \{1, \ldots, n\}.$$

In the form of inclusion into normal cones, i.e., GEs, this problem reads as $-F(x) \in \mathcal{N}_{\mathbb{R}^n_{\geq 0}}(x)$ and $-x \in \mathcal{N}_{\mathbb{R}^n_{\geq 0}}(F(x))$.

The NCP is generalized by adding further equality constraints. This can be interpreted as the cone complementarity problem with $K = \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}_{\geq 0}$ with $n = n_1 + n_2$. Let $G : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}^{n_1}$, $H : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}^{n_2}$, $y \in \mathbb{R}^{n_1}, z \in \mathbb{R}^{n_2}$ and $x = (y, z) \in \mathbb{R}^n$. The mixed complementarity problem is finding x such that

$$0 = G(y, z), \tag{2.19}$$

$$0 \leq z \perp H(y, z) \geq 0. \tag{2.20}$$

The KKT system in Eq. (2.7) is one of the most famous examples of mixed complementarity problems.

A widely studied subclass of NCPs is when $F(x) = Mx + q$ is an affine function. This gives rise to the Linear Complementarity Problem (LCP):

$$0 \leq x \perp Mx + q \geq 0. \tag{2.21}$$

The solution set of this LCP has the widely used notation $\mathrm{SOL}(M, q)$. They play also a fundamental role in solution methods and the theoretical analysis for VIs and complementarity problems, as they arise from their *structural linearization*. The monograph [69] collects the most important theoretical and algorithmic results for LCPs.

2.2.3 Nonsmooth equations

Variational inequalities can be written as equivalent generalized equations. Now we show that VIs can be written also as *standard* but nonsmooth systems of

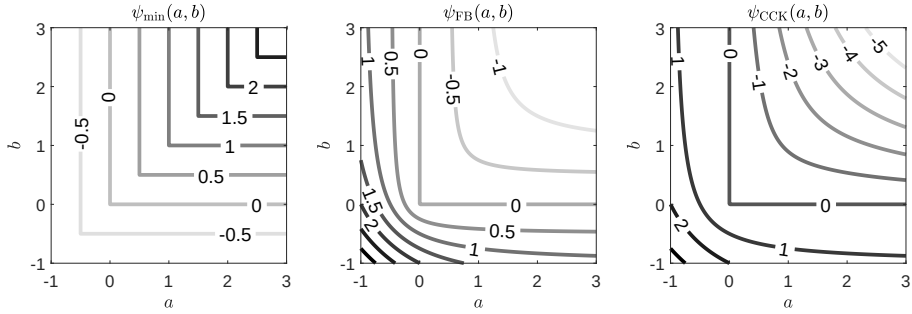


Figure 2.6: Level curve of C-functions. The left plot shows the min function, the middle plot the Fischer-Burmeister function, and the right plot the Chen-Chen-Kanzow function with $\lambda = 0.5$.

equations. We start with the simpler step of writing NCP (2.18) as nonsmooth equation systems. For this purpose, we define the so-called C-functions.

Definition 2.23 (C-Functions)**.** *A function $\psi_{\mathrm{C}} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is called a C-function if for any pair $(a, b) \in \mathbb{R}^2$ the following is satisfied*

$$\psi_{\mathrm{C}}(a, b) = 0 \iff a, b \geq 0 \text{ and } ab = 0.$$

Given two vectors $x, y \in \mathbb{R}^n$, the vector-valued C-function $\Psi_{\mathrm{C}} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ is defined component-wise, i.e., $\Psi_{\mathrm{C}}(x, y) := (\psi_{\mathrm{C}}(x_1, y_1), \ldots, \psi_{\mathrm{C}}(x_n, y_n))$.

In the literature these functions are sometimes called NCP functions [178]. Therefore, the NCP (2.18) is equivalent to finding zeros in the following nonsmooth system of equations:

$$\Psi_{\mathrm{C}}(F(x), x) = 0.$$

Let us list some of the most popular C-functions. For concrete examples, in our notation, we replace the C in the subscript with some acronym related to the name of the C-function. The first example is the min-function, i.e., it holds that $\psi_{\min}(a, b) = \min(a, b) = 0$. This function is differentiable everywhere except at the line $a = b$. An equivalent formulation is given via the natural residual function $\psi_{\min}(a, b) = \psi_{\mathrm{NR}}(a, b) = \frac{1}{2}(a + b - \sqrt{(a - b)^2})$. The Fischer-Burmeister (FB) function is given by $\psi_{\mathrm{FB}}(a, b) = \sqrt{(a^2 + b^2)} - (a + b)$. This function is convex and differentiable everywhere except at $(0, 0)$. The function ψ^p_{FB}, with $p \geq 2$ is differentiable everywhere, however, has a zero gradient at

the origin. Another example is the Chen-Chen-Kanzow function $\psi_{\mathrm{CCK}}(a,b) = \psi_{\mathrm{FB}}(a,b) - \lambda \max(0,a)\max(0,b)$ with $\lambda > 0$. The level curves of our three examples are depicted in Figure 2.6.

2.3 Mathematical programs with complementarity constraints

In this section, we turn our attention to Mathematical Programs with Complementarity Constraints (MPCC). They are a central tool in this thesis as all discrete-time optimal control problem formulations that we arrive at are MPCCs.

A large source of MPCCs are so-called bilevel optimization problems. These are NLPs where the solution set of a *lower-level* optimization problem is part of the *upper-level* feasible set. The upper-level variables enter the lower-level problem as parameters. Replacing the lower-level problem by its KKT conditions one ends up with an MPCC.

A more general class are Mathematical Programs with Equilibrium Constraints (MPEC) which are NLPs with a VI, representing an equilibrium condition, as a constraint. Solutions of VI are characterized by complementarity conditions. However, they might be only necessary conditions as in the case of KKT conditions for nonconvex NLPs. Thus by replacing the VI with complementarity conditions we obtain an MPCC which is usually a relaxation of the initial MPEC. Of course, when the complementarity conditions are also sufficient, the two problems become equivalent. We review the standard MPCC theory and solution methods. In the literature, the acronym MPEC is often used for MPCCs, since it is easier to pronounce.

2.3.1 Introduction to MPCCs

The complementarity constraint in an NLP formulation of the MPCC (as Eq. (2.23)) lead to a violation of standard CQs at all feasible points. This requires special treatment both from the numerical and theoretical points of view. The theory presented here has its origins in [96, 104, 151, 187, 236].

We regard an NLP similar to (2.1), but now with additional complementarity conditions:

$$0 \leq G(x) \perp H(x) \geq 0,$$

where $G, H : \mathbb{R}^n \to \mathbb{R}^m$ are assumed to be twice continuously differentiable functions. These conditions can be equivalently stated in several ways:

1. $G(x) \geq 0,\ H(x) \geq 0,\ G_i(x)H_i(x) \leq 0,\ i = 1, \dots m,$
2. $G(x) \geq 0,\ H(x) \geq 0,\ G(x)^\top H(x) \leq 0,$
3. $G(x) \geq 0,\ H(x) \geq 0,\ G_i(x)H_i(x) = 0,\ i = 1, \dots m,$
4. $G(x) \geq 0,\ H(x) \geq 0,\ G(x)^\top H(x) = 0,$
5. $\Phi_{\mathrm{C}}(G(x), H(x)) = 0.$

In our exposition, we pick to work with the first variant.

Remark 2.24. *Several authors [104, 179, 248] introduce slack variables for the functions $G(x)$ and $H(x)$ to have only linear functions in the complementarity conditions:*

$$s_G = G(x),\ s_H = H(x), \tag{2.22a}$$

$$s_G \geq 0,\ s_H \geq 0, \tag{2.22b}$$

$$s_{G,i} s_{H,i} \leq 0,\ \textit{for all } i \in \{1, \dots, m\}. \tag{2.22c}$$

This does not change any of the theoretical considerations and we stick to the notation of most of the MPCC-literature [12, 96, 108, 139, 146, 151, 219, 228, 236], where nonlinear functions $G(x)$ and $H(x)$ are treated directly in the complementarity conditions. However, for the efficacy of numerical solvers it is often beneficial to introduce the slacks [104], see also Section 2.4.

Regard the following NLP

$$\min_{x \in \mathbb{R}^n} \quad f(x) \tag{2.23a}$$

$$\text{s.t.} \quad g(x) = 0, \tag{2.23b}$$

$$h(x) \geq 0, \tag{2.23c}$$

$$G(x) \geq 0, \tag{2.23d}$$

$$H(x) \geq 0, \tag{2.23e}$$

$$G_i(x)H_i(x) \leq 0,\ i = 1, \dots, m. \tag{2.23f}$$

The feasible set of this NLP is denoted by Ω_{MPCC}. The constraints (2.23d)-(2.23f) introduce some difficulty that prohibits a straightforward application of the NLP theory from Section 2.1. Hence, they need to be treated with special care. We illustrate this with the following example from [236].

Example 2.25 (The global minimum is not a KKT point)**.** *Regard the following MPCC:*

$$\begin{aligned}
\min_{x\in\mathbb{R}^3} \quad & x_1 + x_2 - x_3 \\
\text{s.t.} \quad & h_1(x) = 4x_1 - x_3 \geq 0, && \mid \mu_1, \\
& h_2(x) = 4x_2 - x_3 \geq 0, && \mid \mu_2, \\
& G(x) = x_1 \geq 0, && \mid \mu_G, \\
& H(x) = x_2 \geq 0, && \mid \mu_H, \\
& G(x)H(x) = x_1 x_2 \leq 0, && \mid \mu_{GH}.
\end{aligned}$$

It is not difficult to see that $x^* = (0, 0, 0)$ is the global minimizer of this NLP. At the solution all inequalities are active. We try to find Lagrange multipliers $\mu^* = (\mu_1^*, \mu_2^*, \mu_G^*, \mu_H^*, \mu_{GH}^*) \geq 0$ such that the KKT conditions (2.7) *are satisfied. Starting with $\nabla_x L(x^*, \mu^*) = 0$ we have*

$$0 = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} - \mu_1^* \begin{bmatrix} 4 \\ 0 \\ -1 \end{bmatrix} - \mu_2^* \begin{bmatrix} 4 \\ 0 \\ -1 \end{bmatrix} - \mu_G^* \begin{bmatrix} 4 \\ 0 \\ -1 \end{bmatrix} - \mu_H^* \begin{bmatrix} 4 \\ 0 \\ -1 \end{bmatrix} + \mu_{GH}^* \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

From the nonnegativity of the multipliers and this condition, we obtained that

$$\begin{aligned}
& \mu_G^* = 1 - 4\mu_1^* \geq 0, \ \mu_1^* \in [0, 0.25], \\
& \mu_H^* = 1 - 4\mu_2^* \geq 0, \ \mu_2^* \in [0, 0.25], \\
& \mu_1^* + \mu_2^* = 1.
\end{aligned}$$

We conclude that there are no Lagrange multipliers that satisfy the KKT conditions. The KKT conditions are not necessary for the global minimizer in this example.

The reason for the degeneracy in the last example is the absence of CQs. This is not a pathological example. Typically, most standard CQs are violated for generic MPCCs (2.23). For example, there is no feasible point x that satisfies the inequality (2.23f) strictly and this implies that the MFCQ is violated

at every feasible point [236], cf. Definition 2.11. This has several negative consequences: (a) the set of Lagrange multipliers is necessarily unbounded, (b) the gradients of the active constraints are linearly dependent at all feasible points, and (c) the linearization of (2.23) can be inconsistent arbitrarily close to a stationarity point [99]. The tangent cone is usually a nonconvex cone due to the complementarity constraints, cf. Example 2.12. Since the linearized feasible cone is a polyhedral convex cone, in most cases we can only expect the GCQ to hold [98]. This prohibits us to equate the tangent and linear feasible cone and to pass from the FONC to the KKT conditions. To resolve these theoretical difficulties, besides the standard geometric FONC in Theorem 2.5, for MPCC several tailored stationarity concepts that can characterize local minima are derived.

2.3.2 First-order optimality conditions for MPCCs

The stationarity concepts for MPCC are defined utilizing several auxiliary NLPs. We first define the index sets which depend on a feasible point $x \in \Omega_{\text{MPCC}}$:

$$\mathcal{I}_{+0}(x) = \{i \in \{1, \dots, m\} \mid G_i(x) > 0, H_i(x) = 0\},$$

$$\mathcal{I}_{0+}(x) = \{i \in \{1, \dots, m\} \mid G_i(x) = 0, H_i(x) > 0\},$$

$$\mathcal{I}_{00}(x) = \{i \in \{1, \dots, m\} \mid G_i(x) = 0, H_i(x) = 0\}.$$

For ease of notation, we often omit the argument in the index sets, as it is clear from the context which point is meant. The set $\mathcal{I}_{00}$ is called the set of degenerate indices and is usually the source of theoretical and numerical difficulties. If $\mathcal{I}_{00}$ is empty, we say that a solution x^* satisfies *strict complementarity*. We proceed with the definition of the auxiliary NLPs [236].

Definition 2.26 (Auxiliary NLP)**.** *Let $x^* \in \Omega_{\text{MPCC}}$. We define the following auxiliary NLPs:*

- *the Relaxed NLP (RNLP) for $x^* \in \Omega_{\text{MPCC}}$ is defined as*

$$\min_{x \in \mathbb{R}^n} \quad f(x) \tag{2.24a}$$

$$\text{s.t.} \quad g(x) = 0, \tag{2.24b}$$

$$h(x) \geq 0, \tag{2.24c}$$

$$G_i(x) = 0,\ H_i(x) \geq 0,\ i \in \mathcal{I}_{0+}(x^*), \tag{2.24d}$$

$$G_i(x) \geq 0,\ H_i(x) = 0,\ i \in \mathcal{I}_{+0}(x^*), \tag{2.24e}$$

$$G_i(x) \geq 0,\ H_i(x) \geq 0,\ i \in \mathcal{I}_{00}(x^*), \tag{2.24f}$$

- *the Tight NLP (TNLP) for $x^* \in \Omega_{\text{MPCC}}$ is defined as*

$$\min_{x \in \mathbb{R}^n} \quad f(x) \tag{2.25a}$$

$$\text{s.t.} \quad g(x) = 0, \tag{2.25b}$$

$$h(x) \geq 0, \tag{2.25c}$$

$$G_i(x) = 0,\ H_i(x) \geq 0,\ i \in \mathcal{I}_{0+}(x^*), \tag{2.25d}$$

$$G_i(x) \geq 0,\ H_i(x) = 0,\ i \in \mathcal{I}_{+0}(x^*), \tag{2.25e}$$

$$G_i(x) = 0,\ H_i(x) = 0,\ i \in \mathcal{I}_{00}(x^*). \tag{2.25f}$$

- *Let $(\mathcal{I}_1, \mathcal{I}_2)$ be a partition of $\mathcal{I}_{00}$ such that $\mathcal{I}_1 \cup \mathcal{I}_2 = \mathcal{I}_{00}$ and $\mathcal{I}_1 \cap \mathcal{I}_2 = \emptyset$. The Branch NLP ($\text{BNLP}_{(\mathcal{I}_1,\mathcal{I}_2)}$) for $x^* \in \Omega_{\text{MPCC}}$ is defined as*

$$\min_{x \in \mathbb{R}^n} \quad f(x) \tag{2.26a}$$

$$\text{s.t.} \quad g(x) = 0, \tag{2.26b}$$

$$h(x) \geq 0, \tag{2.26c}$$

$$G_i(x) \geq 0,\ H_i(x) = 0,\ i \in \mathcal{I}_{+0}(x^*) \cup \mathcal{I}_1(x^*), \tag{2.26d}$$

$$G_i(x) = 0,\ H_i(x) \geq 0,\ i \in \mathcal{I}_{0+}(x^*) \cup \mathcal{I}_2(x^*). \tag{2.26e}$$

Figure 2.7 illustrates the feasible sets of the auxiliary NLPs for $i \in \mathcal{I}_{00}$. We denote the feasible sets of the RNLP, TNLP, and $\text{BNLP}_{(\mathcal{I}_1,\mathcal{I}_2)}$ by Ω_{RNLP}, Ω_{TNLP} and $\Omega_{\text{BNLP}_{(\mathcal{I}_1,\mathcal{I}_2)}}$, respectively. The usual NLP concepts such as first-order optimality conditions, stationary points, second-order conditions, and constraint qualification for MPCC are defined in terms of these auxiliary NLP. To see that this approach makes sense we look at how these problems and their solutions are related. It is not difficult to see that the following holds for $x^* \in \Omega_{\text{MPCC}}$ [236]:

$$\Omega_{\text{TNLP}} = \bigcap_{(\mathcal{I}_1,\mathcal{I}_2)} \Omega_{\text{BNLP}_{(\mathcal{I}_1,\mathcal{I}_2)}} \subset \Omega_{\text{MPCC}} = \bigcup_{(\mathcal{I}_1,\mathcal{I}_2)} \Omega_{\text{BNLP}_{(\mathcal{I}_1,\mathcal{I}_2)}} \subseteq \Omega_{\text{RNLP}}. \tag{2.27}$$

The same relations hold for the corresponding tangent cones at x^* as well. Furthermore, for a feasible point of the MPCC (2.23) $x^* \in \Omega_{\text{MPCC}}$ the following

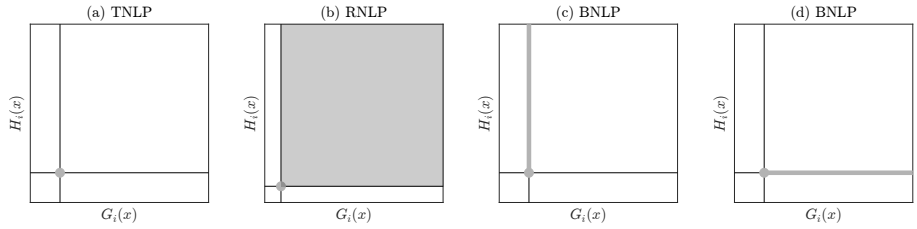


Figure 2.7: Feasible sets of auxiliary NLP.

can be said [236]. If x^* is a local minimizer of the RNLP, then it is a local minimizer of the MPCC. The converse is not true. If x^* is a local minimizer of the MPCC then it is a local minimizer of the TNLP. The point x^* is a local minimizer of the MPCC if and only if it is a local minimizer of every $\text{BNLP}_{(\mathcal{I}_1,\mathcal{I}_2)}$. The last assertion highlights the combinatorial nature of MPCCs, since $2^{|\mathcal{I}_{00}|}$ branch NLPs must be checked to make conclusions about optimality. Fortunately, as we will see below, under reasonable assumptions we do not have to check every branch NLP but only the RNLP or TNLP to characterize a stationary point of the MPCC.

All these difficulties arise because of the degenerate indices $i \in \mathcal{I}_{00}$. If this set is empty, all auxiliary NLP collapse to the same problem, and there is no combinatorial structure due to the BNLP anymore. It can be seen that the tangent cone of Ω_{MPCC} will be convex since there will be no rays that start from the degenerate point, cf. Example 2.12. Assuming that other constraints in the MPCC do not cause the violation of the ACQ, then we have that the standard ACQ holds for the MPCC and thus we can apply the KKT-conditions to verify the stationarity of $x^* \in \Omega_{\text{MPCC}}$ in this fortunate case. In other words, $x^* \in \Omega_{\text{MPCC}}$ is a local minimizer of the MPCC if and only if it is a local minimizer of the RNLP/TNLP, which are equal in this case [236].

Next, we define the MPCC Lagrangian, MPCC CQs, and MPCC stationarity concepts.

Definition 2.27 (MPCC Lagrangian)**.** *The function* $\mathcal{L}^{\text{MPCC}} : \mathbb{R}^n \times \mathbb{R}^{n_g} \times \mathbb{R}^{n_h} \times \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$,

$$\mathcal{L}^{\text{MPCC}}(x, \lambda, \mu, \nu, \xi) \coloneqq f(x) - \lambda^\top g(x) - \mu^\top h(x) - \nu^\top G(x) - \xi^\top H(x), \quad (2.28)$$

with the MPCC Lagrange multipliers $\lambda \in \mathbb{R}^{n_g}$, $\mu \in \mathbb{R}^{n_h}$, $\nu \in \mathbb{R}^m$ *and* $\xi \in \mathbb{R}^m$, *is called the MPCC Lagrangian function of the MPCC* (2.23).

The MPCC Lagrangian is the *standard* Lagrangian for the RNLP/TNLP. It differs from the standard Lagrangian from Definition 2.13 for the MPCC (2.23) in omitting the bilinear terms $G_i(x)H_i(x) \leq 0$ and their multipliers.

If $\mathcal{I}_{00} \neq \emptyset$, then the linearized feasible cone cannot locally capture the structural nonconvexity of the complementarity constraints. To have a more handy tool, the MPCC linearized feasible cone $\mathcal{F}^{\text{MPCC}}_{\Omega_{\text{MPCC}}}(x)$ is used [96, 219, 236].

Definition 2.28 (MPCC linearized feasible cone)**.** *The MPCC linearized feasible cone of* Ω_{MPCC} *at a feasible point* x *is defined as*

$$\begin{aligned}\mathcal{F}^{\text{MPCC}}_{\Omega_{\text{MPCC}}}(x) = \{d \in \mathbb{R}^n \mid & \nabla g(x)^\top d = 0,\\ & \nabla h_i(x)^\top d \geq 0, \ \textit{for all } i \in \mathcal{A}(x),\\ & \nabla G_i(x)^\top d = 0, \ \textit{for all } i \in \mathcal{I}_{0+}(x),\\ & \nabla H_i(x)^\top d = 0, \ \textit{for all } i \in \mathcal{I}_{+0}(x),\\ & 0 \leq \nabla G_i(x)^\top d \perp \nabla H_i(x)^\top d \geq 0, \ \textit{for all } i \in \mathcal{I}_{00}(x)\}.\end{aligned}$$

The combinatorial structure is kept for the degenerate index set $\mathcal{I}_{00}$ and this cone is nonconvex.

Since the standard CQs fail to hold for MPCCs, the KKT conditions are usually not applicable for characterizing a stationary point. Note that the purely geometric standard FONC of Theorem 2.5 can still be used, as it does not require any CQs. Stationary points that satisfy these conditions are in the MPCC community called geometric Bouligand stationary points [98, 165]. Some other concepts rely on the auxiliary NLPs and their Lagrange multipliers. They are summarized in the next definition.

Definition 2.29 (Stationarity concepts for MPCCs)**.** *Let* x^* *be feasible for the MPCC* (2.23)*.*

- *Geometric Bouligand Stationarity (Geometric B-Stationarity) [98, 187]: A point* $x^* \in \Omega_{\text{MPCC}}$ *is called geometric B-stationary if the following holds:*

$$\nabla f(x^*)^\top d \geq 0, \quad \textit{for all } d \in \mathcal{T}_{\Omega_{\text{MPCC}}}(x^*).$$

- *Algebraic Bouligand Stationarity (or just B-Stationarity) [187, 236]: A point* $x^* \in \Omega_{\text{MPCC}}$ *is called B-stationary if* $d = 0$ *is a solution of the*

following linear program with complementarity constraints:

$$\min_{d \in \mathbb{R}^n} \quad \nabla f(x^*)^\top d \tag{2.29}$$

$$\text{s.t.} \quad d \in \mathcal{F}^{\text{MPCC}}_{\Omega_{\text{MPCC}}}(x^*). \tag{2.30}$$

- *Weak Stationarity (W-Stationarity) [236]: A point $x^* \in \Omega_{\text{MPCC}}$ is called weakly stationary if the corresponding TNLP admits the satisfaction of the KKT conditions, i.e., there exist Lagrange multipliers λ, μ, ν and ξ such that:*

$$\begin{aligned}
&\nabla_x \mathcal{L}^{\text{MPCC}}(x^*, \lambda^*, \mu^*, \nu^*, \xi^*) = 0,\\
&g(x^*) = 0,\\
&0 \leq \mu^* \perp h(x^*) \geq 0,\\
&G(x^*) \geq 0, \nu_i^* = 0, \quad \text{for all } i \in \mathcal{I}_{+0}(x^*),\\
&H(x^*) \geq 0, \xi_i^* = 0, \quad \text{for all } i \in \mathcal{I}_{0+}(x^*),\\
&G_i(x^*) = 0,\ \nu_i^* \in \mathbb{R}, \quad \text{for all } i \in \mathcal{I}_{0+}(x^*) \cup \mathcal{I}_{00}(x^*),\\
&H_i(x^*) = 0,\ \xi_i^* \in \mathbb{R}, \quad \text{for all } i \in \mathcal{I}_{+0}(x^*) \cup \mathcal{I}_{00}(x^*).
\end{aligned}$$

- *Strong Stationarity (S-stationarity) [236]: A point $x^* \in \Omega_{\text{MPCC}}$ is called S-stationary if it is weakly stationary and $\nu_i^* \geq 0, \xi_i^* \geq 0$ for all $i \in \mathcal{I}_{00}(x^*)$. In other words it is a KKT point of the corresponding RNLP.*

- *Clarke Stationarity (C-stationarity) [236]: A point $x^* \in \Omega_{\text{MPCC}}$ is called C-stationary if it is weakly stationary and $\nu_i^* \xi_i^* \geq 0$ for all $i \in \mathcal{I}_{00}(x^*)$.*

- *Mordukhovich Stationarity (M-stationarity) [236]: A point $x^* \in \Omega_{\text{MPCC}}$ is called M-stationary if it is weakly stationary and if either $\nu_i^* > 0$ and $\xi_i^* > 0$ or $\nu_i^* \xi_i^* = 0$ for all $i \in \mathcal{I}_{00}(x^*)$.*

- *Abadie Stationarity (A-stationarity) [96]: A point $x^* \in \Omega_{\text{MPCC}}$ is called A-stationary if it is weakly stationary and $\nu_i^* \geq 0$ or $\xi_i^* \geq 0$ for all $i \in \mathcal{I}_{00}(x^*)$.*

The feasible sets for the MPCC multipliers ν^* and ξ^* are depicted in Figure 2.8. Note that S-stationarity corresponds to the KKT conditions of the RNLP. If x^* is M-stationary, then the KKT conditions of at least one BNLP can be satisfied at x^* [165]. The many different stationarity concepts might be confusing and

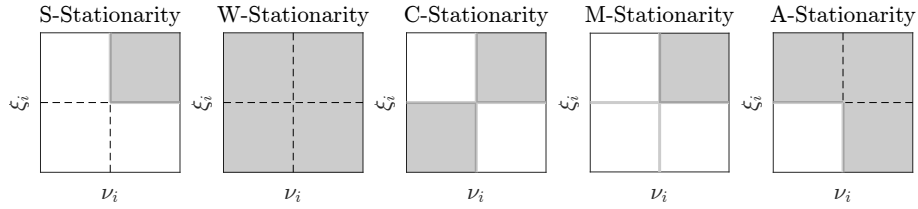


Figure 2.8: Sign restrictions for MPCC multiplier in different stationarity concepts.

one might be wondering if some of them are needed at all. Some of them might even not be necessary as they allow trivial descent directions. We illustrate this with the following counterexample from [180].

Example 2.30 (Descent direction for M-stationarity)**.** *Regard the following MPCC:*

$$\begin{aligned} \min_{x\in\mathbb{R}^2} \quad & (x_1-1)^2 + x_2^2(x_2+1) \\ \text{s.t.} \quad & x_1 \geq 0, \\ & x_2 \geq 0, \\ & x_1 x_2 \leq 0. \end{aligned}$$

At the point $x^* = (0,0)$ *we have that*

$$0 = \nabla_x \mathcal{L}^{\text{MPCC}}(x^*, \nu^*, \xi^*) = \begin{bmatrix} -2 \\ 0 \end{bmatrix} - \nu^* \begin{bmatrix} 1 \\ 0 \end{bmatrix} - \xi^* \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

is satisfied for the multipliers $\nu^* = -2$ *and* $\xi^* = 0$*. Hence,* x^* *is M-, C-, A-, and W-stationary. The directions* $d = \{(d_1, 0) \mid d_1 > 0\}$ *are descent directions.*

However, these stationarity concepts are crucial for studying numerical methods for MPCC. As we will see in the next section, MPCCs are usually solved by solving a (finite) sequence of related and more regular NLPs. Depending on the underlying assumptions, the accumulation points of these methods are some of the stationarity points defined above. Therefore, it is important to understand under which conditions these stationarity concepts are indeed necessary for optimality. It turns out that all of them can be necessary for local optimality if some additional specialized CQs hold [104, 236]. As in the regular NLP case, some of the CQs might be strong but too restrictive, or weak but too difficult to verify in practice.

Similar to the discussion after introducing the auxiliary NLP, we note that most difficulties vanish if there are no degenerate indices. Observe that if $\mathcal{I}_{00} = \emptyset$, then all multiplier-based stationarity concepts collapse to the same. The next definition lists some of the most popular MPCC-tailored CQs [151, 241].

Definition 2.31 (MPCC constraint qualifications)**.** *Let x be a feasible point for the MPCC* (2.23)*.*

- *The MPCC-LICQ holds at a point x if the LICQ holds for the RNLP at x.*
- *The MPCC-MFCQ holds at a point x if the MFCQ holds for the RNLP at x.*
- *The MPCC-ACQ holds if and only if $\mathcal{F}^{\mathrm{MPCC}}_{\Omega_{\mathrm{MPCC}}}(x) = \mathcal{T}_{\Omega_{\mathrm{MPCC}}}(x)$.*
- *The MPCC-GCQ holds if and only if $\mathcal{F}^{\mathrm{MPCC}}_{\Omega_{\mathrm{MPCC}}}(x)^\circ = \mathcal{T}_{\Omega_{\mathrm{MPCC}}}(x)^\circ$.*

The following implications hold [151, 241]:

$$\text{MPCC-LICQ} \implies \text{MPCC-MFCQ} \implies \text{MPCC-ACQ} \implies \text{MPCC-GCQ}.$$

Note that the first two constraint CQs are defined in terms of the RNLP. The linearized feasible cone of these NLPs is always convex and we can expect the standard ACQ to be violated. This motivated the definition of the MPCC-ACQ and MPCC-GCQ in terms of the nonconvex cone $\mathcal{F}^{\mathrm{MPCC}}_{\Omega_{\mathrm{MPCC}}}(x)$ [95, 96, 97, 98]. Interestingly, Flegel and Kanzow [98] prove that under the MPCC-LICQ the standard GCQ holds for a generic MPCC (2.23).

The MPCC-CQs are needed for establishing when some stationarity concept is necessary for a local minimizer of an MPCC. The results from [104, 151, 187, 217, 236] are summarized in the diagram in Figure 2.9. This diagram was inspired by [175, Theorem 5.13] and [165, Figure 2]. The implications between different stationarity concepts involving multipliers follow from their definition [236]. For the missing CQ definitions see [151, 241]. We make a few comments on the relations. As already discussed, if the ACQ does not hold, the KKT conditions are not applicable for characterizing a Geometric B-stationarity point. B-stationarity always implies geometric B-stationarity, the converse requires additionally the MPCC-GCQ to hold [96]. S-stationarity corresponds to the KKT conditions of the RNLP and it implies B-stationarity because of the inclusion relation in Eq. (2.27). Conversely, if the MPCC-LICQ holds, then B-stationarity implies also S-stationarity [236]. This is unfortunately not anymore true if the MPCC-LICQ is replaced by weaker CQs. For a counterexample, one can verify that in Example 2.25 the MPCC-LICQ is violated at x^*, where the weaker MPCC-MFCQ holds [241]. This means that B-stationary points that

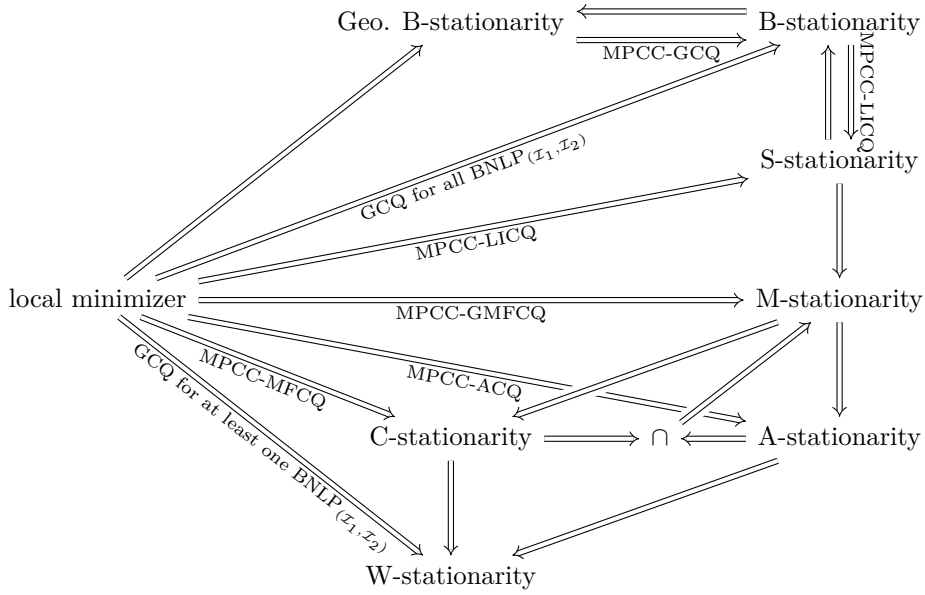


Figure 2.9: Diagram summarizing MPCC-CQs and necessity of different stationarity concepts for optimality.

are not S-stationary can not be identified via any stationarity concept based on the auxiliary NLPs [277]. Instead, an exponential number of linear programs (2.29) must be solved to verify B-stationarity.

As seen from the discussion above, the desired and easy-to-verify concept is S-stationarity. It does not allow trivial descent directions and requires no combinatorial exploration. Arguably, the needed assumptions might be too restrictive [180, 169]. To this end, we restate a remarkable theorem that relates S-stationary and B-stationary points. It additionally relates the MPCC and standard Lagrange multipliers. It was independently derived in slightly different forms in [104, Proposition 4.1], [12, Theorem 2.2] and [146, Proposition 1]. We denote the standard Lagrange multipliers for the complementarity constraint in (2.23) by (μ_G, μ_H, μ_{GH}) and remind the reader that the relevant MPCC multipliers are denoted by (ν, ξ).

Theorem 2.32. *A point x^* is strongly stationary for the MPCC* (2.23) *if and only if it is stationary for the RNLP* (2.24)*. Furthermore, if $(\mu_G^*, \mu_H^*, \mu_{GH}^*)$ are standard Lagrange multipliers associated with x^*, then the MPCC-multipliers*

(ν^*, ξ^*) can be computed by

$$\nu_i^* = \mu_{G,i}^* - \mu_{GH,i}^* H_i(x^*) \tag{2.31a}$$

$$\xi_i = \mu_{H,i}^* - \mu_{GH,i}^* G_i(x^*). \tag{2.31b}$$

Conversely, if (ν^*, ξ^*) are MPCC-multipliers associated with x^*, then any $(\mu_G^*, \mu_H^*, \mu_{GH}^*)$ satisfying (2.31) with

$$\mu_{GH}^* \geq \bar{\mu}_{GH} = \max\left(0, \max_{i \in \mathcal{I}_{0+}} -\frac{\nu_i^*}{H_i(x^*)}, \max_{i \in \mathcal{I}_{+0}} -\frac{\xi_i^*}{G_i(x^*)}\right), \tag{2.32}$$

are standard Lagrange multipliers of (2.23) associated with x^*.

The proof follows from the direct comparison of the KKT conditions of the RNLP and the MPCC. This theorem reveals that if MPCC-LICQ holds, which implies the uniqueness of the multipliers (ν^*, ξ^*), then the standard multipliers of (2.23) are the unbounded ray defined by (2.31) with the origin at $\bar{\mu}_{GH}$. An interesting interpretation of the result is given in [104]. The standard multipliers $(\mu_G^*, \mu_H^*, \mu_{GH}^*)$ must be nonnegative, whereas the MPCC multipliers (ν^*, ξ^*) can become negative for a sufficiently large μ_{GH}^*. This suggests that a constraint regarded as an inequality should be treated as an equality constraint. Of course, such an analysis is only a posteriori possible.

There exist also second-order optimality conditions tailored to MPCCs. They are defined in terms of S-stationary points and the corresponding MPCC multipliers. We omit their statement for brevity and reefer the reader to [228, 236] for more details.

2.4 MPCC solution methods

In this section, we review some standard MPCC solution methods. As seen in Section 2.1.2, the core of most NLP solvers are Newton-type methods for finding zeros of the KKT conditions of a regular NLP satisfying LICQ or MFCQ. Newton-type methods usually assume locally isolated solutions for superlinear convergence [149]. This assumption is not satisfied for MPCCs since due to the violation of MFCQ, the set of Lagrange multipliers is necessarily unbounded.

In the last three decades, many tailored MPCC solution methods were proposed. A recent survey of MPCC methods is given in [165]. The references [106, 139, 159] provide comparisons of several solutions strategies. Good overviews are also provided in the PhD theses [175] and [277]. We classify the MPCC solution methods as:

(I) NLP solution methods,

(II) regularization methods,

(III) exact penalty methods,

(IV) lifting methods,

(V) implicit methods,

(VI) combinatorial methods.

Some authors treat the methods from I-IV as a single class [165, 175], as they mainly use standard NLP methods. The availability of robust NLP codes makes their implementation easy and practical. Thus, we make a finer distinction and look closer at their properties. In this thesis, they are used for solving the discrete-time optimal control problems derived in Chapters 7 to 9. Most of them are implemented in the open-source package `nosnoc` [206].

Methods from classes I-IV require solving a sequence of related and more regular NLPs, which are parameterized by a parameter $\sigma > 0$. We denote the solution of the initial MPCC (2.23) by x^* and the solution of the regularized NLP by $x^*(\sigma)$. The obvious goal is that $x^*(\sigma) \to x^*$ as $\sigma \to 0$.

In this thesis, we do not use methods from classes V and VI for several reasons. They are more difficult to implement than methods from classes I to IV and there are currently no mature open-source software implementations available. Moreover, they require tools from nonsmooth optimization or they might not be applicable to the MPCCs arising form discretizing optimal control problems. Nevertheless, we provide several references for these approaches.

2.4.1 NLP solution methods

Methods from this class simply threats the MPCC as a nonlinear program and applies directly standard NLP techniques such as SQP and IP methods for solving (2.23) [102, 103, 178, 185]. Despite the degeneracy discussed at the beginning of the last section (ill-conditioning, inconsistent linearizations), this approach can perform surprisingly well in practice. However, they tend to have convergence difficulties or to converge to spurious stationary points if the MPCC-LICQ does not hold. Depending on how the complementarity constraints enter the NLP, we distinguish between direct solution methods and NCP reformulations.

Direct solution This approach consists of the straightforward application of a standard NLP solver to the MPCC (2.23), possibly with alternative equivalent formulations of the complementarity constraints (2.23d)-(2.23f). Fletcher and Leyffer study the practical performance of SQP methods on numerous MPCCs in [101] and investigate their local convergence properties in [103]. Under the assumptions that the MPCC-LICQ and MPCC-SOSC hold, that all QPs remain feasible, and other technical assumptions, they show quadratic convergence to S-stationary points. The use of slack variables s_G and s_H in Eq. (2.22) is necessary for this result.

Moreover, in practice the lumped formulation $s_G^\top s_H \leq 0$ gives better performance than the component-wise $s_{G,i} s_{H,i} \leq 0,\ i = 1, \ldots, m$. A possible explanation is that the lumped complementarity constraint allows nonmonotonous changes in the complementarity pairs during the iterates, which improves the convergence [179]. The equality constraint formulation of the bilinear term usually results in worse performance. The authors show with several counterexamples [103] that their assumptions cannot be relaxed.

Interior-point methods perform reasonably well if applied directly to the NLP formulation (2.23) [266, 267]. However, their performance improves when paired with relaxation and exact penalty formulations, as we will highlight several times below.

NLP formulations with NCP functions In this approach the complementarity constraints (2.23d)-(2.23f) are replaced by a C-functions $\Psi_{\mathrm{C}}(G(x), H(x)) = 0$. The resulting NLP is solved by a standard globalized NLP solver. The use of SQP methods in such formulations was studied by Leyffer [178]. Since the C-functions are not differentiable $(0, 0)$, their subgradient is used. Evidently, using squared versions (or higher powers) of C-functions will not improve the situation and lead to a violation of LICQ at $(0, 0)$ since we obtain a zero-gradient at this point. Similar to [103], assuming MPCC-LICQ, MPCC-SOSC, Lipschitz continuity of the NLP functions and their derivatives and other technical assumptions, local superlinear convergence to S-stationary points was shown. Leyffer tests this approach on a wide number of test problems and shows that different NCP functions can lead to large differences in performance. Similar to the first approach, lumping all NCP functions in one constraint, i.e., $\mathbf{e}^\top \Psi_{\mathrm{C}}(G(x), H(x)) \leq 0$ often gives better performance.

2.4.2 Regularization methods

Regularization methods involve relaxing or smoothing the complementarity constraints using a parameter $\sigma > 0$. This results in a more standard and regular NLP that typically satisfies the standard MFCQ. As a result, the regularized NLP can be solved using a readily available solver. A smaller value of σ yields a better approximation to the original problem. When $\sigma = 0$, the problem becomes equivalent to the initial mathematical program with MPCC defined in equation (2.23).

In these approaches, a sequence of the relaxed NLPs for $\sigma^k \to 0$ is solved. If the problems are solved exactly, under mild assumptions, accumulation points of the sequence of solutions $x^*(\sigma^k)$ are C-stationary points [237, 228, 139]. Hoheisel et al. [139] provide a detailed numerical and theoretical comparison of several methods from this family. Solving the NLPs inexactly usually weakens the convergence results [159]. Of course, stronger assumptions also result in convergence to M- or S-stationary points. We provide details for some of the most popular regularization methods, which we use in this thesis.

Global relaxation/smoothing method by Scholtes [237] This is probably the easiest-to-implement approach and relaxes the bilinear constraint (2.23f) as

$$G_i(x)H_i(x) \leq \sigma, \text{ for all } i \in \{1, \dots, m\}.$$

An illustration of the relaxed feasible set is given in Figure 2.10. Alternatively, the bilinear term might be smoothed

$$G_i(x)H_i(x) = \sigma, \text{ for all } i \in \{1, \dots, m\}.$$

Lumped versions $G(x)^\top H(x) \leq \sigma$ or $G(x)^\top H(x) = \sigma$ are also used frequently [237]. Observe that in contrast to the smoothed variant, the relaxed version contains the feasible set Ω_{MPCC}, and one might find with it a stationary point without driving $\sigma \to 0$. Assuming MPCC-LICQ, Scholtes [237] shows convergence to C-stationary points. Hoheisel et al. [139] obtain the same result under the weaker MPCC-MFCQ. Ralph et al. [228] study the convergence speed of this approach and show that, under rather strict assumptions, the local solution map $x^*(\sigma)$ of the relaxed formulation is a piecewise continuous function and that the solution converges with a rate $O(\sigma)$. Milder assumptions (MPCC-MFCQ and RNLP-SOSC) result in the rate $O(\sigma^{\frac{1}{2}})$ for the relaxed variant and $O(\sigma^{\frac{1}{4}})$ for the smoothed variant.

For more efficacy, Raghunathan and Biegler [227], Liu and Sun [185] propose interior-point methods where the relaxation parameter σ is proportional to the

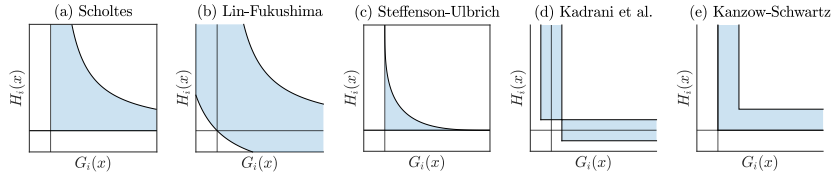


Figure 2.10: Illustration of the regularized complementarity sets.

barrier parameter τ. Under some stronger assumptions, the former strategy is shown to converge quadratically to S-stationary points. The next four more sophisticated regularization schemes converge to M-stationary points under fairly mild conditions and perform reasonably well in practice [139].

The smooth relaxation method by Lin and Fukushima [182] This method is similar to Scholtes' regularization and replaces the complementarity conditions (2.23d)-(2.23f) by:

$$G_i(x)H_i(x) \leq \sigma^2, \text{ for all } i \in \{1, \ldots, m\},$$

$$(G_i(x) + \sigma)(H_i(x) + \sigma) \geq \sigma^2, \text{ for all } i \in \{1, \ldots, m\}.$$

Figure 2.10 (b) shows an illustration of the feasible set. Lin and Fukushima obtain similar convergence results as Scholtes [237]. Hoheisel et al. [139] extend this result by proving convergence to C-stationary points under the MPCC-MFCQ. Moreover, they show that the feasible points of the relaxed NLP satisfy the MFCQ in a neighborhood of a point $x \in \Omega_{\text{MPCC}}$.

The local relaxation method by Steffensen and Ulbrich [248, 277] Almost all regularization methods make global changes to the feasible set. Motivated by the fact that most difficulties arise for degenerate complementarity pairs $i \in \mathcal{I}_{00}$, Steffensen and Ulbrich follow a different approach. Their main idea is to relax the complementarity constraint only locally at the corner of the L-shaped set arising from the complementarity constraints.

The relaxation is achieved with the following steps: the L-shaped set is rotated with a linear transformation by $\frac{\pi}{4}$ counterclockwise for every complementarity pair, and one obtains the graph of the abs-function. On the interval $[-\sigma, \sigma]$, the kink is replaced by a sufficiently smooth function such that the continuity of the functions and their derivatives is preserved at the interval boundaries. Finally, the inverse transformation is carried out, and a locally relaxed set is

obtained, cf. Figure 2.10 (c). This reasoning expressed in equations reads as

$$\Phi_{\mathrm{SU}}(G_i(x), H_i(x), \sigma) \leq 0, \text{ for all } i \in \{1, \ldots, m\},$$

where $\Phi_{\mathrm{SU}} : \mathbb{R} \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is defined in terms of the auxiliary functions $\phi^a_{\mathrm{SU}} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ and $\phi^b_{\mathrm{SU}} : [-1, 1] \to \mathbb{R}$ as follows

$$\Phi_{\mathrm{SU}}(y_1, y_2; \sigma) = y_1 + y_2 - \phi^a_{\mathrm{SU}}(y_1 - y_2, \sigma),$$

$$\phi^a_{\mathrm{SU}}(z, \sigma) = \begin{cases} |z|, & \text{if } |z| \geq \sigma, \\ \sigma \phi^b_{\mathrm{SU}}(\frac{z}{\sigma}), & \text{if } |z| < \sigma. \end{cases}$$

The function ϕ^b_{SU} satisfies the following properties:

(i) ϕ^b_{SU} it is twice continuously differentiable on $[-1, 1]$,

(ii) $\phi^b_{\mathrm{SU}}(1) = \phi^b_{\mathrm{SU}}(-1) = 1$,

(iii) $\frac{\mathrm{d}}{\mathrm{d}z}\phi^b_{\mathrm{SU}}(-1) = -1$ and $\frac{\mathrm{d}}{\mathrm{d}z}\phi^b_{\mathrm{SU}}(1) = 1$,

(iv) $\frac{\mathrm{d}^2}{\mathrm{d}z^2}\phi^b_{\mathrm{SU}}(-1) = \frac{\mathrm{d}^2}{\mathrm{d}z^2}\phi^b_{\mathrm{SU}}(1) = 0$,

(v) $\frac{\mathrm{d}^2}{\mathrm{d}z^2}\phi^b_{\mathrm{SU}}(z) > 0$ for all $z \in (-1, 1)$.

Examples of such functions are $\phi^b_{\mathrm{SU}}(z) = \frac{1}{8}(-z^4 + 6z^2 + 3)$ and $\phi^b_{\mathrm{SU}}(z) = \frac{2}{\pi}\sin(z\frac{\pi}{2} + \frac{3\pi}{2}) + 1$ [248].

Under the MPCC-CRCQ (cf. [139, Defintion 2.4]) convergence to C-stationary and under the MPCC-LICQ to M-stationary points is shown [248]. A lumped version $\sum_i^m \Phi_{\mathrm{SU}}(G_i(x), H_i(x), \sigma) \leq 0$ can be used as well.

The nonsmooth relaxation method by Kadrani et al. [157] Another interesting relaxation reads as:

$$G_i(x) \geq -\sigma, \; H_i(x) \geq -\sigma, \text{ for all } i \in \{1, \ldots, m\}$$

$$(G_i(x) - \sigma)(H_i(x) - \sigma) \leq 0, \text{ for all } i \in \{1, \ldots, m\}.$$

Figure 2.10 (d) illustrates the nonsmooth feasible set obtained from the constraint above. The convergence study of [157] is carried out assuming the MPCC-LICQ. Once again, Hoheisel et al. [139] improve the result and show convergence to M-stationary points under the MPCC-CPLD (cf. [139, Defintion 2.4], a CQ weaker than the MPCC-MFCQ and stronger than the MPCC-ACQ). It is evident from the structure of the feasible set of this relaxation that verifying standard CQs is more difficult. Thus, only standard GCQ is shown under the MPCC-LICQ [157, 139, 98].

The relaxation method by Kanzow and Schwartz [158] This relaxation has stronger theoretical properties than the previous one and a more satisfactory shape of the feasible set, cf. Figure 2.10 (e). In contrast to the approach of Kadrani et al., it contains the feasible set of the MPCC. This relaxation is modeled with the following equations:

$$\Phi_{\mathrm{KS}}(G_i(x), H_i(x), \sigma) \leq 0, \text{ for all } i \in \{1, \ldots, m\},$$

with $\Phi_{\mathrm{KS}} : \mathbb{R} \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ and $\phi_{\mathrm{KS}} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ where $\Phi_{\mathrm{KS}}(y_1, y_2, \sigma) = \phi_{\mathrm{KS}}(y_1 - \sigma, y_2 - \sigma)$ and

$$\phi_{\mathrm{KS}}(y_1, y_2) = \begin{cases} y_1 y_2, & \text{if } y_1 + y_2 \geq 0, \\ -\frac{1}{2}(y_1^2 + y_2^2), & \text{if } y_1 + y_2 < 0. \end{cases}$$

The function ϕ_{KS} is a continuously differentiable C-function [158]. Under the MPCC-CPLD (cf. [139, Definition 2.4]) convergence to M-stationary points is shown [158, 139]. The relaxed NLP is shown to satisfy at least the GCQ.

Smoothed NCP functions Early MPCC algorithms considered smoothed and everywhere differentiable variants of C-functions from the Definition 2.23. We denote them by $\tilde{\Phi}_{\mathrm{C}}(a, b, \sigma)$. For example, the smoothed Fischer-Burmeister functions read as $\tilde{\Phi}_{\mathrm{FB}}(a, b, \sigma) = a + b + \sqrt{a^2 + b^2 + \sigma^2}$. Facchinei et al. [88] show the convergence of this approach to C-stationary points.

Two sided-relaxation method by DeMiguel et al. [72] All methods above are governed by a single scalar parameter σ. The relaxation of DeMiguel et al. works with the slack formulation (2.22) and uses three nonnegative parameters $\sigma_G, \sigma_H, \sigma_{GH} \in \mathbb{R}^m$:

$$s_G \geq -\sigma_G, \; s_H \geq -\sigma_H,$$

$$s_{G,i} s_{H,i} \leq \sigma_{GH,i}, \text{ for all } i \in \{1, \ldots, m\}.$$

The parameters are updated in each iteration depending on the sign of the multipliers to enforce convergence to S-stationary points. Either σ_G and σ_H or σ_{GH} are reduced, but never both at the same time. This relaxation is paired with an interior-point algorithm. Convergence to S-stationary points is given under MPCC-LICQ and strong SOSC. Milder assumptions result in convergence to C-stationary points [72].

2.4.3 Exact penalty methods

Exact penalty reformulations are one of the most often used approaches to treat degenerate NLPs [33, 57, 111]. It is no surprise that these ideas are adapted to MPCCs. In exact penalty algorithms, the bilinear term (2.23b) is added to the objective in some suitable form and multiplied by a penalty factor ρ. To be consistent with our notation above and the implementations in `nosnoc` [206], we use $\rho^k = \frac{1}{\sigma^k}$. Assuming sufficient regularity of other constraints and having the bilinear term in the objective, we obtain a regular NLP that usually satisfies the MFCQ. Under suitable regularity assumptions and for a sufficiently large and finite ρ, the solution $x^*(\sigma)$ matches the solution x^* of the initial MPCC after a single NLP solve. In practice, a sequence of NLP is solved to improve the convergence and to estimate the correct penalty parameter value. We discuss several of the most widely used formulations below.

ℓ_1-exact penalty formulation This formulation penalizes the $G(x)^\top H(x)$, which corresponds to the ℓ_1 norm of the complementarity residual in the objective and solves a sequence of the following NLPs

$$\min_x \quad f(x) + \rho^k G(x)^\top H(x) \tag{2.33a}$$

$$\text{s.t.} \quad g(x) = 0, \tag{2.33b}$$

$$h(x) \geq 0, \tag{2.33c}$$

$$G(x) \geq 0,\ H(x) \geq 0. \tag{2.33d}$$

The formulation with the complementarity slacks (2.22) is also used frequently, where the term $\rho s_G^\top s_H$ is added to the objective. This approach was first proposed in [91] for solving practical problems. Anitescu [11] provided the first convergence analysis for the ℓ_1 penalty approach paired with active-set SQP methods. Remarkably, Ralph et al. [228] show that an MPCC solution x^* is also a solution to (2.33) for a sufficiently large ρ and that regularity conditions of the MPCC (e.g., MPCC-LICQ) imply regularity of the NLP (2.33). However, if the local minimizers are only B-stationary but not S-stationary points, the penalty parameter must grow to infinity [165].

Leyffer et al. [179] propose an interior-point algorithm to solve the NLP (2.33) while dynamically updating the penalty parameter ρ. For each fixed ρ^k, the barrier subproblem is solved inexactly to a tolerance proportional to the barrier parameter τ^k. Strategies to steer the penalty parameter that avoid too large increases and unbounded subproblems are proposed as well. Convergence to

an S-stationary point is shown under the MPCC-LICQ, RNLP-SOSC. Under additional conditions, a superlinear convergence rate is proved.

Fukushima et al. [107] suggest an SQP method paired with a penalized NCP function. Hu and Ralph [145] relate relaxation methods of [237] and give conditions for convergence to B-stationary points. They study more general formulations than (2.33) and suggest for example to use $\sum_{i=1}^{m} \Phi_{\mathrm{FB}}(G_i(x), H_i(x))^3$ as a penalty function. Furthermore, by comparing the KKT conditions, Leyffer et al. [179] show that there exists a one-to-one correspondence between the iterates of a smoothing and penalty approach.

Hall et al. [127, 128] proposes a Sequential Convex Programming (SCP) method for solving the penalty problem arising from quadratic programs with complementarity constraints. In particular, the method makes use of the fact that QP matrices do not need to be re-factorized in an SCP approach, which enables fast and cheap iterations. The algorithm is paired with an exact analytic line search. The same convergence results as in [228] can be applied.

The great practical difficulty in exact penalty methods is steering the penalty parameter. Byrd et al. [57] introduce a line-search SQP ℓ_1-exact penalty method for general degenerate NLPs. They propose penalty update rules based on solving LPs and QPs with a trust region to predict the decrease of the merit function. Favorable theoretical properties and good numerical performance on a series of test problems including MPCCs are reported. Unfortunately, a robust open-source implementation is still not available. Thierry and Biegler [266, 267] adapt the ℓ_1 strategy of Byrd et al. [57] including the penalty steering rules, to solve degenerate problems with `IPOPT` [281]. Good practical performance is reported on the MacMPEC test set [177] with an improvement in terms of speed and robustness compared to a direct application of `IPOPT`.

ℓ_∞-exact penalty formulation As in general nonlinear programming, using an ℓ_∞ penalty function for MPCC is also very common [33]. Thereby a sequence of the following NLPs needs to be solved

$$
\begin{aligned}
\min_{x,s} \quad & f(x) + \rho^k s \\
\text{s.t.} \quad & g(x) = 0, \\
& h(x) \geq 0, \\
& G(x) \geq 0,\ H(x) \geq 0, \\
& G_i(x) H_i(x) \leq s,\ i = 1, \ldots, m,
\end{aligned}
$$

$$0 \leq s \leq \bar{s}.$$

where $s \in \mathbb{R}$ is a slack variable and $\bar{s} > 0$ is its upper bounds. This enables us to express the ℓ_∞ norm smoothly. Observe that if the bilinear terms are lumped together, and we use the constraint $G(x)^\top H(x) \leq s$, we end up with an ℓ_1 formulation.

Anistescu's elastic mode [12, 16] A mixture of the approaches above is the elastic mode, which takes an ℓ_1 norm of the bilinear complementarity terms and an ℓ_∞ penalization of the relaxation of the standard equality and inequality constraints. The elastic mode NLP reads as

$$\begin{aligned}
\min_{x,s} \quad & f(x) + \rho^k (G(x)^\top H(x) + s) \\
\text{s.t.} \quad & -s \leq g_i(x) \leq s, \qquad \text{for all } i \in \{1, \ldots, n_g\}, \\
& h_i(x) \geq -s, \qquad \text{for all } i \in \{1, \ldots, n_h\}, \\
& G(x) \geq 0, H(x) \geq 0, \\
& 0 \leq s \leq \bar{s},
\end{aligned}$$

where $s \in \mathbb{R}$ is the elastic mode variable and $\bar{s}$ is its upper bound. Usually, a sequence of NLPs is solved inexactly with an increasing ρ^k. Anitescu et al. [16] show under the MPCC-LICQ and other assumptions the global convergence of an elastic mode SQP approach to C-, M- and S-stationary points. The elastic mode with a fixed penalty parameter is implemented in SNOPT as a fallback strategy if an infeasible or unbounded QP is detected [112].

Finally, we mention the family of augmented Lagrangian methods for MPCC, which also belong to the class of penalty methods [106, 150]. We do not treat them in detail here. Assuming the MPCC-LICQ, and that the sequence of Lagrange multipliers is bounded, convergence to S-stationary points is shown in [106, 150].

2.4.4 Lifting methods

Lifting methods are somewhat in between relaxation and penalty methods. The main idea is to introduce *lifting variables* and regard a more regular feasible set in a higher-dimensional space whose orthogonal projection is the L-shaped set, coming from the complementarity constraints. Some of them require penalization of the lifting variables to recover the solution of the initial problem

[131], and others might require additional regularization [249]. Unfortunately, they have weaker theoretical properties than regularization methods, cf. Section 2.4.7. Thus, we do not implement these methods and do not treat them in further detail there. We mention the methods of Stein [249], Hatz et. al [131], and Izmailov et al.[150].

2.4.5 Combinatorial methods

The methods explicitly treat the combinatorial nature of the complementarity constraints (2.23d)-(2.23f). They have the strongest convergence properties as they are usually guaranteed to converge to B-stationary points. They can be subdivided into branching methods [165, 23], active-set methods [180, 169, 165, 110] and pivoting methods [165]. These methods rely on guessing the correct active set by solving a linear program with complementarity constraints (LPCC) from the definition of algebraic B-stationarity [180, 169], cf. Definition 2.29. If $d = 0$ solves the LPCC, a B-stationarity point is found. To promote faster convergence rates, based on the active-set guess, an equality-constrained QP can be solved [180, 169]. As the solution of an LPCC can be computationally expensive, Kirches et al. [169] regard LPCC only with complementarity and bound constraints, which in turn can be solved with linear complexity. They suggest to treat the remaining equality and inequality constraints in an augmented Lagrangian fashion. This is later done by Guo and Deng [120], where convergence to M-stationary points is proven. The main practical drawback of this method class is the lack of robust open-source implementations. For more references we refer the reader to [175, Section 5.6.2] and [165, Section 3.2].

2.4.6 Implicit methods

Implicit methods are tailored to bilevel optimization problems. The lower-level optimization problem is parameterized by the upper-level optimization variables. It is assumed that for every choice of upper-level variables, the lower-level problem has a (locally) unique solution. Hence it can be viewed as an implicit function of the upper-level variables with locally isolated branches. The lower-level variables are eliminated via their KKT conditions, and a reduced nonsmooth optimization problem is solved, e.g., via semi-smooth Newton methods. An overview of such methods is provided in [218].

Type	**Method**	CQ for MPCC	Limiting Point	Subproblem NLP satisfies	Citation
Direct	Fletcher et al.	MPCC-LICQ	S	GCQ	[104, 103]
	Leyffer	MPCC-LICQ	S	GCQ	[179]
Regular-	Scholtes	MPCC-MFCQ	C	MFCQ	[237, 139]
ization	Lin-Fukushima	MPCC-MFCQ	C	MCFQ	[182, 139]
	Kadrani et al.	MPCC-MFCQ	M	GCQ	[139, 157]
	Steffensen-Ulbrich	MPCC-CPLD	C	ACQ	[139, 248]
	Kanzow-Schwartz	MPCC-CPLD	M	GCQ	[139, 158]
	Raghunathan-Biegler	MPCC-LICQ	S	MFCQ	[227]
Lifting	Stein	MPCC-LICQ	C	LICQ	[249]
	Izmailov-Solodov	MPCC-LICQ	C	LICQ	[147]
	Hatz et al.	MPCC-LICQ	S	GCQ	[131]
Penalty	ℓ_1-Penalty	MPCC-LICQ	S	LICQ	[228, 178]
	Leyffer et al.	MPCC-LICQ	C	LICQ	[178, 228]
	ℓ_∞-Penalty	MPCC-LICQ	S	LICQ	[228, 33]
	Elastic mode	MPCC-LICQ	C	MFCQ	[12, 16]

Table 2.1: Overview of convergence properties of MPCC tailored methods from groups I to IV.

2.4.7 Summary of MPCC methods

With a standard NLP solver at hand the methods from classes I-IV are easy to implement. Table 2.1 provides an overview of known convergence results for direct, regularization, lifting, and penalty methods. Together with the commutative diagram after Definition 2.31, it might help one to decide which algorithm to choose to compute a stationary point of the MPCC (2.23). The strongest multiplier-based stationarity concept is S-stationarity, followed by M-, C-, A- and W-stationarity, sorted from stronger to weaker. One should not be discouraged by the weaker limiting points of the relaxation methods. In contrast to the other methods, these results are obtained under much weaker assumptions. More restrictive assumptions give a better result. For example, the global relaxation method by Scholtes [237] converges to B-stationary points under the MPCC-LICQ and the upper-level strict complementarity, cf. [228, Definition 2.6].

We note that all methods converging to an S-stationary point under the MPCC-LICQ also assume other restrictive assumptions as the upper-level strict complementarity [104, 131, 178, 227]. In practice, one never solves the NLP subproblems exactly, due to the solver tolerances and finite arithmetic precision. However, solving the NLPs in the sequence inexactly can weaken convergence results [159, 16]. For example, under inexact solves the methods of Kadrani et al., Kanzow-Schwartz [158] and Steffensen-Ulbrich [248] converge only to W-

stationary points. Surprisingly, the methods of Scholtes and Lin-Fukushima are immune to this, and they still converge to C-stationary points [159]. However, if some stronger assumptions are not satisfied, usually the MPCC-LICQ, they can experience slow convergence rates and converge to M-stationary points or weaker. This motivated the development of combinatorial methods, for which currently no mature open-source implementations are available.

Chapter 3

Direct Optimal Control Methods

In direct optimal control approaches, we first discretize the infinite-dimensional optimal control problem and solve a finite-dimensional NLP, e.g., with some of the methods from the previous chapter. From now on, we call the direct optimal control methods for smooth dynamical systems *standard methods*. The standard methods are already at a very mature stage with many real-world applications [38, 78, 229]. However, the theory and algorithms for smooth systems do not naturally extend to nonsmooth systems, and standard methods may fail in surprising ways, as we will explore in later chapters. Despite these challenges, numerical discretization and direct optimal control methods for smooth dynamical systems serve as the starting point for the generalizations we aim to develop in this thesis. Hence, it is necessary to be familiar with them. Optimal solutions of nonsmooth optimal control problems consist usually of piecewise smooth functions, where the approximations of the smooth parts can essentially be obtained by the established methods. In this chapter, we present concepts which are used as building blocks in the development of numerical methods for nonsmooth dynamical systems in this thesis.

Outline. This chapter is structured as follows. In Section 3.1 we provide some basic definitions regarding controlled dynamical systems. Section 3.2 discusses some standard numerical simulations and sensitivity computation for ODEs and differential algebraic equations. The chapter concludes with Section 3.3, where we discuss optimal control problems and common solution approaches, with a focus on direct optimal control methods.

3.1 Controlled dynamical systems

We start with regarding smooth controlled dynamical systems, i.e., a system where an external control function (also called control input, manipulated variable, input, or just control) enters the right-hand side and thus manipulates the evolution of the differential and algebraic states. In this thesis, we will solve optimal control problems to compute these control inputs (or approximation of them).

For the exposition in this section, we assume that the control input is known and sufficiently regular (e.g., measurable or continuous). In particular, we are interested in Ordinary Differential Equations (ODE) and Differential Algebraic Equations (DAE). We review some standard numerical integration methods with a focus on implicit Runge-Kutta methods. The section finishes by reviewing some facts about sensitivity propagation for smooth ODEs.

3.1.1 Ordinary differential equations (ODEs)

Definition 3.1. *Let $t \in \mathbb{R}$ be the time, $x(t) \in \mathbb{R}^{n_x}$ the differential states and $u(t) \in \mathbb{R}^{n_u}$ a given control function. Denote by $\dot{x}(t) = \frac{\mathrm{d}x(t)}{\mathrm{d}t}$ the derivative of the differential state w.r.t. time. Let $F : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ be a function such that the Jacobian $\frac{\partial F}{\partial \dot{x}}(\cdot)$ is invertible. The system of equations:*

$$F(t, \dot{x}(t), x(t), u(t)) = 0, \tag{3.1}$$

is called an ordinary differential equation. Given a function $f : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ then a system of equations:

$$\dot{x}(t) = f(t, x(t), u(t)), \tag{3.2}$$

is called an explicit ordinary differential equation.

Note that an explicit ODE (3.2) can be always written as an implicit ODE (3.1) by defining $F(t, \dot{x}(t), x(t), u(t)) = \dot{x}(t) - f(t, x(t), u(t))$. To simplify our exposition, we focus on explicit ODEs from now on. We will often call explicit ODEs just a differential equations or dynamical systems or simply dynamics. The function $f(\cdot)$ is often called the right-hand side (r.h.s.) of the ODE or vector field. In this chapter, we additionally assume that the functions $F(\cdot)$ and $f(\cdot)$ are sufficiently smooth in all arguments, e.g., in direct optimal control (see Section 3.3) we need them to be twice continuously differentiable.

The explicit dependence on t in $F(\cdot)$ and $f(\cdot)$ can always be removed by introducing a differential clock state y with the dynamics $\dot{y}(t) = 1$. Similarly, if

there is a dependence on a time-independent parameter p, an auxiliary state p with the dynamics $\dot{p} = 0$ can be introduced to remove the explicit dependence.

Definition 3.2 (Initial Value Problem)**.** *Regard a time interval $t \in [0, T]$ and suppose the control function $u(t)$ is known. The ODE from Eq.* (3.2) *together with a given an initial condition $x(0) = x_0 \in \mathbb{R}^{n_x}$ is called an Initial Value Problem (IVP) and is usually written as*

$$x(0) = \bar{x}_0, \tag{3.3a}$$

$$\dot{x}(t) = f(t, x(t), u(t)), \text{ for all } t \in [0, T]. \tag{3.3b}$$

As for any system of equations, the question of the existence and uniqueness of solutions plays a central role also for IVPs. Answers to them are provided in the famous Picard-Lindelöf Theorem.

Theorem 3.3 (Picard-Lindelöf Theorem)**.** *Regard the IVP from Definition 3.2 and suppose that the function $f(\cdot)$ is Lipschitz continuous in x and continuous in t and u. Then there exists some $\epsilon > 0$ such that the IVP* (3.3) *has a unique solution $x(t)$ on the time interval $t \in \mathcal{B}_\epsilon(0)$.*

3.1.2 Differential algebraic equations (DAEs)

Many practical problems are more intuitively and easier modeled by DAEs. An extensive list is given in [47]. Formally, DAEs arise when the Jacobian $\frac{\partial F}{\partial \dot{x}}(\cdot)$ in (3.1) is singular. Given an IVP arising from a DAE, it is useful to distinguish between differential and algebraic states. For the differential states an initial value x_0 is provided, whereas the algebraic variables can usually be expressed as functions of the differential states and controls, hence, they do not need an initial value. DAEs are formally defined as follows.

Definition 3.4 (Differential Algebraic Equations)**.** *Let $t \in \mathbb{R}$ be the time, $x(t) \in \mathbb{R}^{n_x}$ the differential states, $z(t) \in \mathbb{R}^{n_x}$ the algebraic states and $u(t)\mathbb{R}^{n_u}$ the control function. Given $F : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ the system of equations:*

$$F(t, \dot{x}(t), x(t), z(t), u(t)) = 0. \tag{3.4}$$

is called a differential algebraic equation.

If we could find an explicit expression for z as a function of x and u we could transform the DAE into an ODE. It turns out that sometimes one has to differentiate the equation (3.4) several times w.r.t. time until z can be extracted,

e.g., by algebraic manipulations. As we will see later, the number of times that we need to differentiate does not only determine the properties of the solutions but also the quality of numerical approximations obtained by standard integration methods applied to the DAE. This leads to the following definition that serves as a handy criterion for classifying DAEs.

Definition 3.5 (Differential index [125])**.** *The DAE* (3.4) *has the differential index m, if m is the minimal number of analytic differentiations such that the set of differential equations*

$$\frac{\mathrm{d}^m}{\mathrm{d}t^m} F(\dot{x}(t), x(t), z(t), u(t)) = 0,$$

corresponds to an ODE as defined in Definition 3.1.

Semi-explicit DAEs

The DAE (3.4) is fully implicit. In practice, one encounters more often so-called semi-explicit DAEs. For example, the smooth pieces of solution trajectories of nonsmooth systems studied in this thesis are often described by semi-explicit DAEs. We will focus on them in the sequel and provide their canonical form for indexes one to three.

Given the functions $f : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ and $g : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_z}$, such that the Jacobian $\frac{\partial g}{\partial z}$ is nonsingular, the system of equations:

$$\dot{x}(t) = f(t, x(t), z(t), u(t)), \tag{3.5a}$$

$$0 = g(t, x(t), z(t), u(t)), \tag{3.5b}$$

is called a semi-explicit DAE of index 1. The invertibility of the Jacobian $\frac{\partial g}{\partial z}(\cdot)$ guarantees via the implicit function theorem the existence of a locally single-valued $z(x)$, hence, an ODE is easily obtained.

Given the functions $f : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ and $g : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_z}$, such that the Jacobian $\frac{\partial g}{\partial x} \frac{\partial f}{\partial z}$ is nonsingular, the system of equations:

$$\dot{x}(t) = f(t, x(t), z(t), u(t)), \tag{3.6a}$$

$$0 = g(t, x(t), u(t)), \tag{3.6b}$$

is called a semi-explicit DAE of index 2 [125]. In the study of so-called sliding modes of nonsmooth dynamical systems we often locally encounter DAEs of index 2, cf. Chapters 4 and 6.

Suppose now that we have the differential states $x \in \mathbb{R}^{n_x}$ and $y \in \mathbb{R}^{n_y}$ and the algebraic state $z \in \mathbb{R}^{n_z}$. Given the functions $f_x : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_y} \to \mathbb{R}^{n_x}$, $f_y : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_y} \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_y}$, and $g : \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_z}$, such that the Jacobian $\frac{\partial g}{\partial x} \frac{\partial f_x}{\partial y} \frac{\partial f_y}{\partial z}$ is nonsingular, the system of equations:

$$\dot{x}(t) = f_x(t, x(t), y(t)), \tag{3.7a}$$

$$\dot{y}(t) = f_y(t, x(t), y(t), z(t), u(t)), \tag{3.7b}$$

$$0 = g(t, x(t), u(t)), \tag{3.7c}$$

is called a semi-explicit DAE of index 3. The Euler-Lagrange equations for constrained multi-body systems result in DAEs of index 3 [125]. In mechanical systems with state jumps studied in Chapter 8, DAEs of this kind play a central role.

Note that a higher-index DAE can usually be replaced by a DAE of a lower index paired with consistency conditions, i.e., the initial value must satisfy the algebraic equations that are replaced by equations obtained by differentiating them by time. This procedure is often used for theoretical and computational considerations and is called *index reduction*. The existence and uniqueness results for ODEs can be extended to DAEs [230].

3.2 Numerical integration methods

Even though we know that a solution to the IVP associated with ODE (3.2) or the DAE (3.5) exists, we can compute it analytically only in some special cases. Therefore, one is usually forced to find approximations of the true solution at discrete points in time. A continuous time approximation can be constructed based on the computed values at these points. The computation of a numerical approximation is called a simulation of the differential equation, or numerical integration. The excellent textbooks [18, 47, 122, 125] provide a broad and detailed overview of the field.

More precisely, we regard the integration interval $[0, T]$ and compute solution approximations $x_n \approx x(t_n)$ at N discrete time-points $0 = t_0 < t_1 < \ldots < t_N = T$. Thereby the integration interval is split into the integration subintervals $[t_n, t_{n+1}]$ where $t_{n+1} = t_n + h_n$ and h_n is called the integration step size. In this section, for ease of notation we assume a constant integration step size, i.e., $h_n = h = \frac{T}{N}$ for all $n = 0, \ldots, N-1$. Moreover, we assume the control to be known and for simplicity, we set $u(t) = u_0$ for $t \in [0, T]$. The extensions to more sophisticated control parametrizations are straightforward.

Standard integration methods rely on linear combinations of previously computed values and function evaluations $f(\cdot)$. There are several distinguishing features of standard integration methods. If the methods use several previously computed values x_n, x_{n-1}, ..., then we speak of *multi-step methods*. These methods require a reliable starting procedure for computing the first few points [124]. On the other hand, *single-step* or *one-step* methods only use the previous value x_n, but evaluate the function $f(\cdot)$ at *stage points* inside the integration subintervals to achieve better accuracy. Furthermore, both method classes can be explicit or implicit. Explicit methods rely only on function evaluations to compute the next step, whereas implicit methods require the solution of a root-finding problem. The latter is more computationally expensive per step but is usually more accurate for given number of stage points and has better stability properties and allows larger step sizes.

In this thesis, we focus on one-step methods and their most famous example, namely the Runge-Kutta (RK) methods. Runge-Kutta methods can be applied generically to ODEs and even high-index DAEs. Note that for special classes of smooth ODEs such as Hamiltonian systems [126, 123] or highly oscillatory systems [130, 123] tailored integration methods exist.

Before we proceed with some basic definitions for one-step methods, we want to highlight that in numerical optimal control, it is often instructive to think of any integration method as a discrete-time dynamical system. In this sense, given $x_0 = \bar{x}_0$, a one-step integration method that recursively generates a sequence of approximations x_n for $x(t_n)$ is compactly represented by

$$x_{n+1} = \phi_f(x_n, z_n, u_0), \tag{3.8a}$$

$$0 = \phi_{\text{int}}(x_n, z_n, u_0). \tag{3.8b}$$

Here z_n collects all internal variables of the integration method, including the discrete-time approximations of the algebraic variables. The function $\phi_f(\cdot)$ is the *state-transition map* and Eq. (3.8b) collects all internal equations of the method. Depending on the underlying IVP and the chosen integration method, the function $\phi_{\text{int}}(\cdot)$ determines the internal variables z_n either explicitly or implicitly. Several examples of such methods are given below. We will use the discrete-time system representation (3.8) for all discretization methods in this thesis, for smooth and nonsmooth systems as well. We proceed with defining the notions of convergence and accuracy for one-step methods.

Definition 3.6 (Numerical integration error)**.** *Let $x(t)$ be the exact solution to the IVP from Definition 3.2 on $[0, T]$, with the initial value $x(0) = x_0$. Regard the one-step method* (3.8) *for computing an approximation x_n of $x(t)$ at time t_n, with the given control $u(t) = u_0$ and internal integration methods variables*

z_{n-1}. *The local integration error at the time points* t_n *is defined as*

$$e(t_n) = \|x(t_n) - \phi_f(x(t_{n-1}), z_{n-1}, u_0)\|.$$

The global or transported integration error at t_n *is defined as*

$$E(t_n) = \|x(t_n) - x_n\|,$$

where x_n *is computed recursively using* (3.8) *starting with* $x_0 = \bar{x}_0$.

The local error tells us about the mismatch of a single integration step. The consistency measures the difference between the true slope and the approximate slope arising from the integration method. Finally, the global error measures the overall accumulated error during the simulation.

Definition 3.7 (Order of accuracy/convergence)**.** *A numerical integration method is said to be convergent, if all its points* x_n *converge to the exact solution* $x(t_n)$ *for* $h \to 0$*, that is* $\lim_{h\to 0} \sup_{t_n} \|E(t_n)\| = 0$*. The method is said to have order* $p > 0$ *if the local errors satisfy*

$$\lim_{h\to 0} \sup_{t_n} \|e(t_n)\| = O(h^{p+1}). \tag{3.9}$$

The errors and orders of convergence for the algebraic variables z are defined accordingly.

3.2.1 Runge-Kutta methods

Runge-Kutta (RK) methods are the most famous class of one-step methods. They are the starting point for tailored numerical methods for nonsmooth differential equations that are developed in Chapter 7. We first define them for explicit ODEs as in (3.2) in two different variants and give afterwards extensions to the case of semi-explicit DAEs (3.5).

Definition 3.8 (Runge-Kutta method in differential form)**.** *Consider the IVP from Definition 3.2. Let* n_s *be a positive integer, called the number of stages. Given the matrix* $A \in \mathbb{R}^{n_\mathrm{s}\times n_\mathrm{s}}$ *with the entries* $a_{i,j}$ *for* $i, j \in \{1, \dots, n_\mathrm{s}\}$*, and the vectors* $b, c \in \mathbb{R}^{n_\mathrm{s}}$*. The system of equations:*

$$k_{n,i} = f(t_n + c_i h, x_n + h \sum_{j=1}^{n_\mathrm{s}} a_{i,j} k_{n,j}, u_0), \quad \text{for all } i \in \{1, \dots, n_s\}, \tag{3.10a}$$

$$x_{n+1} = x_n + h \sum_{i=1}^{n_\mathrm{s}} b_i k_{n,i}, \tag{3.10b}$$

is called a n_s*-stage Runge-Kutta (RK) method in the differential form.*

The coefficients c_i specify the *stage points* $t_n + c_i h$, while $a_{i,j}$ are the internal coefficients and b_i are the weights. To have a useful RK method, the coefficients $a_{i,j}, b_i$ and c_i have to satisfy certain properties [124]. To have a method that converges it must hold that

$$\sum_{i=1}^{n_\mathrm{s}} b_i = 1.$$

The internal coefficients $a_{i,j}$ should satisfy the consistency conditions, i.e.,

$$\sum_{j=1}^{n_\mathrm{s}} a_{i,j} = c_i.$$

Some additional conditions lead to well-defined problems whose solutions provide coefficients for concrete RK methods [56, 122, 124]. To relate Eq. (3.10) to the general form in (3.8), note that the internal variables in (3.10) are $z_n = (k_{n,1}, \dots k_{n,n_\mathrm{s}}) \in \mathbb{R}^{n_\mathrm{s} n_x}$. The equations (3.10a) are the internal computations summarized in $\phi_{\mathrm{int}}(x_n, z_n, u_0) = 0$ and the state transition map reads as $\phi_f(x_n, z_n, u_0) = x_n + h \sum_{i=1}^{n_\mathrm{s}} b_i k_{n,i}$.

The formulation in Definition 3.8 is called *differential* since the derivatives of the states at stage points k_i are the unknowns. Alternatively, the values of the states at the stage points $x_{n,i} \in \mathbb{R}^{n_x}$ can be the unknowns, which brings us to the integral form.

Definition 3.9 (Runge-Kutta method in integral form)**.** *Consider the IVP from Definition 3.2. Let n_s be a positive integer, called the number of stages. Given the matrix $A \in \mathbb{R}^{n_\mathrm{s} \times n_\mathrm{s}}$ with the entries $a_{i,j}$ for $i, j \in \{1, \dots, n_\mathrm{s}\}$, and the vectors $b, c \in \mathbb{R}^{n_\mathrm{s}}$. The system of equations:*

$$x_{n,i} = x_n + h \sum_{j=1}^{n_\mathrm{s}} a_{i,j} f(t_n + c_j h, x_{n,j}, u_0), \quad \textit{for all } i \in \{1, \dots, n_s\} \tag{3.11a}$$

$$x_{n+1} = x_n + h \sum_{i=1}^{n_\mathrm{s}} b_i f(t_n + c_i h, x_{n,i}, u_0), \tag{3.11b}$$

is called a n_s-stage Runge-Kutta (RK) method in integral form.

The extension of RK methods to nonsmooth ODEs developed later in Chapter 7 is implemented both in integral and differential form in the open-source package `nosnoc`.

It is very common to characterize RK methods by their Butcher tableau [55]:

$$\begin{array}{c|cccc} c_1 & a_{1,1} & a_{1,2} & \dots & a_{1,n_\mathrm{s}} \\ c_2 & a_{2,1} & a_{2,2} & \dots & a_{2,n_\mathrm{s}} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ c_{n_\mathrm{s}} & a_{n_\mathrm{s},1} & a_{n_\mathrm{s},2} & \dots & a_{n_\mathrm{s},n_\mathrm{s}} \\ \hline & b_1 & b_2 & \dots & b_{n_\mathrm{s}} \end{array} \qquad \begin{array}{c|c} c & A \\ \hline & b^\top \end{array}$$

The sparsity pattern of the matrix A determines if the method is explicit or implicit. For Explicit RK (ERK) methods only the entries below the diagonal of A are nonzero, i.e., A is strictly lower triangular. For $n_\mathrm{s} \leq 4$, the number of stages in the best ERK methods matches their order. Beyond that, the number of stages grows faster than the order. ERK methods are usually good for nonstiff problems as they are computationally cheap and provide good accuracy.

If A is not strictly lower triangular we speak of an Implicit RK (IRK) method. A subclass of IRK methods are semi-implicit RK methods. In this case, the matrix A has a sparsity pattern that can be computationally exploited, see [226, Chapter 2] and [195, 122] for details. For computing an integration step with IRK methods, Newton's method must be employed for obtaining the value k_i to compute x_{n+1}. Note that since already a root-finding algorithm is used, IRK methods are also suitable for implicit ODEs and DAE formulations from Definitions 3.1 and 3.4, respectively.

Implicit methods are computationally more expensive per step, but they have in general a higher accuracy order for the same number of stages. Nevertheless, they can be more efficient depending on the required accuracy. They have also superior properties when it comes to the simulation of DAEs and stiff ODEs [122]. Informally speaking, stiff problems are differential equations for which explicit methods do not work well [126]. They are differential equations that have largely different time scales. For more details on stiff differential equations and definitions, cf. [56, Section 112] and [126]. Closely related to the notion of stiffness is the notion of numerical stability of integration methods. Intuitively, if the analytic solution is bounded the numerical solution of a stable method should also be bounded for a reasonable step size. Formally, a few different stability concepts exist, e.g. A-stability, L-stability, cf. [56, 126].

The existence of solutions to IVPs does not automatically imply the existence of solutions to its discrete-time counterpart obtained via the IRK equations. Fortunately, only mild additional conditions are required. In general, besides the smoothness assumptions on the function $f(\cdot)$, sufficiently small step sizes h are sufficient to guarantee invertiblity of the Jacobian of the residual function in the IRK equations $\frac{\partial \phi_\mathrm{int}}{\partial (x,z)}(\cdot)$. More on this topic can be found in [126, Section 14].

Collocation methods

An interesting subclass of IRK methods often used in practice and also in this thesis are collocation methods. The main idea is to approximate the solution $x(t)$ of the IVP (3.3) on every interval $[t_n, t_{n+1}]$ by a polynomial $q_n(t)$ of degree n_s. The polynomial satisfies the differential equations at the *collocation points* (i.e., stage points) and matches the initial value at $t = t_n$. This is summarized in the next definition.

Definition 3.10 (Collocation methods)**.** *Consider the IVP from Definition 3.2. Let n_s be a positive integer and $c_1, \dots, c_{n_\mathrm{s}} \in [0,1]$ distinct numbers. The corresponding collocation polynomial $q_n(t)$ of degree n_s is implicitly defined by*

$$q_n(t_n) = x_n,$$

$$\dot{q}_n(t_n + c_i h) = f(t_n + c_i, q_n(t_n + c_i h), u_0), \text{ for all } i \in \{1, \dots, n_\mathrm{s}\}.$$

The numerical approximation x_{n+1} is given by $x_{n+1} = q_n(t_{n+1})$. The time points $t_n + c_i h$ are called the collocation points.

The choice of the collocation points c_i uniquely determines the IRK method and its properties. Note that in contrast to general RK methods, c_i must be distinct numbers. Moreover, the collocation polynomials $q_n(t)$ provide a continuous time approximation of $x(t)$ on the interval $[t_n, t_{n+1}]$. The accuracy order inside the interval is at least n_s and less or equal to the accuracy order p of the method (defined for the endpoint t_{n+1}). In general, the accuracy order of collocation methods at grid points t_n is $n_\mathrm{s} \leq p \leq 2n_\mathrm{s}$ [126].

There are several mathematically equivalent ways to represent the polynomial $q_n(t)$. In practical implementations, it is common to use an interpolating polynomial through the initial value and state values at the collocation points [38]. In this case, it is necessary that $c_1 \neq 0$ (which is e.g., violated for Lobatto collocation). Another similar, but more general and less restrictive representation, which also simplifies the relation to IRK methods is commonly used in the exposition of collocation methods [122, 126, 229]. Thereby, $q_n(t)$ is parametrized by x_n and the state derivatives at the collocation points:

$$q_n(t) = x_n + \int_{t_n}^{t} \dot{q}_n(\tau) \mathrm{d}\tau.$$

The polynomial $\dot{q}_n(t)$ of degree $n_\mathrm{s} - 1$ is expressed using the Lagrange interpolating polynomials $\ell_i(t)$:

$$\dot{q}_n(t) = \sum_{i=1}^{n_\mathrm{s}} \ell_i\Big(\frac{t - t_n}{h}\Big) \underbrace{f(t_n + c_i, q_n(t_n + c_i h), u_0)}_{=k_{n,i}} = \sum_{i=1}^{n_\mathrm{s}} \ell_i\Big(\frac{t - t_n}{h}\Big) k_{n,i},$$

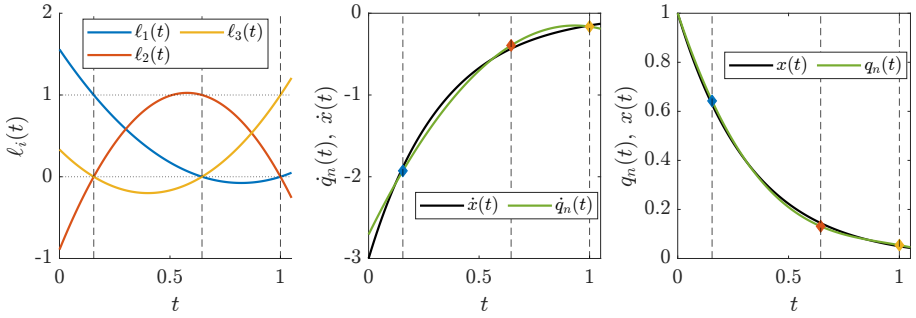


Figure 3.1: Illustration of the Lagrange interpolating polynomials $\ell_i(t)$ with $n_\mathrm{s} = 3$ (left plot) The functions $\dot{q}_n(t)$ and $\dot{x}(t)$ (middle plot). The functions $q_n(t)$ and $x(t)$ (right plot). The vertical dashed lines correspond to the stage points c_i.

where the polynomials ℓ_i are defined as

$$\ell_i(\tau) = \prod_{j=1, i\neq j}^{n_\mathrm{s}} \frac{\tau - c_j}{c_i - c_j}.$$

By direct computation and comparing to (3.10), the missing coefficient for the Butcher tableau is computed by:

$$a_{i,j} = \int_0^{c_i} \ell_j(t)\mathrm{d}t, \quad b_i = \int_0^1 \ell_i(t)\mathrm{d}t, \text{ for all } i, j \in 1, \ldots, n_\mathrm{s}.$$

Two very popular classes of collocation/IRK methods are the Gauss-Legendre (GL) and Radau IIA methods. GL methods are optimal in terms of accuracy, i.e., they have the highest possible accuracy order $p = 2n_\mathrm{s}$ for a given number of stages n_s and they are A-stable [126]. On the other hand, Radau IIA methods have an accuracy order of $p = 2n_\mathrm{s} - 1$ and are L-stable. L-stability is a desirable property when one needs to integrate very stiff systems [126]. Moreover, for Radau IIA methods it holds that $c_{n_\mathrm{s}} = 1$. This property is favorable for tailored methods for nonsmooth Filippov systems, as it simplifies their implementation and theoretical analysis, cf. Chapter 7. Another interesting family of collocation methods with $c_{n_\mathrm{s}} = 1$ are Lobatto methods, cf. [126].

Example 3.11. *The derivation of the collocation method is illustrated for the example* $\dot{x}(t) = -3x$, $x(0) = 1$ *and* $T = 1$. *The analytic solution to this IVP is* $x(t) = e^{-3t}$. *We use a Radau IIA method with* $n_\mathrm{s} = 3$ *and take a single integration step with* $h = 1$. *The stage points are and* $c_1 = \frac{4-\sqrt{6}}{10}$, $c_2 = \frac{4+\sqrt{6}}{10}$ *and* $c_3 = 1$. *Figure 3.1 illustrates the Lagrange polynomials (left plot), the*

polynomial $\dot{q}_n(t)$ compared to the time derivative of the exact solution $\dot{x}(t)$ (middle plot) and the polynomial $q_n(t)$ compared to the exact solution $x(t)$ (right plot). It can be seen that the error inside the interval (t_n, t_{n+1}) is greater than on the boundary for $t = t_{n+1}$.

Runge-Kutta methods for DAEs

We turn our attention now to RK methods for semi-explicit DAEs. The extension is conceptually straightforward but their theoretical properties for DAE of different indexes are heterogeneous. Even though one can apply ERK methods to DAE, the DAEs are often stiff. Therefore, it is more natural to consider IRK methods. They also have superior theoretical properties for DAEs compared to ERK. Moreover, in this thesis, direct transcription methods are often used for discretizing optimal control problems, cf. Section 3.3.2. This means that the IRK equations end up being constraints of an NLP, which is anyway numerically solved with a Newton-type method.

The IRK equations for the semi-explicit DAE of index 1 (3.5) read as

$$k_{n,i} = f(t_n + c_i h, x_n + h \sum_{j=1}^{n_\mathrm{s}} a_{i,j} k_{n,j}, z_{n,i}, u_0),\ i \in \{1, \dots, n_\mathrm{s}\}, \tag{3.12a}$$

$$0 = g(t_n + c_i h, x_n + h \sum_{j=1}^{n_\mathrm{s}} a_{i,j} k_{n,j}, z_{n,i}, u_0),\ i \in \{1, \dots, n_\mathrm{s}\}, \tag{3.12b}$$

$$x_{n+1} = x_n + h \sum_{i=1}^{n_\mathrm{s}} b_i k_{n,i}, \tag{3.12c}$$

$$0 = g(t_{n+1}, x_{n+1}, z_{n+1}, u_0). \tag{3.12d}$$

Here $z_{n,i},\ i \in \{1, \dots, n_\mathrm{s}\}$ are the stage values for the algebraic variables and z_{n+1} is the approximation of $z(t_{n+1})$. To obtain the latter, the equation (3.12d) is added. For IRK methods $c_{n_\mathrm{s}} = 1$ it is automatically satisfied. All standard results about order and convergence for IRK methods hold also for the formulation (3.12) [125]. The RK equations (3.12) are stated in their differential form. The integral form is obtained similar to (3.11) by having instead of $k_{n,i}$ the stage values $x_{n,i}$ as degrees of freedoms. Note that in both forms the algebraic variables are kept in *integral form*, i.e., the unknowns are $z_{n,i}$.

IRK methods for index-2 DAEs (3.6) are formulated similarly. It is assumed that the initial values x_n and z_n are consistent, i.e.,

$$g(t_n, x_n, u_0) = 0,$$

$$\nabla_x g(t_n, x_n, u_0)^\top f(t_n, x_n, z_n, u_0) = 0.$$

The IRK equations read as

$$k_{n,i} = f(t_n + c_i h, x_n + h \sum_{j=1}^{n_\mathrm{s}} a_{i,j} k_{n,j}, z_{n,i}, u_0),\ i \in \{1, \dots, n_\mathrm{s}\}, \tag{3.13a}$$

$$0 = g(t_n + c_i h, x_n + h \sum_{j=1}^{n_\mathrm{s}} a_{i,j} k_{n,j}, u_0),\ i \in \{1, \dots, n_\mathrm{s}\}, \tag{3.13b}$$

$$x_{n+1} = x_n + h \sum_{i=1}^{n_\mathrm{s}} b_i k_{n,i}. \tag{3.13c}$$

Note that for methods with $c_{n_\mathrm{s}} \neq 1$ the boundary value z_{n+1} cannot be computed as simply as in the previous case. To compute this value we need a fully differential representation. Thereby, the algebraic stage value in (3.13) are replaced by

$$z_{n,i} = z_n + h \sum_{i=1}^{n_\mathrm{s}} a_{i,j} k^z_{n,i},\ i \in \{1, \dots, n_\mathrm{s}\},$$

where $k^z_{n,i}$ are the derivatives of the algebraic variables at the stage points. Then, z_{n+1} is computed via

$$z_{n+1} = z_n + h \sum_{i=1}^{n_\mathrm{s}} b_i k^z_{n,i}.$$

The IRK equations for the DAE of index 3 in Eq. (3.7) are similar to (3.13). We omit the full statement for brevity here, but details can be found in [122, Chapter 6].

Summary for Runge-Kutta methods

Fixed step size Runge-Kutta methods are very popular in numerical optimal control for several reasons [226]. They are fairly easy to implement, the computations have a bounded and deterministic runtime and they have good theoretical properties (high accuracy, can handle stiff problems, etc.). Lobatto and Radau methods for DAEs, which have the right boundary point as a stage point are sometimes easier to implement. As we will see in the next chapter, their straightforward generalizations to nonsmooth systems lose many of the beneficial properties, e.g., accuracy order and convergence of sensitivities.

However, they still serve as a basis for tailored methods where these properties can be recovered. We provide in Table 3.1 an overview of accuracy orders for the IRK methods used in this thesis.

3.2.2 Sensitivity computation

In Newton-type optimization, we need to compute first and often second-order derivatives. More concretely, in numerical optimal control, we need the derivatives of IVP solution approximations w.r.t. to the initial value and controls. These derivatives are called *(numerical) sensitivities*. We discuss here briefly how to compute first-order sensitivities. An excellent reference that treats numerical sensitivity propagation in great detail is the PhD thesis of R. Quirynen [226]. Details on higher-order sensitivities can be found in [7, 226].

Let us regard the IVP from Definition (3.2) on the time interval $[0, T]$ with the initial value $x(0) = \bar{x}_0$. We assume a constant control input u over this interval. This corresponds to a single control interval in a discretized OCP with a piecewise constant control parametrization, which is a common choice in direct optimal control, cf. Section 3.3. Next, denote the solution (or its approximation) by $x(t; x_0, u)$. We are interested in the sensitivities at $t = T$ w.r.t x_0 and u, i.e., the partial derivatives $\frac{\partial x(T;x_0,u)}{\partial x_0} \in \mathbb{R}^{n_x \times n_x}$, $\frac{\partial x(T;x_0,u)}{\partial u} \in \mathbb{R}^{n_x \times n_u}$. Additionally, one could be interested in the case of DAEs in the sensitivities of the algebraic variables, or in adjoint sensitivities. We omit these cases here for brevity and refer to [226, Chapter 2].

Nowadays derivatives of differentiable functions are usually computed via automatic differentiation. Thereby, an expression graph for the function

Method	n_s	ODE	DAE index 1		DAE index 2	
		x	x	z	x	z
Gauss-Legendre	odd	$2n_\mathrm{s}$	$2n_\mathrm{s}$	n_s	$n_\mathrm{s}+1$	$n_\mathrm{s}-1$
	even	$2n_\mathrm{s}$	$2n_\mathrm{s}$	$n_\mathrm{s}+1$	n_s	$n_\mathrm{s}-2$
Radau IA	odd/even	$2n_\mathrm{s}-1$	$2n_\mathrm{s}-1$	n_s	n_s	$n_\mathrm{s}-1$
Radau IIA	odd/even	$2n_\mathrm{s}-1$	$2n_\mathrm{s}-1$	$2n_\mathrm{s}-1$	$2n_\mathrm{s}-1$	n_s
Lobatto IIIA	odd	$2n_\mathrm{s}-2$	$2n_\mathrm{s}-2$	$2n_\mathrm{s}-2$	$2n_\mathrm{s}-2$	$n_\mathrm{s}-1$
	even	$2n_\mathrm{s}-2$	$2n_\mathrm{s}-2$	$2n_\mathrm{s}-2$	$2n_\mathrm{s}-2$	n_s
Lobatto IIIC	odd/even	$2n_\mathrm{s}-2$	$2n_\mathrm{s}-2$	$2n_\mathrm{s}-2$	$2n_\mathrm{s}-2$	$n_\mathrm{s}-1$

Table 3.1: Overview of accuracy orders for some IRK methods for ODEs, DAEs of index 1 and 2 [122].

is created and by applying the chain rule, every expression in the graph is differentiated. For more details see e.g., [115]. Efficient and accurate computation of derivatives of solution approximations to IVPs is slightly more involved and requires some care. In the following we discuss in more detail the two main strategies for computing sensitivities:

1. differentiate-then-discretize,
2. discretize-then-differentiate.

Differentiate-then-discretize

This approach consists of augmenting the IVP problem by two additional matrix differential equations, which are linear in the sensitivities and which are called Variational Differential Equations (VDEs) [62]. The *forward* sensitivities are obtained as solutions of:

$$\dot{S}^x(t) = \frac{\partial f}{\partial x}(t, x(t), u(t))S^x(t),\ S^x(0) = I_{n_x}, \tag{3.14a}$$

$$\dot{S}^u(t) = \frac{\partial f}{\partial x}(t, x(t), u(t))S^u(t) + \frac{\partial f}{\partial u}(t, x(t), u(t)),\ S^u(0) = \mathbf{0}, \tag{3.14b}$$

where $S^x(t) \coloneqq \frac{\partial x(t)}{\partial x_0} \in \mathbb{R}^{n_x \times n_x}$ and $S^u(t) \coloneqq \frac{\partial x(t)}{\partial u} \in \mathbb{R}^{n_x \times n_u}$. This approach consists of a continuous-time propagation of the sensitivities, which can be discretized with any accuracy. However, in dynamic optimization, we need accurate sensitivities of the solution approximations, but not necessarily of the analytic solution itself. If no care is taken, it is clear that $S^x(T)$ is not necessarily identical to the derivative of the solution approximation $\frac{\partial x(T;x_0,u)}{\partial x_0}$. The same holds for the derivative w.r.t. u. One can also compute *backward* or adjoint sensitivities via a similar system of equations to (3.14) [61]. This system is then simulated backward in time.

Discretize-then-differentiate

Guided by the fact that the derivatives of the solution approximation are needed, one could differentiate the integration method itself. Complementary to the previous approach, this can be interpreted as a discrete-time sensitivity propagation and there are a few approaches to doing these computations. The simplest way from a implementation perspective is to use finite differences. We remind the reader that $x(t; x_0, u)$ is the numerical solution approximation of

the IVPs. We can regard this function as a black box and call it as often as needed. A partial derivative w.r.t. $x_{0,i}$ at $t = T$ is computed by

$$\frac{\partial x(T; x_0, u)}{\partial x_{0,i}} \approx \frac{x(T; x_0 + h\delta_i, u) - x(T; x_0, u)}{h},$$

where $\delta_i \in \mathbb{R}^{n_x}$ is a vector which has a one at the i-th entry and zeros otherwise. In the literature, this approach is known as *external numerical differentiation* [7, 229]. Note that if an adaptive step size integrator is used, the values $x(T; x_0 + h\delta_i, u)$ and $x(T; x_0, u)$ might be coming from two different discretized functions. To avoid interference with the step size control that causes such discontinuities, one has usually to sacrifice accuracy and use a larger h [7]. To avoid such pitfalls Bock has proposed the principle of *internal numerical differentiation* [39, 41]. Thereby, the nominal trajectory for x_0 is computed adaptively, and for all other function calls the adaptive parts of the method (step size, number of Newton iterations, etc.) are frozen. This results in a differentiable integrator code.

Alternatively, instead of using finite differences, one can apply Automatic Differentiation (AD) to the resulting integrator code. Thereby, one can apply AD either to the whole code (known as external AD) or apply AD independently to each part of the integrator (known as internal AD), see e.g., [226] for details. The latter is usually more efficient in terms of runtime as some problem structure can be exploited [7, 226].

In the end, one obtains a code that can be interpreted as a discrete time VDE. We write this discrete time system in a similar style as it is done for integration methods in Eq. (3.8):

$$S^x_{n+1} = \psi^x_{S^x}(x_n, S^x_n, Y_n, u),\ 0 = \psi_{x,\mathrm{int}}(x_n, S^x_n, Y_n, u), \tag{3.15a}$$

$$S^u_{n+1} = \psi_{S^u}(x_n, S^u_n, Y_n, u),\ 0 = \psi_{u,\mathrm{int}}(x_n, S^u_n, Y_n, u), \tag{3.15b}$$

where Y_n are internal variables, $S^x_n = \frac{\partial x_n}{\partial x_0}$ and $S^u_n = \frac{\partial x_n}{\partial u}$. Similar to Eq. (3.8), the functions $\psi_{x,\mathrm{int}}(\cdot)$ and $\psi_{u,\mathrm{int}}(\cdot)$ collect all internal computations. Clearly, in this case, S^x_N is the exact derivative of the solution approximation and is thus equal to $\frac{\partial x(T;x_0,u)}{\partial x_0}$. Of course, the same is true for the derivatives w.r.t. u. Interestingly, the differentiated equations of an explicit RK method with a fixed step size are equivalent to the equations obtained by applying the same RK method to the VDE equations (3.14) [282].

3.3 Numerical optimal control

Many of the ideas that we review in the following are in some way applicable to the more difficult case of nonsmooth optimal control problems. Thus it is

necessary to be familiar with them. For ease of exposition let us regard an explicit and sufficiently smooth ODE, e.g., as introduced in Section 3.1. An optimal control problem is the optimization problem:

$$\min_{x(\cdot),u(\cdot)} \quad \int_0^T L(x(t),u(t))\,\mathrm{d}t + M(x(T)) \tag{3.16a}$$

$$\text{s.t.} \quad x(0) = \bar{x}_0, \tag{3.16b}$$

$$\dot{x}(t) = f(t,x(t),u(t)), \quad \text{for almost all } t \in [0,T], \tag{3.16c}$$

$$0 \leq g_\mathrm{p}(t,x(t),u(t)), \quad \text{for almost all } t \in [0,T], \tag{3.16d}$$

$$0 \leq g_\mathrm{t}(x(T)), \tag{3.16e}$$

This is an infinite-dimensional optimization problem since the decision variables $x(\cdot)$ and $u(\cdot)$ are infinite-dimensional. The functions $g_\mathrm{p} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_{g_\mathrm{p}}}$ and $g_\mathrm{t} : \mathbb{R}^{n_x} \to \mathbb{R}^{n_{g_\mathrm{t}}}$ define the path and terminal constraints, respectively. We have also infinitely many constraints as (3.16c) and (3.16d) must be satisfied for all $t \in [0,T]$. This interval is called the control horizon and we assume for now $T > 0$ to be fixed. The first term in the objective (3.16a), defined by the function $L : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ are the stage/running costs. They are also known as the Lagrange term. The function $M : \mathbb{R}^{n_x} \to \mathbb{R}$ defines the terminal cost, which is sometimes called the Mayer term. Finally, $\bar{x}_0 \in \mathbb{R}^{n_x}$ is the initial value that we assume to be fixed, but it can also be a degree of freedom.

The integral term in the objective (3.16a) is often replaced by a terminal cost term $x_\mathrm{q}(T)$ of a quadrature state $x_\mathrm{q}(t) \in \mathbb{R}$. The dynamics of the quadrature state reads as:

$$\dot{x}_\mathrm{q}(t) = L(x(t),u(t)),\ x_\mathrm{q}(0) = 0.$$

This simple state augmentation enables e.g., to easily integrate the objective and the dynamics with the same integrator code.

Time transformations and numerical time

In time optimal control problems the goal is to minimize the final time T. An OCP formulation where this is possible is obtained by re-scaling the dynamics in (3.16c) and letting T be a degree of freedom. Instead of regarding the IVP over the time interval $[0,T]$ we regard a re-scaled system over $\tau \in [0,1]$. We call throughout this thesis t the *physical time*, and τ the *numerical time*. Time derivatives w.r.t. the numerical time τ are compactly denoted by $\frac{\mathrm{d}x(\tau)}{\mathrm{d}\tau} = x'(\tau)$.

The physical and numerical time are related via $t = \tau T$ or:

$$\frac{\mathrm{d}t(\tau)}{\mathrm{d}\tau} = T. \tag{3.17}$$

Therefore, we have

$$\frac{\mathrm{d}x(\tau)}{\mathrm{d}\tau} = \frac{\mathrm{d}x}{\mathrm{d}t}\frac{\mathrm{d}t}{\mathrm{d}\tau},$$

and the re-scaled dynamics read as

$$x'(\tau) = Tf(\tau, x(\tau), u(\tau)), \tau \in [0, 1]. \tag{3.18}$$

Remark 3.12. *The time transformation is also applicable in multi-stage OCPs. For example, in dynamic walking problems [240], given a fixed order of contact configurations of a walking biped, one can optimize the length of every step to generate an optimal walking or running motion.*

Some of the core ideas of this thesis will leverage this separation into numerical and physical time. It is useful to think of T in (3.18) as a *speed-of-time* variable. The numerical time τ runs with a speed of one and the physical time with a speed of T. In this setting, a trivial consideration is to set the speed of both times to one and regard $\tau \in [0, T]$, i.e., numerical and physical time are equal, which matches the usual case without time transformations. In the treatment of nonsmooth systems, we will sometimes stop the evolution of physical time and make the r.h.s. in (3.17) discontinuous. This technique is part of time-freezing and it allows one to transform nonsmooth systems with discontinuous solutions in t into systems with continuous solutions in τ, cf. Chapters 8 and 9.

3.3.1 Solution methods for OCPs

In general, there is no closed-form solution for the OCP (3.16). Consequently, the OCP is in practice solved numerically, which means that at some point a finite-dimensional problem needs to be solved. Numerical methods for solving the OCP (3.16) are divided into three large groups:

1. dynamic programming methods,
2. indirect methods,
3. direct methods.

It is well-known that dynamic programming methods suffer from the *curse of dimensionality*. Indirect methods are the most accurate one of the three, however they require solving a difficult two-point boundary problem, which is in practice often prohibitively difficult [229]. On the other hand, direct methods are based on the mature standard time-discretization and nonlinear optimization methods. They allow for simple initialization of the problem and provide reliably locally optimal solutions with good accuracy. Therefore, we mainly focus on direct methods in this thesis and thus we discuss the first two only briefly. A comparison of all approaches can be found e.g., in [36, 80].

Dynamic programming methods are based on Bellman's principle of optimality [29]. They propagate the optimal cost-to-go function backward in time. To obtain a solution of the OCP (3.16) in continuous time one has to solve the Hamilton-Jacobi-Bellman partial differential equation [184]. In discrete time, a dynamic programming recursion is regarded [35]. In contrast to the other two, this approach always finds the global minimizer. However, this requires a tabulation of the cost-to-go function in a discretized space, which quickly becomes computationally intractable for state dimensions more than three to five.

Indirect methods are mostly based on Pontryagin's maximum principle [224], which gives first-order necessary optimality conditions for the OCP (3.16) in continuous-time. These optimality conditions are a boundary-value problem that is solved numerically. Hence, this approach is often called *first-optimize-then-discretize*. Indirect methods provide general solution approximations of high accuracy but require a good initial guess. In practice, one often uses the more robust, but less accurate direct methods to find such an initial guess.

In contrast to the indirect methods, within direct methods, one first discretizes the OCP (3.16) and solves afterwards a finite-dimensional NLP. They are often called *first-discretize-then-optimize* methods. We follow this approach also for OCPs with nonsmooth dynamics. Therefore, we take a closer look at some of the most popular direct methods in the next section.

3.3.2 Direct optimal control

There are several reasons for the popularity and success of direct methods. The resulting NLP has the form of (2.1) and can be solved with any available NLP code. Moreover, many tailored real-time feasible algorithms for feedback control exist [78]. Very good and mature implementations of direct methods have facilitated their application [9, 81, 170, 140, 278]. The resulting NLPs are often easy to initialize and have good convergence properties [229]. Direct methods parameterize the state and control trajectories, which results in an

NLP that is a discrete-time approximation of (3.16). Depending on how this is done, we usually distinguish between direct transcription, direct single, and direct multiple shooting.

For ease of exposition we pick in all three approaches the same control parametrization. The prediction horizon $[0, T]$ is divided in $N-1$ equidistant control intervals defined by the grid $0 = t_0 < t_1 \ldots < t_N = T$. For simplicity, we pick a piecewise constant control parametrization for all control intervals:

$$u(t) = u_i,\ t \in [t_i, t_{i+1}),\ \text{for all } i \in \{1, \ldots, N-1\}.$$

The discrete-time control variables are collected in $U = (u_0, \ldots, u_{N-1}) \in \mathbb{R}^{Nn_u}$. Extensions to more elaborate control parametrizations, e.g., via polynomials, and to non-equidistant control discretization grids are straightforward.

Direct transcription

In direct transcription methods, the dynamics (3.16c) are replaced by the integration method equations (3.8) on every control interval. One of the key advantages of this class of methods is that they do not require an additional solver for the DAE. Moreover, the sensitivities are automatically computed within the NLP solver, since the discretization equations are constraints of the NLP. To avoid confusion, we remind the reader that in Section 3.2, in the exposition of integration methods, we regarded a single control. Here we consider N control intervals and perform on each of these intervals N_{int} intermediate integration steps. Of course, the number of intermediate steps can be different on every control interval but we keep them the same to reduce the notational overhead. Denote by s_i the approximation of the state $x(t)$ at the control grid points t_i for all $i \in \{0, \ldots, N\}$ and by $x_{i,n}$ the approximations obtained by single integration steps within the control interval, where $n \in \{1, \ldots, N_{\text{int}}\}$. As in Section 3.2 a single integration step is compactly represented by

$$x_{i,n+1} = \phi_f(x_{i,n}, y_{i,n}, u_i),$$

$$0 = \phi_{\text{int}}(x_{i,n}, y_{i,n}, u_i),$$

where $y_{i,n}$ collects internal variables of the n-th integration step in the i-th control interval. We go a step further and collect all N_{int} integration steps over the i-th control interval in the discrete-time system representation:

$$s_{i+1} = \Phi_f(s_i, Y_i, u_i), \tag{3.19a}$$

$$0 = \Phi_{\text{int}}(s_i, Y_i, u_i), \tag{3.19b}$$

where $\Phi_f(s_i, Y_i, u_i) = x_{i,N_{\text{int}}}$ is the state transition map and $\Phi_{\text{int}}(\cdot)$ collects all intermediate computations of all integration steps within one control interval. The vector $Y_i = (x_{i,0}, y_{i,0}, \ldots, x_{i,N_{\text{int}}-1}, y_{i,N_{\text{int}}}, x_{i,N_{\text{int}}})$ collects all intermediate integration variables of the i-th control interval.

In direct methods, the path constraints are not imposed for every $t \in [0, T]$, but only on a finite number of points to have a finite number of inequalities. Often, these constraints are imposed just on the control grid points:

$$g_{\text{p}}(s_i, u_i) \geq 0, \text{ for all } i \in \{0, \ldots, N-1\}.$$

To avoid violation of the constraint in between these points, one could additionally impose the path constraints on all intermediate integration and stage points or even on some other points obtained by interpolation.

Finally, the objective function (3.16a) is approximated by a discrete-time sum $\sum_i^{N-1} \ell(s_i, Y_i, u_i) + M(s_N)$. The term $\ell(s_i, Y_i, u_i)$ can efficiently be computed by introducing the quadrature state $x_{\text{q}}(\cdot)$ as discussed above.

Bringing all these steps together, we obtain from the OCP (3.16) the following structured and sparse NLP:

$$\min_{S,Y,U} \sum_{i=1}^{N-1} \ell(s_i, Y_i, u_i) + M(s_N) \tag{3.20a}$$

$$\text{s.t.} \quad s_0 = \bar{x}_0, \tag{3.20b}$$

$$s_{i+1} = \Phi_f(s_i, Y_i, u_i), \qquad \text{for all } i \in \{0, \ldots, N-1\}, \tag{3.20c}$$

$$0 = \Phi_{\text{int}}(s_i, Y_i, u_i), \qquad \text{for all } i \in \{0, \ldots, N-1\}, \tag{3.20d}$$

$$0 \leq g_{\text{p}}(s_i, u_i), \qquad \text{for all } i \in \{0, \ldots, N-1\}, \tag{3.20e}$$

$$0 \leq g_{\text{t}}(s_N), \tag{3.20f}$$

where $S = (s_0, \ldots, s_N)$ collects all state variables at the control grid point and $Y = (Y_0, \ldots, Y_{N-1})$ collects all intermediate variables.

Direct transcription is a *fully simultaneous* approach to optimal control as the optimization and simulation problem are solved at the same time [38, 78, 229]. Thus, the discrete-time states S and control trajectory U are feasible only at a converged solution when the simulation problem equations and inequality constraints are satisfied. One of the most popular direct transcription methods is direct collocation based on Gauss-Legendre and Radau IIA IRK methods [37, 78, 226, 229].

Direct multiple shooting

The direct multiple shooting method for solving optimal control problems was introduced by Bock and Plitt [44]. Nowadays, this method is widely used and implemented in several popular software packages such as `MUSCOD-II`[81], `ACADO` [140] and `acados` [278], to name a few.

In shooting methods the internal integration variables Y_i are hidden from the optimizer. Instead, for a given control u_i and initial value s_i an external *integrator* code solves $\Psi_{\text{int}}(s_i, Y_i, u_i) = 0$ in (3.19) for Y_i, i.e., we obtain $Y_i(s_i, u_i)$ and define the maps $\Phi(s_i, u_i) \coloneqq \Psi_f(s_i, Y_i(s_i, u_i), u_i)$ and $\hat{\ell}(s_i, u_i) \coloneqq \ell(s_i, Y_i(s_i, u_i), u_i)$. Depending on the underlying integration method, this can be done explicitly or implicitly. We express the resulting map as

$$s_{i+1} = \Phi(s_i, u_i), \quad \text{for all } i \in \{0, \dots, N-1\}. \tag{3.21}$$

The integrator reports to the optimizer function values and their derivatives w.r.t. to the initial value $\frac{\partial \Psi(s_i, u_i)}{\partial s}$ and controls $\frac{\partial \Psi(s_i, u_i)}{\partial u}$. A proper implementation takes care of all the sensitivity-related pitfalls discussed in Section 3.2.2. Moreover, $\Psi(\cdot)$ can be more general and represent even an adaptive step size integrator. Note that the use of such integrators in a direct transcription method is significantly more involved [37, 223]. This highlights one of the advantages of direct multiple shooting.

The constraints and objective are discretized as in the previous section and with direct multiple shooting we obtain the following NLP:

$$\min_{S,U} \sum_{i=1}^{N-1} \hat{\ell}(s_i, u_i) + M(s_N) \tag{3.22a}$$

$$\text{s.t.} \quad s_0 = \bar{x}_0, \tag{3.22b}$$

$$s_{i+1} = \Phi(s_i, u_i), \qquad \text{for all } i \in \{0, \dots, N-1\}, \tag{3.22c}$$

$$0 \leq g_{\text{p}}(s_i, u_i), \qquad \text{for all } i \in \{0, \dots, N-1\}, \tag{3.22d}$$

$$0 \leq g_{\text{t}}(s_N), \tag{3.22e}$$

Same as direct transcription, direct multiple shooting enables an easy initialization as all the state values s_i at all control grid discretization points are explicitly available. Moreover, this also enables us to perform all integrator calls for all nodes in parallel. Continuity of the overall final trajectory is ensured via the constraints (3.22c).

Direct single shooting

Using the currently available control vector U and initial value $\bar{x}_0$, direct single shooting completely removes the variables S from the NLP (3.22). For simplicity, we evaluate the path constraints on the same grid as in the previous cases. Let us denote the obtained state values, at the control discretization grid points by $\tilde{s}_i(\tilde{s}_0, U)$ and we highlight here the dependence on U and the initial value $\tilde{s}_0$. These values are computed by recursive calls of the integrator in Eq. (3.21). We obtain the following NLP:

$$\min_{\tilde{s}_0, U} \sum_{i=1}^{N-1} \ell(\tilde{s}_i(\tilde{s}_0, U), u_i) + M(\tilde{s}_N(U)) \tag{3.23a}$$

$$\text{s.t.} \quad \tilde{s}_0 = \bar{x}_0, \tag{3.23b}$$

$$0 \leq g_{\mathrm{p}}(\tilde{s}_i(\tilde{s}_0, U), u_i), \qquad \text{for all } i \in \{0, \ldots, N-1\}, \tag{3.23c}$$

$$0 \leq g_{\mathrm{t}}(\tilde{s}_N(\tilde{s}_0, U)), \tag{3.23d}$$

The initial value constraint (3.23b) could also be removed, however, it is very useful when one has to solve quickly several related parametric NLPs [75]. Direct single shooting is a *sequential* approach as the simulation and optimization problem iterations are carried out sequentially. Direct multiple shooting is a simultaneous approach, sometimes also considered to be hybrid, i.e., both sequential and simultaneous as parts of the computations are performed outside of the optimization routine.

Obviously, the NLP (3.23) has less variables than (3.20) or (3.22). But in contrast to these problems, it is dense and has no significant sparsity structure. Having the intermediate integration variables s_i and Y_i explicitly in the optimization problems is an example of *lifting* [8]. Informally speaking, the additional variables help to distribute the nonlinearities. For example, in a recursive call of an integrator in direct single shooting the nested integrator functions in $\tilde{s}_{i+1}(\tilde{s}_0, U) = \Phi(\Phi(\Phi(\cdots(\Phi(\tilde{s}_0, u_0), u_1), \cdots), u_{i-1}), u_i)$ might reinforce the nonlinearity and significantly harm the convergence. This gives an intuition for the often observed better convergence rate of the simultaneous approaches. Direct multiple shooting performs often better for unstable dynamical systems [229]. If the integration steps on every shooting interval are decoupled, all function and derivative evaluations can be performed in parallel. Interestingly, the cost per iteration (function evaluations and linear system solve) in a Newton-type method can be shown to be the same for both direct single and multiple shooting [8]. We use the lifting technique also to reduce the nonlinearity of some smooth parts of nonsmooth systems, cf. Section 6.3.

Chapter 4

Nonsmooth Dynamical Systems

This chapter provides an introduction to nonsmooth dynamical systems. We regard ODEs whose vector fields and solutions do not have strong continuity and differentiability properties. We start by outlining differences between *hybrid* and *nonsmooth* dynamical systems. Both notions are often used to model the same problems, which my lead to confusion. Nonsmoothness inevitably causes some numerical and theoretical difficulties. Therefore, we motivate and justify why nonsmoothness is sometimes necessary. Moreover, we show why smoothing the discontinuities may not lead to satisfying results. We highlight several qualitative and numerical phenomena which appear with the introduction of nonsmoothness in ODEs and which can not be encountered within smooth dynamical systems.

Outline. The content of this chapter is as follows. Section 4.1 provides a high-level introduction to the field. Section 4.2 illustrates on simple examples numerous phenomena encountered within nonsmooth systems and their time discretization, which are not encountered with smooth systems. In Section 4.3, we review some basics from nonsmooth analysis and nonsmooth dynamical system formulations. We recall some basic existence and uniqueness results. Section 4.4 concludes the chapter and provides further references for related topics not covered here.

4.1 Introduction

Hybrid and nonsmooth dynamical systems exhibit both continuous and discrete behavior. The discrete and continuous dynamics do not just coexist but are closely interconnected. They usually arise when first principle models are coupled with if-then or either-or conditions or from empirical laws such as Coulomb friction. This leads to some circumstances that are never encountered in smooth dynamical systems. The discrete events cause *switches* in the dynamics and/or *jumps* in the solution trajectories. After a switch, the system might even evolve in a lower dimensional subspace. In many mathematical modeling frameworks, discrete states are introduced, which are constant between two switches. The dynamics for a fixed discrete state are called *mode* of the overall hybrid system.

In the literature, the terms *nonsmooth* and *hybrid* systems are often used interchangeably, or the differences are often not clarified enough. In the next section, we precisely describe the conceptual differences between nonsmooth and hybrid dynamical systems. In an introductory discussion, one can regard them to be the same. We proceed with some classifications and descriptions of the main properties of the arising classes.

The first big classification is based on how the discrete events or switches, are triggered. We distinguish between:

1.) *Internal switches*: triggered implicitly, depending on the systems' differential state,

2.) *External switches*: triggered explicitly, independent of the differential state.

Some typical solutions of a hybrid or nonsmooth system with internal switches are illustrated in Figure 4.1. We regard some set $R_i \subset \mathbb{R}^{n_x}$ and a well-defined smooth ODE, e.g., with a Lipschitz continuous vector field, defined on this set. Starting at some initial value $x(0) \in R_i$ the system evolves locally according to the given dynamics for some time until it reaches the boundary ∂R_i. This is called an *event* or *switch*. The corresponding time is called the *event time* or *switching time*. In this instance of time, a discrete and instantaneous change in the trajectory happens, and the system changes its mode. There are several typical outcomes. The trajectory might continue to evolve in a new region $R_j \subset \mathbb{R}^{n_x}$ equipped with another vector field. This causes a kink in the solution and is illustrated in Figure 4.1 (a). Or it could *slide* on the boundary between the two regions as shown in Figure 4.1 (b). Furthermore, some parts of the state space might be even infeasible as often encountered in mechanical systems with unilateral constraints, cf. Chapter 8. In such cases, the trajectory $x(t)$ might

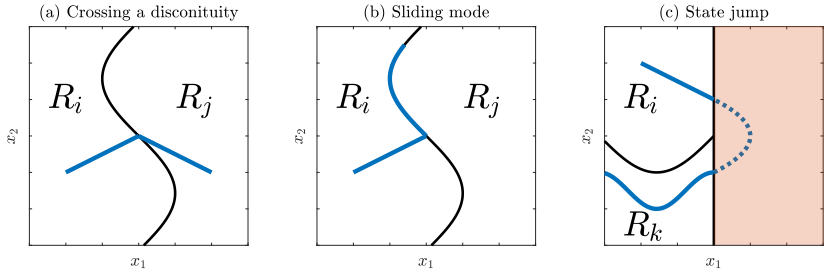


Figure 4.1: Illustration of typical solutions of nonsmooth dynamical systems. The blue curve is the solution trajectory $x(t)$. The red-shaded area is infeasible for the dynamical system. The blue dashed line is not a part of the trajectory but indicates that there is a state jump.

need to *jump* to some place in R_i or another region $R_k \subset \mathbb{R}^{n_x}$ and then evolve continuously with the same or a different vector field, respectively, see Figure 4.1 (c). We see here that the *switches* and *jumps* are triggered by reaching some surface in the state space. We speak of *state-dependent*, *autonomous*, *implicit*, or *internal* switches and jumps. These systems are the central topic of this thesis, and various aspects of modeling, reformulations, numerical methods, and direct optimal control are the topics of the forthcoming chapters. Of course, the simple illustration above does not cover the rich behavior of nonsmooth systems, and throughout this thesis, we will introduce more details and study some specific features of such systems.

The second class consists of dynamical systems with *external, explicit* or *controlled* switches and jumps. Typical examples are systems from the field of engineering with discrete actuators where on and off decisions are made. In optimal control problems, such systems are readily modeled via ODEs that are smooth in the states and equipped with integer-valued control functions. Consequently, the resulting optimal control problems require the solution of mixed-integer optimization problems which are in general very difficult to solve. However, in the last two decades, the field of mixed-integer optimal control has enjoyed great successes in the development of the theory and numerical methods [235, 30, 167, 175, 53]. For a recent survey, cf. [193]. Many tailored algorithms enable even real-world applications with sampling times in the range of minutes and seconds [54, 193]. In general, a dynamical system might have both internal and external switches, which is a particularly difficult class of systems.

The rich behavior of nonsmooth dynamical systems allows many other classifications. E.g., depending on the mathematical framework (cf. Section 4.3)

that is used or the continuity properties of the solutions and their derivatives. The solution trajectories of nonsmooth dynamical systems, modeling the physical world, are usually smooth function pieces joined by kinks and jumps. Hence, the discontinuities arise either in the trajectory or its time derivatives. This leads to one possible classification which covers numerous, but not all, mathematical frameworks for nonsmooth ODEs. It regards the continuity of the trajectory $x(t; x_0)$ and its time derivatives. A similar classification is used in [174]. We distinguish NonSmooth Dynamics (NSD) with the following properties:

(NSD1) continuous, but nonsmooth r.h.s., jump discontinuity in the 2^{nd} time derivative - continuously differentiable solutions, e.g., $\dot{x} = 1 + |x|$,

(NSD2) discontinuous r.h.s., jump discontinuity in the first time derivative - absolutely continuous solutions, e.g., $\dot{x} \in 2 - \text{sign}(x)$,

(NSD3) jump discontinuity in the trajectory - solutions are functions of bounded variation, e.g., the bouncing ball, $\ddot{q}(t) = -9.81$, $\dot{q}(t^+) = -e\dot{q}(t^-)$ if $q(t) = 0$ and $\dot{q}(t^-) < 0$, $e \in [0, 1]$.

4.1.1 Hybrid versus nonsmooth dynamical systems

This section aims to clarify what is usually meant by hybrid and nonsmooth dynamical systems in the literature. These terms cover all kinds of systems with continuous and discrete dynamics. This description is not restrictive and permits numerous modeling frameworks to capture essentially the same features of a dynamical system. However, this sometimes leads to ambiguity and confusion. For example, many articles and textbooks [113, 186, 189], when they speak of hybrid systems have a very concrete modeling framework in mind that has a very precise definition.

The applied mathematics community usually deals with nonsmooth dynamical systems [3, 4, 49, 186, 93, 19, 273, 258]. Here, the coupling between the discrete and continuous parts is rather implicit and represented via nonsmooth and set-valued (even unbounded) maps. This gives rise to the notions of differential inclusions, dynamic complementarity systems, differential variational inequalities, and more. They are all particular instances of nonsmooth dynamical systems. In the analysis of nonsmooth systems, one can often rely on the powerful tools for static problems from the fields of convex analysis, mathematical programming, and nonsmooth analysis [19, 89, 234].

A different approach, most often taken in the control community, is to represent the coupling between discrete and continuous parts of the systems explicitly. The explicit interconnection is readily represented via finite hybrid automata

[186, 272], which are finite state machines accompanied by an ODE and DAE for every discrete state. In this case, one usually speaks of *hybrid (dynamical) systems*. Transitions are triggered by so-called *guards*, i.e., switching conditions. They may be accompanied by *reset maps*, i.e., state jump laws. One of the main benefits of this approach is that it is often not too difficult to define meaningful notions of solutions and to study properties such as the existence, uniqueness of solutions, and stability of the controlled dynamical system [186]. There are several formal definitions for hybrid automata which only differ in details, e.g., [188, Definition II.1], [272, Definition 1.2.3], [186, Definition 3.1], [113, Definition 2.2], [24] to name a few. Some of these frameworks introduce hybrid time domains, i.e., distinct *clocks* for the continuous and discrete states [113]. A very similar notion is hybrid inclusions, which are often used to study standard control theoretical questions [113]. It is important to realize that all relations, modes, and mode transitions are explicitly stated. On the one hand, this might simplify the analysis as nothing is hidden. On the other hand, constructing a hybrid automaton might be cumbersome and error-prone, since all possible relations must be stated explicitly.

As the terms nonsmooth and hybrid are not restrictive, it is difficult to define them rigorously and to highlight all differences explicitly. However, we can summarize the main differences as follows. The discrete nature in nonsmooth dynamical systems is encoded implicitly via nonsmooth and set-valued equations. In contrast, in hybrid automata (in the literature often just called hybrid systems), integer states are introduced, and all transitions are expressed explicitly, e.g., with the help of reset, guard maps and automata.

Both approaches, together with their strengths and weaknesses, are very useful and have their place in lively research areas. A classical example that highlights the advantages of the nonsmooth over hybrid modeling approach is granular matter. For m particles $n = \frac{m(m-1)}{2}$ complementarity conditions are needed to represent the contact conditions between all particles. The combinatorial nature is hidden in the complementarity constraints. For deriving a hybrid automaton, one needs to explicitly treat all 2^n modes. In this example, the hybrid systems approach leads to an exponential growth in relations. On the other hand, the generality of hybrid automata enables easier modeling of some problems. For example, systems with hysteresis are easy to represent via a hybrid automaton, and the formulation as a nonsmooth dynamical system is nontrivial. In this thesis, we focus on nonsmooth dynamical systems. In particular, we always express the discrete parts via complementarity conditions at some point.

4.1.2 Why nonsmooth dynamical systems?

Combining first principles with logical expressions necessarily leads to nonsmooth dynamical systems. Many smooth dynamical systems cannot be stabilized with smooth control functions [68, 229]. Sophisticated control strategies, such as MPC, result in control laws that are at best piecewise smooth functions [229]. Physical systems can have qualitatively different modes. Idealizing the mode changes to be instantaneous leads to discontinuities in the differential equations. In the literature, one can find very extensive lists of applications of nonsmooth models, e.g., [3, 4, 51, 68, 155, 186].

Mechanics and robotics provide one of the largest sources of nonsmooth dynamical systems. Contact between perfectly rigid bodies introduces discontinuities in the velocities. In nature, no perfectly rigid bodies exist. However, the compression and decompression phases during contact happen at much shorter time scales compared to other motions of the bodies. Such time scales are beyond the resolutions of practical sensors, and resolving them with numerical integration methods is computationally very expensive. Moreover, estimating all parameters that describe the complex phenomena of contacts with friction is very difficult. It turns out, regarding the rigid limits, which introduce discontinuities in the model, is sufficiently physically accurate and numerically beneficial [49, 174, 255]. To describe the impacts of rigid bodies, we only need to estimate the friction coefficient and coefficient of restitution. Other examples of such modeling are ideal diodes and power electronics in general [3]. In this case, very nonlinear voltage-current characteristics are replaced by complementarity conditions. One of the main strengths of nonsmooth systems is their simplicity of formulation.

Nonsmoothness leads to difficulties that are not encountered with smooth dynamical systems. Therefore, a related question is: Why not just smooth everything? Indeed, smoothing and using an adaptive step-size integrator is often useful, when it comes to quickly obtaining a simulation result with moderate accuracy. However, the error is dominated by the smoothing parameter, and for good accuracy, very stiff ODEs must be treated, which quickly becomes impractical. For instance, frequent switches or sliding modes will require very small step sizes in the integration. The numerical experiments in Section 4.2.5 reveal many pitfalls of smoothed models that cannot be resolved even with the use of very advanced adaptive step-size integrators. Moreover, Section 4.2 shows many phenomena that only occur in the presence of nonsmoothness. Smoothing removes such effects, which might blur the overall picture. E.g., smoothing a Coulomb friction model might lead to slipping of the rigid body in situations where it should be at rest. More importantly, in the case of direct optimal control, smoothing and standard methods for smooth systems

have some non-obvious fundamental limitations. For example, the numerical sensitivities of smoothed systems only converge to the correct value if the step size shrinks faster than the smoothing parameter [259]. As a consequence, many optimization variables are needed for accurate solutions. We can conclude that using nonsmooth dynamical systems combined with tailored numerical methods can be of great practical benefit.

4.1.3 Numerical simulation of nonsmooth systems

Efficient numerical integration methods are necessary to make some classes of models practically useful. Proper time discretization is crucial in standard direct methods for solving OCPs with nonsmooth ODEs. It is well-known that standard integration methods applied to nonsmooth dynamical systems have only first-order accuracy [4]. State jumps and the lack of uniqueness of solutions complicate the computations even more. Moreover, even if the numerical approximations converge to the true solutions, the numerical sensitivities usually do not [203, 259], cf. Chapter 5.

There are several ways to discretize and simulate a nonsmooth ODE, and they can be divided as follows:

1.) switch-detecting, event-driven, active-set methods [4, 213, 250],

2.) time-stepping methods, event-capturing [3, 4, 199, 255],

3.) smoothing and penalty methods [4, 203, 259].

In the following, we highlight their main features. In Section 4.2.5, we compare all three approaches in a numerical example and highlight their strengths and weaknesses.

Switch-detecting methods

The first approach is the only one of the three that is designed to deliver solutions of higher-order accuracy. It follows a simple idea: use a high-accuracy smooth ODE or DAE solver until a switching event is detected. Consequently, the concerned time interval is equal to the union of a finite number of a priori unknown subintervals on which the mode of the system is unchanged. For switch detection, a look at the signs and possibly the derivatives of a switching function is needed. The solution approximation is interpolated, and a root-finding method finds the switching time. Afterwards, the integration is restarted

and continued with a smooth ODE or DAE solver, possibly with the new model and initial value. If the solution is nonunique, an adequate mode is selected, or the user is allowed to choose. Note that for external switches, the switching times are known in advance, and the simulation is less difficult. Selecting the correct mode after a switch might not be straightforward. In [272], this difficulty is illustrated with the following example. Imagine a pile of boxes subject to Coulomb friction between each of them. Remove the support from one side such that the boxes start to move due to gravity. Determining which boxes slide or stick with respect to each other is nontrivial. Most high-accuracy methods assume that the trajectory crosses the discontinuity or only passes through two regions at a time [4]. An exception is Stewart's high accuracy method, which can deal with almost all switching cases [250, 252].

Higher-order accuracy schemes should always be preferred in direct optimal control. First, the numerical error should not dominate over the modeling error. Second, fewer optimization variables are needed for the same accuracy. Using an integrator with a complicated switch detection and mode selection procedure within an optimization code is often prohibitive. In Chapter 7, we introduce an event-based method that overcomes these limitations.

Time-stepping methods

The second approach does not try to exactly detect the switching times and provides, in general, only first-order accuracy [4]. Time-stepping methods advance the integration with a fixed step size and are good at capturing multiple simultaneous switching events. They usually require solving a linear or nonlinear complementarity problem and thus preserve the nonsmoothness. They are especially powerful and fast in simulating large or fast systems with many switches [15, 4, 3, 255], e.g., in the simulation of granular matter. However, they always deliver wrong numerical sensitivities [259, 292] and thus are not a suitable choice for direct optimal control. Convergence of standard Runge-Kutta methods with order one is studied in [83, 161, 264, 258].

Smoothing methods

The last approach essentially considers smooth approximations of the original system. This might yield a very stiff ODE. However, they are easy to implement and are often useful for quick tests or the generation of good initial guesses for more elaborate methods. For example, in nonsmooth mechanics (cf. Chapter 8), the non-penetration constraints that cause discontinuities in the velocity are replaced by repulsion laws [49]. To achieve reasonable accuracy with such

an approach, very stiff repulsion laws are needed, which in turn require very small-step sizes.

4.2 Phenomena specific to nonsmooth dynamical systems

The presence of nonsmoothness leads to numerous interesting phenomena. Before giving formal definitions for several classes of nonsmooth dynamical systems, we provide a series of simple examples where each highlights some property inherent to nonsmooth systems. The goal of these examples is to make the reader aware of some limitations of standard approaches, and to provide more intuition for nonsmooth dynamical systems.

4.2.1 Infinitely many switches in finite time - Zeno's phenomenon

We have seen above that nonsmooth systems encounter switches and jumps. Some models encounter even infinitely many switches or jumps in a finite time interval. In the literature, this is called the *Zeno phenomenon* [49, 186, 272]. We provide two examples of this. The first has infinitely many state jumps, and the second has infinitely many switches, i.e., discontinuous changes in the vector field.

Regard the dynamics of a one-dimensional bouncing ball with elastic impacts:

$$\begin{aligned} \dot{q}(t) &= v(t), \\ m\,\dot{v}(t) &= -g, \\ v(t^+) &= -\epsilon_{\mathrm{r}} v(t^-), \text{ if } v(t^-) \leq 0 \text{ and } q(t) = 0. \end{aligned}$$

Here, $m = 1$ is the ball's mass, $g = 9.81$ is the gravitational acceleration, and $e \in (0, 1]$ is the coefficient of restitution. For simplicity, let us assume that $q(0) = 0$ and $v(0) > 0$. Then, the first impact happens at $t_1 = \frac{2v(0)}{g}$, the second after $t_2 = t_1 + \frac{2\epsilon_{\mathrm{r}} v(0)}{g}$. It can be seen that the time between two impacts is $\Delta_{k+1} = t_{k+1} - t_k = \frac{2\epsilon_{\mathrm{r}}^k v(0)}{g}$. Since $\epsilon_{\mathrm{r}} < 1$, we see that the resulting geometric series converges, i.e., $\sum_{k=1}^{\infty} \Delta_k < \infty$. Thus, we have infinitely many impacts in finite time. An example trajectory is given in Figure 4.2.

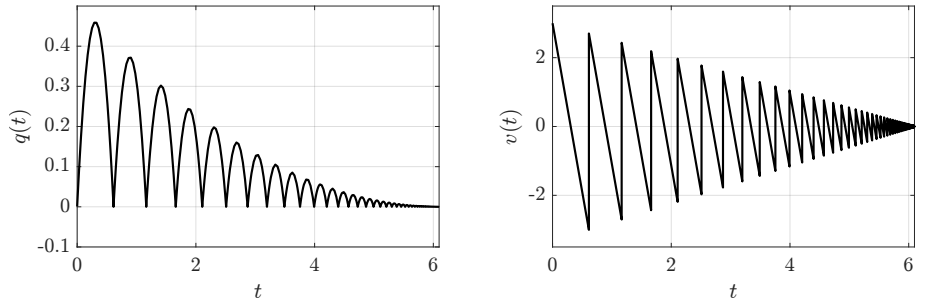


Figure 4.2: The position $q(t)$ of the one dimensional bouncing ball (left). The right plot shows the discontinuous velocity of the ball $v(t)$.

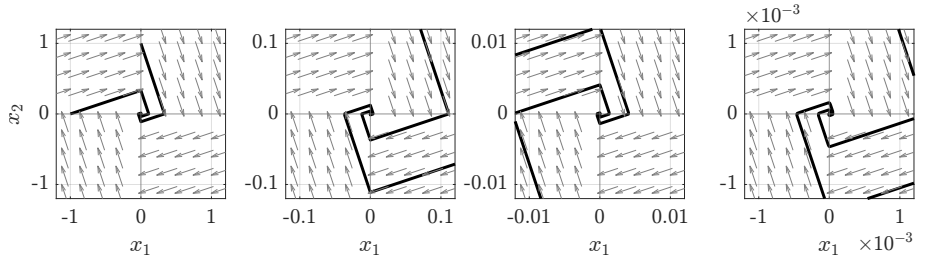


Figure 4.3: Two solution trajectories of the Filippov system in Eq. (4.1). The plots from left to right show the origin zoomed in.

Zeno behavior can also happen with systems without jumps but with a discontinuous right-hand side. Regard the following discontinuous ODE [272, Eq. (1.12)]:

$$\dot{x}_1(t) \in -\text{sign}(x_1(t)) + 2\text{sign}(x_2(t)), \tag{4.1a}$$

$$\dot{x}_2(t) \in -2\text{sign}(x_1(t)) - \text{sign}(x_2(t)). \tag{4.1b}$$

Example trajectories for several initial values are depicted in Figure 4.3. The trajectories always end up spiraling down to the origin.

Of course, real physical systems do not experience the Zeno phenomenon. However, some practical models still can experience this kind of behavior [291]. Thus, it is important to investigate conditions under which infinitely many switches occur or not in finite time [60, 243, 291]. Simulating Zeno systems with an accurate event-driven method is also impractical as it requires many restarts of the integrator. On the other hand, time-stepping methods do not have difficulties with such systems [4, 255].

4.2.2 Reduced system dimensions and sliding modes

Another very interesting property of some nonsmooth dynamical systems is that they can reduce their dimension along the trajectory. This dimension reduction is very common for ODEs with a discontinuous r.h.s. when they are cast into Filippov systems, cf. Section 4.3 and Chapter 6. If the trajectory needs to evolve on the surfaces of discontinuity, we speak of *sliding modes*. An example trajectory was given in Figure 4.1 (b).

4.2.3 Stability and instability due to switches and jumps

The stability of the modes of a nonsmooth dynamical system does not imply the stability of the overall system. We illustrate this fact with an example from [46]. It consists of a piecewise linear system with two modes of operation:

$$\dot{x} = \begin{cases} A_1 x, & \text{if } x_1 x_2 \leq 0, \\ A_2 x, & \text{if } x_1 x_2 > 0. \end{cases} \tag{4.2}$$

with

$$A_1 = \begin{bmatrix} -1 & 10 \\ -100 & -1 \end{bmatrix}, \; A_2 = \begin{bmatrix} -1 & 100 \\ -10 & -1 \end{bmatrix}.$$

It can be seen that both linear subsystems are stable. However, the overall nonsmooth system is unstable. Example trajectories are given in Figure 4.4. Interestingly, the time-reversed version of the example above is stable.

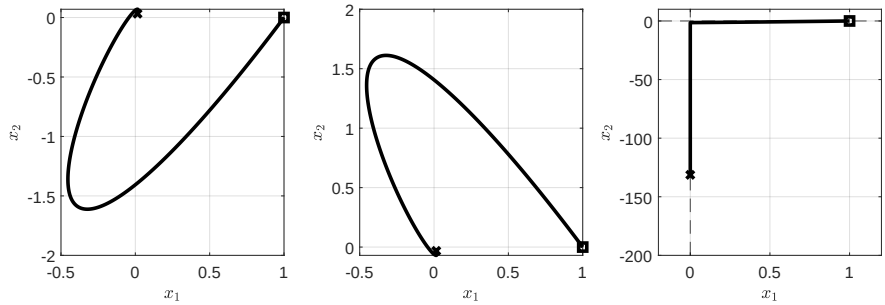


Figure 4.4: The left plot shows a trajectory for the linear systems $\dot{x} = A_1 x$, the square marks the initial point and the cross the final point. The middle plots show a trajectory of the system $\dot{x} = A_2 x$. The right plot shows a trajectory of the unstable system (4.2).

4.2.4 Numerical chattering

Simulating systems with sliding modes is more difficult as it can lead to undesired *chattering* of the numerical solution around the surface of discontinuity. We consider the following example to illustrate this:

$$\dot{x}(t) \in -\text{sign}(x(t)), \tag{4.3}$$

with $x(0) \neq 0$. We use the explicit Euler, Runge-Kutta 4 method and MATLAB adaptive step-size integrators `ode45` and `ode89`. MATLAB defines $\text{sign}(0) = 0$, but in the simulation example, this value is never reached. Figure 4.5 shows the simulation results with several standard integration methods. In Figure 4.6, we can see that lowering the step size or the tolerance of the adaptive integrator does not improve the situation but makes the chattering only faster. Interestingly, even very sophisticated adaptive step-size codes fail to deliver qualitatively good approximations even for such simple examples.

We see from the example above that not only the integration order but also the qualitative behavior plays an important role. Next, we compare the explicit and implicit Euler methods. The explicit Euler discretization with the step-size

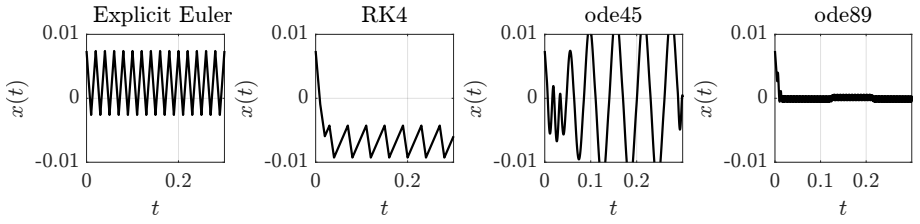


Figure 4.5: Simulation results for the example in (4.3) obtained with several integration methods.

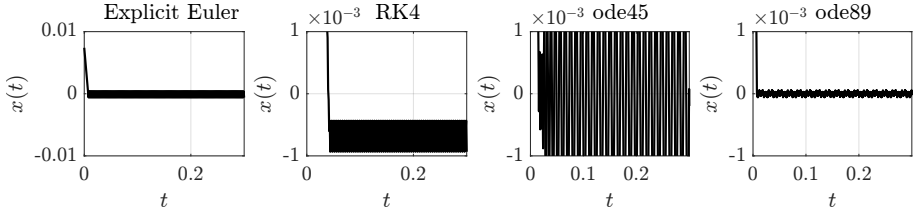


Figure 4.6: Simulation results for the example in (4.3) obtained with several integration methods, now with a smaller step size and tolerance for the adaptive integrators.

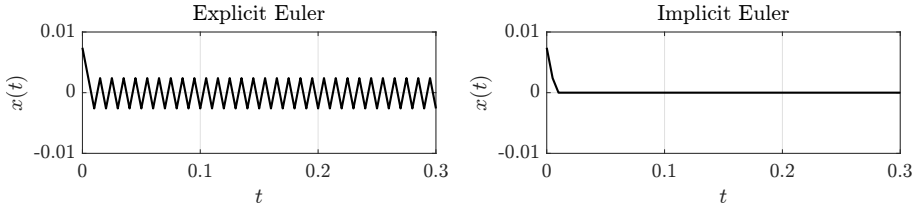


Figure 4.7: Simulation results for the example in (4.3) obtained with explicit and implicit Euler methods.

h for our example reads as:

$$x_{k+1} = x_k - h\text{sign}(x_k).$$

The sign$(\cdot)$ function can be expressed as a solution map to a Linear Program (LP)

$$s \in \arg\min_{\hat{s}} -x\hat{s} \text{ s.t. } -1 \leq \hat{s} \leq 1.$$

Using the KKT conditions of this LP allows us to derive a more sophisticated implicit Euler discretization for the example [4, 3, 258]:

$$\begin{aligned} x_{k+1} &= x_k - hs_{k+1}, \\ 0 &= -x_{k+1} - \lambda^{\mathrm{L}}_{k+1} - \lambda^{\mathrm{U}}_{k+1}, \\ 0 &\leq 1 + s_{k+1} \perp \lambda^{\mathrm{L}}_{k+1} \geq 0, \\ 0 &\leq 1 - s_{k+1} \perp \lambda^{\mathrm{U}}_{k+1} \geq 0. \end{aligned}$$

In general, implicit methods for nonsmooth ODEs require solving a nonlinear complementarity problem at every time step. The simulation results of the two Euler schemes described above are compared in Figure 4.7. Even if both explicit and implicit Euler discretizations converge to the solution of a nonsmooth system [27, Propositions 2.7 and 4.4], their qualitative behavior might be very different if sliding modes occur. The implicit Euler discretization yields in many cases provably chattering-free sliding motions [5]. Already this simple example suggests that tailored implicit integrators are needed for discretizing nonsmooth ODEs.

4.2.5 Order integration of accuracy

Due to the lack of smoothness, standard higher-order integration methods for smooth ODEs applied to nonsmooth system experience in general first-order

accuracy [4], cf. Definition 3.7. We study the behavior of three integration approaches for nonsmooth ODEs described in Section 4.1.3 on the example of an ODE with a discontinuous right-hand side.

Regard the initial value problem:

$$\dot{x} = \begin{cases} A_1 x, & \|x\|_2^2 < 1, \\ A_2 x, & \|x\|_2^2 > 1, \end{cases} \tag{4.4}$$

with $A_1 = \begin{bmatrix} 1 & \omega \\ -\omega & 1 \end{bmatrix}$, $A_2 = \begin{bmatrix} 1 & -\omega \\ \omega & 1 \end{bmatrix}$, $\omega = 2\pi$ and $x(0) = (e^{-1}, 0)$ for $t \in [0, T]$. The solution of the IVP is given in Figure 4.8. It can be shown that the switch happens at $t_\mathrm{s} = 1$ and that

$$x(T) = \begin{bmatrix} e^{(T-1)} \cos(2\pi(T-1)) \\ -e^{(T-1)} \sin(2\pi(T-1)) \end{bmatrix},$$

for $T > t_\mathrm{s}$. Thus, given a numerical approximation $\hat{x}(t)$, we can determine the global integration error $E(T) = \|x(T) - \hat{x}(T)\|$ and observe the accuracy order of the integrator. We take an irrational number for $T = \frac{\pi}{2}$ so that $t_\mathrm{s} = 1$ never coincides with the integration grid. Thereby, we avoid accidental switch detection via the discretization grid of the integrator, which would blur the results.

We regard solution approximations to this IVP obtained by MATLAB's state-of-the-art adaptive step-size integrators `ode15s`, `ode23`, `ode45` and `ode89`. To vary the minimum step size and have control over the error we change the tolerances of the integrator. The relative tolerance `reltol` is varied from 1

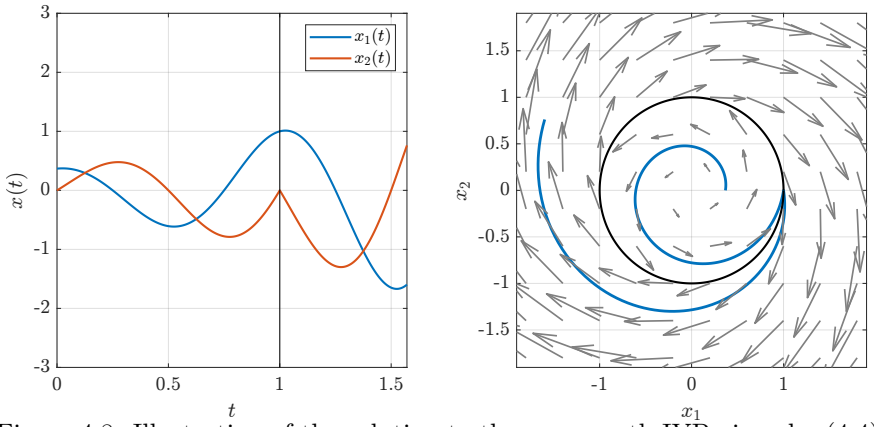


Figure 4.8: Illustration of the solution to the nonsmooth IVP given by (4.4).

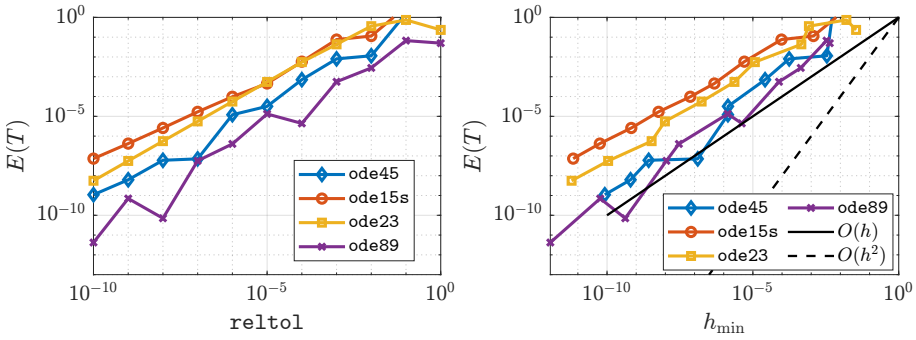


Figure 4.9: Integration error $E(T) = \|x(T) - \hat{x}(T)\|$ as a function of `reltol` for several MATLAB integrators (left plot). The integration error $E(T)$ as a function of $h_{\min}$ (right plot).

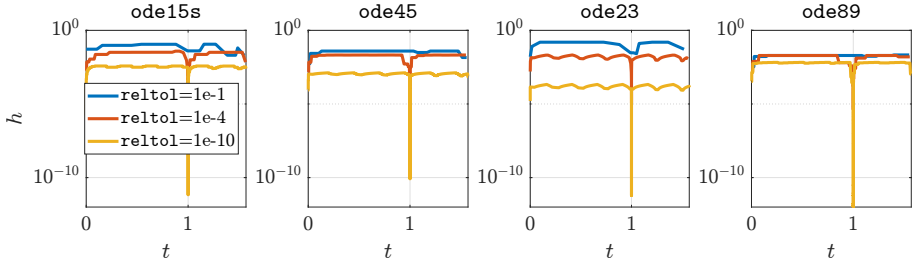


Figure 4.10: The integration step sizes of several MATLAB integrators over time for different relative tolerances.

to 10^{-10} and the absolute tolerance is set to `abstol`=`reltol`/10 At the k-th integration step the integrator estimates the local error $e(k)$. The step is accepted if the error satisfies: $\|e(k)\| \leq \max(\texttt{reltol}\|x(k)\|, \texttt{abstol})$, where $x(k)$ is the current solution approximation, cf. `https://de.mathworks.com/help/matlab/ref/odeset.html`. The integration errors $E(T)$ are plotted over the minimum step-size $h_{\min}$ that the integrator takes during simulation.

In the first experiment, we directly apply the integrators as time-stepping methods to the nonsmooth ODE (4.4). The discontinuity is not treated explicitly. The errors as a function of the relative tolerance and the minimum step-size $h_{\min}$ are given in Figure 4.9. It can be seen that all integrators achieve only first-order accuracy and thus lose their high-accuracy properties. In Figure 4.9, we plot the step sizes that the integrator takes over time for different values of `reltol`. It can be seen that all integrators must reduce their step size drastically around the switch to achieve the prescribed tolerance. Of course, in the case of

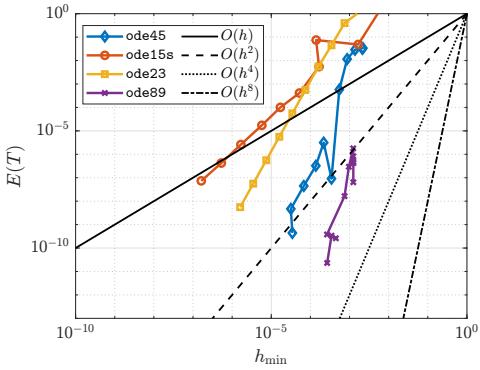


Figure 4.11: The integration error $E(T)$ vs. $h_{\min}$, now with switch detection.

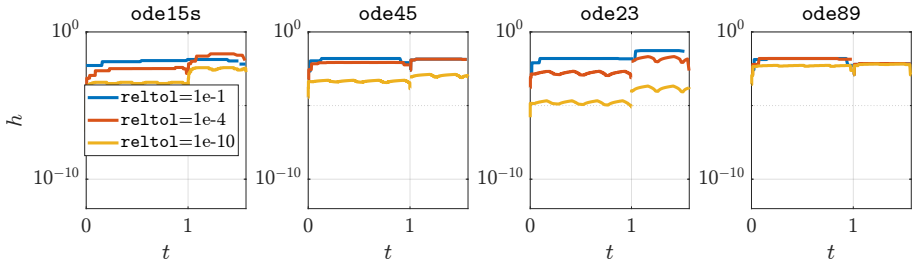


Figure 4.12: The integration step sizes of several MATLAB integrators over time for different relative tolerances, now with switch detection.

many switches, this makes the integration very expensive.

In the second experiment, we use explicit switch detection options of the integrators. Otherwise, we do not change anything compared to the previous experiment. The integration accuracy plots are given in Figure 4.11. They reveal that all integrators recover their high-accuracy properties. Similarly, we plot the step sizes over time in Figure 4.12. In contrast to Figure 4.9, we see that the integrators do not have to make unreasonably small step sizes to achieve high accuracy. Event-detection methods restart the integration process after a switch is detected. MATLAB's integrators usually take a very small integration step at the beginning of the simulation. This explains the step sizes being sometimes smaller around the switch. We remind the reader that deciding with which mode to proceed with the integration after a switch is detected is sometimes a nontrivial task, cf. the discussion in Section 4.1.3.

In our last experiment, we investigate if smoothing improves the situation. After all, a smooth ODE is obtained and the integrators are expected to work well when applied to it. We regard a smooth approximation of the ODE (4.4), where

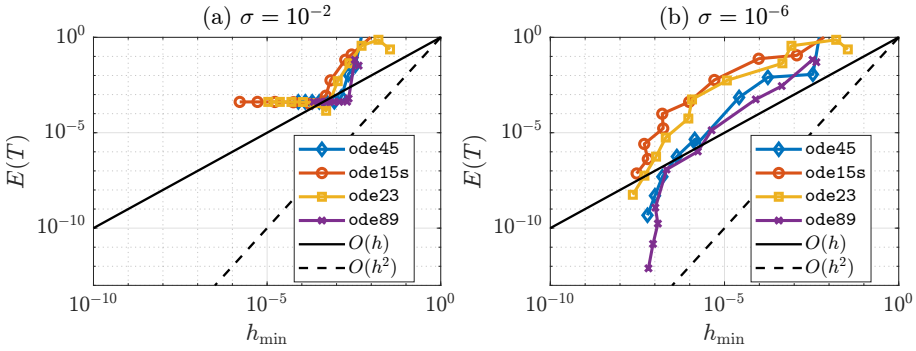


Figure 4.13: The integration error $E(T)$ vs. $h_{\min}$ with smoothing.

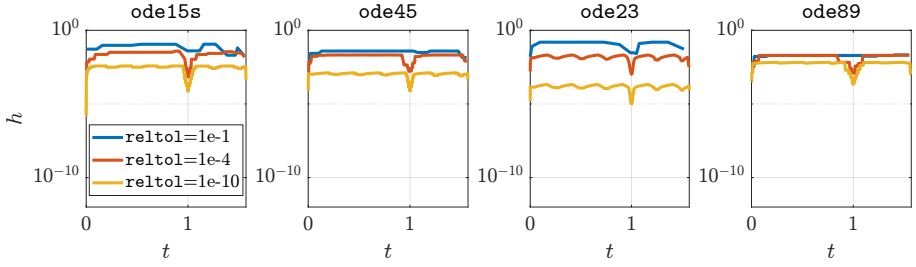


Figure 4.14: The integration step sizes of several MATLAB integrators over time for different relative tolerances, now with smoothing and $\sigma = 10^{-2}$.

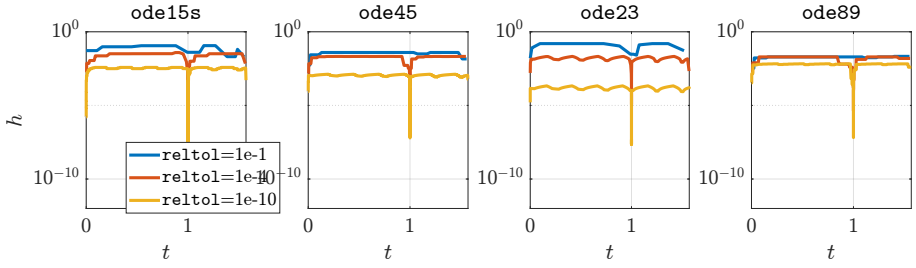


Figure 4.15: The integration step sizes of several MATLAB integrators over time for different relative tolerances, now with smoothing and $\sigma = 10^{-6}$.

we replace a $\operatorname{sign}(x)$ function with $\tanh(x/\sigma)$. The smooth approximation reads as:

$$\dot{x} = \frac{1 - \tanh\left(\frac{\|x\|_2^2 - 1}{\sigma}\right)}{2} A_1 x + \frac{1 + \tanh\left(\frac{\|x\|_2^2 - 1}{\sigma}\right)}{2} A_2 x.$$

The experiments are performed for the values $\sigma = 10^{-2}$ and $\sigma = 10^{-6}$. Under mild conditions, the solution of the smoothed ODE converges to the true solution with the rate $O(\sigma)$ [259]. The results are depicted in Figure 4.13. It can be seen that for the smaller value of σ, the integrators perform well. However, it is no surprise that the smoothing error dominates the overall error, and decreasing the tolerance cannot improve the situation further. On the other hand, for very small values of the smoothing parameters, we obtain a very fast transition around the discontinuity. This makes the system very stiff, and the integrator shows similar behavior to the case when the nonsmooth ODE (4.4) is considered. The high integration orders are lost even for `ode15s` which is well suited for stiff systems. The step sizes over time are depicted in Figure 4.14 and 4.15 for $\sigma = 10^{-2}$ and $\sigma = 10^{-6}$, respectively. We can see that in these cases very small steps are taken around the switch and there is almost no difference to the case of a discontinuous system, cf. in Figure 4.9. We can see that for a sufficiently small step size, here $h = o(\sigma)$, the integrators recover their order properties. Therefore, when high accuracy is required, we gain nothing with smoothing.

One could argue that smoothing is still useful for moderate accuracy since smaller steps are taken only at the transition. However, in the presence of sliding modes, most adaptive step size integrators have serious difficulties. To illustrate this, we simulate a smoothed version of the chattering example in Eq. (4.3):

$$\dot{x} = -\tanh\left(\frac{x}{\sigma}\right). \tag{4.5}$$

We take $\sigma = 10^{-5}$. The step sizes over time that the integrators take are given in Figure 4.16. We can see that even though the dynamics for $x = 0$ is trivial, all integrators except `ode15s` take extremely small steps. This makes the integration unreasonably computationally expensive. The integrator `ode15s`, which deals well with stiff systems performs quite well. However, having a large step size and small smoothing parameter with standard integration methods

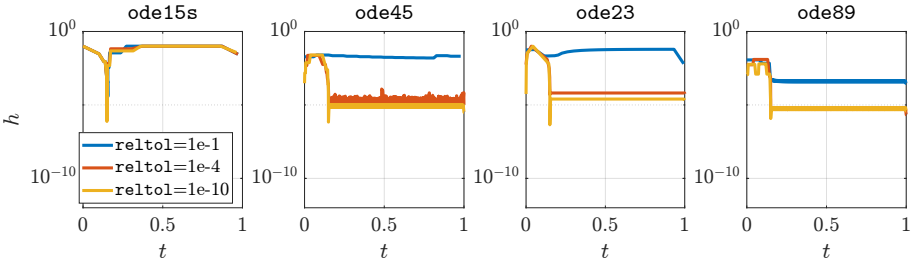


Figure 4.16: The integration step sizes of several MATLAB integrators over time for the ODE (4.5).

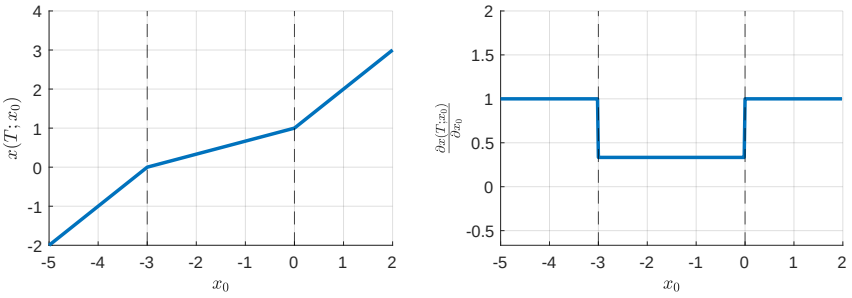


Figure 4.17: The final value $x(T;x_0)$ as function of the initial value x_0 for the ODE (4.6) (left plot). The sensitivity of the solution w.r.t. x_0 (right plot).

for smooth ODEs leads necessarily to wrong numerical sensitivities [259]. This makes this approach practical for the simulation of sliding modes, but almost useless for direct optimal control.

To summarize, our experiments reveal that besides switch detection, there is no simple way to achieve high accuracy with standard methods or smoothing. High-accuracy discretizations are desirable in direct optimal control as they result in a much smaller number of degrees of freedom in the NLP to be solved. On the other hand, using off-the-shelf event-detection methods in direct optimal control is difficult. These methods include complicated routines such as switch detection and mode selection, which are not readily included in the optimization. In Chapter 7, we develop an event-detecting method that performs all these steps implicitly by solving a nonlinear complementarity problem. This resolves all the issues discussed in this section. We will repeat this experiment with the new methods in Section 7.3.4.

4.2.6 The sensitivities are discontinuous

As discussed in Section 3.2.2, the sensitivities of ODE solutions w.r.t. the initial values or parameters are of crucial importance in direct optimal control. By now, it is no surprise that their computation becomes more complicated for nonsmooth dynamical systems. In the next chapter, we show that they cannot be computed correctly by the plain application of time-stepping methods. In the case of smoothing, they are correct only under quite restrictive assumptions on the smoothing parameter. Here, we illustrate with two examples that the sensitivities are discontinuous. Later in Chapter 6, a more formal discussion about nonsmooth sensitivities is given.

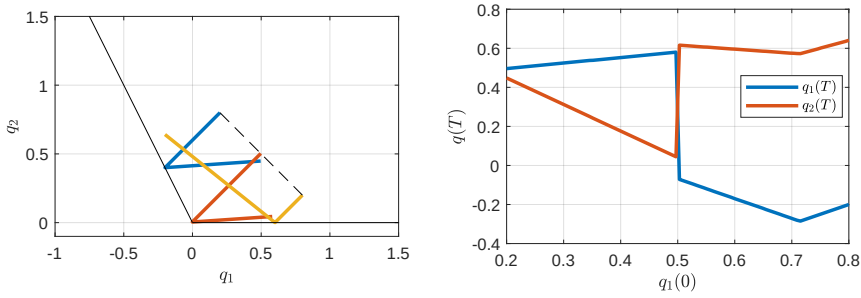


Figure 4.18: Example trajectories for the 2D particle with elastic impacts (left plot). The final positions $q(T)$ as function of the initial horizontal position $q_1(0)$ (right plot).

As a first example, we regard a simple ODE with a discontinuous r.h.s.:

$$\dot{x} = 2 - \operatorname{sign}(x). \tag{4.6}$$

We look at the solution of this ODE for $t \in [0, 1]$ for different values of the initial values $x(0) = x_0$. Figure 4.17 shows the final value $x(T; x_0)$ as function of the initial value x_0 and the nonsmooth sensitivity $\frac{\partial x(T;x0)}{\partial x_0}$. Depending on the initial value, the system is in either of the two possible modes of operation without switching or a switch taking place. This discrete behavior is also seen in the nonsmooth dependence of the final value w.r.t. the initial value.

Similar difficulties arise for systems with state jumps. To illustrate this, we consider a frictionless 2D particle with elastic impacts, with a coefficient of restitution of $\epsilon_r = 0.9$. The particle approaches a "corner" where the vertical wall makes an obtuse angle with the floor, with the velocity $v(0) = (-1, -1)$. The dynamic equations read as:

$$\ddot{q} = \begin{bmatrix} \lambda_1 \\ -g + \lambda_2 \end{bmatrix},$$

$$0 \le f_1(q) \perp \lambda_1 \ge 0,$$

$$0 \le f_2(q) \perp \lambda_2 \ge 0,$$

$$\text{if } f_i(t) = 0 \text{ and } \nabla_q f_i(q(t))^\top v(t^-) \le 0, \text{ then}$$

$$\nabla_q f_i(q(t))^\top v(t^+) = -\epsilon_r \nabla_q f_i(q(t))^\top v(t^-),\ i \in \{1, 2\},$$

with $f_1(q) = q_2 + 2q_1$ and $f_2(q) = q_2$.

Three example trajectories are depicted in the left plot of Figure 4.18. The initial position is varied along the black dashed line. The final position of the ball as a function of the initial horizontal position is plotted in the right plot of Figure 4.18. It can be seen that the end position is a discontinuous function of the initial position and that the sensitivity is nonsmooth. For the special case when the walls are orthogonal in the previous example, the end position may be continuous functions [49]. However, in general, we have more complex geometries, and friction complicates the situation even more.

4.3 Modeling frameworks for nonsmooth dynamical systems

In this section, we review some of the most widely used mathematical modeling frameworks for nonsmooth dynamical systems. Historically, their development had different motivations. Hence, all of them have their own tailored numerical methods and theory. It depends on the problem at hand what is the best choice. Of course, they are not completely unrelated. We have seen in Section 2.2 that variational inequalities, projection operators, complementarity problems, and inclusions into normal cones are closely related. It is no surprise that there exists a strong link between their dynamic counterparts. Some notions are very general and encompass many of the more specialized concepts that we treat here. We highlight some relations and give references for more elaborate studies about the relationships and equivalences. The goal of this section is not to give an exhaustive list of all known formalisms. For more details, we refer the reader to monographs [3, 4, 19, 49, 51, 246, 258].

4.3.1 Some basics from nonsmooth and set-valued analysis

In this section, we provide some elementary definitions from set-valued analysis that we need to define nonsmooth dynamical systems and to state some standard existence and uniqueness results. We have already encountered set-valued (or multi-valued) maps in the definitions of generalized equations and variational inequalities in Section 2.2. In this section, we consider more general maps. For more details on this topic, we refer the reader to the excellent monograph by Rockafellar and Wets [233].

Single-valued mappings take a vector as an input and give a singleton as an output. A natural generalization is to consider sets as outputs. Thereby, we consider a map F that associates with any input vector $x \in X \subseteq \mathbb{R}^n$ a set $F(x) \in Y \subseteq \mathbb{R}^m$. Here, F maps from X to the set of all subsets of Y, i.e., $\mathcal{P}(Y)$.

There are two common notations for this, $F : X \rightrightarrows Y$ and $F : X \to \mathcal{P}(Y)$. We use the latter. Set-valued maps are conveniently characterized via their graphs:

$$\mathrm{gr}F = \{(x, y) \in X \times Y \mid y \in F(x)\}.$$

The domain and range of $F : X \to \mathcal{P}(Y)$ are defined as:

$$\mathrm{dom}F = \{x \mid F(x) \neq \emptyset\},$$

$$\mathrm{img}F = \{y \mid \text{ there exists } x \in X \text{ such that } y \in F(x)\}.$$

The inverse map is $F^{-1} : Y \to \mathcal{P}(X)$ is defined as

$$F^{-1}(y) = \{x \in X \mid (x, y)\ \mathrm{gr}F\}.$$

Continuity

The notion of continuity for set-valued functions is more complicated than for single-valued functions. Two common notions are *outer* and *inner semi-continuity*. In the literature they are sometimes called upper and lower semi-continuity for set-valued maps [19, 246]. However, if $F(x)$ is always a singleton these definitions do not reduce to the definitions of upper and lower semi-continuity for single-valued functions and this might cause confusion. For a discussion accompanied by examples, we refer to [4, Section 2.1.2]. Some equivalent, arguably more complicated, definitions via notions of set convergence can be found in [234, Chapter 5]. These notions can be defined with different set-convergence concepts, cf. [233, Chapter 4]. Here, we use simpler but equivalent definitions which do not need set convergence.

Definition 4.1 (Outer and inner semi-continuity)**.** *A set-valued map $F(\cdot)$ is called outer semi-continuous (OSC) (resp. inner semi-continuous (ISC)) at $x_0 \in X$ if for every $\epsilon > 0$ there exists a $\delta > 0$ such that $F(x) \subset F(x_0) + \epsilon\mathcal{B}(0)$ (resp. $F(x_0) \subset F(x) + \epsilon\mathcal{B}(0)$) for all $x \in x_0 + \delta\mathcal{B}(0)$. A set-valued map $F(\cdot)$ is said to be outer semi-continuous (resp. inner semi-continuous) if it is so for all $x_0 \in X$.*

A well-known result is that the set-valued map $F(\cdot)$ is outer semi-continuous if and only if $\mathrm{gr}F$ is a closed set in $X \times Y$. A simple example of OSC functions are all single-valued continuous functions.

Definition 4.2 (Continuity)**.** *A set-valued map $F(\cdot)$ is called continuous at $x_0 \in X$ if it is both OSC and ISC at this point. A set-valued map $F(\cdot)$ is said to be continuous if it is so for all $x \in X$.*

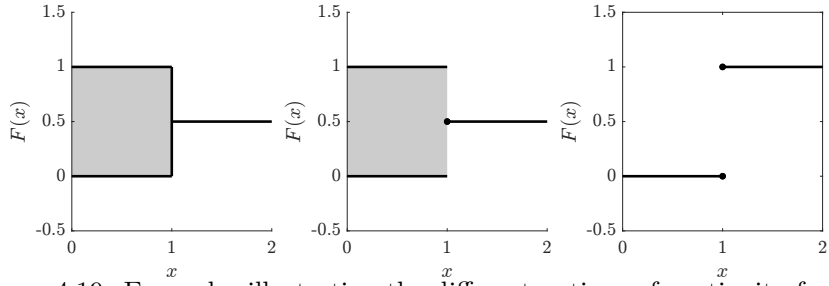


Figure 4.19: Examples illustrating the different notions of continuity for set-valued maps.

There exist set-valued extensions for other notions of continuity, e.g., Lipschitz continuity. We omit the restating here.

To provide some intuition for the last three definitions we provide some examples and non-examples in Figure 4.19. In the first plot from left to right, the function is at $x = 1$ OSC but not ISC. Otherwise, it is continuous. Similarly, in the second plot, the function is ISC at $x = 1$ but not OSC. The function in the third plot is OSC at $x = 1$, however, it is not convex valued at this point. A selection f of F is a single-valued mapping such that $f(x) \in F(x)$ for all $x \in X$. Note that OSC functions do not necessarily have continuous selections. We will encounter such examples in the sequel for the case of differential inclusions arising from ODEs with a discontinuous r.h.s..

We have seen in Theorem 2.19 that the monotonicity of single-valued functions plays an important role in the existence of solutions of generalized equations. The monotonicity of set-valued maps is also important in the analysis of evolution problems with such maps.

Definition 4.3 (Maximal monotone maps)**.** *A set-valued map $F : X \to \mathcal{P}(Y)$ is said to be monotone on X if for all pairs (x_1, y_1) and (x_2, y_2) in $\mathrm{gr}F$ it holds that:*

$$(x_1 - x_2)^\top (y_1 - y_2) \geq 0.$$

It is strictly monotone if the last inequality is strict for all $x_1 \neq x_2$. It is maximal monotone if $\mathrm{gr}F$ is not properly contained in the graph of any other monotone mapping.

Continuous and monotone maps are also maximal monotone. For example, the third plot from left to right in Figure 4.19 is monotone but not maximal monotone. By filling in the gap at $x = 1$ we would obtain a maximal monotone map.

The one-sided Lipschitz condition

The last notion we introduce in this section is the one-sided Lipschitz (OSL) condition. Lipschitz continuity is one of the most important properties in the study of ODEs, cf. Theorem 3.3. Similarly, the OSL condition is crucial for proving the uniqueness of solutions of many kinds of nonsmooth dynamical systems discussed below [68].

Definition 4.4 (Lipschitz continuity)**.** *A function $f : \mathbb{R}^n \to \mathbb{R}^n$ is called Lipschitz continuous if there exists a nonnegative Lipschitz constant $L \geq 0$ such that, for all $x_1, x_2 \in \mathbb{R}^n$:*

$$\|f(x_1) - f(x_2)\| \leq L\|x_1 - x_2\| \tag{4.7}$$

Definition 4.5 (One-sided Lipschitz condition)**.** *A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is said to be one-sided Lipschitz if there exists a constant $l \in \mathbb{R}$ such that, for all $x_1, x_2 \in \mathbb{R}^n$:*

$$(f(x_1) - f(x_2))^\top (x_1 - x_2) \leq l\|x_1 - x_2\|^2. \tag{4.8}$$

Note that there are no significant restrictions on l. However, some authors require l to be nonnegative [68]. Notably, the OSL condition can be satisfied even by discontinuous functions. This OSL condition was introduced in numerical analysis in the study of stiff differential equations. One-sided Lipschitz constants are usually smaller than Lipschitz constants and provide thus sharper bounds [125, 56], e.g., it is useful for sharp Gronwall-type inequalities [126, Lemma 12.1]. Continuity of the vector field and OSL are sufficient to prove uniqueness of solutions for ODEs, cf. [68, Proposition 2]. Thus the assumptions of Theorem 3.3 (Picard-Lindelöf) can be relaxed. A vector field violating the OSL can still have a unique solution, cf. [68, Example 8]. However, there exist examples that violate the OSL and have nonunique solutions, cf. [68, Example 7].

The definition of OSL for set-valued maps is extended as follows.

Definition 4.6 (One-sided Lipschitz condition)**.** *The set-valued map $F : \mathbb{R}^n \to \mathcal{P}(\mathbb{R}^n)$ satisfies the one-sided Lipschitz condition with the constant ρ if for all x_1 and x_2 and all selections $y_1 \in F(x_1)$ and $y_2 \in F(x_2)$ it holds that*

$$(y_1 - y_2)^\top (x_1 - x_2) \leq \rho\|x_1 - x_2\|^2. \tag{4.9}$$

4.3.2 Differential inclusions

A very general class of nonsmooth dynamical systems arises by replacing the right-hand side of a smooth ODE with a set. We obtain nonsmooth dynamical

systems called Differential Inclusions (DI):

$$\dot{x}(t) \in \mathcal{F}(t, x(t)) \text{ for almost all } t \in [0, T], \tag{4.10}$$

Here $\mathcal{F} : \mathbb{R} \times \mathbb{R}^{n_x} \to \mathcal{P}(\mathbb{R}^{n_x})$ is a set-valued map which assigns to any point in time t and $x \in \mathbb{R}^{n_x}$ a set $\mathcal{F}(t, x) \subseteq \mathbb{R}^{n_x}$. An element $y \in \mathcal{F}(t, x(t))$ for a fixed $(t, x(t))$ is called a *selection*.

There are many motivations to work with such a mathematical object [246]. Clearly, any ODE $\dot{x} = f(t, x)$ can be rewritten as a DI by setting $\mathcal{F}(x, t) = \{f(t, x)\}$. A set of differential inequalities $\dot{x} \leq f_i(t, x)$ for $i \in \{1, \ldots, n\}$ can be written in the form of (4.10). If we consider a fully implicit ODE, $g(\dot{x}, x, t) = 0$, finding a (locally) unique explicit ODE $\dot{x} = f(t, x)$ is not always possible. However, in the settings of DIs, there always exist an explicit DI $\dot{x} \in \mathcal{F}(t, x) = \{v \mid g(v, x, t) = 0\}$. They are also useful if the r.h.s. of an ODE is not accurately known. A solution $x(\cdot)$ to the ODE $\dot{x} = f(x)$ is also a solution of the DI $\dot{x} \in f(x) + \mathcal{B}_\epsilon(0)$. A control system of the form $\dot{x} = f(t, x, u)$ and $u \in U$ can be rewritten as DI $\dot{x} \in \cup_{u \in U} f(t, x, u)$. The theory of DIs is very useful to prove the existence of optimal controls, cf. [246]. Many discontinuous laws arise in the study of mechanical and electronic devices. Often it is impossible to study ODEs with a discontinuous right-hand side with standard methods from analysis and DIs are an indispensable concept for their analysis. We come back to this in more detail in Section 4.3.5 and Chapter 6. We can conclude that DIs are both a powerful tool for modeling and mathematical analysis.

Same as for ODEs, some standard questions naturally arise also for DIs. For example, one is interested in the questions of the existence and uniqueness of solutions, the dependence of solutions on initial conditions and parameters, and the computation of accurate numerical approximations to solutions. The theory of DI is mature and detailed answers to the questions above can be found in the literature [4, 19, 246]. As with any system of equations, a DI (4.10) has in its full generality no solution. However, by making some not-too-restrictive assumptions, the existence of solutions can be guaranteed. Moreover, these assumptions on $\mathcal{F}(t, x)$ determine the properties of the time-discretization methods and which tools from the analysis are applicable. The next assumption summarizes some standard conditions [19].

Assumption 4.7 (Basic assumptions for DI)**.** *The function $\mathcal{F} : [0, T] \times \mathbb{R}^{n_x} \to \mathcal{P}(\mathbb{R}^{n_x})$ has the following properties:*

i) $\|y\| \leq C(t)(1 + \|x\|)$ for all t, x and $y \in \mathcal{F}(t, x)$, where $C(\cdot)$ is an integrable function,

ii) $\mathcal{F}(t, \cdot)$ is OSC in x for all t,

iii) the set $\mathcal{F}(t,x)$ is nonempty, closed and convex for all t and x.

Next, we restate a standard existence result.

Theorem 4.8 (Existence of solution, Theorem 4, p. 101 in [19].)**.** *Regard the initial value problem related to the DI* (4.10) *with the initial value $x(0)=x_0$. Suppose that the function $F:[0,T]\times\mathbb{R}^{n_x}\to\mathcal{P}(\mathbb{R}^{n_x})$ satisfies the conditions of Assumption 4.7. Then there exists an absolutely continuous solution $x(\cdot)$ to this initial value problem.*

We briefly discuss the conditions of Assumption 4.7. The boundedness condition i) ensures that the solution does not go to infinity in finite time. It turns out that it is difficult to relax the other two conditions to prove existence. E.g., showing existence with a nonconvex set on the r.h.s. is significantly more difficult, see [19] and [246] for detailed discussions on this matter. At this point, nothing can be concluded about uniqueness. Consider the DI $\dot{x}\in[-1,1]$ from [4, Example 2.14]. Any $x=at$, with $a\in[-1,1]$ is a solution.

However, even if $\mathcal{F}(\cdot)$ is set-valued, one should not always think of multiple solutions. Most DIs coming from practical problems have unique solutions. Of course, some additional assumptions are needed. The uniform one-sided Lipschitz condition guarantees the uniqueness of solutions [161].

Assumption 4.9. *The function $\mathcal{F}:[0,T]\times\mathbb{R}^{n_x}\to\mathcal{P}(\mathbb{R}^{n_x})$ satisfies the uniform one-sided Lipschitz (UOSL) condition with the constant ρ:*

$$(y_2-y_1)^\top(x_2-x_1)\leq\rho\|x_2-x_1\|^2, \tag{4.11}$$

uniformly for all t and all selections $y_2\in\mathcal{F}(t,x_2)$ and $y_1\in\mathcal{F}(t,x_1)$.

Theorem 4.10 (Uniqueness of solution, Theorem 2.5 in [161])**.** *Regard the initial value problem related to the DI* (4.10) *with the initial value $x(0)=x_0$. Suppose that the function $\mathcal{F}:[0,T]\times\mathbb{R}^{n_x}\to\mathcal{P}(\mathbb{R}^{n_x})$ satisfies the conditions of Assumptions 4.7 and 4.9. Then there exists a $T>0$ such that $x(\cdot)$ is a unique solution to the initial value problem on $[0,T]$.*

A widely studied subclass of DIs is when the function $-F$ is *maximal monotone*, cf. Definition 4.3. In the case of maximal monotone DIs, existence and uniqueness can be proven even if $\mathcal{F}(t,x)$ is not a compact subset of $\mathbb{R}^{n_x}$ anymore. Examples are inclusions into normal cones to convex sets, cf. Section 4.3.7. We omit here restating some classical results related to maximal monotone DIs and refer the reader to [19, Chapter 3].

Measure differential inclusions

The theorems above regard DIs whose solutions $x(\cdot)$ are in the class of absolutely continuous functions. Most classic literature treats such DIs, and they are sometimes called *ordinary* DIs [19, 246]. However, many applications result in models where $x(\cdot)$ must be a discontinuous function of time. This gives rise to the so-called Measure Differential Inclusions (MDI). They were formally defined by Moreau [196, 197, 198]. In MDIs, a solution $x(\cdot)$ is not anymore continuous but usually a function of bounded variation. The right-hand side is a closed, convex, but unbounded set. An additional question that arises with discontinuous $x(\cdot)$ is: What values should $x(t^+)$ take at points of discontinuity t? This is usually resolved by state jump laws, cf. Chapter 8. Much more on MDIs can be found in e.g., [4, 49, 51, 197, 198, 258].

4.3.3 Differential variational inequalities

Given an initial value $x(0) = x_0$, a Differential Variational Inequality (DVI) [220] is the problem of finding functions $x : [0, T] \to \mathbb{R}^{n_x}$ and $z : [0, T] \to \mathbb{R}^{n_z}$ such that

$$\dot{x}(t) = f(t, x(t), z(t)), \tag{4.12a}$$

$$z(t) \in K, \text{ for almost all } t, \tag{4.12b}$$

$$0 \leq (\hat{z} - z(t))^\top F(t, x(t), z(t)), \text{ for all } \hat{z} \in K \text{ and for almost all } t. \tag{4.12c}$$

The function $f : [0, T] \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_z} \to \mathbb{R}^{n_x}$ defines the *main dynamics* of the system. The algebraic variable $z(\cdot)$ is determined by the VI (4.12c) defined by the function $F : [0, T] \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_z} \to \mathbb{R}^{n_z}$ and the closed convex set K. DVIs were formally introduced by Pang and Stewart [220], although many special cases of DVIs were studied many decades before. This will become apparent from the discussion in the sequel. The formulation in Eq. (4.12) is very general and further assumptions on $f(\cdot), F(\cdot)$ and K determine the existence, uniqueness, and qualitative properties of $x(\cdot)$ and $z(\cdot)$. A DVI is more than a simple coupling of an ODE and a VI, see [258, Section 5.2.2.] for examples where the VI has a unique solution for all $x(t)$ but the associated DVI does not.

DVIs have more structure than DIs and are a useful abstraction for studying many practical and theoretical problems. For example, they can be used for studying Filippov systems, cf. Chapter 6. Real-time MPC algorithms can be interpreted as time-discretization schemes for DVIs [17]. Several important models fit into this form, most prominently rigid bodies with friction and impact [255, 258] (cf. Chapter 8) and electric circuits with electronic devices [3]. We

refer the reader to [51, 220, 258] for an extensive collection of theoretical results and application examples.

From the sole definition in Eq. (4.12), we cannot conclude anything about the qualitative properties of the solutions, e.g., to which class, NSD1 to NSD3, would a DVI fit into, cf. Section 4.1. One way to classify DVI is their *index* [220], that is how many times $F(t, x(t), z(t)) = 0$ has to be differentiated w.r.t. time t to find $z(t)$ as a function of $x(t)$. This definition is similar to the Definition 3.5 for the index of DAEs. A similar concept is the *relative degree*, cf. [51, Appendix C]. Index zero DVIs have continuously differentiable solutions and fall into NSD1 [220, Proposition 5.1]. Index one DVIs have in general absolutely continuous solutions [258] and correspond to NSD2 systems. Index two systems are systems with state jumps [258, Chapter 6] and fit into NSD3. The specification in (4.12) is not sufficient to calculate a solution of a DVI of index 2, and we need to specify *state jump laws*. In general, for higher index DVI solutions fail to exist [258], and if they exist they are distributions [51]. Such cases are beyond the scope of this thesis.

When it comes to the uniqueness and existence of solutions, the uniqueness of solutions of index zero systems is proven in [220]. Uniqueness for some index one DVIs is proven by Stewart [256]. For index two DVIs only existence can be proven [253]. For examples of nonuniqueness see [258, Chapter 6].

DVI can be easily cast into differential inclusions [258]. Denote the set of all solutions, parameterized by $x(t)$, of the VI (4.12c) by $\mathrm{SOL}(F(t, x(t), \cdot), K)$. The DVI (4.12) reads as:

$$\dot{x}(t) \in f(t, x(t), \mathrm{SOL}(F(t, x(t), \cdot), K)),\ x(0) = x_0.$$

Depending on the properties of the set $\mathrm{SOL}(F(t, x(t), \cdot), K)$ this can also be an MDI, e.g., in the case of index two differential variational inequalities. Making further assumptions on the set K, we arrive at different formalisms. In the trivial case of $K = \{0\}$, we have that $z(t) = 0$ and (4.12) reduces to an ODE. Similarly, if $K = \mathbb{R}^{n_z}$, then $z \in \mathbb{R}^{n_z}$ and $F(t, x(t), z(t)) = 0$ and the DVI (4.12) reduces to an DAE. An interesting subclass of DVIs arises if K is a convex cone, which we regard in the next section.

4.3.4 Dynamic complementarity systems

If the set K is a convex cone, then a VI is equivalent to a complementarity problem, cf. Proposition 2.21. This gives rise to a very practical class of nonsmooth dynamical systems, namely the Dynamic Complementarity Systems

(DCS):

$$\dot{x}(t) = f(t, x(t), z(t)),\ x(0) = x_0, \tag{4.13a}$$

$$K \ni z(t) \perp F(t, x(t), z(t)) \in K^*, \text{ for almost all } t, \tag{4.13b}$$

Obviously, DCS can be studied with the tools for DVIs. However, from a computational point of view, it is easier to treat a finite number of complementarity conditions instead of infinitely many inequalities in the VI. Depending on the complementarity problem and *main* dynamics, there are several classes of DCS, e.g., Linear Complementarity Systems (LCS), Mixed LCS, nonlinear complementarity systems, gradient complementarity systems, and so on. For an overview see [3, Section 2.4.3] or [51]. Some references studying the existence of solutions and other properties are [48, 52, 133, 257]. In particular, LCSs are extensively studied in [59, 244, 273, 134]. DCSs can also be seen as nonsmooth differential algebraic equations (DAEs) [51] if the conditions (4.13b) are expressed as the zero level set of a C-function, cf. Definition 2.23. In this thesis, we are mainly interested in the case of $K = \mathbb{R}^{n_z}_{\geq 0}$. We show in Chapter 6 that Filippov systems can be rewritten as DCS. In Chapter 7, we develop discretization methods for these DCS with many favorable theoretical and practical properties. From the discussions above, the following implications are apparent: DI $\supset$ DVI $\supset$ DCS $\supset$ ODE.

4.3.5 Discontinuous ODEs and Filippov systems

ODEs with discontinuous r.h.s. (or discontinuous systems for short) are a class with mature theory. Now, we regard an ODE with a discontinuous right-hand side and study the following IVP

$$\dot{x}(t) = f(t, x(t)),\ x(0) = x_0. \tag{4.14}$$

This class of systems is ubiquitous in the application of nonsmooth dynamical systems. The pioneering work of Filippov provided a solid theoretical foundation for their study [93]. It turned out that standard tools of analysis were not sufficient to study the existence and uniqueness of solutions. Even the concept of solutions needed refinement.

Recall that classical solutions $x(\cdot)$ to (4.14) on an interval $[0, T]$ are continuously differentiable. Clearly, with a discontinuous $f(\cdot)$, this is impossible. If one gives up the requirement of continuous differentiability, new notions of solutions can be defined. For example, for Carathéodory solutions the ODE should be satisfied in integral form $x(t) = x(0) + \int_0^t f(t, x(t))\mathrm{d}t$ for $t > 0$, where the integral is the Lebesgue integral. This allows discontinuities of $f(\cdot)$ in x. Therefore, in

some situations, a Carathéodory solution might be of use, as $x(\cdot)$ is absolutely continuous, and it requires (4.14) to be satisfied for almost all $t \in [0, T]$ [68]. However, this relaxation is not always sufficient to have a solution. For the sake of illustration, consider the following initial value problem from [68, Example 5]:

$$\dot{x} = \begin{cases} 1, & x < 0, \\ -1, & x \geq 0, \end{cases} \tag{4.15}$$

with $x(0) = x_0$. For $x_0 > 0$, there exist the solution $x(t) = x(0) - t$ for $t \in [0, x_0)$. Similarly, for For $x_0 < 0$, there exist the solution $x(t) = x(0) + t$ for $t \in [0, -x_0)$. For t larger then $|x_0|$ in both cases, each solution reaches the point $x(t) = 0$ and cannot leave it as the vector fields from both sides push towards it. However, since $\dot{x} = 0 \neq -1$, we have no solution in the classical or Carathéodory sense.

To study discontinuous ODEs, we need a generalized solution concept. In the case of a continuous right-hand side, it should coincide with the classical solution, and it should have a solution for every initial value. How to proceed is a modeling step, and Filippov suggested to embed the ODE (4.14) into the following DI [93]:

$$\dot{x}(t) \in \mathcal{F}_{\mathrm{F}}(t, x(t)), \; x(0) = x(0), \tag{4.16}$$

where the set-valued map $\mathcal{F}_{\mathrm{F}} : \mathbb{R} \times \mathbb{R}^{n_x} \to \mathcal{P}(\mathbb{R}^{n_x})$ is defined as:

$$\mathcal{F}_{\mathrm{F}}(t, x(t)) := \bigcap_{\delta > 0} \bigcap_{\mu(N) = 0} \overline{\mathrm{conv}} f(t, x + \delta \mathcal{B}(x) \setminus N) \tag{4.17}$$

The DI (4.16) is a so-called Filippov system. An absolutely continuous function $x(\cdot)$ is said to be a Filippov solution if it is a solution to the IVP (4.16). Given a point x, the idea behind this definition is to regard the closed convex hull of all neighboring values in a ball $x \in \delta B(x)$ instead of only $f(x)$. Thereby, all values of $f(\cdot)$ on the sets N of measure zero are neglected. By definition, this set is convex. If $f(\cdot)$ is bounded, the set-valued map $\mathcal{F}_{\mathrm{F}}(\cdot)$ is compact and outer semi-continuous. Moreover, if $f(\cdot)$ is continuous at x we obtain $\mathcal{F}_{\mathrm{F}}(t, x(t)) = \{f(t, x(t))\}$. A proof for the last assertions can be found in [19, Proposition 1, p. 102]. This means that the conditions of Assumption 4.7 are satisfied, and Theorem 4.8 implies the existence of solutions. If further Assumption 4.9 holds, Theorem 4.10 is also applicable. Several specialized theorems for the uniqueness of Filippov systems can be found in the monograph [94]. There are several other extensions applicable to (4.14). For an excellent survey on this topic with many illustrative examples the reader is referred to [68].

We revisit the example in (4.15) and apply Filippov's extension to it. We obtain

$$\dot{x} \in \begin{cases} \{1\}, & x < 0, \\ [-1,1], & x = 0, \\ \{-1\}, & x \geq 0, \end{cases} \tag{4.18}$$

It can be seen that this DI has a solution for any $x(0)$ and for all $t \in [0,\infty)$, since for $t > |x_0|$ we have that $\dot{x} = 0 \in [-1,1]$.

A closely related solution concept is the one of Krasovskiĭ [171]. The only difference to Filippov's extension is that it does not ignore sets N of measure zero in (4.17). The following example from [4, Eq. (2.17)] illustrates the significance of this detail. Regard the ODE with a discontinuous r.h.s.:

$$\dot{x} = \begin{cases} 1, & \text{if } x \in \mathbb{Q}, \\ 0, & \text{if } x \in \mathbb{R} \setminus \mathbb{Q}. \end{cases} \tag{4.19}$$

The set of rational numbers $\mathbb{Q}$ has a measure of zero in $\mathbb{R}$. Filippov DI is trivial and reads $\dot{x} \in \{0\}$ and has always a unique solution. On the other hand, Krasovskiĭ's DI reads as $\dot{x} \in [0,1]$ and has infinitely many solutions.

In practice, one does not often encounter systems as (4.19), but ones with some special structure that should be exploited. As we see in the forthcoming chapters, a great number of practical nonsmooth systems have a piecewise smooth vector field. These systems are called Piecewise Smooth Systems (PSS). They are closely related to the classes of switched and variable structure systems [51]. Many results related to PSS are collected in the monographs [34, 155]. A generic PSS reads as

$$\dot{x}(t) = f_i(x(t)), \text{ if } x(t) \in R_i \subset \mathbb{R}^{n_x},\ i \in \mathcal{J} \coloneqq \{1,\dots,n_f\}, \tag{4.20}$$

where R_i are disjoint open sets and $f_i(\cdot)$ are smooth functions on an open neighborhood of $\overline{R}_i$, n_f is a positive integer. Note that the ODE is not defined on the boundaries ∂R_i. However, Filippov extensions ignore these sets, and this causes no trouble. In fact, one can assign any value to ∂R_i without changing the corresponding Filippov DI (4.16). On the other hand, the extensions of Krasovskiĭ are not applicable for (4.20). We study this class of systems and their Filippov extensions in great detail in Chapter 6.

4.3.6 Projected dynamical systems

Projected Dynamical Systems (PDS) regard a closed convex set K and a vector field $f(\cdot)$ whose domain contains K. In the interior of K, the flow is governed

by $-f(\cdot)$, and on the boundary of K, the vector field is changed such that $x(t)$ stays in K. In equations, this reads as:

$$\dot{x} \in \text{proj}_{\mathcal{T}_K(x(t))}(-f(x(t))), \text{ for almost all } t \in [0,T] \tag{4.21}$$

where $\text{proj}_{\mathcal{T}_K(x(t))}(\cdot))$ is the projection operator and $\mathcal{T}_K(\cdot)$ is the tangent cone defined in Definition 2.4. Using tools from convex analysis for the projection operator, several other equivalent definitions are possible, cf. [51, Section 2.5]. Note that the r.h.s. is in general a discontinuous function, and one cannot expect the existence of a classical solution. Solutions $x(\cdot)$ to IVPs related to the PDS (4.21) are usually absolutely continuous. Thus, PDS belong to the class of NSD2. Under continuity of $f(\cdot)$ and some other conditions, the uniqueness of solutions is shown in [50, Theorem 1]. Under appropriate assumptions, a PDS can be rewritten as a DCS [135]. Brogliato et al. [50] show that a PDS is equivalent to the following DI:

$$-\dot{x}(t) \in f(x(t)) + \mathcal{N}_K(x(t)), \tag{4.22}$$

which is related to a type of DI that we discuss next.

4.3.7 Moreau's sweeping processes

Sweeping processes are a particular class of differential inclusions first introduced by J.J. Moreau [196, 197]. They are inclusions into normal cones of *moving* convex sets $C(t)$. A *moving set* is a set-valued function $C : [0,T] \to \mathcal{P}(\mathbb{R}^{n_x})$, that for every $t \in [0,T]$ is a nonempty closed convex set. Note that the set can possibly change its shape over time. The name *sweeping process* is inspired by the fact that such DI model particles that are swept in the space by the moving set.

The first-order sweeping process is a differential inclusion and reads as

$$-\dot{x}(t) \in \mathcal{N}_{C(t)}(x(t)), \text{ for almost all } t \in [0,T],\ x(0) = x_0 \in C(0). \tag{4.23}$$

Assuming continuity of $C(t)$, a solution to the last problem satisfies the following properties: $x(t) \in C(t)$ for all $t \in [0,T]$, $x(\cdot)$ is differentiable for almost all $t \in (0,T)$ and $x(\cdot)$ satisfies the inclusion for almost all $t \in [0,T]$. They find application in nonsmooth mechanics, and electronics [4, 3, 51] and have mature theory and simulation methods [49, 199].

A related class is the perturbed sweeping process which reads as

$$-\dot{x}(t) \in \mathcal{N}_{C(t)}(x(t)) + f(t, x(t))), \text{ for almost all } t \in [0,T],$$
$$x(0) = x_0 \in C(0),$$

Note that the PDS (4.22) can be interpreted as a perturbed sweeping process with a constant set $C(t) = K$. The assumptions of convexity can be relaxed, and some other definitions of the normal cone might be used, which results in more general DIs than (4.23), cf. [271]. Note that if the set $C(t)$ *jumps*, then the solution $x(t)$ must jump as well, cf. [4, Section 2.2] for more details.

The normal cone to a convex set is an unbounded set and thus Theorem 4.8 is not applicable. However, the existence and uniqueness of absolutely continuous solutions under appropriate assumptions were already proven by Moreau in [197]. Furthermore, the existence and uniqueness of an absolutely continuous solution of the perturbed sweeping process are proven in [87, Theorem 1].

The so-called second-order sweeping processes [197, 198] are also widely used in practice. They have been developed for Lagrangian systems with unilateral constraints, cf. Chapter 8. Thereby, the velocity states are usually discontinuous functions of bounded variation and the position states are absolutely continuous functions of time. Electric circuits with idealized electronic elements can also be recast into this form, cf. [3, Section 2.5.4]. We omit an exposition of these systems and refer the reader to [49, Chapter 5] for more details.

4.4 Conclusions and further reading

This chapter provides an introduction to nonsmooth dynamical systems, which are the main topic of this thesis. However, the broad terms nonsmooth and hybrid are used often interchangeably in the literature for essentially the same class of systems and this might cause confusion. We clarify the subtle differences in introductory discussions, discuss strengths and weaknesses and provide some illustrative examples. The discrete-continuous nature in dynamical systems is in the nonsmooth dynamical systems community usually expressed via nonsmooth mappings. On the other hand, hybrid dynamical systems are usually represented by some kind of hybrid automaton. Both modeling approaches have their strengths and weaknesses, which we discussed and illustrated in some examples.

The nonsmoothness in the dynamics gives rise to some phenomena not seen in smooth dynamical systems, e.g., Zeno behavior, and nonsmooth sensitivities, to name a few. Moreover, this causes some numerical difficulties not encountered when dealing with smooth dynamical systems. We investigate these difficulties in a series of simple examples and illustrate the somewhat surprising limitations of standard simulation methods. This motivates further the development of tailored methods in later chapters of this thesis.

We finish this chapter by reviewing several standard modeling frameworks

for nonsmooth dynamical systems. This list is far from extensive, and we provide only a brief overview that fits our needs in this thesis. We restate some well-known existence and uniqueness results for these systems. Some links between the formalisms above are highlighted as well. However, the connections might be non-obvious and difficult to obtain. An extensive summary of the relationships between various nonsmooth dynamical systems is provided in [51]. Of course, there are many details and possible classifications for automaton-based representations of hybrid systems. To learn more about these topics, we refer the reader to [113, 186].

Finally, we list some references with more details on nonsmooth dynamical systems. For an introduction to set-valued and nonsmooth analysis, the reader might look into the textbook by Rockafellar and Wets [234]. To learn more about differential inclusions (possibly with a nonconvex set in the r.h.s.) with absolutely continuous solutions, see the textbooks by Aubin and Cellina [19] and by Smirnov [246]. Of course, a great reference is also Filippov's classic text [94]. The theory and numerical methods for differential variational inequalities are extensively treated in Stewart's monograph [258]. One of the largest sources of nonsmooth dynamical systems comes from nonsmooth mechanics. A great collection of results is provided by Brogliato [49]. Other good references on this topic are [174, 4, 258], to name a few. Numerical methods for nonsmooth dynamical systems are treated in great detail by Acary and Brogliato [4] and Acary et. al. [3]. Dynamical systems with a discontinuous r.h.s. are reviewed in great detail in [68]. Sweeping processes are discussed in detail by Brogliato [49] and, of course, in Moreau's work [197, 198].

Chapter 5

Limitations in Nonsmooth Direct Optimal Control

In this chapter, we look closer at standard direct methods for optimal control problems subject to nonsmooth dynamical systems. By standard we mean the application of time-stepping methods in direct transcription or direct multiple shooting. In this case, after discretization one obtains a nonsmooth nonlinear program. Usually, one can smooth the nonsmoothness before or after the discretization explicitly, or the smoothing might happen implicitly in the NLP solution method. For example, the latter happens if one applies relaxation methods for solving an MPCC (cf. Section 2.4.2), which is obtained by discretizing an OCP subject to a dynamic complementarity system. Alternatively, the nonsmoothness can be treated by appropriate mixed-integer reformulations. However, taking these steps without care might lead to spurious and inaccurate solutions. In this chapter, we discuss some non-obvious, but fundamental limitations of these standard methods.

We show on counterexamples from the literature that the numerical sensitivities obtained in a direct method paired with a time-stepping discretization are wrong, no matter how small the step size is. Moreover, we show that the sensitivities of the smooth approximation of nonsmooth systems converge to correct value, only if the step size h is shrinks faster than the smoothing parameter σ. These two facts were for NSD2 systems first shown in the seminal paper of Stewart and Anitescu [259]. We show that similar difficulties arise also for NSD3 systems. This highlights the necessity of the tailored methods developed in the later chapters of this thesis.

Outline. We start the chapter by surveying standard direct methods for OCP with nonsmooth dynamical systems in Section 5.1. In Section 5.2, we discuss the fundamental limitations of standard discretization and smoothing methods in direct optimal control of NSD2-type systems. Similarly, in Section 5.3 we study the limitations of direct methods for NSD3-type systems. Section 5.4 summarizes the findings and concludes this chapter. This chapter is partially based on [203].

5.1 Survey on direct optimal control methods for nonsmooth systems

In this section, we provide an overview of direct optimal control methods found in the literature. This survey can not cover all existing methods but highlights some standard approaches with a focus on methods that solve an MPCC at some point. More references can be found in [43, 51, 293]. Most direct methods discussed here either solve a nonsmooth NLP, e.g., an MPCC obtained after a possibly sophisticated discretization of the nonsmooth dynamical system, or a mixed-integer nonlinear program. Some heuristic approaches try to guess the switching sequence and formulate a multi-stage OCP, which results in a smooth NLP after discretization.

We compare them by several criteria. First, what kind of optimization problem needs to be solved, e.g., NLP, MPCC, or a mixed-integer program? Second, we regard the accuracy of the discretization scheme. As we have seen in the previous sections, most discretization methods for nonsmooth systems have only first-order accuracy. Third, we analyze if the numerical sensitivities are correct, and if so, we state under which condition. As a last criterion, we compare for which classes of systems (NSD1 to NSD3) the method at hand is suitable.

A summary of the methods we survey is given in Table 5.1. In the sequel, we provide more detailed comments on each of these methods and mention some similar ones, which are not listed in the table. Moreover, we discuss some specific advantages and limitations not shown in the table.

Several fundamental problems within direct optimal control of nonsmooth dynamical systems are revealed in the seminal paper by Stewart and Anitescu [259]. They reveal that standard direct methods with fixed step sizes are doomed to fail since the numerical sensitivities obtained by differentiating the result of a simulation are wrong, no matter how small the step size is. We study their counterexample in detail in the next section. It is also shown that the numerical sensitivities of a smooth approximation of an ODE

Method	Optimization problem	Discretization accuracy	Sensitivities correct	Classes covered	Citation
Stewart and Anitescu	NLP	first-order	no*	NSD1, NSD2	[259]
Kirches	NLP	higher-order	yes	NSD1-NSD3	[166]
Lin and Othsuka	MPCC	first-order	no*	NSD1-NSD3	[183]
Baumrucker and Biegler	MPCC	higher-order	yes	NSD1,NSD2	[28]
Nurkanović et al.	MPCC	higher-order	yes	NSD1-NSD3	[213]
Caspari et al.	MPCC	first-order	no*	NSD1, NSD2	[64]
Sperl	MPCC/NLP	higher-order	yes	NSD1, NSD2	[247]
Mayer, Schlöder et al.	MPVC	first-order	no*	NSD1-NSD3	[43, 168]
Vieira et al.	MPCC	first-order	no*	NSD1, NSD2	[279]
Posa and Tedrake	MPCC	first-order	no	NSD3	[225]
Bemporad	MILP	first-order	yes	NSD1-NSD3	[32]
Avraam	MINLP	higher-order	yes	NSD1-NSD3	[20]

*Some methods may include implicit smoothing of the dynamics either while solving the MPCC with a homotopy approach or explicit smoothing before the discretization. In such cases, the sensitivities are correct if the step size is smaller than the smoothing parameter.

Table 5.1: Overview of direct methods for solving optimal control systems with nonsmooth dynamical systems.

with a discontinuous r.h.s. are only correct if the step size approaches zero faster than the smoothing parameter. This makes accurate approximations computationally expensive since for moderate accuracy a very small step size is needed. Nevertheless, smoothing paired with a homotopy approach can lead to decent solutions.

Section 4.2.6 shows that the parametric sensitivities of nonsmooth ODEs have jump discontinuities when the trajectory passes through or enters a surface of discontinuity. High-accuracy event-based integration methods, which hide the external switch detection procedure from the optimizer, can be applied within a direct multiple shooting setting [44]. Kirches [166] follows such a line of thought and regards an event-detecting integrator that can treat switched systems (possibly with state jumps) within a direct multiple shooting approach. Moreover, this integrator computes correct numerical sensitivities by appropriately updating the sensitivities at points of discontinuity, cf. Section 6.2.6. However, there are several pitfalls. On the one hand, derivative-based optimization algorithms will likely fail due to the non-Lipschitz sensitivities. On the other hand, at points of discontinuity, the sensitivities are not even defined. To circumvent the mentioned difficulties in the convergence of an SQP method, the switching sequence is not allowed to change, which implies locally smooth sensitivity.

Recently, Lin and Ohtsuka [183] have proposed a non-interior-point homotopy method for optimal control problems with differential variational inequalities. The core of their approach is a tailored solver for MPCCs arising from the discretization of OCPs with DVIs. In their setting, the set K in the DVI (4.12) is assumed to be a box. This covers many systems from NSD1 to NSD3, including DCS with state jumps and Filippov systems. The DVI is discretized with the implicit Euler method, hence, only first-order accuracy can be obtained. On the one hand, the complementarity constraints in the MPCC arising from the DVI are treated with a Scholtes relaxation [237], cf. Section 2.4.2. On the other hand, the complementarity constraints in the KKT conditions of the MPCC are replaced by a smoothed Fischer-Burmeister function, cf. Definition 2.23. For computing a Newton-step for perturbed MPCC's KKT conditions an efficient Riccati recursion is utilized. Note that the relaxation of the complementarity constraints implicitly smooths the DVI [203], and the sensitivities are only correct if the step size approaches zero faster than the smoothing parameter [259]. The authors report faster computation times and fewer iterations compared to using `IPOPT` [281] together with Scholtes regularization.

One of the most remarkable methods is presented by Baumrucker and Biegler [28]. They consider piecewise smooth systems with a single switching surface (or multiple independent switching surfaces, cf. Section 6.2.5), i.e., NSD2-type systems. The PSS is treated as a Filippov system and transformed into an equivalent DCS. The key novelty is that they allow variable step sizes and modify further the complementarity conditions. In numerical examples, they demonstrated that this yields exact switch detection, higher-order integration accuracy, and correct numerical sensitivities. Unfortunately, a formal proof of the appealing properties of the method is not provided in [28]. Moreover, having the step sizes left to the optimizer as a degree of freedom, without additional conditions, results in under-determined systems of equations. As a consequence, the optimizer can play with the discretization accuracy, possibly in an undesired way.

The method of Baumrucker and Biegler [28] inspired the development of the Finite Elements with Switch Detection (FESD) [213], cf. Chapter 7. This method generalizes the ideas of [28], provides a sound theory, and overcomes some of the mentioned drawbacks. Moreover, two different ways to transform a Filippov system into a DCS are used, which are discussed in detail in the next chapter.

Several other authors transform a Filippov system into a DCS and solve an MPCC after the discretization. For example, Caspari et al. [64] treat the same class of systems as [28]. Their main idea is to use a C-function (cf. Definition 2.23) to pass from a DCS to a nonsmooth DAE. Furthermore, the C-function is

further smoothed, and a nonlinear DAE is obtained. The authors use direct single shooting for solving the resulting smooth OCPs. This approach of solving the discrete-time OCP is solving an MPCC with a smoothed C-function, as discussed in Section 2.4.1. Of course, the same strategy can be applied in a direct multiple shooting approach. However, the method suffers from the same issues as other smoothing approaches for ODEs with the discontinuous right-hand side. Namely, the sensitives are only correct if the step size is smaller than the smoothing parameter. The authors also notice that updating the smoothing parameter in a homotopy loop improves the convergence.

This improved convergence property is investigated in [203]. This work shows that the problems revealed by Stewart and Anitescu [259] carry over to DCS that is smoothed while solving an MPCC with a relaxation or smoothing strategy, cf. Section 2.4.2. The homotopy can help to converge to better local minima, since in the earlier stages of the homotopy for larger values of the smoothing parameter the sensitivities are still correct. Solving these more regular problems can bring the optimizer closer to the correct solution without getting stuck in artificial local minima due to the wrong sensitivities. We investigate this state of affairs in detail in the next section.

Motivated by the fact that the early stages of the homotopy have sub-problems with correct sensitivities and lead to convergence to good local solutions led to the development of a two-stages approach in the master thesis of Sperl [247]. Again, the same class of problems as in [28, 64, 203] is regarded. In the first stage, an MPCC with a relaxation homotopy is solved, as in [213]. The main assumption is that the optimal active set in the DCS is correct, i.e., the switching sequence is found correctly by solving an MPCC. Note that the optimizer might still converge to a spurious local solution due to the wrong sensitivities, but the switching sequence might be correct. This is exploited in the second stage, where the switching sequence is fixed, and a multi-stage discrete-time OCP is formulated. Each stage corresponds to a fixed active set, i.e., fixed mode of the PSS, and the lengths of the stages are left as degrees of freedom. In other words, only the switching sequence is fixed but not the switching times. Therefore, high accuracy and correct sensitivities are obtained. The final problem is a smooth and regular NLP that can be much easier solved than an MPCC. Sperl [247] reported promising numerical results, but a convergence analysis was not carried out.

If the switching sequence is known (or at least a good guess is available), optimizing for the switching times by solving a multi-stage OCP is a common approach in nonsmooth optimal control of NSD1-NSD3 systems [162, 240]. In the literature, several other heuristic two-stage approaches can be found, cf. [293] for a survey. The first stage determines the switching sequence, and the second solves a multi-stage OCP. However, most of them do not allow variable

stage lengths, and neither they provide a sound convergence theory.

Bock et al. [43] use generalized disjunctive programming [117] to enumerate all modes of a PSS and transform the OCP subject to a PSS into an OCP with integer controls and smooth ODEs. This allows us to apply the powerful methods for OCPs as partial outer convexification [235]. This formulation allows one to treat simultaneously external and internal switches. After discretization, a Mathematical Program with Vanishing Constraint (MPVC) needs to be solved. This is a problem class related to MPCCs, which is slightly more regular. The approach of Bock et al. cannot treat sliding modes generically, but requires a relaxation of the switching surface into an $\epsilon-$band region with a nonempty interior. Switch detection is not treated explicitly, only a heuristic mesh refinement that splits intervals with a switch is provided. Therefore, only first-order accuracy can be expected in general. Later, Kirches et al. extended this approach to dynamical systems with state jumps [168].

Many authors derive adaptations of Pontryagin's conditions for fairly general formulations of hybrid or nonsmooth dynamical systems [114, 144, 242, 262]. Some authors focus on specific classes of nonsmooth systems, e.g., sweeping processes [138]. Moreover, some of these references provide iterative methods for solving the optimality conditions [144, 138, 279]. Guo and Ye [121], and Vieira et al. [279] study optimal control of a DCS with absolutely continuous solutions, i.e., NSD2 systems. Furthermore, in [279], direct and indirect methods are proposed for numerically solving the OCP. Thus, after discretization, an MPCC is solved via a penalty or relaxation approach, cf. Sections 2.4.2 and 2.4.3. An implicit Euler discretization is used without an explicit treatment of the switches. To the best of our knowledge, none of these references focuses explicitly on computing the correct sensitivities or using higher-order order discretization methods. Consequently, the numerical sensitivities are not computed correctly, and only first-order accuracy can be obtained.

In the last decade, Contact-Implicit Trajectory Optimization (CITO) received a lot of attention in the robotics community. The contact condition between rigid bodies and friction gives rise to nonsmooth dynamical systems of the type NSD3, cf. Chapter 8. Trajectory optimization problems subject to such dynamics result in nonsmooth OCPs, where the contact sequence of the rigid bodies is discovered fully implicitly. Many authors take a direct approach to solving this OCP. One of the first works to do so is by Posa and Tedrake [225]. They use a semi-implicit Euler discretization [260, 15] for the discretization and solve the MPCC with an SQP method. Similar approaches with different (possibly smoothed) contact models, adapted time-stepping methods, and MPCC solution strategies are presented in [263, 63, 142, 141, 191, 86], to name a few. The recent survey [283] provides a broad overview of the use of optimization in legged locomotion, including many references on CITO.

Time-stepping schemes for rigid bodies with impact have only first-order accuracy [4, 255], and the numerical sensitivities are usually wrong [292]. However, all of these approaches share some properties and their practical success can be explained by several factors: 1) the CITO problems are so difficult to solve that even feasible solutions solve complex robotic tasks (spurious local minima can be found even with wrong derivatives), 2) during the solution of the discretized OCP, smoothing of the dynamics is done explicitly or implicitly, which improves the convergence, 3) using finite differences can result in correct sensitivities, despite the nonsmoothness of the dynamics.

One of the oldest approaches in nonsmooth optimal control is to enumerate all modes of a nonsmooth system with integer variables. This is already explicitly done in various hybrid systems formalisms, cf. Section 4.1.1. Bemporad and Morari [31] provide an elaborate framework to model logical relations between hybrid systems modes. The modes are assumed to be linear. Moreover, the modeling is carried out in discrete time with a fixed sampling rate, hence, there is no switch detection. This can lead to first-order accuracy at best. However, the resulting optimal control problems are mixed-integer linear programs, which can often be solved very efficiently. In the PhD thesis of Avraam [20], a more general approach is proposed. The nonsmooth dynamics are modeled via the hybrid systems formalism of Barton and Pantelidas [26], which enables one to model NSD1-NSD3 systems in continuous time. Avraam [20] proposes a direct collocation approach, where the unspecified switching sequences with variable stage lengths are expressed via linear integer constraints. This formulation enables exact switch detection for NSD1-NSD3 systems. The downside is that a mixed-integer nonlinear program must be solved, which can be computationally prohibitively expensive.

To summarize, most practical methods use only first-order accuracy discretization schemes, with possibly incorrect sensitivities. On the other hand, the high-accuracy ones treat the discontinuity in the integrator, which complicates the use of derivative-based optimization. They do not treat sliding modes appropriately or handle only systems with a single switching surface, i.e., only two regions.

5.2 Fundamental limitations of standard direct methods for NSD2 systems

In this section, we show why standard direct methods applied to ODEs with a discontinuous right hand-side are doomed to fail. We recall some results from the seminal paper of Stewart and Anitscu [259]. In particular, we show

on a counterexample, that the numerical sensitivities in a direct approach are wrong, no matter how small the integrator step size is. Moreover, it is shown that the numerical sensitivities of the smoothed approximation converge to the correct value only if the integrator step size shrinks faster than the smoothing parameter, i.e., $h = o(\sigma)$. We show that the same difficulties carry over to DCSs, which are equivalent to the initial discontinuous ODEs. These limitations are illustrated in a simple optimal control example.

5.2.1 A bimodal NSD2 system

For sake of simplicity, we consider a bimodal piecewise smooth system with a scalar switching function $\psi(x)$:

$$\dot{x}(t) = f(x) = \begin{cases} f_1(x), & \psi(x) < 0, \\ f_2(x), & \psi(x) > 0. \end{cases}$$

The Filippov convexification of this system is the differential inclusion (cf. Section 4.3.5):

$$\dot{x}(t) \in F(x) = \begin{cases} \{f_1(x)\}, & \psi(x) < 0, \\ \{f_2(x)\}, & \psi(x) > 0, \\ \overline{\text{conv}}\{f_1(x), f_2(x)\}, & \psi(x) = 0. \end{cases} \tag{5.1}$$

We assume that $f_1,\ f_2 : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ and $\psi(x) : \mathbb{R}^{n_x} \to \mathbb{R}$ are sufficiently smooth, and that $\nabla\psi(x) \neq 0$, whenever $\psi(x) = 0$. We are interested in the following non-degenerate cases:

1. *Crossing a discontinuity*: we have $\nabla\psi(x)^\top f_1(x) > 0$, $\nabla\psi(x)^\top f_2(x) > 0$, whenever $\psi(x) = 0$, i.e., $\frac{\mathrm{d}\psi(x(t))}{\mathrm{d}t} > 0$ before and after the switching time t_s, so the dynamical system will not stay on the zero manifold $\psi(x) = 0$.
2. *Sliding modde*: we have $\nabla\psi(x)^\top f_1(x) > 0$, $\nabla\psi(x)^\top f_2(x) < 0$, whenever $\psi(x) = 0$, i.e., when the dynamical system reaches the zero manifold $\psi(x) = 0$, it stays there.

The set-valued step functions $\gamma(x)$ defined as $\gamma(x) = \{1\}$ for $x > 0$, $\gamma(x) = \{0\}$ for $x < 0$, $\gamma(x) = [0, 1]$ for $x = 0$. We denote single-valued selections of $\gamma(x)$ by $\alpha(x)$. Using this definition, the ODE (5.1) can equivalently be written

$$\dot{x}(t) \in f(x) \coloneqq \{f_1(x)(1 - \alpha(x)) + f_2(x)\alpha(x) \mid \alpha(x) \in \gamma(\psi(x))\}. \tag{5.2}$$

For the sliding mode case, when the trajectory is trapped on the manifold $\psi(x) = 0$, we compute the selection of $\alpha(x) \in [0, 1]$, from the condition $\dot{\psi}(x(t)) = \nabla\psi(x)^\top f(x) = 0$, and its value is $\alpha(x) = \frac{\nabla\psi(x)^\top (f_1(x) - f_2(x))}{\nabla\psi(x)^\top f_1(x)}$. This selects the unique convex combination of f_1 and f_2 to keep the trajectory on the manifold $\psi(x) = 0$.

5.2.2 The numerical sensitives are wrong independent of the step size

We illustrate this claim by adapting the counterexample given by Stewart and Anitescu [259]. For this purpose, we revisit the example from Section 4.2.6:

$$\dot{x} \in 2 - \operatorname{sign}(x), \; x(0) = -1. \tag{5.3}$$

The goal is to obtain an approximation of $\frac{\partial x(T,x_0)}{\partial x_0}$ by differentiating the discretization method, as it is usually required in direct optimal control, cf. Chapter 3. The analytic solution of (5.3) is:

$$x(t) = \begin{cases} -1 + 3t & \text{if } t \leq -\frac{x_0}{3}, \\ (t + \frac{x_0}{3}) & \text{if } t \geq -\frac{x_0}{3}. \end{cases} \tag{5.4}$$

The exact sensitivity at e.g., $T = 2 > -\frac{x_0}{3}$ is $\frac{\partial x(2,x_0)}{\partial x_0} = \frac{1}{3}$.

Following [259], we regard a θ-method for the discretization of (5.3) with a fixed step size h

$$x_{k+1} \in x_k - h(2 - \operatorname{sign}(x_k + \theta(x_{k+1} - x_k))), \tag{5.5}$$

where $x_k \approx x(t_k)$ is the numerical approximation of $x(t)$ at $t = kh$. For $\theta = 0$ we have the explicit Euler method, for $\theta = 0.5$ the midpoint rule and for $\theta = 1$ the implicit Euler method. These time-stepping methods are known to converge to the true solution as $h \downarrow 0$ [83, 264]. Depending on the sign of $x_k + \theta(x_{k+1} - x_k)$ we have that:

$$x_{k+1} = \begin{cases} x_k + 3h & \text{if } x_k + \theta(x_{k+1} - x_k) < 0, \\ x_k + h & \text{if } x_k + \theta(x_{k+1} - x_k) > 0, \\ -\frac{1-\theta}{\theta} x_k & \text{if } x_k + \theta(x_{k+1} - x_k) = 0. \end{cases}$$

More explicitly, by using the expressions for x_{k+1}, we obtain that

$$x_{k+1} = \begin{cases} x_k + 3h & \text{if } x_k < -3h\theta, \\ x_k + h & \text{if } x_k > -h\theta, \\ -\frac{1-\theta}{\theta} x_k & \text{if } x_k \in [-3\theta h, -\theta h]. \end{cases} \tag{5.6}$$

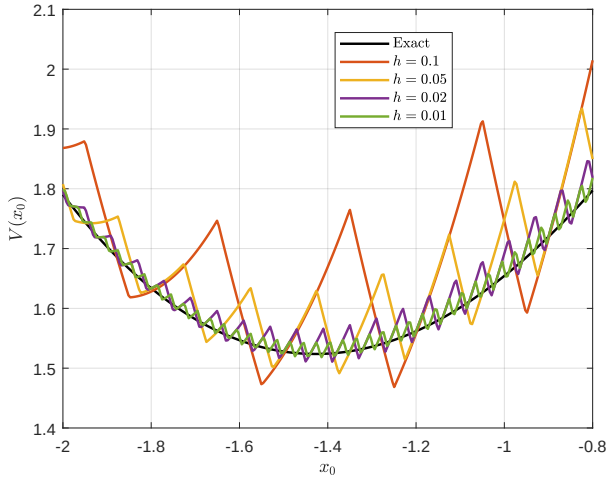


Figure 5.1: The objective function value $V(x_0)$ for different step size h.

Let N be the number of integration steps, i.e., $h = \frac{T}{N}$. By the chain rule we have the approximation $\frac{\partial x(2,x_0)}{\partial x_0} \approx \frac{\partial x_N}{\partial x_0} = \frac{\partial x_N}{\partial x_{N-1}} \cdots \frac{\partial x_1}{\partial x_0}$. From equation (5.6), it follows that $\frac{\partial x_{k+1}}{\partial x_k}$ is either 1 or $-\frac{1-\theta}{\theta}$. For example, for the implicit Euler method with $\theta = 1$, we have $\frac{\partial x_{k+1}}{\partial x_k}$ to be zero or one. Similarly, for $\frac{1}{2} < \theta < 1$ and $x_k \in [-3\theta h, -\theta h]$ we have that $x_{k+1} = \frac{1-\theta}{\theta} x_k > 0$, i.e., there is at most one k such that $x_k \in [-3\theta h, -\theta h]$ and $\frac{\partial x_{k+1}}{\partial x_k} = -\frac{1-\theta}{\theta} \neq \frac{1}{3}$. Therefore, we conclude that no matter which $\theta \in [0, 1]$ and step size $h > 0$ we pick, we cannot make $\frac{\partial x_N}{\partial x_0}$ approach the true value of $\frac{\partial x(2,x_0)}{\partial x_0} = \frac{1}{3}$. In other words, the numerical sensitivities are always incorrect.

The consequences of this are illustrated on a simple optimal control problem from [259]:

$$\min_{x_0, x(\cdot)} \quad \int_0^T x(t)^2 \mathrm{d}t + (x(2) - 5/3)^2 \tag{5.7a}$$

$$\text{s.t.} \quad x(0) = x_0, \tag{5.7b}$$

$$\dot{x}(t) \in 2 - \operatorname{sign}(x(t)), \quad t \in [0, 2]. \tag{5.7c}$$

Here, the free initial value x_0 is the only effective degree of freedom. Denote by $V(x_0)$ the (nonsmooth) objective value for the unique feasible trajectory

starting at $x(0) = x_0$. The equivalent reduced problem reads as

$$\min_{x_0 \in \mathbb{R}} V(x_0).$$

Figure 5.1 show the objective approximation obtained with the midpoint rule for different step sizes h compared to the exact value. One can observe, by decreasing the step size h, the objective values converge to the correct values but the sensitivities do not. Moreover, the wrong sensitivities introduce numerous artificial local minima.

5.2.3 Smooth approximations of NSD2 systems

An obvious and reasonable approach to solve an OCP with an NSD2 system is to regard a smooth approximation of the nonsmooth system (5.1) and then to apply the methods for smooth OCPs from Chapter 3. For this purpose, we regard smooth approximations of the DI (5.1), as studied in [259, 285].

Definition 5.1. *We say that the monotonously increasing function $\alpha_\sigma(z) \coloneqq \alpha_1(z/\sigma)$, with $\alpha_1(z) \in \mathcal{C}^\infty$ is a smoothing function if: a) $\alpha_1(0) = 1/2$, b) $\lim_{z\to\infty} \alpha_1(z) = 1$ c) $\lim_{z\to-\infty} \alpha_1(z) = 0$, with the smoothing parameter $\sigma \in \mathbb{R}_{>0}$. For the special case when $\sigma = 0$, we define $\alpha_0(z) : \mathbb{R} \to \mathcal{P}(\mathbb{R})$ to be the set-valued step function, i.e., $\alpha_0(z) = \gamma(z)$.*

The function $\alpha_\sigma(z)$ does not need to be exactly one for $z \geq \sigma$ and exactly zero for $z \leq -\sigma$. However, this definition of the smoothing functions simplifies the analysis [259, 285]. The function must approach quickly the corresponding value whenever $z \notin [-\sigma, \sigma]$. Examples of such functions are: $\alpha_\sigma(z) = (1 + \tanh(z/\sigma))/2$ or $\alpha_\sigma(z) = (z/\sigma + \sqrt{1 + (z/\sigma)^2})/2\sqrt{1 + (z/\sigma)^2}$.

The smoothed approximation of (5.1) as

$$\dot{x}(t) = f_1(x)(1 - \alpha_\sigma(\psi(x))) + f_2(x)\alpha_\sigma(\psi(x)). \tag{5.8}$$

Under certain assumptions, the solutions of the smoothed system (5.8) converge to the trajectories of the DI (5.1), cf. [285, Lemma 1].

To use direct methods for OCPs with the nonsmooth system (5.1) or with its smooth approximation (5.8), one of the crucial ingredients is to compute the correct numerical sensitivities. A remarkable result from [259] provides the condition under which smoothing will deliver meaningful approximations.

Theorem 5.2 (Theorem 2, [259])**.** *Assume that $\nabla\psi(x)^\top f_1(x) > 0$ and $\nabla\psi(x)^\top f_2(x) > 0$, whenever $\psi(x) = 0$. Denote by $x_h(t, x_0)$ the numerical*

approximation of $x(t, x_0)$. *Assume that we integrate the smoothed model equation* (5.8) *using the explicit Euler method with a time step* $h = o(\sigma)$. *Then, the numerical sensitivities denoted by* $\frac{\partial x_h(t,x_0)}{\partial x_0}$ *converge to the sensitivities of the original problem* $\frac{\partial x(t,x_0)}{\partial x_0}$ *as* $\sigma \to 0$.

A similar result is proven for the trapped case: $\nabla\psi(x)^\top f_1(x) > 0$, $\nabla\psi(x)^\top f_2(x) < 0$, whenever $\psi(x) = 0$, cf. [259, Theorem 3]. Unfortunately, using higher-order integration schemes does not circumvent the condition $h = o(\sigma)$ as discussed in [259] and confirmed in the example later in Section 5.2.5. We remind the reader that we have seen in Section 4.2.5 that higher order integration methods applied to smooth and stiff approximations recover their accuracy properties when $h = o(\sigma)$.

5.2.4 The bimodal system as a DCS

We have seen in Section 4.3 that it is ubiquitous in direct optimal control to transform the NSD2 system into an equivalent Dynamic Complementarity System (DCS). Let us derive this reformulation for the bimodal system (5.2). The set-valued step function $\alpha_0(\psi(x))$ can be expressed as the solution map of a parametric LP:

$$\alpha_0(z) = \arg\min_{\hat{\alpha}} -z\hat{\alpha} \text{ s.t. } 0 \leq \hat{\alpha} \leq 1. \tag{5.9}$$

Using the KKT conditions of the LP (5.9), the DI (5.2) can be rewritten as a DCS:

$$\dot{x}(t) = F(x(t), \alpha(t)) \coloneqq f_1(x(t))(1 - \alpha(t)) + f_2(x(t))\alpha(t), \tag{5.10a}$$

$$\psi(x) = z(t), \tag{5.10b}$$

$$z(t) = \lambda^{\mathrm{p}}(t) - \lambda^{\mathrm{n}}(t), \tag{5.10c}$$

$$0 \leq \lambda^{\mathrm{n}}(t) \perp \alpha(t) \geq 0, \tag{5.10d}$$

$$0 \leq \lambda^{\mathrm{p}}(t) \perp 1 - \alpha(t) \geq 0. \tag{5.10e}$$

This is a common approach to rewrite a Filippov system into an equivalent DCS. We come back to this reformulation for more general piecewise smooth systems in Chapter 6.

After the time discretization of the OCP subject to the DCS (5.10), we obtain an MPCC that can be efficiently solved via the methods from Section 2.3. In

MPCC solutions methods, the discretized complementarity constraints in (5.10) are smooth or relaxed. We restrict our analysis to the case of Scholtes relaxation and smoothing method [237], as it is usually the most reliable method from the ones listed in Section 2.3.

We regard the different representations of the graph of the set-valued function $\alpha_0(z)$, denoted as:

$$C^0 \coloneqq \{(z, \alpha) \mid \alpha \in \alpha_0(z)\}. \tag{5.11}$$

The graph of the function described by the LP (5.9) reads as

$$\begin{aligned} C^{\mathrm{DCS}}_{\mathrm{exact}} \coloneqq \{(z, \lambda^{\mathrm{n}}, \lambda^{\mathrm{p}}, \alpha) \mid\ & z = \lambda^{\mathrm{p}} - \lambda^{\mathrm{n}},\ \lambda^{\mathrm{n}}\alpha = 0, \\ & \lambda^{\mathrm{p}}(1-\alpha) = 0,\ \lambda^{\mathrm{n}}, \lambda^{\mathrm{p}} \geq 0,\ 0 \leq \alpha \leq 1\}. \end{aligned} \tag{5.12}$$

We define the graphs of the corresponding smoothed and relaxed problems, which fit into Definition (5.1), respectively:

$$\begin{aligned} C^{\mathrm{DCS}}_{\mathrm{smoothed}}(\sigma) \coloneqq \{(z, \lambda^{\mathrm{n}}, \lambda^{\mathrm{p}}, \alpha) \mid\ & z = \lambda^{\mathrm{p}} - \lambda^{\mathrm{n}},\ \lambda^{\mathrm{n}}\alpha = \sigma, \\ & \lambda^{\mathrm{p}}(1-\alpha) = \sigma,\ \lambda^{\mathrm{n}}, \lambda^{\mathrm{p}} \geq 0,\ 0 \leq \alpha \leq 1\}, \end{aligned} \tag{5.13}$$

and

$$\begin{aligned} C^{\mathrm{DCS}}_{\mathrm{relaxed}}(\sigma) \coloneqq \{(z, \lambda^{\mathrm{n}}, \lambda^{\mathrm{p}}, \alpha) \mid\ & z = \lambda^{\mathrm{p}} - \lambda^{\mathrm{n}},\ \lambda^{\mathrm{n}}\alpha \leq \sigma, \\ & \lambda^{\mathrm{p}}(1-\alpha) \leq \sigma,\ \lambda^{\mathrm{n}}, \lambda^{\mathrm{p}} \geq 0,\ 0 \leq \alpha \leq 1\}. \end{aligned} \tag{5.14}$$

Now we highlight a few simple relations between the sets defined above.

Proposition 5.3. *The projection of the set $C^{\mathrm{DCS}}_{\mathrm{exact}}$ onto $\mathbb{R}^2$, where C^0 is defined, is equal to C^0.*

Proof. This relation can be seen from the LP representation of the function $\alpha_0(z)$ (5.9). For $z < 0$ we have $y = 0$ and for $z > 0$, the solution is $y = 1$. If we have $z = 0$, the solution is the whole feasible set $y \in [0, 1]$. □

Proposition 5.4. *For the sets $C^{\mathrm{DCS}}_{\mathrm{exact}}$, $C^{\mathrm{DCS}}_{\mathrm{relaxed}}(\sigma)$ and $C^{\mathrm{DCS}}_{\mathrm{smoothed}}(\sigma)$, if the same $\sigma \geq 0$ is used in the latter two, the following holds:*

$$C^{\mathrm{DCS}}_{\mathrm{exact}} \subseteq C^{\mathrm{DCS}}_{\mathrm{relaxed}}(\sigma),\ \textit{and}\ C^{\mathrm{DCS}}_{\mathrm{smoothed}}(\sigma) \subseteq C^{\mathrm{DCS}}_{\mathrm{relaxed}}(\sigma).$$

Proof. The set $C^{\mathrm{DCS}}_{\mathrm{relaxed}}$ can equivalently be written as

$$\begin{aligned} C^{\mathrm{DCS}}_{\mathrm{relaxed}}(\sigma) \coloneqq \{(z, \lambda^{\mathrm{n}}, \lambda^{\mathrm{p}}, \alpha) \mid\ & z = \lambda^{\mathrm{p}} - \lambda^{\mathrm{n}},\ \lambda^{\mathrm{n}}\alpha \leq \sigma, \\ & \lambda^{\mathrm{p}}(1-\alpha) \leq \sigma,\ \lambda^{\mathrm{n}}, \lambda^{\mathrm{p}} \geq 0,\ 0 \leq \alpha \leq 1\}. \end{aligned} \tag{5.15}$$

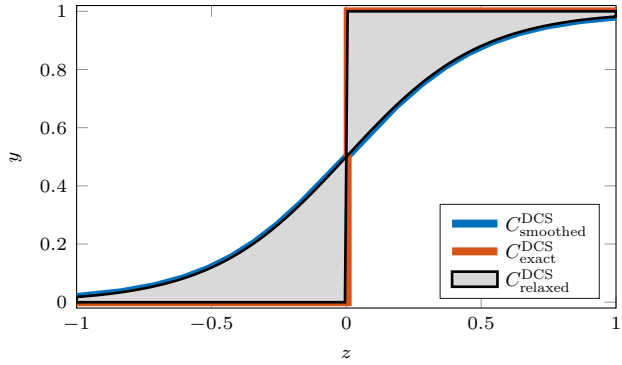


Figure 5.2: Illustrations of the projections of the sets $C^{\mathrm{DCS}}_{\mathrm{exact}}$, $C^{\mathrm{DCS}}_{\mathrm{relaxed}}$ and $C^{\mathrm{DCS}}_{\mathrm{smoothed}}$.

It follows from the definition of $C^{\mathrm{DCS}}_{\mathrm{exact}}$ in (5.12) that any vector $w_{\mathrm{exact}} := (z, \lambda^{\mathrm{n}}, \lambda^{\mathrm{p}}, \alpha) \in C^{\mathrm{DCS}}_{\mathrm{exact}}$ satisfies also the equations above, hence $w_{\mathrm{exact}} \in C^{\mathrm{DCS}}_{\mathrm{relaxed}}$. Pick any $\delta > 0$ such that $\delta < \sigma$. Using the definition (5.15) for any $w_\delta \in C^{\mathrm{DCS}}_{\mathrm{relaxed}}(\delta)$ it also holds that $w_\delta \in C^{\mathrm{DCS}}_{\mathrm{relaxed}}(\sigma)$. We conclude that $C^{\mathrm{DCS}}_{\mathrm{exact}} \subset C^{\mathrm{DCS}}_{\mathrm{relaxed}}$. Equality follows from picking $\sigma = 0$ in (5.15), which concludes the first part of the proof.

For the second relation, equality follows trivially for taking inequalities as equalities in (5.14) for the same σ as in (5.13). Furthermore, $C^{\mathrm{DCS}}_{\mathrm{smoothed}} \subset C^{\mathrm{DCS}}_{\mathrm{relaxed}}$ follows with the same argument as in the first part of the proof, by picking an appropriate $\delta > 0$ and $\delta < \sigma$. □

Figure 5.2 illustrates the result of Proposition 5.4. Proposition 5.3 implies an equivalence between the solutions of the Filippov DI (5.1) and the DCS in (5.10). Consequently, the results from Theorem 5.2 for the approximated Filippov DI (5.8) transfer to the smoothed or relaxed DCS. This means when discretizing a smoothed or relaxed version of (5.10), we need a step size of $h = o(\sigma)$ to obtain correct numerical sensitivities.

5.2.5 Failure of standard direct optimal control

To demonstrate the limits of direct optimal control approaches, we regard a discretized version of the OCP (5.7). Note that the solution $x(t; x_0)$ is piecewise constant, and the objective is a cubic function of time. Therefore, if there were locally no switches or if their time of occurrence was known, a third-order

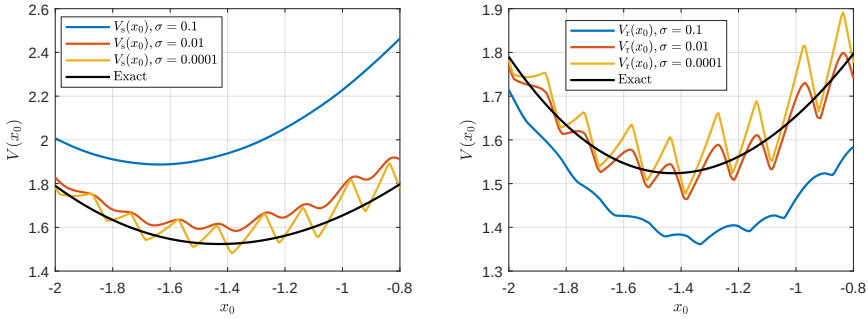


Figure 5.3: Value of the objective functions $V_{\mathrm{s}}(x_0)$ for the smoothed MPCC (left), and $V_{\mathrm{r}}(x_0)$ for the relaxed MPCC (right), for several values of the homotopy parameter σ.

method would approximate the solution exactly. To integrate the objective function with the same accuracy as the trajectory, we introduce the quadrature state:

$$\dot{y}(t) = x(t)^2,\ y(0) = 0.$$

In the regarded numerical example, we use a Gauss-Legendre method with two stage points, $n_s = 2$, which has the order four, cf. Section 3.2. We reformulate the dynamics into DCS as in (5.10) and discretize the OCP (5.7). This discretization corresponds to direct transcription (cf. Section 3.3.2):

$$\min_{w} \quad y_N + (x_N - 5/3)^2 \tag{5.16a}$$

$$\text{s.t.} \quad y_0 = 0, \tag{5.16b}$$

$$y_{k+1} = y_k + h \sum_{j=1}^{n_s} b_j (x_{k,j})^2,\ k = 0, \dots, N-1, \tag{5.16c}$$

$$x_{k+1} = x_k + h \sum_{j=1}^{n_s} b_j (3(1 - \alpha_{k,j}) + \alpha_{k,j}),\ k = 0, \dots, N-1, \tag{5.16d}$$

$$x_{k,i} = x_k + h \sum_{j=1}^{n_s} a_{i,j} (3(1 - \alpha_{k,j}) + \alpha_{k,j}),\ k = 0, \dots, N-1,\ i = 1, \dots, n_s, \tag{5.16e}$$

$$\psi(x_{k,j}) = \lambda^{\mathrm{p}}_{k,j} - \lambda^{\mathrm{n}}_{k,j},\ k = 0, \dots, N-1,\ j = 1, \dots, n_s, \tag{5.16f}$$

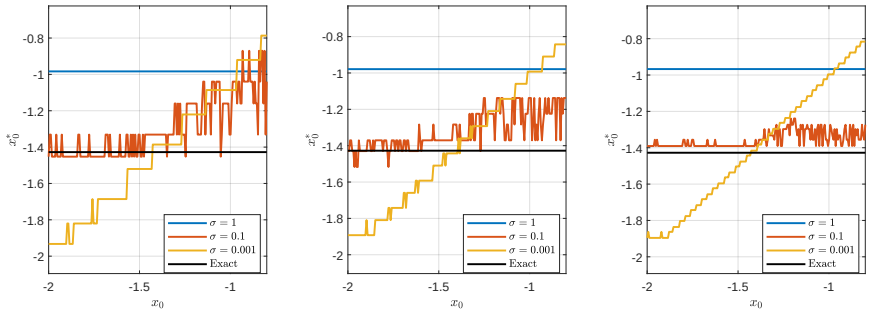


Figure 5.4: Value of the optimal solution x_0^* against different initial values x_0 for which an initial feasible solution The MPCC is solved by a relaxation method for different initial values of σ. The left plot is for $h = 0.1$, the middle for $h = 0.05$ and the right plot for $h = 0.02$.

$$0 \leq \lambda_{k,j}^{\mathrm{n}} \perp \lambda_{k,j}^{\mathrm{p}} \geq 0, \;\; k = 0, \ldots, N-1, \;\; j = 1, \ldots, n_s, \tag{5.16g}$$

$$0 \leq \lambda_{k,j}^{\mathrm{p}} \perp 1 - \lambda_{k,j}^{\mathrm{p}} \geq 0, \;\; k = 0, \ldots, N-1, \;\; j = 1, \ldots, n_s. \tag{5.16h}$$

In $w = (x_0, y_0, x_{0,1}, \alpha_{0,1}, \lambda_{0,1}^{\mathrm{n}}, \lambda_{0,1}^{\mathrm{p}}, \ldots, x_{0,n_s}, \alpha_{0,n_s}, \lambda_{0,n_s}^{\mathrm{n}}, \lambda_{0,n_s}^{\mathrm{p}}, \ldots, y_N, x_N)$, we collect all optimization variables. The NLP (5.16) is an MPCC, which is solved in a homotopy approach, where the complementarity constraints are smoothed or relaxed, cf. Section 2.3. For sake of illustration, we use $N = 20$, which results in a constant step size of $h = 0.01$. The NLPs in the homotopy loop are solved with `IPOPT` [281] through its `CasADi` [9] interface.

Figure 5.3 illustrate the values of the objective for the smoothing ($V_{\mathrm{s}}(x_0)$, left plot) and relaxation approach ($V_{\mathrm{r}}(x_0)$, right plot), respectively. We can observe that the values of the objective converge to the analytic solution, but the derivatives do not. Especially after $h \gg \sigma$, spurious local solutions appear. Similar to our simulation experiments in Section 4.2.5, we see again that very nonlinear smooth approximations behave practically as nonsmooth systems.

In our second experiment, we initialize the OCP at feasible solutions corresponding to different x_0 and see to which solution the optimizer converges. In total we take 200 samples from $x_0 \in [-2, -0.8]$. We perform this experiment for a different number of integration steps $N = 20$, $N = 40$, and $N = 100$, which results in the step sizes $h = 0.1$, $h = 0.05$, and $h = 0.02$. The MPCCs are solved with a Scholtes relaxation homotopy approach, with three different values of the initial homotopy parameter σ_0 of 1, 10^{-1}, and 10^{-4}. The final σ is always 10^{-8} and the homotopy parameter is updated via the rule: $\sigma_{n+1} = 0.1\sigma_n$, n is

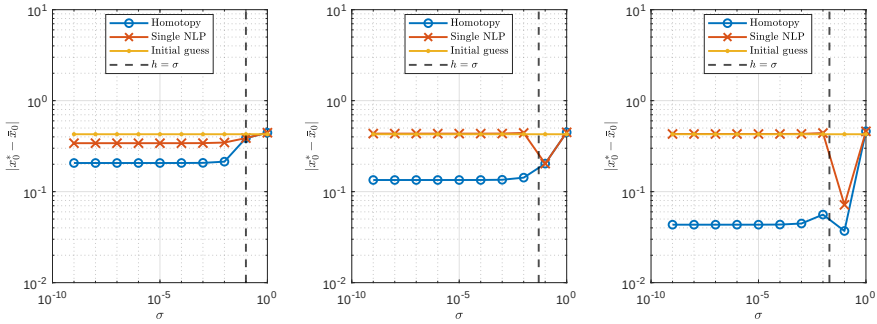


Figure 5.5: Distance of numerically computed optimal solution x_0^* to the analytic solution $\bar{x}_0$, for different initial σ. The left plot is for $h = 0.1$, the middle for $h = 0.05$ and the right plot for $h = 0.02$.

the number of the problem in the sequence. Figure 5.4 shows the solution of a relaxed MPCC corresponding to (5.16) for different initial guesses for x_0 and different starting values of σ_0. As long as the condition $h = o(\sigma)$ is satisfied one can see the NLP solver converges to a unique local solution. As soon as $h \gg \sigma$, the solver converges to the nearest spurious local solution, which results in the stair-like curves in Figure 5.3. In practice, smoothing and relaxation might work well due to the homotopy, since for larger values of σ, the derivatives are correct and the optimizer gets into a "good region". However, the numerical sensitivity is wrong as soon as $h = o(\sigma)$ becomes violated.

In our last experiment, we compare the solution obtained after every homotopy iteration with relaxed complementarity constraints to the solutions obtained by a single NLP solve with the same fixed σ. As an initial guess, we take $x_0 = -1$. In the homotopy approach, the parameter σ is updated with the following rule: $\sigma_{n+1} = 0.1\sigma_n$, with $\sigma_0 = 1$, n is the number of the problem in the sequence. Figure 5.5 illustrates the results for three different step sizes. One can observe that with the homotopy we get in a good region, and even for smaller σ, the approximate solution x_0^* is close to the analytic solution $\bar{x}_0$. On the other hand, with the single NLP solve, as soon as $h = o(\sigma)$ is violated, the NLP converges to the nearest spurious local solution.

We can see that for $h \ll \sigma$, the homotopy approach and single NLP find the same unique solution. However, as this criterion becomes violated the single NLP approach makes sometimes almost no progress. Even the homotopy approach delivers inaccurate solutions, despite a small step size. If the homotopy converges to the "best artificial minimum", the error is still of order $O(h)$ since the switches are not detected. The standard direct methods fail catastrophically even on a

simple example. Nevertheless, all obtained solutions are at least feasible, which might explain why these approaches sometimes lead to satisfactory results in practice. In Section 7.3.6, we repeat the experiments above and manage to resolve the accuracy issues with a novel discretization method that will be introduced in Chapter 7.

5.3 Limitations of direct methods for NSD3 systems

In this section, we illustrate that direct methods for systems with state jumps suffer from the same limitations as in the NSD2 case. We focus on the class of systems of nonsmooth mechanics, which is arguably the most widely studied class of NSD3 systems. In Chapter 8, we study these problems in more detail and develop methods that resolve the issues that we highlight below.

As discussed in Section 5.1, contact implicit trajectory optimization (CITO), i.e., direct optimal control with rigid body models subject to impacts and friction is a topic of growing popularity in the robotics community. This gave rise to many physics simulations software packages, which essentially have at their core a time-stepping method [255] for the simulation of the nonsmooth equations of motion [141, 265, 268, 284]. Most of them provide also sensitivity information w.r.t. to initial values and controls. In the robotics community, this is often known under the term *differentiable physics*. This is somewhat a misnomer since the sensitivities are nonsmooth, cf. Section 4.2.6. Zhong et. al. [292] show in several experiments in which the exact sensitivities are known, that none of the widely used software packages computes them correctly. Moreover, the results obtained by different open-source implementations do not agree.

Some of the packages detect the switching time (time of impact) explicitly, which improved the situation only on simple examples. However, on even slightly more complex tasks, the sensitivities are wrong in all cases [292]. There is no reason to believe that just detecting the switching time will lead to the computation of correct sensitivities by differentiating the integrator. The sensitivity formulae are more complicated, and an explicit jump formula must be derived from the switching conditions to update the sensitivity past points of discontinuity [25, 40]. This is also the case for NSD2 systems, cf. Sections 4.2.6 and 6.2.6.

Nevertheless, these methods often perform well in practice. There are several reasons for this. On the one hand, during the solution process, e.g., in a homotopy loop, the dynamics are smoothed, and the sensitivities are correct and direct the solver to a good region. On the other hand, Zhong et al. [292]

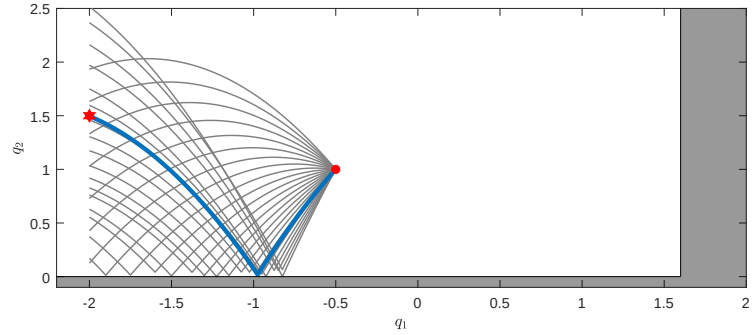


Figure 5.6: Several feasible trajectories of the OCP (5.17). The blue trajectory is an optimal solution that has one state jump.

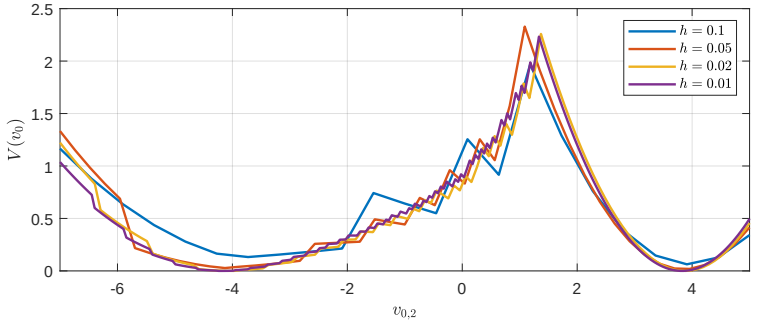


Figure 5.7: The objective function value $V(v_0)$ for different step sizes h. For values of $v_{0,2}$ that do not lead to impacts, the system is locally smooth, and the sensitivities are correct (right part of the graph).

observe that the sensitivities, despite being wrong in their value, have the correct sign and thus lead to descent directions.

Here, we recall one optimal control example from [292] (Task 2) to illustrate that the sensitivities do not converge. We refer the interested reader to the excellent paper of Zhong et al. [292] for more details and further examples. The goal is to find an initial velocity v_0 of a two-dimensional particle such that it reaches from the initial position $q_0 = (-0.5, 1)$ the target position $q_{\text{goal}} = (-2, 1.5)$ in $T = 0.6$. The impact is elastic with $\epsilon_{\text{r}} = 0.92$, and no control forces are applied. The optimal control problem reads as

$$\min_{v_0, x(\cdot), \lambda(\cdot)} \quad \|q(T) - q_{\text{goal}}\|^2 \tag{5.17a}$$

$$\text{s.t.} \quad q(0) = q_0, v(0) = v_0 \tag{5.17b}$$

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

$$\dot{v}(t) = \begin{bmatrix} -\lambda_1(t) \\ -g + \lambda_2(t) \end{bmatrix}, \tag{5.17d}$$

$$0 \leq f_1(q(t)) \perp \lambda_1(t) \geq 0, \tag{5.17e}$$

$$0 \leq f_2(q(t)) \perp \lambda_2(t) \geq 0, \text{ for a.a. } t \in [0, T], \tag{5.17f}$$

$$\text{if } f_i(t) = 0 \text{ and } \nabla_q f_i(q(t))^\top v(t^-) \leq 0, \text{ then} \tag{5.17g}$$

$$\nabla_q f_i(q(t))^\top v(t^+) = -\epsilon_\mathrm{r} \nabla_q f_i(q(t))^\top v(t^-), \ i \in \{1, 2\}, \tag{5.17h}$$

with the gap functions $f_1(q) = 1.6 - q_1$ and $f_2(q) = q_2$. The only effective degree of freedom is the initial velocity v_0, and we can look at the reduced problem:

$$\min_{v_0} V(v_0)$$

We discretize the OCP (5.17) with the Anitescu-Potra method [15], a standard time-stepping method for elastic impacts, fix the horizontal velocity $v_{0,1} = -2.5$ and evaluate the objective function for different $v_{0,2}$. Figure 5.6 illustrates numerous feasible trajectories with and without impacts. In Figure 5.7, we illustrate the reduced objective as a function of $v_{0,2}$ for different step sizes. Similar to the NSD2-case, we can observe that the trajectories converge, but the derivatives do not. Depending on the step size, for $v_{0,2}$ greater than approximately one, no impacts happen, and the system is locally smooth. For the other values, the objective experiences numerous artificial local minima due to the wrong sensitivities.

5.4 Conclusion and summary

Despite the natural and intuitive way of modeling discrete-continuous effects with nonsmooth differential equations, it is challenging to handle them numerically in optimal control. Due to the nonsmoothness most standard numerical simulation methods have only first-order accuracy [258]. Furthermore, the sensitivities of the solution trajectories of these dynamic systems are nonsmooth w.r.t. the initial value and parameters. Because of these difficulties, the powerful tools from smooth optimization and direct optimal control for smooth differential equations cannot be readily applied to this class of problems. Reliable tailored numerical algorithms and accompanying software implementations are not yet

available or not yet as widespread as for optimal control with smooth dynamical systems.

In this chapter, we have seen that standard direct methods fail on even very simple optimal control problems. Differentiating time-stepping discretization yields wrong sensitivities no matter how small the step size is. Smoothing works only if the step size is sufficiently small to resolve the severe nonlinearity introduced by smoothing. On the other hand, the smoothed systems can give accurate numerical sensitivities, provided that $h = o(\sigma)$. From an implementation point of view, one can use standard integrator codes in direct multiple shooting or direct collocation and a standard NLP solver. However, to obtain a more accurate solution with this approach, one has to take a very small step size. For example, if one wishes to have an integration accuracy of 10^{-6}, the smoothing parameter has to be in the same order. Since $h = o(\sigma)$, for a prediction horizon of $T = 1$, the discretized optimal control problem has to have millions of variables. This makes this approach impractical.

Moreover, the wrong sensitivities can lead to artificial local minima, and the optimizer might get stuck arbitrarily close to the initial guess. Reformulating e.g., NSD2 systems into equivalent dynamic complementarity systems and solving Mathematical Programs with Complementarity Constraints (MPCC) with tailored methods does not improve the situation at all. Due to the equivalence, the same difficulties remain. To obtain correct numerical sensitivities, one must have a step size of $h = o(\sigma)$, where σ is the relaxation or smoothing parameter in the MPCC.

Nevertheless, in practice, standard direct methods sometimes deliver satisfactory results. The reasons is that Newton-type optimization with completely wrong sensitivities but correct function evaluations leads to convergence to suboptimal but feasible solutions [42]. This is sometimes called as zero-order optimization, as only the functions zero-order information (i.e., the function evaluations) is correct. However, these solutions are not optimal and have only first-order accuracy. Estimating the loss of optimality in this case is generally not impossible. Hence, the standard methods can only be considered as heuristics in this case. Therefore, it is necessary to develop tailored methods with a sound theory that overcome the fundamental limitations of standard methods.

Chapter 6

Reformulation of Filippov Systems into Dynamic Complementarity Systems

Filippov systems are a focal point of this thesis. This chapter examines Piecewise Smooth Systems (PSS), their embedding into Filippov Differential Inclusions (DIs), and their connection to Dynamical Complementarity Systems (DCS). There are several motivations for this. First, Filippov systems arising from PSS are fairly regular systems with many favorable properties that can be exploited in algorithmic design. Second, many practical problems can be modeled as PSS. Moreover, in Chapters 8 and 9 we introduce the time-freezing reformulation. It provides an exact reformulation of several classes of systems with state jump (NSD3) into PSS (NSD2). Furthermore, NSD1 systems can be viewed as a special case of an NSD2 system, where the vector field is continuous across region boundaries. This enables a unified theoretical and algorithmic treatment of many systems of types NSD1 to NSD3.

Some basic definitions and properties of DCS and PSS are already discussed in Sections 4.3.4 and 4.3.5, respectively. However, here we invest some effort in understanding how to relate Filippov systems to DCS. We explore two reformulations to pass from a Filippov system to a DCS: Stewart's approach [250] and the Heaviside step functions approach [6, 74]. In both cases, solving a Linear Program (LP) enables the determination of the convex multipliers required to compute a selection of the Filippov set. Using the KKT conditions of this LP, we can pass from a Filippov system to an equivalent DCS. We study

these DCSs in detail, i.e., the properties of the Lagrange multipliers in the DCS, properties of the ODE/DAE obtained from fixing the active set, and what happens at active-set changes. This pays off as we can use these insights to develop the Finite Elements with Switch Detection (FESD) method in the next chapter. This discretization method possesses numerous advantageous theoretical and computational properties when applied to the resulting DCS.

Outline. The content of this chapter is as follows. First, in Section 6.1, we give definitions of the PSS, Filippov systems, and the solutions we are interested in. Next, we study Stewart's approach [250] to pass from a Filippov system to a DCS in Section 6.2. In Section 6.3, we perform a similar study for the reformulation obtained via set-valued Heaviside step functions. The chapter is concluded in Section 6.4. Section 6.2 is partially based on [213], and Section 6.3 on [207, 208].

6.1 Piecewise smooth and Filippov systems

In this section, we provide some more assumptions and definitions for PSS and Filippov systems. For ease of reading, we summarize the most important notation and symbols used throughout the chapter.

Notation

We remind the reader of some notation heavily used in this chapter. For some matrix $A \in \mathbb{R}^{n\times m}$, its i-th row is denoted by $A_{i,\bullet}$ and its j-th column is denoted by $A_{\bullet,j}$. Given a set $\mathcal{I} \subseteq \{1,\dots,n\}$, the matrix $A_{\mathcal{I},\bullet} \in \mathbb{R}^{|\mathcal{I}|\times m}$ is a submatrix of A consisting of the rows $A_{i,\bullet}$ for $i \in \mathcal{I}$. The submatrix $A_{\bullet,\mathcal{J}} \in \mathbb{R}^{n\times|\mathcal{J}|}$ is defined accordingly. For a given vector $x \in \mathbb{R}^n$ and set $\mathcal{I} \subseteq \{1,\dots,n\}$, we define the projection matrix $P_{\mathcal{I}} \in \mathbb{R}^{|\mathcal{I}|\times n}$ which has zeros or ones as entries. It selects all component $x_i, i \in \mathcal{I}$ from the vector x, i.e., $x_{\mathcal{I}} = P_{\mathcal{I}}x \in \mathbb{R}^{|\mathcal{I}|}$ and $x_{\mathcal{I}} = [x_i \mid i \in \mathcal{I}]$.

In Table 6.1, we summarize the key symbols used in this chapter.

Table 6.1: Key symbols used throughout this chapter.

Symbol	Description	First appearance
n_x	dimension of the differential state x	Eq. (6.1)
n_f	number of regions/modes in the PSS	Eq. (6.1)
n_u	dimension of the control function u	Eq. (6.1)

n_ψ	dimension of switching functions $\psi(x)$	Sec 6.2.1
n_β	number of lifting variables	Sec. 6.3.8
n_{sys}	number of subsystems in sum of Filippov systems	Sec. 6.2.5
x	differential state	Eq. (6.1)
u	control function	Eq. (6.1)
e	vector with all ones	Eq. (6.8)
θ	Filippov's convex multipliers	Eq. (6.2)
λ	Lagrange multipliers in Stewart's DCS	Eq. (6.8)
μ	Lagrange multipliers in Stewart's DCS	Eq. (6.8)
$t_{\text{s},n}$	n-th switching point	Sec. 6.2.2
S	sign matrix for a compact definition of R_i	Eq. (6.16)
α	selection of set-valued step function	Sec. 6.3.3
λ^{p}	Lagrange multiplier in step DCS	Eq. (6.44)
λ^{n}	Lagrange multiplier in step DCS	Eq. (6.44)
β	Lifting variable in the step DCS	Sec. 6.3.8
$f_i(x,u)$	modes of the PSS system	Eq. (6.1)
$g(x)$	Stewart's indicator functions	Eq. (6.4)
$\Psi(\cdot,\cdot)$	C-function	Def. 2.23, Eq. (6.9)
$F(x,u)$	a matrix in $\mathbb{R}^{n_x \times n_f}$ that collects all PSS modes	Eq. (6.6)
$G(x,\theta,\lambda,\mu)$	algebraic equations in Stewart's DCS as nonsmooth DAE	Eq. (6.9)
$G(x,\theta,\alpha,\lambda_{\text{p}},\lambda_{\text{n}})$	algebraic equations in the step DCS as nonsmooth DAE	Eq. (6.45)
$\psi(x)$	switching functions	Sec. 6.2.1, 6.3.3
$M_{\mathcal{I}}(x)$	Stewart's matrix for the study of the fixed-active DCS and active set prediction LCP	Eq. (6.16)
$\psi_{i,j}(x)$	local switching function	Eq. (6.23)
$S^x(t,0,x_0)$	sensitivity matrix	Sec. 6.2.6
$J(x(t_{\text{s}};x_0))$	sensitivity jump matrix	Eq. (6.25)
$\gamma(x)$	scalar set-valued step function	Eq. (6.32)
$\Gamma(x)$	vector-valued version of the step function	Eq. (6.32)
$W_{\mathcal{K},\mathcal{I}}(x)$	auxiliary matrix used in the study of the step DCS (6.44)	Sec. 6.3.5
$B_{\mathcal{K},\mathcal{I}}(\alpha)$	auxiliary matrix used in the study of the step DCS (6.44)	Sec. 6.3.5
$G_\beta(\alpha,\beta)$	expression relating α and lifting variables β	Sec. 6.3.8

$G_\theta(\theta, \alpha, \beta)$	lifted expression for θ	Sec. 6.3.8
$G_{\text{Lift}}(\theta, \alpha, \beta)$	concatenation of $G_\beta(\alpha, \beta)$ and $G_\theta(\theta, \alpha, \beta)$	Sec. 6.3.8
R_i	regions of the PSS	Eq. (6.1)
∂R_i	boundary of regions of the PSS	Sec. 6.1.1
$\mathcal{J}$	index set of PSS modes	Eq. (6.1)
$\mathcal{F}_{\text{F}}(x, u)$	Filippov set	Eq. (6.2), Eq. (6.3)
$\mathcal{F}_{\text{S}}(x, u)$	Filippov set via step functions	Eq. (6.42)
$\mathcal{I}(x)$	active set for Filippov systems	Eq. (6.5)
$\mathcal{T}_n$	$n-$th time interval with fixed active set $\mathcal{T}_n = (t_{\text{s},n}, t_{\text{s},n+1})$	Sec. 6.2.2
$\mathcal{I}_n$	the fixed active set $\mathcal{I}_n = \mathcal{I}(x(t)), t \in I_n$	Sec. 6.2.2
$\mathcal{I}_n^0$	active set at $t_{\text{s},n}$, i.e. $\mathcal{I}_{n,0} = \mathcal{I}(t_{\text{s},n})$	Sec. 6.2.2
$\mathcal{C}$	index set of all switching functions $\psi_i(x)$	Sec. 6.3.3
$\mathcal{C}_i$	index set of all switching functions involved in the definition of R_i	Sec. 6.3.3
$\mathcal{K}$	index set of all switching functions $\psi_i(x)$ that are zero for a given $\mathcal{I}$	Sec. 6.3.5

6.1.1 Piecewise smooth differential equations

This section introduces some assumptions on the PSS and its Filippov convexification [93]. We have already discussed ODEs with a Discontinuous Right-Hand Side (DRHS) in Section 4.3.5. Here we treat piecewise smooth systems that are such ODEs but where the discontinuities appear in a structured way. The generic PSS reads as

$$\dot{x}(t) = f_i(x(t), u(t)), \text{ if } x(t) \in R_i \subset \mathbb{R}^{n_x},\ i \in \mathcal{J} := \{1, \ldots, n_f\}, \tag{6.1}$$

where R_i are disjoint, connected, and open sets. They are assumed to be nonempty and to have piecewise smooth boundaries ∂R_i. We assume that $\overline{\bigcup_{i \in \mathcal{J}} R_i} = \mathbb{R}^{n_x}$ and that $\mathbb{R}^{n_x} \setminus \bigcup_{i \in \mathcal{J}} R_i$ is a set of measure zero. The functions $f_i(\cdot)$ are assumed to be Lipschitz and at least twice continuously differentiable functions on an open neighborhood of $\overline{R}_i$. Furthermore, n_f is a positive integer, and $u(t)$ is a sufficiently regular externally chosen control function.

Many practical problems give rise to ODE as in (6.1), e.g., in sliding mode control [5], mechanics problems with Coulomb friction [255], state-constrained ODE derived from Pontryagin's maximum principle [224], electronic circuits [3], biological systems [6], vaccination strategies [67], transportation systems and traffic flow networks [21], constrained optimization algorithms viewed as

dynamic systems [132] and many more. Many interesting results about PSS are collected in the monograph [34]. As we will see in Chapter 8 and 9, systems with state jumps, including impact mechanics, robotics, and hybrid systems with hysteresis can be transformed into systems matching the form of (6.1) via the *time-freezing reformulation* [202, 205, 212]. Consequently, efficient and accurate numerical optimal control algorithms for this class of systems are of great interest.

6.1.2 Filippov convexification

Initial value problems arising from the nonsmooth ODE (6.1) usually fail to have classic Carathéodory solutions, for a counterexample, cf. Section 4.3.5. As already discussed, to have a meaningful solution concept for this class of ODEs, they are replaced by a differential inclusion. Here we follow Filippov's embedding of such ODE into DIs as it has many advantageous theoretical properties, cf. 4.3.5. The main idea of Filippov is to replace the r.h.s. of (6.1) with a bounded convex set. For the readers convince we restate the definition in Eq. (4.16) for (6.1) and obtain the following Differential Inclusion (DI):

$$\dot{x}(t) \in \mathcal{F}_{\mathrm{F}}(x(t), u(t)) \coloneqq \bigcap_{\delta>0} \bigcap_{\mu(N)=0} \overline{\mathrm{conv}} f(x + \delta B(x) \setminus N, u(t)), \tag{6.2}$$

where $\mathcal{B}(x)$ is the Euclidean unit ball at x in $\mathbb{R}^{n_x}$, $\mu(\cdot)$ is the Lebesgue measure on $\mathbb{R}^{n_x}$ and $\overline{\mathrm{conv}}(\cdot)$ maps a subset of $\mathbb{R}^{n_x}$ to its closed convex hull.

Let $\mathcal{I}(x) \coloneqq \{i \mid x \in \overline{R}_i\} \subseteq \mathcal{J}$ be the *active-set* at $x \in \mathbb{R}^{n_x}$. Due to the special structure of (6.1) the Filippov DI (6.2) can be written as

$$\dot{x} \in \mathcal{F}_{\mathrm{F}}(x, u) = \overline{\mathrm{conv}}\{f_i(x, u) \mid i \in \mathcal{I}(x)\}.$$

This means that in the interior of the regions R_i, the Filippov set $\mathcal{F}_{\mathrm{F}}(x, u)$ is equal to $\{f_i(x)\}$ and on the boundary between regions, we have a convex combination of the neighboring vector fields. If $\dot{x}$ exists, functions $\theta_i(\cdot)$ which serve as convex multipliers can be introduced, and the Filippov DI can be written as [250]:

$$\dot{x} \in \mathcal{F}_{\mathrm{F}}(x, u) = \Big\{ \sum_{i\in\mathcal{J}} f_i(x, u)\, \theta_i \mid \sum_{i\in\mathcal{J}} \theta_i = 1,\ \theta_i \geq 0,\ \theta_i = 0 \text{ if } x \notin \overline{R_i}, \forall i \in \mathcal{J} \Big\}. \tag{6.3}$$

We call the functions $\theta_i(\cdot)$ *Filippov multipliers*. As it will be seen later, the functions $\theta_i(\cdot)$ lack any continuity properties. But it can be shown that they are at least measurable [92, 250]. Given (6.3), we will compute *piecewise active* solutions [250], which are defined as follows.

Definition 6.1 (Piecewise active solution [250])**.** *For an initial value $x(0) = x_0$, a given measurable control function $u(t)$ and a compact interval $[0, T]$, a function $x : [0, T] \to \mathbb{R}^{n_x}$ is said to be a solution of (6.2), if $\dot{x}(t) \in \mathcal{F}_{\mathrm{F}}(x(t), u(t))$ almost everywhere on $[0, T]$. This function is called a piecewise active solution if the active-set $\mathcal{I}(x(t))$ is a piecewise constant function of time, and it changes its value only finitely many times on $[0, T]$. A time point $t_{\mathrm{s}} \in [0, T]$ is called a switching point if $\mathcal{I}(x(t))$ is not constant in any sufficiently small neighborhood of t_{s}.*

The overall ODE solution $x(t)$ is continuous and consists of smooth pieces connected by nondifferentiable points ("kinks") at the switching times t_{s}. Note that this definition excludes Zeno's phenomenon, i.e., solutions with infinitely many switches in finite time, cf. Section 4.2.1.

In the following two sections, we show that, given a finite representation of the regions R_i via sufficiently smooth constraints functions $c : \mathbb{R}^{n_x} \to \mathbb{R}^m$, we can compute the Filippov multipliers θ for every x as the solution of a linear program. Moreover, we will see that for a constant active set $\mathcal{I}(x)$ one can derive an ODE or Differential Algebraic Equation (DAE) (for sliding modes) from (6.3).

6.2 Stewart's reformulation

One elegant way how to compute the Filippov multipliers as a solution of a Linear Program (LP) is introduced by Stewart in [250, 260]. Using the KKT conditions of this LP, we can pass from the Filippov system in (6.3) to a Dynamic Complementarity System (DCS). In the sequel, we study this DCS in detail, both for a fixed active set and at active-set changes. The properties of the DCS lay the foundations for the development of high-accuracy discretization methods in the next chapter.

In Stewart's reformulation, the main assumption is that the regions R_i are given as

$$R_i = \{x \in \mathbb{R}^{n_x} \mid g_i(x) < \min_{j \in \mathcal{J} \setminus \{i\}} g_j(x)\}. \tag{6.4}$$

We call $g_i(\cdot),\ i \in \mathcal{J}$ *Stewart's indicator functions*, and we assume they are at least twice continuously differentiable. At first look, this reformulation might not seem to be the most intuitive one. However, such a reformulation is, e.g., obtained if the state space $\mathbb{R}^{n_x}$ is tessellated into Voronoi regions, cf. Chapter 9. Furthermore, in Section 6.2.1, we provide a constructive way how to pass

from a more natural representation of R_i via switching functions $c(\cdot)$ to (6.4). Throughout this thesis we assume additionally that $g_i(\cdot), f_i(\cdot)$ and $\nabla g_i(\cdot)$ are Lipschitz continuous.

Note that due to the definition of the sets R_i in (6.4), the active set can be defined as

$$\mathcal{I}(x) \coloneqq \Big\{ i \mid g_i(x) = \min_{j \in \mathcal{J}} g_j(x) \Big\}. \tag{6.5}$$

We define the vectors $\theta = (\theta_1, \ldots, \theta_{n_f}) \in \mathbb{R}^{n_f}$, $g(x) = (g_1(x), \ldots, g_{n_f}(x)) \in \mathbb{R}^{n_f}$ and the matrix $F(x) = \big[f_1(x), \ldots, f_{n_f}(x)\big] \in \mathbb{R}^{n_x \times n_f}$. Using the specific representations (6.4), from equation (6.3), one can deduce that the Filippov DI can be written as

$$\dot{x} = F(x,u)\theta(x). \tag{6.6}$$

The algebraic variables $\theta(x)$ are a solution of the LP parameterized by x and denoted by LP(x)

$$\theta(x) \in \arg\min_{\tilde{\theta} \in \mathbb{R}^{n_f}} \quad g(x)^\top \tilde{\theta} \tag{6.7a}$$

$$\text{s.t.} \quad e^\top \tilde{\theta} = 1 \tag{6.7b}$$

$$\tilde{\theta} \geq 0. \tag{6.7c}$$

Using the Karush–Kuhn–Tucker (KKT) conditions of LP(x) and (6.6), we obtain the dynamic complementarity system

$$\dot{x} = F(x,u)\theta, \tag{6.8a}$$

$$0 = g(x) - \lambda - \mu e, \tag{6.8b}$$

$$1 = e^\top \theta, \tag{6.8c}$$

$$0 \leq \theta \perp \lambda \geq 0, \tag{6.8d}$$

where the algebraic variables $\lambda \in \mathbb{R}^{n_f}$ and $\mu \in \mathbb{R}$ are the Lagrange multipliers of the inequality and equality constraints in (6.7), respectively. To have an even more compact representation, we use a C-function Ψ (cf. Definition 2.23) for the complementarity conditions and rewrite the KKT conditions of the LP (6.7) as the nonsmooth equation:

$$G(x,\theta,\lambda,\mu) \coloneqq \begin{bmatrix} g(x) - \lambda - \mu e \\ 1 - e^\top \theta \\ \Psi(\theta,\lambda) \end{bmatrix} = 0. \tag{6.9}$$

It provides $2n_f + 1$ conditions for the $2n_f + 1$ algebraic variables θ, λ and μ. The DCS reads in compact form as a nonsmooth differential algebraic equation:

$$\dot{x} = F(x, u)\theta, \tag{6.10a}$$

$$0 = G(x, \theta, \lambda, \mu). \tag{6.10b}$$

Example 6.2. *We illustrate this formulation on the simple example of* $\dot{x} \in 2 - \operatorname{sign}(x)$. *This ODE is characterized by the regions* $R_1 = \{x \in \mathbb{R} \mid x < 0\}$ *and* $R_2 = \{x \in \mathbb{R} \mid x > 0\}$, *with* $f_1(x) = 3$, $f_2(x) = 1$ *and* $F(x) = [3 \quad 1]$. *It can be verified that with the functions* $g_1(x) = x$ *and* $g_2(x) = -x$, *we have a representation of the regions as in* (6.4). *Moreover, we have the multipliers* $\theta, \lambda \in \mathbb{R}^2$ *and* $\mu \in \mathbb{R}$. *Thus, the corresponding DCS reads as:*

$$\dot{x} = \begin{bmatrix} 3 & 1 \end{bmatrix} (\theta_1, \theta_2), \tag{6.11a}$$

$$0 = x - \lambda_1 - \mu, \tag{6.11b}$$

$$0 = -x - \lambda_2 - \mu, \tag{6.11c}$$

$$0 = \theta \perp \lambda \geq 0, \tag{6.11d}$$

$$1 = \theta_1 + \theta_2. \tag{6.11e}$$

6.2.1 How to obtain Stewart's indicator functions?

Definition (6.4) might not be the most intuitive way to represent the sets R_i. In many practical examples some smooth scalar functions $\psi_i(\cdot)$, $i = 1, \ldots, n_\psi$ called *switching functions*, are given. We use them to define the following basis regions.

Definition 6.3 (Basis regions)**.** *Given* n_ψ *scalar switching functions* $\psi_i(x)$, $i \in \mathcal{C} \coloneqq \{1, \ldots, n_\psi\}$, *we define the* 2^{n_ψ} *basis regions:*

$$R'_1 = \{x \in \mathbb{R}^{n_x} \mid \psi_1(x) > 0, \psi_2(x) > 0, \ldots, \psi_{n_\psi - 1}(x) > 0, \psi_{n_\psi}(x) > 0\},$$

$$R'_2 = \{x \in \mathbb{R}^{n_x} \mid \psi_1(x) > 0, \psi_2(x) > 0, \ldots, \psi_{n_\psi - 1}(x) > 0, \psi_{n_\psi}(x) < 0\},$$

$$\vdots$$

$$R'_{n_f} = \{x \in \mathbb{R}^{n_x} \mid \psi_1(x) < 0, \psi_2(x) < 0, \ldots, \psi_{n_\psi - 1}(x) < 0, \psi_{n_\psi}(x) < 0\},$$

such that $\mathbb{R}^{n_x} = \cup_{i=1}^{n_f} \tilde{R}_i$. *These definitions are compactly expressed via a dense sign matrix* $S \in \mathbb{R}^{2^{n_\psi} \times n_\psi}$*:*

$$S = \begin{bmatrix} 1 & 1 & \dots & 1 & 1 \\ 1 & 1 & \dots & 1 & -1 \\ \vdots & \vdots & \dots & \vdots & \vdots \\ -1 & -1 & \dots & -1 & -1 \end{bmatrix}. \tag{6.12}$$

The matrix S *has no repeating rows and no zero entries. The sets* $\tilde{R}_i$ *are defined using the rows of the matrix* S*:*

$$R_i' = \{x \in \mathbb{R}^{n_x} \mid S_{i,j}\psi_j(x) > 0, j \in \mathcal{C}\},\ i = 1, \dots, 2^{n_\psi}. \tag{6.13}$$

Note that the boundaries of the regions $\partial R_i'$ are subsets of the zero-level sets of appropriate functions $\psi_j(x)$.

The next proposition provides a constructive way to find the functions $g(\cdot)$ from the more intuitive representation of the regions via $\psi(\cdot)$.

Proposition 6.4. *Let the function* $g : \mathbb{R}^{n_x} \to \mathbb{R}^{n_f}$ *be defined as*

$$g(x) = -S\psi(x), \tag{6.14}$$

then for all $x \in R_i'$ *the following statements are true:*

(i) $g_i(x) < g_j(x)$, *for* $i \neq j$,

(ii) the definitions (6.4) *and* (6.13) *define the same set, i.e.,* $R_i = R_i'$.

Proof. For (i), note that for $x \in R_i'$ all terms in the sum $g_i(x) = -S_{i,\bullet}\psi(x) = -\sum_k S_{i,k}\psi_k(x)$ are strictly positive. On the other hand, for any $g_j(x) = -S_{j,\bullet}\psi(x) = -\sum_k S_{j,k}\psi_k(x)$, $j \neq i$ and $x \in R_i'$, due to (6.13), all terms in the sum where $S_{j,k} \neq S_{i,k}$ are strictly negative. Therefore $S_{i,\bullet}\psi(x) > S_{j,\bullet}\psi(x)$, thus (i) holds.

For (ii), first regard the rows $S_{j,\bullet}$ that differ from $S_{i,\bullet}$ only in the k-th column. Then $g_i(x) - g_j(x) = -(S_{i,k} - S_{j,k})\psi_k(x) < 0$. If $S_{i,k} = 1$, then $g_i(x) - g_j(x) = -2\psi_k(x) < 0$. Likewise, for $S_{i,k} = -1$, then $g_i(x) - g_j(x) = 2\psi_k(x) < 0$. Therefore, from (6.4), we recover the definition of (6.13) by looking at the rows where $S_{i,k}$ and $S_{j,k}$ differ by one element. For all rows j that differ from $S_{i,\bullet}$ by more than one column, by similar reasoning, we obtain inequalities that do not tighten (6.13), since $g_i(x) - g_j(x)$ consists of a sum of the terms from the inequalities where only one component of $\psi(x)$ is left. Therefore, statement (ii) holds, and this completes the proof. □

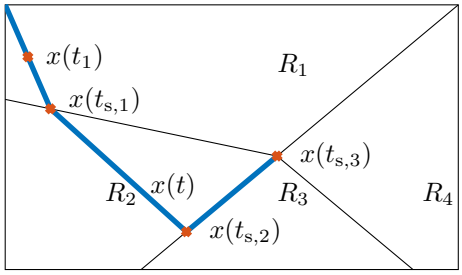


Figure 6.1: Illustration of active sets at different points. It can be seen that $\mathcal{I}(x(t_1)) = \mathcal{I}_0 = \{1\}$. At $x(t_{s,1})$ the trajectory crosses the surface of discontinuity between R_1 and R_2, hence $\mathcal{I}(x(t_{s,1})) = \mathcal{I}_1^0 = \{1, 2\}$ and later $\mathcal{I}_1 = \{2\}$. The segment between $x(t_{s,2})$ and $x(t_{s,3})$ is a sliding mode and we have $\mathcal{I}_2^0 = \{2, 3\}$ and $\mathcal{I}_2 = \{2, 3\}$. Finally we have at $x(t_{s,3})$ that $\mathcal{I}_3^0 = \{1, 2, 3, 4\}$.

Example 6.5. *For our tutorial example* $\dot{x} \in 2 - \operatorname{sign}(x)$ *and the corresponding DCS* (6.11) *we have* $\psi(x) = x$, $S = \begin{bmatrix} -1 & 1 \end{bmatrix}^\top$ *and we obtain* $g(x) = -S\psi(x) = (x, -x)$ *as used in Example 6.2.*

6.2.2 Fixed active set

For a given solution $x(\cdot)$ let us denote all switching points by $0 = t_{s,0} < t_{s,1} < \cdots < t_{s,N_{sw}} = T$. The fixed active set between two switches is denoted by $\mathcal{I}_n := \mathcal{I}(x(t))$, $t \in (t_{s,n}, t_{s,n+1}) =: \mathcal{T}_n$ and at a switching point $t_{s,n}$ by $\mathcal{I}_n^0 := \mathcal{I}(x(t_{s,n}))$. Note that $\mathcal{I}_n^0 = \mathcal{I}_n \cup \mathcal{I}_{n-1}$. These definitions are illustrated in Figure 6.1.

In this subsection, we regard the DCS (6.8) for a single fixed $\mathcal{I}_n$. To simplify our notation we denote this active set by $\mathcal{I}$ and the according time interval $\mathcal{T} = (0, T)$ in the reminder of this subsection. Depending on the active set, the DCS (6.8) reduces either to an ODE or to a DAE.

In the DAE case, $x(t)$ is on the boundary of one or more regions R_i, and we speak of sliding modes. In this case, $|\mathcal{I}| > 1$, and typically we obtain an index 2 differential algebraic equation. Consequently, two or more equal entries of $g(x)$ are the smallest components of this vector, and the solution θ of the LP(x) is not unique and lies on a facet of the unit simplex. To compute the values of θ, we must treat the DCS as a DAE. We define $F_{\mathcal{I}}(x, u) := F(x, u)P_{\mathcal{I}}^\top$, which selects the appropriate columns of $F(x, u)$. For $t \in \mathcal{T}$, we have $\theta_i = 0, i \notin \mathcal{I}$

and $\lambda_i = 0, i \in \mathcal{I}$, thus the DCS (6.8) reduces to the DAE

$$\dot{x} = F_{\mathcal{I}}(x,u)\theta_{\mathcal{I}}, \tag{6.15a}$$

$$0 = g_{\mathcal{I}}(x) - \mu e, \tag{6.15b}$$

$$1 = e^\top \theta_{\mathcal{I}}. \tag{6.15c}$$

There are $|\mathcal{I}|+1$ nontrivial algebraic equations and $|\mathcal{I}|+1$ unknown algebraic variables, namely μ and θ_i for $i \in \mathcal{I}$, since we regard $\theta_i(t) = 0,\ i \notin \mathcal{I}$ as fixed.

In the ODE case, $x(t)$ is in the interior of some region R_i we have $|\mathcal{I}| = 1$. The algebraic variables μ and θ_i can be computed explicitly from (6.15) and we have $\theta_i = 1$ and $\mu = g_i(x)$. Thus, the DCS reduces to the ODE $\dot{x} = f_i(x)$.

Next, we provide sufficient conditions for the uniqueness of the solution of the DAE (6.15) for a given $|\mathcal{I}| \geq 1$. We define the matrix

$$M_{\mathcal{I}}(x) = \nabla g_{\mathcal{I}}(x)^\top F_{\mathcal{I}}(x,u) \in \mathbb{R}^{|\mathcal{I}|\times|\mathcal{I}|}. \tag{6.16}$$

Note that entries of this matrix arise by taking the total time derivative of (6.15b).

Assumption 6.6. *Given a fixed active set $\mathcal{I}(x(t)) = \mathcal{I}$ for $t \in \mathcal{T}$, it holds that the matrix $M_{\mathcal{I}}(x(t))$ is invertible and $e^\top M_{\mathcal{I}}(x(t))^{-1} e \neq 0$ for all $t \in \mathcal{T}$.*

Proposition 6.7. *Suppose that Assumption 6.6 holds. Given the initial value $x(0)$, then the DAE* (6.15) *has a unique solution for all $t \in \mathcal{T}$.*

Proof. For a given $x(\cdot)$ we can differentiate equation (6.15b) w.r.t. t and obtain the following index 1 DAE

$$\dot{x} = F_{\mathcal{I}}(x,u)\theta_{\mathcal{I}}, \tag{6.17a}$$

$$\dot{\mu} = -v, \tag{6.17b}$$

$$\begin{bmatrix} M_{\mathcal{I}}(x) & e \\ e^\top & 0 \end{bmatrix} \begin{bmatrix} \theta_{\mathcal{I}} \\ v \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{6.17c}$$

with the algebraic variables $\theta_{\mathcal{I}}$ and $v \in \mathbb{R}$. For a given initial condition $x(0)$, $\mu(0)$ can be directly computed from any component of (6.15b). Using the Schur complement and Assumption 6.6, we conclude that we can find unique $\theta_{\mathcal{I}}$ and v by solving the linear system (6.17c). Therefore, the DAE (6.15) can be reduced to an ODE. Since the functions $f_i(\cdot)$ are assumed to be Lipschitz the resulting ODE has a unique solution $x(t), t \in \mathcal{T}$. □

Note that even though the DAE has a unique solution for a given active set $\mathcal{I}$, there might be multiple $\mathcal{I}$ that give a well-defined ODE, as we discuss in the subsequent sections.

6.2.3 Active-set changes and continuity of λ and μ

Every active-set change in (6.8d) corresponds to crossing a discontinuity, entering or leaving a sliding mode, or a spontaneous leaving of a surface of discontinuity. These events in time are called *switches*.

There are four qualitatively different switching cases possible. We illustrate them with a few simple examples. Other more complex behavior emerges from the combination of these simple cases.

Example 6.8. *There are four possible switching cases which we illustrate with the following examples:*

(a) crossing a surface of discontinuity, $\dot{x}(t) \in 2 - \operatorname{sign}(x(t))$,

(b) sliding mode, $\dot{x}(t) \in -\operatorname{sign}(x(t)) + 0.2\sin(5t)$,

(c) leaving sliding mode $\dot{x}(t) \in -\operatorname{sign}(x(t)) + t$.

(d) spontaneous switch, $\dot{x}(t) \in \operatorname{sign}(x(t))$,

In case (a), for $x(0) < 0$ *the trajectory reaches* $x = 0$ *and crosses it. In example (b), for any finite* $x(0)$*, the trajectory reaches* $x = 0$ *and stays there. On the other hand, in example (c), the DI has a unique solution, and for* $x(0) = 0$ *the trajectory leaves* $x = 0$ *at* $t = 1$*. In the last example, for* $x(0) = 0$ *the DI has infinitely many solutions, and* $x(t)$ *can spontaneously leave* $x = 0$ *at any* $t \geq 0$*. The trajectories are illustrated in Figure 6.2.*

The example in (d) is the only of the four that does not satisfy the one-sided Lipschitz condition in Assumption 4.9 and has a nonunique solution. Note that there is a qualitative difference between a spontaneous switch (d) and leaving a sliding mode (c). The spontaneous switch happens only when the solutions are nonunique. Leaving a sliding mode happens for systems with a unique solution. Notably, the corresponding θ_i is a locally continuous function of time in this case.

It is useful to take a closer look at how are the active sets $\mathcal{I}_n, \mathcal{I}^0_{n+1}$ and $\mathcal{I}_{n+1}$ related to different kinds of switches. For the readers' convince we write these sets explicitly for our example:

a) $\mathcal{I}_0 = \{1\}$, $\mathcal{I}^0_1 = \{1, 2\}$ and $\mathcal{I}_1 = \{2\}$,

b) $\mathcal{I}_0 = \{1\}$, $\mathcal{I}^0_1 = \{1, 2\}$ and $\mathcal{I}_1 = \{1, 2\}$,

c) $\mathcal{I}_0 = \{1, 2\}$, $\mathcal{I}^0_1 = \{1, 2\}$ and $\mathcal{I}_1 = \{2\}$,

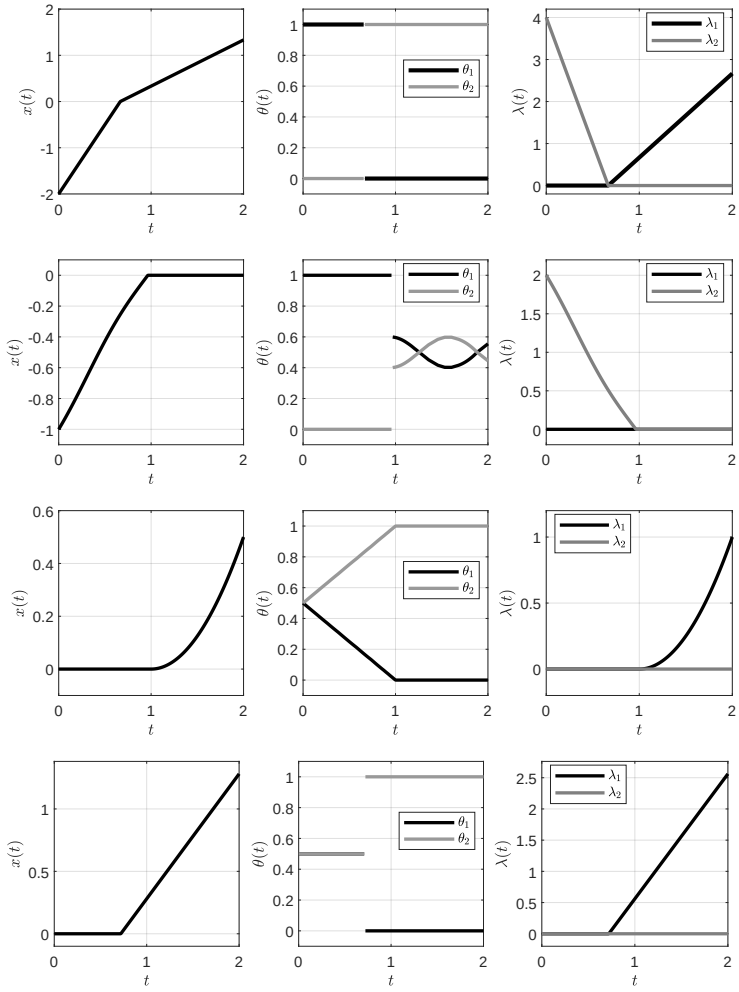


Figure 6.2: Illustration of example solution trajectories for different switching cases. The rows from top to bottom show $x(t)$, $\theta(t)$ and $\lambda(t)$ for the cases (a)-(d) in Example 6.8, respectively.

d) $\mathcal{I}_0 = \{1, 2\}$, $\mathcal{I}_1^0 = \{1, 2\}$ and $\mathcal{I}_1 = \{2\}$.

The relations are for the general case summarized in Table 6.2. Recall that in all cases $\mathcal{I}_n \cup \mathcal{I}_{n+1} = \mathcal{I}_{n+1}^0$.

We can see in Figure the functions $\theta(t)$ are discontinuous, except in the case of

Switch type	$\mathcal{I}_n$ vs. $\mathcal{I}^0_{n+1}$	$\mathcal{I}^0_{n+1}$ vs. $\mathcal{I}_{n+1}$
crossing a surface of discontinuity	$\mathcal{I}_n \subset \mathcal{I}^0_{n+1}$	$\mathcal{I}^0_{n+1} \supset \mathcal{I}_{n+1}$
entering a sliding mode	$\mathcal{I}_n \subset \mathcal{I}^0_{n+1}$	$\mathcal{I}^0_{n+1} = \mathcal{I}_{n+1}$
leaving sliding mode	$\mathcal{I}_n = \mathcal{I}^0_{n+1}$	$\mathcal{I}^0_{n+1} \supset \mathcal{I}_{n+1}$
spontaneous switch	$\mathcal{I}_n = \mathcal{I}^0_{n+1}$	$\mathcal{I}^0_{n+1} \supset \mathcal{I}_{n+1}$

Table 6.2: Relationship between $\mathcal{I}_n, \mathcal{I}^0_{n+1}$ and $\mathcal{I}_{n+1}$ for different switching cases.

continuously leaving a sliding mode in case (c). On the other hand, the functions $\lambda(t)$ are continuous in all cases. Let us formalize this observation. From (6.3), Eq. (6.8b) and the complementarity conditions (6.8d) for $i \in \mathcal{I}(x)$ it follows that $\theta_i \geq 0$ and $\lambda_i = 0$. Likewise, for $i \notin \mathcal{I}(x)$ it follows that $\theta_i = 0$ and $\lambda_i \geq 0$. Hence, for $i \in \mathcal{I}(x)$ from (6.8b) and (6.5) we conclude that $\mu = \min_{j \in \mathcal{J}} g_j(x)$.

Lemma 6.9. *The functions $\lambda(t)$ and $\mu(t)$ in* (6.8) *are continuous in time.*

Proof: The function $\mu(t)$ is a minimum of continuous functions and is thus continuous. Therefore, continuity of $\lambda(t) = g(x(t)) - \mu(t)e$ follows from the continuity of $x(t)$ and $g(x)$ and Equation (6.8b). □

Remark 6.10. *Continuity of $\lambda(t)$ implies that at an active-set change of a component i at $t_{s,n+1}$ some $\lambda_i(t_{s,n+1})$ must be zero. Moreover, for some $i \notin \mathcal{I}_n$, in the case of crossing a discontinuity or entering a sliding mode with $i \in \mathcal{I}_{n+1}$, it holds that $\dot{\lambda}_i(t^-_{s,n+1}) < 0$. Likewise, in the case of leaving a sliding mode or a spontaneous switch, with $i \in \mathcal{I}_n$ and $i \notin \mathcal{I}_{n+1}$, it follows that $\dot{\lambda}(t^+_{s,n+1}) > 0$. If some of the first-order one-sided derivatives of $\lambda(\cdot)$ are zero at a switching point $t_{s,n+1}$, then one must look at higher-order derivatives to determine if it stays active or not. See the right column of plots in Figure 6.2 for the claims in this remark.*

We exploit the continuity of $\lambda(\cdot)$ and $\mu(\cdot)$ later in the derivation of the FESD method in the next chapter.

6.2.4 Predicting the new active set

In this subsection, we restate a more technical result from [250], which is later needed in the convergence proof of the FESD method. The reader not interested in the proofs may skip this part.

As already noted in Remark 6.10, switches are characterized by the time derivative of $\lambda(\cdot)$. Using the analytic expression for the right time derivative

of $\lambda(\cdot)$ one can construct a Linear Complementarity Problem (LCP) with the data at $x(t_{\mathrm{s},n})$ and predict $\mathcal{I}_n$.

We define the vector $w_{\mathcal{I}}(t) \coloneqq \frac{\mathrm{d}}{\mathrm{d}t}\lambda_{\mathcal{I}}(t) = M_{\mathcal{I}}(x(t))\theta_{\mathcal{I}}(t) - \dot{\mu}(t)e$. Now for the active set $\mathcal{I}_n^0$ at a switching point $t_{\mathrm{s},n}$, one can construct the following mixed LCP between $\dot{\lambda}_{\mathcal{I}_n^0}$ and $\theta_{\mathcal{I}_n^0}$:

$$w_{\mathcal{I}_n^0} = M_{\mathcal{I}_n^0}(x)\theta_{\mathcal{I}_n^0} - \dot{\mu}e, \tag{6.18a}$$

$$1 = e^\top \theta_{\mathcal{I}_n^0}, \tag{6.18b}$$

$$0 \leq w_{\mathcal{I}_n^0} \perp \theta_{\mathcal{I}_n^0} \geq 0. \tag{6.18c}$$

Does this mixed LCP has a solution at all, or is the solution unique depends on the properties of the matrix $M_{\mathcal{I}_n^0}(x)$. Fortunately, the problem can be formulated into an equivalent one that is simpler to analyze. First, the problem matrix $M_{\mathcal{I}_n^0}(x)$ is altered. For a sufficiently large $\alpha > 0$ all entries of the matrix $M_{\mathcal{I},\alpha}(x) = M_{\mathcal{I}}(x) + \alpha ee^\top$ are strictly positive. This means the matrix $M_{\mathcal{I},\alpha}(x)$ is strictly copositive, i.e., for any $a \geq 0, a \neq 0$ it holds that $a^\top M_{\mathcal{I},\alpha}(x)a > 0$ [89]. Moreover, one can derive an LCP equivalent to (6.18) [250, Lemma 3.3]:

$$0 \leq \tilde{w}_{\mathcal{I}_n^0} = M_{\mathcal{I}_n^0,\alpha}(x)\tilde{\theta}_{\mathcal{I}_n^0} - e \perp \tilde{\theta}_{\mathcal{I}_n^0} \geq 0. \tag{6.19}$$

The motivation for rewriting (6.18) as (6.19) is twofold. It is both easier to prove solution existence and to compute a solution for an LCP with a strictly copositive matrix than for the initial mixed LCP [250]. The solution of the initial LCP $(w_{\mathcal{I}_n^0}, \theta_{\mathcal{I}_n^0})$ can be reconstructed via

$$\theta_{\mathcal{I}_n^0} = \frac{\tilde{\theta}_{\mathcal{I}_n^0}}{e^\top \tilde{\theta}_{\mathcal{I}_n^0}},\ w_{\mathcal{I}_n^0} = \frac{\tilde{w}_{\mathcal{I}_n^0}}{e^\top \tilde{\theta}_{\mathcal{I}_n^0}}.$$

For further details, cf. [250, Lemma 3.3]. Now, there is a one-to-one correspondence between the active set in a neighborhood of a switching point $t_{\mathrm{s},n}$ and the solutions of the LCP (6.19). This is summarized in the next theorem.

Theorem 6.11 (Theorem 3.2 [250])**.** *Let $x(t)$ be a solution in the sense of Definition 6.1 for $t \in [t_a, t_b]$, with $\mathcal{I}^0 = \mathcal{I}(x(t_a))$ and $\mathcal{I} = \mathcal{I}(x(t))$ for all $t \in (t_a, t_b)$. Suppose Assumption 6.6 holds for all $t \in (t_a, t_b)$. Then for each $t \in (t_a, t_b)$ there is a solution of the LCP* (6.19) *such that*

$$\{i \mid \tilde{\theta}_i > 0\} \subseteq \mathcal{I} \subseteq \{i \mid \tilde{w}_i = 0\}.$$

Conversely, let $x_0 \in \mathbb{R}^{n_x}$ and t_a be given with $\mathcal{I}^0 = \mathcal{I}(x_0)$. Then if $(\tilde{w}_{\mathcal{I}^0}, \tilde{\theta}_{\mathcal{I}^0})$ is a solution of the LCP (6.19) *such that*

$$\{i \mid \tilde{\theta}_i > 0\} = \mathcal{I} = \{i \mid \tilde{w}_i = 0\}$$

and the conditions of Assumptions 6.6 are satisfied for $\nabla g_i(x)$ *and* $f_i(x,u)$, $i \in \mathcal{I}$, *then there is a* $t_b > t_a$ *and a solution* $x(\cdot)$ *in the sense of Definition 6.1 on* $[t_a, t_b]$ *such that* $x(t_a) = x_0$ *and* $\mathcal{I}(x(t)) = \mathcal{I}$ *for all* $t \in (t_a, t_b)$.

Regard an LCP

$$0 \leq M\theta + q \perp \theta \geq 0, \tag{6.20}$$

with $M \in \mathbb{R}^{l\times l}$ and $q \in \mathbb{R}^l$. The given LCP (6.20) is compactly denoted by $\mathrm{LCP}(M,q)$ and its set of solutions is denoted by $\mathrm{SOL}(M,q) \subseteq \mathbb{R}^l$. If a solution satisfies $(M\theta + q) + \theta > 0$, we say that strict complementarity holds.

In the convergence analysis of the FESD method introduced in the next chapter, we rely on the regularity properties of the LCP (6.19). In particular, to prove convergence we require the solutions of the LCP to be *strongly stable* [89, 250]. A solution $(w^*, \theta^*) \in \mathrm{SOL}(M,q)$ of a given $\mathrm{LCP}(M,q)$ is said to be strongly stable if there is a neighborhood U of θ^* and a neighborhood V of the problem data $M \in \mathbb{R}^{l\times l}$ and $q \in \mathbb{R}^l$, such that the intersection of U with the solution set of an LCP constructed from the data from one point in V is a singleton [69]. We state a regularity assumption about the LCP (6.19).

Assumption 6.12. *Consider a solution* $x(t)$ *in the sense of Definition 6.1 for* $t \in [0,T]$, *and let* $\mathcal{S} = \{t_{\mathrm{s},0}, \ldots, t_{\mathrm{s},N_{\mathrm{sw}}}\}$ *be the set of switching points. The solutions of the LCP* (6.19) *are strongly stable and satisfy strict complementarity for all* $t \in [t_\mathrm{s} - \epsilon, t_\mathrm{s} + \epsilon] \cap [0,T]$, $t_\mathrm{s} \in \mathcal{S}$, *for a sufficiently large* $\epsilon > 0$.

The strict complementarity assumption is needed to obtain a tight prediction of the next active set $\mathcal{I}$, cf. first part of Theorem 6.11. From the proof of [250, Theorem 3.2] it follows that the strict complementarity condition implies that the one-sided time derivatives of $\lambda_i(t), i \notin \mathcal{I}(x(t))$ are nonzero, see also Remark 6.10. Without this assumption, one can obtain only an over-approximation of $\mathcal{I}$. However, it can be relaxed at the cost of looking at higher-order time derivatives of $\lambda_i(t_{\mathrm{s},n}), i \in \mathcal{I}_n^0$ and constructing an appropriate LCP for determining the active sets past some switching point, cf. [251, Section 4.2] for derivations. This happens for instance in Example 6.8 (d), where both $\theta_1(t_1^+) = 0$ and $\dot{\lambda}_1(t_1^+) = 0$ at the switching point $t_1 = 1$. Clearly, $\ddot{\lambda}_1(t_1^+) > 0$ and we can deduce that $\mathcal{I}_1 = \{1\}$. Strong stability is assumed in order to apply some results on parametric LCPs. In our case, we will use it to draw the same conclusions from an LCPs constructed at t and t', where t and t' are sufficiently close.

Example 6.13. *We illustrate Theorem 6.11 on the example* $\dot{x} = 2 - \mathrm{sign}(x)$ *with* $x(0) = -1$, *cf. first row in Figure 6.2. It is easy to see that* $t_{\mathrm{s},1} = -\frac{1}{3}$ *and that the relevant active sets are* $\mathcal{I}_0 = \{1\}$, $\mathcal{I}_1 = \{2\}$ *and* $\mathcal{I}_1^0(t_{\mathrm{s},1}) = \{1,2\}$. *The*

LCP (6.19) *for our example at* $t_{s,1}$ *reads as*

$$0 \leq \tilde{w}_{\mathcal{I}_1^0} = \begin{bmatrix} 3+\alpha & 1+\alpha \\ -3+\alpha & -1+\alpha \end{bmatrix} \tilde{\theta}_{\mathcal{I}_1^0} - e \perp \tilde{\theta}_{\mathcal{I}_1^0} \geq 0.$$

With $\alpha = 5$ *this LCP has the unique solution* $\tilde{\theta}_{\mathcal{I}_1^0} = (0, \frac{1}{4})$ *and* $\tilde{w}_{\mathcal{I}_1^0} = (\frac{1}{2}, 0)$ *and according to the last theorem it correctly predicts* $\mathcal{I}_1 = \{2\}$.

Some lemmata about parametric LCPs

In this section, we restate three lemmas about LCPs from [250]. They will be useful in the theoretical analysis of the FESD method. Again, readers not interested in the convergence theory can skip this section.

The first lemma is about the strong stability of perturbed LCP.

Lemma 6.14 ([250][Lemma A.1])**.** *Suppose that all entries of* M *are positive in* (6.20) *and all solutions of* $\mathrm{LCP}(M, q)$ *are strongly stable. Then any LCP with the data* $\hat{M}$ *and* $\hat{q}$*, such that* $\|M - \hat{M}\|$ *and* $\|q - \hat{q}\|$ *are sufficiently small, has the same number of solutions as* $\mathrm{LCP}(M, q)$*. Moreover, the number of solutions is finite.*

The second lemma is about the active sets of perturbed LCP, where strict complementarity holds at a solution.

Lemma 6.15 ([250][Lemma A.2])**.** *Suppose that all entries of* M *are positive in* (6.20) *and all solutions of* $\mathrm{LCP}(M, q)$ *are strongly stable. If* $\hat{M}_n \to M$*,* $\hat{q}_n \to q$*, then* $\mathrm{SOL}(\hat{M}_n, \hat{q}_n) \to \mathrm{SOL}(M, q)$*, as* $n \to \infty$*, in the Hausdorff metric. Moreover, if* $(w, \theta) \in \mathrm{SOL}(M, q)$*, such that* $w + \theta > 0$*, then there is a* $(\hat{w}_n, \hat{\theta}_n) \in \mathrm{SOL}(\hat{M}_n, \hat{q}_n)$ *for sufficiently large* n *such that* $\{i \mid \hat{\theta}_{n,i} > 0\} = \{i \mid \theta_i > 0\}$.

The third lemma is about a parametric LCP and its number of solutions.

Lemma 6.16 ([250][Lemma A.3])**.** *Let* $M(t)$ *and* $q(t)$ *be continuous functions where all entries of* $M(t)$ *are positive and all solutions of* $\mathrm{LCP}(M(t), q(t))$ *are strongly stable for all* $t \in [t', t'']$*. Then* $|\mathrm{SOL}(M(t), q(t))|$ *is constant. Moreover, if* $\mathrm{LCP}(M(t), q(t))$ *has a unique solution denoted by* $(w^*(t), \theta^*(t))$ *such that* $w^*(t) + \theta^*(t) > 0$ *for all* $t \in [t', t'']$*, then* $\mathcal{I}(t) = \{i \mid \theta_i^*(t) > 0\}$ *is a constant set.*

6.2.5 Sum of Filippov systems

The reformulation from the last subsection given by the DCS (6.8) fails on some simple examples such as: $\dot{x}_1 \in -\mathrm{sign}(x_1)$, $\dot{x}_2 \in -\mathrm{sign}(x_2)$, $x \in \mathbb{R}^2$. This

example satisfies the one-sided Lipschitz condition and has a unique Filippov solution, cf. Theorem 4.10. However, as shown in [252] at $(0,0)$, the DAE arising from (6.8) fails to have a unique solution. One can see that $\dot{x}_1 \in -\text{sign}(x_1)$ and $\dot{x}_2 \in -\text{sign}(x_2)$ are completely independent, and thus they should be treated in such a way.

Stewart introduced a generalization of his reformulation for such cases in [252]. One should identify the n_{sys} independent subsystems with index $k = 1, \ldots, n_{\text{sys}}$, where each subsystem has n_f^k modes. We equip all variables related to the $k-$th subsystem with the superscript k. Instead of (6.3) one can write

$$\dot{x} \in \Big\{ \sum_{k=1}^{n_{\text{sys}}} \sum_{i=1}^{n_f^k} \theta_i^k f_i^k(x,u) \;\Big|\; \sum_{i=1}^{n_f^k} \theta_i^k = 1,\ \theta^k \geq 0, k = 1, \ldots, n_{\text{sys}} \Big\}. \tag{6.21}$$

Finding the functions $g^k(\cdot) \in \mathbb{R}^{n_f^k}$ from $c^k(\cdot) \in \mathbb{R}^{n_f^k}$ for every subsystem works the same way as in Section 6.2.1. Thereby, the regions of every subsystem are defined via the matrix S^k and the switching functions $c^k(x) \in \mathbb{R}^{n_c^k}$. Every mode's convex combination is encoded by its own parametric linear program (6.7), constructed with the k-th modes' switching functions $g^k(x) \in \mathbb{R}^{n_f^k}$. Thus, we can derive the DCS

$$\dot{x} = \sum_{k=1}^{n_{\text{sys}}} F^k(x,u)\theta^k, \tag{6.22a}$$

$$0 = g^k(x) - \lambda^k - \mu^k e, \qquad \text{for all } k \in \{1, \ldots n_{\text{sys}}\}, \tag{6.22b}$$

$$1 = e^\top \theta^k, \qquad \text{for all } k \in \{1, \ldots n_{\text{sys}}\}, \tag{6.22c}$$

$$0 \leq \theta^k \perp \lambda^k \geq 0, \qquad \text{for all } k \in \{1, \ldots n_{\text{sys}}\}, \tag{6.22d}$$

where $F^k(x,u) = [f_1^k(x,u), \ldots, f_{n_f^k}^k(x,u)] \in \mathbb{R}^{n_x \times n_f^k}$ and $g^k(x) \in \mathbb{R}^{n_f^k}$, $\theta^k \in \mathbb{R}^{n_f^k}$, $\lambda^k \in \mathbb{R}^{n_f^k}$ and $\mu^k \in \mathbb{R}$, for all $k \in \{1, \ldots, n_{\text{sys}}\}$. For ease of notation, in the remainder of the thesis, we treat the case with $n_{\text{sys}} = 1$, as all extensions are straightforward.

To the best of the author's knowledge, there are no general conditions known which identify when the r.h.s. of (4.20) is partially separable as in (6.21) and there might even be multiple ways to write a system in this form. In practice, it is usually easy to identify the structure of (6.21) by inspection. For example, this occurs if we have multiple surfaces with friction, or multiple objects touching the same frictional surface [252].

6.2.6 Sensitivities with respect to parameters and initial values

Correct calculation of derivatives of solutions w.r.t. parameters (e.g., discretized control functions) and initial values is crucial for efficient numerical optimal control algorithms and verifying the optimality of a solution. For smooth ODEs, this is discussed in Section 3.2.2. However, this is not straightforward for ODE with a discontinuous r.h.s., as the sensitivity usually exhibits jumps when switches happen, cf. Sections 4.2.6 and 5.2. As any constant parameter $\hat{p}$ can be modeled via adding the state $\dot{p} = 0$ and $p(0) = \hat{p}$, we restrict our attention to sensitivities w.r.t. initial values.

Regard the DCS given by Eq. (6.8) on a time interval $[0, T]$ with the initial condition $x(0) = x_0$. Assume that the surface ∂R_j is reached at $t_\mathrm{s}(x_0) \in (0, T)$ and that $x_0 \in R_i$. We regard the case where the solution crosses a co-dimension one surface of discontinuity ∂R_j. Other cases are when the trajectory: (a) slides on the surface of discontinuity after reaching it, (b) starts on a surface of discontinuity and stays on it or leaves it, or (c) goes from one surface to another. They can be analyzed with the same arguments as below, but we omit these cases here for brevity, cf. [94, Section 2.11].

In the case of crossing, we have for $t \in [0, t_\mathrm{s})$ that $\mathcal{I}(x(t)) = \{i\}$ and from (6.8) it follows that $\dot{x} = f_i(x)$. After crossing ∂R_j at t_s we have $\mathcal{I}(x(t)) = \{j\}$ for $t \in (t_\mathrm{s}, T]$ and $\dot{x} = f_j(x)$. At t_s it holds that

$$\psi_{i,j}(x(t)) \coloneqq g_i(x(t)) - g_j(x(t)) = 0. \tag{6.23}$$

Thus, the system can be compactly represented by

$$\dot{x}(t) = \begin{cases} f_i(x(t)), & \psi_{i,j}(x(t)) < 0, \\ f_j(x(t)), & \psi_{i,j}(x(t)) \geq 0. \end{cases} \tag{6.24}$$

We are interested in the exact sensitivity matrix $S^x(t, 0; x_0)$ of a solution $x(t; x_0)$ of the system (6.24), i.e.,

$$S^x(t, 0; x_0) = \frac{\partial x(t; x_0)}{\partial x_0} \in \mathbb{R}^{n_x \times n_x}$$

The function $S^x(t, 0; x_0)$ obeys smooth linear variational differential equations, as in (3.14), on both sides of t_s, but exhibits a jump at t_s [94]. This is summarized in the following proposition. The proof is standard and can be found, for example, in [259].

Proposition 6.17. *Regard the system* (6.24) *with* $x(0) = x_0 \in R_i$ *on an interval* $[0, T]$ *with a switch at* $t_\mathrm{s} \in (0, T)$. *Assume that the functions* $f_i(x)$, $f_j(x), \psi_{i,j}(x)$ *are continuously differentiable along* $x(t), t \in [0, T]$. *Assume the solution* $x(t)$

reaches the surface of discontinuity transversally, i.e., $\nabla\psi_{i,j}(x(t_\mathrm{s}))^\top f_i(x(t_\mathrm{s})) > 0$. *Then the sensitivity* $S^x(T,0;x_0)$ *of a solution* $x(t;x_0)$ *of the system described by the ODE* (6.24) *is given by*

$$\begin{aligned} &S^x(T,0;x_0) = S^x(T,t_\mathrm{s}^+;x(t_\mathrm{s}))J(x(t_\mathrm{s};x_0))S^x(t_\mathrm{s}^-,0;x_0) \text{ with} \\ &J(x(t_\mathrm{s};x_0)) \coloneqq I + \frac{(f_j(x(t_\mathrm{s};x_0)) - f_i(x(t_\mathrm{s};x_0)))\nabla\psi_{i,j}(x(t_\mathrm{s};x_0))^\top}{\nabla\psi_{i,j}(x(t_\mathrm{s};x_0))^\top f_i(x(t_\mathrm{s};x_0))}. \end{aligned} \tag{6.25}$$

Proof. For $t < t_\mathrm{s}$, the solution $x(t;x_0)$ satisfies the ODE $\dot{x} = f_i(x(t;x_0))$ and the sensitivity matrix $S^x(t,0;x_0) = \frac{\partial x(t;x_0)}{\partial x_0}$ obeys (cf. Eq. (3.14))

$$\dot{S^x}(t) = \frac{\partial f(x)}{\partial x} S^x(t),\ S^x(0) = I.$$

At $t = t_\mathrm{s}$ the solution reaches the surface of discontinuity:

$$\psi_{i,j}(x(t_\mathrm{s}(x_0;x_0))) = 0. \tag{6.26}$$

For $t > t_\mathrm{s}$, one has $y(t) = f_*(y(t;y_0))$ which is related to the solution via

$$y(t;y_0) = x(t + t_\mathrm{s}(x_0);x_0),\ y_0(x_0) = x(t_\mathrm{s}(x_0);x_0). \tag{6.27}$$

Note that $y(t - t_\mathrm{s}(x_0);x_0) = x(t;x_0)$. Therefore, the sensitivity, for $t > t_\mathrm{s}$ can be computed via

$$\begin{aligned} S^x(t,0;x_0) = \frac{\partial x(t,x_0)}{\partial x_0} &= \frac{\partial y(t - t_\mathrm{s}(x_0)); y_0(x_0)}{\partial x_0} \\ &= \frac{\partial y(t - t_\mathrm{s}}{\partial t}\frac{\partial t_\mathrm{s}(x_0)}{\partial x_0}^\top + S^y(t - t_\mathrm{s};y_0)\frac{\partial y_0(x_0)}{\partial x_0}, \\ &= -f_*(x(t))\frac{\partial t_\mathrm{s}(x_0)}{\partial x_0}^\top + S^y(t - t_\mathrm{s};y_0)\frac{\partial y_0(x_0)}{\partial x_0}, \end{aligned} \tag{6.28}$$

We can compute $\frac{\partial y_0(x_0)}{\partial x_0}$ at $t = t_\mathrm{s}^-$ using (6.27)

$$\frac{\partial y_0(x_0)}{\partial x_0} = \frac{\partial x(t_\mathrm{s}(x_0);x_0)}{\partial x_0} = f_i(x)\frac{\partial t_\mathrm{s}(x_0)}{\partial x_0}^\top + S^x(t_\mathrm{s}^-,0;x_0). \tag{6.29}$$

Using the implicit function theorem (cf. [85, Theorem 1B.1]) for (6.26), again at $t = t_\mathrm{s}^-$ we obtain

$$\frac{\partial t_\mathrm{s}(x_0)}{\partial x_0}^\top = -\frac{\nabla\psi_{i,j}(x(t_\mathrm{s}(x_0;x_0)))^\top S^x(t_\mathrm{s}^-,0;x_0)}{\nabla\psi_{i,j}(x(t_\mathrm{s}(x_0;x_0)))^\top f_i(x)}. \tag{6.30}$$

Let $t \to t_{\mathrm{s}}^+$ in (6.28), then $S^y(t - t_{\mathrm{s}}; y_0) \to I$. Plugging (6.29) and (6.30) into (6.28) for the remaining unknown terms we obtain

$$\begin{aligned} S^x(t_{\mathrm{s}}^+, 0; x_0) &= f_*(x(t)) \frac{\nabla \psi_{i,j}(x(t_{\mathrm{s}}(x_0; x_0)))^\top S^x(t_{\mathrm{s}}^-, 0; x_0)}{\nabla \psi_{i,j}(x(t_{\mathrm{s}}(x_0; x_0)))^\top f_i(x)} \\ &- f_i(x) \frac{\nabla \psi_{i,j}(x(t_{\mathrm{s}}(x_0; x_0)))^\top S^x(t_{\mathrm{s}}^-, 0; x_0)}{\nabla \psi_{i,j}(x(t_{\mathrm{s}}(x_0; x_0)))^\top f_i(x)} S^x(t_{\mathrm{s}}^-, 0; x_0) + S^x(t_{\mathrm{s}}^-, 0; x_0) \end{aligned} \tag{6.31}$$

Finally, from the chain rule we have $S^x(T, 0, x_0) = S^x(T, t_{\mathrm{s}}^+, x_0) S^x(t_{\mathrm{s}}^+, 0, x_0)$ and (6.31) we obtain (6.25). □

This proposition can also be adapted to the case of sliding modes. We obtain similar expressions for the sensitivity jump formula as in (6.25). The only change needed to be made is to replace $f_j(x)$ with $f^*(x)$, where $f^*(x)$ defines the sliding vector field [93].

We have seen in Section 5.2 that numerical sensitivities obtained via standard methods fail to converge to their correct values (6.25). This may impair the progress of the optimizer. Hence, special care is needed when discretizing Filippov systems to solve optimal control problems.

6.3 Heaviside step reformulation

In this section, we present an alternative to Stewart's approach to passing from the Filippov DI (6.3) to a dynamic complementarity system. At the core of the transformation is a linear program whose solution map corresponds to the set-valued Heaviside step function, which we for brevity, often call step function. It is no surprise that also this DCS has similar properties to the one derived in the previous section. However, depending on the shape and description of the regions R_i, one formulation might be advantageous over the other.

Smoothed single-valued versions of the step function are commonly used in the numerical treatment of piecewise smooth systems [34, 119, 190]. Here, the key idea is to use the nonsmooth set-valued step function to determine in which region or on what boundaries the trajectory is. The arguments of the step functions are the usual switching functions $\psi(x)$ and the Filippov multipliers θ are obtained as products of all relevant step functions and their complements. Moreover, the set-valued step functions allow us to formulate even more general DIs than Filippov DIs, e.g., Aizerman–Pyatnitskii systems [Chapter 2][94]. They arise from the modeling of logical conjunctions within a dynamical system, where the Boolean variables are replaced by step functions or a smooth approximation of it. A popular example of the use of step functions in PSS is gene-regulatory

networks [6, 190]. Another common use of step functions is to express Filippov sets in sliding modes of co-dimension higher than one [73, 74]. In this case, the step function reformulation acts as a nonlinear change of coordinates and one obtains a system that is often easier to study than the initial Filippov DI. For example, this approach yields provably unique sliding vector fields in many cases for sliding modes of co-dimension 2 [73, 74]. For higher co-dimensions, at least the existence of the sliding modes can be guaranteed [74].

6.3.1 Set-valued Heaviside step functions

The set-valued version of the Heaviside step function, denoted as $\gamma : \mathbb{R} \to \mathcal{P}(\mathbb{R})$, is defined as follows:

$$\gamma(y) = \begin{cases} \{1\}, & y > 0, \\ [0,1], & y = 0, \\ \{0\}, & y < 0. \end{cases} \tag{6.32}$$

Here $\mathcal{P}(\mathbb{R})$ represents the power set of $\mathbb{R}$. The set-valued Heaviside step function, often referred to simply as the step function, provides an intuitive way to model Boolean relationships within a dynamical system. In the modeling process, *switching functions* $\psi_i : \mathbb{R}^{n_x} \to \mathbb{R}$, for $i = 1, \ldots, n_\psi$, are commonly used as arguments of the step functions. We denote the concatenation of all scalar Heaviside step functions as $\Gamma(\psi(x)) \coloneqq (\gamma(\psi_1(x)), \ldots, \gamma(\psi_{n_\psi}(x)))$, where $\psi(x) = (\psi_1(x), \ldots, \psi_{n_\psi}(x))$. Let us denote by $\alpha \in \mathbb{R}^{n_c}$ a selection $\alpha \in \Gamma(y)$.

Let $\psi(x)$ be a continuous function of x, where $x(t)$ is some continuous function of time. A well-known way to express the function $\Gamma(\psi(x))$ [6, 28] is the use of the solution map of the parametric linear program:

$$\Gamma(\psi(x)) = \arg \min_{\alpha \in \mathbb{R}^{n_\psi}} \quad -\psi(x)^\top \alpha \tag{6.33a}$$

$$\text{s.t.} \quad 0 \leq \alpha_i \leq 1, \ i = 1, \ldots, n_\psi. \tag{6.33b}$$

Note that all components of α are decoupled in this LP, i.e., every α_i can be obtained by solving a one-dimensional LP with the objective $-\psi_i(x)\alpha_i$ and the feasible set $0 \leq \alpha_i \leq 1$. Let $\lambda^{\mathrm{n}}, \lambda^{\mathrm{p}} \in \mathbb{R}^{n_\psi}$ be the Lagrange multipliers for the lower and upper bound on α in (6.33b), respectively. The KKT conditions of (6.33) read as

$$\psi(x) = \lambda^{\mathrm{p}} - \lambda^{\mathrm{n}}, \tag{6.34a}$$

$$0 \leq \lambda^{\mathrm{n}} \perp \alpha \geq 0, \tag{6.34b}$$

$$0 \leq \lambda^{\mathrm{p}} \perp e - \alpha \geq 0, \tag{6.34c}$$

Let us look at a single component α_j and the associated functions $\psi_j(x)$. From the LP (6.33) and its KKT conditions, one can see that for $\psi_j(x) > 0$, we have $\alpha_j = 1$. Since the upper bound is active, we have that $\lambda_j^{\mathrm{n}} = 0$ and from (6.34a) that $\lambda_{\mathrm{p},j} = \psi_j(x) > 0$. Likewise, for $\psi_j(x) < 0$, we obtain $\alpha_j = 0$, $\lambda_j^{\mathrm{p}} = 0$ and $\lambda_j^{\mathrm{n}} = -\psi_j(x) > 0$. On the other hand, $\psi_j(x) = 0$ implies that $\alpha_j \in [0, 1]$ and $\lambda_j^{\mathrm{p}} = \lambda_j^{\mathrm{n}} = 0$. From this discussion, it can be seen that $\psi(x)$, λ^{n} and λ^{p} are related by the following expressions:

$$\lambda^{\mathrm{p}} = \max(\psi(x), 0), \ \lambda^{\mathrm{n}} = -\min(\psi(x), 0). \tag{6.35}$$

That is, λ^{p} collects the positive parts of $\psi(x)$ and λ^{n} the absolute value of the negative parts of $\psi(x)$. From this relation, we can immediately conclude the following:

Lemma 6.18. *Let $\psi(x(t))$ be a continuous function of time, then the functions $\lambda^{\mathrm{p}}(t)$ and $\lambda^{\mathrm{n}}(t)$ are continuous in time.*

6.3.2 Aizerman–Pyatnitskii differential inclusions

So far we have introduced dynamic complementarity and Filippov systems. Now we introduce another special case of the discontinuous ODE (4.14) and relate it to the others later with the help of Heaviside step functions. Regard and ODE with DRHS of the form of:

$$\dot{x} = f(x, u, v(x)), \tag{6.36}$$

where $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \in \mathbb{R}^{n_v}$ is continuous in all arguments, but $v(x)$ is discontinuous, e.g. it could be the usual single-valued Heaviside step function. Such systems appear commonly in sliding mode control or the modeling of gene-regulatory networks [6]. It is assumed that at each point of discontinuity, for the components $v_i(x)$ a closed convex set $V_i(x)$ is given, out of which the corresponding arguments of (6.36) take their values. Hence, one can define the following differential inclusion:

$$\dot{x} \in \mathcal{F}_{\mathrm{AP}}(x, u) := \Big\{ f(x, u, v) \mid v_i \in V_i(x), i = 1, \ldots, n_v \Big\}. \tag{6.37}$$

The set $\mathcal{F}_{\mathrm{AP}}(x, u)$ is in general nonconvex, except in some special cases. For example, $\mathcal{F}_{\mathrm{AP}}(x, u)$ is convex if $v(x)$ enters the r.h.s. of (6.36) linearly and $V(x)$ is closed convex. The DI (6.37) does not have as rich a theory as Filippov DIs, and there are fewer results available on the existence of solutions and convergence of numerical methods [6]. These systems are called Aizerman–Pyatnitskii DIs,

cf. [6] and [94, page 55, Definition c]. Time-stepping methods for such systems were developed in [6].

In this thesis, we focus on the case where the functions $V_i(x)$ are given by set-valued Heaviside step functions. In particular, we regard the DI where $V(x) = \Gamma(\psi(x))$, which can be written as:

$$\dot{x} \in \mathcal{F}(x, u, \Gamma(\psi(x))). \tag{6.38}$$

6.3.3 Filippov set expressed via Heaviside step functions

Next, we derive the expression for the Filippov set via set-valued step functions and the associated DCS. For the readers' convince we restate the Filippov DI (6.3) for the PSS (6.1):

$$\dot{x} \in \mathcal{F}_{\mathrm{F}}(x, u) = \Big\{ \sum_{i \in \mathcal{J}} f_i(x, u)\, \theta_i \mid \sum_{i \in \mathcal{J}} \theta_i = 1,\ \theta_i \geq 0,\ \theta_i = 0 \text{ if } x \notin \overline{R}_i, \forall i \in \mathcal{J} \Big\}. \tag{6.39}$$

The next question we address is: How can we find an (implicit) function of x for computing the Filippov multipliers θ in (6.39)? To derive such functions, we will make use of the set-valued step functions and the definition of the regions R_i, which are expressed via the switching functions $\psi_j(x)$ as in Definition 6.3. We assume that the regions of the PSS are equal to the basis regions R'_i defined in Definition 6.3, i.e., $R_i = R'_i$ for $i = 1, \ldots, 2^{n_\psi}$. Moreover, we assume the functions $\psi_j(\cdot)$ to be Lipschitz continuous, sufficiently differentiable and that $\nabla\psi(x) = \frac{\partial \psi(x)}{\partial x}^\top \in \mathbb{R}^{n_x \times n_\psi}$ has rank n_ψ.

For the sake of clarity, we start by illustrating what we want to achieve on a simple example and give in the sequel the general expression.

Example 6.19 (Step representation)**.** *We regard a PSS with four regions defined via two scalar switching functions $\psi_1(x)$ and $\psi_2(x)$. The regions are equal to the base sets from Definition 6.3 and read as $R_1 = \{x \in \mathbb{R}^{n_x} \mid \psi_1(x) > 0, \psi_2(x) > 0\}$, $R_2 = \{x \in \mathbb{R}^{n_x} \mid \psi_1(x) > 0, \psi_2(x) < 0\}$, $R_3 = \{x \in \mathbb{R}^{n_x} \mid \psi_1(x) < 0, \psi_2(x) > 0\}$ and $R_4 = \{x \in \mathbb{R}^{n_x} \mid \psi_1(x) < 0, \psi_2(x) < 0\}$. The corresponding sing matrix $S \in \mathbb{R}^{4\times 2}$ read as*

$$S = \begin{bmatrix} 1 & 1 \\ 1 & -1 \\ -1 & 1 \\ -1 & -1 \end{bmatrix}.$$

The corresponding Filippov system (defined by (6.3) *reads as:*

$$\dot{x} \in \{\sum_{i=1}^{4} \theta_i f_i(x) \mid \theta \geq 0, \sum_{i=1}^{4} \theta_i = 1,\ \theta_i = 0, \text{if } x \notin \overline{R_i}\}. \tag{6.40}$$

Let $x \in R_1$*, then* $\alpha_1 \in \gamma(\psi_1(x)) = \{1\}$ *and* $\alpha_2 \in \gamma(\psi_2(x)) = \{1\}$*, thus* $\theta_1 = \alpha_1\alpha_2 = 1$*. On the other hand, by direct evaluation of the step functions, one can see that* $\theta_1 = \alpha_1\alpha_2 = 0$ *if* $x \notin \overline{R_1}$*, since at least one of the selections* α_1 *or* α_2 *is zero. Similarly, for* $x \in R_2$*, we observe that* $\alpha_1 \in \gamma(\psi_1(x)) = \{1\}$ *and* $\alpha_2 \in \gamma(\psi_2(x)) = \{0\}$ *and we can set* $\theta_2 = \alpha_1(1-\alpha_2) = 1$*. By direct evaluation of we can see that* $\theta_2 > 0$ *if* $x \in \overline{R_2}$ *and* $\theta_2 = 0$*, otherwise. Following the same pattern, we conclude that* $\theta_3 = (1-\alpha_1)\alpha_2$ *and* $\theta_4 = (1-\alpha_1)(1-\alpha_2)$*. Thus, we can define the system*

$$\dot{x} \in \Big\{\alpha_1\alpha_2 f_1(x) + \alpha_1(1-\alpha_2) f_2(x) + (1-\alpha_1)\alpha_2 f_3(x) + (1-\alpha_1)(1-\alpha_2) f_4(x) \\ \mid \alpha_1 \in \gamma(\psi_1(x)), \alpha_2 \in \gamma(\psi_2(x))\Big\}. \tag{6.41}$$

Since $\alpha_1, \alpha_2 \in [0,1]$ *it is clear that* $\theta_i \in [0,1], i \in \{1,\dots,4\}$*. Moreover, direct calculation shows that* $\sum_{i=1}^{4} \theta_i = 1$*. Therefore, we conclude that the sets in the r.h.s. of* (6.40) *and* (6.41) *are the same sets, i.e., we can express the Filippov set via set-valued Heaviside step functions.*

Furthermore, we can observe how the sign pattern in S *determines how* α_j *enters the expression for* θ_i*. For* $S_{i,j} = 1$ *we have* α_j*, for* $S_{i,j} = -1$ *we have* $(1-\alpha_j)$*. In summary, the definition of* θ_i *consists of products of of* α_j *and* $(1-\alpha_k)$*, i.e., it is multi-affine in the selections* α_j*,* $j = 1,\dots,n_\psi$*.*

We generalize the patterns observed in the previous example and define the set

$$\mathcal{F}_{\mathrm{H}}(x,u) \coloneqq \Big\{\sum_{i=1}^{n_f} \prod_{j=1}^{n_\psi} \Big(\frac{1-S_{i,j}}{2} + S_{i,j}\alpha_i\Big) f_i(x,u) \mid \alpha \in \Gamma(\psi(x))\Big\}. \tag{6.42}$$

Note that we have

$$\frac{1-S_{i,j}}{2} + S_{i,j}\alpha_i = \begin{cases} \alpha_i, & \text{if } S_{i,j} = 1, \\ 1-\alpha_i, & \text{if } S_{i,j} = -1. \end{cases}$$

Similar definitions of $\mathcal{F}_{\mathrm{H}}(x,u)$ as in (6.42) can be found in [74, Section 4.2] and [119, Section 2.1]. Observe that this set has the same form as the r.h.s. $\mathcal{F}_{\mathrm{AP}}(x,u)$ in (6.37). Next, we show that $\mathcal{F}_{\mathrm{H}}(x,u)$ is indeed the same set as $\mathcal{F}_{\mathrm{F}}(x,u)$, i.e., the set in the r.h.s. of (6.3).

Lemma 6.20 (Lemma 1.5 in [74])**.** *Let $a_1, a_2, \ldots, a_m \in \mathbb{R}$. Consider the 2^m non-repeated products of the form $p_i = (1 \pm a_1)(1 \pm a_2)\cdots(1 \pm a_m)$, then it holds that $\sum_{i=1}^{2^m} p_i = 2^m$.*

Proposition 6.21. *Let*

$$\theta_i = \prod_{j=1}^{n_\psi} \Big(\frac{1 - S_{i,j}}{2} + S_{i,j}\alpha_j \Big), \text{ for all } i \in \mathcal{J} = \{1, \ldots, n_f\}, \tag{6.43}$$

then it holds that $\mathcal{F}_{\mathrm{F}}(x, u) = \mathcal{F}_{\mathrm{H}}(x, u)$.

Proof. We only need to show that $\theta_i \geq 0$ for all $i \in \mathcal{J}$ and $\sum_{i\in\mathcal{J}} \theta_i = 1$. It is easy to see that $\theta_i \in [0, 1]$ since it consists of a product of terms that takes value in $[0, 1]$.

Without loss of generality, regard θ_1 and suppose that $x \notin \overline{R_1}$. This means that $x \in R_i, i \neq 1$ and that at least one $\psi_j(x) < 0, j \in \mathcal{C}$, which implies that $\alpha_j = 0$. From (6.43) it follows that $\theta_1 = 0$ if $x \notin \overline{R_1}$. By similar arguments it follows that $\theta_i = 0$ if $x \notin \overline{R_i}$ for $i = 1, \ldots, n_f$.

Next we show that $\sum_{i\in\mathcal{J}} \theta_i = 1$. We introduce the change of variables:

$$\frac{1 + b_j}{2} = \alpha_j, \ \frac{1 - b_j}{2} = 1 - \alpha_j.$$

Then all θ_i are of the form

$$\theta_i = 2^{-n_\psi} \prod_{j=1}^{n_\psi} (1 \pm b_j).$$

By applying Lemma 6.20, we conclude that $\sum_{i\in\mathcal{J}} \theta_i = 1$ and the proof is complete. □

To pass from the definition in Eq. (6.42) to a dynamic complementarity system, we state the KKT conditions of (6.33) to obtain an algebraic expression for $\Gamma(\psi(x))$. Combining this with the definition of the Filippov set in (6.42) and the expression for θ_i in (6.43), we obtain the following DCS:

$$\dot{x} = F(x, u)\, \theta, \tag{6.44a}$$

$$0 = \theta_i - \prod_{j=1}^{n_\psi} \Big(\frac{1 - S_{i,j}}{2} + S_{i,j}\alpha_j \Big), \text{ for all } i \in \mathcal{J}, \tag{6.44b}$$

$$0 = \psi(x) - \lambda^{\mathrm{p}} + \lambda^{\mathrm{n}}, \tag{6.44c}$$

$$0 \leq \lambda^{\mathrm{n}} \perp \alpha \geq 0, \tag{6.44d}$$

$$0 \leq \lambda^{\mathrm{p}} \perp e - \alpha \geq 0, \tag{6.44e}$$

where $F(x,u) = [f_1(x,u), \ldots, f_{n_f}(x,u)] \in \mathbb{R}^{n_x \times n_f}$, $\theta = (\theta_1, \ldots, \theta_{n_f}) \in \mathbb{R}^{n_f}$ and $\lambda^{\mathrm{p}}, \lambda^{\mathrm{n}}, \alpha \in \mathbb{R}^{n_\psi}$. We group all algebraic equations into a single function and use a C-function $\Psi(\cdot,\cdot)$ for the complementarity condition to obtain a more compact expression:

$$G(x,\theta,\alpha,\lambda^{\mathrm{p}},\lambda^{\mathrm{n}}) \coloneqq \begin{bmatrix} \theta_1 - \prod_{j=1}^{n_\psi} \left(\frac{1-S_{1,j}}{2} + S_{1,j}\alpha_j\right) \\ \vdots \\ \theta_{n_f} - \prod_{j=1}^{n_\psi} \left(\frac{1-S_{n_f,j}}{2} + S_{n_f,j}\alpha_j\right) \\ \psi(x) - \lambda^{\mathrm{p}} + \lambda^{\mathrm{n}} \\ \Psi(\lambda^{\mathrm{n}}, \alpha) \\ \Psi(\lambda^{\mathrm{p}}, e - \alpha) \end{bmatrix}. \tag{6.45}$$

Finally, we obtain a compact representation of (6.44) in the form of a nonsmooth DAE:

$$\dot{x} = F(x,u)\theta, \tag{6.46a}$$

$$0 = G(x,\theta,\alpha,\lambda^{\mathrm{p}},\lambda^{\mathrm{n}}). \tag{6.46b}$$

6.3.4 Active-set changes and continuity of λ^{p} and λ^{n}

Active-set changes are paired with discontinuities in some of the algebraic variables. We have already seen in Section 6.2.3 that $\theta(t)$ is usually a discontinuous function of time. The LP Lagrange multipliers λ and μ in Stewart's reformulation are continuous functions of time. We have seen above that the same holds for λ^{p} and λ^{n}. We can exploit this property for the development of numerical simulation methods in the next chapter.

At an active-set change, we at least one of the switching functions $\psi_j(x)$ either becomes zero, or if it was zero it becomes nonzero. It follows from $\psi_j(x(t)) = \lambda_j^{\mathrm{p}}(t) - \lambda_j^{\mathrm{n}}(t)$ (in (6.35)), that also both $\lambda_j^{\mathrm{p}}(t)$ and $\lambda_j^{\mathrm{n}}(t)$ must be zero at an active-set change. Thus, they have qualitatively the same properties as λ in Stewart's formulation, cf. Section 6.2.3. We illustrate the different switching cases from Example 6.8 now with the DCS formulation via step functions in Figure 6.3. Whenever $x(t)$ has a kink, which corresponds to a switch and discontinuity in the dynamics, both the Lagrange multipliers $\lambda^{\mathrm{p}}(t)$ and $\lambda^{\mathrm{n}}(t)$ are zero at that time.

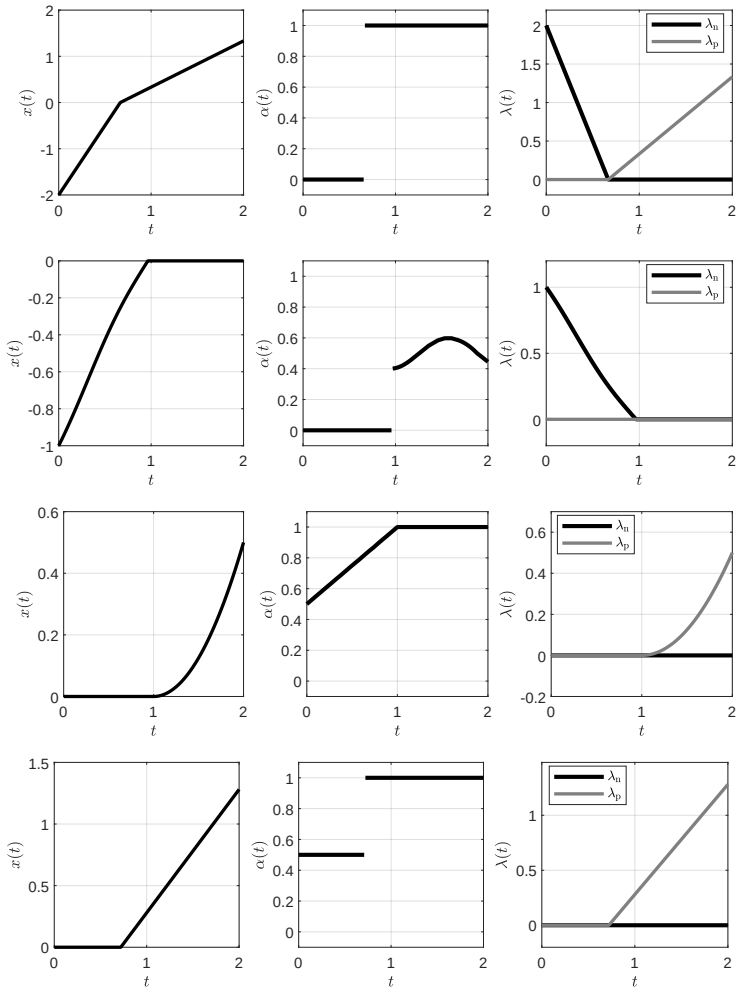


Figure 6.3: Illustration of example solution trajectories for different switching cases. The rows from top to bottom show $x(t)$, $\alpha(t)$, $\lambda^{\mathrm{p}}(t)$ and $\lambda^{\mathrm{n}}(t)$ for the cases (a)-(d) in Example 6.8, respectively.

6.3.5 Fixed active set in the Heaviside step formulation

We study the properties of the DCS (6.44) for a fixed active set $\mathcal{I}(x(t))$. Without loss of generality, the corresponding time interval is $\mathcal{T} = (0, T)$. For a fixed active set, the DCS (6.44) reduces either to an ODE or to a Differential Algebraic

Equation (DAE).

We start with the simpler ODE case. Let $\psi_j(x) \neq 0$ for all $j \in \mathcal{C} \coloneqq \{1, \ldots, n_\psi\}$, then $x(t)$ is in the interior of some region R_i. It can be seen from the LP (6.33) that $\alpha_j \in \{0, 1\}$ for all $j \in \mathcal{C}$. This implies that $\theta_i = 1$ and $\theta_k = 0, k \neq i$. Therefore, $\mathcal{I}(x(t)) = \{i\}$ and the Filippov DI reduces to $\dot{x} \in F_{\mathrm{F}}(x) = \{f_i(x)\}$, i.e., we have locally an ODE.

Next, we regard the case when $\mathcal{I}(x(t))$ is not a singleton, i.e., the trajectory evolves at the boundary of two or more regions. Consequently, we have at least one $\psi_j(x) = 0$. Let us associate with $\mathcal{I}(x(t))$ the index set $\mathcal{K}(x(t)) = \{j \in \mathcal{C} \mid \psi_j(x) = 0\}$, i.e., the set of indices of all switching functions that are zero for a given active set $\mathcal{I}(x(t))$.

In the sequel, we make use of the following notation. For a given vector $a \in \mathbb{R}^n$ and set $\mathcal{I} \subseteq \{1, \ldots, n\}$, we define the projection matrix $P_\mathcal{I} \in \mathbb{R}^{|I| \times n}$, which has zeros or ones as entries. It selects all component $a_i, i \in \mathcal{I}$ from the vector a, i.e., $a_\mathcal{I} = P_\mathcal{I} a \in \mathbb{R}^{|\mathcal{I}|}$ and $a_\mathcal{I} = [a_i \mid i \in \mathcal{I}]$.

Following the discussion from the previous section, for all nonzero $\psi_j(x)$, i.e., $j \in \mathcal{C} \setminus \mathcal{K}$, we can compute $\alpha_j \in \{0, 1\}$ via the LP (6.33) and $\lambda_{\mathrm{p},j}, \lambda_{\mathrm{n},j}$ via (6.35). Next, we have that $\lambda_{\mathrm{p},j} = \lambda_{\mathrm{n},j} = 0$ for all $j \in \mathcal{K}$. It is left to determine α_j for all $j \in \mathcal{K}$ and thus implicitly all θ_i, for all $i \in \mathcal{I}$. Recall that $\theta_i = 0$ for all $i \notin \mathcal{I}$. By fixing the already known variables in (6.44) we obtain the DAE:

$$\dot{x} = F_\mathcal{I}(x, u)\, \theta_\mathcal{I}, \tag{6.47a}$$

$$\theta_i - \prod_{j=1}^{n_\psi} \Big(\frac{1 - S_{i,j}}{2} + S_{i,j} \alpha_j \Big) = 0,\ i \in \mathcal{I}, \tag{6.47b}$$

$$\psi_j(x) = 0,\ j \in \mathcal{K}, \tag{6.47c}$$

where we define $F_\mathcal{I}(x, u) \coloneqq F(x, u) P_\mathcal{I}^\top \in \mathbb{R}^{n_x \times |\mathcal{I}|}$, i.e., we select only the columns of $F(x, u)$ with the index $i \in \mathcal{I}$. Note that α_j for all $j \in \mathcal{C} \setminus \mathcal{K}$ are known and thus no degrees of freedom. We keep them for ease of notation. Thus we have a DAE with $|\mathcal{I}| + |\mathcal{K}|$ unknowns, namely $\theta_\mathcal{I} \in \mathbb{R}^{|\mathcal{I}|}$ and $\alpha_\mathcal{K} \in \mathbb{R}^{|\mathcal{K}|}$, and $|\mathcal{I}| + |\mathcal{K}|$ algebraic equations in (6.47b) and (6.47c).

Next, we investigate conditions under which the DAE (6.47) is well-posed. For this purpose, we define the matrix

$$W_{\mathcal{K},\mathcal{I}}(x, u) = \nabla \psi_\mathcal{K}(x)^\top F_\mathcal{I}(x, u) \in \mathbb{R}^{|\mathcal{K}| \times |\mathcal{I}|},$$

where $\nabla \psi_\mathcal{K}(x) = \big[\nabla \psi_j(x) \mid j \in \mathcal{K}\big] \in \mathbb{R}^{n_x \times |\mathcal{K}|}$ is a matrix, whose columns are the gradients of the switching functions that are zero for the given active set

$\mathcal{I}$. Moreover, we define a compact notation for the partial Jacobian $B_{\mathcal{K},\mathcal{I}}(\alpha) \in \mathbb{R}^{|\mathcal{I}|\times|\mathcal{K}|}$ of (6.47b) w.r.t. to $\alpha_{\mathcal{K}}$, with the elements:

$$B_{i,j}(\alpha) \coloneqq \frac{\partial}{\partial \alpha_j} \prod_{l\in\mathcal{C}} \frac{1-S_{i,l}}{2} + S_{i,l}\alpha_l,\ i \in \mathcal{I}, j \in \mathcal{K}.$$

Assumption 6.22. *Given a fixed active set $\mathcal{I}(x(t)) = \mathcal{I}$ for $t \in \mathcal{T}$, it holds that the matrix functions $W_{\mathcal{K},\mathcal{I}}(x,u)$ and $B_{\mathcal{K},\mathcal{I}}(\alpha)$ are Lipschitz continuous, and that $W_{\mathcal{K},\mathcal{I}}(x,u)B_{\mathcal{K},\mathcal{I}}(\alpha)$ has rank $|\mathcal{K}|$, i.e. it is full rank, for all $t \in \mathcal{T}$.*

Proposition 6.23. *Suppose that Assumption6.22 holds. Given an initial value $x(0)$, the DAE (6.47) has a unique solution for all $t \in \mathcal{T}$.*

Proof. First, we differentiate (6.47c) with respect to t, such that algebraic variables appear explicitly in the algebraic equations (this correspond to so-called index reduction in the theory of DAEs, cf. [126]):

$$\dot{x} = F_{\mathcal{I}}(x,u)\ \theta_{\mathcal{I}}, \tag{6.48a}$$

$$\theta_i - \prod_{j\in\mathcal{C}} \frac{1-S_{i,j}}{2} + S_{i,j}\alpha_j = 0,\ i \in \mathcal{I}, \tag{6.48b}$$

$$W_{\mathcal{K},\mathcal{I}}(x,u)\theta_{\mathcal{I}} = 0. \tag{6.48c}$$

Next, we prove that the partial Jacobian of (6.48b)-(6.48c) w.r.t. to the algebraic variables $(\theta_{\mathcal{I}}, \alpha_{\mathcal{K}})$ has rank $|\mathcal{I}| + |\mathcal{K}|$, i.e., it is an invertible matrix. We omit the dependencies on α and x for brevity. The Jacobian of (6.48b)-(6.48c) w.r.t. to $(\theta_{\mathcal{I}}, \alpha_{\mathcal{K}})$ has the form

$$A = \begin{bmatrix} I_{|\mathcal{I}|} & -B_{\mathcal{K},\mathcal{I}} \\ W_{\mathcal{K},\mathcal{I}} & \mathbf{0} \end{bmatrix}.$$

To prove that this matrix has full rank, we show that the only solution $(v,w) \in \mathbb{R}^{|\mathcal{I}|} \times \mathbb{R}^{|\mathcal{K}|}$, to the following linear system is the zero vector:

$$\begin{bmatrix} I_{|\mathcal{I}|} & -B_{\mathcal{I},\mathcal{K}} \\ W_{\mathcal{K},\mathcal{I}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} = 0. \tag{6.49}$$

From the first line we have $v = B_{\mathcal{I},\mathcal{K}}w$ and substituting this into the second line we have $W_{\mathcal{I},\mathcal{K}}B_{\mathcal{I},\mathcal{K}}w = 0$. Since the matrix $W_{\mathcal{I},\mathcal{K}}B_{\mathcal{I},\mathcal{K}} \in \mathbb{R}^{|\mathcal{K}|\times|\mathcal{K}|}$ has rank $|\mathcal{K}|$, the only solution to (6.49) is $w = 0$, and $v = Bw = 0$. Hence, A has full rank.

Now we can apply the implicit function theorem [85, Theorem 1B.1] to (6.48b)-(6.48c), which guarantees the existence of continuously differentiable functions

$\theta_{\mathcal{I}}(x)$ and $\alpha_{\mathcal{K}}(x)$. Since the function $\theta_{\mathcal{I}}(x)$ is continuously differentiable, it is also Lipschitz continuous for a fixed $\mathcal{I}(x(t))$ on $t \in \mathcal{T}$. By substituting $\theta_{\mathcal{I}}(x)$ into (6.48a), we have a product of two Lipschitz continuous functions (all columns of $F_{\mathcal{I}}(x,u)$, are Lipschitz by assumption), and the DAE (6.48) reduces to an ODE with a Lipschitz continuous r.h.s. This enables us to apply Theorem 3.3 to obtain the assertion of the proposition. □

We make a few comments on Assumption 6.22. The rank condition can be checked explicitly, since one can simply compute the matrix $W_{\mathcal{K},\mathcal{I}}(x,u)B_{\mathcal{K},\mathcal{I}}(\alpha)$. We have already assumed Lipschitz continuity of all columns of $f_i(x,u)$ and of the gradients $\nabla\psi_j(x)$. Here we additionally assume it for the matrix $W_{\mathcal{K},\mathcal{I}}(x,u)$, whose entries are computed as inner products on these vectors. The entries of the matrix $B_{\mathcal{K},\mathcal{I}}(\alpha)$ are multi-affine terms, which are also Lipschitz, at least on the bounded domains we consider here. In Stewart's reformulation, we consider a square matrix with entries $\nabla g_i(x)^\top f_j(x,u)$ (which is structurally similar to $W_{\mathcal{K},\mathcal{I}}(x,u)$). For well-posedness with a fixed active set, the invertibility of this matrix is assumed [213, **?**].

In [74], the authors make some assumptions on the signs of the entries of $W_{\mathcal{K},\mathcal{I}}(x)$ and prove the existence, but not uniqueness, of solutions with a fixed-point argument. For the case of $|\mathcal{K}| \leq 2$, i.e., sliding modes with co-dimension one or two, and with additional assumptions on the signs of the entries of $W_{\mathcal{K},\mathcal{I}}(x)$ they even prove the uniqueness of solutions.

Observe that for a given $x(t)$, there might be several $\mathcal{I}(x(t))$ that yield meaningful DAEs of the form of Eq. (6.47). This may happen when the Filippov DI does not have unique solutions, such as in Example 6.8 case (d).

The trajectory may stay in sliding mode or leave it any point of time. The trajectory pieces in either scenario are well-posed, even thought the overall trajectory is not unique.

6.3.6 Predicting a new active set

Recall that in Stewart's reformulation, we have $\mathcal{I}(x) = \{i \in \mathcal{J} \mid g_i(x) = \min_{j\in\mathcal{J}} g_j(x)\}$. The definition of $\mathcal{I}(x)$ via the switching functions $c_j(x)$ is slightly more involved. This can be seen from the definition:

$$\mathcal{I}(x) = \{i \in \mathcal{J} \mid \theta_i > 0\}, \tag{6.50}$$

and the fact that θ_i is related to the functions $c_j(x)$ in a more complicated way. Using Eq. (6.43) and (6.50) we obtain:

$$\mathcal{I}(x) = \Big\{ i \in \mathcal{J} \mid \prod_{j=1}^{n_\psi} \Big(\frac{1 + (2\alpha_j - 1) S_{i,j}}{2} \Big) \gamma(\psi_j(x)) > 0 \Big\}.$$

The LCP in Theorem 6.11 is constructed from the total time derivative of λ. Here, we would need to use the total time derivatives of λ^{p} and λ^{n}. It is clear from Eq. (6.44) we cannot easily obtain an LCP which relates $\dot{\lambda}_{\mathrm{p}}$, $\dot{\lambda}_{\mathrm{n}}$ and θ. Therefore, we omit to derive such an auxiliary problem for predicting the next active set at a switching point. The price we pay is that we need slightly more restrictive assumptions to prove the convergence of the FESD scheme for the step reformulation (6.44). However, omitting this step has no consequences for the derivation of the FESD method (6.44) nor is it used in the computations.

6.3.7 Efficient modeling with Heaviside step functions

In this section, we show how to efficiently represent common geometries of the PSS regions with the use of step functions. This is useful for reducing the complexity of the modeling process. Moreover, we introduce a lifting algorithm, which makes the multi-affine terms for the Filippov multipliers with the help of auxiliary variables less nonlinear.

Overview of expressions for θ via step functions

Table 6.3 provides an overview of how are the elementary algebraic expressions for the multipliers θ_i are related to the geometric definition of a region R_i. For this purpose, we regard the following two sets: $A = \{x \mid \psi_A(x) > 0\}, B = \{x \mid \psi_B(x) > 0\}$, and let $\alpha_A \in \gamma(\psi_A(x))$ and $\alpha_B \in \gamma(\psi_B(x))$. All other more complicated expressions can be obtained by using the one listed in Table 6.3.

Remark 6.24 (Sum of Filippov systems)**.** *In practice, one often encounters DIs that arise from the sum of several Filippov systems. This occurs, for example, if we have multiple surfaces with friction, or multiple objects touching the same frictional surface [252]. All developments from this section can be extended to this case and are implemented in* `nosnoc`*. For the sake of brevity, we omit the corresponding equations as they follow similar lines as those derived in Section 6.2.5.*

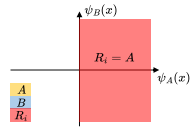
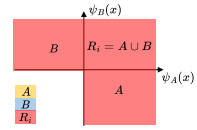
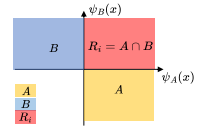
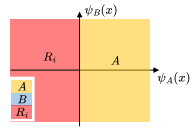
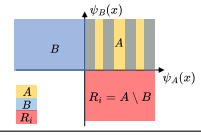
Definition of R_i	Expression for θ_i	Sketch
$R_i = A$	$\theta_i = \alpha_A$	
$R_i = A \cup B$	$\theta_i = \alpha_A + \alpha_B$	
$R_i = A \cap B$	$\theta_i = \alpha_A \alpha_B$	
$R_i = \text{int}(\mathbb{R}^{n_x} \setminus A) = \{x \mid \psi_A(x) < 0\}$	$\theta_i = 1 - \alpha_A$	
$R_i = A \setminus B$	$\theta_i = \alpha_A - \alpha_B$	

Table 6.3: Expressions of θ_i for different definitions of R_i.

6.3.8 A lifting algorithm for the multi-affine terms

Depending on the size of the matrix $S_{i,\bullet}$ has, the expression for θ_i in Eq. (6.43) might be very nonlinear since it is a product of n_ψ affine terms. To reduce the nonlinearity, we introduce auxiliary lifting variables as in the *lifted* Newton's method [8], which iterates on a larger but less nonlinear problem. We apply this approach to the expressions in Eq. (6.43) involving multi-affine terms with α_j. Whenever we have more than two terms in the multi-affine expression for θ_i, we introduce lifting variables β_k to end up with an equivalent formulation, which has only bilinear terms. We exploit the structure of the matrix S and derive an easy-to-implement algorithm that automates the lifting procedure. To give an idea of the final results we aim to obtain, we illustrate the lifting procedure with a simple example.

Example 6.25. *Regard a PSS with $n_\psi = 3$ switching functions and $n_f = 8$ modes, i.e., the PSS regions match the basis sets, $R_i = \tilde{R}_i$, $i = \{1, \ldots, 8\}$. The*

matrix $S \in \mathbb{R}^{8\times 3}$, and the expression for the multipliers $\theta \in \mathbb{R}^8$ read as

$$S = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1 \\ 1 & -1 & -1 \\ -1 & 1 & 1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \\ -1 & -1 & -1 \end{bmatrix}, \; G_{\mathrm{F}}(\theta, \alpha) = \begin{bmatrix} \theta_1 - \alpha_1\alpha_2\alpha_3 \\ \theta_2 - \alpha_1\alpha 2(1-\alpha_3) \\ \theta_3 - \alpha_1(1-\alpha_2)\alpha_3 \\ \theta_4 - \alpha_1(1-\alpha_2)(1-\alpha_3) \\ \theta_5 - (1-\alpha_1)\alpha_2\alpha_3 \\ \theta_6 - (1-\alpha_1)\alpha_2(1-\alpha_3) \\ \theta_7 - (1-\alpha_1)(1-\alpha_2)\alpha_3 \\ \theta_8 - (1-\alpha_1)(1-\alpha_2)(1-\alpha_3) \end{bmatrix} = 0$$

We can introduce the lifting variable $\beta \in \mathbb{R}^4$ and obtain

$$G_\beta(\alpha, \beta) = \begin{bmatrix} \beta_1 - \alpha_1\alpha_2 \\ \beta_2 - \alpha_1(1-\alpha_2) \\ \beta_3 - (1-\alpha_1)\alpha_2 \\ \beta_4 - (1-\alpha_1)(1-\alpha_2) \end{bmatrix} = 0, \; G_\theta(\theta, \alpha, \beta) = \begin{bmatrix} \theta_1 - \beta_1\alpha_3 \\ \theta_2 - \beta_1(1-\alpha_3) \\ \theta_3 - \beta_2\alpha_3 \\ \theta_4 - \beta_2(1-\alpha_3) \\ \theta_5 - \beta_3\alpha_3 \\ \theta_6 - \beta_3(1-\alpha_3) \\ \theta_7 - \beta_4\alpha_3 \\ \theta_8 - \beta_4(1-\alpha_3) \end{bmatrix} = 0$$

The equation $G_\beta(\alpha, \beta) = 0$ relates the lifting variables β_i with the α_j, whereas $G_\theta(\theta, \alpha, \beta)$ provides new expressions for θ_i via β_i and the remaining α_j. By replacing $G_{\mathrm{F}}(\theta, \alpha) = 0$ with $G_{\mathrm{lift}}(\theta, \alpha, \beta) \coloneqq (G_\beta(\alpha, \beta), G_\theta(\theta, \alpha, \beta)) = 0$, we obtain an equivalent system of equations that only consists of bilinear terms.

We proceed by outlining a general lifting algorithm. The expressions for θ_i consist of the product of n_ψ affine terms. Our goal is to have at most n_{d} terms in the multi-affine expression for θ_i. For example, if we pick $n_{\mathrm{d}} = 2$, we have only bilinear expressions in the equations defining θ and β. Thus, the parameter $n_\psi \geq n_{\mathrm{d}} \geq 2$, controls the number of terms in the multi-affine expressions and implicitly the number of new lifting variables $\beta \in \mathbb{R}^{n_\beta}$. Given the matrix S, our goal is to automatically obtain the constraint $G_{\mathrm{lift}}(\theta, \alpha, \beta) = 0$.

The algorithm outlined above can be implemented using a symbolic framework such as `CasADi` [9]. We provide the pseudo code in Algorithm 1, which introduces the lifting algebraic variables β and new *lifted* expressions for θ_i, namely $G_{\mathrm{lift}}(\theta, \alpha, \beta)$. Note that we make use of three helper variables, the matrix $\tilde{S}$ and the vectors $\tilde{\theta}$ and $\tilde{N}$. The matrix $\tilde{S}$ is a submatrix of S, where we have removed the rows with index i, for which we already have a (lifted) expression for θ_i. The vector $\tilde{N}$, defined in line 4, collects the number of nonzero entries of every row $\tilde{S}$. In other words, it keeps track of how many terms are in the initial expressions for θ_i, that are not yet lifted.

Algorithm 1 Lifting algorithm for the step DCS (6.44)

1: **Input:** S, n_{d}
2: **Initialize:** $\tilde{S} \leftarrow S$, $k \leftarrow 0$; $\tilde{\theta} \leftarrow e \in \mathbb{R}^{n_f}$, $G_\theta(\theta, \alpha, \beta) \leftarrow [\,]$, $G_\beta(\alpha, \beta) \leftarrow [\,]$.
3: **for** $j = 1 : n_\psi$ **do**
4: $\quad \tilde{N} \leftarrow \sum_{j=1}^{n_\psi} |\tilde{S}_{\bullet,j}|$
5: $\quad \mathcal{I}_j \leftarrow \{i \mid \tilde{N}_i = j\}$
6: $\quad \tilde{\theta} \leftarrow \tilde{\theta} \cdot \left(\frac{e - \tilde{S}_{\bullet,j}}{2} + S^{\mathrm{temp}}_{\bullet,j} \cdot \alpha_j \right)$
7: $\quad$ **if** $\mathcal{I}_j \neq \emptyset$ **then**
8: $\quad\quad G_\theta(\theta, \alpha, \beta) \leftarrow (G_\theta(\theta, \alpha, \beta), \theta_{k+\mathcal{I}_j} - \tilde{\theta}_{\mathcal{I}_j})$
9: $\quad\quad$ Remove entries of $\tilde{\theta}$ with index in $\mathcal{I}_j$
10: $\quad\quad$ Remove rows of $\tilde{S}$ with index in $\mathcal{I}_j$
11: $\quad\quad k \leftarrow k + \max(\mathcal{I}_j)$
12: $\quad$ **end if**
13: $\quad$ **if** $j \in \{n_{\mathrm{d}}, \ldots, n_\psi - 1\}$ **then**
14: $\quad\quad \{\mathcal{I}_{\mathrm{red}}, \mathcal{I}_{\mathrm{full}}\} = \mathtt{unique}(\tilde{S}_{\bullet,\{1,\ldots,j\}})$,
15: $\quad\quad \beta \leftarrow (\beta, \beta^j)$ where $\beta^j \in \mathbb{R}^{|\mathcal{I}_{\mathrm{red}}|}$
16: $\quad\quad G_\beta(\alpha, \beta) \leftarrow (G_\beta(\alpha, \beta)\beta^j - \tilde{\theta}_{\mathcal{I}_{\mathrm{red}}})$
17: $\quad\quad \tilde{\theta} \leftarrow \beta^j_{\mathcal{I}_{\mathrm{full}}}$
18: $\quad$ **end if**
19: **end for**
20: $G_{\mathrm{Lift}}(\theta, \alpha, \beta) \coloneqq (G_\theta(\theta, \alpha, \beta), G_\beta(\alpha, \beta))$
21: **Output:** $G_{\mathrm{Lift}}(\theta, \alpha, \beta)$, β

The main loop iterates from $j = 1$ to n_ψ and provides in every iteration the expressions for all θ_i that have exactly j terms in their multi-affine expression. The index set $\mathcal{I}_j = \{i \mid \tilde{N}_i = j\}$, defined in line 5, contains the indices of θ, that have exactly j entries in their corresponding multi-affine expression. In line 6, we define the auxiliary variable $\tilde{\theta}$, which stores the intermediate expressions for θ with up to j terms in the product. The index k stores the index of the last θ_k for which a lifted expression was derived. For $j \leq n_{\mathrm{d}}$ the expressions for θ_i are unaltered. This is treated in lines 7-11.

As soon as $j > n_{\mathrm{d}}$, the algorithm introduces new lifting variables β^j (line 15) and changes the expression for $\tilde{\theta}$ accordingly. This is done in lines 13-17. A key tool is the function `unique` in line 14. It is available in `MATLAB` and in the `numpy` package in `python`. It works as follows, given a matrix $A \in \mathbb{R}^{m \times n}$ it returns the matrix $\tilde{A} \in \mathbb{R}^{p \times n}$, with $p \leq m$. This is the matrix constructed from A by removing its repeating rows. More importantly for our needs, it returns the index sets $\mathcal{I}_{\mathrm{red}}$ and $\mathcal{I}_{\mathrm{full}}$, with $|\mathcal{I}_{\mathrm{red}}| = p$ and $|\mathcal{I}_{\mathrm{full}}| = m$. The index sets have the properties $A = \left[\tilde{A}_{i,\bullet} \mid i \in \mathcal{I}_{\mathrm{full}}\right] \in \mathbb{R}^{m \times n}$ and $\tilde{A} = \left[A_{i,\bullet} \mid i \in \mathcal{I}_{\mathrm{red}} \in \mathbb{R}^{p \times n}\right]$.

Method	n_f	n_β	n_{alg}	n_{comp}	n_{eq}
Stewart	2^{n_ψ}	0	$2 \cdot 2^{n_\psi}+1$	2^{n_ψ}	$2^{n_\psi}+1$
Heaviside	$[2, 2^{n_\psi}]$	$\begin{cases} 2^{n_\psi}-2^{n_{\text{d}}}, & n_{\text{d}} \leq n_\psi \\ 0, & n_{\text{d}} > n_\psi \end{cases}$	$n_f+3n_\psi+n_\beta$	$2n_\psi$	$n_\psi+n_\beta+n_f$

Table 6.4: Comparisons of the problem sizes in Stewart's and the step reformulation for a fixed n_ψ.

This enables the use of the same β^j for several θ_i if they share the same terms in the corresponding multi-affine expressions, cf. lines 15-17. After the loop is finished, the algorithm outputs $G_{\text{Lift}}(\theta, \alpha, \beta)$ and β. One can verify that Algorithm 1 produces the same output as Example 6.25. It can be shown, that for a given $n_{\text{d}} < n_\psi$, the total number of new lifting variables is $n_\beta = 2^{n_\psi} - 2^{n_{\text{d}}}$.

6.3.9 Comparisons of Stewart's and the Heaviside step reformulation

We compare Stewart's reformulation (6.8) and the Heaviside step reformulation (6.44) based on their total number of algebraic variables, complementarity constraints, and equality constraints for a given number of switching functions n_ψ. The total number of regions (and multipliers θ_i) in Stewart's reformulation is always $n_f = 2^{n_\psi}$. In this reformulation, in order to reduce the number of multipliers θ_i, we cannot exploit the setting if regions R_i are defined as unions of basis sets $\tilde{R}_j$, cf. Section 6.2. On the other hand, in the step reformulation, depending on the geometry of the regions R_i, n_f is an integer in $[2, 2^{n_\psi}]$. In the step reformulation, we may introduce n_β lifting variables to reduce the nonlinearity. If $n_{\text{d}} > n_\psi$, this leads to $n_\beta = 2^{n_\psi} - 2^{n_{\text{d}}}$ additional lifting variables and equations.

Regarding the number of algebraic variables, in Stewart's reformulation, we have $\lambda \in \mathbb{R}^{2^{n_\psi}}$ and $\mu \in \mathbb{R}$. In the step reformulation, we have $\alpha, \lambda^{\text{p}}, \lambda^{\text{n}} \in \mathbb{R}^{n_\psi}$. Thus, the total number of algebraic variables in the former case is $n_{\text{alg}} = 2 \cdot 2^{n_\psi} + 1$, and in the later case $n_{\text{alg}} = n_f + 3n_\psi + n_\beta$. The number of complementarity constraints n_{comp} in Stewart's case is $n_{\text{comp}} = 2^{n_\psi}$, and in the step case $n_{\text{comp}} = 2n_\psi$, i.e., we have exponential versus linear complexity. Finally, in Stewart's reformulation, we have in total $n_{\text{eq}} = 2^{n_\psi} + 1$ equality constraints ($g_i(x) = \lambda_i - \mu$ and $e^\top \theta = 1$). In the step case, we have $n_{\text{eq}} = n_f + n_\beta + n_\psi$ constraints, for the definitions of θ_i, β_i and the constraints $\psi_i(x) = \lambda_i^{\text{p}} - \lambda_i^{\text{n}}$, respectively. The numbers of variables and constraints are summarized in Table 6.3.9.

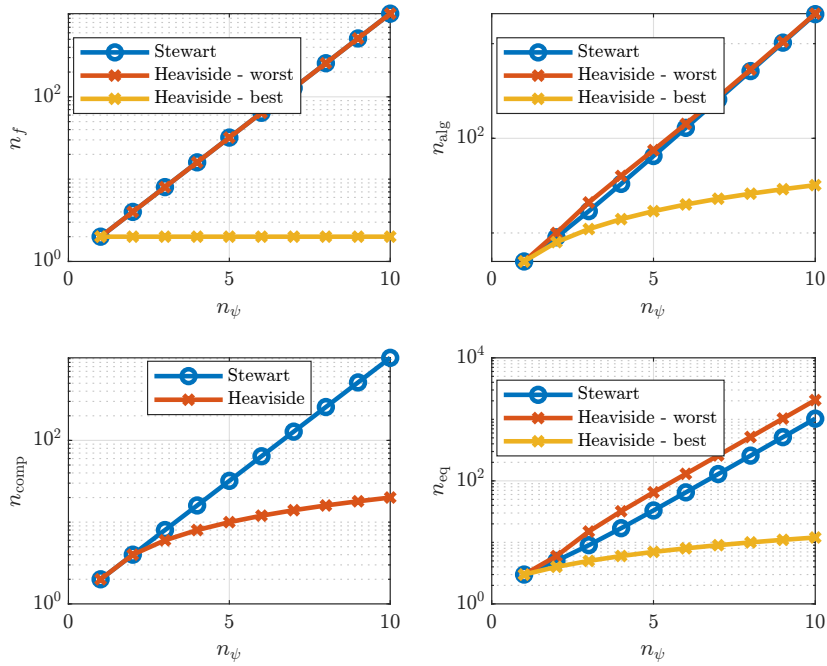


Figure 6.4: Comparisons of the complexities of Stewart's and the Heaviside step reformulation.

Figure 6.4 illustrates the different quantities for several n_ψ. We plot for the step reformulation two extreme scenarios:

1. Worst complexity case - every basis set defines a PSS region, $n_f = 2^{n_\psi}$, we lift to have only bilinear terms, i.e., $n_{\text{d}} = 2$ (maximizes the number of lifting variables).

2. Best complexity case - o no lifting and only two regions ($n_f = 2$) for all n_ψ.

Note that in both cases the step reformulation has the same number of complementarity constraints. For smaller values of n_ψ, both reformulations have similar complexity. For a large number of switching functions, the step reformulation has fewer variables. However, if there is no lifting, the problem might become very nonlinear for large n_ψ.

6.4 Conclusions and summary

In this section we relate the various formalism discussed in this capter and provide some concluding remarks.

6.4.1 Relations between different formalisms

The diagram in Figure 6.5 summarizes the relationships between the nonsmooth systems studied in this chapter. The first column consists of the different ODEs with DRHS that we have treated. After treating a generic ODE with DRHS in (4.14), we specialize in two structured cases: PSS in (6.1) and ODEs (6.36), where their dynamics contain some discontinuous expressions of x, e.g., as Heaviside step functions.

These ODEs may not have a classical or Carathéodory solution, so we embed them into differential inclusions and obtain more general notions, such as the Filippov solution. These concepts are summarized in the second column. A generic ODE with DRHS (4.14) is generalized to Filippov DIs (6.2). If the ODE is more structured as a piecewise smooth system, then its Filippov extensions become more explicit (6.3). In both cases, the r.h.s. is now a convex and compact set. If the discontinuous function in the structured ODE (6.36) is replaced by set-valued extensions, we obtain an Aizerman–Pyatnitskii DI (6.37). The set on the r.h.s. may even be nonconvex. In Proposition 6.21 we show that when we use PSS and Heaviside step functions, the Aizerman-Pyatnitskii DI and the Filippov extension for PSS are equivalent.

The third column consists of dynamic complementarity systems obtained from the DIs, which are useful representations for numerical computations. The Stewart DCS (6.8) and the Heaviside step DCS (6.44) and are both instances of a generic DCS (4.13). They are derived from the corresponding differential inclusions using the KKT conditions of parametric LPs, as shown in Sections 6.2 and 6.3, respectively. We conclude this section by illustrating the different formalisms with an example.

Example 6.26. *Let us illustrate the different classes of nonsmooth systems on the discontinuous ODE (which is in the form of* (4.14)*):*

$$\dot{x} = \begin{cases} 1, & x > 0, \\ 3, & x < 0. \end{cases}$$

This is a PSS (as in (6.1)*) with the switching function $\psi(x) = x$, with the regions $R_1 = \{x \mid x > 0\}$ and $R_2 = \{x \mid x < 0\}$. The Filippov extensions*

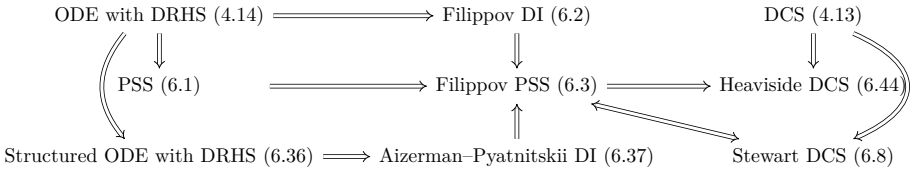


Figure 6.5: Summary of relations between nonsmooth systems treated in this paper.

of this PSS (Eq. (6.3)*) reads as:* $\dot{x} \in \{\theta_1 + 3\theta_2 \mid \theta \geq 0, \theta_1 + \theta_2 = 1\}$*, with* $\theta = (\theta_1, \theta_2)$*. In the form of an Aizerman–Pyatnitskii DI* (6.38)*, the system reads as* $\dot{x} \in 3 - 2\gamma(x)$*. We can write also as a DCS of the form of* (6.44)*:*

$$\dot{x} = \begin{bmatrix} 3 & 1 \end{bmatrix} \theta,$$

$$\theta_1 = \alpha,\ \theta_2 = 1 - \alpha,\ x = \lambda^{\mathrm{p}} - \lambda^{\mathrm{n}},$$

$$0 \leq \lambda^{\mathrm{p}} \perp \alpha \geq 0,\ 0 \leq \lambda^{\mathrm{n}} \perp 1 - \alpha \geq 0.$$

Similarly, by defining the indicator function $g(x) = (-x, x)$*, we can state Stewart DCS* (6.8)*:*

$$\dot{x} = \begin{bmatrix} 3 & 1 \end{bmatrix} \theta,$$

$$-x = \lambda_1 - \mu,\ x = \lambda_2 - \mu,\ \theta_1 + \theta_2 = 1,$$

$$0 \leq \theta_1 \perp \lambda_1 \geq 0,\ 0 \leq \lambda_2 \perp \theta_2 \geq 0.$$

6.4.2 Summary

This chapter has focused on piecewise smooth systems and their Filippov extensions, with a particular emphasis on the derivation of an equivalent Dynamic Complementarity System (DCS) from a Filippov system. This is motivated by the powerful computational methods and theoretical results for complementarity problems. After an appropriate discretization of an optimal control problem with such a DCS, we obtain a mathematical complementarity problem, which usually can be efficiently solved with the methods described in Section 2.4.

The guiding idea in transforming a Filippov system into a DCS is to express the Filippov multipliers θ as the solution of a parametric linear program. The KKT conditions of this problem enable us to transform a Filippov DI into

an equivalent DCS. Thereby, we followed two different approaches. The first reformulation is introduced by Stewart [250], and the second one is based on set-valued Heaviside step functions, which appeared in the literature in similar forms at different locations [6, 74]. In the Heaviside step reformulation, we directly work with the switching functions $\psi(x)$ to derive the LP. Stewart's approach needs specific indicator functions $g(x)$. In Section 6.2.1, we show that they can always be derived from a more intuitive representation via switching functions $\psi(x)$. In both cases, we rely on representing the definition of the regions R_i compactly via a sign matrix S.

We study in detail the properties of the DCS obtained by both approaches. In particular, we study the well-possesses of the ODEs or DAEs obtained by fixing the active set in the DCS. We show that the Lagrange multipliers of the LPs are continuous functions of time, even across active-set changes. In the next chapter, we leverage this property to develop highly accurate discretization methods for Filippov systems.

A natural question to ask is: Why do we need different reformulations at all? It is no surprise that both reformulations share similar theoretical properties. Stewart's reformulation is very efficient if all components of the switching function $\psi(x)$ are involved in the definitions of the regions R_i, which results in a dense matrix S. These are the cases where it is advantageous to use it over the Heaviside step reformulation. Moreover, Stewart's reformulation is very efficient if the regions R_i arise from a Voronoi partition as in Chapter 9.

However, as soon as some function $\psi_j(x)$ is not involved in the definition of a region R_i, there is some redundancy in Stewart's definition of the regions R_i. In this case, it is beneficial to use the step reformulations, as we need fewer algebraic variables. Depending on number of switching functions defining a region R_i, the expressions for the Filippov multipliers θ_i might be very nonlinear. Motivated by the ideas of the lifted Newton's method [8], in Section 6.3.8, we introduce a lifting algorithm, which provides an equivalent formulation with more variables that is only mildly nonlinear. Furthermore, the step reformulation is also suitable to represent more general systems than Filippov systems, e.g., Aizerman–Pyatnitskii inclusions [6, 94].

The obvious question to be addressed in future work is: Are there other convex problem formulations whose solutions can be related to θ? This would enable us to derive further similar DCSs and to extend the ideas from the next chapter also to these cases. The more practical questions are: What is the most compact representation for a given problem? Can one guarantee that the specific LP for a given problem results in a DCS with a minimal number of algebraic variables? It was shown in [136] that every piecewise continuous affine function can be represented as the solution to a parametric linear program. It would be

interesting to derive formulae in this manner also for the multiplier θ.

Chapter 7

Finite Elements with Switch Detection

This chapter introduces the method of Finite Elements with Switch Detection (FESD), a numerical discretization method for Filippov differential inclusions arising from piecewise smooth systems. In the previous chapter, we derived dynamic complementarity systems equivalent to such Filippov systems. Here we develop a discretization method of higher-order accuracy for these systems. We discretize optimal control problems with Filippov systems with the FESD method and obtain mathematical programs with complementarity constraints (MPCCs). With appropriate reformulations, MPCCs can be efficiency solved with nonlinear programming methods, cf. Section 2.4. Thus, we obtain highly accurate solutions to nonsmooth optimal control problems solely based on continuous optimization techniques. Notably, we do not need to use integer variables or nonsmooth optimization methods. We show that FESD enables us to overcome the fundamental limitations of standard direct methods discussed in Chapter 5.

Outline. In Section 7.1, we highlight the main ideas and mention some related work that we build upon. In Section 7.2, we start from a standard RK discretization and derive the FESD method step-by-step based on the algorithmic ingredients mentioned above. Thereby, we focus on Stewart's reformulation for Filippov systems, cf. Section 6.2. Afterwards, in Section 7.3 several relevant theoretical properties of the FESD method are studied. The theoretical findings are illustrated on numerical simulation and optimal control examples. The derivation of the FESD method for the step reformulation (cf. Section 6.3) relies

on the same algorithmic ideas as for Stewart's reformulation. For completeness, we provide in Section 7.4 the derivation of FESD for this case as well. We avoid on purpose to state both DCS in a more abstract form and to do the derivations only once, as this would hide some important details, which are important for the implementation. Section 7.5 discusses how to use FESD in numerical optimal control. Finally, Section 7.6 summarizes the chapter and outlines future research. Sections 7.2, 7.3 and 7.5 are mainly based on the article [213]. Section 7.4 is based on [207].

7.1 Introduction and related work

We build on the idea of varying the step size and allowing switches only to take place at the boundaries of the finite elements introduced by Baumrucker and Biegler [28]. On the one hand, these degrees of freedom introduce additional nonlinearity and nonconvexitiy. On the other hand, they are needed to overcome the fundamental limitations of low accuracy and wrong sensitivities of standard fixed step size (time-stepping) methods. The method of [28] can deal only with systems with one switching function (or sums of Filippov subsystems with a single switching function, cf. Section 6.2.5). Furthermore, it regards only Radau IIA implicit Runge-Kutta methods.

Biegler and coworkers [66, 65, 10] use also similar ideas to derive discrete-time formulations of singular optimal control problems. The goal is to exactly detect the location of control function's breakpoints. These methods usually include a mesh refinement stage to add more elements based on some error measurements. They are used both in direct [66, 65], and indirect optimal control [10].

In this chapter, we introduce the method of Finite Elements with Switch Detection (FESD), which is based on three key ingredients. We start with a standard Runge-Kutta (RK) discretization for the DCS from the previous chapter. First, inspired by [28], we let the integrator step sizes be degrees of freedom to enable the detection of switches exactly in time. However, this introduces more degrees of freedom than we have conditions in the RK-equations. Thus, we need to introduce more conditions, which lead to the two remaining ingredients. The second ingredient is additional complementarity conditions that we call the *cross complementarity conditions*. They enable exact switch detection, which enables us to recover the high-order accuracy that the RK methods enjoy for smooth ODE, cf. Section 3.2. Furthermore, this also enables the correct computation of numerical sensitivities. However, if no switches occur, the cross complementarity conditions are trivially satisfied. Therefore, we have an under-determined system of equations with spurious degrees of

freedom. The third and last ingredient are conditions that allow the step size to change only when switches occur and to overcome this issue. We call this *step equilibration*.

We have seen in Sections 4.1.3 and 4.2.5, that if standard Runge-Kutta (RK) methods are naively applied to a nonsmooth ODE, their accuracy is at best of order one. Of course, the FESD method can also be used to simulate Filippov systems (an example is given in Section 7.3.4). The FESD method falls into the class of event-driven methods. However, in contrast to most such methods, it does not need a separate root-finding procedure for switch detection. In Section 5.2, we have studied the fact that the numerical sensitivities obtained from a standard discretization are always wrong. This harms the progress of most direct methods for optimal control problems with nonsmooth dynamical systems, which are reviewed in Section 5.1.

The FESD method can efficiently deal with multiple and simultaneous switches, including sliding modes on higher co-dimension surfaces, and thus is more general than [28]. Moreover, in FESD one can use any Runge-Kutta method. As a theoretical contribution, we provide convergence results for the FESD method, show the local uniqueness of the solutions and prove the convergence of numerical sensitivities to the correct values. Notably, in an optimal control problem benchmark with FESD, we achieve up to five orders of magnitude more accurate solutions than a standard approach for the same computational time. Chapters 8 and 9 introduce exact reformulations of several classes of dynamical systems with state jumps (NSD3) into Filippov systems (NSD2). Remarkably, one can seamlessly apply the FESD method to these systems and thus obtain a high-accuracy discretization of optimal control problems with NSD3 systems. The FESD method, with its many variations, is implemented in the open-source software package `nosnoc` [2, 206].

7.2 FESD for Stewart's reformulation

This section is structured as follows. In Subsection 7.2.1, we first state the standard RK equations for Stewart's DCS from the previous chapter. Afterwards, we extend these equations by the three ingredients needed for FESD. In Subsection 7.2.2, we let the step sizes be degrees of freedom and discuss the consequences. In Subsection 7.2.3, we introduce the cross complementarity conditions. They ensure exact switch detection and that active-set changes can happen only on finite element's boundary. However, if no switches occur the cross complementarity conditions are implied by the standard complementarity conditions. In this case, the step sizes h_n are still allowed to vary, and we

encounter spurious degrees of freedom. This would let the optimizer play with the integrator's accuracy in a possibly undesired way. To resolve this, we introduce in Subsection 7.2.4 the step equilibration conditions. They provide additional equations only if no switches occur and lead to a well-posed problem. Finally, in Section 7.2.5, we summarize all conditions that extend the standard RK equations and lead to the FESD discretization.

7.2.1 Standard Runge-Kutta discretization

In the exposition of the FESD method, we regard a single control interval $[0, T]$ with a constant externally chosen control input $q \in \mathbb{R}^{n_u}$, i.e., we set $u(t) = q$ for $t \in [0, T]$. Extensions with more complex smooth parametrizations of the control function are straightforward.

As a starting point in our analysis, we regard a standard Runge-Kutta (RK) discretization of the DCS (6.8), which we restate for the reader's convenience:

$$\dot{x} = F(x, u)\theta, \tag{7.1a}$$

$$0 = g(x) - \lambda - \mu e, \tag{7.1b}$$

$$1 = e^\top \theta, \tag{7.1c}$$

$$0 \leq \theta \perp \lambda \geq 0, \tag{7.1d}$$

For a more compact notation, we work with the nonsmooth DAE formulation of the DCS (6.10), which we restate:

$$\dot{x} = F(x, u)\theta, \tag{7.2a}$$

$$0 = G(x, \theta, \lambda, \mu). \tag{7.2b}$$

Suppose the initial value $x(0) = s_0$ is given. We divide the control interval $[0, T]$ into N_{FE} finite elements (i.e., integration intervals) $[t_n, t_{n+1}]$ via the grid points $0 = t_0 < t_1 < \ldots < t_{N_{\mathrm{FE}}} = T$. On each of the finite elements we regard an n_{s}-stage Runge-Kutta method which is characterized by the Butcher tableau entries $a_{i,j}, b_i$ and c_i with $i, j \in \{1, \ldots, n_{\mathrm{s}}\}$ [126], see also Section 3.2. The fixed step-size reads as $h_n = t_{n+1} - t_n,\ n = 0, \ldots, N_{\mathrm{FE}} - 1$. The approximation of the differential state at the grid points t_n is denoted by $x_n \approx x(t_n)$.

We regard a differential representation of the Runge-Kutta method, where the derivatives of states at the stage points $t_{n,i} \coloneqq t_n + c_i h_n,\ i = 1, \ldots, n_{\mathrm{s}}$, are degrees of freedom, cf. Definition 3.8. For a single finite element, they are

summarized in the vector $V_n := (v_{n,1}, \dots, v_{n,n_\mathrm{s}}) \in \mathbb{R}^{n_\mathrm{s} n_x}$. The stage values for the algebraic variables are collected in the vectors: $\Theta_n := (\theta_{n,1}, \dots, \theta_{n,n_\mathrm{s}}) \in \mathbb{R}^{n_\mathrm{s} \cdot n_f}$, $\Lambda_n := (\lambda_{n,1}, \dots, \lambda_{n,n_\mathrm{s}}) \in \mathbb{R}^{n_\mathrm{s} \cdot n_f}$ and $M_n := (\mu_{n,1}, \dots, \mu_{n,n_\mathrm{s}}) \in \mathbb{R}^{n_\mathrm{s}}$. We also define the vector $Z_n = (x_n, \Theta_n, \Lambda_n, M_n, V_n)$ which collects all internal variables. With x_n^{next} we denote the value at t_{n+1}, which is obtained after a single integration step. Finally, the RK equations for a single finite element for the DCS (7.2) are given by:

$$G_{\mathrm{rk}}(x_n^{\mathrm{next}}, Z_n, h_n, q) := \begin{bmatrix} v_{n,1} - F(x_n + h_n \sum_{j=1}^{n_\mathrm{s}} a_{1,j} v_{n,j}, q)\theta_{n,1} \\ \vdots \\ v_{n,n_\mathrm{s}} - F(x_n + h_n \sum_{j=1}^{n_\mathrm{s}} a_{n_\mathrm{s},j} v_{n,j}, q)\theta_{n,n_\mathrm{s}} \\ G(x_n + h_n \sum_{j=1}^{n_\mathrm{s}} a_{1,j} v_{n,j}, \theta_{n,1}, \lambda_{n,1}, \mu_{n,1}) \\ \vdots \\ G(x_n + h_n \sum_{j=1}^{n_\mathrm{s}} a_{n_\mathrm{s},j} v_{n,j}, \theta_{n,n_\mathrm{s}}, \lambda_{n,n_\mathrm{s}}, \mu_{n,n_\mathrm{s}}) \\ x_n^{\mathrm{next}} - x_n - h_n \sum_{i=1}^{n_\mathrm{s}} b_i v_{n,i} \end{bmatrix} = 0. \tag{7.3}$$

They are similar to the RK equations for semi-explicit index-1 DAE, cf. Section 3.2.1 and Eq. (3.12).

Next, we summarize the equations for all N_{FE} finite elements over the whole interval $[0, T]$ in a discrete-time system manner. For this purpose, we introduce some additional shorthands. All variables of all finite elements for a single control interval are collected in the vectors $\mathbf{x} = (x_0, \dots, x_{N_{\mathrm{FE}}}) \in \mathbb{R}^{(N_{\mathrm{FE}}+1)n_x}$, $\mathbf{V} = (V_0, \dots, V_{N_{\mathrm{FE}}-1}) \in \mathbb{R}^{N_{\mathrm{FE}} n_\mathrm{s} n_x}$ and $\mathbf{h} := (h_0, \dots, h_{N_{\mathrm{FE}}-1}) \in \mathbb{R}^{N_{\mathrm{FE}}}$. Note that the simple continuity condition $x_{n+1} = x_n^{\mathrm{next}}$ holds. We collect all stage values of the Filippov multipliers in the vector $\mathbf{\Theta} = (\Theta_0, \dots, \Theta_{N_{\mathrm{FE}}-1}) \in \mathbb{R}^{n_\theta}$, where $n_\theta = N_{\mathrm{FE}} n_\mathrm{s} n_f$. The vectors $\mathbf{\Lambda} \in \mathbb{R}^{n_\theta}$, $\mathbf{M} \in \mathbb{R}^{n_\mu}$ for the stage values of the Lagrange multipliers are defined accordingly, with $n_\mu = N_{\mathrm{FE}} n_\mathrm{s}$. The vector $\mathbf{Z} = (\mathbf{x}, \mathbf{V}, \mathbf{\Theta}, \mathbf{\Lambda}, \mathbf{M}) \in \mathbb{R}^{n_\mathbf{Z}}$ collects all internal variables and $n_\mathbf{Z} = (N_{\mathrm{FE}} + 1)n_x + N_{\mathrm{FE}} n_\mathrm{s} n_x + 2n_\theta + n_\mu$.

All computations over a single control interval, which we call here the *standard discretization* are summarized in the following equations, which resemble a discrete-time system:

$$s_1 = F_{\mathrm{std}}(\mathbf{Z}), \tag{7.4a}$$

$$0 = G_{\mathrm{std}}(\mathbf{Z}, \mathbf{h}, s_0, q), \tag{7.4b}$$

where $s_1 \in \mathbb{R}^{n_x}$ is the approximation of $x(T)$ and

$$F_{\mathrm{std}}(\mathbf{Z}) = x_{N_{\mathrm{FE}}},$$

$$G_{\text{std}}(\mathbf{Z}, \mathbf{h}, s_0, q) \coloneqq \begin{bmatrix} x_0 - s_0 \\ G_{\text{rk}}(x_1, Z_0, h_0, q) \\ \vdots \\ G_{\text{rk}}(x_{N_{\text{FE}}}, Z_{N_{\text{FE}}-1}, h_{N_{\text{FE}}-1}, q) \end{bmatrix}.$$

Note that $\mathbf{h}$ are given parameters implicitly fixed by the chosen discretization grid. In the nonsmooth ODE community, these methods are known as *time-stepping* methods, cf. Section 4.1.3. In contrast to event-based methods, they assume fixed step-sizes h_n and do not try to detect the switches. The theoretical properties of RK methods for DI and DCS have been studied by many authors, e.g., [83, 161, 264, 258]. It is usually impossible to obtain high-accuracy solutions with this method, as this can only happen if active-set changes happen coincidentally at t_n. Despite the high accuracy in this unlikely case, the numerical sensitivities would still be wrong, cf. Section 5.2.

When active-set changes happen within a finite element, the smooth RK method tries to approximate a nonsmooth trajectory. An example is illustrated in the left plot in Figure 7.1. Thereby, we try to approximate a nonsmooth function by a smooth polynomial, which results in a poor approximation. We proceed now with extending the equations (7.4) by lettings $\mathbf{h}$ to be degrees of freedom and defining the cross complementarity conditions.

7.2.2 The step-sizes as degrees of freedom

To obtain high-accuracy approximations of $x(\cdot)$, we allow the optimization routine to vary the lengths h_n of the finite elements such that all switching points coincide with grid points t_n. Additionally, the condition $\sum_{n=0}^{N_{\text{FE}}-1} h_n = T$ ensures that we regard a time interval of unaltered length. Consequently, we must ensure that active-set changes do not happen in the interior of a finite element. In this case, smooth functions are approximated by smooth polynomials within a finite element, cf. the right plot in Figure 7.1. A key assumption in any event-based method is that there are finitely many switches in finite time. We also assume that there are enough finite elements to capture every switch that occurs in the time interval $[0, T]$.

Having the step sizes as degrees of freedom in (7.4) leaves us with an underdetermined system of equations with possibly infinitely many solutions. We proceed with introducing additional conditions that lead to a well-defined system of equations and ensure that h_n takes correct values.

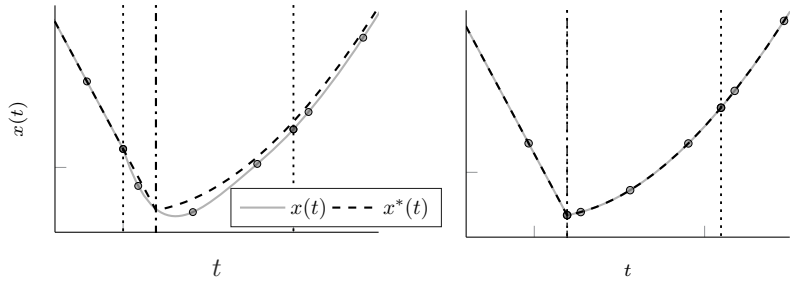


Figure 7.1: Illustration of the analytic solution and a polynomial solution approximation to a PSS via an IRK Radau IIA method of order 7. The left plot shows an approximation with a fixed step size where an active-set change happens on a stage point. The right plot shows an approximation obtained with FESD (based on the same IRK method) where the switch happens on the boundary. The circles represent the stage values, the vertical dotted lines the finite elements boundaries, and the vertical dashed line the switching time t_s.

7.2.3 Cross complementarity

So far, we have considered complementarity conditions only for every stage point:

$$0 \leq \theta_{n,m} \perp \lambda_{n,m} \geq 0, \text{ for all } n \in \{1, \dots, N_\mathrm{FE}\}, m \in \{1, \dots, n_\mathrm{s}\}. \tag{7.5}$$

For brevity, we often use the equivalent formulation via C-functions $\Phi(\theta_{n,m}, \lambda_{n,m}) = 0$. Next, we want to prohibit active-set changes on stage points inside a finite element. To achieve this, we define additional conditions on the variables $\theta_{n,m}$ and $\lambda_{n,m}$.

For ease of exposition, we assume that the underlying RK scheme satisfies $c_{n_\mathrm{s}} = 1$ (e.g., Radau and Lobatto methods [126]). This means that the right boundary point of a finite element is a stage point since $t_{n+1} = t_n + c_{n_\mathrm{s}} h_n$ for $c_{n_\mathrm{s}} = 1$. At the end of this section, we detail how to treat the case with $c_{n_\mathrm{s}} \neq 1$ (e.g., Gauss-Legendre methods).

Continuity of λ and μ

The additional conditions we introduce are motivated by the continuity of λ and μ, cf. Section 6.2.3. Thereby, the boundary values of an approximation of $\lambda(\cdot)$ and $\mu(\cdot)$ on an interval $[t_n, t_{n+1}]$ play a crucial role in FESD. We denote their values at t_n and t_{n+1} by $\lambda_{n,0}$, $\mu_{n,0}$ and λ_{n,n_s}, μ_{n,n_s}, respectively. We

impose continuity conditions for the discrete-time counterparts of λ and μ for all $n \in \{0, \dots, N_{\mathrm{FE}} - 1\}$:

$$\lambda_{n,n_\mathrm{s}} = \lambda_{n+1,0},\ \mu_{n,n_\mathrm{s}} = \mu_{n+1,0}. \tag{7.6}$$

In the sequel, we use only the right boundary points λ_{n,n_s} and μ_{n,n_s}, which are anyway degrees of freedom in the RK equations (7.4).

Remark 7.1. *Note that $\lambda_{0,0}$ and $\mu_{0,0}$ are not defined via Eq.* (7.6)*, as we do not have a preceding finite element in this case. However, they play an important role in the developments below, especially if the boundary point is the only stage point, as it is the case for the implicit Euler method. Fortunately, we can pre-compute $\lambda_{0,0}$ explicitly. Since x_0 is known, we obtain $\mu_{0,0} = \min_i g_i(x_0)$ and thus we have $\lambda_{0,0} = g(x_0) - \mu_{0,0}$.*

Moving the switching points to the boundary

The function $\lambda(\cdot)$ is a continuous function of time, and we consider an interval (t_n, t_{n+1}) with a fixed active set $\mathcal{I}_n$. In the interior of the this interval its components are either zero or positive on the whole interval. The stage values $\lambda_{n,i}$ of the discrete-time counterpart should satisfy this property as well. This is achieved by the *cross complementarity* conditions, which read for all $n \in \{0, \dots, N_{\mathrm{FE}} - 1\}$ as

$$0 = \mathrm{diag}(\theta_{n,m})\lambda_{n,m'},\ m \in \{1, \dots, n_\mathrm{s}\},\ m' \in \{0, \dots, n_\mathrm{s}\},\ m \neq m'. \tag{7.7}$$

In contrast to Eq. (7.5) here we have conditions between all stage variables within a finite element. We show below that the boundary point $\lambda_{n+1,0} = \lambda_{n,0}$ of the previous finite element plays a key role in the switch detection.

Some of the appealing properties of the constraints (7.7) are given by the next lemma. In our notation $\theta_{n,m,i}$ is the i-th component of the vector $\theta_{n,m}$.

Lemma 7.2. *Regard a fixed $n \in \{0, \dots, N_{\mathrm{FE}} - 1\}$ and a fixed $i \in \mathcal{J}$. If any $\theta_{n,m,i}$ with $m \in \{1, \dots, n_\mathrm{s}\}$ is positive, then all $\lambda_{n,m',i}$ with $m' \in \{0, \dots, n_\mathrm{s}\}$ must be zero. Conversely, if any $\lambda_{n,m',i}$ is positive, then all $\theta_{n,m,i}$ are zero.*

Proof. Let $\theta_{n,m,i}$ be positive, and suppose $\lambda_{n,j,i} = 0$ and $\lambda_{n,k,i} > 0$ for some $k, j \in \{0, \dots, n_\mathrm{s}\}, k \neq j$, then $\theta_{n,m,i}\lambda_{n,k,i} > 0$ which violates (7.7), thus all $\lambda_{n,m',i} = 0,\ m' \in \{0, \dots, n_\mathrm{s}\}$. The converse is proven in a similar way. □

The results of Lemma 7.2 are illustrated in Figure 7.2. Note that in contrast to the left plot illustrating the standard complementary conditions, in the right plot, all stage points inside a finite element have the same active set. Moreover, on the finite element boundary, we have $\lambda_{n,n_\mathrm{s},i} = 0$.

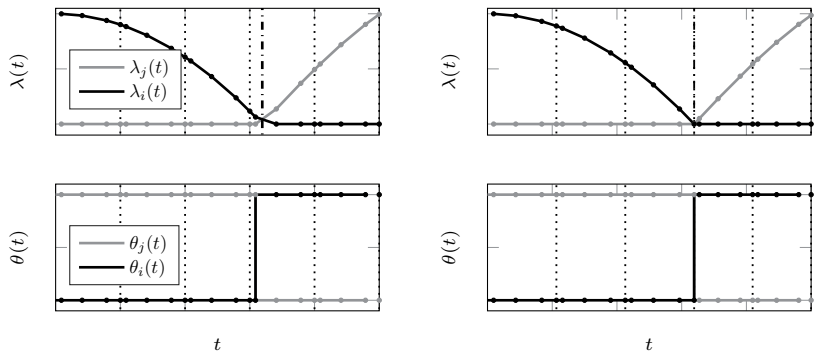


Figure 7.2: An illustration of the standard complementarity conditions $\Psi(\boldsymbol{\Theta}, \boldsymbol{\Lambda}) = 0$ (left plot) and the standard complementarity conditions augmented by $0 = G_{\text{cross}}(\boldsymbol{\Theta}, \boldsymbol{\Lambda})$ (right plot). The dots represent the stage values. The vertical dotted line marks the finite element boundaries, and the vertical dashed line marks the switching time t_{s}. In the standard case (left plot), an active-set change can happen at any complementarity pair. With the cross complementarities (7.9) (right plot), an active-set change can only happen on the boundaries of a finite element.

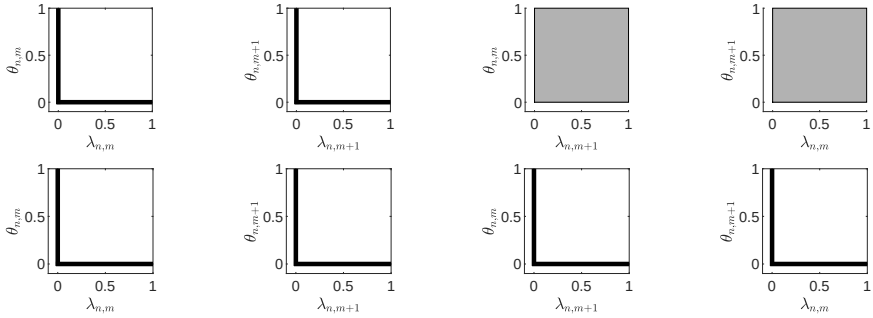


Figure 7.3: Illustration of several 2D projections of the feasible set for $(\lambda_{n,m}, \theta_{n,m}, \theta_{n,m+1}, \lambda_{n,m+1})$ for the standard complementarity conditions in (7.5) (top plot) and with adding the cross complementarities (7.9) (bottom plot).

In Figure 7.3, we illustrate several 2D projections of the feasible set for $\boldsymbol{\Theta}$ and $\boldsymbol{\Lambda}$ with and without the cross complementarity conditions. The top figure shows the feasible set for the standard complementarity conditions given in Eq. 7.5,

and the bottom shows the feasible set, which is tightened due to the cross complementarities in Eq. 7.7. This provides a geometric illustration of why the active set does not change within a finite element.

Implicit switch detection

It is left to explain how the switch detection for the solution approximation works. This is later formalized in Section 7.3. Note that for $x_n^{\text{next}} = x_{n+1}$ we have from the KKT conditions of the LP(x_{n+1}) (cf. Eq.(6.9)) that

$$\mu_{n,n_\text{s}} = \min_j g_j(x_{n+1}).$$

Moreover, if the active-set changes in the i-th component between the n-th and $n+1$-st finite element, then from Lemma 7.2 it follows that $\lambda_{n,n_\text{s},i} = 0$. Therefore, we obtain from (7.3) implicitly the condition

$$g_i(x_{n+1}) = \lambda_{n,n_\text{s},i} - \mu_{n,n_\text{s}},$$

which is equal to

$$0 = g_i(x_{n+1}) - g_j(x_{n+1}) = \psi_{i,j}(x_{n+1}), \tag{7.8}$$

where $\psi_{i,j}(x_{n+1}) = 0$ defines the switching surface between R_i and R_j. This condition forces h_n to adapt such that the switch is detected exactly. Note that the condition (7.8) appears only if active-set changes happen. Therefore, switch detection is entirely implicit.

Equivalent formulations

The conditions (7.7) are given in their sparsest form. Due to the non-negativity of Λ_n and Θ_n, there are many equivalent formulations of this condition, e.g., all conditions above can be summed up for a single finite element or even for all finite elements on the regarded control interval. Moreover, instead of the component-wise products in $\theta_{n,m}$ and $\lambda_{n,m'}$, we can use the inner products of these vectors. Thus, we use a more compact form of (7.7), where we combine the conditions for two neighboring finite elements. The motivation for this is that we end up, together with $\sum_{n=0}^{N_\text{FE}} h_n = T$, with the same number of new conditions as we have new degrees of freedom by varying h_n. The conditions

read as:

$$G_{\text{cross}}(\boldsymbol{\Theta}, \boldsymbol{\Lambda}) :=$$

$$\begin{bmatrix} \sum_{m=1}^{n_\text{s}} \sum_{\substack{m'=0, \\ m' \neq m}}^{n_\text{s}} \theta_{0,m}^\top \lambda_{0,m'} + \sum_{m=1}^{n_\text{s}} \sum_{\substack{m'=0, \\ m' \neq m}}^{n_\text{s}} \theta_{1,m}^\top \lambda_{1,m'} \\ \vdots \\ \sum_{m=1}^{n_\text{s}} \sum_{\substack{m'=0, \\ m' \neq m}}^{n_\text{s}} \theta_{N_\text{FE}-2,m}^\top \lambda_{N_\text{FE}-2,m'} + \sum_{m=1}^{n_\text{s}} \sum_{\substack{m'=0, \\ m' \neq m}}^{n_\text{s}} \theta_{N_\text{FE}-1,m}^\top \lambda_{N_\text{FE}-1,m'} \end{bmatrix}. \tag{7.9}$$

7.2.4 Step size equilibration

If no switches happen, i.e., the active sets $\mathcal{I}_n$ do not change between two neighboring finite elements, then the cross complementarity conditions in (7.9) are trivially satisfied. This yields spurious degrees of freedom in the step-sizes h_n, and the optimizer can adapt the grid in an undesirable way and harm the discretization accuracy. Also, the path-constraint discretization can be exploited unfavorably, just to decrease the objective value. To resolve this problem, we introduce *step equilibration* conditions.

The step size should only change if a switch occurs and otherwise be constant. Consequently, we obtain a piecewise uniform discretization grid for the differential and algebraic states on the regarded control interval. This decouples the integrator accuracy from the optimizer and results in piecewise equidistant discretization grids between the switches. To accomplish this, we derive an indicator function that is zero only if a switch occurs, otherwise, its value is strictly positive.

If some $\lambda_i(t_n)$ is equal to zero and its left or right time derivative is nonzero, then an active-set change has occurred. Instead of looking at the time derivatives, in the discrete-time case, we exploit the non-negativity of $\lambda_{n,m}$ and the fact that the active set is fixed for the whole finite element (due to cross complementarity, cf. Lemma 7.2). For $n \in \{1, \ldots, N_\text{FE} - 1\}$, we define the following backward and forward sums of the stage values over the finite elements $[t_{n-1}, t_n]$ and $[t_n, t_{n+1}]$:

$$\sigma_n^{\lambda,\text{B}} = \sum_{m=0}^{n_\text{s}} \lambda_{n-1,m}, \;\; \sigma_n^{\lambda,\text{F}} = \sum_{m=0}^{n_\text{s}} \lambda_{n,m}.$$

They are zero when the left and right time derivatives are zero, respectively. Likewise, they are positive when the left and right time derivatives are nonzero.

Analogously, the sums for $\theta_{n,m}$ are defined as:

$$\sigma_n^{\theta,\mathrm{B}} = \sum_{m=1}^{n_\mathrm{s}} \theta_{n-1,m},\ \sigma_n^{\theta,\mathrm{F}} = \sum_{m=1}^{n_\mathrm{s}} \theta_{n,m}.$$

Additionally, we define the following vectors for all $n \in \{1, \ldots, N_\mathrm{FE} - 1\}$:

$$\pi_n^\lambda = \mathrm{diag}(\sigma_n^{\lambda,\mathrm{B}})\sigma_n^{\lambda,\mathrm{F}},\ \pi_n^\theta = \mathrm{diag}(\sigma_n^{\theta,\mathrm{B}})\sigma_n^{\theta,\mathrm{F}}.$$

If there is an active-set change in the i-th complementarity pair, then at most one of the i-th components of $\sigma_n^{\lambda,\mathrm{B}}$ and $\sigma_n^{\lambda,\mathrm{F}}$ is nonzero, hence their product, i.e., the i-th component of π_n^λ, is zero. Due to complementarity, the same holds for π_n^θ. For sliding modes the corresponding components of π_n^λ are zero, and of π_n^θ, they are positive (due to complementarity). Thus, the i-th component of

$$\upsilon_n = \pi_n^\lambda + \pi_n^\theta,$$

is only zero if there is an active-set change in the i-th complementarity pair at t_n. A function that has the desired properties is defined as:

$$\eta_n(\boldsymbol{\Theta}, \boldsymbol{\Lambda}) := \prod_{i=1}^{n_f} (\upsilon_n)_i.$$

This scalar function summarizes the effects of all components. It is zero only if an active-set change happens at the boundary point t_n and is otherwise strictly positive. Finally, the constraints that remove possible spurious degrees of freedom in h_n read as:

$$0 = G_\mathrm{eq}(\mathbf{h}, \boldsymbol{\Theta}, \boldsymbol{\Lambda}, T) := \begin{bmatrix} (h_1 - h_0)\eta_1(\boldsymbol{\Theta},\boldsymbol{\Lambda}) \\ \vdots \\ (h_{N_\mathrm{FE}-1} - h_{N_\mathrm{FE}-2})\eta_{N_\mathrm{FE}-1}(\boldsymbol{\Theta},\boldsymbol{\Lambda}) \end{bmatrix}. \tag{7.10}$$

Since many products are involved in $\eta_n(\boldsymbol{\Theta}, \boldsymbol{\Lambda})$, one can replace it by $\tilde{\eta}_n(\boldsymbol{\Theta}, \boldsymbol{\Lambda}) := \tanh(\eta_n(\boldsymbol{\Theta}, \boldsymbol{\Lambda}))$ to have a better scaling. A numerical example illustrating the effect of the step equilibration is given in Subsection 7.5.2.

7.2.5 The FESD discretization

Now we have all ingredients to extend the standard RK discretization (7.4) to the FESD discretization. We use a discrete-time representation:

$$s_1 = F_\mathrm{fesd}(\mathbf{Z}), \tag{7.11a}$$

$$0 = G_{\text{fesd}}(\mathbf{Z}, \mathbf{h}, s_0, q, T), \tag{7.11b}$$

where $F_{\text{fesd}}(\mathbf{x}) = x_{N_{\text{FE}}}$ is the state transition map and $G_{\text{fesd}}(\mathbf{x}, \mathbf{h}, \mathbf{Z}, q, T)$ collects all other internal computations including all RK steps within the regarded control interval:

$$G_{\text{fesd}}(\mathbf{Z}, \mathbf{h}, s_0, q, T) := \begin{bmatrix} G_{\text{std}}(\mathbf{Z}, \mathbf{h}, s_0, q) \\ G_{\text{cross}}(\mathbf{\Theta}, \mathbf{\Lambda}) \\ G_{\text{eq}}(\mathbf{h}, \mathbf{\Theta}, \mathbf{\Lambda}) \\ \sum_{n=0}^{N_{\text{FE}}-1} h_n - T \end{bmatrix}.$$

For a fixed control function q, horizon length T and initial value s_0, the formulation (7.11) can be used as an integrator with exact switch detection for PSS (6.1). Since Filippov DIs do not always have unique solutions, one cannot expect uniqueness of solutions for their discrete-time counterparts (7.11) in all cases. In simulation methods, a common approach is to either pick one local solution obtained by the solver for the nonlinear complementarity problem (7.11) or to enumerate all possible solutions at an active-set change [6, 250]. In this thesis, we regard only the first option.

Note that in sliding modes, we implicitly obtain differential algebraic equations of index 2, cf. Section 6.2.2. To achieve good accuracy in practice, it is usually required to use stiffly accurate methods, e.g., Radau IIA methods [126].

Remark on RK methods with $c_{n_s} \neq 1$

We outline how to extend the FESD method when an RK scheme with $c_{n_s} \neq 1$ is regarded. In contrast to the developments so far, with $c_{n_s} \neq 1$ the variables λ_{n,n_s}, μ_{n,n_s} do not correspond the boundary values $\lambda(t_{n+1})$ and $\mu(t_{n+1})$ anymore, since $t_n + c_{n_s} h_n < t_{n+1}$. We denote the boundary points in this case by λ_{n,n_s+1}, μ_{n,n_s+1}. They are computed from the KKT conditions of $\text{LP}(x_{n+1})$ for $n = 0, \dots N_{\text{FE}} - 2$:

$$\begin{bmatrix} g(x_{n+1}) - \lambda_{n,n_s+1} - \mu_{n,n_s+1} e \\ 1 - e^\top \theta_{n,n_s+1} \\ \Psi(\theta_{n,n_s+1}, \lambda_{n,n_s+1}) \end{bmatrix} = 0. \tag{7.12}$$

Next, we replace (7.6) by the following continuity conditions for the discrete-time versions of λ and μ for $n = 0, \dots, N_{\text{FE}} - 1$:

$$\lambda_{n,n_s+1} = \lambda_{n+1,0}, \ \mu_{n,n_s+1} = \mu_{n+1,0}. \tag{7.13}$$

With slight abuse of notation, we add the new variables $\theta_{n,n_s+1}, \lambda_{n,n_s+1}$ and μ_{n,n_s+1} to the vectors $\mathbf{\Theta}$, $\mathbf{\Lambda}$ and $\mathbf{M}$, respectively. The vector $\mathbf{Z}$ is redefined

accordingly. The cross complementarity conditions (7.9) are now modified such that next to the stage points, we include the boundary points with the index $n_s + 1$:

$$G_{\text{cross}}(\boldsymbol{\Theta}, \boldsymbol{\Lambda}) :=$$

$$\begin{bmatrix} \sum_{m=1}^{n_s} \sum_{\substack{m'=0,\\ m' \neq m}}^{n_s+1} \theta_{0,m}^{\top} \lambda_{0,m'} + \sum_{m=1}^{n_s} \sum_{\substack{m'=0,\\ m' \neq m}}^{n_s+1} \theta_{1,m}^{\top} \lambda_{1,m'} \\ \vdots \\ \sum_{m=1}^{n_s} \sum_{\substack{m'=0,\\ m' \neq m}}^{n_s+1} \theta_{N_{\text{FE}}-2,m}^{\top} \lambda_{N_{\text{FE}}-2,m'} + \sum_{m=1}^{n_s} \sum_{\substack{m'=0,\\ m' \neq m}}^{n_s} \theta_{N_{\text{FE}}-1,m}^{\top} \lambda_{N_{\text{FE}}-1,m'} \end{bmatrix}.$$

For the whole control time, we have in total $(N_{\text{FE}} - 1)(2n_f + 1)$ new variables.

7.3 Convergence theory of FESD for Stewart's reformulation

In this section, we present the main convergence result of the FESD method. First, we prove that even though the FESD system (7.11) is always overdetermined it still has a locally isolated solution. Second, we show that the numerical solution approximation $\hat{x}_h(\cdot)$ generated by FESD converges to a solution $x(\cdot)$ in the sense of Definition 6.1, with the same order that the underlying RK method has for smooth ODEs. Additionally, we prove that the numerical sensitivities converge to their correct values with high accuracy. The theoretical results are illustrated with numerical examples.

7.3.1 Main assumptions

We start by introducing some notation and stating assumptions related to the FESD formulation (7.11), which are useful for our theoretical study in this section.

Assumption 7.3. *(Runge-Kutta method) A Butcher tableau with the entries $a_{i,j}, b_i$ and c_i, $i, j \in \{1, \ldots, n_s\}$ related to an n_s-stage Runge-Kutta (RK) method is used in the FESD* (7.11)*. Moreover, we assume that:*

(a) If the same RK method is applied to the differential algebraic equation (6.15) *on an interval $[t_a, t_b]$, it has a global accuracy of $O(h^p)$ for the differential states.*

(b) The RK equations applied to (6.15) *have a locally isolated solution for a sufficiently small* $h_n > 0$.

This assumption aims to consider a broad class of RK methods, and both assumptions are standard [126]. For an introduction to RK methods, cf. Section 3.2 and references therein.

Assumption 7.4. *(Solution existence) For given parameters* s_0, q *and* T*, there exists a solution to the FESD problem* (7.11)*, such that for all* $n \in \{0, \dots, N_{\mathrm{FE}} - 1\}$ *it holds that* $h_n \geq 0$.

This assumption means that there exists a meaningful solution and that we can compute it. If the FESD method is used in direct optimal control, non-negativity of the step sizes can easily be achieved by adding box constraints on h_n. This is the strongest assumption we make in our theoretical study. Ideally, one would prove the existence of solutions. Since the system is over-determined this cannot be done straightforwardly by applying standard existence results [89]. As we will show below, in practice numerical solvers have no trouble computing such solutions.

We state a technical assumption that ensures regularity of the FESD problem (7.11).

Assumption 7.5. *(Regularity) Given the complementarity pairs* $\Psi(\theta_{n,m}, \lambda_{n,m}) = 0$*, for all* $n \in \{0, \dots N_{\mathrm{FE}} - 1\}$ *there exists an* $m \in \{1, \dots, n_{\mathrm{s}}\}$ *and* $i \in \{1, \dots, n_f\}$*, such that the strict complementarity property holds, i.e.,* $\theta_{n,m,i} + \lambda_{n,m,i} > 0$*. Moreover, for the RK equations* (7.3) *it holds for all* $n \in \{0, \dots N_{\mathrm{FE}} - 1\}$*, that at least one entry of the vector* $\nabla_{h_n} G_{\mathrm{rk}}(x_{n+1}, Z_n, h_n, q)$ *is nonzero.*

Once all stage values are computed by solving (7.11), we can use some interpolation method to construct the solution approximation candidate in continuous time, cf. Assumption 7.3. For example, if we use collocation-based IRK methods continuous-time approximation $\hat{x}_n(t; h_n)$ on every finite element is easily obtained via Lagrange polynomials [126, 38]. We append the approximation for every finite element and write

$$\hat{x}_h(t) = \hat{x}_n(t; h_n) \text{ if } t \in [t_n, t_{n+1}], \tag{7.14}$$

where $h = \max_{n \in \{0, \dots N_{\mathrm{FE}} - 1\}} h_n$. Similarly, continuous-time representations can be found for the algebraic variables, and we denote them compactly by $\hat{\lambda}_h(t)$, $\hat{\theta}_h(t)$ and $\hat{\mu}_h(t)$. Similar to the definitions in Section 6.2.3, the fixed active set in this case is denoted by $\mathcal{I}(\hat{x}_h(t)) = \hat{\mathcal{I}}_n,\ t \in (\hat{t}_{\mathrm{s},n}, \hat{t}_{\mathrm{s},n+1})$ and the active set at switching point $\hat{t}_{\mathrm{s},n}$ by $\mathcal{I}(\hat{x}_h(\hat{t}_{\mathrm{s},n})) = \hat{\mathcal{I}}_n^0$.

7.3.2 Solutions of the FESD problem are locally isolated

In this subsection, we analyze some properties of solutions of the FESD problem (7.11). Again, for ease of exposition, we regard $c_{n_\mathrm{s}} = 1$ and give the extensions later. For the reader's convenience, we restate the problem but discard the trivial state transition map $s_1 = F_{\mathrm{fesd}}(\mathbf{Z}) = x_{N_{\mathrm{FE}}}$:

$$G_{\mathrm{fesd}}(\mathbf{Z}, \mathbf{h}, s_0, q, T) = \begin{bmatrix} G_{\mathrm{std}}(\mathbf{Z}, \mathbf{h}, s_0, q, T) \\ G_{\mathrm{cross}}(\boldsymbol{\Theta}, \boldsymbol{\Lambda}) \\ G_{\mathrm{eq}}(\mathbf{h}, \boldsymbol{\Theta}, \boldsymbol{\Lambda}) \\ \sum_{n=0}^{N_{\mathrm{FE}}-1} h_n - T \end{bmatrix} = 0. \tag{7.15}$$

For ease of reading, we summarize the definitions and dimensions of the most relevant quantities:

- Degrees of freedom: $\mathbf{Z} = (\mathbf{x}, \mathbf{V}, \boldsymbol{\Theta}, \boldsymbol{\Lambda}, \mathbf{M}) \in \mathbb{R}^{n_\mathbf{Z}}$, $\mathbf{h} \in \mathbb{R}^{N_{\mathrm{FE}}}$.
- Number of degrees of freedom: $n_\mathbf{Z} + N_{\mathrm{FE}}$, with $n_\mathbf{Z} = (N_{\mathrm{FE}} + 1)n_x + N_{\mathrm{FE}} n_\mathrm{s} n_x + 2n_\theta + n_\mu$.
- Dimension of $\boldsymbol{\Theta}$ and $\boldsymbol{\Lambda}$: $n_\theta = N_{\mathrm{FE}} n_\mathrm{s} n_f$.
- Dimension of $\mathbf{M}$: $n_\mu = N_{\mathrm{FE}} n_\mathrm{s}$,
- Parameters: $(s_0, q, T) \in \mathbb{R}^{n_x + n_u + 1}$,
- Standard RK equations: $G_{\mathrm{std}} : \mathbb{R}^{n_\mathbf{Z}} \times \mathbb{R}^{N_{\mathrm{FE}}} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R} \to \mathbb{R}^{n_\mathbf{Z}}$,
- Cross complementarity: $G_{\mathrm{cross}} : \mathbb{R}^{n_\theta} \times \mathbb{R}^{n_\theta} \to \mathbb{R}^{N_{\mathrm{FE}}-1}$,
- Step equilibration: $G_{\mathrm{eq}} : \mathbb{R}^{n_{N_{\mathrm{FE}}}} \times \mathbb{R}^{n_\theta} \times \mathbb{R}^{n_\theta} \to \mathbb{R}^{N_{\mathrm{FE}}-1}$ and
- FESD equations: $G_{\mathrm{fesd}} : \mathbb{R}^{n_\mathbf{Z}} \times \mathbb{R}^{N_{\mathrm{FE}}} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R} \to \mathbb{R}^{n_\mathbf{Z} + 2N_{\mathrm{FE}} - 1}$.

The vectors $s_0 \in \mathbb{R}^{n_x}$, $q \in \mathbb{R}^{n_u}$ and $T \in \mathbb{R}$ are given parameters, hence we have $n_\mathbf{Z} + N_{\mathrm{FE}}$ unknowns and $n_\mathbf{Z} + 2N_{\mathrm{FE}} - 1$ equations. Consequently, for $N_{\mathrm{FE}} > 1$, which we always assume in FESD, the system (7.15) is overdetermined. However, we show in the next theorem that for a given active set $N_{\mathrm{FE}} - 1$ equations in (7.15) are implicitly satisfied, and we always end up with a square system. As a consequence, Eq. (7.15) has under reasonable assumptions locally unique solutions. Nevertheless, since we do not know the active set a priori, we can also not know which equations are binding and which are implicitly satisfied and thus must regard all equations in (7.15).

Lemma 7.6 (Corollary 6.1 in [192]). *Let $A_1 \in \mathbb{R}^{k \times m}$ and $A_2 \in \mathbb{R}^{m \times q}$, then*

$$\operatorname{rank}(A_1) + \operatorname{rank}(A_2) - m \leq \operatorname{rank}(A_1 A_2) \leq \min(\operatorname{rank}(A_1), \operatorname{rank}(A_2)).$$

Theorem 7.7. *Suppose that Assumptions 7.3, 7.4 and 7.5 hold. Let s_0, q_0 and $T > 0$ be some fixed parameters such that $G_{\text{fesd}}(\mathbf{Z}^*, \mathbf{h}^*, s_0, q, T) = 0$. Let $P^* \subseteq \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}$ be the set of all parameters $(\hat{s}_0, \hat{q}, \hat{T})$ such that $\mathbf{Z} \in \mathbb{R}^{n_\mathbf{Z}}$, which is the solution of $G_{\text{fesd}}(\mathbf{Z}, \mathbf{h}, \hat{s}_0, \hat{q}, \hat{T}) = 0$, has the same active set as $\mathbf{Z}^*$. Additionally, suppose that $G_{\text{fesd}}(\cdot)$ is continuously differentiable in s_0, q and T for all $(s_0, q, T) \in P^*$. Then there exists a neighborhood $P \subseteq P^*$ of (s_0, q_0, T) such that there exist continuously differentiable single valued functions $\mathbf{Z}^* : P \to \mathbb{R}^{n_\mathbf{Z}}$ and $\mathbf{h}^* : P \to \mathbb{R}^{N_{\text{FE}}}$.*

Proof. We regard the active sets for every finite element $\hat{\mathcal{I}}_n$ for all $n \in \{0 \dots, N_{\text{FE}} - 1\}$ that correspond to the solution $(\mathbf{Z}^*, \mathbf{h}^*)$. First, we look closer at the equations $G_{\text{cross}}(\mathbf{\Theta}^*, \mathbf{\Lambda}^*) = 0$ and $G_{\text{eq}}(\mathbf{h}^*, \mathbf{\Theta}^*, \mathbf{\Lambda}^*) = 0$. If two neighboring finite elements have the same active set, i.e., $\hat{\mathcal{I}}_n = \hat{\mathcal{I}}_{n+1}$, then the $(n+1)$-th entry of $G_{\text{cross}}(\mathbf{\Theta}^*, \mathbf{\Lambda}^*)$ is implicitly satisfied due to the point-wise complementarity conditions $\Psi(\Theta_n, \Lambda_n) = 0$ and $\Psi(\Theta_{n+1}, \Lambda_{n+1}) = 0$. Moreover, by construction we have $\eta_{n+1}(\mathbf{\Theta}^*, \mathbf{\Lambda}^*) > 0$ and the $(n+1)$-th entry of $G_{\text{eq}}(\mathbf{h}^*, \mathbf{\Theta}^*, \mathbf{\Lambda}^*, T) = 0$ is binding, i.e., it implies $h^*_{n+1} = h^*_n$. On the other hand, if $\hat{\mathcal{I}}_n \neq \hat{\mathcal{I}}_{n+1}$, we have by construction that $\eta_{n+1}(\mathbf{\Theta}^*, \mathbf{\Lambda}^*) = 0$ and then $(n+1)$-th entry of $G_{\text{eq}}(\mathbf{h}^*, \mathbf{\Theta}^*, \mathbf{\Lambda}^*, T) = 0$ vanishes, i.e., is satisfied for any h^*_n and h^*_{n+1}. However, the $(n+1)$-th entry of $G_{\text{cross}}(\mathbf{\Theta}^*, \mathbf{\Lambda}^*) = 0$ is now binding, cf. Lemma 7.2.

We collect the binding n_1 cross complementarity conditions, with $0 \leq n_1 \leq N_{\text{FE}} - 1$, in the equation $G^*_{\text{cross}}(\mathbf{\Theta}^*, \mathbf{\Lambda}^*) = 0$, and the $N_{\text{FE}} - 1 - n_1$ implicitly satisfied into $G^{\text{res}}_{\text{cross}}(\mathbf{\Theta}^*, \mathbf{\Lambda}^*) = 0$. Likewise, we collect the binding n_2 step equilibration conditions, with $1 \leq n_2 \leq N_{\text{FE}} - 1$, in $G^*_{\text{eq}}(\mathbf{h}^*, \mathbf{\Theta}^*, \mathbf{\Lambda}^*) = 0$. The remaining $N_{\text{FE}} - 1 - n_2$ conditions are implicitly satisfied and are collected in $G^{\text{res}}_{\text{eq}}(\mathbf{h}^*, \mathbf{\Theta}^*, \mathbf{\Lambda}^*) = 0$. Note that $n_1 + n_2 = N_{\text{FE}} - 1$. We highlight that $\sum_{n=0}^{N_{\text{FE}}-1} h_n - T$ is always binding.

We can further simplify our system of equations by eliminating some degrees of freedom using $G^*_{\text{eq}}(\mathbf{h}^*, \mathbf{\Theta}^*, \mathbf{\Lambda}^*) = 0$. All components of this vector are of the form $\eta_n(h_n - h_{n+1})$ with $\eta_n > 0$. Therefore, we have n_2 equations of the form of $h_n = h_{n+1}$ and can remove n_2 degrees of freedom. Furthermore, we can express any $h_j = T - \sum_{i=0, i\neq j}^{N_{\text{FE}}-1} h_n$ and remove another degree of freedom. In total we removed $n_2 + 1$ degrees of freedom and can regard a reduced number of unknown step-sizes, which we denote by $\tilde{\mathbf{h}}^* \in \mathbb{R}^{n_1}, n_1 = N_{\text{FE}} - n_2 - 1$. With a slight abuse of notation, we redefine the standard RK equations accordingly and obtain $G_{\text{std}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T) = 0$ with $G_{\text{std}} : \mathbb{R}^{n_\mathbf{Z}} \times \mathbb{R}^{n_1} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R} \to \mathbb{R}^{n_\mathbf{Z}}$.

To summarize, for a fixed active set we can rewrite (7.15) in a reduced form as

$$G^*_{\text{fesd}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T) \coloneqq \begin{bmatrix} G_{\text{std}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T) \\ G^*_{\text{cross}}(\mathbf{\Theta}^*, \mathbf{\Lambda}^*) \end{bmatrix} = 0, \tag{7.16}$$

with $G^*_{\text{fesd}}(\mathbf{Z}^*, \mathbf{h}^*, s_0, q, T) \in \mathbb{R}^{n_\mathbf{z}+n_1}$. These conditions imply

$$G^{\text{res}}_{\text{fesd}}(\mathbf{h}^*, \boldsymbol{\Theta}^*, \boldsymbol{\Lambda}^*) := \begin{bmatrix} G^{\text{res}}_{\text{cross}}(\boldsymbol{\Theta}^*, \boldsymbol{\Lambda}^*) \\ G^{\text{res}}_{\text{eq}}(\mathbf{h}^*, \boldsymbol{\Theta}^*, \boldsymbol{\Lambda}^*) \end{bmatrix} = 0, \tag{7.17}$$

with $G^{\text{res}}_{\text{fesd}}(\mathbf{h}^*, \boldsymbol{\Theta}^*, \boldsymbol{\Lambda}^*) \in \mathbb{R}^{N_{\text{FE}}-1}$. Thus, for a given active set we can discard (7.17) and regard only the equivalent reduced problem (7.16), which is a square system of equations.

Next, we show that the Jacobian matrix $\nabla_{(\mathbf{Z},\tilde{\mathbf{h}})} G^*_{\text{fesd}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T)^\top$ has full rank. This enables us to apply the implicit function theorem (cf. [85, Theorem 1B.1]) and establish the result of this theorem. We take a closer look at the matrix:

$$\begin{aligned} &\nabla_{(\mathbf{Z},\tilde{\mathbf{h}})} G^*_{\text{fesd}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T)^\top \\ &= \begin{bmatrix} \nabla_\mathbf{Z} G_{\text{std}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T)^\top & \nabla_{\tilde{\mathbf{h}}} G_{\text{std}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T)^\top \\ \nabla_\mathbf{Z} G^*_{\text{cross}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T)^\top & \nabla_{\tilde{\mathbf{h}}} G^*_{\text{cross}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T)^\top \end{bmatrix}. \end{aligned}$$

Under Assumption 7.4, for a fixed active set and a fixed h^*_n the equation $G_{\text{std}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T) = 0$ boils down to the RK equations for the differential algebraic equation (6.15). Due to Assumption 7.3 the RK system $G_{\text{std}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T) = 0$ has a locally isolated solution. A necessary and sufficient condition for this property is the invertibility of the Jacobian $\nabla_\mathbf{Z} G_{\text{std}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T)^\top$ [85, Theorem 1B.8]. Thus, we have that $\operatorname{rank}(\nabla_\mathbf{Z} G^*_{\text{fesd}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T)^\top) = n_\mathbf{Z}$. Second, due to the block diagonal structure of $\nabla_{\tilde{\mathbf{h}}} G^*_{\text{std}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T)$ and Assumption 7.5 we can deduce that $\operatorname{rank}(\nabla_{\tilde{\mathbf{h}}} G^*_{\text{fesd}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T)^\top) = n_1$. Third, due to the nonegativity of $(\boldsymbol{\Theta}, \boldsymbol{\Lambda})$ and Assumption 7.5 by direct computation it can be verified that $\operatorname{rank}(\nabla_\mathbf{Z} G^*_{\text{cross}}(\boldsymbol{\Theta}, \boldsymbol{\Lambda})^\top) = n_1$ and $\nabla_{\tilde{\mathbf{h}}} G^*_{\text{cross}}(\boldsymbol{\Theta}, \boldsymbol{\Lambda})^\top = 0$.

We introduce more compact notation and summarize the results so far with:

- $M_1 = \nabla_\mathbf{Z} G_{\text{std}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T)^\top \in \mathbb{R}^{n_\mathbf{z} \times n_\mathbf{Z}}$ with $\operatorname{rank}(M_1) = n_\mathbf{Z}$
- $M_2 = \nabla_{\tilde{\mathbf{h}}} G_{\text{std}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T)^\top \in \mathbb{R}^{n_\mathbf{z} \times n_1}$ with $\operatorname{rank}(M_2) = n_1$ and
- $M_3 = \nabla_\mathbf{Z} G_{\text{cross}}(\boldsymbol{\Theta}, \boldsymbol{\Lambda})^\top \in \mathbb{R}^{n_1 \times n_\mathbf{z}}$ with $\operatorname{rank}(M_3) = n_1$.

To show that $\nabla_{(\mathbf{Z},\tilde{\mathbf{h}})} G^*_{\text{fesd}}(\mathbf{Z}^*, \tilde{\mathbf{h}}^*, s_0, q, T)^\top$ has a rank of $n_\mathbf{Z} + n_1$, we show that the linear system

$$\begin{bmatrix} M_1 & M_2 \\ M_3 & 0 \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} = 0,$$

with $v \in \mathbb{R}^{n_\mathbf{z}}$ and $w \in \mathbb{R}^{n_1}$ has zero as the only solution.

From the first line in this linear system, we have that $v = -M_1^{-1} M_2 w$. Since $n_\mathbf{z} > n_1$, from Lemma 7.6, we conclude that $\operatorname{rank}(M_1^{-1} M_2) = n_1$. Next, from the second part of our linear system, we have that $-M_3 M_1^{-1} M_2 w = 0$. Again, using Lemma 7.6, we conclude that $\operatorname{rank}(M_3 M_1^{-1} M_2) = n_1$. Hence, we have $w = 0$ and $v = 0$ to be the only solution of the regarded linear system. This completes the proof. □

Remark 7.8. *We note that one cannot apply more general forms of implicit function theorems for generalized and nonsmooth equations [85]. They usually require Lipschitz continuity of the solution map to reason about local uniqueness, but the solution map for FESD is not continuous in general, but only piecewise continuous.*

Example 7.9. *To illustrate the discontinuity of the solution map, we regard the example of* $\dot{x} \in 2 - \operatorname{sign}(x) + x^2$, *with* $N_{\mathrm{FE}} = 2$, $T = 0.2$ *and vary* $x_0 \in [-0.7, 0.1]$. *A solution approximation is obtained via FESD based on the Radau IIA method of order 3. Consider an initial value* x_0 *such that no switch occurs and a perturbed initial value* $x_0 + \epsilon$ *where a single switch occurs on the time interval of interest. Clearly, in the first case, we have an equidistant grid with* $h_0 = h_1$ *and in the second case* h_0 *jumps to* $\hat{t}_{\mathrm{s},1}$. *We conclude that* $h_0(x_0)$ *is not a Lipschitz function, see Figure 7.4 for an illustration.*

Extension for the case of $c_{n_\mathrm{s}} \neq 1$

In the case of $c_{n_\mathrm{s}} \neq 1$, we need to solve the additional LP (7.12) to obtain the boundary points. Note that if we have $g_i(x_{n+1}) = \min_{j \in \mathcal{J}} g_j(x_{n+1})$ for more than one i, the variables $\theta_{n,n_\mathrm{s}+1}$ are not unique and the $\mathrm{LP}(x_{n+1})$ has infinitely many solutions. However, these variables are neither used in the cross

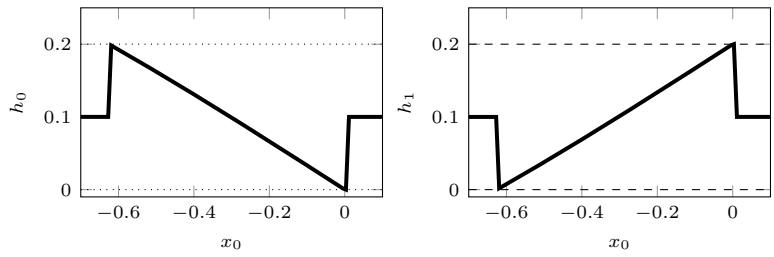


Figure 7.4: Illustration of the discontinuity of the solution map of (7.15) for the PSS $\dot{x} \in 2 - \operatorname{sign}(x) + x^2$ for $T = 0.2$ and $N_{\mathrm{FE}} = 2$.

complementarities nor step equilibration. Therefore, we can discard θ_{n,n_s+1} and simplify (7.12) to:

$$\lambda_{n,n_s+1} = g(x_{n+1}) - \mu_{n,n_s+1}e,$$

$$\lambda_{n,n_s+1} \geq 0,$$

which has $n_f + 1$ unknowns and n_f equalities and n_f inequalities for a given x_{n+1}. Now suppose that the first m_1 components of λ_{n,n_s+1} are zero (e.g., implied by cross complementarity) and the remaining m_2 are strictly positive, with $m_1 + m_2 = n_f$. We have that

$$g_i(x_{n+1}) = \mu_{n,n_s+1},\ i = 1, \ldots, m_1, \tag{7.18a}$$

$$\lambda_{n,n_s+1,i} = 0,\ i = 1, \ldots, m_1, \tag{7.18b}$$

$$\lambda_{n,n_s+1,j} = g_j(x_{n+1}) - \mu_{n,n_s+1},\ j = n_f - m_2 + 1, \ldots, n_f. \tag{7.18c}$$

As the first m_1 relations all assign the same value to μ_{n,n_s+1}, we can discard $m_1 - 1$ of them and thus we end up with a system of $m_1 + m_2 + 1 = n_f + 1$ equations and $n_f + 1$ unknowns. This system has still the important property that $\mu_{n,n_s+1} = \min_i g_i(x_{n+1})$. With this simplification for an RK method with $c_{n_s} \neq 1$ we have $n_\theta = N_{\mathrm{FE}} n_s n_f$, $n_\lambda = n_\theta + (N_{\mathrm{FE}} - 1)n_f$. The new variables are determined by the square linear system (7.18). Hence, it is straightforward to extend Theorem 7.7 for the case of $c_{n_s} \neq 1$.

7.3.3 Convergence and order of FESD

In this subsection, we prove that under reasonable assumptions the sequence of approximations $\hat{x}_h(\cdot)$ generated by the FESD method converges with high order to a solution of (6.1) in the sense of Definition 6.1. Recall that $h = \max_{n \in \{0, \ldots N_{\mathrm{FE}}-1\}} h_n$. The proof is inspired by the proof of Theorem 4.3 in [250]. We consider also $t_{s,0} = 0$ as a switching point since at this time point the active set for the first interval $(t_{s,0}, t_{s,1})$ is determined.

Note that for generating solution approximations with FESD it is sufficient to consider only two finite elements at a time, i.e., $N_{\mathrm{FE}} = 2$ in Eq. (7.15), and then to append the solutions to construct $\hat{x}_h(t),\ t \in [0, T]$ via Eq. (7.14). This requires of course to have only one switch in the regarded time interval, which can always be achieved with a sufficiently small h. We define the set of all discretization grid points as $\mathcal{G} = \{t_0, t_1, \ldots, t_{N_{\mathrm{FE}}}\}$.

Theorem 7.10. *Let $x(\cdot)$ be a solution of* (6.1) *in the sense of Definition 6.1 for $t \in [0, T]$ with $x(0) = x_0$. Suppose the following is true:*

(a) The assumptions 6.6 and 6.12 are satisfied.

(b) The assumption 7.3, 7.4 and 7.5 hold for the FESD problem (7.11).

Then $x(\cdot)$ is a limit point of the sequence of approximations $\hat{x}_h(\cdot)$, defined in Eq. (7.14) *as $h \downarrow 0$. Moreover, for sufficiently small $h > 0$, the solution of* (7.11) *generates a solution approximation $\hat{x}_h(t)$ on $[0, T]$ such that:*

$$|\hat{t}_{\mathrm{s},n} - t_{\mathrm{s},n}| = O(h^p) \text{ for every } n \in \{0, \dots, N_{\mathrm{sw}}\}, \tag{7.19a}$$

$$\|\hat{x}_h(t_n) - x(t_n)\| = O(h^p), \text{ for all } t_n \in \mathcal{G}. \tag{7.19b}$$

Proof. The proof will be carried out by induction, where we regard the switching events with index $n \in \{0, \dots, N_{\mathrm{sw}}\}$ and the corresponding time intervals $(t_{\mathrm{s},n}, t_{\mathrm{s},n+1})$, with a slight abuse of notation where $t_{\mathrm{s},N_{\mathrm{sw}}+1} = T$ is not necessarily a switching point.

Regard $n = 0$, where we have trivially that $t_{\mathrm{s},0} = 0$, thus

$$|\hat{t}_{\mathrm{s},0} - t_{\mathrm{s},0}| = 0 = O(h^p), \ \|\hat{x}_h(0) - x(0)\| = 0 = O(h^p).$$

Moreover, $\mathcal{I}(x_0) = \mathcal{I}(\hat{x}_h(0)) = \mathcal{I}_0^0$.

Now we suppose (7.19) is true for n, i.e., at $t = t_{\mathrm{s},n}$. We show that the same statements are true for $n + 1$. By the induction hypothesis and due to continuity of $g_i(x), i \in \mathcal{J}$, we have that for sufficiently small h the equality $\mathcal{I}(\hat{x}(\hat{t}_{\mathrm{s},n})) = \mathcal{I}(x(t_{\mathrm{s},n})) = \mathcal{I}_n^0$ holds. Moreover, by Lipschitz continuity of $f_i(x)$ and $\nabla g_i(x), i \in \mathcal{J}$, it follows that (cf. Section 6.2.4)

$$M_{\mathcal{I}_n^0}(\hat{x}_h(\hat{t}_{\mathrm{s},n})) \to M_{\mathcal{I}_n^0}(x(t_{\mathrm{s},n})) \text{ as } h \downarrow 0.$$

According to Theorem 6.11 the solution of the LCP (6.19) corresponding to $M_{\mathcal{I}_n^0}(x(t_{\mathrm{s},n}))$ determines the new index set $\mathcal{I}_n = \{i \in \mathcal{I}_n^0 \mid \tilde{\theta}_i > 0\}$. Moreover, by Assumption 6.12 this LCP is strongly stable and due to Lemma 6.15, for sufficiently small $h > 0$ the solution of the LCP corresponding $M_{\mathcal{I}_n^0}(\hat{x}_h(\hat{t}_{\mathrm{s},n}))$ has a solution such that $\hat{\mathcal{I}}_n = \{i \in \mathcal{I}_n^0 \mid \tilde{\theta}_i > 0\} = \{i \in \mathcal{I}_n^0 \mid \theta_i > 0\} = \mathcal{I}_n$. Thus, we conclude that both the solution approximation and the solution *predicted* the same active set $\mathcal{I}_n$ in a neighborhood of $\hat{t}_{\mathrm{s},n}$ and $t_{\mathrm{s},n}$, respectively.

It is left to verify that such an active set $\mathcal{I}_n$ predicted by the solution approximation is indeed feasible for the FESD problem. Note that by the induction hypothesis and the reasoning above the solution approximation and $x(\cdot)$ have the same corresponding active set in a neighborhood of $t_{\mathrm{s},n}$. For a fixed active set, as a consequence of Proposition 6.7 the arising DAE (6.15) has a unique solution. Finally, under this setting with the given active sets

in a neighborhood of $t_{\mathrm{s},n}$, according to Theorem 7.7 there is a locally unique solution to a FESD problem, thus we can construct an appropriate $\hat{x}_h(\cdot)$.

Note that one can make arbitrarily many integration steps with a fixed $\mathcal{I}_n$ before the next switch in time is reached. Again, due to Theorem 7.7, the corresponding FESD problem has a locally unique solution.

Now we provide an error estimate for the solution approximation until the next switching point. First, we define $\tilde{x}(t)$ to be the exact *extended* solution of the DAE (6.15) with the fixed active set $\mathcal{I}_n$ on the interval $t \in [t_{\mathrm{s},n}, \tilde{t}]$, with $\tilde{x}(t_{\mathrm{s},n}) = x(t_{\mathrm{s},n})$ and $\tilde{t} > t_{\mathrm{s},n+1}$. Obviously, it holds that $x(t) = \tilde{x}(t)$ for all $t \in [t_{\mathrm{s},n}, t_{\mathrm{s},n+1}]$. Second, from the discussions in Section 7.2.3 we know that active-set changes can only happen at boundaries of the finite elements, thus it holds that $\hat{t}_{\mathrm{s},n} \in \mathcal{G}$ for all $n \in \{0, \ldots, \hat{N}_{\mathrm{sw}}\}$. Third, by the induction hypothesis we have $\|\hat{x}_h(\hat{t}_{\mathrm{s},n}) - x(\hat{t}_{\mathrm{s},n})\| = O(h^p)$. As noted above, for a fixed active set $\mathcal{I}_n$ and fixed h_n the FESD equations boil down to standard RK equations applied to (6.15). Thus, from Assumption 7.3 we have the estimate

$$\|\hat{x}_h(\hat{t}_{\mathrm{s},n+1}) - \tilde{x}(\hat{t}_{\mathrm{s},n+1})\| = O(h^p). \tag{7.20}$$

With the help of this estimate, in the next few steps we prove that $|\hat{t}_{\mathrm{s},n+1} - t_{\mathrm{s},n+1}| = O(h^p)$. We distinguish two cases (see also Table 6.2):

I) crossing a discontinuity or entering a sliding mode, i.e., $\mathcal{I}_n \subset \mathcal{I}(x(t_{\mathrm{s},n+1}))$ and $\mathcal{I}_{n+1} \subseteq \mathcal{I}(x(t_{\mathrm{s},n+1}))$.

II) a spontaneous switch or laving sliding mode, i.e., $\mathcal{I}_n = \mathcal{I}(x(t_{\mathrm{s},n+1}))$ and $\mathcal{I}_{n+1} \subset \mathcal{I}(x(t_{\mathrm{s},n+1}))$

Within Case I we need to distinguish the two scenarios: I.a. $\hat{t}_{\mathrm{s},n+1} > t_{\mathrm{s},n+1}$ and I.b. $\hat{t}_{\mathrm{s},n+1} \leq t_{\mathrm{s},n+1}$.

Case I.a. Regard the following indices $j \in \mathcal{I}_n$ and $i \in \mathcal{I}(x(t_{\mathrm{s},n+1})) \setminus \mathcal{I}_n$. This means that $\min_k g_k(x((t_{\mathrm{s},n+1})) = g_i(x(t_{\mathrm{s},n+1})) = g_j(x(t_{\mathrm{s},n+1})) = \mu(t_{\mathrm{s},n+1})$ holds and one can locally regard the following *switching* function

$$\psi_{i,j}(x(t)) = g_i(x(t)) - g_j(x(t)) = \lambda_i(t) - \lambda_j(t).$$

Note that this function is Lipschitz continuous. It must by definition become zero when an active-set change happens.

Due to the strict complementarity assumed in Assumption 6.12 (see also part 9 of the proof of [250, Theorem 4.3], and the remarks after Assumption 6.12) we have at $t^-_{\mathrm{s},n+1}$ that $\dot{\lambda}_j(t^-_{\mathrm{s},n+1}) = 0$ and $\dot{\lambda}_i(t^-_{\mathrm{s},n+1}) < 0$. Therefore, it holds that:

$$\psi_{i,j}(x(t_{\mathrm{s},n+1})) = 0, \ \frac{\mathrm{d}}{\mathrm{d}t}\psi_{i,j}(x(t^-_{\mathrm{s},n+1})) < 0. \tag{7.21}$$

Obviously, the same assertion holds for $\tilde{x}(t)$. Moreover, due to the smoothness of $\tilde{x}(t)$, we have $\psi_{i,j}(\tilde{x}(t)) < 0$ for $t \in (t_{\mathrm{s},n+1}, t_{\mathrm{s},n+1} + \epsilon)$ for some $\epsilon > 0$.

Similarly, for the solution approximation we have $\psi_{i,j}(\hat{x}_h(t)) = g_i(\hat{x}_h(t)) - g_j(\hat{x}_h(t))$. Since $\hat{t}_{\mathrm{s},n+1} > t_{\mathrm{s},n+1}$, due to continuity of $\psi_{i,j}(\cdot)$ and $\hat{x}_h(\cdot)$ it follows that

$$\psi_{i,j}(\hat{x}_h(t_{\mathrm{s},n+1})) > 0 \text{ and } \frac{\mathrm{d}}{\mathrm{d}t}\psi_{i,j}(\hat{x}_h(t_{\mathrm{s},n+1})) < 0.$$

Now from Lipschitz continuity of $\psi_{i,j}(\cdot)$ and (7.20) we can establish that

$$\begin{gathered} |\underbrace{\psi_{i,j}(\hat{x}_h(\hat{t}_{\mathrm{s},n+1}))}_{=0} - \underbrace{\psi_{i,j}(\tilde{x}(\hat{t}_{\mathrm{s},n+1}))}_{<0}| \leq L_\psi \|\hat{x}_h(\hat{t}_{\mathrm{s},n+1}) - \tilde{x}(\hat{t}_{\mathrm{s},n+1})\|, \\ |\psi_{i,j}(\tilde{x}(\hat{t}_{\mathrm{s},n+1}))| = O(h^p). \end{gathered} \tag{7.22}$$

Note that in contrast to $\psi_{i,j}(x(t))$, the function $\psi_{i,j}(\tilde{x}(t))$ is smooth in a neighborhood of $t_{\mathrm{s},n+1}$. Thus, we regard the first-order Taylor approximation of $\psi_{i,j}(\cdot)$ at $\tilde{x}(t_{\mathrm{s},n+1})$.

$$\psi_{i,j}(\tilde{x}(t)) = \psi_{i,j}(\tilde{x}(t_{\mathrm{s},n+1})) + \frac{\mathrm{d}}{\mathrm{d}t}\psi_{i,j}(\tilde{x}(t_{\mathrm{s},n+1}))(t - t_{\mathrm{s},n+1}) + o(|t - t_{\mathrm{s},n+1}|).$$

Since $\psi_{i,j}(\tilde{x}(t_{\mathrm{s},n+1}))$ is decreasing, there exists some positive constant C_a with $C_a < |\frac{\mathrm{d}}{\mathrm{d}t}\psi_{i,j}(\tilde{x}(t_{\mathrm{s},n+1})|$ such that for sufficiently small h and $t \in [t_{\mathrm{s},n+1}, \hat{t}_{\mathrm{s},n+1}]$ it holds that

$$\psi_{i,j}(\tilde{x}(t)) \leq \psi_{i,j}(\tilde{x}(t_{\mathrm{s},n+1})) - C_a(t - t_{\mathrm{s},n+1}).$$

The arguments above are illustrated in the left plot of Figure 7.5. From the last inequality and (7.22) at $t = \hat{t}_{\mathrm{s},n+1}$ we have that $\psi_{i,j}(\tilde{x}(\hat{t}_{\mathrm{s},n+1})) < 0$, $\psi_{i,j}(\tilde{x}(t_{\mathrm{s},n+1})) = 0$ and conclude that

$$\hat{t}_{\mathrm{s},n+1} - t_{\mathrm{s},n+1} \leq O(h^p). \tag{7.23}$$

This completes the consideration of case I.a.

Case I.b. We apply similar arguments as in I.a. Under the assumption of $\hat{t}_{\mathrm{s},n+1} < t_{\mathrm{s},n+1}$, following similar lines as in the proof of [250, Theorem 4.3], we first prove $\hat{t}_{\mathrm{s},n+1} \to t_{\mathrm{s},n+1}$ and establish subsequently the convergence rate.

Regard the set $H = \{h > 0 \mid \hat{t}_{\mathrm{s},n+1} < t_{\mathrm{s},n+1}\}$. By the assumption of case I.b and the induction hypothesis, it holds that $\hat{t}_{\mathrm{s},n+1} \in [\hat{t}_{\mathrm{s},n}, t_{\mathrm{s},n+1}]$. Since this is a bounded set, there must be a subsequence $H' \subseteq H$ with $h \downarrow 0$ such that

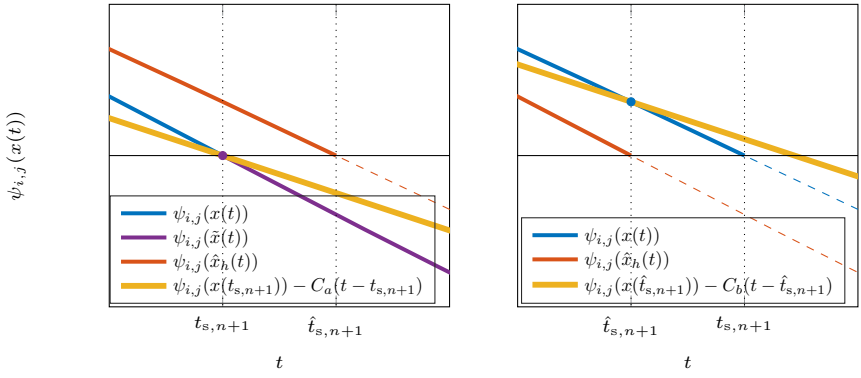


Figure 7.5: The left plot shows an illustration of the argument of Case I.a: $\hat{t}_{s,n+1} > t_{s,n+1}$ and the right plot shows an illustration of the argument of Case I.b: $\hat{t}_{s,n+1} \leq t_{n+1}$, for establishing $|\hat{t}_{s,n+1} - t_{s,n+1}| = O(h^p)$.

$\hat{t}_{s,n+1} \to \bar{t}$. We show now that $\bar{t} = \hat{t}_{s,n+1}$. We regard again the function $\psi_{i,j}(\cdot)$ for some $j \in \mathcal{I}_n$ and $i \in \mathcal{I}(x(t_{s,n+1})) \setminus \mathcal{I}_n$

$$\psi_{i,j}(x(t)) = g_i(x(t)) - g_j(x(t)),$$

which becomes zero at an active-set change and is positive otherwise. Similarly, active-set changes for the solution approximation happen when $\psi_{i,j}(\hat{x}_h(\hat{t}_{s,n+1})) = 0$. We remind the reader that earlier it was shown that $\hat{\mathcal{I}}_n = \mathcal{I}_n$.

By taking $h \downarrow 0$ and $h \in H'$ from (7.20) it follows that $\psi_{i,j}(x(\bar{t})) = 0$. By the definition of a switching point, there must be a $i \notin \mathcal{I}_n$, but $i \in \mathcal{I}(x(\bar{t}))$. However, this contradicts the assumption that $\mathcal{I}(x(t)) = \mathcal{I}_n$ for $t \in (t_{s,n}, t_{s,n+1})$ and we conclude that $\bar{t} \notin (t_{s,n}, t_{s,n+1})$.

On the other hand at $t_{s,n}$, due to strict complementarity in the active-set determining LCP (cf. Theorem6.11 and Assumption 6.12), if some $i \in \mathcal{I}(x(t_{s,n})) \setminus \mathcal{I}_n$ and $j \in \mathcal{I}_n$, we know that

$$\frac{\mathrm{d}}{\mathrm{d}t}\psi_{i,j}(x(t^+_{s,n})) = g_i(x(t^+_{s,n})) - g_j(x(t^+_{s,n})) > 0.$$

Due to continuity of the functions $g_i(\cdot), i \in \mathcal{J}$, and the induction hypothesis, there exists some $\epsilon > 0$ such that

$$\frac{\mathrm{d}}{\mathrm{d}t}(g_i(\hat{x}_h(t)) - g_j(\hat{x}_h(t)) > 0 \text{ for } t \in [\hat{t}_{s,n}, \hat{t}_{s,n} + \epsilon].$$

However, when a switch happens the derivative in the last line must be negative at t (cf. Remark 6.10), thus $\hat{t}_{\mathrm{s},n+1} > \hat{t}_{\mathrm{s},n} + \epsilon$, i.e., with $h \downarrow 0, h \in H'$, $\bar{t} > t_{\mathrm{s},n} + \epsilon$. This means that $\bar{t} \neq t_{\mathrm{s},n}$ and the only option that is left is $\hat{t}_{\mathrm{s},n+1} \to \bar{t} = \hat{t}_{\mathrm{s},n+1}$ as $h \downarrow 0, h \in H'$.

Now we continue with establishing the convergence rate for $\hat{t}_{\mathrm{s},n+1} \to t_{\mathrm{s},n+1}$. From $\hat{t}_{\mathrm{s},n+1} \leq t_{\mathrm{s},n+1}$ we have from the definition of $\psi_{i,j}(\cdot)$ that $\psi_{i,j}(x(\hat{t}_{\mathrm{s},n+1})) > 0$ and $\psi_{i,j}(\hat{x}_h(\hat{t}_{\mathrm{s},n+1})) = 0$. Using the fact that $\tilde{x}(\hat{t}_{\mathrm{s},n+1}) = x(\hat{t}_{\mathrm{s},n+1})$ and (7.20) we have

$$\begin{aligned}|\psi_{i,j}(x(\hat{t}_{\mathrm{s},n+1}))| &= |\psi_{i,j}(x(\hat{t}_{\mathrm{s},n+1})) - \psi_{i,j}(\hat{x}_h(\hat{t}_{\mathrm{s},n+1}))| \\ &\leq L_\psi \|\tilde{x}(\hat{t}_{\mathrm{s},n+1}) - \hat{x}_h(\hat{t}_{\mathrm{s},n+1})\| = O(h^p).\end{aligned}$$

We again use a first-order expansion (an illustration of the argument is given in the right plot of Figure 7.5):

$$\psi_{i,j}(x(t)) = \psi_{i,j}(x(\hat{t}_{\mathrm{s},n+1})) + \frac{\mathrm{d}}{\mathrm{d}t}\psi_{i,j}(x(\hat{t}_{\mathrm{s},n+1}))(t - \hat{t}_{\mathrm{s},n+1}) + o(|t - \hat{t}_{\mathrm{s},n+1}|).$$

Once again, due to assumption 6.12, we have that $\frac{\mathrm{d}}{\mathrm{d}t}\psi_{i,j}(x(t_{\mathrm{s},n+1})) < 0$. Note that $\frac{\mathrm{d}}{\mathrm{d}t}\psi_{i,j}(x(t^-_{\mathrm{s},n+1})) < 0$. From $\hat{t}_{\mathrm{s},n+1} \to t_{\mathrm{s},n+1}$ and continuity of $\psi_{i,j}(\cdot)$ it follows that for sufficiently small $h > 0$:

$$\frac{\mathrm{d}}{\mathrm{d}t}\psi_{i,j}(x(\hat{t}_{\mathrm{s},n+1})) < 0.$$

Using similar reasoning as in case I.a., there exists some $C_b > 0$ such that from the last equation at $t = t_{\mathrm{s},n+1}$ it follows $0 \leq O(h^p) - C_b(t_{\mathrm{s},n+1} - \hat{t}_{\mathrm{s},n+1})$, i.e.,

$$t_{\mathrm{s},n+1} - \hat{t}_{\mathrm{s},n+1} \leq O(h^p). \tag{7.24}$$

Putting (7.23) and (7.24) together, we obtain the first part of the induction statement, i.e.,

$$|t_{\mathrm{s},n+1} - \hat{t}_{\mathrm{s},n+1}| = O(h^p). \tag{7.25}$$

Case II.. Next, we consider the case of $\mathcal{I}_n = \mathcal{I}^0_{n+1}$, which may happen under the assumptions of this theorem when $x(\cdot)$ is not unique, as discussed below. This part of the proof follows closely (with appropriate modifications) the lines of part 8 of the proof of [250][Theorem 4.3].

By the induction hypothesis we have $\mathcal{I}^0_n = \hat{\mathcal{I}}^0_n$ and it was shown above that $\mathcal{I}_n = \hat{\mathcal{I}}_n$. By Assumption 6.12 the LCP$(M_{\mathcal{I}_n,\alpha}(x(t_{\mathrm{s},n})), -e)$ is strongly stable. Let $(\tilde{w}^*, \tilde{\theta}^*) \in \mathrm{SOL}(M_{\mathcal{I}_n,\alpha}(x(t_{\mathrm{s},n})), -e)$ and due to Assumption 6.12 it holds

that $\tilde{w}^* + \tilde{\theta}^* > 0$. From Theorem 6.11 we have $\mathcal{I}_n = \{i \mid \tilde{\theta}_i^* > 0\}$. Due to the induction hypothesis, Lipschitz continuity and (7.20) we have $M_{\mathcal{I}_n,\alpha}(\hat{x}_h(\hat{t}_{\mathrm{s},n})) \to M_{\mathcal{I}_n,\alpha}(x(t_{\mathrm{s},n}))$ as $h \downarrow 0$. All assumptions of Lemma 6.14 are satisfied and it follows that the corresponding LCPs that predict the new active set (cf. Theorem 6.11) have the same number of solutions, i.e.,

$$|\mathrm{SOL}(M_{\mathcal{I}_n,\alpha}(x(t_{\mathrm{s},n})), -e)| = |\mathrm{SOL}(M_{\mathcal{I}_n,\alpha}(\hat{x}_h(\hat{t}_{\mathrm{s},n})), -e)| = N_{\mathrm{sol}}.$$

Since Assumption 6.12 holds, one can apply Lemma 6.16 to the $\mathrm{LCP}(M_{\mathcal{I}_n,\alpha}(x(t_{\mathrm{s},n})), -e)$ and conclude that for all $t \in [t_{\mathrm{s},n}, t_{\mathrm{s},n+1}]$ it holds that

$$|\mathrm{SOL}(M_{\mathcal{I}_n,\alpha}(x(t)), -e)| = N_{\mathrm{sol}}.$$

Since $\mathcal{I}_n = \mathcal{I}_{n+1}^0$ at $t = t_{\mathrm{s},n+1}$ we have

$$|\mathrm{SOL}(M_{\mathcal{I}_{n+1}^0,\alpha}(x(t_{\mathrm{s},n+1})), -e)| = N_{\mathrm{sol}}.$$

Recall that other possible switching cases where $\mathcal{I}_n \neq \mathcal{I}_{n+1}^0$ are covered in Case I. We must now distinguish two cases: either $\mathrm{SOL}(M_{\mathcal{I}_n,\alpha}(x(t_{\mathrm{s},n})), -e)$ is a singleton or not. It is certainly not an empty set as all entries of the matrix $M_{\mathcal{I}_n,\alpha}$ are positive for a sufficiency large $\alpha > 0$ and thus this matrix is strictly copositive. This is a sufficient condition for the set $\mathrm{SOL}(M_{\mathcal{I}_n,\alpha}(x(t_{\mathrm{s},n}), -e))$ to be nonempty [250, Theorem 3.6].

In case $\mathrm{LCP}(M_{\mathcal{I}_n,\alpha}(x(t_{\mathrm{s},n})), -e)$ has a unique solution $(\tilde{w}^*(t_{\mathrm{s},n}), \tilde{\theta}^*(t_{\mathrm{s},n}))$ with $\mathcal{I}_n = \{i \mid \tilde{\theta}^*(t_{\mathrm{s},n}) > 0\}$, it can be deduced from Lemma 6.16 that the LCP given by $\mathrm{LCP}(M_{\mathcal{I}_{n+1}^0,\alpha}(x(t_{\mathrm{s},n+1})), -e)$ has as well a unique solution with $\mathcal{I}_{n+1} = \{i \mid \tilde{\theta}^*(t_{\mathrm{s},n+1}) > 0\} = \mathcal{I}_n$. This contradicts the definition of a switching point, cf. Definition 6.1. Hence, we conclude that $|\mathrm{SOL}(M_{\mathcal{I}_{n+1}^0,\alpha}(x(t)), -e)| > 1$ for all $t \in [t_{\mathrm{s},n}, t_{\mathrm{s},n+1}]$.

Note that, with a slight abuse from notation, it follows from the reasoning above that any $t^* \in [t_{\mathrm{s},n}, t_{\mathrm{s},n+1}]$ can be a switching point. Therefore, whenever the active-set change in the FESD problem occurs at $\hat{t}_{\mathrm{s},n+1}$, we can simply choose $t^* = \hat{t}_{\mathrm{s},n+1}$. Hence, we obtain $|t_{\mathrm{s},n+1} - \hat{t}_{\mathrm{s},n+1}| = 0 = O(h^p)$ and this completes the considerations of Case II.

To complete the induction step it is left to prove that (7.19b) holds for $t = \hat{t}_{\mathrm{s},n+1}$. For $\hat{t}_{\mathrm{s},n+1} \leq t_{\mathrm{s},n+1}$ we have $\tilde{x}(\hat{t}_{\mathrm{s},n+1}) = x(\hat{t}_{\mathrm{s},n+1})$ and the assertion follows directly from (7.20). Note that for any other $t_n \in \mathcal{G}$ that is not a switching point (7.19b) follows immediately from Assumption 7.3.

It is left to investigate the case of $\hat{t}_{\mathrm{s},n+1} > t_{\mathrm{s},n+1}$. Using Lipschitz continuity of $x(\cdot)$, $\tilde{x}(\cdot)$, the fact that $\tilde{x}(t_{\mathrm{s},n+1}) = x(t_{\mathrm{s},n+1})$ and (7.25) one obtains

$$\|\hat{x}_h(\hat{t}_{\mathrm{s},n+1}) - x(\hat{t}_{\mathrm{s},n+1})\| \leq \|\hat{x}_h(\hat{t}_{\mathrm{s},n+1}) - \tilde{x}(\hat{t}_{\mathrm{s},n+1})\| + \|x(t_{\mathrm{s},n+1}) - x(\hat{t}_{\mathrm{s},n+1})\|$$

$$+ \|\tilde{x}(\hat{t}_{\mathrm{s},n+1}) - \tilde{x}(t_{\mathrm{s},n+1})\| \leq O(h^p) + (L_x + L_{\tilde{x}})|t_{\mathrm{s},n+1} - \hat{t}_{\mathrm{s},n+1}| = O(h^p).$$

Moreover, from Lipschitz continuity of $g_i(\cdot), i \in \mathcal{J}$ and the last inequality for sufficiently small $h > 0$ we have that $\mathcal{I}(x(t_{\mathrm{s},n+1})) = \mathcal{I}(\hat{x}(\hat{t}_{\mathrm{s},n+1}))$, which completes the induction step for $n+1$. With the use of an interpolation scheme, from (7.19) it follows that we can make a continuous-time approximation $\hat{x}_h(t)$ for $t \in [0, T]$ with the accuracy $O(h^{\bar{p}})$, $1 \leq \bar{p} \leq p$ for $t \notin \mathcal{G}$. Therefore, it follows that the sequence of approximations $\hat{x}_h(t)$ generated by the FESD method converges to a solution $x(t)$ in the sense of Definition 6.1 for $t \in [0, T]$. The proof is completed. □

In the proof of Theorem 7.10, we establish that both the solution approximation $\hat{x}_h(t)$ and exact solution $x(t)$ predict the same new active set at switching points. Thereby, we make use of Theorem 6.11, which in turn is applicable if Assumption 6.12 holds. As discussed in Section 6.2.3, this assumption does not hold for the cases of a unique leaving of a sliding mode as in case (d) of Example 6.8. Consequently, we cannot apply Theorem 6.11 to predict $\mathcal{I}_n$ with the data at $t_{\mathrm{s},n}$. The solution is to construct an LCP based on higher order derivatives of λ, which enables one to predict the new active set in case of leaving sliding mode, cf. Remark 6.10. Such an extension is beyond the scope of this thesis, but we refer the interested reader to [251, Section 4.2].

7.3.4 Illustrating the integration order

In section 4.2.5, we investigated the order of accuracy of standard integration methods on the example in Eq. (4.4). Now we illustrate the results of Theorem 7.10 for several RK schemes on the same example, which we restate for the reader's convenience:

$$\dot{x} = \begin{cases} A_1 x, & \|x\|_2^2 < 1, \\ A_2 x, & \|x\|_2^2 > 1, \end{cases} \tag{7.26}$$

with

$$A_1 = \begin{bmatrix} 1 & 2\pi \\ -2\pi & 1 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 1 & -2\pi \\ 2\pi & 1 \end{bmatrix},$$

and $x(0) = (e^{-1}, 0)$. Figure 4.8 illustrates the solution of this IVP. We compare

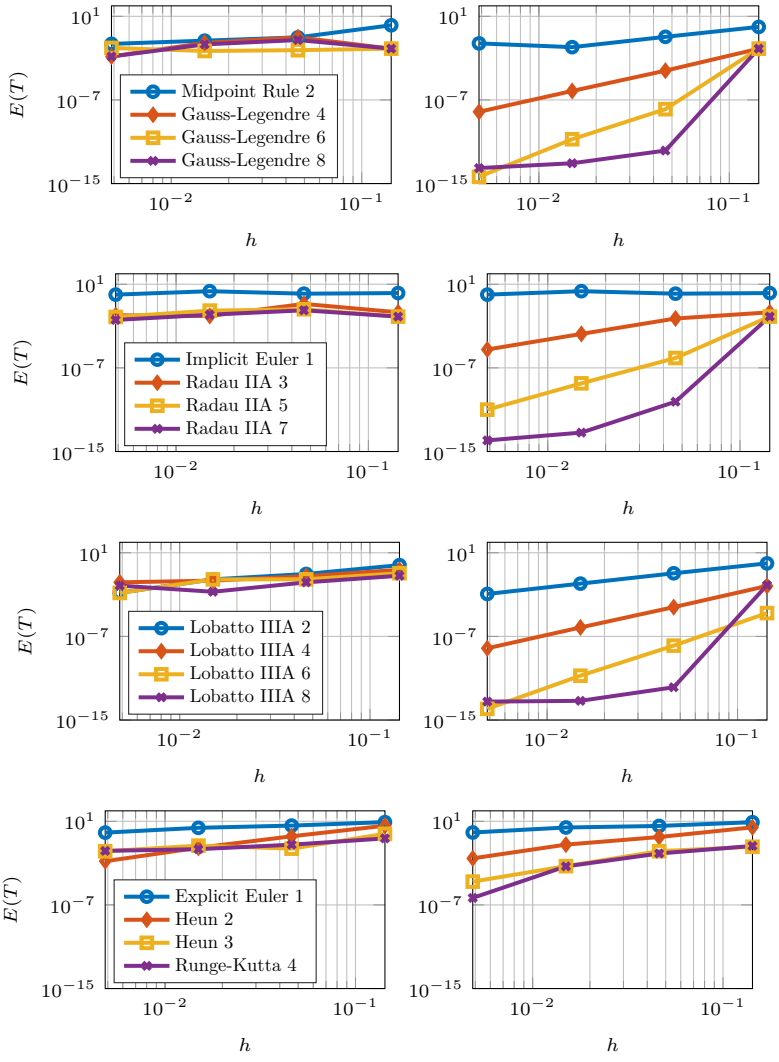


Figure 7.6: Integration error $E(T) = \|x(T) - \hat{x}_h(T)\|$ vs. the step-step size h for different RK methods: standard (left plot) vs. FESD (right plot). The legend provides the name of the used RK method and its order of the global integration error.

the results obtained via FESD to the ones obtained with the standard RK discretization from Section 7.2.1. More information on standard RK methods for DAE can be found in Section 3.2.1. Following Section 6.2, we can write

Method	Global error estimate
Radau IIA	$h^{2n_\mathrm{s}-1}$
Gauss-Legendre	h^{2n_s}
Lobatto IIIA	$h^{2n_\mathrm{s}-2}$
Explicit RK	h^{n_s}

Table 7.1: List of analyzed RK methods and their accuracy order for ODE [126]. Note that for Explicit-RK methods the assertion in the table is true for $n_\mathrm{s} \leq 4$, otherwise $p < n_\mathrm{s}$.

(7.26) in the form of the DCS (6.8), where:

$$F(x) = \begin{bmatrix} A_1 x & A_2 x \end{bmatrix}, \; g(x) = (c(x), -c(x),$$

where $g(x)$ is obtained via Eq. (6.14). We compare the global integration error $E(T) = \|x(T) - \hat{x}_h(T)\|$ obtained with different methods. More precisely, we regard solution approximations to this IVP obtained by standard explicit and implicit RK methods (7.4) and FESD (7.11) with $N_\mathrm{FE} = 2$ and different step-sizes. Both integration methods are implemented in `nosnoc`. Table 3.1 summarize the accuracy order for few commonly used RK methods. Here, we list in Table 7.1 the RK methods considered in the numerical example, together with their order. In the case of explicit methods, we consider the one with $n_\mathrm{s} \leq 4$.

Several other IRK methods are available in `nosnoc`, but we omit the full comparison for brevity. The nominal step h size is obtained by dividing T by the number of simulation steps N_sim. We take an irrational number for $T = \frac{\pi}{2}$ so that $t_\mathrm{s} = 1$ never coincides with a finite element boundary. Therefore, we avoid accidental switch detection via the discretization grid for standard discretization.

The numerical error as a function of the step-sizes for the RK and FESD methods from Table 7.1 are depicted in Figure 7.6. We can see that the standard discretization achieves in all cases only first-order accuracy (left plots), which is consistent with the experiments from Section 4.2.5. On the other hand, the FESD method recovers always the high accuracy order that the underlying RK method has for smooth ODE (right plots). This verifies the result of Theorem 7.10 in practice and demonstrates how FESD can be used as an event-based integrator without an external switch detection procedure. The *saturation* in the right plots of some high-accuracy methods is due to the round-off errors in floating point arithmetic which limit the possible accuracy on a computer.

7.3.5 Convergence of discrete-time sensitivities

Numerical sensitivities are a crucial ingredient for computing stationary points and verifying their optimality in direct optimal control. Here we denote them by $\hat{S}_h^x(t;0,x_0)$ or sometimes more compactly by $\hat{S}_h^x(t,x_0)$.

As extensively discussed in Section 5.2, a fundamental limitation of standard discretization methods for nonsmooth systems is that $\hat{S}_h^x(t,x_0) \not\to S^x(t,x_0)$ as $h \downarrow 0$. In direct optimal control, this can cause convergence to artificial stationary points arbitrarily close to the initialization point [203, 259]. Fortunately, the sensitivities of solutions generated by the FESD method converge to correct values (cf. Subsection 3.2.2). This is shown in the next theorem, but before we state it, we make one more assumption on the time derivatives of the solution approximation.

Assumption 7.11. *(RK derivatives) Regard the RK methods from Assumption 7.3 applied to the differential algebraic equations* (6.15)*. Suppose that the derivatives of the numerical approximation for the same RK method converge with order* $1 \leq q \leq p$*, i.e.,* $\|\dot{\hat{x}}_h(t) - \dot{x}(t)\| = O(h^q),\ t \in \mathcal{G}$.

The aim of the assumption about the convergence of the derivatives of the numerical approximation is to cover a broad class of RK methods, and the value of q depends on the specific choice of an RK method. For example, for collocation-based implicit RK methods for ODE in general it holds that $q = p - 1$ [124, Theorem 7.10]. More specifically, methods that contain the boundary point of a finite element denoted by $\bar{t}$, that is $c_{n_\mathrm{s}} = 1$ satisfy $p = q$. This assertion follows directly from Lipschitz continuity and the fact that the numerical approximations satisfy the ODE at every stage point:

$$\|\dot{\hat{x}}_h(\bar{t}) - \dot{x}(\bar{t})\| = \|f_i(\hat{x}_h(\bar{t})) - f_i(x(\bar{t}))\| \leq L_{f_i}\|\hat{x}_h(\bar{t}) - x(\bar{t})\| = O(h^p).$$

Theorem 7.12 (Convergence to exact sensitivities)**.** *Suppose the assumptions of Theorem 7.10 and Assumption 7.11 hold. Assume that a single active-set change happens at time* $t_{\mathrm{s},n}$*, i.e.,* $||\mathcal{I}_n| - |\mathcal{I}_{n+1}|| \leq 1, n \in \{0, \dots N_{\mathrm{sw}}\}$*. Then for* $h \downarrow 0$ *it holds that* $\hat{S}_h^x(t,x_0) \to S^x(t,x_0)$ *with the convergence rate*

$$\|\hat{S}_h^x(t,x_0) - S^x(t,x_0)\| = O(h^q),\ \textit{for all } t \in \mathcal{G}. \tag{7.27}$$

Proof. Regard partition of $[0,T]$: $0 < \tilde{t}_1 < \cdots < \tilde{t}_k < \cdots < \tilde{t}_{N_k} < T$ such that in the open interval between every two neighboring points there is a single switching point with $N_k > N_{\mathrm{sw}}$. Then by the chain rule, we have

$$S^x(T;0,x_0) = S^x(T;\tilde{t}_{N_k}, x(\tilde{t}_{N_k})) \cdots S^x(\tilde{t}_{k+1};\tilde{t}_k, x(\tilde{t}_k)) \cdots S^x(\tilde{t}_1;0,x_0).$$

Without loss of generality we can assume that on $[0, \tilde{t}_1]$ a single switch occurs at $t_\mathrm{s} \in (0, \tilde{t}_1)$. We show convergence of the sensitivities on this interval. Convergence on $[0, T]$ follows by inductively applying the same arguments on every sub-interval $[\tilde{t}_k, \tilde{t}_{k+1}]$.

Regard the two smooth pieces of the approximation $\hat{x}_h(t)$: 1) $\hat{x}_{h,1}(t, x_0)$ for $t \leq \hat{t}_\mathrm{s}$ and, 2) $\hat{x}_{h,2}(t, y_0(x_0))$ where $y_0(x_0) = \hat{x}_{h,2}(0, y_0(x_0)) = \hat{x}_{h,1}(\hat{t}_\mathrm{s}, x_0)$. With this definition, we have for $t \geq \hat{t}_\mathrm{s}$

$$\hat{x}_{h,2}(t - \hat{t}_\mathrm{s}, y_0(x_0)) = \hat{x}_h(t, x_0). \tag{7.28}$$

From Theorem 7.10 we know that $|\hat{t}_\mathrm{s} - t_\mathrm{s}| = O(h^p)$. Obviously, the value of $\hat{t}_\mathrm{s}$ depends on x_0 and we know from Theorem 7.10 that we obtain implicitly at a switching point the condition

$$\psi_{i,j}(\hat{x}_{h,1}(\hat{t}_\mathrm{s}(x_0), x_0)) = g_i(\hat{x}_{h,1}(\hat{t}_\mathrm{s}(x_0), x_0)) - g_j(\hat{x}_{h,1}(\hat{t}_\mathrm{s}(x_0), x_0)) = 0, \tag{7.29}$$

where $i \notin \mathcal{I}_0$, $i \in \mathcal{I}_1$ and $j \in \mathcal{I}_0$.

For computing $\hat{S}^x_h(\cdot)$ on $[\hat{t}_\mathrm{s}, t_1]$ from Eq. (7.28) we have

$$\begin{aligned}\frac{\partial \hat{x}_h(t, x_0)}{\partial x_0} &= \frac{\partial \hat{x}_{h,2}(t - \hat{t}_\mathrm{s}(x_0), y_0(x_0))}{\partial x_0} = -\dot{\hat{x}}_{h,2}(t - \hat{t}_\mathrm{s}(x_0), y_0(x_0)) \nabla_{x_0} \hat{t}_\mathrm{s}(x_0)^\top \\ &\quad + \frac{\partial \hat{x}_{h,2}(t - \hat{t}_\mathrm{s}(x_0), y_0(x_0))}{\partial y_0} \nabla_{x_0} y_0(x_0).\end{aligned} \tag{7.30}$$

Next, we compute the expressions for the two unknowns $\nabla_{x_0} \hat{t}_\mathrm{s}(x_0)^\top$ and $\nabla_{x_0} y_0(x_0)$. Denote by $\hat{S}^x_{h,1}(t; 0, x_0) = \frac{\partial \hat{x}_{h,1}(t, x_0)}{\partial x_0}$. Using the implicit function theorem for (7.29), we can compute

$$\nabla_{x_0} \hat{t}_\mathrm{s}(x_0)^\top = -\frac{\nabla \psi_{i,j}(\hat{x}_{h,1}(\hat{t}_\mathrm{s}(x_0), x_0))^\top \hat{S}^x_{h,1}(\hat{t}_\mathrm{s}, x_0)}{\nabla \psi_{i,j}(\hat{x}_{h,1}(\hat{t}_\mathrm{s}(x_0), x_0))^\top \dot{\hat{x}}_{h,1}(\hat{t}_\mathrm{s}(x_0), x_0)}. \tag{7.31}$$

At $t = \hat{t}_\mathrm{s}$, we exploit the fact that $\hat{x}_{h,2}(0, y_0(x_0)) = y_0(x_0) = \hat{x}_{h,1}(\hat{t}_\mathrm{s}, x_0)$, thus

$$\nabla_{x_0} y_0(x_0) = \frac{\partial \hat{x}_{h,1}(\hat{t}_\mathrm{s}(x_0), x_0)}{\partial x_0} = \dot{\hat{x}}_{h,1}(\hat{t}_\mathrm{s}(x_0), x_0) \nabla_{x_0} \hat{t}_\mathrm{s}(x_0)^\top + \hat{S}^x_{h,1}(\hat{t}_\mathrm{s}; x_0).$$

Combing the last line with (7.31) we obtain

$$\nabla_{x_0} y_0(x_0) = \left[I - \frac{\dot{\hat{x}}_{h,1}(\hat{t}_\mathrm{s}(x_0), x_0) \nabla \psi_{i,j}(\hat{x}_{h,1}(\hat{t}_\mathrm{s}(x_0), x_0))^\top}{\nabla \psi_{i,j}(\hat{x}_{h,1}(\hat{t}_\mathrm{s}(x_0), x_0))^\top \dot{\hat{x}}_{h,1}(\hat{t}_\mathrm{s}(x_0), x_0)} \right] \hat{S}^x_{h,1}(\hat{t}_\mathrm{s}; x_0). \tag{7.32}$$

We are interested in $\frac{\partial x_h(t;x_0)}{\partial x_0}$ when $t \downarrow \hat{t}_\mathrm{s}$. Note that $\frac{\partial \hat{x}_{h,2}(t-\hat{t}_\mathrm{s}(x_0),y_0(x_0))}{\partial y_0} \to I$ as $t \downarrow \hat{t}_\mathrm{s}$. Thus from (7.30), (7.31) and (7.32) one obtains

$$\begin{aligned} &\frac{\partial x_h(\hat{t}_\mathrm{s}^+;x_0)}{\partial x_0} = \hat{J}_h(\hat{x}_h(\hat{t}_\mathrm{s};x_0))\frac{\partial x_h(\hat{t}_\mathrm{s}^-;x_0)}{\partial x_0}, \text{ with} \\ &\hat{J}_h(\hat{x}_h(\hat{t}_\mathrm{s};x_0)) \coloneqq \left[I + \frac{(\dot{\hat{x}}_h(\hat{t}_\mathrm{s}^+,x_0) - \dot{\hat{x}}_h(\hat{t}_\mathrm{s}^-,x_0))\nabla\psi_{i,j}(\hat{x}_h(\hat{t}_\mathrm{s}^-,x_0))^\top}{\nabla\psi_{i,j}(\hat{x}_h(\hat{t}_\mathrm{s}^-,x_0))^\top \dot{\hat{x}}_h(\hat{t}_\mathrm{s}^-,x_0)}\right]. \end{aligned} \tag{7.33}$$

By the chain rule, we have that for $t > \hat{t}_\mathrm{s}$

$$\hat{S}_h^x(t;x_0) = \hat{S}_{h,2}^x(t;\hat{t}_\mathrm{s}^+,y_0)\hat{J}_h(\hat{x}_h(\hat{t}_\mathrm{s};x_0))\hat{S}_{h,1}^x(t_s^-;0,x_0). \tag{7.34}$$

First, note that for a fixed active set, the FESD equations for $\hat{x}_{h,1}(t,x_0)$ and $\hat{x}_{h,2}(t,y_0)$ boil down to RK equations for the DAE (6.15) with fixed step-size h_n, cf. Theorem 7.7. Differentiating the RK equations to obtain $\hat{S}_h^x(\cdot)$ results in the same RK method applied to the variational differential equations of the system at hand. Thus, the numerical sensitivities converge in this setting to the continuous-time sensitivities with the same accuracy $O(h^p)$ as for the solution of the system [7].

Second, as $h \downarrow 0$, due to assumption of this theorem, in $\hat{J}_h(\hat{x}_h(\hat{t}_\mathrm{s};x_0))$ the functions $\dot{x}_h(t)$ converge to $f(x(t))$ with order q. Due to Theorem 7.10, all other terms converge with order p. Thus, $\|\hat{J}_h(\hat{x}_h(\hat{t}_\mathrm{s};x_0)) - J(x(t_\mathrm{s};x_0))\| = O(h^q)$.

Summarizing the last two arguments and applying them inductively for every active-set change, we conclude that $\hat{S}_h^x(t;x_0) \to S^x(t;x_0)$ as $h \downarrow 0$ with the order $q = \min(p,q)$. This completes the proof. □

The only restrictive assumption we make is that a single active-set change happens at a time. For multiple simultaneous switches, the derivation becomes quite involved even in continuous-time case [94]. Hence, we made this assumption for simplicity. We proceed by illustrating the results Theorem 7.12 in a numerical example.

7.3.6 Illustration of numerical sensitivity convergence

In Section 5.2, we showed that standard methods fail even on simple examples due to the wrong numerical sensitivities. To demonstrate the improvements in FESD compared to standard methods, we repeat the three experiments from Section 5.2.5.

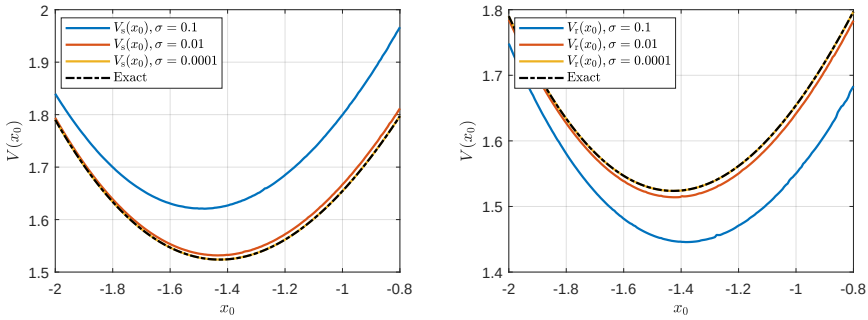


Figure 7.7: Value of the objective functions $V_{\mathrm{s}}(x_0)$ for the smoothed MPCC (left), and $V_{\mathrm{r}}(x_0)$ for the relaxed MPCC (right), for several values of the homotopy parameter σ.

The goal was to numerically solve the OCP:

$$\min_{x_0 \in \mathbb{R}, x(\cdot) \in \mathcal{C}^0} \quad \int_0^2 x(t)^2 \, \mathrm{d}t + (x(2) - 5/3)^2 \tag{7.35a}$$

$$\text{s.t.} \quad x(0) = x_0, \tag{7.35b}$$

$$\dot{x}(t) \in 2 - \operatorname{sign}(x(t)), \ t \in [0, 2]. \tag{7.35c}$$

Denote by $V_*(x_0)$ be the objective value for the unique feasible trajectory starting at $x(0) = x_0$.

In the first experiment, we evaluate $V(x_0)$ by simulating the trajectory of (7.35c) for various x_0 with FESD (7.11). Same as in Section 5.2.5, we pick the Gauss-Legendre fourth-order with $n_{\mathrm{s}} = 2$ and $N_{\mathrm{FE}} = 20$. Figure 7.7 shows the results for using a Scholtes relaxation and smoothing approach while solving the FESD MPCC problem. In contrast to a standard approach (cf. Figure 5.3), both the derivatives and function values converge to the exact value, and no artificial local minima appear.

In the second experiment, we solve the OCP (7.35) for different initial guesses x_0 and different starting values of the relaxation parameter σ_0. The MPCCs are solved with a relaxation homotopy approach, where three different values of the initial homotopy parameter σ_0 of 1, 10^{-1}, and 10^{-4}. The initial guess for the arising MPCC is obtained from a forward simulation of (7.35c) with the same method and grid as in the discretized OCP. The final σ is always 10^{-8} and the homotopy parameter is updated via the rule: $\sigma_{n+1} = 0.1\sigma_n$, and n is the number of the problem in the sequence. Similar to Section 5.2.5, this is

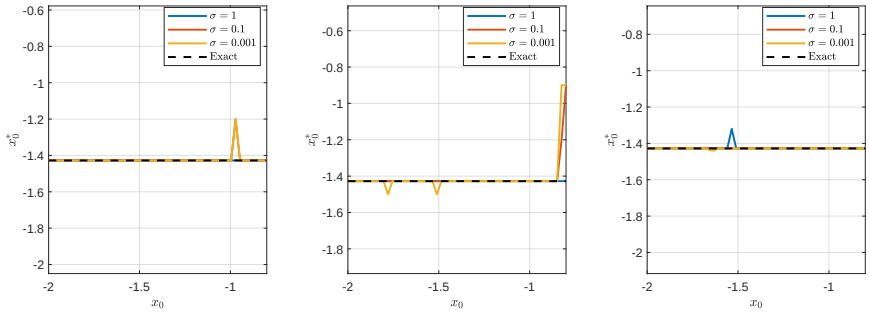


Figure 7.8: Value of the optimal solution x_0^* against different initial values x_0 for which an initial feasible solution guess was computed. The MPCC is solved by a relaxation method for different initial values of σ. The left plot is for $N_{\mathrm{FE}} = 20$, the middle for$N_{\mathrm{FE}} = 40$ and the right plot for $N_{\mathrm{FE}} = 100$.

done for $N_{\mathrm{FE}} = 20$, $N_{\mathrm{FE}} = 40$ and $N_{\mathrm{FE}} = 100$. The results are depicted in Figure 7.8. The few outliers correspond to problem instances where `IPOPT` did not converge. Remarkably, the FESD solution matches the analytic solution for all x_0, regardless of the initial σ and initialization point. In contrast to the standard approach (cf. Figure 5.4), FESD converges to the unique true solution and does not get stuck at artificial local solutions. This highlights that the numerical sensitivities converge to the true ones, as shown in Theorem 7.12.

In the last experiment, we solve a sequence of NLP in a homotopy procedure, where σ is decreased via the rule $\sigma_{n+1} = 0.1\sigma_n$ and $\sigma_0 = 1$. We also compute the solution obtained by a single NLP solve with a fixed σ_n. Again, the experiment is carried out for $N_{\mathrm{FE}} = 20$, $N_{\mathrm{FE}} = 40$, and $N_{\mathrm{FE}} = 100$. Figure 7.9 shows from left to right the results for $N_{\mathrm{FE}} = 20$, $N_{\mathrm{FE}} = 40$ and $N_{\mathrm{FE}} = 100$, respectively. Moreover, we plot the solutions obtained by the standard RK discretization. In the standard case, the homotopy approach led to slightly better results but still with large errors, cf. Figure 5.5. With FESD, both strategies converge to the exact solution for all step sizes, because the derivatives are correct. Nevertheless, in practice, the homotopy strategy yields faster and more reliable convergence than a single NLP solve.

The experiments above demonstrate the superiority of FESD to standard discretization even in a simple example. Remarkably, Figures 7.9 and 7.8 suggest that FESD seems to be well-behaved also on the intermediate relaxed problems, in the sense that the accuracy is high, and the numerical sensitivities are correct. This is not covered by our theoretical analysis but might be very useful for real-time algorithms that solve the OCPs for larger final values of

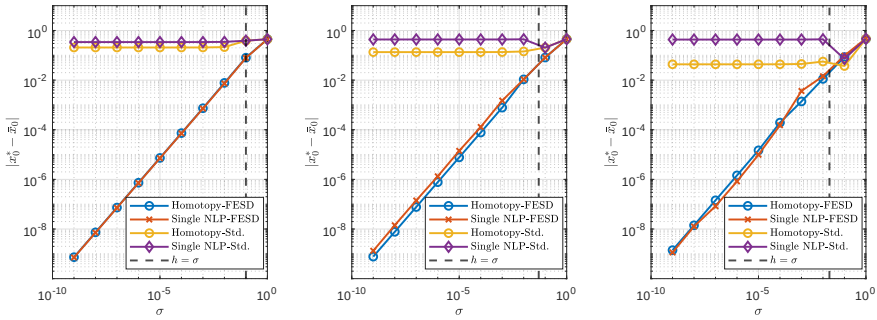


Figure 7.9: Distance of numerically computed optimal solution x_0^*, obtained with FESD and the standard discretization, to the analytic solution $\bar{x}_0$, for different σ. The left plot is for $N_{\mathrm{FE}} = 20$, the middle for$N_{\mathrm{FE}} = 40$ and the right plot for $N_{\mathrm{FE}} = 100$.

the homotopy parameters. Moreover, the experiments show a less surprising fact, i.e., that a homotopy with a larger initial σ_0 usually converges more reliably [159, 237].

7.4 FESD for the Heaviside step representation

We follow now similar lines as in Section 7.2 and start with the RK discretization for the step reformulation DCS. We subsequently introduce the cross complementarity and step equilibration conditions also for this formulation. The section finishes with a summary of the FESD method for the step reformulation.

7.4.1 Standard Runge-Kutta discretization

As a starting point in our analysis, we regard a standard RK discretization of the DCS (6.44), which we restate for the readers' convenience:

$$\dot{x} = F(x, u)\, \theta, \tag{7.36a}$$

$$\theta_i = \prod_{j=1}^{n_\psi} \Big(\frac{1 + S_{i,j}(2\alpha_j - 1)}{2} \Big), \text{ for all } i \in \mathcal{J}, \tag{7.36b}$$

$$\psi(x) = \lambda^{\mathrm{p}} - \lambda^{\mathrm{n}}, \tag{7.36c}$$

$$0 \leq \lambda^{\mathrm{n}} \perp \alpha \geq 0, \tag{7.36d}$$

$$0 \leq \lambda^{\mathrm{p}} \perp e - \alpha \geq 0. \tag{7.36e}$$

For ease of notation, we work with the nonsmooth DAE formulation of the DCS (6.46), which restate for convenience

$$\dot{x} = F(x,u)\theta, \tag{7.37a}$$

$$0 = G(x, \theta, \alpha, \lambda^{\mathrm{p}}, \lambda^{\mathrm{n}}) \tag{7.37b}$$

In the sequel, one should keep in mind that (7.37b) collects all algebraic equations including the complementarity conditions (6.44d)-(6.44e).

We regard a single control interval $[0,T]$ with a constant externally chosen control input $q \in \mathbb{R}^{n_u}$, i.e., we set $u(t) = q$ for $t \in [0,T]$. In Section 7.5, we will treat the discretization of OCPs with multiple control intervals. Let $x(0) = s_0$ be the initial value. The control interval $[0,T]$ is divided into N_{FE} finite elements $[t_n, t_{n+1}]$ via the grid points $0 = t_0 < t_1 < \ldots < t_{N_{\mathrm{FE}}} = T$. On each of the finite elements we regard an n_{s}-stage Runge-Kutta method which is characterized by the Butcher tableau entries $a_{i,j}, b_i$ and c_i with $i,j \in \{1,\ldots,n_{\mathrm{s}}\}$ [126]. The step-sizes read as $h_n = t_{n+1} - t_n,\ n = 0,\ldots,N_{\mathrm{FE}} - 1$. The approximation of the differential state at the grid points t_n is denoted by $x_n \approx x(t_n)$.

We regard the differential representation of the Runge-Kutta method. Thus, the derivatives of states at the stage points $t_{n,i} \coloneqq t_n + c_i h_n,\ i = 1,\ldots,n_{\mathrm{s}}$, are degrees of freedom. For a single finite element, we summarize them in the vector $V_n \coloneqq (v_{n,1},\ldots,v_{n,n_{\mathrm{s}}}) \in \mathbb{R}^{n_{\mathrm{s}} n_x}$. The stage values for the algebraic variables are collected in the vectors: $\Theta_n \coloneqq (\theta_{n,1},\ldots,\theta_{n,n_{\mathrm{s}}}) \in \mathbb{R}^{n_{\mathrm{s}} \cdot n_f}$, $A_n \coloneqq (\alpha_{n,1},\ldots,\alpha_{n,n_{\mathrm{s}}}) \in \mathbb{R}^{n_{\mathrm{s}} \cdot n_c}$, $\Lambda_n^{\mathrm{p}} \coloneqq (\lambda_{n,1}^{\mathrm{p}},\ldots,\lambda_{n,n_{\mathrm{s}}}^{\mathrm{p}}) \in \mathbb{R}^{n_{\mathrm{s}} \cdot n_c}$ and $\Lambda_n^{\mathrm{n}} \coloneqq (\lambda_{n,1}^{\mathrm{n}},\ldots,\lambda_{n,n_{\mathrm{s}}}^{\mathrm{n}}) \in \mathbb{R}^{n_{\mathrm{s}} \cdot n_c}$.

We collect all *internal* variables in the vector $Z_n = (x_n, \Theta_n, A_n, \Lambda_n^{\mathrm{p}}, \Lambda_n^{\mathrm{n}}, V_n)$. The vector x_n^{next} denotes the state value at t_{n+1}, which is obtained after a single integration step. Now, we can state the RK equations for the DCS (7.37) for a single finite element as

$$0 = G_{\mathrm{rk}}(x_n^{\mathrm{next}}, Z_n, h_n, q) \coloneqq \tag{7.38}$$

$$\begin{bmatrix} v_{n,1} - F(x_n + h_n \sum_{j=1}^{n_\mathrm{s}} a_{1,j} v_{n,j}, q)\theta_{n,1} \\ \vdots \\ v_{n,n_\mathrm{s}} - F(x_n + h_n \sum_{j=1}^{n_\mathrm{s}} a_{n_\mathrm{s},j} v_{n,j}, q)\theta_{n,n_\mathrm{s}} \\ G(x_n + h_n \sum_{j=1}^{n_\mathrm{s}} a_{1,j} v_{n,j}, \theta_{n,1}, \alpha_{n,1}, \lambda^\mathrm{p}_{n,1}, \lambda^\mathrm{n}_{n,1}) \\ \vdots \\ G(x_n + h_n \sum_{j=1}^{n_\mathrm{s}} a_{n_\mathrm{s},j} v_{n,j}, \theta_{n,n_\mathrm{s}}, \alpha_{n,n_\mathrm{s}}, \lambda^\mathrm{p}_{n,n_\mathrm{s}}, \lambda^\mathrm{n}_{n,n_\mathrm{s}}) \\ x_n^{\mathrm{next}} - x_n - h_n \sum_{i=1}^{n_\mathrm{s}} b_i v_{n,i} \end{bmatrix}.$$

Next, we summarize the equations for all N_FE finite elements over the entire interval $[0, T]$ in a discrete-time system format. To make it more manageable, we use some additional shorthand notation and group all variables of all finite elements for a single control interval into the following vectors: $\mathbf{x} = (x_0, \ldots, x_{N_\mathrm{FE}}) \in \mathbb{R}^{(N_\mathrm{FE}+1)n_x}$, $\mathbf{V} = (V_0, \ldots, V_{N_\mathrm{FE}-1}) \in \mathbb{R}^{N_\mathrm{FE} n_\mathrm{s} n_x}$ and $\mathbf{h} \coloneqq (h_0, \ldots, h_{N_\mathrm{FE}-1}) \in \mathbb{R}^{N_\mathrm{FE}}$. Recall that the simple continuity condition $x_{n+1} = x_n^{\mathrm{next}}$ holds. We collect the stage values of the Filippov multipliers in the vector $\mathbf{\Theta} = (\Theta_0, \ldots, \Theta_{N_\mathrm{FE}-1}) \in \mathbb{R}^{n_\theta}$ and $n_\theta = N_\mathrm{FE} n_\mathrm{s} n_f$. Similarly, we group the stage values of the algebraic variables specific to the step representation in vectors $\mathbf{A}, \mathbf{\Lambda}^\mathrm{p}, \mathbf{\Lambda}^\mathrm{n} \in \mathbb{R}^{n_\alpha}$, where $n_\alpha = N_\mathrm{FE} n_\mathrm{s} n_c$. Finally, we collect all internal variables in the vector $\mathbf{Z} = (\mathbf{x}, \mathbf{V}, \mathbf{\Theta}, \mathbf{A}, \mathbf{\Lambda}^\mathrm{p}, \mathbf{\Lambda}^\mathrm{n}) \in \mathbb{R}^{n_\mathbf{z}}$, where $n_\mathbf{z} = (N_\mathrm{FE} + 1)n_x + N_\mathrm{FE} n_\mathrm{s} n_x + n_\theta + 3n_\alpha$.

All computations over a single control interval of the standard discretization (denoted by the subscript std in the corresponding functions) are summarized in the following equations:

$$s_1 = F_\mathrm{std}(\mathbf{Z}), \tag{7.39a}$$

$$0 = G_\mathrm{std}(\mathbf{Z}, \mathbf{h}, s_0, q), \tag{7.39b}$$

where $s_1 \in \mathbb{R}^{n_x}$ is the approximation of $x(T)$ and

$$F_\mathrm{std}(\mathbf{Z}) = x_{N_\mathrm{FE}},$$

$$G_\mathrm{std}(\mathbf{Z}, \mathbf{h}, s_0, q) \coloneqq \begin{bmatrix} x_0 - s_0 \\ G_\mathrm{rk}(x_1, Z_0, h_0, q) \\ \vdots \\ G_\mathrm{rk}(x_{N_\mathrm{FE}}, Z_{N_\mathrm{FE}-1}, h_{N_\mathrm{FE}-1}, q) \end{bmatrix}.$$

In (7.39), $\mathbf{h}$ is a given parameter and implicitly fixes the discretization grid. In contrast to standard RK discretizations, we will now proceed by letting $\mathbf{h}$ be degrees of freedom and introduce the cross-complementarity conditions for the step reformulation.

7.4.2 Cross complementarity

For ease of exposition, suppose that the underlying RK scheme satisfies $c_{n_\mathrm{s}} = 1$, e.g., this holds for Radau and Lobatto schemes [126]. This means that the right boundary point of a finite element is a stage point, since $t_{n+1} = t_n + c_{n_\mathrm{s}} h_n$. We provide extensions for $c_{n_\mathrm{s}} \neq 1$, at the end of the section.

The goal is to derive additional constraints that will allow active-set changes only at the boundary of a finite element. Moreover, in this case, the step-size h_n should adapt such that the switch is detected exactly. Recall that for the step reformulation at every stage point we have the complementarity conditions:

$$0 \leq \lambda^{\mathrm{n}}_{n,m} \perp \alpha_{n,m} \geq 0, \qquad n = 1, \ldots, N_{\mathrm{FE}}, m = 1, \ldots, n_\mathrm{s}, \tag{7.40a}$$

$$0 \leq \lambda^{\mathrm{p}}_{n,m} \perp e - \alpha_{n,m} \geq 0, \qquad n = 1, \ldots, N_{\mathrm{FE}}, m = 1, \ldots, n_\mathrm{s}. \tag{7.40b}$$

As a first step, we exploit the continuity of the Lagrange multipliers λ^{p} and λ^{n}, cf. Section 6.3.4. We regard the boundary values of the approximation of λ^{p} and λ^{n} on an interval $[t_n, t_{n+1}]$. They are denoted by $\lambda^{\mathrm{p}}_{n,0}$, $\lambda^{\mathrm{n}}_{n,0}$ (which we define blow) at t_n and $\lambda^{\mathrm{p}}_{n,n_\mathrm{s}}$, $\lambda^{\mathrm{n}}_{n,n_\mathrm{s}}$ at t_{n+1}.

Next, we impose a continuity condition for the discrete-time versions of λ^{p} and λ^{n} for all $n \in \{0, \ldots, N_{\mathrm{FE}} - 1\}$:

$$\lambda^{\mathrm{p}}_{n+1,0} = \lambda^{\mathrm{p}}_{n,n_\mathrm{s}}, \ \lambda^{\mathrm{n}}_{n+1,0} = \lambda^{\mathrm{n}}_{n,n_\mathrm{s}}. \tag{7.41}$$

Note that $\lambda^{\mathrm{p}}_{0,0}$ and $\lambda^{\mathrm{n}}_{0,0}$ are not defined via Eq. (7.41), as we do not have a preceding finite element for $n = 0$. Nevertheless, they are crucial for determining the active set in the first finite element. They are not degrees of freedom but parameters determined by a given x_0. Using equation (6.35) we obtain $\lambda^{\mathrm{p}}_{0,0} = \max(\psi(x_0), 0)$ and $\lambda^{\mathrm{n}}_{0,0} = -\min(\psi(x_0), 0)$.

We have seen in Section 6.3.4 that, due to continuity, $\lambda^{\mathrm{p}}_i(t)$ and $\lambda^{\mathrm{n}}_i(t)$ must be zero at an active set change. Moreover, on an interval $t \in (t_n, t_{n+1})$ with a fixed active set, the components of these multipliers are either zero or positive on the whole interval. The discrete-time counterparts, i.e., the stage values $\lambda^{\mathrm{p}}_{n,m}$ and $\lambda^{\mathrm{n}}_{n,m}$ should satisfy these properties as well. We achieve these goals via the cross complementarity conditions, which read for all $n \in \{0, \ldots, N_{\mathrm{FE}} - 1\}$ as:

$$0 = \operatorname{diag}(\lambda^{\mathrm{n}}_{n,m'})\alpha_{n,m}, \qquad m = 1, \ldots, n_\mathrm{s}, \ m' = 0, \ldots, n_\mathrm{s}, \ m \neq m', \tag{7.42a}$$

$$0 = \operatorname{diag}(\lambda^{\mathrm{p}}_{n,m'})(e - \alpha_{n,m}), \quad m = 1, \ldots, n_\mathrm{s}, m' = 0, \ldots, n_\mathrm{s}, \ m \neq m'. \tag{7.42b}$$

In contrast to Eq. (7.41), here we have conditions relating variables corresponding to different RK stages within a finite element. Eq. (7.42) extends

the complementarity conditions for the same RK-stage, i.e., for $m = m'$, which are part of the standard RK equations, cf. Eq. (7.40).

Some of the claims about the constraints (7.42) are formalized by the next lemma. Recall that in our notation $\alpha_{n,m,j}$ is the j-th component of the vector $\alpha_{n,m}$.

Lemma 7.13. *Regard a fixed $n \in \{0, \dots, N_{\mathrm{FE}}-1\}$ and a fixed $j \in \mathcal{C}$. If any $\alpha_{n,m,j}$ with $m \in \{1, \dots, n_{\mathrm{s}}\}$ is positive, then all $\lambda^{\mathrm{n}}_{n,m',j}$ with $m' \in \{0, \dots, n_{\mathrm{s}}\}$ must be zero. Conversely, if any $\lambda^{\mathrm{n}}_{n,m',j}$ is positive, then all $\alpha_{n,m,j}$ are zero.*

Proof. Follows from similar lines as the proof of Lemma 7.2. □

An analogous statement holds for $\lambda^{\mathrm{p}}_{n,m}$ and $(e - \alpha_{n,m})$.

It is now left to discuss why the boundary points $\lambda^{\mathrm{p}}_{n+1,0} = \lambda^{\mathrm{p}}_{n,0}$ and $\lambda^{\mathrm{n}}_{n+1,0} = \lambda^{\mathrm{n}}_{n,0}$ of the previous finite element are included in the cross complementarity conditions (7.42). It turns out, they are the key to switch detection. A consequence of Lemma 7.13 is that, if the active set changes in the j-th component between the n-th and $(n+1)$-st finite element, then it must hold that $\lambda^{\mathrm{p}}_{n,n_{\mathrm{s}},j} = \lambda^{\mathrm{p}}_{n+1,0,j} = 0$ and $\lambda^{\mathrm{n}}_{n,n_{\mathrm{s}},j} = \lambda^{\mathrm{n}}_{n+1,0,j} = 0$. Since $x^{\mathrm{next}}_n = x_{n+1}$, we have from (7.38) the condition

$$\psi_j(x_{n+1}) = 0,$$

which defines the switching surface between two regions. Therefore, we have implicitly a constraint that forces h_n to adapt such that the switch is detected exactly.

For clarity, the conditions (7.42) are given in their sparsest form. However, the nonnegativity of $\alpha_{n,m}, \lambda^{\mathrm{p}}_{n,m}$ and $\lambda^{\mathrm{n}}_{n,m}$ allows many equivalent and more compact forms. For instance, we can use inner products instead of componentwise products or we can even summarize all constraints for a finite element, or for all finite elements in a single equation. In the sequel, we use a formulation where, together with the constraint $\sum_{n=0}^{N_{\mathrm{FE}}-1} h_n = T$, we have the same number of new equations as new degrees of freedom by varying h_n. Thus, we combine constraints of two neighboring finite elements and have the compact formulation

$$G_{\mathrm{cross}}(\mathbf{A}, \mathbf{\Lambda}^{\mathrm{p}}, \mathbf{\Lambda}^{\mathrm{n}}) = 0 \tag{7.43}$$

whose entries are for all $n \in \{0, \dots, N_{\mathrm{FE}} - 2\}$ given by

$$G_{\mathrm{cross},n}(\mathbf{A}, \mathbf{\Lambda}^{\mathrm{p}}, \mathbf{\Lambda}^{\mathrm{n}}) = \sum_{k=n}^{n+1} \Big(\sum_{m=1}^{n_{\mathrm{s}}} \sum_{\substack{m'=0,\\ m' \neq m}}^{n_{\mathrm{s}}} \alpha_{k,m}^{\top} \lambda^{\mathrm{n}}_{k,m'} + (e - \alpha_{k,m})^{\top} \lambda^{\mathrm{p}}_{k,m'} \Big).$$

We remind the reader that we use this seemingly complicated form only to obtain a square system of equations. This simplifies the study of the well-posedness of the FESD equations later. However, in an implementation one can use any of the equivalent, more sparse, or dense formulations. Many possible variants are implemented in `NOSNOC` [206], and the user can control the sparsity.

7.4.3 Step size equilibration

To complete the derivation of the FESD method for the step reformulation (7.37), we need to derive the step equilibration conditions. Here, *step* is referred to the integration step size and is not to be confused with the set-valued step function.

If no active-set changes happen, the cross complementarity constraints (7.42) are implied by the standard complementarity conditions (7.40). Therefore, we end up with a system of equations with more degrees of freedom than conditions. The step equilibration constraints aim to remove the degrees of freedom in the appropriate h_n if no switches happen. This results in a piecewise uniform discretization grid for the differential and algebraic states on the considered time interval.

We achieve the goals outlined above via the equation:

$$0 = G_{\mathrm{eq}}(\mathbf{h}, \mathbf{A}, \mathbf{\Lambda}^{\mathrm{p}}, \mathbf{\Lambda}^{\mathrm{n}}) \coloneqq \begin{bmatrix} (h_1 - h_0)\eta_1(\mathbf{A}, \mathbf{\Lambda}^{\mathrm{p}}, \mathbf{\Lambda}^{\mathrm{n}}) \\ \vdots \\ (h_{N_{\mathrm{FE}}-1} - h_{N_{\mathrm{FE}}-2})\eta_{N_{\mathrm{FE}}-1}(\mathbf{A}, \mathbf{\Lambda}^{\mathrm{p}}, \mathbf{\Lambda}^{\mathrm{n}}) \end{bmatrix}, \tag{7.44}$$

where η_n is an indicator function that is zero only if a switch occurs, otherwise its value is strictly positive. This provides a condition that removes the spurious degrees of freedom. In the remainder of this section, we derive a possible expression for η_n.

The derivations below are motivated by the following facts. Let t_n be a switching point with $\psi_j(x(t_n)) = 0$ for some $j \in \mathcal{C}$. Consequently, it holds that $\lambda_j^{\mathrm{p}}(t_n) = \lambda_j^{\mathrm{n}}(t_n) = 0$. If, for example, a switch occurs at $t_{\mathrm{s},n}$ such that $\psi(x(t_{\mathrm{s},n}^-)) < 0$ and $\psi(x(t_{\mathrm{s},n}^+)) > 0$, we have that $\dot{\lambda}^{\mathrm{n}}{}_j(t_{\mathrm{s},n}^-) < 0$, $\dot{\lambda}^{\mathrm{p}}{}_j(t_{\mathrm{s},n}^-) = 0$ and that $\dot{\lambda}^{\mathrm{n}}{}_j(t_{\mathrm{s},n}^+) = 0$, $\dot{\lambda}^{\mathrm{n}}{}_j(t_{\mathrm{s},n}^+) > 0$. The symmetric case is possible as well. The absolute values of these directional derivatives help us to encode the switching logic. This state of affairs can be seen in Figure 6.3.

Now, instead of looking at the time derivatives, in the discrete-time case, we exploit the non-negativity of $\lambda_{n,m}^{\mathrm{p}}$, $\lambda_{n,m}^{\mathrm{n}}$, and the fact that the active set is fixed for the whole finite element. For $n \in \{1, \ldots, N_{\mathrm{FE}} - 1\}$, we define the

Switching case	$\sigma_n^{\lambda^{\mathrm{n}},\mathrm{B}}$	$\sigma_n^{\lambda^{\mathrm{n}},\mathrm{F}}$	$\sigma_n^{\lambda^{\mathrm{p}},\mathrm{B}}$	$\sigma_n^{\lambda^{\mathrm{p}},\mathrm{F}}$	$\pi_n^{\lambda^{\mathrm{n}}}$	$\pi_b^{\lambda^{\mathrm{p}}}$	υ_n
No switch	1	1	0	0	1	0	1
Crossing	1	0	0	1	0	0	0
Entering sliding mode	1	0	0	0	0	0	0
Leaving sliding mode	0	0	0	1	0	0	0
Spontaneous switch	0	0	0	1	0	0	0

Table 7.2: Overview of switching cases for the step size equilibration

following backward and forward sums of the stage values over the neighboring finite elements $[t_{n-1}, t_n]$ and $[t_n, t_{n+1}]$:

$$\sigma_n^{\lambda^{\mathrm{p}},\mathrm{B}} = \sum_{m=0}^{n_{\mathrm{s}}} \lambda_{n-1,m}^{\mathrm{p}}, \quad \sigma_n^{\lambda^{\mathrm{p}},\mathrm{F}} = \sum_{m=0}^{n_{\mathrm{s}}} \lambda_{n,m}^{\mathrm{p}},$$

$$\sigma_n^{\lambda^{\mathrm{n}},\mathrm{B}} = \sum_{m=0}^{n_{\mathrm{s}}} \lambda_{n-1,m}^{\mathrm{n}}, \quad \sigma_n^{\lambda^{\mathrm{n}},\mathrm{F}} = \sum_{m=0}^{n_{\mathrm{s}}} \lambda_{n,m}^{\mathrm{n}}.$$

They are zero if the left and right time derivatives are zero, respectively. Likewise, they are positive when the left and right time derivatives are nonzero.

Moreover, for all $n \in \{1, \ldots, N_{\mathrm{FE}} - 1\}$ we define the following variables to summarize the logical dependencies:

$$\pi_n^{\lambda^{\mathrm{n}}} = \mathrm{diag}(\sigma_n^{\lambda^{\mathrm{n}},\mathrm{B}})\sigma_n^{\lambda^{\mathrm{n}},\mathrm{F}} \in \mathbb{R}^{n_\psi},$$

$$\pi_n^{\lambda^{\mathrm{p}}} = \mathrm{diag}(\sigma_n^{\lambda^{\mathrm{p}},\mathrm{B}})\sigma_n^{\lambda^{\mathrm{p}},\mathrm{F}} \in \mathbb{R}^{n_\psi},$$

and

$$\upsilon_n = \pi_n^{\lambda^{\mathrm{n}}} + \pi_n^{\lambda^{\mathrm{p}}} \in \mathbb{R}^{n_\psi}.$$

The switching cases and sign logic is summarized in Table 7.2. For readability, we put in the table a one if a variable is positive and a zero if it is zero. Let us discuss how the variables above encode the switching logic, and for this purpose, we regard the j-th switching functions $\psi_j(x)$. If no switch occurs, and for example, we have that $\psi_j(x(t)) < 0$ during the regard time interval, it follows that $\lambda_j^{\mathrm{n}}(t) > 0$ and $\lambda_j^{\mathrm{p}}(t) = 0$ during this time interval. In the discrete time setting, we have $\sigma_{n,j}^{\lambda^{\mathrm{n}},\mathrm{B}}, \sigma_{n,j}^{\lambda^{\mathrm{n}},\mathrm{F}} > 0$ and $\sigma_{n,j}^{\lambda^{\mathrm{p}},\mathrm{B}} = \sigma_{n,j}^{\lambda^{\mathrm{p}},\mathrm{F}} = 0$. This means that $\pi_{n,j}^{\lambda^{\mathrm{n}}} > 0$, $\pi_{n,j}^{\lambda^{\mathrm{n}}} = 0$ and $\upsilon_{n,j} > 0$. It can be seen that the symmetric case with $\psi_j(x(t)) > 0$ leads also to $\upsilon_{n,j} > 0$, hence we do not enumerate all symmetric cases in Table 7.2.

On the other hand, if we have a switch of the crossing type (cf. top plots in Figure 6.3 with $\psi_j(x(t)) < 0$ for $t < t_{\mathrm{s},n}$ and $\psi_j(x(t)) > 0$ for $t > t_{\mathrm{s},n}$, it follows that $\lambda_j^{\mathrm{n}}(t) < 0, \lambda_j^{\mathrm{p}}(t) = 0$ for $t < t_{\mathrm{s},n}$ and $\lambda_j^{\mathrm{n}}(t) = 0, \lambda_j^{\mathrm{p}}(t) > 0$ for $t > t_{\mathrm{s},n}$. In the discrete-time setting we obtain the sign pattern as in the second row of Table 7.2, with $\upsilon_{n,j} = 0$.

In general, if there is an active-set change in the j-th complementarity pair, then at most one of the j-th components of $\sigma_n^{\lambda^{\mathrm{p}},\mathrm{B}}$ and $\sigma_n^{\lambda^{\mathrm{p}},\mathrm{F}}$, or $\sigma_n^{\lambda^{\mathrm{n}},\mathrm{B}}$ and $\sigma_n^{\lambda^{\mathrm{n}},\mathrm{F}}$ is nonzero. In these cases, we obtain that $\upsilon_{n,j} = 0$, and if now switch happens we have that $\upsilon_{n,j} > 0$.

In other words, $\upsilon_{n,j}$ is only zero if there is an active-set change in the j-th complementarity pair at t_n, otherwise, it is strictly positive. We summarize all logical relations for all switching functions into a single scalar expression and define

$$\eta_n(\mathbf{A}, \mathbf{\Lambda}^{\mathrm{p}}, \mathbf{\Lambda}^{\mathrm{n}}) \coloneqq \prod_{i=1}^{n_\psi} (\upsilon_n)_i.$$

It is zero only if an active-set change happens at the boundary point t_n, otherwise, it is strictly positive.

7.4.4 The FESD discretization

We have now introduced all extensions needed to pass from a standard RK discretization (7.39) to the FESD discretization for the step reformulation. With a slight abuse of notation, we collect all equations in a discrete-time system form:

$$s_1 = F_{\mathrm{fesd}}(\mathbf{Z}), \tag{7.45a}$$

$$0 = G_{\mathrm{fesd}}(\mathbf{Z}, \mathbf{h}, s_0, q, T), \tag{7.45b}$$

where $F_{\mathrm{fesd}}(\mathbf{x}) = x_{N_{\mathrm{FE}}}$ is the state transition map and $G_{\mathrm{fesd}}(\mathbf{x}, \mathbf{h}, \mathbf{Z}, q, T)$ collects all other internal computations including all RK steps within the regarded time interval:

$$G_{\mathrm{fesd}}(\mathbf{Z}, \mathbf{h}, s_0, q, T) \coloneqq \begin{bmatrix} G_{\mathrm{std}}(\mathbf{Z}, \mathbf{h}, s_0, q, T) \\ G_{\mathrm{cross}}(\mathbf{A}, \mathbf{\Lambda}^{\mathrm{p}}, \mathbf{\Lambda}^{\mathrm{n}}) \\ G_{\mathrm{eq}}(\mathbf{h}, \mathbf{A}, \mathbf{\Lambda}^{\mathrm{p}}, \mathbf{\Lambda}^{\mathrm{n}}) \\ \sum_{n=0}^{N_{\mathrm{FE}}-1} h_n - T \end{bmatrix}. \tag{7.46}$$

Here, control variable q, horizon length T, and initial value s_0 are given parameters.

Remark on RK methods with $c_{n_\mathrm{s}} \neq 1$. The extension for the case of an RK method with $c_{n_\mathrm{s}} \neq 1$ follows similar lines as in Stewart's formulation [213]. We have that $t_n + c_{n_\mathrm{s}} h_n < t_{n+1}$. Hence, the variables $\lambda^{\mathrm{p}}_{n,n_\mathrm{s}}$ and $\lambda^{\mathrm{n}}_{n,n_\mathrm{s}}$ do not correspond to the boundary values $\lambda^{\mathrm{n}}(t_{n+1})$ and $\lambda^{\mathrm{p}}(t_{n+1})$ anymore. We denote the true boundary points by $\lambda^{\mathrm{p}}_{n,n_\mathrm{s}+1}$ and $\lambda^{\mathrm{n}}_{n,n_\mathrm{s}+1}$. They can be computed from the KKT conditions of the step reformulation LP (6.33). For all $n \in \{1, \dots, N_{\mathrm{FE}} - 2\}$ we have

$$\begin{bmatrix} \psi(x_{n+1}) - \lambda^{\mathrm{p}}_{n,n_\mathrm{s}+1} + \lambda^{\mathrm{n}}_{n,n_\mathrm{s}+1} \\ \Psi(\lambda^{\mathrm{n}}_{n,n_\mathrm{s}+1}, \alpha_{n,n_\mathrm{s}+1}) \\ \Psi(\lambda^{\mathrm{p}}_{n,n_\mathrm{s}+1}, e - \alpha_{n,n_\mathrm{s}+1}) \end{bmatrix} = 0. \tag{7.47}$$

These equations are appended to the FESD equation in (7.46).

However, to make the switch detection work, we must update the continuity conditions for the discrete-time versions of the Lagrange multipliers and adapt the cross-complementarity conditions accordingly. For all $n = \{0, \dots, N_{\mathrm{FE}} - 1\}$, Eq (7.41) is replaced by:

$$\lambda^{\mathrm{p}}_{n,n_\mathrm{s}+1} = \lambda^{\mathrm{p}}_{n+1,0},\ \lambda^{\mathrm{n}}_{n,n_\mathrm{s}+1} = \lambda^{\mathrm{n}}_{n+1,0}. \tag{7.48}$$

We append to the vectors $\mathbf{A}, \mathbf{\Lambda}^{\mathrm{p}}$ and $\mathbf{\Lambda}^{\mathrm{n}}$ the new variables $\alpha_{n,n_\mathrm{s}+1}, \lambda^{\mathrm{p}}_{n,n_\mathrm{s}+1}$ and $\lambda^{\mathrm{n}}_{n,n_\mathrm{s}+1}$ accordingly. For the whole control interval, we have in total $3(N_{\mathrm{FE}}-1)n_c$ new variables. It is only left to state the modified cross complementarity conditions, including the expressions' $n_\mathrm{s}+1$-st points. More explicitly, the n-th component of (7.43) reads now for all $n \in \{0, \dots, N_{\mathrm{FE}} - 2\}$ as

$$G_{\mathrm{cross},n}(\mathbf{A}, \mathbf{\Lambda}^{\mathrm{p}}, \mathbf{\Lambda}^{\mathrm{n}}) = \sum_{k=n}^{n+1} \sum_{m=1}^{n_\mathrm{s}} \sum_{\substack{m'=0, \\ m' \neq m}}^{n_\mathrm{s}+1} \alpha_{k,m}^{\top} \lambda^{\mathrm{n}}_{k,m'} + (e - \alpha_{k,m})^{\top} \lambda^{\mathrm{p}}_{k,m'}.$$

The theoretical analysis of the FESD method for the Heaviside step reformulation follows similar lines as the developments in Section 7.3. For brevity, we omit restating all assumptions and theorems explicitly and refer the interested reader to [207].

7.5 FESD in direct optimal control

In Sections 7.2 and 7.4, we have developed the FESD method for Stewart's and the step reformulation, respectively. Thereby, all developments focus on a single control interval with a given control input. We show how to use the FESD

method in direct optimal control, i.e., in a first-discretize-then-optimize approach. For the sake of simplicity, our discussion is based on Stewart's reformulation. However, it is easy to extend the method to the step reformulation. We consider the following continuous-time optimal control problem on a control horizon $[0, T_{\text{ctrl}}]$:

$$\min_{x(\cdot),u(\cdot),z(\cdot)} \quad \int_0^{T_{\text{ctrl}}} L_{\text{stg}}(x(t), u(t))\mathrm{d}t + L_{\text{end}}(x(T_{\text{ctrl}})) \tag{7.49a}$$

$$\text{s.t.} \quad x(0) = x_0, \tag{7.49b}$$

$$\dot{x}(t) = F(x(t), u(t))\theta(t), \qquad t \in [0, T_{\text{ctrl}}], \tag{7.49c}$$

$$0 = g(x(t)) - \lambda(t) - \mu(t)e, \qquad t \in [0, T_{\text{ctrl}}], \tag{7.49d}$$

$$1 = e^\top \theta(t), \qquad t \in [0, T_{\text{ctrl}}], \tag{7.49e}$$

$$0 \le \theta(t) \perp \lambda(t) \ge 0, \qquad t \in [0, T_{\text{ctrl}}], \tag{7.49f}$$

$$0 \le G_{\text{ineq}}(x(t), u(t)), \qquad t \in [0, T_{\text{ctrl}}], \tag{7.49g}$$

$$0 \le G_{\text{end}}(x(T_{\text{ctrl}})), \tag{7.49h}$$

where x_0 is a given initial value, $u(t) \in \mathbb{R}^{n_u}$ is the control function, $z(t) = (\lambda(t), \theta(t), \mu(t)) \in \mathbb{R}^{2n_f+1}$ collects the algebraic variables. The function $L_{\text{stg}} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ defines the stage cost (the Lagrange objective term), and $L_{\text{end}} : \mathbb{R}^{n_x} \to \mathbb{R}$ is the terminal cost (the Mayer term). The path and terminal constraints are collected in the functions $G_{\text{ineq}} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_{g1}}$ and $G_{\text{end}} : \mathbb{R}^{n_x} \to \mathbb{R}^{n_{g2}}$, respectively.

7.5.1 A multiple shooting-type discretization

We discretize the OCP (7.49) with $N \ge 1$ control intervals indexed by k. The control function approximation is taken to be piecewise constant on an equidistant grid. This is the usual choice in direct multiple shooting [44] and is required by many practical applications of feedback control but could be easily generalized to any other local control parameterization. The constant controls are collected in $\mathbf{q} = (q_0, \ldots, q_{N-1}) \in \mathbb{R}^{Nn_u}$. All internal variables of every control interval are additionally equipped with an index k. On every control interval k with a fixed duration of $T = \frac{T_{\text{ctrl}}}{N}$, we apply the discretization outlined in equation (7.11) with N_{FE} internal finite elements. The state values at the control interval boundaries are collected in the vector $\mathbf{s} = (s_0, \ldots, s_N) \in \mathbb{R}^{(N+1)n_x}$. Additionally, the following vectors collect all internal variables of

every discretization step: $\mathcal{H} = (\mathbf{h}_0, \ldots, \mathbf{h}_{N-1})$ and $\mathcal{Z} = (\mathbf{Z}_0, \ldots, \mathbf{Z}_{N-1})$. The FESD discretization of the OCP (7.49) together with the aforementioned control discretization reads as:

$$\min_{\mathbf{s},\mathbf{q},\mathcal{H},\mathcal{Z}} \quad \sum_{k=1}^{N-1} \hat{L}_{\text{stg}}(s_k, \mathbf{x}_k, q_k) + \hat{L}_{\text{end}}(s_N) \tag{7.50a}$$

$$\text{s.t.} \quad s_0 = \bar{x}_0, \tag{7.50b}$$

$$s_{k+1} = F_{\text{fesd}}(\mathbf{Z}_k), \qquad k = 0, \ldots, N-1, \tag{7.50c}$$

$$0 = G_{\text{fesd}}(\mathbf{Z}_k, \mathbf{h}_k, s_k, q_k, T_k), \qquad k = 0, \ldots, N-1, \tag{7.50d}$$

$$0 \leq G_{\text{ineq}}(\mathbf{x}_k, q_k), \qquad k = 0, \ldots, N-1, \tag{7.50e}$$

$$h_{\min} e \leq \mathbf{h}_k \leq h_{\max} e, \qquad k = 0, \ldots, N-1, \tag{7.50f}$$

$$0 \leq G_{\text{end}}(s_N). \tag{7.50g}$$

where $\hat{L}_{\text{stg}} : \mathbb{R}^{n_x} \times \mathbb{R}^{(N_{\text{FE}}+1)n_s n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ and $\hat{L}_{\text{end}} : \mathbb{R}^{n_x} \to \mathbb{R}$ are the discretized stage and terminal costs, respectively. The scalars $h_{\min}$ and $h_{\max}$ are the lower and upper bounds for the step sizes. The box constraint (7.50f) prohibits negative step sizes and bounds the variability of the step size. Note that within this formulation, at every control interval the constraint $\sum_{n=1}^{N_{\text{FE}}} h_{n,k} = T$ is imposed as part of (7.50d), cf. Eq (7.10). It is easy to see that with the step representation we obtain an NLP very similar (7.50) after discretization.

Recall that Eq. (7.50d) contains complementarity constraints. Therefore, the NLP (7.50) is a Mathematical Program with Complementarity Constraints (MPCC). As discussed in Section 2.4, MPCC can often be solved efficiently via reformulations and homotopy approaches. Several relaxation, smoothing, and penalty homotopy approaches are implemented `nosnoc` [206].

7.5.2 A numerical optimal control example

In this section, we illustrate in an optimal control problem several features developed in this chapter: FESD for Cartesian Products of Filippov systems (Section 6.2.5), handling multiple sliding modes, step equilibration (Sec. 7.2.4 and 7.4.3), and equidistant control discretization grids (Section 7.5.1). We also include a benchmark where we compare the accuracy and computational time of the standard and the FESD method.

We regard the following optimal control problem:

$$\min_{x(\cdot),u(\cdot)} \quad \int_0^4 u(t)^\top u(t) + v(t)^\top v(t)\,\mathrm{d}t+; \tag{7.51a}$$

$$\text{s.t.} \quad x(0) = (\frac{2\pi}{3}, \frac{\pi}{3}, 0, 0), \tag{7.51b}$$

$$\dot{x}(t) = \begin{bmatrix} -\text{sign}(c(x(t))) + v(t) \\ u(t) \end{bmatrix}, \quad t \in [0,4], \tag{7.51c}$$

$$-2e \leq v(t) \leq 2e, \quad t \in [0,4], \tag{7.51d}$$

$$-10e \leq u(t) \leq 10e, \quad t \in [0,4], \tag{7.51e}$$

$$q(4) = q_{\text{final}}, \tag{7.51f}$$

with $q \in \mathbb{R}^2, v \in \mathbb{R}^2, u \in \mathbb{R}^2$ and $x = (q, v)$. The switching functions are defined as $c_1^1(x) = q_1 + 0.15q_2^2$, $c_1^2(x) = -0.05q_1^3 + q_2$ and the function $c(x) = (c_1^2(x), c_1^2(x))$ defines the region boundaries. Note that we have two subsystems, cf. Section 6.2.5. It can be seen that for $v(t) = 0$ the vector fields of q point in all regions towards the origin in the (q_1, q_2) plane. Setting the control functions to zero, results in trajectories going to the origin with sliding modes on the surfaces of discontinuity defined by $c(x) = 0$. On the other hand, by increasing the value of $v(t)$ via the control functions $u(t)$, the vector fields can change their direction, and sliding modes can be left or not achieved at all.

The goal in the OCP is to reach $q_{\text{final}} = (-\frac{\pi}{6}, -\frac{\pi}{4})$ with a minimum control effort. The trajectory has to: (1) first reach $\varphi_1(x) = 0$; (2) slide towards $\mathcal{M} = \{q \in \mathbb{R}^2 \mid c(x) = 0\}$; (3) stay there for some time; (4) exit $\mathcal{M}$ and slide on $\varphi_2(x) = 0$; (5) and then leave the sliding mode as late as possible to reach q_{final}. The seemingly simple example comprises several difficult switching cases in its solution. The described solution is illustrated in Figure 7.10, and the optimal controls $u(t)$ and state $v(t)$ are depicted in Figure 7.11. The OCP is discretized with the FESD Radau IIA method of order 3 (i.e., $n_\text{s} = 2$), $N = 6$ control intervals and with $N_\text{FE} = 6$ finite elements on every control interval.

This solution comprises four switches and different sliding modes on nonlinear manifolds, including twice sliding on co-dimension one manifolds and once on the co-dimension two manifold $\mathcal{M}$, and additionally leaving the sliding mode twice. One can see in Figure 7.11 that the control is discretized on an equidistant grid despite the variable length of the finite elements. This illustrates the multiple shooting-type discretizations described in the previous section.

In the first experiment, we demonstrate the effects of step equilibration for a solution of the OCP (7.51). The left plot in Figure 7.12 depicts the indicator

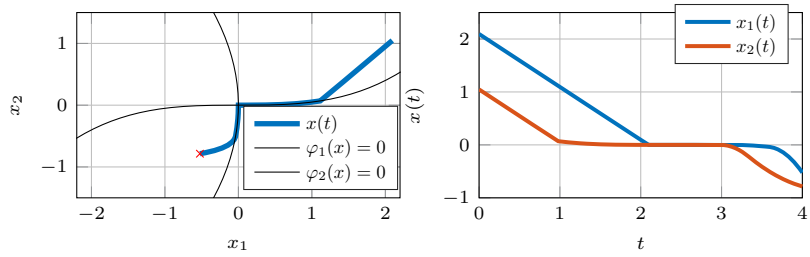


Figure 7.10: A solution $x(t)$ to the OCP (7.51).

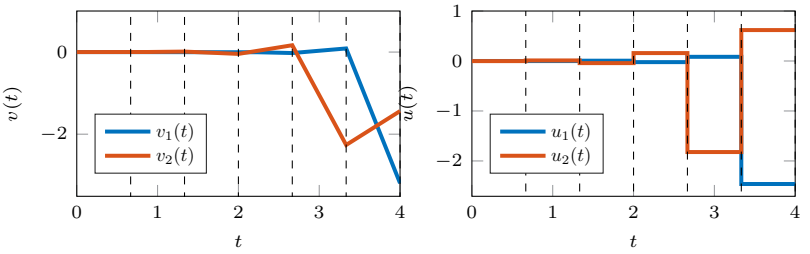


Figure 7.11: The left plot shows the solution trajectories for $v(t)$. The right plot shows optimal controls obtained via the FESD discretization of the OCP (7.51). The vertical dashed lines highlight the control discretization grid.

function η. The function is only zero if a switch occurs, cf. the right plot of Figure 7.10. The resulting step sizes $h_{k,n}$ with and without step equilibration are depicted in the middle and right plots, respectively. For the right plot we discard the step equilibration conditions $G_{\text{eq}}(\mathbf{h}, \mathbf{\Theta}, \mathbf{\Lambda}) = 0$. Obviously, without them, the optimizer varies the step size in a somewhat random way. On the other hand, with step equilibration, we obtain a piecewise equidistant grid, where the step-size changes only when a switch happens.

In the second experiment, we compare the accuracy of an OCP solution obtained with the standard and FESD method as a function of the CPU time. We take the optimal controls and perform a high-accuracy simulation of the system dynamics in (7.51), which we denote by $x_{\text{int}}(t)$. As a metric, we take the terminal constraint satisfaction of the high accuracy solution, i.e., $E(T) = \|x_{\text{int}}(T) - x_{\text{final}}\|$. We set $N = 6$, and vary the number of finite elements per stage N_{FE} from 1 to 7, and the number of stage points n_{s} from 1 to 4. The experiment is performed for the following RK methods: Radau IIA, Gauss-Legendre, Lobatto IIIC, and Explicit-RK.

For Radau IIA-FESD and Lobatto IIIC-FESD we solved the arising MPCC

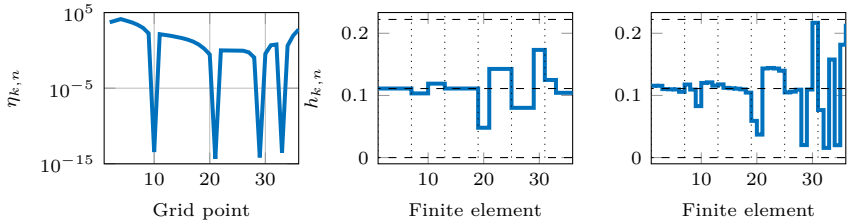


Figure 7.12: The left plot depicts the switching indicator function $\eta(\cdot)$ at the solution of the OCP (7.51). The middle plot shows the step size $h_{n,k}$ with step equilibration. The right plot shows the step sizes without step equilibration. The horizontal dashed lines correspond to the minimum, maximum, and nominal step size. The vertical dotted lines correspond to control interval boundaries.

with an elastic mode homotopy approach [16], cf. Section 2.4.3. In the other scenarios, we were not able to solve all problems to convergence with the elastic mode approach. Therefore, the MPCC was solved with a Scholtes relaxation homotopy approach, cf. Section 2.4.2. The second approach is slightly slower than elastic mode but more robust, and all problems were solved successfully. The terminal error as a function of the total CPU time is given in Figure 7.13. The source code of all examples is available in the repository of the open source tool `nosnoc` [2].

We can draw several conclusions from the experiments. The FESD method outperforms the standard approach in all experiments. For example, for a CPU time of ≈ 1 second with FESD, we achieve five orders of magnitude more accurate solutions than with the standard approach. A better solution than the most accurate one of the standard approaches can be achieved with FESD by an order of magnitude faster CPU time. The Radau IIA and Lobatto IIIC are the most efficient methods in this benchmark, whereas the Gauss-Legendre and Explicit-RK methods perform poorly. This is no surprise since the solution trajectories contain sliding mode arcs which require solving nonlinear DAE of index 2. Radau IIA and Lobatto IIIC usually perform well and have good theoretical properties for higher index DAE, whereas Gauss-Legendre and Explicit RK even loose the high accuracy orders that they have for ODE, cf. Section 3.2.

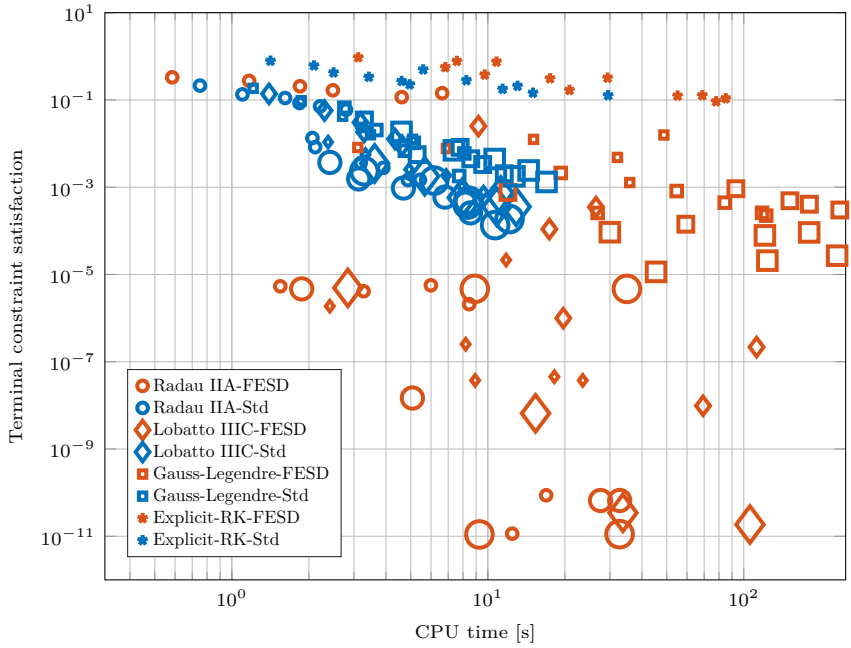


Figure 7.13: Terminal constraint satisfaction vs. CPU time for the Standard and FESD method. The size of the marker indicates the number of stage points, the smallest corresponds to $n_s = 1$ and the largest to $n_s = 4$.

7.6 Conclusions and summary

This chapter presents the Finite Elements with Switch Detection (FESD) method, which enables direct optimal control of a broad class of nonsmooth dynamical systems with high simulation accuracy. The method is developed for both Stewart's and the Heaviside step reformulation of Filippov systems into dynamic complementarity systems, which were discussed in Chapter 6.

With FESD, we can:

1) precisely determine the time of reaching or leaving the region boundaries, which is crucial for the high accuracy of integration methods,

2) exactly compute the sensitivities across regions to correctly address the nonsmoothness,

3) and appropriately handle the possible evolution on region boundaries that occurs in sliding modes.

It is important to note that it is not known beforehand whether an ODE or a DAE needs to be treated on a specific time interval with a fixed active set. Both cases are handled using Runge-Kutta methods within FESD to provide high-accuracy solutions. A key component of FESD is the automatic detection of the active set and switching time, which allows for the unified treatment of sliding modes and crossings of region boundaries.

We have established a solid theoretical foundation for FESD and have proven that it has the same accuracy as the underlying RK method for smooth differential equations. Additionally, we demonstrate that FESD delivers correct numerical sensitivities, which overcomes the fundamental limitations of time-stepping-based direct methods discussed in Chapter 5. An implementation of the FESD method described here is available in the open-source package `nosnoc`. Compared to standard discretization methods, FESD typically yields solutions that are several orders of magnitude more accurate for a similar amount of CPU time.

Another significant advantage of the new approach for direct optimal control is that guessing of the number or order of switches is not required. FESD can handle multiple or simultaneous switches and sliding modes. With time-freezing that we introduce in the two subsequent chapters, several classes of nonsmooth dynamical systems with state jumps can be recast into a piecewise smooth system, allowing for a unified treatment of various classes of nonsmooth systems in direct optimal control.

Future work includes relaxing some of the possibly restrictive assumptions in the theoretical analysis, and exploring open questions such as the following: Are all limit points of the solution approximations indeed solutions to the Filippov DI? What are the conditions under which the FESD problem has a nonempty and possibly compact solution set, and if unique Filippov solutions imply a unique solution to the corresponding discrete-time FESD problem? It has been shown in [137] that discrete-time optimal control problems with linear complementarity systems (which are MPCCs) are quite regular (i.e., all stationary points are strongly stationary points, cf. Definition 2.29). A natural question is if MPCCs obtained via the FESD discretization have any special properties and are they more regular than generic MPCCs?

Chapter 8

The Time-Freezing Reformulation for Nonsmooth Mechanical Systems

In this chapter, we regard rigid bodies subject to unilateral constraints and friction. They are one of the largest and most studied class of problems with state jumps, whit many practical applications in robotics. When two bodies collides, due to the rigidity assumption, the velocities must jump to avoid interpenetration and deformation. This greatly complicates formulating and solving optimal control problems with such systems.

These difficulties motivate the development of the time-freezing reformulation in this chapter. With time-freezing, we reformulate nonsmooth rigid body models into equivalent ODEs with a discontinuous right-hand, whose solutions do not have state jumps anymore. The main idea is to introduce an auxiliary differential equation to mimic the state jump map in the part of the state space that is infeasible for the original system. Additionally, we introduce a clock state, which does not evolve during the runtime of the auxiliary system. The pieces of the trajectory where the clock state evolves recover the solution of the original system with jumps. The resulting time-freezing system is a nonsmooth ODE with a discontinuity in the first time derivative of the trajectory, rather than in the trajectory itself. This enables us to seamlessly apply the FESD method from Chapter 7. Furthermore, this helps to overcome the fundamental computational limitations of time-stepping-based direct methods. In summary, it allows for a unified algorithmic treatment of several different classes of nonsmooth dynamical

systems in direct optimal control.

Outline. The chapter is structured as follows. Section 8.1 defines Complementarity Lagrangian Systems (CLS) and discusses related work. Section 8.4.1 illustrates the main ideas of time-freezing on a guiding example. In Section 8.3, we develop the time-freezing reformulation for elastic impacts, and in Section 8.4 for inelastic impacts with friction. Next, in Section 8.5, we formulate an Optimal Control Problem (OCP) subject to time-freezing systems, and relate it to the OCP subject to the initial CLS. Section 8.6 provides several numerical optimal control problems with time-freezing and FESD. This chapter is mainly based on the articles [202] and [212].

8.1 Introduction

The moment of contact between two bodies happens at such a short time scale that the modeling of all physical circumstances that lead to these fast dynamics is difficult. An accurate model of elastic body deformation requires partial differential equations. However, estimating all relevant parameters can be difficult in practice. The short time scale involved in impact events leads to very stiff systems, which makes their numerical treatment just as challenging as using a nonsmooth model. Moreover, compared to other motions, the duration of an impact is often beyond the resolution of practical sensors [49, 129].

A practical and successful approach is to assume rigidity, meaning that the bodies cannot change their shape. Consequently, when a rigid body collides with another body or an obstacle, the normal contact velocity must change instantaneously to maintain geometric feasibility. This is often a good modeling approximation that fits the needs of most applications such as robotic locomotion and manipulation, the study of human and animal gait, and granular matter [49, 255].

Remarkably, the macroscopic event of an impact is fully described by only two parameters: the coefficients of restitution and friction. The value of the post-impact velocity is determined by *restitution laws*, i.e., algebraic point-wise state jump laws. We distinguish between *elastic impacts*, i.e., the normal velocity after impact is positive, and *inelastic impacts* where the post-impact normal velocity is zero. Another crucial aspect of nonsmooth mechanics models is dry friction. The most widely used is Coulomb's friction model [49, 198, 255], which introduces another discontinuity. In a simple planar case, the friction force is always directed opposite to the tangential velocity. When the body is at rest, there is a set of possible forces.

Complementarity conditions offer an effective way of expressing all described nonsmooth effects. When combined with the equations of motion for an unconstrained rigid body, they give rise to a specific type of dynamic complementarity system known as a Complementary Lagrangian System (CLS). Since the velocity of a body changes due to the action of forces, an instantaneous change in velocity is a consequence of an impulsive force. This complicates the mathematical study of nonsmooth mechanical systems as the forces become distributions or measures [49, 198, 255]. Accordingly, the equations of motion are no longer ODEs but measure differential inclusions, which are discussed in Section 4.3.2.

It is generally unknown when and how often state jumps will occur, making the simulation or incorporation of these models into optimal control challenging. Section 5.3 has already revealed that standard time-stepping methods for these systems have fundamental limitations. The previous chapter introduced numerical methods for efficiently handling ODEs with a discontinuous right-hand side but these methods cannot be applied to systems with discontinuous solution trajectories. In this chapter, we introduce an exact reformulation of CLS into piecewise smooth systems, such that we can apply FESD.

Notation summary

We recall some of the most important notation conventions used in this chapter for the reader's convenience. Key symbols and definitions used in this chapter are summarized in Table 8.1. Time derivatives of a function $x(t)$ w.r.t. to the physical time t are compactly denoted by $\dot{x}(t) \coloneqq \frac{\mathrm{d}x(t)}{\mathrm{d}t}$, and of a function $y(\tau)$ w.r.t. to the numerical time τ by $y'(\tau) \coloneqq \frac{\mathrm{d}y(\tau)}{\mathrm{d}\tau}$. For the left and the right limits, we use the notation $x(t_{\mathrm{s}}^{+}) = \lim_{t \to t_{\mathrm{s}},\ t > t_{\mathrm{s}}} x(t)$ and $x(t_{\mathrm{s}}^{-}) = \lim_{t \to t_{\mathrm{s}},\ t < t_{\mathrm{s}}} x(t)$, respectively.

Table 8.1: Key symbols used throughout this chapter.

Symbol	Description	First appearance
n_x	dimension of the differential state x	Sec. 8.1.1
n_q	dimension of the states q and v	Sec. 8.1.1
n_u	dimension of the controls	Sec. 8.1.1
n_y	dimension of the time-freezing state	Sec. 8.4.1
n_{t}	dimension of the tangent space	Sec. 8.1.1
x	differential state	Sec. 8.1.1.
q	*position* state	Eq. (8.1)
v	*velocity* state	Eq. (8.1)

v_{t}	tangential *velocity* at contact points	Sec. 8.4.3
u	control function	Eq. (8.1)
λ_{n}	Lagrange multiplier, normal contact force	Eq. (8.1)
λ_{t}	Lagrange multiplier, friction force	Eq. (8.1)
y	extended state of time-freezing system	Secs. 8.3.1, 8.4.1
t	physical time	Secs. 8.3.1, 8.4.1
τ	numerical time	Secs. 8.3.1, 8.4.1
a_{n}	constant of the auxiliary dynamics	Prop. 8.8
a_{t}	constant of the auxiliary dynamics	Eq. (8.43)
λ^{n},λ^{p}	dual variables in step reformulation	Eq. (8.56)
α	selection of set-valued step function	Eq. (8.56)
μ	coefficient of friction	Eq. (8.1f)
ϵ_{t}	relaxation parameter in friction model	Eq. (8.52)
s	speed of time control variable	Eq. (8.58)
$\tilde{f}_v(q,v,u)$	all forces that act on the body	Eq. (8.1)
$f_v(q,v,u)$	vector field of the *velocity* state	Eq. (8.1)
$n(q)$	normal to the CLS constraint surface	Eq. (8.1)
$M(q)$	inertia matrix	Eq. (8.1)
$f_c(q)$	constraint function in the CLS	Eq. (8.1)
$B(q)$	matrix whose column span the tangent space at contact points	Eq. (8.1)
$b_j(q)$	j-th column of B	Eq. (8.1)
$f_{\mathrm{ODE}}(x,u)$	unconstrained dynamics when $f_c(q)>0$	Sec. 8.3.1, Eq. (8.18)
$D(q)$	Delassus' matrix/scalar	Eq. (8.25a)
$\varphi(x,u)$	determines if contact persists	Eq. (8.25b)
$f_{\mathrm{DAE}}(x,u)$	dynamics equivalent to the DAE (8.20) (constrained dynamics when $f_c(q)=0$)	Eq. (8.27)
$c_i(y)$	time-freezing PSS switching functions	Sec. 8.3.1, Eq. (8.28) and Sec. 8.4.3, Eq. (8.44)
$f_{\mathrm{aux,n}}(y)$	auxiliary dynamics	Defs. 8.2 and 8.6
$\gamma(x,u)$	time-rescaling factor of the time-freezing sliding mode	Eq. (8.32)
$f^{-}_{\mathrm{aux,t}}(y)$	auxiliary dynamics for tan. directions	Eq. (8.43)
$f_{\mathrm{Slip}}(x,u)$	dynamics for slipping motion in contact phases	Eq. (8.40)
$f_{\mathrm{Stick}}(x,u)$	dynamics for sticking motion in contact phases	Eq. (8.42)
$\tilde{D}(q)$	generalization of $D(q)$	Eq. (8.41)
$\tilde{\varphi}(x,u)$	generalization of $\varphi(x,u)$	Eq. (8.41)
$g_{\mathrm{F}}(\theta,\alpha)$	expression for relating θ and α	Eq. (8.57)

$\Psi(x(T))$	terminal cost	Eq. (8.53)
$g(x,u)$	path and terminal constraints	Eq. (8.53)
$r(x)$	terminal constraints	Eq. (8.53)
Σ	switching surface	Sec. 8.4.1
$F_{\mathrm{TF}}(y,u)$	Filippov set for the time-freezing system	Def. 8.3, 8.7 and 8.13
Q	region where all auxiliary dynamics are defined	Eq. (8.44)

8.1.1 Complementarity Lagrangian systems

There are various ways to represent nonsmooth Lagrangian systems, e.g., as second-order sweeping process [49, 198, 199], measure differential inclusion [49, 255, 258], hybrid system [156, 186], differential variational inequality [255, 258] or dynamic complementarity system [4, 49, 258]. The cited literature provides comprehensive theoretical and computational results for these systems. Here the focus is on the DCS formulation of nonsmooth mechanical systems, which is more specifically called a Complementarity Lagrangian System (CLS). The nonsmoothness in CLS is explicitly available through the complementarity constraints, and not hidden in more abstract set-valued mappings. This facilitates the development of the time-freezing reformulation and numerical methods.

The complementarity Lagrangian System equations with friction read as:

$$\dot{q} = v, \tag{8.1a}$$

$$M(q)\dot{v} = \tilde{f}_{\mathrm{v}}(q,v,u) + \sum_{i=1}^{n_{\mathrm{c}}} (n_i(q)\lambda_{\mathrm{n,i}} + B_i(q)\lambda_{\mathrm{t,i}}), \tag{8.1b}$$

$$0 \le \lambda_{\mathrm{n}} \perp f_c(q) \ge 0, \tag{8.1c}$$

$$\begin{aligned} &\text{if } f_{c,i}(q(t_{\mathrm{s}})) = 0 \text{ and } n_i(q(t_{\mathrm{s}}))^\top v(t_{\mathrm{s}}^-) < 0 \text{ then} \\ &0 = n_i(q(t_{\mathrm{s}}))^\top (v(t_{\mathrm{s}}^+) - \epsilon_{\mathrm{r}} v(t_{\mathrm{s}}^-)),\ i = 1,\dots,n_{\mathrm{c}} \end{aligned} \tag{8.1d}$$

$$\lambda_{\mathrm{t,i}} \in \arg\min_{\tilde{\lambda}_{\mathrm{t},i} \in \mathbb{R}^{n_{\mathrm{t}}}} \quad -v^\top B_i \tilde{\lambda}_{\mathrm{t},i} \tag{8.1e}$$

$$\text{s.t.} \quad \|\tilde{\lambda}_{\mathrm{t},i}\|_2 \le \mu \lambda_{\mathrm{n,i}},\ i = 1,\dots,n_{\mathrm{c}}. \tag{8.1f}$$

We express the state of the rigid body through generalized coordinates $q(t) \in \mathbb{R}^{n_q}$ and generalized velocities $v \in \mathbb{R}^{n_q}$. The matrix $M(q)$ is the inertia matrix and

is assumed to be symmetric positive definite. The function $\tilde{f}_{\mathrm{v}} : \mathbb{R}^{n_q} \times \mathbb{R}^{n_q} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_q}$ collects all generalized forces that act on the body. For ease of notation, in the development of the time-freezing reformulation, we multiply the velocity dynamics by $M(q)^{-1}$ and introduce the shorthand $f_v(q, v, u) \coloneqq M(q)^{-1}\tilde{f}_v(q, v, u)$.

Equation (8.1c) is the Signorini contact condition. We denote by $n_i(q) = \nabla_q f_{c,i}(q)$ the contact normal of the i-th contact and collect all contact normal in the matrix $N(q) = \begin{bmatrix} n_1(q), \dots, n_{n_c}(q) \end{bmatrix} \in \mathbb{R}^{n_q \times n_c}$. If the i-th constraint is active, meaning $f_{c,i}(q) = 0$, a Lagrange multiplier $\lambda_{\mathrm{n,i}}$ is introduced to ensure the constraint remains feasible. The multiplier $\lambda_{\mathrm{n,i}}$ represents a generalized normal contact force that acts in the direction of the contact normal $n_i(q)$ and maintains $f_{c,i}(q) \geq 0$. We assume that the contacts do not involve adhesion, i.e., there is no gluing between bodies. If $f_{c,i}(q) > 0$, there is no contact force and $\lambda_{\mathrm{n,i}} = 0$. The contact forces are grouped in the vector $\lambda_{\mathrm{n}} = (\lambda_{\mathrm{n},1}, \dots, \lambda_{\mathrm{n},n_c}) \in \mathbb{R}^{n_c}$.

Furthermore, when a constraint becomes active, due to the rigidity hypothesis, the negative normal velocity $n_i(q)^\top v$ must become instantaneously nonnegative. At this point, there are infinitely many choices for the post-impact velocity. This ambiguity is resolved with a restitution law, which computes the post-impact velocity as a function of the pre-impact velocity. Here Newton's impact law is used, which is given in Eq. (8.1d), where t_{s} is the time of impact and $\epsilon_{\mathrm{r}} \in [0, 1]$ is the coefficient of restitution. If $\epsilon_{\mathrm{r}} \in (0, 1]$, we have (partially) elastic impacts and the constraint becomes inactive after the impact, i.e., $f_{c,i}(q(t)) > 0$ for $t > t_{\mathrm{s}}$. On the other hand, for $\epsilon_{\mathrm{r}} = 0$ we obtain $f_{c,i}(q(t)) = 0$ for $t > t_{\mathrm{s}}$ and the dynamics are subject to $f_{c,i}(q) = 0$ and thus we obtain locally a DAE. These two cases lead to qualitatively different behavior and the corresponding time-freezing reformulations are somewhat different.

The last ingredient in the CLS model is the nonsmooth Coulomb friction model, which is given in (8.1e)-(8.1f). Before we explain the model, we define the functions used in it. In the three-dimensional case, the tangent space at the i-th contact point $\{q \in \mathbb{R}^{n_q} \mid f_{c_i}(q) = 0\}$ is spanned by the columns of the matrix $B_i(q) = \begin{bmatrix} b_{i,1}(q) & b_{i,2}(q) \end{bmatrix} \in \mathbb{R}^{n_q \times n_{\mathrm{t}}}$, with $n_{\mathrm{t}} = 2$. In two dimensions, it is spanned by a single vector, which we also denote by $B_i(q) \in \mathbb{R}^{n_q \times n_{\mathrm{t}}}$, with $n_{\mathrm{t}} = 1$. We denote the tangential velocity at the i-th contact by $v_{\mathrm{t,i}} = B_i(q)^\top v$. The scalar $\mu_i > 0$ is the coefficient of friction for the i-th contact.

The optimization problem (8.1e)-(8.1f) expresses that the dissipation of the kinetic energy between two objects in contact is maximized. Consequently, the friction law has the following features:

- the friction force has the opposite direction to the tangential slip direction $v_{\mathrm{t,i}}$ and

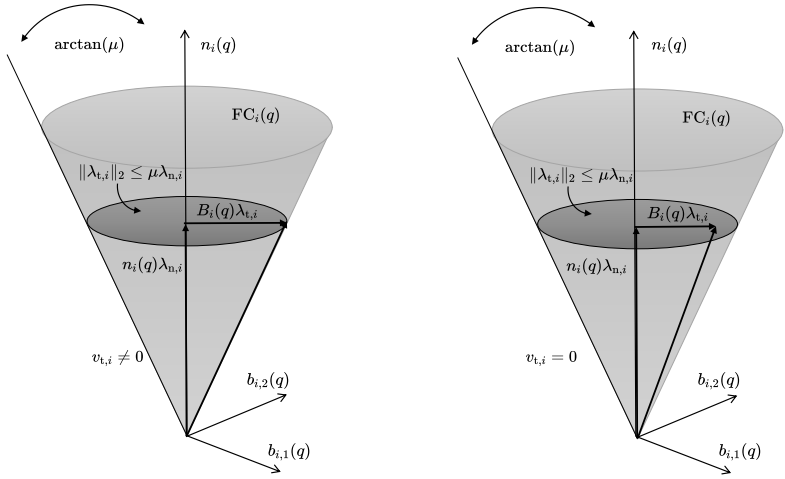


Figure 8.1: Illustration of the three-dimensional friction cone $\mathrm{FC}_i(q)$ for slip motions (left figure) and stick motions (right plot).

- the maximal magnitude of the friction force $\lambda_{\mathrm{t,i}}$ is μ_i (the coefficient of friction) times the normal contact force $\lambda_{\mathrm{n,i}}$.

The set of all possible contact forces at the i-th contact is called the friction cone and is defined as

$$\mathrm{FC}_i(q) = \{n_i(q)\lambda_{\mathrm{n,i}} + B_i(q)\lambda_{\mathrm{t,i}} \mid \lambda_{\mathrm{t,i}} \geq 0, \|\lambda_{\mathrm{t,i}}\|_2 \leq \mu\lambda_{\mathrm{n,i}}\}.$$

The total friction cone is the sum of all friction cones generated by each contact:

$$\mathrm{FC}(q) = \sum_{i \in \{i \mid f_{c,i}(q)=0\}} \mathrm{FC}_i(q).$$

In the presence of friction, a rigid body can be either in slip or stick mode. During slip mode, the friction force takes its maximal magnitude but is still less than (or equal to) the magnitude of the sum of all other forces acting in the same plane. In stick mode, the friction force is greater or equal to the sum of the other forces acting in the same plane, and the body has a zero tangential velocity w.r.t. the regarded contact. Figure 8.1 illustrates friction cones for both the case of slip and stick motions.

8.1.2 Related work

To address the challenges posed by state jumps, various transformations have been proposed in the literature to achieve the same outcome as time-freezing. Two widely used methods are: (a) Coordinate transformations (as in [164, 294]), and (b) Smoothing/penalization (as in [263]) or compliant impact models [49].

Coordinate transformations can be effective for specific situations. For example, the Zhuravlev-Ivanov transformations [49, Sec. 1.4.3] are limited to mechanical systems with a single unilateral constraint. A more comprehensive approach involves using "gluing functions" within the hybrid systems framework [164], but this method only applies to systems with a single constraint and no generic method for finding the required gluing function exists.

Smoothing or penalization can lead to realistic approximations, but simulating the resulting stiff differential equations can be difficult. Compliant models may result in unrealistic effects or underestimated velocities [49, Sec. 2.2]. Smoothing leads to very stiff ODEs and standard numerical methods can perform similarly as if they were directly applied to the original nonsmooth system, cf. Section 4.2. To obtain more accurate solutions and maintain stability in numerical integration for smooth ODEs, prohibitively small step sizes are necessary. Sometimes artificial damping and changes in the inertia matrix are introduced to keep the integrator stable. However, all these model changes introduce additional errors, which might not be easy to quantify [199].

The time-freezing reformulation was initially introduced in [212] for rigid bodies with partially elastic impacts, and it was subsequently extended to the inelastic case in [202]. Remarkably, this very same idea was independently introduced later by Halm and Posa in [129]. In their research, the time-freezing reformulation is used to obtain a differential inclusion with a bounded right-hand side. This enabled the application of standard solution existence results for differential inclusions, cf. Section 4.3.2. They address the nonuniqueness in simulating rigid body models with multiple frictional impacts through a stochastic approach by randomly varying auxiliary dynamics parameters and considering multiple possible outcomes. However, [129] was not focused on developing numerical methods for the whole time-freezing systems, which is a nontrivial task. They simulate the auxiliary dynamics to compute the different outcomes that may happen during simultaneous impacts. In contrast to their work, the focus in this thesis is on developing high-accuracy numerical simulation and direct optimal control methods for time-freezing systems that are equivalent to CLS.

Due to the nonsmoothness and presence of inequality constraints standard numerical methods for smooth ODEs and DAEs cannot be applied directly

to a CLS. Instead, tailored time-stepping methods for are necessary. Most time-stepping methods for CLS are based on the (semi-)implicit Euler method and require solving a Linear Complementarity Problem (LCP) at each time step. These methods are known for their stability, ease of use, and ability to handle a large number of contacts [4, 154, 153, 198, 199, 15, 255]. However, they are limited to first-order accuracy, even in the absence of contacts. One of the first nonsmooth numerical methods in this class is the Jean-Moreau time-stepping scheme [154, 153, 198, 199]. It treats the unilateral constraints on velocity level and uses Newton's impact law. The Schatzman–Paoli [221, 222] scheme deals directly with the inequality constraints at the position level. Similar widely used methods are the Anitescu-Potra method (at velocity level) [15] and the Stewart-Trinkle method (at position level) [260]. A general convergence result of time-stepping methods was provided by Stewart in [254]. Several modifications have been made to these methods, such as those to handle stiff systems [14] or to improve computational efficiency by solving convex quadratic programs instead of LCPs [13]. A comprehensive review of time-stepping methods for CLS can be found in [261]. Event-driven methods, which are less commonly used in the simulation of rigid bodies, are surveyed in [4]. However, they are not practical for direct optimal control problems due to their external switch-detection procedure. For an overview of direct methods for optimal control problems subject to CLS, cf. Section 5.1.

8.2 The time-freezing reformulation

In this section, we introduce a guiding example on which we illustrate the main ideas in the time-freezing reformulation.

8.2.1 A guiding example

Example 8.1 (Guiding example)**.** *Consider a frictionless point mass in two dimensions above a horizontal table. The mass is* $m = 1$ kg *and* $g = 9.81\,\mathrm{m/s^2}$ *is the gravitational acceleration. Denote by* $q(t) \coloneqq (q_1(t), q_2(t))$ *and* $v(t) \coloneqq (v_1(t), v_2(t))$ *its position and velocity, respectively, and let* $\lambda_\mathrm{n}(t)$ *be the normal contact force. The dynamics are given by the CLS:*

$$\dot{q}(t) = v(t), \tag{8.2a}$$

$$m\dot{v}(t) = \begin{bmatrix} 0 \\ -mg \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \lambda_\mathrm{n}(t) + \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix}, \tag{8.2b}$$

$$0 \leq \lambda_{\mathrm{n}}(t) \perp q_2(t) \geq 0, \tag{8.2c}$$

$$v_2(t_{\mathrm{s}}^+) = -\epsilon_{\mathrm{r}} v_2(t_{\mathrm{s}}^-), \text{ if } q_2(t_{\mathrm{s}}) = 0 \text{ and } v_2(t_{\mathrm{s}}^-) < 0. \tag{8.2d}$$

where $u = (u_1, u_2) \in \mathbb{R}^2$ is an externally chosen thrust force, which shall be found, e.g., by solving an optimal control problem. The complementarity condition (8.2c) *states: the point-mass is either not in contact ($q_2 > 0$) and there is no reaction force ($\lambda_{\mathrm{n}} = 0$), or there is contact ($q_2 = 0$) and a normal reaction force ($\lambda_{\mathrm{n}} \geq 0$). If the particle hits the table ($q_2(t_{\mathrm{s}}) = 0$) with a negative normal velocity ($v_2(t_{\mathrm{s}}^-) < 0$), then the normal velocity must jump to a positive value (elastic impact, with $\epsilon_{\mathrm{r}} > 0$) or become zero (inelastic impact, with $\epsilon_{\mathrm{r}} = 0$) to not violate the constraint $q_2 \geq 0$. The point-wise state jump law* (8.2d) *determines the velocity after an impact, with $\epsilon_{\mathrm{r}} \in [0, 1]$ being the coefficient of restitution.*

It is important to note that the jump discontinuities are state-dependent, i.e., they occur at time points that are not known a priori. In the sequel, we take a closer look at the qualitative difference between elastic and inelastic impacts.

Elastic impacts

For partially elastic impacts, we have $\epsilon_{\mathrm{r}} \in (0, 1]$, which means that the post-impact velocity is strictly positive if the pre-impact velocity is nonzero. Consequently, the particle does not stay on the table but bounces back. For $f_c(q) \coloneqq q_2 > 0$, we have no contact force, and the particles' time evolution is described by the ODE:

$$\dot{q}(t) = v(t),$$

$$m\dot{v}(t) = \begin{bmatrix} 0 \\ -mg \end{bmatrix},$$

which we compactly denote by $\dot{x} = f_{\mathrm{ODE}}(x)$ (free flight). If $f_c(q) = 0$, then λ_{n} is a Dirac impulse that ensures $v_2(t_{\mathrm{s}}^+) = -\epsilon_{\mathrm{r}} v_2(t_{\mathrm{s}}^-)$. We can summarize the dynamics of the particle in the case of elastic impacts as:

$$\dot{x}(t) = f_{\mathrm{ODE}}(x(t)), \qquad \text{if } f_c(q(t)) > 0,$$

$$v_2(t_{\mathrm{s}}^+) = -\epsilon_{\mathrm{r}} v_2(t_{\mathrm{s}}^-), \qquad \text{if } f_c(q(t_{\mathrm{s}})) = 0,\ \nabla f_c(q(t_{\mathrm{s}}))v(t_{\mathrm{s}}^-) < 0.$$

The plots in the left column of Figure 8.2 show an example geometric trajectory, velocities, and a phase plot for the elastic impact case.

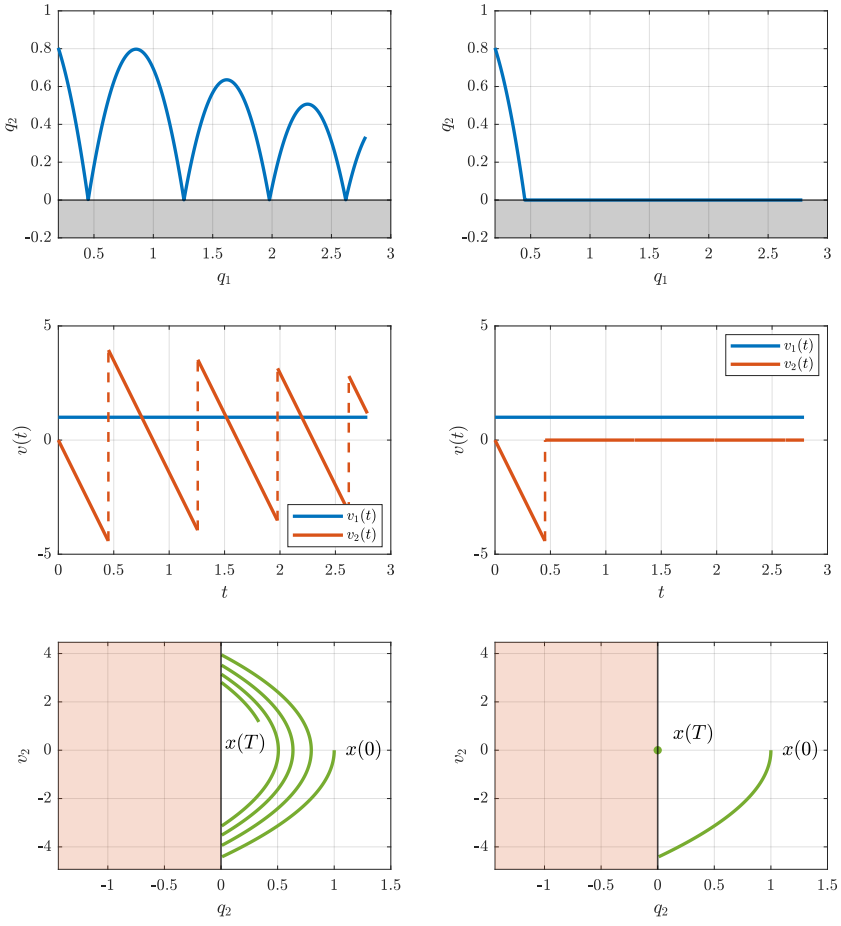


Figure 8.2: The left plots are for the elastic impact case and the right plots are for the inelastic case. The first row shows the geometric trajectories of the particle in the (q_1, q_2) plane. The second row shows the velocities as a function of time. The last row illustrates the phase plots in the (q_2, v_2) plane. The red shaded area corresponds to the infeasible region for the particle ($q_2 < 0$).

Inelastic impacts

With $\epsilon_r = 0$, we speak of inelastic impacts and obtain qualitatively different trajectories. In this case, after the particle hits the table, it stays on it, and the time evolution continues in the horizontal direction, i.e., the particle slides

on the table. The particles' post-impact motion is described by the DAE of index 3:

$$\begin{aligned} \dot{q}(t) &= v(t), \\ m\dot{v}(t) &= \begin{bmatrix} 0 \\ -mg \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \lambda_{\mathrm{n}}(t), \\ 0 &= q_2(t). \end{aligned}$$

By taking two times the total derivative of the constraint $q_2(t) = 0$, one can explicitly compute the normal contact force and obtain $\lambda_{\mathrm{n}} = mg$. Using this, instead of a DAE, we can regard an equivalent ODE, which we denote by $\dot{x} = f_{\mathrm{DAE}}(x)$. The dynamics of this system can be summarized as follows:

$$\begin{aligned} \dot{x}(t) &= f_{\mathrm{ODE}}(x(t)), && \text{if } f_c(q(t)) > 0, \\ v_2(t_{\mathrm{s}}^+) &= 0, && \text{if } f_c(q) = 0,\ \nabla_q f_c(q) v < 0, \\ \dot{x}(t) &= f_{\mathrm{DAE}}(x(t)), && \text{if } f_c(q(t)) = 0. \end{aligned}$$

Compared to the elastic impacts, here we have a more complicated switching behavior. Moreover, in the presence of external forces or more complicated gap functions $f_c(q)$, switches from the DAE (constrained mode) to the unconstrained ODE mode are also possible, cf. Section 8.4. The plots in the right column of Figure 8.2 show an example trajectory for the inelastic impact case.

8.2.2 Main ideas behind time-freezing

The time-freezing reformulation transforms dynamic systems with state jumps (NSD3) into piecewise smooth systems (NSD2), where no state jumps are present. It builds upon two main ideas:

1. The state jumps are mimicked by an *auxiliary dynamical system* $\dot{x} = f_{\mathrm{aux}}(x)$ in the *infeasible* region, i.e., $f_c(q) < 0$. The trajectory endpoints of the auxiliary ODE satisfy the state jump law on some finite time interval.
2. We introduce a *clock state* $t(\tau)$ that stops counting whenever the auxiliary dynamical system is active, i.e., $t'(\tau) = 0$ for $f_c(q) < 0$, and $t'(\tau) = 1$ for $f_c(q) > 0$.

Consequently, by taking the pieces of the trajectory when the clock state was active, we can recover the solution of the original system with discontinuous trajectories.

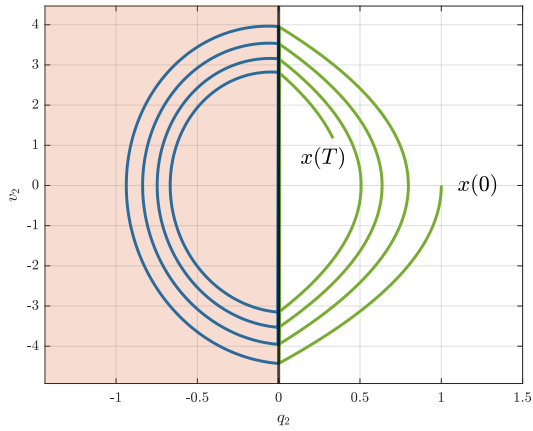


Figure 8.3: The phase plot of the time-freezing systems (8.3) in the (q_2, v_2) plane.

We illustrate this idea on the guiding example for the case of elastic impacts. Inelastic impacts are slightly more complicated, and we treat them in detail in Section 8.4. The auxiliary dynamics in the example are defined in $q_y < 0$. We extend the state space by the clock state t, which results in the differential state vector $y = (x, t) \in \mathbb{R}^{n_x+1}$. The unconstrained dynamics are augmented by the clock state and read as:

$$\frac{\mathrm{d}}{\mathrm{d}\tau} y(\tau) = f_1(y) \coloneqq \begin{bmatrix} f_{\mathrm{ODE}}(x(\tau)) \\ 1 \end{bmatrix}.$$

An auxiliary ODE for the example system is given by:

$$\frac{\mathrm{d}}{\mathrm{d}\tau} y(\tau) = f_2(y(\tau)) = \begin{bmatrix} 0 \\ v_y(\tau) \\ 0 \\ -kq_y(\tau) - cv_y(\tau) \\ 0 \end{bmatrix},$$

where $k, c > 0$ are appropriately chosen constant, such that the endpoints of its trajectory between two switches on the interval $[\tau_\mathrm{s}, \tau_\mathrm{r}]$ satisfy the state jump law: $v_2(\tau_\mathrm{r}) = -\epsilon_\mathrm{r} v_2(\tau_\mathrm{s})$. In Section 8.3, we will provide constructive ways to select auxiliary dynamics. Therefore, the overall time-freezing system reads as:

$$\frac{\mathrm{d}}{\mathrm{d}\tau} y(\tau) = \begin{cases} f_1(y(\tau)), & \text{if } f_c(q(\tau)) > 0, \\ f_2(y(\tau)), & \text{if } f_c(q(\tau)) < 0. \end{cases} \tag{8.3}$$

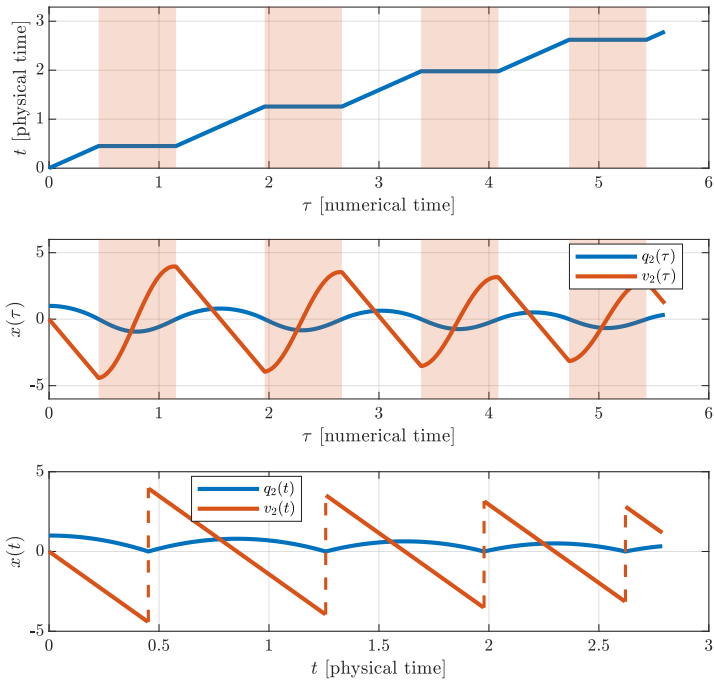


Figure 8.4: The top plot shows the evolution of the clock state $t(\tau)$ and the red-shaded area indicates time intervals when the time is frozen. The middle plot show the states $q_2(\cdot)$ and $v_2(\cdot)$ as a function of the numerical time τ and the bottom plot as a function of the physical time t.

The time of the time-freezing system τ is called *numerical time*. Figure 8.3 illustrates the phase plot of the time-freezing system. Note that in contrast to the bottom left plot in Figure 8.2, the trajectory evolves also in the red prohibited region. The top plot in Figure 8.4 show the evolution of the clock state $t(\tau)$. The intervals with $t'(\tau) = 1$ are referred to as *physical time* and those with $t'(\tau) = 0$ as *virtual time*. The red shaded area indicates the intervals when the time is frozen. In the middle plot, the trajectory of the time-freezing system in numerical time τ. Note that the trajectory is now a continuous function of numerical time τ. The bottom plot shows not the trajectory of the time-freezing system but is plotted over physical time $t(\tau)$. With this simple transformation, we recover the discontinuous trajectory of the original system with state jumps.

8.3 Time-freezing for elastic impacts

In this section, we present the time-freezing reformulation for elastic impacts for one impact at a time and without friction. The main ideas were already introduced in Section 8.2.2. Here we provide constructive ways to select auxiliary dynamics for a given coefficient of restitution $\epsilon_{\mathrm{r}} \in (0, 1]$. Moreover, we show the equivalence of solutions between the CLS and the time-freezing system.

8.3.1 The time-freezing system

We regard a CLS without friction, with a single unilateral constraint, i.e., $n_{\mathrm{c}} = 1$, and drop the subscript i for notational convenience. In this case, from (8.1) we obtain the CLS:

$$\dot{q}(t) = v(t), \tag{8.4}$$

$$\dot{v}(t) = f_{\mathrm{v}}(q(t), v(t), u(t)) + M(q(t))^{-1} n(q(t)) \lambda_{\mathrm{n}}(t), \tag{8.5}$$

$$0 \leq \lambda_{\mathrm{n}}(t) \perp f_c(q) \geq 0, \tag{8.6}$$

$$\begin{aligned} &0 = n(q(t_{\mathrm{s}}))^{\top} (v(t_{\mathrm{s}}^{+}) + \epsilon_{\mathrm{r}} v(t_{\mathrm{s}}^{-})), \\ &\quad \text{if } f_c(q(t_{\mathrm{s}})) = 0 \text{ and } n(q(t_{\mathrm{s}}))^{\top} v(t_{\mathrm{s}}^{-}) < 0. \end{aligned} \tag{8.7}$$

Such systems are sometimes called *vibro-impact systems* [49]. It can be seen that if $f_c(q) > 0$, then the dynamics reduce to the ODE:

$$\begin{aligned} \dot{q}(t) &= v(t), \\ \dot{v}(t) &= f_{\mathrm{v}}(q(t), v(t), u(t)), \end{aligned}$$

which we compactly denoted by $\dot{x}(t) = f_{\mathrm{ODE}}(x(t), u(t))$. When the constraint $f_c(q) \geq 0$ becomes active, and the normal velocity is negative, i.e., $n(q(t_{\mathrm{s}}))^{\top} v(t_{\mathrm{s}}^{-}) < 0$, the velocity is reinitialized according to the state jump law

$$0 = n(q(t_{\mathrm{s}}))^{\top} (v(t_{\mathrm{s}}^{+}) + \epsilon_{\mathrm{r}} v(t_{\mathrm{s}}^{-})) \tag{8.8}$$

If initialized with a nonzero normal velocity, the system is mostly in the ODE mode, and the velocity is reinitialized whenever contact takes place. There are two cases in which the regarded CLS can have an active constraint. Either it is initialized with $f_c(q(0)) = 0$ and $n(q(0))^{\top} v(0) = 0$, or when after infinitely many jumps, with $\epsilon_{\mathrm{r}} \in (0, 1)$, the normal velocity shrinks to zero, cf. Section

4.2.1. Here we focus on the interchange between the free flight phase modeled by $\dot{x}(t) = f_{\text{ODE}}(x(t), u(t))$ and the state jumps.

In the general case, the time of impact t_s is not known a priori. Simulating and incorporating such models with additional algebraic conditions into optimal control problems is difficult. To alleviate all these difficulties, we propose the following approach. First, we relax the constraint $f_c(q) \geq 0$ and define an auxiliary dynamical system on $f_c(q) < 0$ to mimic the restitution law (8.8); Second, we introduce a clock state $t(\tau)$ that evolves according to $t'(\tau) = 1$. The time τ denoted as *numerical time* is now the time of the differential equation. Third, we "freeze" the time whenever $x(\tau)$ is the infeasible region, i.e., $t'(\tau) = 0$.

The state space is augmented with the clock state

$$y(\tau) = \begin{bmatrix} q(\tau) \\ v(\tau) \\ t(\tau) \end{bmatrix} \in \mathbb{R}^{n_y},$$

with $n_y = n_x + 1$ and we regard all functions now over the numerical time τ. We define $c(y) = f_c(q)$ as the switching function of the PSS we wish to obtain. This splits the state space into the following two parts:

$$R_{\text{ODE}} = \{y \in \mathbb{R}^{n_y} \mid f_c(q) > 0\}, \tag{8.9}$$

$$R_{\text{aux}} = \{y \in \mathbb{R}^{n_y} \mid f_c(q) < 0\}, \tag{8.10}$$

where R_{ODE} is the *feasible region* and R_{aux} is the *prohibited region.* To mimic the restitution law, we define the auxiliary ODE, whose endpoints satisfy the restitution law on a finite time interval.

Definition 8.2 (Auxiliary ODE for elastic impacts)**.** *An auxiliary dynamical, for elastic normal velocity jumps, $y'(\tau) = f_{\text{aux,n}}(y(\tau))$ satisfies for every initial value $y(\tau_\text{s}) = (q_\text{s}, v_\text{s}, t_s)$, with $f_c(q_\text{s}) = 0$ and $n(q_\text{s})^\top v_\text{s} < 0$, a well-defined and finite time interval $[\tau_\text{s}, \tau_\text{r}]$, with the length $\tau_{\text{jump}} = \tau_\text{r} - \tau_\text{s}$, the following properties:*

(i) $f_c(q(\tau)) \leq 0$ *and* $t'(\tau) = 0$ *for all* $\tau \in [\tau_\text{s}, \tau_\text{r}]$,

(ii) $n(q(\tau_\text{r}))^\top v(\tau_\text{r}) = -\epsilon_\text{r} n(q(\tau_\text{s}))^\top v(\tau_\text{s})$, *and*

(iii) $f_c(q(\tau_\text{r})) = 0$.

The introduced ideas are collected in the definition of a time-freezing system for elastic impacts.

Definition 8.3 (Time-freezing system for elastic impacts)**.** *Let $\tau \in \mathbb{R}$ be the numerical time and $y(\tau) \coloneqq (x(\tau), t(\tau)) \in \mathbb{R}^{n_y}$ the differential states and $u(\tau) \in \mathbb{R}^{n_u}$ a given control function. The time-freezing PSS for a given is $f_{\mathrm{aux,n}}(y)$ is a PSS defined over the regions R_{ODE} and R_{aux} in* (8.9) *and:*

$$y' = \begin{cases} \begin{bmatrix} f_{\mathrm{ODE}}(x,u) \\ 1 \end{bmatrix}, & \text{if } y \in R_{\mathrm{ODE}}, \\ f_{\mathrm{aux,n}}(y), & \text{if } y \in R_{\mathrm{aux}}. \end{cases}$$

The corresponding Filippov system, which we call the time-freezing system, is defined as

$$y' \in F_{\mathrm{TF}}(y,u) \coloneqq \left\{ \theta_1 (f_{\mathrm{ODE}}(x,u), 1) + \theta_2 f_{\mathrm{aux,n}}(y) \mid e^\top \theta = 1, \theta \geq 0 \right\}, \tag{8.11}$$

with $\theta = (\theta_1, \theta_2)$.

Observe that we got rid of the conditional algebraic restitution law (8.8). The resulting time-freezing system is a nonsmooth ordinary differential equation where the discontinuity is in the first-time derivative of the trajectory rather than in the trajectory itself.

8.3.2 Auxiliary dynamics for elastic impacts

To make further use of Definition 8.3, we must specify how to construct an appropriate auxiliary ODE. We start with the simpler case of a linear constraint of the form of $f_c(q) = n^\top q$ and a constant inertia matrix $M(q)$. Afterwards, we extend this formulation to general nonlinear constraints and inertia matrices that are not necessarily constant.

The next proposition provides a constructive way of selecting the auxiliary ODE from Definition 8.2 for linear scalar constraints.

Proposition 8.4 (Auxiliary dynamics)**.** *Suppose that $f_c(q) = n^\top q$, where n is the constraint normal and the inertia matrix $M(q)$ is constant. Regard an initial value $y(\tau_{\mathrm{s}}) = (q_{\mathrm{s}}, v_{\mathrm{s}}, t_{\mathrm{s}})$ such that $f_c(q_{\mathrm{s}}) = 0$ and $n^\top v_{\mathrm{s}} \leq 0$ holds. The ODE is given by*

$$y' = f_{\mathrm{aux,n}}(y) \coloneqq \begin{bmatrix} n n^\top q \\ M(q)^{-1} n (-k n^\top q - c n^\top q) \\ 0 \end{bmatrix} \tag{8.12}$$

with $k > 0$ and

$$c = 2 |\ln(\epsilon_{\mathrm{r}})| \sqrt{\frac{k}{(\ln(\epsilon_{\mathrm{r}})^2 + \pi^2)}}, \tag{8.13}$$

is an auxiliary dynamical system from Definition 8.2 with

$$\tau_{\text{jump}} = \sqrt{\frac{(\pi^2 + \ln(\epsilon_{\text{r}})^2)}{k}}. \tag{8.14}$$

Proof. Without loss of generality, we assume that $\tau_{\text{s}} = 0$. In the linear case, the contact normal n is constant for all q. As the velocity jumps happen along the constraint normal n, the main idea is to project the position and velocity onto the normal (corresponds to the multiplication by $n^\top$ in (8.12)), use a two-dimensional linear spring-damper model and embed the result back (correspond to the multiplication by n and $M(q)^{-1}n$ in (8.12))). Let us introduce the change of variables $\tilde{q} = n^\top q$, $\tilde{v} = n^\top v$ and regard the two-dimensional ODE:

$$\tilde{q}' = \tilde{v},$$

$$\tilde{v}' = -k\tilde{q} - c\tilde{v},$$

with the initial conditions $\tilde{q}(0) = n^\top q(0) = f_c(0) = 0$ and $\tilde{v}(0) = n^\top v(0) \leq 0$. The analytic solution of this ODE is given by

$$\tilde{q}(\tau) = \frac{\tilde{v}(0)}{\omega} e^{\frac{-c\tau}{2}} \sin(\omega\tau),$$

$$\tilde{v}(\tau) = \tilde{v}(0) e^{\frac{-c\tau}{2}} \Big(-\frac{c}{2\omega} \sin(\omega\tau) + \cos(\omega\tau) \Big),$$

with $\omega = \frac{\sqrt{4k - c^2}}{2}$. Using this analytic expression, one can compute, that for a given $k > 0$ and $c > 0$ computed via (8.13), we have that

$$\tilde{v}(\tau_{\text{r}}) = -\epsilon_{\text{r}} \tilde{v}(0),$$

i.e., the state jump law is satisfied. Using the analytic solution, we can compute that $\tau_{\text{r}} - \tau_{\text{s}} = \tau_{\text{jump}}$ is given by (8.14). Moreover, it can be seen that $f_c(q(\tau)) = \tilde{q}(\tau) \leq 0$ for all $\tau \in [\tau_{\text{s}}, \tau_{\text{r}}]$. All conditions of Definition 8.2 are satisfied, and the proof is complete. □

The use of spring-damper systems to model mechanical impact is an old idea, which is discussed in Chapter 2 of [49]. However, to accurately represent rigid body impact dynamics, the system must become infinitely stiff, making it impractical for numerical computations. Additionally, spring-damper models may produce negative contact forces, as noted in Remark 2.3 of [49].

Here these difficulties are circumvented by introducing a clock state and freezing it. This allows us to use even relatively small values for the spring constant, represented by k, while still recovering the exact impact law in (8.8). We only

use the endpoints of the trajectory produced by (8.12) and discard all other pieces of the trajectory, thus avoiding the problems associated with standard compliant models. In the case of activation of multiple perfect frictionless constraints, the negative reaction force is in the normal cone to the feasible set at this point [4], and hence the auxiliary dynamic has to evolve in this part of the state space. In case the constraints are orthogonal, the desired vector field is simply the sum of the neighboring fields, otherwise, the analysis is more involved.

An example of a time-freezing system from Definition 8.3 with an auxiliary ODE from the last proposition was already derived in Subsection 8.2.2. The blue line in Figure 8.3 shows parts of the trajectory corresponding to $y' = f_{\text{aux,n}}(y)$ and the green line corresponding to $y' = (f_{\text{ODE}}(x,u),1)$. The trajectories as functions of time are given in Figure 8.4.

8.3.3 Auxiliary dynamics for nonlinear constraints

We can extend the auxiliary ODE from Proposition 8.4 to handle general nonlinear constraints. However, in this case, the constraint normal, represented by $n(q) = \nabla_q f_c(q)$, and the inertia matrix represented by $M(q)$, are no longer constant, since $q(\tau) \neq q_{\text{s}}$ holds for $\tau \in (\tau_{\text{s}}, \tau_{\text{r}})$. Moreover, if the contact happens close to the intersection point of surfaces defined by the zero level sets of two or more constraints, $q(\tau)$ might *dive* into the wrong region and make the reformulation invalid, cf. Figure 8.3.

As a result, we cannot use a two-dimensional spring-damper model and embed it back into the full space. To overcome this difficulty, we propose introducing an auxiliary one-dimensional state, denoted by $p(\tau)$, which takes on the role of $n(q)^\top q$. Whenever $y \in R_{\text{aux}}$, we *freeze* the evolution of $q(\tau)$. This freezing ensures that $n(q(\tau))$ and $M(q(\tau))$ remain constant for $\tau \in [\tau_{\text{s}}, \tau_{\text{r}}]$. As a result, we can use a modified version of the auxiliary dynamics from Proposition 8.4.

The extended state space of the time-freezing system is now given by $\hat{y}(\tau) = (y(\tau), p(\tau)) \in \mathbb{R}^{n_x+2}$. Let $\hat{x}(\tau) \coloneqq (q(\tau), v(\tau), p(\tau))$. For $y \in R_{\text{ODE}}$ we keep $y'(\tau) = (f_{\text{ODE}}(x,u),1)$ and set $p'(\tau) = 0$ and $p(0) = 0$, since $p(\tau)$ is not needed in this region. For $y \in R_{\text{aux}}$ the auxiliary dynamics mimic the state jump in $n(q)^\top v$, but now in the space spanned by $[\nabla p \ \nabla(n^\top v)] \in \mathbb{R}^{n_q \times 2}$ instead of $[\nabla f_c(q) \ \nabla(n^\top v)] \in \mathbb{R}^{n_q \times 2}$. The state jump is emulated in $p \leq 0$ (and $f_c(q)) \leq 0$). The evolution of $q(\tau)$ in this region is frozen, i.e., $q'(\tau) = 0$. Next, in the space spanned by $[\nabla p \ \nabla(n^\top v)]$, we use a 2D damped linear oscillator to mimic the state jump. The result is then embedded back into $\mathbb{R}^{n_x+2}$. This behavior is

modeled by the following vector field

$$f^{-}_{\mathrm{aux},n}(\hat{y}) = \begin{bmatrix} \mathbf{0}_{n_q,1} \\ M(q)^{-1}n(-kp - cn(q)^{\top}v) \\ n(q)^{\top}v \\ 0 \end{bmatrix}.$$

It is left to define a vector field in the region $p > 0$ and $f_c(q) < 0$. The solution does not enter this region, and the problem should not be initialized there. However, we define a vector field that points outwards of it, in case the trajectory enters this region due to numerical errors

$$f^{+}_{\mathrm{aux,n}}(y(\tau)) = \begin{bmatrix} n(q)a \\ n(q)a \\ a \\ 0 \end{bmatrix},$$

where a is a positive constant. Bringing these two modes together we have the piecewise smooth auxiliary ODE

$$\hat{y}' = f_{\mathrm{aux,n}}(\hat{y}) = \begin{cases} f^{+}_{\mathrm{aux,n}}(\hat{y}), & \text{if } y \in R_{\mathrm{aux}} \cap \{\hat{y} \in \mathbb{R}^{n_x+2} \mid p > 0\}, \\ f^{-}_{\mathrm{aux,n}}(\hat{y}), & \text{if } y \in R_{\mathrm{aux}} \cap \{\hat{y} \in \mathbb{R}^{n_x+2} \mid p < 0\}, \end{cases} \tag{8.15}$$

By extending the definition for the ODE in $y \in R_{\mathrm{ODE}}$ by $\hat{y}' = (f_{\mathrm{ODE}}(x,u), 0, 1)$, we can extend Definition 8.3 accordingly.

An illustration of a solution of the time-freezing system with the novel auxiliary ODE for the example in Eq. (8.15) is depicted in Figure 8.5. The solution trajectory of the old auxiliary ODE evolves in the $q_2 - v_2$ plane (blue (dashed) curve) and $q_2 < 0$, $\tau \in (\tau_{\mathrm{s}}, \tau_{\mathrm{r}})$. In contrast to that, the solution trajectory of the novel ODE evolves in the $p - q_2$ plane (orange (dotted) curve), and it does not enter $q_2 < 0$. Therefore, it holds that $q(\tau) = q_{\mathrm{s}}$ for all $\tau \in [\tau_{\mathrm{s}}, \tau_{\mathrm{r}}]$ and thus the constraint normal $n(q)$ and inertia matrix $M(q)$ stay constant during this time interval. Consequently, nonlinear constraints $f_c(q)$ can be treated naturally with the novel auxiliary ODE.

8.3.4 Solution relationship

In this section, we show how the solutions of the initial nonsmooth differential equation with a state jump law (8.4) and the corresponding time-freezing system (8.11) are related. Note that the function $t(\tau; x_0)$ is monotone by construction.

Theorem 8.5 (Solution relationship - elastic impacts)**.** *Regard the initial value problems corresponding to:*

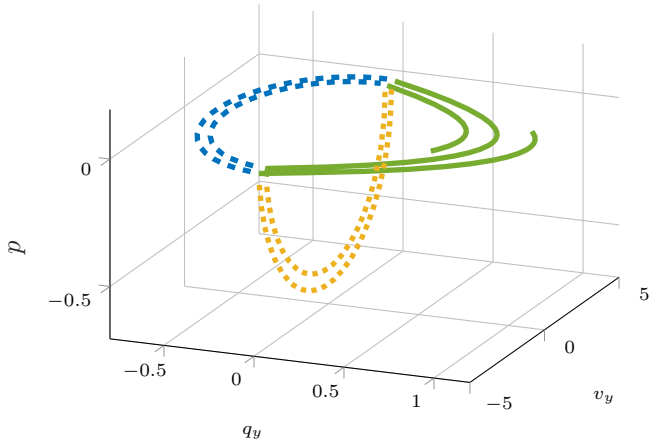


Figure 8.5: The green (solid) and orange (dotted) curve show a solution to the time-freezing system corresponding to the illustrative example in Eq. (8.2) but with the novel auxiliary ODE (8.15). The blue (dashed) and green (solid) curve show a solution with the auxiliary ODE from Proposition 8.4.

(i) the time-freezing system in Eq. (8.11) *with a given* $y(0) = (q_0, v_0, 0) \in \mathbb{R}^{n_y}$ *and* $f_c(q_0) > 0$ *on a time interval* $[0, \tau_\mathrm{f}]$*, and*

(ii) the CLS from Eq. (8.4) *with the initial value* $x(0) = (q_0, v_0) \in \mathbb{R}^{n_x}$ *on the time interval* $[0, t_\mathrm{f}] \coloneqq [0, t(\tau_\mathrm{f})]$*, with* $f_c(q(t_\mathrm{f})) > 0$*.*

Assume that an auxiliary dynamics $f_{\mathrm{aux,n}}(y)$ *from Definition 8.2 exists and that there is at most one time point* $t_\mathrm{s} = t(\tau_\mathrm{s})$ *where* $f_c(q(t_\mathrm{s})) = 0$ *and* $n(q(t_\mathrm{s}))^\top v(t_\mathrm{s}^-) < 0$ *on the time interval* $[0, t_\mathrm{f}]$*, Then the solution of the two IVPs:* $x(t; x_0)$ *and* $y(\tau; y_0)$ *fulfill at any* $t \neq t_\mathrm{s}$

$$x(t(\tau)) = Ry(\tau), \text{ with } R = \begin{bmatrix} I_{n_x} & \mathbf{0}_{n_x,1} \end{bmatrix}, \tag{8.16}$$

Proof.: Let $y_1(\tau; y_0)$ denote the solution of the IVP given by (8.11) and y_0 for $\tau \in (0, \hat{\tau})$. Similarly, let $x_1(t(\tau); x_0)$ denote the solution to (8.4) and x_0 for $t(\tau) \in (0, t(\hat{\tau}))$. Note that if there is no $t_\mathrm{s} \in (0, t_\mathrm{f})$ such that $f_c(x(t_\mathrm{s})) = 0$ on this interval, then $t(\tau) = \int_0^\tau \mathrm{d}\tau_1 = \tau$. Then setting $\hat{\tau} = \tau_\mathrm{f}$, it follows that (8.16) holds, since we have that $x_1(t; x_0) = x(t; x_0)$ and $y_1(\tau; x_0) = y(\tau; y_0)$ and $y \in R_\mathrm{ODE}$.

If we have some $t_\mathrm{s} \in (0, t_\mathrm{f})$ so that $f_c(x(t_\mathrm{s})) = 0$, then from the first part of the proof we have that (8.16) holds for all $\tau \in (0, \tau_\mathrm{s}^-)$ and hence for all $t(\tau) \in (0, t_\mathrm{s}^-)$,

with $t_\mathrm{s} = \tau_\mathrm{s}$. It is only left to prove that (8.16) holds for $\tau \in (\tau_\mathrm{s}^+, \tau_\mathrm{f})$ and the respective $t(\tau)$. By definition the solution of an auxiliary ODE $y'(\tau) = f_{\mathrm{aux,n}}(y(\tau))$ satisfies the restitution law and we have that $n(q(\tau_\mathrm{s}))^\top v(\tau_\mathrm{r}) = -\epsilon_\mathrm{r} n(q(\tau_\mathrm{s}))^\top v(\tau_\mathrm{s})$. We denote by $y_\mathrm{s} = y(\tau_\mathrm{r})$. Observe that $t'(\tau) = 0$ for $\tau \in (\tau_\mathrm{s}, \tau_\mathrm{r})$, hence $t(\tau_\mathrm{r}) = t(\tau_\mathrm{s}) = t_\mathrm{s}$ holds. Using this, we have that $y_1(\tau - \tau_\mathrm{r}, y_\mathrm{s}) = y(\tau, y_0)$ for $\tau \in (\tau_\mathrm{r}, \tau_\mathrm{f})$ and with denoting $x_\mathrm{s} = R y_\mathrm{s}$, we see that $x_1(t(\tau) - t_\mathrm{s}; x_\mathrm{s}) = x(t(\tau), x_0)$ for $t(\tau) \in (t_\mathrm{s}^+, t_\mathrm{f})$. Since the intervals $(t_\mathrm{s}, t_\mathrm{f})$ and $(\tau_\mathrm{r}, \tau_\mathrm{f})$ have the same length and $x_\mathrm{s} = R y_\mathrm{s}$, from the definitions of the corresponding IVPs, we conclude that relation (8.16) holds. This completes the proof. □

The assumptions that we have at most one state jump on the time interval $(0, t_\mathrm{f})$ can always be satisfied by shortening the regarded time interval and can be generalized inductively. Furthermore, we avoid the analysis of the case with infinite switches in finite time (Zeno behavior). For a desired physical simulation t_f we always have to take a longer pseudo simulation time $\tau_\mathrm{f} = t_\mathrm{f} + N_\mathrm{J}\tau_\mathrm{jump}$, where N_J is the number of state jumps on $(0, t_\mathrm{f})$ for the original system. We do not know a priori the number N_J. Note that τ_jump has the same value for all state jumps.

8.4 Time-freezing for inelastic impacts

In this section, we extend the time-freezing reformulation to the case of inelastic impacts, i.e., the coefficient of restitution ϵ_r is zero. Rigid body models with friction and inelastic impacts are indispensable in modern robotic control applications, as any complex task requires exploiting contacts and friction [49, 129, 142, 225, 263, 255]. We regard again a single unilateral constraint, but now with friction. Moreover, we outline some ideas on how to extend the developments here to the case of multiple and simultaneous impacts. We discuss how to construct auxiliary dynamics and formalize the relationship between the time-freezing system and CLS. For simplicity and ease of exposition, we first focus on the case without friction. Extensions with frictional impacts are given in Section 8.4.3.

8.4.1 The time-freezing reformulation

From Eq. (8.1), we have that a CLS without friction and inelastic impacts reduces to

$$\dot{q} = v, \tag{8.17a}$$

$$\dot{v} = f_{\mathrm{v}}(q, v, u) + M(q)^{-1} n(q) \lambda_{\mathrm{n}}, \tag{8.17b}$$

$$0 \leq \lambda_{\mathrm{n}} \perp f_c(q) \geq 0, \tag{8.17c}$$

$$\begin{aligned} &0 = n(q(t_{\mathrm{s}}))^{\top} v(t_{\mathrm{s}}^{+}), \\ &\quad \text{if } f_c(q(t_{\mathrm{s}})) = 0 \text{ and } n(q(t_{\mathrm{s}}))^{\top} v(t_{\mathrm{s}}^{-}) < 0, \end{aligned} \tag{8.17d}$$

Different modes of the CLS

We start by investigating different possible modes of the CLS (8.17). Afterwards, the time-freezing reformulation with its needed ingredients is introduced. For the CLS (8.17) we can distinguish two modes of operation:

(i) the system is not in contact (unconstrained case, free flight), i.e., $f_c(q) > 0$, which implies $\lambda_{\mathrm{n}} = 0$,

(ii) the system is in contact, i.e., $f_c(q) = 0$ and $\lambda_{\mathrm{n}} \geq 0$.

In the first case, the system evolves according to the ODE:

$$\dot{q} = v, \tag{8.18}$$

$$\dot{v} = f_{\mathrm{v}}(q, v, u). \tag{8.19}$$

We write this ODE compactly as $\dot{x} = f_{\mathrm{ODE}}(x, u) \coloneqq (v, f_{\mathrm{v}}(q, v, u))$. We call an active-set change from $\lambda_{\mathrm{n}}(t_{\mathrm{s}}^{-}) = 0,\ f_c(q(t_{\mathrm{s}}^{-})) \geq 0$ to $\lambda_{\mathrm{n}}(t_{\mathrm{s}}^{+}) \geq 0,\ f_c(q(t_{\mathrm{s}}^{+})) = 0$, which triggers a state jump, an *impact*.

After an impact, it holds that $0 = n(q(t_{\mathrm{s}}))^{\top} v(t_{\mathrm{s}}^{+})$. Subsequently, the system evolves according to a DAE of index 3:

$$\dot{q} = v, \tag{8.20a}$$

$$\dot{v} = f_{\mathrm{v}}(q, v, u) + M(q)^{-1} n(q) \lambda_{\mathrm{n}}, \tag{8.20b}$$

$$0 = f_c(q). \tag{8.20c}$$

Note that $f_c(q(t))$ needs to be differentiated twice w.r.t. to time until $\lambda_{\mathrm{n}}(t)$ appears explicitly. The next question to be answered is: Will the system stay in contact (dynamics defined by (8.20) with $f_c(q) = 0$) or will the *contact break* (dynamics defined by (8.18) with $f_c(q) > 0$)? The answer can be found by looking at the *contact Linear Complementarity Problem* (LCP) [49, Section

5.1.2]. Under the standing assumptions, during contact on some time interval $[t_1, t_2]$ the *consistent initialization* conditions hold

$$0 = f_c(q(t)), \tag{8.21}$$

$$0 = \frac{\mathrm{d}}{\mathrm{d}t} f_c(q(t)) = \nabla_q f_c(q(t))^\top v(t). \tag{8.22}$$

Consequently, $\lambda_\mathrm{n}(t) \geq 0, t \in [t_1, t_2]$. Due to the continuity of $q(t)$, $f_c(q(t))$ and $\frac{\mathrm{d}}{\mathrm{d}t} f_c(q(t))$, for contact breaking (i.e., $f_c(q)$ becomes strictly positive) it is required that $\frac{\mathrm{d}^2}{\mathrm{d}t^2} f_c(q(t)) \geq 0$ for $t \in [t_2, t_2 + \hat{\epsilon})$, for some $\hat{\epsilon} > 0$. Therefore, from (8.17c) we deduce that

$$0 \leq \frac{\mathrm{d}^2}{\mathrm{d}t^2} f_c(q(t)) \perp \lambda_\mathrm{n}(t) \geq 0, \ t \in [t_1, t_2]. \tag{8.23}$$

Then, by computing $\frac{\mathrm{d}^2}{\mathrm{d}t^2} f_c(q(t))$ and using the r.h.s. of (8.20b), we obtain the contact LCP in $\lambda_\mathrm{n}(t)$:

$$0 \leq D(q)\lambda_\mathrm{n} + \varphi(x, u) \perp \lambda_\mathrm{n} \geq 0, \tag{8.24}$$

with

$$D(q) = \nabla_q f_c(q)^\top M(q)^{-1} \nabla_q f_c(q), \tag{8.25a}$$

$$\varphi(x, u) = \nabla_q f_c(q)^\top f_\mathrm{v}(q, v, u) + \nabla_q (\nabla_q f_c(q)^\top v)^\top v, \tag{8.25b}$$

where $D(q) > 0$. The solution map of the LCP (8.24) is given by

$$\lambda_\mathrm{n} = \max(0, -D(q)^{-1} \varphi(x, u)). \tag{8.26}$$

From the last equation, we deduce that contact breaking or sticking depends on the sign of the function $\varphi(x, u)$.

In the case of $\varphi(x, u) \leq 0$, from Eq. (8.24) and (8.26) it follows that $\lambda_\mathrm{n}(t) \geq 0$ and $\frac{\mathrm{d}^2}{\mathrm{d}t^2} f_c(q(t)) = 0$. Therefore, we have a persistent contact, and the system evolves according to the DAE (8.20). Using index reduction and the solution map (8.26), we can derive an ODE that is equivalent to the DAE (8.20):

$$\dot{q} = v, \tag{8.27a}$$

$$\dot{v} = f_\mathrm{v}(q, v, u) - \ M(q)^{-1} n(q) D(q)^{-1} \varphi(x, u). \tag{8.27b}$$

We compactly denote this ODE by $\dot{x} = f_\mathrm{DAE}(x, u)$.

In the second case, $\varphi(x, u) > 0$ implies $\lambda_\mathrm{n}(t) = 0$ and $\frac{\mathrm{d}^2}{\mathrm{d}t^2} f_c(q(t)) > 0$. Therefore, the contact breaks, and the system evolves according to the ODE (8.18).

To summarize, if the system switches from the ODE mode in (8.18) to the DAE mode in (8.27), a state jump must occur, except if the active-set changes happen with $n(q(t_\mathrm{s}^-)^\top v(t_\mathrm{s}^-) = 0$. Now the system evolves on the boundary of the feasible set with $f_c(q) = 0$ according to the DAE (8.20), or equivalently according to the ODE defined by (8.27). On the other hand, if we switch from DAE to ODE mode, we have a continuous transition without state jumps, i.e., contact breaking occurs.

Main ideas and auxiliary dynamics

The arguments above reveal that the CLS (8.17) switches between an ODE and a DAE of index 3. This already bears similarity to a piecewise smooth system, but the main obstacle to completing this transition is the state jumps. Note that large parts of the state space, namely $f_c(q) < 0$, are prohibited for the solution trajectories of the CLS.

As in the previous sections, we first relax the constraint and allow $f_c(q) < 0$. We define an auxiliary dynamical system in this *infeasible* region whose trajectory endpoints satisfy the state jump law (8.17d) on some finite time interval. Second, we introduce a *clock state* $t(\tau)$ that stops counting (i.e., $t'(\tau) = 0$), when the auxiliary ODE is active (for $f_c(q) < 0$). By taking the pieces of the trajectory when the clock state was active, one can recover the solution of the original system with discontinuous trajectories. The extended state of the time-freezing system reads as $y \coloneqq (x, t) \in \mathbb{R}^{n_y}$, $n_y = n_x + 1$. The properties of the auxiliary dynamics are summarized in the following definition.

Definition 8.6 (Auxiliary dynamics)**.** *An auxiliary dynamical system, for the normal velocity state jumps, $y'(\tau) = f_{\mathrm{aux,n}}(y(\tau))$ satisfies for every initial value $y(\tau_\mathrm{s}) = (q_\mathrm{s}, v_\mathrm{s}, t_s)$, with $f_c(q_\mathrm{s}) = 0$ and $n(q_\mathrm{s})^\top v_\mathrm{s} < 0$, for a well-defined and finite time interval $(\tau_\mathrm{s}, \tau_\mathrm{r})$, with the length $\tau_\mathrm{jump} = \tau_\mathrm{r} - \tau_\mathrm{s}$, the following properties:*

(i) $f_c(q(\tau)) \leq 0, t'(\tau) = 0$ *for all* $\tau \in (\tau_\mathrm{s}, \tau_\mathrm{r})$,

(ii) $n(q(\tau_\mathrm{r}))^\top v(\tau_\mathrm{r}) = 0$, *and*

(iii) $f_c(q(\tau_\mathrm{r})) = 0$.

To construct the time-freezing system, we take several steps. First, observe that the post-impact velocity (8.17d) is equal to the total time derivative of the constraint $f_c(q) = 0$, i.e., $\frac{\mathrm{d}}{\mathrm{d}t} f_c(q) = n(q)^\top v = 0$. We choose these functions as switching functions, i.e., $c_1(y) = f_c(q)$ and $c_2(y) = n(q)^\top v$, and define the

following regions:

$$\begin{aligned} R_{\text{ODE}} &= \{y \in \mathbb{R}^{n_y} \mid c_1(y) > 0\} \cup \{y \in \mathbb{R}^{n_y} \mid c_1(y) < 0, c_2(y) > 0\}, \\ R_{\text{aux}} &= \{y \in \mathbb{R}^{n_y} \mid c_1(y) < 0, c_2(y) < 0\}. \end{aligned} \tag{8.28}$$

Second, we associate with the region R_{ODE} the *unconstrained dynamics* $y' = (f_{\text{ODE}}(x,u),1)$ and with R_{aux} the auxiliary dynamics from Definition 8.6. We formally define the time-freezing system and its Filippov extension in the next definition.

Definition 8.7 (Time-freezing system - the inelastic case)**.** *Let $\tau \in \mathbb{R}$ be the numerical time and $y(\tau) \coloneqq (x(\tau), t(\tau)) \in \mathbb{R}^{n_y}$ the differential states and $u(\tau) \in \mathbb{R}^{n_u}$ a given control function. The time-freezing PSS for a given $f_{\text{aux,n}}(y)$ is the PSS:*

$$y' = \begin{cases} f_1(y), & \text{if } y \in R_{\text{ODE}}, \\ f_2(y), & \text{if } y \in R_{\text{aux}}, \end{cases} \tag{8.29}$$

where $f_1(y,u) = (f_{\text{ODE}}(x,u),1)$ and $f_2(y) = f_{\text{aux,n}}(y)$. The corresponding Filippov system, which we call the time-freezing system, is defined as

$$y' \in F_{\text{TF}}(y,u) \coloneqq \Big\{\theta_1 f_1(y) + \theta_2 f_2(y) \mid e^\top \theta = 1, \theta \geq 0\Big\}, \tag{8.30}$$

with $\theta = (\theta_1, \theta_2)$.

Figure 8.6 shows the phase plot of the time-freezing system. Note that the region R_{aux} consists of the set $\{y \in \mathbb{R}^{n_y} \mid f_c(q) > 0\}$ (green area), that corresponds to the feasible set of the unconstrained dynamics (8.18) and the set $\{y \in \mathbb{R}^{n_y} \mid c_1(y) < 0, c_2(y) > 0\}$ (yellow area). The solution trajectories never flow in the latter set and the system should not be initialized there. However, as we show later, it is crucial for sliding modes and contact breaking. Region R_{aux} (red shaded area) contains the auxiliary dynamics that mimic the state jump.

To make further use of Definition 8.7, let us specify how to construct an appropriate auxiliary ODE. The next proposition provides a constructive way of selecting the auxiliary ODE from Definition 8.6 for any smooth scalar constraint $f_c(q) = 0$.

Proposition 8.8 (Auxiliary dynamics)**.** *Suppose that $y(\tau_{\text{s}}) = (q_{\text{s}}, v_{\text{s}}, t_{\text{s}})$ is given such that $f_c(q_{\text{s}}) = 0$ and $n(q_{\text{s}})^\top v_{\text{s}} \leq 0$ holds. Then the ODE given by*

$$y' = f_{\text{aux,n}}(y) \coloneqq \begin{bmatrix} \mathbf{0}_{n_q,1} \\ M(q)^{-1} n(q) a_{\text{n}} \\ 0 \end{bmatrix} \tag{8.31}$$

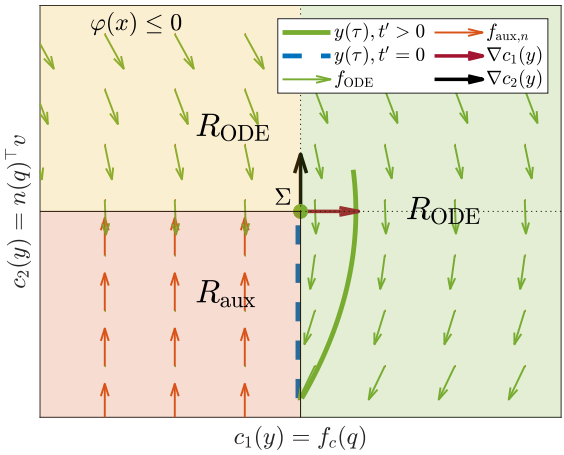


Figure 8.6: Illustration of a phase plot of the time-freezing system from Definition 8.7. The red and yellow shaded areas are infeasible for the CLS (8.17). The trajectories of the auxiliary dynamics (the blue dashed line) flow in the red-shaded area.

with $a_{\text{n}} > 0$ *is an auxiliary dynamical system from Definition 8.6 with* $\tau_{\text{jump}} = -\frac{n(q_{\text{s}})^\top v_{\text{s}}}{D(q_{\text{s}})a_{\text{n}}}$

Proof. According to (8.31) we have $q'(\tau) = \mathbf{0}_{n_q,1},\ \forall \tau \geq \tau_{\text{s}}$, which implies $q(\tau) = q_{\text{s}}$ and $f_c(q(\tau)) = 0,\ \forall \tau \geq \tau_{\text{s}}$. This means also that $M(q(\tau)) = M(q_{\text{s}}),\ \forall \tau \geq \tau_{\text{s}}$. Second, regard the dynamics of $v' = M(q)^{-1}n(q)a_{\text{n}} = M(q_{\text{s}})^{-1}n(q_{\text{s}})a_{\text{n}}$ and rewrite this equation in integral form. By multiplying it from the left by $n(q_{\text{s}})^\top$ we obtain:

$$n(q_{\text{s}})^\top v(\tau) = n(q_{\text{s}})^\top v_{\text{s}} + n(q_{\text{s}})^\top M(q_{\text{s}})^{-1} n(q_{\text{s}}) a_{\text{n}} (\tau - \tau_{\text{s}}).$$

Since the first term on the r.h.s. is negative and the second strictly positive, we deduce that $n(q(\tau_{\text{r}}))^\top v(\tau_{\text{r}}) = 0$ and $f_c(q(\tau_{\text{r}})) = 0$ with $\tau_{\text{jump}} = \tau_{\text{r}} - \tau_{\text{s}} = -\frac{n(q_{\text{s}})^\top v_{\text{s}}}{D(q_{\text{s}})a_{\text{n}}}$. Hence, all conditions from Definition 8.6 are satisfied and the proof is complete. □

Next, we discuss which mode of the time-freezing system matches the persistent contact dynamics of the CLS (8.17). We observe that the set $\Sigma := \{y \mid c_1(y) = f_c(q) = 0, c_2(y) = n^\top v = 0\}$ is defined by the same equations as the consistent initialization conditions (8.21) (but now in $\mathbb{R}^{n_y}$ instead of $\mathbb{R}^{n_x}$ due to the clock state). A sliding mode of the time-freezing PSS evolves on Σ just as the solution of the persistent contact DAE (8.20). Therefore, its dynamics should match the

dynamics of the DAE (8.20). We detail In the next two subsections that this is indeed the case.

To summarize, for $f_c(q) > 0$ (which is a subset of R_{ODE}) the time-freezing system and CLS have the same dynamics. In R_{aux} the auxiliary dynamics mimic the state jump and the sliding mode $y \in \Sigma$ should match the dynamics of the CLS in contact mode. To illustrate the developments so far, we derive a time-freezing PSS for the guiding example.

Example 8.9. *(Guiding example as time-freezing PSS) The state space is* $y = (q, v, t) \in \mathbb{R}^5$ *and the switching functions read as* $c_1(y) = y_2$ *and* $c_2(y) = v_2$. *The two PSS regions are* $R_{\mathrm{ODE}} = \{y \in \mathbb{R}^5 \mid q_2 > 0\} \cup \{y \in \mathbb{R}^5 \mid q_2 < 0, v_2 > 0\}$ *and* $R_{\mathrm{aux}} = \{y \in \mathbb{R}^5 \mid q_2 < 0, v_2 < 0\}$, *cf. Fig. 8.6. The dynamics in* R_{ODE} *are given by* $f_1 = (v_1, v_2, u_1, -g + u_2, 1)$. *The constraint normal is* $n = (0, 1)$ *and by applying Eq.* (8.31) *we find that* $f_2 = (0, 0, 0, a_{\mathrm{n}}, 0)$. *Therefore we have the PSS:*

$$y' = \begin{cases} (v_1, v_2, u_1, -g + u_2, 1), & \text{if } y \in R_{\mathrm{ODE}}, \\ (0, 0, 0, a_{\mathrm{n}}, 0), & \text{if } y \in R_{\mathrm{aux}}. \end{cases}$$

Persistent contact and sliding mode

Depending on the sign of the function $\varphi(x, u)$, a solution y initialized at Σ should either stay at Σ (sliding mode, persistent contact) or leave it (contact breaking). In this subsection, we study the case when a solution of the time-freezing system satisfies the conditions $y(\tau) \in \Sigma$ and $\varphi(x(\tau), u(\tau)) \leq 0$ for some $\tau \in [\tau_1, \tau_2]$ (persistent contact). During contact, the CLS system satisfies the consistent initialization (8.21) which corresponds to Σ without the clock state. It is desired that under these conditions $y(\tau)$ stays on Σ and that the corresponding sliding mode dynamics match the DAE dynamics (8.27). For a solution to stay in the sliding mode, the surface Σ must be *stable*, i.e., all neighboring vector fields point toward Σ. Since $\nabla c_1(y)^\top f_1(y, u) = 0$, $\nabla c_1(y)^\top f_2(y) = 0$ and $\nabla c_2(y)^\top f_1(y, u) = \varphi(x, u) \leq 0$, $\nabla c_2(y)^\top f_2(y) = D(q)a_{\mathrm{n}} > 0$, we see that this is indeed the case, cf. Figure 8.6. We show next that the sliding mode of the time-freezing system is unique and that it matches the dynamics of the DAE of index 3 after the state jump, as required.

Theorem 8.10 (Unique sliding mode)**.** *Regard the time-freezing system from Definition 8.7 with the auxiliary dynamics from Proposition 8.8. Let* $y(\tau)$ *be a solution of this system with* $y(0) \in \Sigma$ *and* $\tau \in [0, \tau_{\mathrm{f}}]$. *Suppose that* $\varphi(x(\tau), u(\tau)) \leq 0$ *for all* $\tau \in [0, \tau_{\mathrm{f}}]$ *(persistent contact), then the following statements are true:*

(i) the Filippov multipliers $\theta_1, \theta_2 \geq 0$ *in Eq.* (8.30) *are unique,*

(ii) the dynamics of the sliding mode are given by $y' = \gamma(x,u)(f_{\mathrm{DAE}}(x,u),1)$, *where* $\gamma(x,u) \in (0,1]$ *is a time-rescaling factor given by*

$$\gamma(x,u) \coloneqq \frac{D(q)a_{\mathrm{n}}}{D(q)a_{\mathrm{n}} - \varphi(x,u)}. \tag{8.32}$$

Proof. To compute the multipliers θ_1 and θ_2 we use Definition 8.7 and the fact that $y \in \Sigma$. This results in the conditions:

$$c_1(y) = 0,\ c_2(y) = 0,\ \theta_1 + \theta_2 = 1.$$

Since θ does not explicitly appear in the first two conditions, we differentiate them w.r.t. to τ and use Eq. (8.30). We have two unknowns and three conditions, hence the system is over-determined. However, we have by assumption that $c_2(y) = \nabla_q c_1(y)^\top v = 0$ and by direct evaluation, we conclude that $\frac{\mathrm{d}c_1(y)}{\mathrm{d}\tau} = 0$ is satisfied for every θ_1 and θ_2. The conditions that are left are $\frac{\mathrm{d}c_2(y)}{\mathrm{d}\tau} = 0$ and $\theta_1 + \theta_2 = 1$. Since $\nabla_v c_2(y) = \nabla_v(\nabla_q c_1(y)^\top v) = \nabla_q c_1(y)$ and $\frac{\partial c_2(y)}{\partial t} = 0$ we obtain from $\frac{\mathrm{d}c_2(y)}{\mathrm{d}\tau} = 0$ the following equations

$$\begin{aligned}
0 = \frac{\mathrm{d}c_2(y)}{\mathrm{d}\tau} &= \theta_1 \left[\nabla_q c_2(y)^\top \nabla_v c_2(y)^\top\right] \begin{bmatrix} v \\ f_v(q,v,u) \end{bmatrix} \\
&\quad + \theta_2 \left[\nabla_q c_2(y)^\top \nabla_v c_2(y)^\top\right] \begin{bmatrix} 0 \\ M(q)^{-1}\nabla_q c_1(y) a_{\mathrm{n}} \end{bmatrix}, \\
0 &= \theta_1 \underbrace{\nabla_q(\nabla_q f_c(q)v)^\top v + \nabla_q f_c(q)^\top f_v(q,v,u)}_{=\varphi(x,u)<0} \\
&\quad + \theta_2 \underbrace{\nabla_q f_c(q)^\top M(q)^{-1}\nabla_q f_c(q)}_{=D(q)>0} a_{\mathrm{n}}.
\end{aligned}$$

Thus we obtain a system linear in θ

$$\begin{bmatrix} \varphi(x,u) & D(q)a_{\mathrm{n}} \\ 1 & 1 \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

and by solving it we have that $\theta_1 = \frac{D(q)a_{\mathrm{n}}}{D(q)a_{\mathrm{n}} - \varphi(x,u)} = \gamma(x,u)$ and $\theta_2 = \frac{-\varphi(x,u)}{D(q)a_{\mathrm{n}} - \varphi(x,u)}$. Since $D(q)a_{\mathrm{n}} - \varphi(x,u) > 0$ and $\varphi(x,u) \leq 0$, we have always unique $\theta_1, \theta_2 \geq 0$. This completes the first part of the proof.

For the second part, we evaluate

$$y' = \theta_1 f_1(y,u) + \theta_2 f_2(y) = \gamma(x,u) \begin{bmatrix} v \\ f_v(q,v,u) \\ 1 \end{bmatrix}$$

$$+ \frac{-\varphi(x,u)}{D(q)a_\mathrm{n} - \varphi(x,u)} \begin{bmatrix} \mathbf{0}_{n_q,1} \\ M(q)^{-1}\nabla f_c(q) a_\mathrm{n} \\ 0 \end{bmatrix} \cdot D(q)^{-1} D(q).$$

In the second term we use that $\lambda_\mathrm{n} = -D(q)\varphi(x,u)$ (cf. Eq. (8.26)) and the expression for $\gamma(x,u)$ in Eq. (8.32). By comparing the last expression to Eq. (8.27) we obtain $y' = \gamma(x,u)(f_\mathrm{DAE}(x,u),1)$. □

This theorem shows that the sliding mode of the time-freezing system on Σ is unique and equal to the dynamics of the CLS in the persistent contact mode given by Eq. (8.27) but slowed down by the factor $\gamma(x,u)$. Note that for larger values of a_n the factor $\gamma(x,u)$ comes closer to one, which reduces the slow-down, cf. Example 8.11.

However, by plotting $x(\cdot)$ over $t(\tau)$ the solution with a speed of time of one is recovered. We briefly discuss the intuition behind the time slow down by $\gamma(x,u)$. To achieve the sliding mode on Σ, the vector fields from R_aux and R_ODE (the yellow part in Fig. 8.6) must push toward Σ. The resulting dynamics is a convex combination of the two vector fields, and since the speed of time in R_ODE is one and in R_aux zero we obtain a slow down equal to $\theta_1 = \gamma(x,u)$. Moreover, the vector field $f_\mathrm{ODE}(\cdot)$ in the yellow area "stops" the trajectory coming from R_aux and thus enables the sliding mode. This shows the significance of having the vector field $(f_\mathrm{ODE}(\cdot),1)$ in the yellow area $\{y \mid c_1(y) < 0, c_2(y) > 0\}$, even though by construction the solution never flows there.

Contact breaking

It is left to study the case when $y(\tau_1) \in \Sigma$ but $\varphi(x(\tau),u(\tau)) > 0$ for $\tau \in [\tau_1,\tau_2]$. In the CLS for $\varphi(x,u) > 0$ the contact breaks. In the time-freezing system, we expect the trajectory to leave the sliding mode from Σ. Since $\nabla c_1(y)^\top f_1(y,u) = 0$, $\nabla c_1(y)^\top f_2(y) = 0$ and $\nabla c_2(y)^\top f_1(y,u) = \varphi(x,u) > 0$, $\nabla c_2(y)^\top f_2(y) = D(q)a_\mathrm{n} > 0$, the surface Σ is not stable anymore. This scenario is illustrated in Fig. 8.7. We conclude that under these conditions y leaves Σ and enters R_ODE (into the green region with $f_c(q) > 0$). In this case $\theta = (1,0)$, hence $y' = f_1(y,u) = (f_\mathrm{ODE}(x,u),1)$, which matches the unconstrained CLS dynamics (8.18) augmented by the clock state.

Effectively, the time-freezing system switches between the DAE and ODE modes, just as the CLS and the state jump is performed by the auxiliary dynamics while the time is frozen. For $\varphi(x,u) \leq 0$ (persistent contact) the solution of the time-freezing system stays on Σ, just as the solution of the CLS. It leaves the

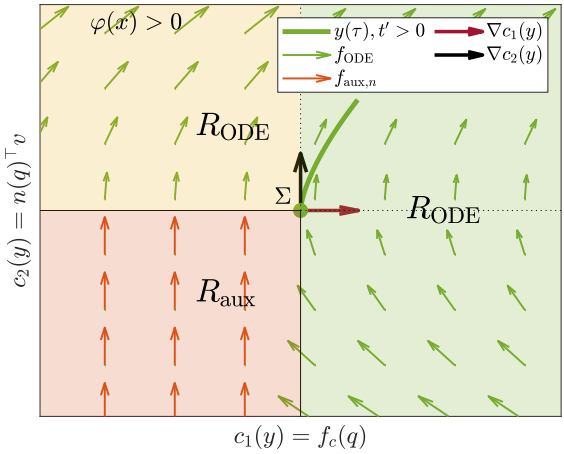


Figure 8.7: Illustration of a phase plot of the time-freezing system from Definition 8.7 with $\varphi(x, u) > 0$. Compared to Figure 8.6, the vector field in R_{ODE} is changed and Σ is not stable anymore, thus leaving Σ into R_{aux} is possible.

sliding mode when $\varphi(x, u) > 0$, which corresponds to contact breaking in the CLS. This relationship is formalized in the next subsection. To illustrate the developments of the last two subsections, we revisit the guiding example and provide a simulation that encompasses all effects discussed so far.

Example 8.11. *(Speed of time and sliding modes) Let us consider the time-freezing PSS from Example 8.9. We choose $a_n = g$. It follows that $D(q) = 1$ and $\varphi(x, u) = -g + u_2$. We choose a control function*

$$u(t) = \begin{cases} (7, 0), & t < 1, \\ (7, 2g(t(\tau) - 1)), & t \geq 1. \end{cases}$$

Let us make a simulation of the time-freezing system with $y(0) = (0, 1, 0, 0, 0)$ for $\tau \in [0, 3.5]$. The result is depicted in Fig. 8.8. The particle hits the ground, slides horizontally on it, and lifts off when the control force $u_2(t)$ is stronger than gravity, cf. the top plot. We see that when the particle hits the ground, the time is frozen and the auxiliary ODE is active (red strips). The vertical velocity v_2 becomes zero with the rate a_{n}. The system is then a sliding mode with a time slow down factor of $\gamma(x, u) = \frac{a_{\text{n}}}{a_{\text{n}} - (-g + u_2)} = \frac{g}{g - (-g + 0)} = 0.5$ for $t < 1$ ($\tau < 2$), cf. bottom left plot. At $t = 1$, which corresponds to $\tau = 2$ the vertical control force becomes nonzero and $\gamma(x, u) = \frac{g}{2g + u_2}$ grows. For $\tau > 2.8$ we have $\varphi(x(\tau), u(\tau)) = -g + u_2 > 0$ and Σ is not stable anymore. The particle lifts off and the contact breaks. Note that the solution of the time-freezing system $y(\cdot)$ is

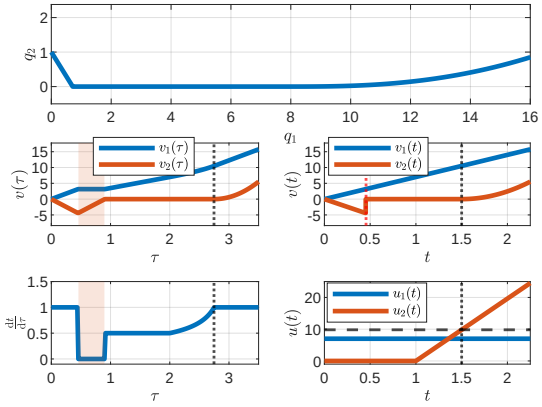


Figure 8.8: Trajectories of the time-freezing system corresponding to the example CLS. The top plot shows the position of the particle. The middle left plot shows the continuous velocities v_1 and v_2 in numerical time τ and the middle right plot shows the discontinuous velocities v_1 and v_2 in physical time t. The bottom left plot shows the speed of time $\frac{\mathrm{d}t}{\mathrm{d}\tau}$ and the bottom right plot the control function $u(t)$.

continuous in numerical time τ (middle left plot) and discontinuous in physical time t (middle right plot).

8.4.2 Solution relationship

We formalize now how to recover the solution of the initial value problem corresponding to the CLS (8.17) from the solution of the time-freezing system from Definition 8.7.

Theorem 8.12 (Solution relationship)**.** *Regard the initial value problems corresponding to:*

(i) the time-freezing system in Eq. (8.30) *with a given $y(0) = (q_0, v_0, 0) \in \mathbb{R}^{n_y}$ and $f_c(q_0) \geq 0$ on a time interval $[0, \tau_\mathrm{f}]$, and*

(i) the CLS from Eq. (8.17) *with the initial value $x(0) = (q_0, v_0) \in \mathbb{R}^{n_x}$ on a time interval $[0, t_\mathrm{f}] \coloneqq [0, t(\tau_\mathrm{f})]$, with $f_c(q(t_\mathrm{f})) \geq 0$ and $n(q(t_\mathrm{f}))^\top v(t_\mathrm{f}) \geq 0$.*

Suppose the following assumptions hold:

(a) the auxiliary dynamics $f_{\mathrm{aux,n}}(y)$ from Proposition 8.8 is used in the time-freezing system in Definition 8.7,

(b) there is at most one time point $t_\mathrm{s} = t(\tau_\mathrm{s})$ where $f_c(q(t_\mathrm{s})) = 0$ and $n(q(t_\mathrm{s}))^\top v(t_\mathrm{s}^-) < 0$ on the time interval $[0, t_\mathrm{f}]$,

Then, the solutions to the two problems are related as follows:

1. *for $t \neq t_\mathrm{s}$:*

$$x(t(\tau)) = Ry(\tau), \text{ with } R = \begin{bmatrix} I_{n_x} & \mathbf{0}_{n_x,1} \end{bmatrix}, \tag{8.33a}$$

$$\lambda_\mathrm{n}(t(\tau)) = \begin{cases} -D(q(t(\tau)))\varphi(x(t(\tau))), & \text{if } y(\tau) \in \Sigma, \\ 0, & \text{otherwise.} \end{cases} \tag{8.33b}$$

2. *for $t = t_\mathrm{s}$:*

$$\lim_{\substack{\epsilon \to 0 \\ \epsilon > 0}} \int_{t_\mathrm{s}-\epsilon}^{t_\mathrm{s}+\epsilon} \lambda_\mathrm{n}(t)\mathrm{d}t = \int_{\tau_\mathrm{s}}^{\tau_\mathrm{r}} a_\mathrm{n}\mathrm{d}\tau. \tag{8.34}$$

Proof. The idea of the proof is to consider the different modes, in which the CLS and the time-freezing system can be and to compare the solutions to establish the result of the theorem. A solution of the initial value problem given by the time-freezing system in Eq. (8.30) with $y(0) = y_0$ is denoted by $y_\mathrm{sol}(\tau; y_0)$ for $\tau \in [0, \hat{\tau}]$. Similarly, for the CLS in Eq. (8.17) and $x(0) = x_0$ for $t(\tau) \in [0, t(\hat{\tau})]$ we use $x_\mathrm{sol}(t(\tau); x_0)$. We must distinguish all possible cases, hence we split the proof into several parts.

Part I (Unconstrained case). Regard the case $f_c(q(\tau)) > 0, \tau \in [0, \hat{\tau}]$. This means that $y \in R_1$, $y' = f_1(y, u) = (f_\mathrm{ODE}(x, u), 1), \tau \in [0, \hat{\tau}]$. It holds that $t(\tau) = \int_0^\tau \mathrm{d}s = \tau$ and by setting $\hat{\tau} = \tau_\mathrm{f}$ we have $t(\tau_\mathrm{f}) = t_\mathrm{f}$. Note that $Ry' = Rf_1(y, u)$, is equivalent to $x' = f_\mathrm{ODE}(x, u)$. Since $[0, \tau_\mathrm{f}] = [0, t_\mathrm{f}]$, this ODE has the same solution as $\dot{x} = f_\mathrm{ODE}(x, u)$, therefore relation (8.33a) holds for $t \in [0, t_\mathrm{f}]$. This means that $f_c(q(t)) > 0$ and $\lambda_\mathrm{n}(t) = 0$ for $t \in [0, t_\mathrm{f}]$. For the time-freezing system this means that $y(\tau) \notin \Sigma$ for $\tau \in [0, \tau_\mathrm{f}]$, hence equation (8.33b) is also satisfied.

Part II (Sliding mode/persistent contact). Regard the case $f_c(q(0)) = 0$ and $n(q(0))^\top v(0) = 0$, i.e., $y(0) \in \Sigma$. Assume that $\varphi(x(\tau), u(\tau)) < 0$, $\tau \in [0, \tau_\mathrm{f}]$. This means that $y(\tau) \in \Sigma$, $\tau \in [0, \tau_\mathrm{f}]$, cf. Section 8.4.1. From Theorem 8.10 we have that $y' = \gamma(x, u)(f_\mathrm{DAE}(x, u), 1)$ for $\tau \in [0, \tau_\mathrm{f}]$ and $t'(\tau) = \gamma(x, u) > 0, \tau \in [0, \tau_\mathrm{f}]$, thus $t(\tau_\mathrm{f}) = t_\mathrm{f} > 0$. On one hand, from $\frac{\mathrm{d}y(t(\tau))}{\mathrm{d}\tau} = \frac{\mathrm{d}y}{\mathrm{d}t}\frac{\mathrm{d}t}{\mathrm{d}\tau}$ we have that $R\frac{\mathrm{d}y}{\mathrm{d}t} = R\frac{1}{\gamma(x,u)}\gamma(x, u)(f_\mathrm{DAE}(x, u), 1) = f_\mathrm{DAE}(x, u)$. On the other hand,

for the CLS we have for $f_c(q(0)) = 0,\ n(q(0))^\top v(0) = 0,\ z(t) \geq 0,\ t \in [0, t_\mathrm{f}]$. Consequently, the CLS reduces to the ODE $\dot{x} = f_\mathrm{DAE}(x, u)$. Similar to the part I, we conclude that (8.33a) holds. Since $f_c(q(t(\tau))) = 0$ and $n(q((\tau)))^\top v((\tau)) = 0$ for $\tau \in [0, \tau_\mathrm{f}]$, the relation for $\lambda_\mathrm{n}(t(\tau))$ in Eq. (8.33b) follows from Eq. (8.26) and (8.27).

Part III (Leaving sliding mode). Now we consider a similar scenario as in part II, with $y_0 \in \Sigma,\ \varphi(x(\tau), u(\tau)) \leq 0$ with $\tau \in [0, \tau_\mathrm{e}), \tau_\mathrm{e} < \tau_\mathrm{f}$ (sliding mode) and $\varphi(x(\tau), u(\tau)) > 0$ for $\tau \in [\tau_\mathrm{e}, \tau_\mathrm{f}]$ (leaving sliding mode). Relations (8.33a) and (8.33b) hold for $\tau \in [0, \tau_\mathrm{e})$ by the same arguments as in part II. For $\tau \geq \tau_\mathrm{e}$, following the arguments in Section 8.4.1, $y(\tau)$ leaves Σ and $y(\tau) \in R_1$ for $\tau \in [0, \tau_\mathrm{e})$. We can apply the arguments of part I and establish the result of the theorem.

Part IV (State jump). This part regards the case of $\tau_\mathrm{s} \in [0, \tau_\mathrm{f}]$, i.e., $t_\mathrm{s} \in [0, t_\mathrm{f}]$. For $\tau \in [0, \tau_\mathrm{s})$ and $t \in [0, t(\tau_\mathrm{s}^-))$ we can apply Part I of the proof by simply setting $\hat{\tau} = \tau_\mathrm{s}$ and deduce that (8.33a) and (8.33b) hold. For $\tau = \tau_\mathrm{s}$ we have $f_c(q(\tau_\mathrm{s})) = 0$ and $n(q(\tau_\mathrm{s}))^\top v(\tau_\mathrm{s}) < 0$. Consequently, $y \in R_2$ amd $y' = f_{\mathrm{aux,n}}(y)$. The assumption $f_c(q(\tau_\mathrm{f})) \geq 0$ and $n(q(t_\mathrm{f}))^\top v(t_\mathrm{f}) \geq 0$ ensures that the time evolution of $y'(\tau) = f_{\mathrm{aux,n}}(y(\tau))$ is finished in $[\tau_\mathrm{s}, \tau_\mathrm{f}]$, i.e., $\tau_\mathrm{r} \leq \tau_\mathrm{f}$. From the proof of Proposition 8.8 we know that by construction $q(\tau) = q(\tau_\mathrm{s}) =: q_\mathrm{s},\ \tau \in [\tau_\mathrm{s}, \tau_\mathrm{r}]$. Consequently, $f_c(q(\tau)) = 0,\ \tau \in [\tau_\mathrm{s}, \tau_\mathrm{r}]$. For $v(\tau)$, from (8.31) we obtain that:

$$v(\tau_\mathrm{r}) = v(\tau_\mathrm{s}) + \int_{\tau_\mathrm{s}}^{\tau_\mathrm{r}} M(q(\tau))^{-1} n(q(\tau)) a_\mathrm{n} \mathrm{d}\tau. \tag{8.35}$$

Multiplying both sides with $n(q_\mathrm{s})^\top$ from the left and noting that $M^{-1}(q(\tau))n(q(\tau))$ is constant since $q(\tau) = q_\mathrm{s},\ \tau \in [\tau_\mathrm{s}, \tau_\mathrm{r}]$, we have

$$\underbrace{n(q_\mathrm{s})^\top v(\tau_\mathrm{r})}_{=0} - n(q_\mathrm{s})^\top v(\tau_\mathrm{s}) = \underbrace{n(q_\mathrm{s})^\top M(q_\mathrm{s})^{-1} n(q_\mathrm{s})}_{=D(q_\mathrm{s})} \underbrace{\int_{\tau_\mathrm{s}}^{\tau_\mathrm{r}} a_\mathrm{n} \mathrm{d}\tau}_{=:\Lambda_1},$$

$$\Lambda_1 = -\frac{n(q_\mathrm{s})^\top v(\tau_\mathrm{s})}{D(q_\mathrm{s})} > 0.$$

Next, we look at the post-impact states of the CLS and compare it to the solution of the time-freezing system. Since in CLS, $v(t)$ is a function of bounded variation [49], we have that $q(t)$ is a continuous function. Thus, $q(t_\mathrm{s}^+) = q(t_\mathrm{s}^-) = q(t_\mathrm{s})$. Furthermore, notice that $q(t_\mathrm{s}) = q_\mathrm{s}$ which implies

$n(q(t_\mathrm{s}))^\top M^{-1}(q(t_\mathrm{s}))n(q(t_\mathrm{s})) = D(q_\mathrm{s})$. Examining,

$$\begin{aligned} v(t_\mathrm{s}^+) = v(t_\mathrm{s}^-) + \underbrace{\lim_{\substack{\epsilon\to 0\\ \epsilon>0}} \int_{t_\mathrm{s}-\epsilon}^{t_\mathrm{s}+\epsilon} f_\mathrm{v}(q(t), v(t))\mathrm{d}t}_{=0} \\ + \lim_{\substack{\epsilon\to 0\\ \epsilon>0}} \int_{t_\mathrm{s}-\epsilon}^{t_\mathrm{s}+\epsilon} M(q(t))^{-1} n(q(t))\lambda_\mathrm{n}(t)\mathrm{d}t, \end{aligned} \tag{8.36}$$

and multiplying both sides with $n(q_\mathrm{s})^\top$ from the left, introducing $\Lambda_2 \coloneqq \lim_{\substack{\epsilon\to 0\\ \epsilon>0}} \int_{t_\mathrm{s}-\epsilon}^{t_\mathrm{s}+\epsilon} \lambda_\mathrm{n}(t)\mathrm{d}t$, we conclude that

$$\Lambda_2 = -\frac{n(q(t_\mathrm{s}))^\top v(t_\mathrm{s}^-)}{D(q_\mathrm{s})} = \Lambda_1. \tag{8.37}$$

By comparing (8.35) and (8.36), due to the last relation we conclude that $v(\tau_\mathrm{r}) = v(t_\mathrm{s}^+) =: v_\mathrm{s}$. Furthermore, by Proposition 8.8 we have $f_c(q(\tau_\mathrm{r})) = 0$ and $n(q(\tau_\mathrm{r}))^\top v(\tau_\mathrm{r}) = 0$. Since $t'(\tau) = 0$ with $\tau \in [\tau_\mathrm{s}, \tau_\mathrm{r}]$, it follows that $t(\tau_\mathrm{r}) = t(\tau_\mathrm{s}) = t_s$. Consequently,

$$f_c(q(t_\mathrm{s})) = f_c(q(\tau_\mathrm{r})) = 0, \tag{8.38a}$$

$$n(q(t_\mathrm{s}))^\top v(t_\mathrm{s}^+) = n(q(\tau_\mathrm{r}))^\top v(\tau_\mathrm{r}) = 0. \tag{8.38b}$$

Let $y_\mathrm{s} \coloneqq (q_\mathrm{s}, v_\mathrm{s}, t_\mathrm{s})$. Note that $y_\mathrm{sol}(\tau - \tau_\mathrm{r}, y_\mathrm{s}) = y(\tau, y_0)$ for $\tau \in [\tau_\mathrm{r}, \tau_\mathrm{f}]$. Likewise, $x_\mathrm{sol}(t - t_\mathrm{s}, x_\mathrm{s}) = x(t, x_0)$ for $t \in (t_\mathrm{s}, t_\mathrm{f}]$, with $x_\mathrm{s} = Ry_\mathrm{s}$. The two initial value problems are initialized with the same initial condition. Since (8.38) holds we can apply Theorem 8.10 for $y \in \Sigma$. Therefore, by using the arguments of parts II or III (depending on $\varphi(x(\tau), u(\tau))$), we deduce that (8.33a) and (8.33b) hold on $[\tau_\mathrm{r}, \tau_\mathrm{f}]$. Additionally, for $\tau \in (\tau_\mathrm{s}, \tau_\mathrm{r})$ we have $t = t_\mathrm{s}$ and Eq. (8.34) follows directly from (8.37).

Part V (Summary). Parts I-IV cover all possible modes of the CLS and the time-freezing system: evolution according to f_ODE (Part I), evolution on Σ according to f_DAE without leaving it (Part II), leaving Σ and continuing to evolve according to f_ODE (Part III), and the state jump (Part IV). To regard any other possible sequence of mode on $[0, \tau_\mathrm{f}]$, the time interval is simply split into sub-intervals with the different mode, and we apply subsequently the arguments from Parts I-IV to verify that (8.33a) and (8.33b) hold for $t \neq t_\mathrm{s}$ and (8.34) for $t = t_\mathrm{s}$. This completes the proof. □

Equivalence for several subsequent impacts is trivially obtained by sequentially applying the argument of the last theorem. Time-freezing enables one to make

the state jump in "slow motion". By plotting the state as a function of physical time we make the "slow" transition "infinitely fast" and recover the discontinuity in time. More formally, this is encapsulated in (8.34), which shows that the integral of a Dirac impulse $\lambda_{\mathrm{n}}(t)$ is the same as the integral of the v-state of the auxiliary ODE over a finite time interval $[\tau_{\mathrm{s}}, \tau_{\mathrm{r}}]$ of nonzero length. To simulate a time-freezing system with Zeno's effect, the numerical time horizon would have to be infinitely long, as every state jump requires $\tau_{\mathrm{jump}} > 0$. In this thesis, we assume to have a finite number of impacts. In practical robotics applications, one is usually interested in solutions with a finite number of impacts.

Possible extensions

In this thesis, we consider a single unilateral constraint. To extend the same ideas for multiple and/or simultaneous impacts one must take care of the chosen impact model [49]. The extension can be made in several ways, e.g., time continues to flow when the first normal velocity component reaches zero or when all of them reach zero. Some multiple impact models suffer from nonuniqueness of solutions [49, 255] and how to proceed is a modeling decision. In the special case when the constraints, e.g., $f_{c,i}(q)$ and $f_{c,j}(q)$ are orthogonal in the kinetic metric, i.e., $f_{c,i}(q)^\top M(q)^{-1} f_{c,j}(q) = 0$, the impacts can be treated independently [49]. In this case, one would take for every constraint the auxiliary dynamics from Prop. 8.8 and the time would continue to flow when all normal velocity components reach zero. Stochastic approaches with multiple outcomes are also possible, e.g., within the binary collision law as shown in [200] or within a stochastic version of Routh's impact model [129].

8.4.3 Frictional impact

If friction is present at the contact point, frictional impulses cause state jumps in the tangential directions. This section extends the time-freezing reformulation for CLS with frictional impacts. Appropriate auxiliary dynamics for the state jumps in the tangential directions are introduced. The time-freezing system covers both stick and slip motions. We regard the CLS with friction from (8.1), but with a single unilateral constraint.

Stick and slip dynamics of the CLS

For a given λ_n the solution map of the convex optimization problem (8.1e)-(8.1f) is given by [198]:

$$\lambda_\mathrm{t} \in \begin{cases} \{-\mu\lambda_\mathrm{n}\frac{v_\mathrm{t}}{\|v_\mathrm{t}\|_2}\}, & \text{if } \|v_\mathrm{t}\|_2 > 0, \\ \{\tilde{\lambda}_\mathrm{t} \mid \|\tilde{\lambda}_\mathrm{t}\|_2 \leq \mu\lambda_\mathrm{n}\}, & \text{if } \|v_\mathrm{t}\|_2 = 0. \end{cases} \tag{8.39}$$

In the 2D case, this solution map simplifies to $\lambda_\mathrm{t} \in -\mu\lambda_\mathrm{n}\mathrm{sign}(v_\mathrm{t})$. When the system is in contact, it can be in *slipping motion*, i.e., it has nonzero tangential velocity $v_\mathrm{t} \neq 0$, or in *sticking motion* with $v_\mathrm{t} = 0$. Similar to the frictionless case, we derive equivalent ODEs which model the stick and slip dynamics during contact phases. If the system is in slip motion and $\lambda_\mathrm{n} > 0$, it follows from (8.39) that $\lambda_\mathrm{t} = -\mu\lambda_\mathrm{n}\frac{v_\mathrm{t}}{\|v_\mathrm{t}\|_2}$ and we have the DAE of index 3

$$\begin{aligned} \dot{q} &= v, \\ \dot{v} &= f_\mathrm{v}(q,v,u) + M(q)^{-1}\Big(n(q) - B(q)\mu\frac{v_\mathrm{t}}{\|v_\mathrm{t}\|_2}\Big)\lambda_\mathrm{n}, \\ 0 &= f_c(q), \end{aligned}$$

Similar to Eq. (8.27), we perform index reduction to obtain an equivalent ODE:

$$\dot{q} = v, \tag{8.40a}$$

$$\dot{v} = f_\mathrm{v}(q,v,u) - M(q)^{-1}\frac{\varphi(x,u)}{n(q)^\top M(q)^{-1}(n(q) - \mu B(q)\frac{v_\mathrm{t}}{\|v_\mathrm{t}\|_2})}\Big(n(q) - B\mu\frac{v_\mathrm{t}}{\|v_\mathrm{t}\|_2}\Big). \tag{8.40b}$$

The r.h.s. of this ODE is compactly denoted by $f_\mathrm{Slip}(x,u)$.

If the system is in sticking motion we have $v_\mathrm{t} = B^\top v = 0$ and $n^\top v = 0$. By differentiating these equations w.r.t. time we can explicitly compute the multipliers $\lambda \coloneqq (\lambda_\mathrm{n}, \lambda_\mathrm{t})$ by

$$\lambda = -\tilde{D}(q)^{-1}\tilde{\varphi}(x,u), \tag{8.41a}$$

$$\tilde{D}(q) = \begin{bmatrix} D_{n,n}(q) & D_{n,B}(q) \\ D_{B,n}(q) & D_{B,B}(q) \end{bmatrix} = \begin{bmatrix} n(q)^\top M(q)^{-1}n(q) & n(q)^\top M(q)^{-1}B(q) \\ B(q)^\top M(q)^{-1}n(q) & B(q)^\top M(q)^{-1}B(q) \end{bmatrix} \tag{8.41b}$$

$$\tilde{\varphi}(x,u) = \begin{bmatrix} n(q) & B(q) \end{bmatrix}^\top f_v(q,v,u) + \nabla_q(\begin{bmatrix} n(q) & B(q) \end{bmatrix}^\top v)^\top v. \tag{8.41c}$$

The ODE describing the sticking motion reads as

$$\dot{q} = v, \tag{8.42a}$$

$$\dot{v} = f_{\mathrm{v}}(q, v, u) - M(q)^{-1} \begin{bmatrix} n(q) & B(q) \end{bmatrix} \tilde{D}(q)^{-1} \tilde{\varphi}(x, u). \tag{8.42b}$$

Its r.h.s. is compactly denoted by $f_{\mathrm{Stick}}(x, u)$. The transition from the stick to the slip mode occurs when the other tangential forces are greater than the maximal friction force λ_{t}, cf. [49, Chapter 5].

Time-freezing for frictional impacts in the 2D case

Denote the single column of $B(q)$ by $b(q)$. Depending on the sign of v_{t}, we define an auxiliary dynamical system to mimic the state jump in the tangential direction $b(q)$. For the $n(q)$-direction we use the dynamics from Proposition 8.8. For the tangential direction and $b(q)^\top v < 0$ we define the *tangential auxiliary dynamics* analogously:

$$y' = f^-_{\mathrm{aux,t}}(y) \coloneqq \begin{bmatrix} \mathbf{0}_{n_q,1} \\ M(q)^{-1} b(q) a_{\mathrm{t}} \\ 0 \end{bmatrix}. \tag{8.43}$$

To account for the sign of the tangential velocity, for $b(q)^\top v > 0$ we use $y' = f^+_{\mathrm{aux,t}}(y) \coloneqq -f^-_{\mathrm{aux,t}}(y)$. Depending on the sign of v_{t}, one of these ODE is active for the same numerical time interval of the length τ_{jump} as $y' = f_{\mathrm{aux,n}}(y)$. Furthermore, we know from Eq. (8.34) in Theorem 8.12 that the impulse bringing the normal velocity $n(q)^\top v < 0$ to zero after an impact is proportional to $a_{\mathrm{n}} \tau_{\mathrm{jump}}$. Thus, by settings $a_{\mathrm{t}} = \mu a_{\mathrm{n}}$, we conclude that the integrals of the auxiliary dynamics satisfy the maximum dissipation principle, i.e., $a_{\mathrm{t}} \tau_{\mathrm{jump}} = -\mu a_{\mathrm{n}} \tau_{\mathrm{jump}} \mathrm{sign}(b(q)^\top v)$.

State jumps in both the normal and tangential directions are treated simultaneously with different auxiliary dynamics. They should be active whenever $y \in Q \coloneqq \{y \in \mathbb{R}^{n_y} \mid c_1(y) < 0, c_2(y) < 0\}$, cf. Fig. 8.6. To treat different signs of the tangential velocity we introduce the switching function $c_3(y) = b(q)^\top v$. Hence, we have in total $n_f = 3$ regions, one for the unconstrained dynamics and two to mimic the state jumps. We extend the definition of the regions in Eq. (8.28) as follows:

$$R_{\mathrm{ODE}} = \{y \in \mathbb{R}^{n_y} \mid c_1(y) > 0\} \cup \{y \in \mathbb{R}^{n_y} \mid c_1(y) < 0, c_2(y) > 0\}, \tag{8.44a}$$

$$R^+_{\mathrm{aux}} = Q \cap \{y \in \mathbb{R}^{n_y} \mid c_3(y) > 0\}, \tag{8.44b}$$

$$R^-_{\mathrm{aux}} = Q \cap \{y \in \mathbb{R}^{n_y} \mid c_3(y) < 0\}. \tag{8.44c}$$

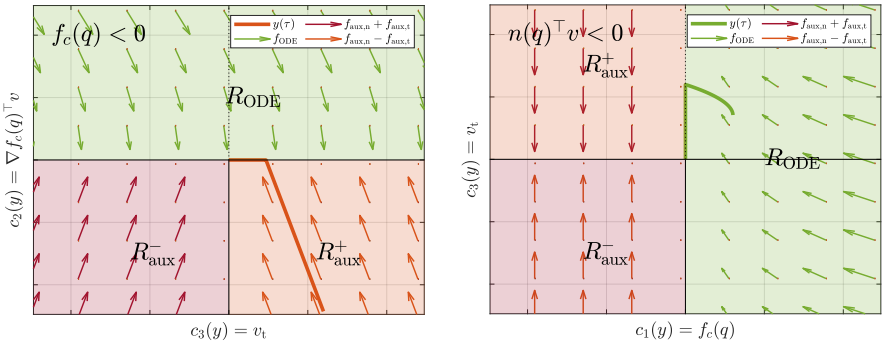


Figure 8.9: Two projections of a phase plot of the time-freezing system from Definition 8.13. The red shaded areas are infeasible for the CLS (8.1) and show the vector fields of the combined auxiliary dynamics. The trajectories of the auxiliary dynamics (red lines) flow in the red-shaded area.

The sum of the corresponding auxiliary dynamics accounts for the simultaneous state jumps, i.e., $f_2(y) = f_{\text{aux,n}}(y) + f^+_{\text{aux,t}}(y)$, $f_3(y) = f_{\text{aux,n}}(y) + f^-_{\text{aux,t}}(y)$. The time-freezing system for the CLS (8.1) is given in the next definition.

Definition 8.13 (Time-freezing system with friction)**.** *Let $\tau \in \mathbb{R}$ be the numerical time, $y(\tau) \coloneqq (x(\tau), t(\tau)) \in \mathbb{R}^{n_y}$ the differential states and $u(\tau) \in \mathbb{R}^{n_u}$ a given control function. The time-freezing PSS is defined by the regions in Eq.* (8.44) *with*

$$f_1(y,u) = (f_{\text{ODE}}(x,u), 1), \text{ for } y \in R_{\text{ODE}}$$

$$f_2(y) = f_{\text{aux,n}}(y) + f^+_{\text{aux,t}}(y), \text{ for } y \in R^+_{\text{aux}},$$

$$f_3(y) = f_{\text{aux,n}}(y) + f^-_{\text{aux,t}}(y), \text{ for } y \in R^-_{\text{aux}}.$$

The corresponding Filippov system, which we call the time-freezing system, is denoted by

$$y' \in F_{\text{TF}}(y,u) = \{\theta_1 f_1(y,u) + \theta_2 f_2(y) + \theta_3 f_3(y) \mid \theta \geq 0, e^\top \theta = 1\}. \tag{8.45}$$

It is assumed that appropriate dynamics $f_{\text{aux,n}}(y), f^+_{\text{aux,t}}(y)$ and $f^-_{\text{aux,t}}(y)$ exist.

Figure 8.9 shows two projections of the state space of the extended time-freezing system. Note that sliding modes can happen on $v_{\text{t}} = 0$, which corresponds to the stick mode. As in the frictionless case, we are interested in the relation of the CLS in contact mode and the corresponding sliding mode of the time-freezing system on Σ.

Theorem 8.14 (Slip-stick sliding mode)**.** *Suppose that the auxiliary dynamics from Proposition 8.8 and Eq.* (8.43) *are used in the time-freezing system from Definition 8.13. Let $y(\tau)$ be a solution of this system with $y(0) \in \Sigma$ and $\tau \in [0, \tau_{\mathrm{f}}]$. Suppose that $\varphi(x(\tau), u(\tau)) \leq 0$ for all $\tau \in [0, \tau_{\mathrm{f}}]$ (persistent contact), then the following statements are true:*

(i) If $v_{\mathrm{t}} \neq 0$ (slip motion), then the sliding mode dynamics are given by $y' = \gamma_{\mathrm{Slip}}(x,u)(f_{\mathrm{Slip}}(x,u), 1)$, where

$$\gamma_{\mathrm{Slip}}(x,u) = \frac{(D_{n,n}(q) - \operatorname{sign}(v_{\mathrm{t}})\mu D_{n,B}(q))a_{\mathrm{n}}}{(D_{n,n}(q) - \operatorname{sign}(v_{\mathrm{t}})\mu D_{n,B}(q))a_{\mathrm{n}} - \varphi(x,u)}. \tag{8.46}$$

(ii) If $v_{\mathrm{t}} = 0$ (stick motion), then the sliding mode dynamics are given by $y' = \gamma_{\mathrm{Slip}}(x,u)(f_{\mathrm{Stick}}(x,u), 1)$, where

$$\gamma_{\mathrm{Stick}}(x,u) = \quad -\frac{(D_{B,B}D_{n,n} - D_{B,n}D_{n,B})a_{\mathrm{n}}}{D_{B,B}\hat{\varphi}_1 - D_{n,B}\hat{\varphi}_2 - D_{B,B}D_{n,n}a_{\mathrm{n}} + D_{B,n}D_{n,B}a_{\mathrm{n}}}. \tag{8.47}$$

The functions $\gamma_{\mathrm{Slip}}(x,u), \gamma_{\mathrm{Stick}}(x,u) \in (0,1]$ are time-rescaling factors.

Proof. In the slip mode in the 2D case, we have that v_{t} is either strictly positive or negative. We start with the case of $v_{\mathrm{t}} = b(q)^\top v > 0$. This means that $y \notin R^-_{\mathrm{aux}}$ and it follows that $\theta_3 = 0$. We follow similar lines as in the proof of Theorem 8.10. From the conditions $c_2(y) = n(q)^\top v = 0$ and $\theta_1 + \theta_2 = 1$ we can compute θ_1 and θ_2. By using $\frac{\mathrm{d}}{\mathrm{d}\tau}c_2(y) = 0$ and Definition 8.13 we compute that

$$0 = \left[\nabla_q(n(q)^\top v)^\top n(q)^\top\right]\left(\theta_1 \begin{bmatrix} v \\ f_v(q,v,u) \end{bmatrix} + \theta_2 \begin{bmatrix} 0 \\ M(q)^{-1}(n(q) - \mu b(q))a_{\mathrm{n}} \end{bmatrix}\right),$$

$$0 = \theta_1 \underbrace{\nabla_q(n(q)^\top v)^\top v + n(q)^\top f_v(q,v,u)}_{=\varphi(x,u)<0} + \theta_2 n(q)^\top M(q)^{-1}(n(q) - \mu b(q))a_{\mathrm{n}}.$$

Using the notation from (8.41b) we can write a linear system in θ

$$\begin{bmatrix} \varphi(x,u) & (D_{n,n}(q) - \mu D_{n,B}(q))a_{\mathrm{n}} \\ 1 & 1 \end{bmatrix}\begin{bmatrix}\theta_1 \\ \theta_2\end{bmatrix} = \begin{bmatrix}0 \\ 1\end{bmatrix},$$

and by solving it we have that

$$\theta_1 = \frac{(D_{n,n}(q) - \mu D_{n,B}(q))a_{\mathrm{n}}}{\psi^+(x,u)},\ \theta_2 = -\frac{\varphi(x,u)}{\psi^+(x,u)}, \theta_3 = 0. \tag{8.48}$$

where $\psi^+(x,u) = (D_{n,n}(q) - \mu D_{n,B}(q))a_{\mathrm{n}} - \varphi(x,u)$ is an auxiliary function introduced for rotational compactness. Next, we plug in these expression into (8.45), compare to (8.40) (for $v_{\mathrm{t}} > 0$) and obtain

$$y' = \frac{(D_{n,n}(q) - \mu D_{n,B}(q))a_{\mathrm{n}}}{\psi^+(x,u)} \begin{bmatrix} v \\ f_v(q,v,u) \\ 1 \end{bmatrix} + \frac{-\varphi(x,u)}{\psi^+(x,u)} \begin{bmatrix} \mathbf{0}_{n_q,1} \\ M(q)^{-1}(n(q) - \mu b(q))a_{\mathrm{n}} \\ 0 \end{bmatrix} \frac{D_{n,n}(q) - \mu D_{n,B}(q)}{D_{n,n}(q) - \mu D_{n,B}(q)},$$

$$y' = \theta_1(f_{\mathrm{Slip}}(x,u), 1).$$

By following the same steps for $b(q)^\top v < 0$ we have that

$$\theta_1 = \frac{(D_{n,n}(q) + \mu D_{n,B}(q))a_{\mathrm{n}}}{\psi^-(x,u)},\ \theta_2 = 0,\ \theta_3 = -\frac{\varphi(x,u)}{\psi^-(x,u)}, \tag{8.49}$$

where $\psi^-(x,u) = (D_{n,n}(q) + \mu D_{n,B}(q))a_{\mathrm{n}} - \varphi(x,u)$. Following the same steps we conclude that and $y' = \theta_1(f_{\mathrm{Slip}}(x,u), 1)$. By combining these two cases and the expressions for θ_1 in (8.48) and (8.49) we obtain the formula for $\gamma_{\mathrm{Slip}}(x,u)$ in Eq. (8.46). This completes the first part of the proof.

In the second part, we have $v_{\mathrm{t}} = b(q)^\top v = 0$. Together with the assumption that $y \in \Sigma$ it follows no θ_i can be set to be zero a priori. As in the previous cases, we compute the analytic expressions for the convex multipliers θ. First, we introduce the change of variables $\beta_1 = \theta_2 + \theta_3$ and $\beta_2 = -\theta_2 + \theta_3$. This allows us to rewrite (8.45) in the following compact form:

$$y' = \theta_1 \begin{bmatrix} v \\ f_v(q,v,u) \\ 1 \end{bmatrix} + \begin{bmatrix} \mathbf{0}_{n_q,1} \\ M(q)^{-1} \begin{bmatrix} n(q) & b(q) \end{bmatrix} A \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} \\ 0 \end{bmatrix}, \tag{8.50}$$

where $A = \begin{bmatrix} a_{\mathrm{n}} & 0 \\ 0 & \mu a_{\mathrm{n}} \end{bmatrix}$. Using this equation and $\frac{\mathrm{d}}{\mathrm{d}\tau} \begin{bmatrix} n(q)^\top & b(q)^\top \end{bmatrix} v = 0$ one obtains:

$$\begin{aligned} 0 = \Big[\nabla_q(\begin{bmatrix} n(q)^\top & b(q)^\top \end{bmatrix} v)^\top \quad \begin{bmatrix} n(q) & b(q) \end{bmatrix}^\top \Big] \cdot \Big(\begin{bmatrix} v \\ f_v(q,v,u) \\ 1 \end{bmatrix} \\ + \begin{bmatrix} \mathbf{0}_{n_q,1} \\ M(q)^{-1} \begin{bmatrix} n(q) & b(q) \end{bmatrix} A \begin{bmatrix} a_{\mathrm{n}}\beta_1 \\ \mu a_{\mathrm{n}}\beta_2 \end{bmatrix} \\ 0 \end{bmatrix} \Big), \end{aligned} \tag{8.51a}$$

$$0 = \hat{\varphi}(x,u)\theta_1 + \hat{D}(q)A\begin{bmatrix}\beta_1\\ \beta_2,\end{bmatrix} \tag{8.51b}$$

In summary, we obtain the linear system:

$$\begin{bmatrix} 1 & 1 & 0 \\ \hat{\varphi}_1(x,u) & a_{\mathrm{n}}D_{n,n}(q) & \mu a_{\mathrm{n}}D_{n,B}(q) \\ \hat{\varphi}_2(x,u) & a_{\mathrm{n}}D_{b,n}(q) & \mu a_{\mathrm{n}}D_{B,B}(q) \end{bmatrix}\begin{bmatrix}\theta_1\\ \beta_1\\ \beta_2\end{bmatrix} = \begin{bmatrix}1\\0\\0\end{bmatrix}.$$

The solution of this linear system reads as:

$$\theta_1 = -\frac{a_{\mathrm{n}}(D_{B,B}(q)D_{n,n}(q) - D_{B,n}(q)D_{n,B}(q))}{\psi^0(x,u)} = \gamma_{\mathrm{Stick}}(x,u),$$

$$\beta_1 = \frac{D_{B,B}(q)\hat{\varphi}_1(x,u) - D_{n,B}(q)\hat{\varphi}_2(x,u)}{\psi^0(x,u)},$$

$$\beta_2 = -\frac{D_{B,n}(q)\hat{\varphi}_1(x,u) - D_{n,n}(q)\hat{\varphi}_2(x,u)}{\psi^0(x,u)}.$$

where $\psi^0(x,u) = D_{B,B}(q)\hat{\varphi}_1(x,u) - D_{n,B}(q)\hat{\varphi}_2(x,u) - D_{B,B}(q)D_{n,n}(q)a_{\mathrm{n}} + D_{B,n}(q)D_{n,B}(q)a_{\mathrm{n}}$. Next, we compute the sliding mode vector field of the time-freezing system $y' = \sum_{i=1}^{3}\theta_i f_i$. Using the last line in (8.51) we have that

$$\begin{bmatrix}\beta_1\\ \beta_2\end{bmatrix} = -A^{-1}\hat{D}(q)^{-1}\hat{\varphi}(x,u).$$

By plugging this into (8.50) we obtain that $y' = \gamma_{\mathrm{Stick}}(x,u)(f_{\mathrm{Stick}}(x,u),1)$. Since $\gamma_{\mathrm{Slip}}(x,u)$ and $\gamma_{\mathrm{Stick}}(x,u)$ correspond to the multiplier θ_1 in sliding modes, it follows that they take values in $(0,1]$. This completes the proof. □

The time-rescaling factors $\gamma_{\mathrm{Stick}}(x,u)$ and $\gamma_{\mathrm{Slip}}(x,u)$ have more complicated algebraic expressions than $\gamma(x,u)$ in (8.32). However, under a simplifying assumption, they have all the same expressions, e.g., when the inertia matrix is diagonal. This assumption holds in the guiding example. We state an immediate consequence of the previous theorem.

Corollary 8.15 (Simplifed time rescaling factors)**.** *Suppose that the assumptions of Theorem 8.14 are satisfied. Furthermore, assume that the unit normal $n(q)$ and tangential vector $b(q)$ are orthogonal in the kinetic metric, i.e., $n(q)M(q)^{-1}b(q) = 0$, then the conclusions of Theorem 8.14 hold with $\gamma_{\mathrm{Slip}}(x,u) = \gamma_{\mathrm{Stick}}(x,u) = \gamma(x,u)$.*

These results generalize Theorem 8.10, and one can see that the sliding mode dynamics match the slip or stick dynamics of the CLS. Finally, we show that CLS with planar contacts, friction, and impacts are equivalent to Filippov systems.

Theorem 8.16 (Solution relationship)**.** *Regard the initial value problems corresponding to: i) the time-freezing system in Definition 8.13 with a given* $y(0) = (q_0, v_0, 0) \in \mathbb{R}^{n_y}$ *and* $f_c(q_0) \geq 0$ *on a time interval* $[0, \tau_\mathrm{f}]$*, ii) the CLS from Eq.* (8.1) *with the initial value* $x(0) = (q_0, v_0) \in \mathbb{R}^{n_x}$ *on a time interval* $[0, t_\mathrm{f}] \coloneqq [0, t(\tau_\mathrm{f})]$*, with* $f_c(q(t_\mathrm{f})) \geq 0$ *and* $n(q(t_\mathrm{f}))^\top v(t_\mathrm{f}) \geq 0$*. Suppose the following assumptions hold:*

(a) the auxiliary dynamics $f_{\mathrm{aux,n}}(y)$ *from Proposition 8.8 and* $f^-_{\mathrm{aux},t}(y)$*,* $f^+_{\mathrm{aux},t}(y)$ *from Eq.* (8.43) *are used in the time-freezing system in Definition 8.13,*

(b) there is at most one time point $t_\mathrm{s} = t(\tau_\mathrm{s})$ *where* $f_c(q(t_\mathrm{s})) = 0$ *and* $n(q(t_\mathrm{s}))^\top v(t_\mathrm{s}^-) < 0$ *on the time interval* $[0, t_\mathrm{f}]$*,*

Then, the solutions to the two problems are related as follows:

1. *For* $t \neq t_\mathrm{s}$*:*

$$x(t(\tau)) = Ry(\tau), \text{ with } R = \begin{bmatrix} I_{n_x} & \mathbf{0}_{n_x,1} \end{bmatrix},$$

$$\lambda(t(\tau)) = \begin{cases} \lambda_{\mathrm{Slip}}(t(\tau)), & \text{if } \ y \in \Sigma,\ v_\mathrm{t} \neq 0, \\ \lambda_{\mathrm{Stick}}(t(\tau)), & \text{if } \ y \in \Sigma,\ v_\mathrm{t} = 0, \\ 0, & \text{otherwise.} \end{cases}$$

with

$$\lambda_{\mathrm{Slip}} = -\frac{\varphi(x,u)}{n(q)^\top M(q)^{-1}(n(q) - \mu b(q)\mathrm{sign}(v_\mathrm{t}))} \begin{bmatrix} 1 \\ -\mu \mathrm{sign}(v_\mathrm{t}) \end{bmatrix}$$

$$\lambda_{\mathrm{Stick}} = -\tilde{D}(q)^{-1}\tilde{\varphi}(x,u).$$

2. *For* $t = t_\mathrm{s}$*:*

$$\lim_{\substack{\epsilon \to 0 \\ \epsilon > 0}} \int_{t_\mathrm{s}-\epsilon}^{t_\mathrm{s}+\epsilon} \lambda_\mathrm{n}(t)\mathrm{d}t = \int_{\tau_\mathrm{s}}^{\tau_\mathrm{r}} a_\mathrm{n}\mathrm{d}\tau,$$

$$\lim_{\substack{\epsilon \to 0 \\ \epsilon > 0}} \int_{t_\mathrm{s}-\epsilon}^{t_\mathrm{s}+\epsilon} |\lambda_\mathrm{t}(t)|\mathrm{d}t = \int_{\tau_\mathrm{s}}^{\tau_\mathrm{r}} \mu a_\mathrm{n}\mathrm{d}\tau,$$

Proof. Theorem 8.14 is applied for $y \in \Sigma$. Otherwise, the proof follows similar lines as the proof of Theorem 8.12. The absolute value in the last equation accounts for the sign of v_t. □

Time-freezing for frictional impacts in the 3D case

This case is more difficult since we cannot treat different directions of v_t with different auxiliary dynamics as in the planar case. The solution map of (8.39) depends discontinuously on $\|v_\mathrm{t}\|_2$. Hence, we must take it as a switching function. Because the set $\|v_\mathrm{t}\|_2 = 0$ has no interior, we cannot use the Filippov extension from Eq. (6.3), which assumes regions R_i with nonempty interior. More general definitions without multipliers θ as in Eq. (6.2) can treat this case, but they are not computationally useful for in this case, as we see in the next section. To alleviate this difficulty, we propose an approximation for (8.39):

$$\lambda_\mathrm{t} = \begin{cases} -\mu\lambda_\mathrm{n}\frac{v_\mathrm{t}}{\|v_\mathrm{t}\|_2}, & \text{if } \|v_\mathrm{t}\|_2 > \epsilon_\mathrm{t}, \\ v_\mathrm{t}, & \text{if } \|v_\mathrm{t}\|_2 < \epsilon_\mathrm{t}, \end{cases} \tag{8.52}$$

with a small parameter $\epsilon_\mathrm{t} > 0$. This expression is exact for $\|v_\mathrm{t}\|_2 > \epsilon_\mathrm{t}$, thus we can make it arbitrarily accurate. For $\|v_\mathrm{t}\|_2 < \epsilon_\mathrm{t}$, the vector field drives the tangential velocity towards $\|v_\mathrm{t}\|_2 = \epsilon_\mathrm{t}$, see Fig. 8.10. In a Filippov setting, a convex combination of the two cases in (8.52) keeps the velocity at $\|v_\mathrm{t}\|_2 = \epsilon_\mathrm{t}$. Hence, in the sticking mode, we have a velocity drift of ϵ_t. By taking $c_3(y) = \|v_\mathrm{t}\|_2 - \epsilon_\mathrm{t}$ as a switching function, we can define regions with nonempty interiors and the corresponding auxiliary dynamics. The auxiliary dynamics mimicking the behavior of (8.52) read as:

$$f^+_{\mathrm{aux,t}}(y) = \begin{bmatrix} \mathbf{0}_{n_q,1} \\ -M(q)^{-1}B(q)a_\mathrm{t}\frac{v_\mathrm{t}}{\|v_\mathrm{t}\|} \\ 0 \end{bmatrix}, \quad f^-_{\mathrm{aux,t}}(y) = \begin{bmatrix} \mathbf{0}_{n_q,1} \\ M(q)^{-1}B(q)v_\mathrm{t} \\ 0 \end{bmatrix}.$$

The regions for the time-freezing system are defined as in (8.44) and the matching time-freezing system is defined analogously to Definition 8.13. Furthermore, one could derive stick-slip dynamics corresponding to the solution map

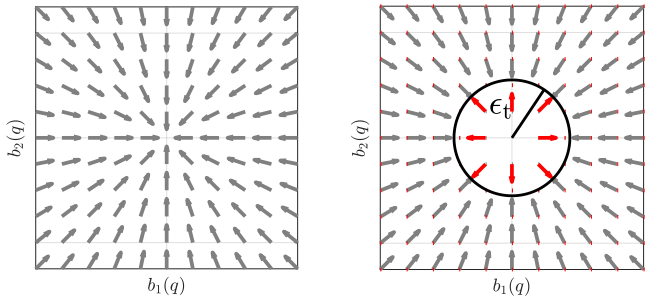


Figure 8.10: The exact friction force Eq. (8.39) (left) and its approximation in Eq. (8.52) (right).

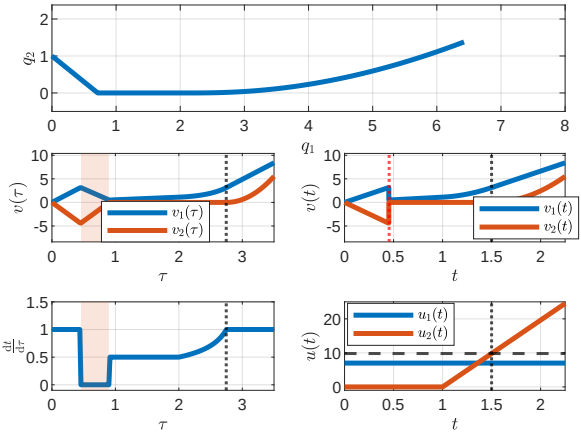


Figure 8.11: Trajectories of the time-freezing system from Example 8.17.

approximation (8.52) and relate it to the time-freezing system by following similar lines as in Theorems 8.14 and 8.16, but we omit the details here. We conclude this section by revisiting Example 8.11, but now with adding friction.

Example 8.17. *(Frictional impact) The time-freezing system model from Example 8.9 is extended by adding friction with a coefficient* $\mu = 0.6$. *In the planar case, we have* $n_\mathrm{t} = 1$, *and the tangent at the contact point is* $b(q) = [1, 0]$. *We have the switching functions* $c_1(y) = q_2$, $c_2(y) = v_2$ *and* $c_3(y) = b(q)^\top v = v_1$. *Following Eq.* (8.44), *the regions of the time-freezing system are* $R_1 = \{y \mid q_2 > 0\} \cup \{y \mid q_2 < 0, v_2 > 0\}$, $R_2 = \{y \mid q_2 < 0, v_2 < 0, v_1 > 0\}$ *and* $R_3 = \{y \mid q_2 < 0, v_2 < 0, v_1 < 0\}$. *The dynamics of the PSS are* $f_1 = (v_1, v_2, u_1, -g + u_2, 1)$, $f_2 = (0, 0, -\mu a_\mathrm{n}, a_\mathrm{n}, 0)$ *and* $f_3 = (0, 0, \mu a_\mathrm{n}, a_\mathrm{n}, 0)$. *The results of the simulation are depicted in Figure 8.11. Note that due to friction there is now also a state jump in the tangential velocity* v_1, *cf. middle plots. Afterwards, the acceleration of* v_1 *is during contact phases smaller due to the friction force. However, the tangential acceleration is increasing over time as the normal contact force becomes weaker because of* u_2. *At* $\tau = 2.8$, *the particle lifts off as in the previous example.*

8.5 Numerical optimal control of time-freezing systems

In this section, we demonstrate how to use time-freezing systems in optimal control and show that its solutions are also optimal for the initial OCP with a CLS. Furthermore, we introduce time transformations and constraints to achieve the desired control grid discretization and final time despite the nonsmooth clock state. All methods and examples from this paper, including a fully-automated reformulation of the CLS into a PSS, are implemented in the open-source tool `nosnoc` [2, 206]. We start by stating the continuous-time optimal control problem (OCP) we wish to solve. Afterwards, we derive an equivalent OCP, now subject to the time-freezing system.

8.5.1 Continuous-time OCP with a CLS

We regard the following continuous-time optimal control problem subject to CLS:

$$\min_{x(\cdot),\lambda(\cdot),u(\cdot),} \quad \Psi(x(T)) \tag{8.53a}$$

$$\text{s.t.} \quad x(0) = \bar{x}_0, \tag{8.53b}$$

$$\text{Eq.(8.1), for a.a. } t \in [0,T] \tag{8.53c}$$

$$0 \leq g(x(t), u(t)),\ t \in [0,T], \tag{8.53d}$$

$$0 \leq r(x(T)), \tag{8.53e}$$

where $\Psi : \mathbb{R}^{n_x} \to \mathbb{R}$ is the terminal cost of the OCP, $\bar{x}_0$ is a given initial value. The functions $g : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_g}$ and $r : \mathbb{R}^{n_x} \to \mathbb{R}^{n_r}$ are the path and terminal constraints, respectively. The CLS dynamics in Eq. (8.53c) (resp. Eq. (8.1)) make this OCP nonsmooth and nonconvex. Without loss of generality, we only consider a terminal cost term here and remind the reader that the integral of a running cost $L : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ over $[0,T]$ can be treated via a terminal cost term by introducing a quadrature state $\ell(t)$

$$\frac{\mathrm{d}}{\mathrm{d}t}\ell(t) = L(x(t), u(t)), t \in [0,T],\ \ell(0) = 0, \tag{8.54}$$

and adding $\ell(T)$ to the objective.

8.5.2 Continuous-time OCP with a time-freezing system

Using the results from the previous sections, we derive now an OCP subject to a time-freezing system. The new OCP is regarded in numerical time $\tau \in [0, \tilde{T}]$. We take four steps in this transformation:

(1) we modify the quadrature state in Eq. (8.54) so that the cost integrated over both numerical and physical time remains unchanged;

(2) we reformulate the time-freezing system into an equivalent dynamic complementarity system to make it possible to apply FESD;

(3) we introduce a time-transformation to ensure that the terminal physical time $t(\tilde{T})$ matches the true control horizon of Eq.(8.53), i.e., $t(\tilde{T}) = T$;

(4) we express the remaining constraints in terms of numerical time.

We start with adapting the objective. This is achieved by replacing the quadrature state (8.54) by:

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\ell(\tau) = \begin{cases} L(x(\tau), u(\tau)), & \text{if } y \in R_{\mathrm{ODE}}, \\ 0, & \text{otherwise.} \end{cases} \tag{8.55}$$

When the time is frozen the cost integral is zero and there are no contributions to the overall objective, i.e., the cost is unchanged when the time is frozen.

Our goal is to apply the FESD method from Chapter 7 to the time-freezing systems. To do so we follow the approach of Chapter 6 and rewrite the time-freezing Filippov systems from Definitions 8.3, 8.7 or 8.7 as dynamic complementarity system. For sake of illustration, we perform this for the time-freezing system from Definition 8.13 and use the step reformulation from Section 6.3. We define $c(y) \coloneqq (c_1(y), c_2(y), c_3(y))$. Following the steps from section 6.3 we can derive from $y' \in F_{\mathrm{TF}}(y, u)$ in Definition 8.13 the equivalent dynamic complementarity system:

$$y' = F(y, u)\ \theta, \tag{8.56a}$$

$$0 = g_{\mathrm{F}}(\theta, \alpha), \tag{8.56b}$$

$$0 = c(y) - \lambda^{\mathrm{p}} + \lambda^{\mathrm{n}}, \tag{8.56c}$$

$$0 \leq \alpha \perp \lambda^{\mathrm{n}} \geq 0, \tag{8.56d}$$

$$0 \leq e - \alpha \perp \lambda^{\mathrm{p}} \geq 0. \tag{8.56e}$$

The matrix $F(y,u) = [f_1(y,u), \dots f_{n_f}(y)] \in \mathbb{R}^{n_y \times n_f}$ collects the modes of the PSS and $\theta = (\theta_1, \dots, \theta_{n_f})$. The last three lines are the KKT conditions of (6.33), where $\lambda^{\mathrm{n}}, \lambda^{\mathrm{p}} \in \mathbb{R}^3$ are the Lagrange multipliers for the lower and upper bounds in (6.33), respectively. We group all algebraic variables of the DCS in the vector $z = (\theta, \alpha, \lambda^{\mathrm{p}}, \lambda^{\mathrm{n}})$. The function g_{F} relates the Filippov multipliers θ with the evaluations of the step functions α:

$$g_{\mathrm{F}}(\theta, \alpha) := \begin{bmatrix} \theta_1 - \alpha_1 + (1-\alpha_1)\alpha_2, \\ \theta_2 - (1-\alpha_1)(1-\alpha_2)(1-\alpha_3) \\ \theta_3 - (1-\alpha_1)(1-\alpha_2)(\alpha_3) \end{bmatrix}. \tag{8.57}$$

These expressions correspond to the signs of $c_j(y)$ in the definitions of the regions $R_{\mathrm{ODE}}, R^+_{\mathrm{aux}}, R^-_{\mathrm{aux}}$, cf. Section 6.3.

The time-freezing system evolves over $\tau \in [0, \tilde{T}]$. During state jumps the physical time evolution is stopped. As a consequence, we have that $t(\tilde{T}) < T$, i.e., the terminal physical time in the time-freezing problem does not match the desired time T. To resolve this, we introduce a time-transformation variable $s(\tau) \in \mathbb{R}$ and impose the terminal constraint on the clock state $t(\tilde{T}) = T$. Consequently, we obtain $t' = s$ and $s > 1$ speeds up the physical time and allows us to catch up and reach the desired time T, cf. Example 8.19. Such time transformations are very common in optimal control when one wants to optimize over the terminal time, cf. Section 3.3.

It is left to impose the path (8.53d) and terminal constraints (8.53e) in numerical time for $x(\tau)$ and $u(\tau)$. Finally, the OCP subject to the time-freezing system reads as:

$$\min_{\substack{y(\cdot), z(\cdot), \\ u(\cdot), s(\cdot)}} \quad \Psi(x(\tilde{T})) \tag{8.58a}$$

$$\text{s.t.} \quad x(0) = \bar{x}_0,\ t(0) = 0, \tag{8.58b}$$

$$y'(\tau) = s(\tau) F(y(\tau), u(\tau)) \theta(\tau), \tau \in [0, \tilde{T}], \tag{8.58c}$$

$$0 = g_{\mathrm{F}}(\theta(\tau), \alpha(\tau)), \tau \in [0, \tilde{T}], \tag{8.58d}$$

$$0 = c(y(\tau)) - \lambda^{\mathrm{p}}(\tau) + \lambda^{\mathrm{n}}(\tau), \tau \in [0, \tilde{T}], \tag{8.58e}$$

$$0 \le \alpha(\tau) \perp \lambda^{\mathrm{n}}(\tau) \ge 0, \tau \in [0, \tilde{T}], \tag{8.58f}$$

$$0 \le e - \alpha(\tau) \perp \lambda^{\mathrm{p}}(\tau) \ge 0, \tau \in [0, \tilde{T}], \tag{8.58g}$$

$$0 \le g(x(\tau), u(\tau)),\ t \in [0, T], \tag{8.58h}$$

$$0 \leq r(x(\tilde{T})), \tag{8.58i}$$

$$t(\tilde{T}) = T. \tag{8.58j}$$

It is important to note, that when the time is frozen ($t' = 0$) the control $u(\tau)$ does not influence $x(\tau)$, since the auxiliary dynamics do not depend on the control, cf. Eq (8.31). Therefore, we could even omit the path constraints whenever $t' = 0$, but we keep it for notational simplicity. Additionally, the integral of the stage cost remains unchanged, since $\frac{\mathrm{d}}{\mathrm{d}\tau}\ell(\tau) = 0$ in this case, cf. (8.55).

Next, we show that the optimal controls obtained by solving the initial OCP (8.53), with appropriate modifications, are also optimal for (8.58). Let $u^*(t), t \in [0, T]$ be an optimal control of (8.53). We construct an $\tilde{u}^*(\tau)$, $t \in [0, \tilde{T}]$ as follows. It can be seen that, when the physical time is evolving ($t' > 0$), we can find the inverse function $\tau^{-1}(t)$ to find the corresponding numerical time τ. We construct a control function for the time-freezing system:

$$\tilde{u}^*(\tau) = \begin{cases} u(\tau^{-1}(t)), & \text{for } t(\tau)' > 0 \\ \hat{u}(\tau), & \text{for } t(\tau)' = 0, \end{cases} \tag{8.59}$$

where $\hat{u}(\tau)$ is any function such that $g(x(\tau), \hat{u}(\tau)) \geq 0$ holds, whenever $t'(\tau) = 0$. Recall that $\hat{u}(\tau)$ does not change the objective nor it changes $x(\tau)$, its only purpose is to extend $u(t)$ to intervals when the time is frozen. For example, we can choose a constant value that does not violate the path constraints. With (8.59) we can extend $u^*(t)$ for the time interval where the physical time is frozen. Conversely, given an optimal control $\tilde{u}^*(\tau)$ of Eq. (8.58), then we expect $u^*(t(\tau))$ to be optimal for (8.53).

Proposition 8.18. *Let $\tilde{u}^*(\tau)$, $\tau \in [0, \tilde{T}]$ be an optimal control obtained by solving the OCP* (8.58)*. Then $u^*(t) = \tilde{u}^*(t(\tau))$, $t \in [0, T]$ is an optimal control of the OCP* (8.53)*. Conversely, let $u^*(t)$, $t \in [0, T]$ be an optimal control of the OCP* (8.53)*, then the control function $\tilde{u}^*(\tau), \tau \in [0, \tilde{T}]$ obtained via Eq.* (8.59) *is optimal for* (8.58)*.*

Proof. For a fixed control function $u(\tau)$ and $s(\tau)$ such that $t(\tilde{T}) = T$, the time-freezing system (8.58c)- (8.58g) and the CLS (8.1) with $u(t(\tau))$ are equivalent in the sense of Theorem 8.16. Thus, a feasible $y(\tau)$ in (8.58) results in a feasible $x(t)$ in (8.53). Due to equation (8.55), both OCPs have the same objective value. Consequently, given a $\tilde{u}(\tau) = \tilde{u}^*(\tau) + \delta\tilde{u}(\tau)$ that improves the objective (8.58a), the corresponding $\tilde{u}(t(\tau))$ would also improve (8.53a). Conversely, for every modified $u(t(\tau)) = u^*(t(\tau)) + \delta u(t(\tau))$ that improves the objective (8.53a), we can construct an appropriate control function $u(\tau)$ via (8.59) that improves (8.58a). Thus, $u^*(t) = \tilde{u}^*(t(\tau))$ is optimal for (8.53). The converse is proved by similar arguments. □

8.5.3 Discrete-time OCP with the time-freezing system

In principle, one can discretize the OCP (8.53) by using any time-stepping integration method for CLS [4, 49, 255] e.g., the Stewart-Trinkle method [252]. Such an approach for direct optimal control was used in [225]. As discussed in Chapter 4, standard time-stepping methods for CLS with friction (8.1) have at best first-order accuracy [4, 255]. Moreover, the numerical sensitivities obtained from such a discretization are always wrong and the NLP solvers converge to spurious solutions, cf. Chapter 5. Therefore, for a moderately accurate solution usually, a large computational effort is needed. However, FESD for Filippov systems does not suffer from these limitations. In conclusion, the fundamental limitations of standard direct optimal control methods are resolved by combining time-freezing and FESD. This enables one to find a more accurate solution approximation for the continuous-time OCP (8.53) by solving (8.58).

We proceed by introducing the discrete-time version of the OCP (8.58) with a multiple shooting-type discretization [44]. The numerical time horizon $[0, \tilde{T}]$ is split into N control intervals $[\tau_k, \tau_{k+1}]$ of equal length [213]. The controls are assumed to be constant over every interval, i.e., $u(\tau) = u_k, \tau \in [\tau_k, \tau_{k+1}], k = 0, \ldots, N-1$, and $y_k = (x_k, t_k) \in \mathbb{R}^{n_y}$ is the discrete-time approximation of the time-freezing state, i.e., $x_k \approx x(\tau_k)$ $t_k \approx t(\tau_k)$. The vectors z_k collect all algebraic and internal integration variables for the $k-$th control interval. The vector $w \coloneqq (y_0, z_0, u_0, s_0, \ldots, y_{N-1}, z_{N-1}, u_{N-1}, s_{N-1}, y_N)$ groups all optimization variables.

Our goal is to have an equidistant control grid, as this is typically required in feedback control applications. It is important to note that, due to intervals with frozen physical time evolution ($t' = 0$), an equidistant grid in numerical time $\{\tau_0, \ldots, \tau_N\}$ does not imply an equidistant grid in physical time $\{t_0, \ldots, t_N\}$. To address this issue, we propose to use a piecewise constant discretization of the time-transformation variable $s(\tau)$, i.e., we have $s_k \in \mathbb{R}, k = 0, \ldots, N-1$. Additionally, we introduce the constraints $t_k = k\frac{T}{N}$, $k = 0, \ldots, N$, cf. Eq. (8.60f) below. It is worth noting that for $k = N$, we have the discrete-time versions of the terminal clock constraint (8.58j). The steps above result in an equidistant control discretization grid in physical time, i.e., $u(t) = u_k$ for $t \in [t_k, t_{k+1}]$ with $t_0 = 0$ and $t_k = t_{k-1} + T/N$. This is further illustrated in Example 8.19.

The discretization of (8.58) reads as:

$$\min_{w} \ \Psi(x_N) \tag{8.60a}$$

$$\text{s.t.} \quad x_0 = \bar{x}_0, \tag{8.60b}$$

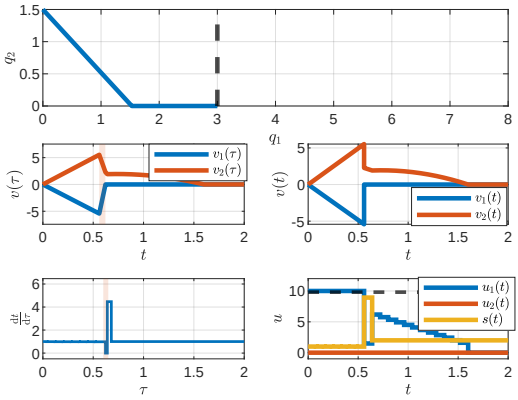


Figure 8.12: Solution to the guiding optimal control example.

$$y_{k+1} = \Phi_f(y_k, z_k, u_k, s_k),\ k = 0,\ldots,N-1, \tag{8.60c}$$

$$0 = \Phi_{\text{int}}(y_k, z_k, u_k),\ k = 0,\ldots,N-1, \tag{8.60d}$$

$$0 \leq \Phi_{\text{c},1}(z_k) \perp \Phi_{\text{c},2}(z_k) \geq 0,\ k = 0,\ldots,N-1, \tag{8.60e}$$

$$t_k = k\frac{T}{N},\ k = 0,\ldots,N, \tag{8.60f}$$

$$1 \leq s_k \leq \bar{s},\ k = 0,\ldots,N-1, \tag{8.60g}$$

$$0 \leq g(x_k, u_k),\ k = 0,\ldots,N-1, \tag{8.60h}$$

$$0 \leq r(x_N). \tag{8.60i}$$

It is common in direct optimal control to write discretization method equations in a compact discrete-time system manner, e.g., as done in Chapters 3 and 7). We do here in Eq. (8.60c)-(8.60e). The function $\Phi_f : \mathbb{R}^{n_y} \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \times \mathbb{R} \to \mathbb{R}^{n_x}$ is the discrete-time state transition map which approximates $y(\tau)$. The function $\Phi_{\text{int}} : \mathbb{R}^{n_y} \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_\Phi}$ collects all internal computations of the underlying integration scheme. The constraints (8.60e) arise from the discretization of the complementarity conditions (8.58f)-(8.58g). These functions are obtained via the FESD discretization, cf. Chapter 7. The constraint (8.60g) bounds s_k, where $\bar{s}$ is its maximal value that has to be sufficiently large to ensure feasibility of (8.60f). We finish this section with a simple OCP derived from the guiding example.

Example 8.19. *(OCP Example) We solve an OCP of the form of* (8.53) *with the guiding example. The initial value is unchanged, i.e.,* $y_0 = (0, 1, 0, 0, 0)$. *The particle should reach at* $t(\tilde{T}) = 2$, *with* $\tilde{T} = 2$, *the position* $q(T) = (3, 0)$ *with zero terminal velocity* $v(T) = (0, 0)$. *We bound the horizontal thrust force* $|u_1| \leq 10$ *and set for simplicity* $u_2 = 0$. *The ball should reach the goal with minimum control effort, which is modeled with the stage cost* $L(x, u) = u_1^2$. *We take* $N = 20$ *control intervals and discretize the OCP equivalent time-freezing OCP with a third-order FESD-Radau IIA scheme with three integration steps on every control interval, cf. Chapter 7. The solution is depicted in Fig. 8.12. We can see that maximum force is applied before the impact since there is still no friction and the motion is cheaper. After the impact, a smaller control force is applied just to reach the target. Note that the* s_k *(yellow line in the bottom right plot) is higher during the control interval when the state jumps happens (to catch up the frozen time) and* $s_k = 2$ *during contact to compensate for the slow down due to* $\gamma(x, u)$. *The resulting speed of time is always one (bottom left plot), except when the state jump happens where a speed-up is needed to compensate for the frozen time. This ensures an equidistant control grid (as intended with the constraint* (8.60f)*) and* $t(\tilde{T}) = 2$ *as desired.*

8.6 Numerical examples with time-freezing

In `nosnoc`'s repository [2, 206], there are several simulation and optimal control examples with time-freezing. For brevity, we have selected three numerical optimal control examples to illustrate the theoretical and algorithmic developments presented in this chapter. In all examples, we initialize the controls with zero, and for the states, we take the provided initial value for all discrete time points. Despite non-instructive starting points, the optimizer finds creative solutions after solving only a few NLPs in the homotopy loop.

8.6.1 Ball inside a box - elastic impacts

In this example, we regard a two-dimensional ball inside a box without gravity. The model equations are given in Example (8.1), and the impacts are fully elastic, i.e., $\epsilon_r = 1$. The goal is to track a reference, which moves on a circle centered at the origin with a radius $R = 1$, with the angular velocity ω, and initial angle $\alpha_0 = \frac{\pi}{4}$. The box is not concentric with the circle and is defined by four gap functions, which model the bottom, right, top, and left side, respectively: $f_c(q) = (q_2 - 1.05R, -q_1 + 1.1R, -q_2 + 1.1R, q_1 + 1.15R)$. For every constraint,

Row of S	**Description**	**Dynamics**
1	unconstrained dynamics	$(v_1, v_2, u_1, u_2, 0)$
2	impact with the bottom wall	$f_{\text{bottom}}(y)$
3	impact at the bottom right corner	$f_{\text{bottom}}(y) + f_{\text{right}}(y)$
4	impact with the right wall	$f_{\text{right}}(y)$
5	impact with the top right corner	$f_{\text{right}}(y) + f_{\text{top}}(y)$
6	impact with top wall	$f_{\text{top}}(y)$
7	impact with top left corner	$f_{\text{top}}(y) + f_{\text{left}}(y)$
8	impact with left wall with	$f_{\text{left}}(y)$
9	impact with the bottom left corner	$f_{\text{left}}(y) + f_{\text{bottom}}(y)$

Table 8.2: Summary of the time-freezing system for the ball in a box problem.

we can find the auxiliary dynamics via Proposition 8.4 and obtain:

$$f_{\text{bottom}}(y) = (0, v_2, 0, -k(q_2 + 1.05R), 0),$$

$$f_{\text{top}}(y) = (0, v_2, 0, -k(q_2 - 1.15R), 0),$$

$$f_{\text{left}}(y) = (v_1, 0, -k(q_1 + 1.15R), 0, 1),$$

$$f_{\text{right}}(y) = (v_1, 0, -k(q_1 - 1.1R), 0, 0).$$

We set $k = 100$, and for fully elastic impacts we have $c = 0$. Moreover, we need the auxiliary dynamics for the impacts at the four corners. Since the constraint defining the corners are orthogonal, the corresponding auxiliary dynamics are simply the sum of the two neighboring ones. In total, we have nine regions: one for the unconstrained dynamics and eight for the auxiliary dynamics. The regions can be encoded via the sign matrix $S \in \mathbb{R}^{9\times 4}$ (cf. Section 6.2):

$$S = \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & 1 \\ -1 & -1 & 1 & 1 \\ 1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & 1 & 1 & -1 \\ -1 & 1 & 1 & -1 \end{bmatrix},$$

The dynamics are summarized in Table 8.2. We collect all modes in the matrix $F(y, u) \in \mathbb{R}^{5\times 9}$ and use Stewart's reformulation (cf. Section 6.2.1) to obtain a DCS. The initial value is:

$$y(0) = (R\sin(\alpha_0), R\cos(\alpha_0), R\omega\cos(\alpha_0), -R\omega\sin(\alpha_0), 0).$$

We introduce a least square objective with the running cost:

$$L(y(\tau), u(\tau)) = (q_1 - R\sin(\omega t(\tau) + \alpha_0))^2 + (q_2 - R\cos(\omega t(\tau) + \alpha_0))^2.$$

The norm of the thrust force is bounded via the inequality constraint

$$u^\top u \leq R_u^2,$$

with $R_u = 50$. The control horizon is taken to be two periods, i.e., $T = 2\frac{2\pi}{|\omega|}$. We use $N = 40$ equidistant control intervals and use a FESD Radau IIA method of order 3 with $N_{\text{FE}} = 4$ on every control interval. Therefore, we have all ingredients formulate an optimal control problem of the form (8.60).

The optimal control problem is solved for $\omega = -2\pi$ and $\omega = -3\pi$. The final results are depicted and compared in Figure 8.13. In the first case, the reference is slow enough, and the thrust force is sufficient for perfect tracking. In the second case, the reference is too fast, and the optimizer discovers that making impacts with the walls can help to reduce the tracking error.

Next, we compare the clock states and speed of time variables $s(\tau)$ for the two cases. This is depicted in Figure 8.14. The vertical dashed lines in the left plots show the boundaries of the control intervals. We can see that an equidistant control grid is obtained, as discussed in Section 8.5. In the slower reference case (top plot), the auxiliary dynamics do not become active, and the physical time is equal to the numerical time. The speed of time variable $s(\tau)$ is constant and equal to one, as no frozen time needs to be compensated. In the faster reference case, we have in total eight impacts. The speed of time takes a larger value during all intervals with impacts to catch up for the frozen time intervals.

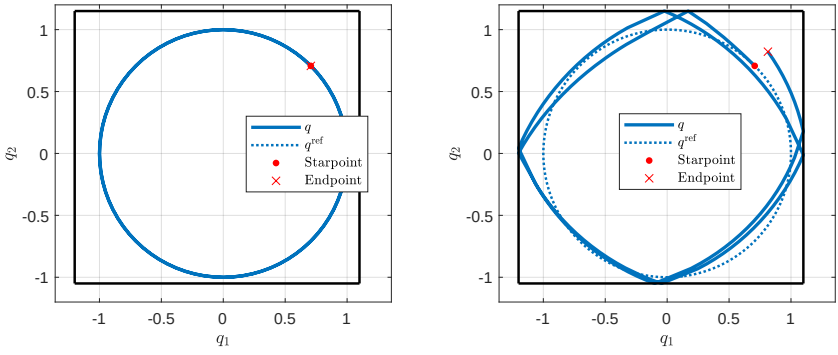


Figure 8.13: Illustration of the solution trajectories for the ball in box optimal control problem. The left plot is for $\omega = -2\pi$, and the right is for $\omega = -3\pi$.

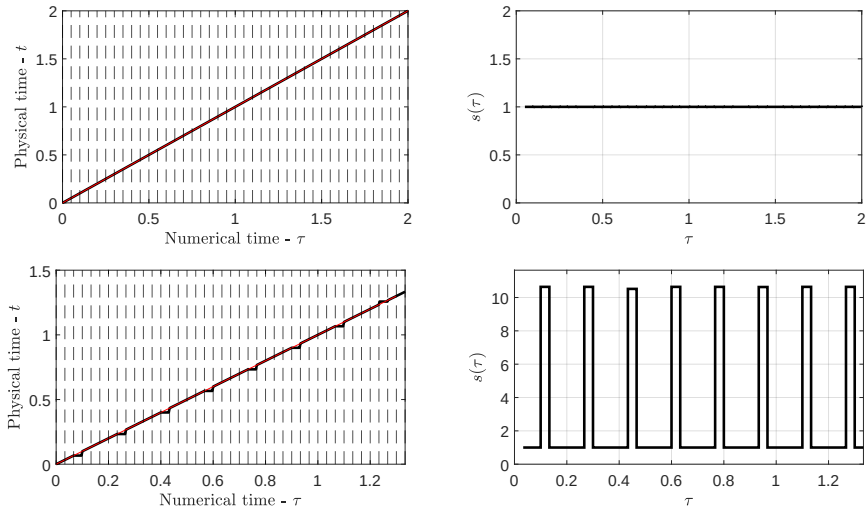


Figure 8.14: The left plots show the clock state as a function of the numerical time $t(\tau)$. The right plots the speed of time variables $s(\tau)$. The top plots are for $\omega = -2\pi$ and the bottom for $\omega = -3\pi$.

Figure 8.15 shows the state and optimal controls for this case. We can see that the velocity state is continuous in numerical time and discontinuous in physical time. Note that the reference stops during time-frozen periods since it is a function of $t(\tau)$.

8.6.2 A hopping robot - inelastic impact with friction

We consider a hopping robot that must jump over three holes to reach a desired target. We derive an OCP formulation for synthesizing dynamic motions of the single-legged 2D robot *Capler* [63]. The robot is described by four degrees of freedom $q = (q_x, q_z, \phi_{\text{knee}}, \phi_{\text{hip}})$. Here, (q_x, q_z) are the coordinates of the robot's base at the hip, and ϕ_{knee}, ϕ_{hip} are the angles of the hip and knee, respectively, cf. left plot in Fig. 8.16. It is actuated by two direct-drive motors at the hip and knee joints. The robot's dynamics are compactly described by the CLS in the form of (8.1). The torques of the two motors $u(t) = (u_{\text{knee}}(t), u_{\text{hip}}(t))$ are the control variables. A detailed derivation of the model equations and all parameters for the robot can be found in [109, Appendix A].

Denote by $p_{\text{foot}}(q) = (p_{\text{foot},x}(q), p_{\text{foot},z}(q))$ and $p_{\text{knee}}(q) = (p_{\text{knee},x}(q), p_{\text{knee},z}(q))$ the kinematic position of the robot's foot and knee, respectively. For the

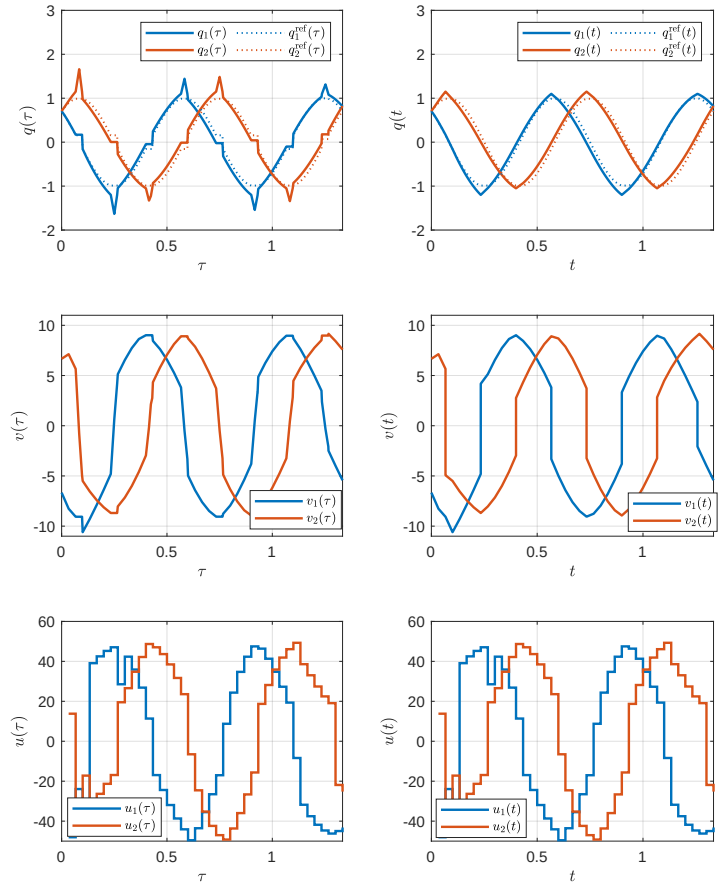


Figure 8.15: A solution to the ball in a box optimal control problem for $\omega = 2\pi$. The left plots are in numerical time τ right in physical time t. The first row shows the positions q, the second the velocities v, and the third the optimal controls u.

unilateral constraint function we take $f_c(q) = p_{\mathrm{foot},z}(q)$. For a planar robot, we need just one tangent, i.e., $b(q) = \nabla_q p_{\mathrm{foot},x}(q)$ and the friction model is exact. The coefficient of friction is $\mu = 0.8$ and the auxiliary ODE constant is $a_{\mathrm{n}} = 200$.

The objective of the OCP is to minimize the integral of the squared control torques, i.e., we have the running cost $L(x, u) = u(\tau)^\top u(\tau)$. The robots should

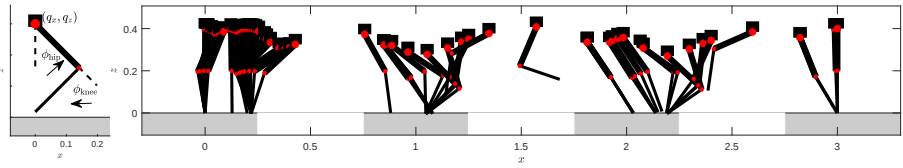


Figure 8.16: Illustration of the robot kinematics (left), several frames of the solution of the discretized OCP (right).

reach a given target position $q_{\text{target}} = (3, 0.4, 0, 0)$ starting from the initial position $q_0 = (0, 0.4, 0, 0)$ with zero velocity $v_0 = \mathbf{0}_{4,1}$. The initial value is $y_0 = (q_0, v_0, 0)$ and the prediction horizon is $T_{\text{ctrl}} = 2.5$ s. We add the following constraints on the states and kinematic positions:

$$
\begin{aligned}
-0.05e \leq (q_x(t), p_{\text{foot},x}(q(t)), p_{\text{knee},x}(q(t))),\\
0.2 \leq q_z(t) \leq 0.55,\\
-\frac{3\pi}{8} \leq \phi_{\text{hip}}(t) \leq \frac{3\pi}{8},\\
-\frac{\pi}{2} \leq \phi_{\text{knee}}(t) \leq \frac{\pi}{2},\\
0.05 \leq p_{\text{knee},z}(q(t)),\\
-0.005 \leq p_{\text{foot},z}(q(t)) \leq 0.2,\ t \in [0, T].
\end{aligned}
$$

Their goal is twofold. On the one hand, they should avoid unnatural and too extensive bending of the joints. On the other hand, they serve as *guiding constraints* during the early phases of the homotopy procedure. In the early iterations, the physics are relaxed, and we want to prohibit the optimizer to go to undesired regions. The control bounds read as

$$
-60e \leq u(t) \leq 60e,\ t \in [0, T].
$$

On the way to the target, the robot must overcome three holes in the ground. Instead of using very complicated expressions for $f_c(q)$, we model the holes as regions that the robot should not enter. This is achieved by constraints inside the OCP requiring that p_{foot} is outside $n_e = 3$ ellipsoids:

$$
\left(\frac{p_{\text{foot},x} - x_{\text{c},k}}{a_k}\right)^2 + \left(\frac{p_{\text{foot},z} - z_{\text{c},k}}{b_k}\right)^2 \geq 1, k = 1, \ldots, n_e.
$$

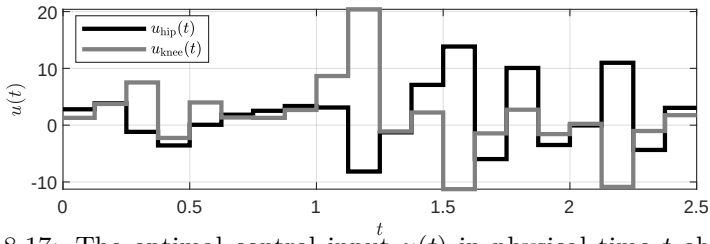


Figure 8.17: The optimal control input $u(t)$ in physical time t obtained by solving the discretization of the optimal control problem.

By appropriately picking $a_k, b_k, x_{\text{c},k}$ and $z_{\text{c},k}$ the desired shapes are trivially selected. In this example, we pick $z_{\text{c},k} = 0$, $a_k = 0.5$ (width of the hole), $b_k = 0.1$ (kept low, should not enforce unnecessarily high jumps). For the centers of the holes, we pick $x_{\text{c},1} = 0.5$, $x_{\text{c},2} = 1.5$, $x_{\text{c},3} = 2.5$. We collect all path constraints (for the holes, on the kinematics, control, and state bounds) into the function $g(x, u) \geq 0$.

Remark 8.20. *Note that the constraints $g(x, u) \geq 0$ cannot become active if the corresponding normal velocity is nonzero, as opposed to activating a constraint $f_c(q) \geq 0$, since no state jump law is associated with path constraints in an OCP. This is one of the main differences between constraints that are part of the dynamics (equipped with a state jump law) and path constraints in the OCP.*

We have now all ingredients to formulate an OCP in the form of Eq. (8.53). `nosnoc` automatically reformulates the CLS into a time-freezing system and discretizes the OCP, such that we obtain a discrete-time problem in the form of (8.58). The resulting mathematical program with complementarity constraints is solved in a homotopy procedure with `IPOPT` [281] equipped with the `MA57` linear solver [143]. The source code for this example is available in `nosnoc`'s repository [2]. The OCP is discretized with a FESD Radau IIA scheme of order 5 [213]. We consider $N = 20$ control intervals with $N_{\text{FE}} = 3$ intermediate integration steps on every interval.

For the initialization of the differential states, we take y_0 at every discretization node. All discrete-time control variables are initialized with zero. Hence, no information about the order, number, or timing of the nonsmooth transitions and jumps is provided. The results of the optimization are shown in the right plot of Fig. 8.16. The approach finds an intuitive dynamic movement by solving only smooth NLP, without providing any hints about the order and number of nonsmooth transitions. The optimal torques are depicted in Fig. 8.17.

8.6.3 Manipulation task - inelastic impacts

In the last example of this section, we regard a manipulation task. We regard two balls that lie in a two-dimensional plane. Only one ball can be controlled by a thrust force, whereas the second can only be moved by making inelastic contact with the first one. The goal is that the balls swap their position with minimal control effort. We model this with the following optimal control problem:

$$
\begin{aligned}
\min_{x(\cdot),\lambda_{\mathrm{n}}(\cdot),u(\cdot),} \quad & \int_0^T \|x(t)-x_{\mathrm{r}}\|_Q^2 + \|u(t)\|_R^2 \,\mathrm{d}t + \|x(T)-x_{\mathrm{r}}\|_{Q_T}^2 \\
\text{s.t.} \quad & x(0) = \bar{x}_0, \\
& \dot{q}(t) = v(t),\ t \in [0,T], \\
& M\dot{v}(t) = \begin{bmatrix} u(t) - c_{\mathrm{f}} \frac{v_1}{\|v_1+\epsilon_{\mathrm{f}}\|} \\ \mathbf{0}_{2,1} - c_{\mathrm{f}} \frac{v_2}{\|v2+\epsilon\mathrm{f}\|} \end{bmatrix} + n(q(t))\lambda_{\mathrm{n}}(t),\ t \in [0,T], \\
& 0 \le \lambda_{\mathrm{n}}(t) \perp f_c(q) \ge 0,\ t \in [0,T], \\
& t0 = n(q(t))^\top v(t_{\mathrm{s}}^+)\ f_c(q(t_{\mathrm{s}}) = 0 \text{ and } n(q(t))^\top v(t_{\mathrm{s}}^-) < 0, \\
& -10e \le q(t) \ge 10e,\ t \in [0,T], \\
& -5e \le v(t) \ge 5e,\ t \in [0,T], \\
& -30e \le u(t) \ge 30e,\ t \in [0,T], \\
& 0 \le \|q_1(t)\|^2 - (r_{\mathrm{ob}} + r_1)^2,\ t \in [0,T], \\
& 0 \le \|q_2(t)\|^2 - (r_{\mathrm{ob}} + r_2)^2,\ t \in [0,T],
\end{aligned}
$$

The states are the positions $q_1 = (q_{1,1}, q_{1,2})$, $q_2 = (q_{2,1}, q_{2,2})$ and velocities $v_1 = (v_{1,1}, v_{1,2})$, $v_2 = (v_{2,1}, v_{2,2})$, respectively. The initial positions are $q_1(0) = (-2, 0)$ and $q_2(0) = (2, 0)$, and the initial velocities are zeros $v_1(0) = v_2(0) = \mathbf{0}_{2,1}$. The reference is $x_{\mathrm{r}} = (q_2(0), q_1(0), v_1(0), v_2(0)$, i.e., the balls should swap their positions and be at rest, and the control effort should be minimal. This is modeled with least squares running and terminal objective terms with the weighting matrices: $Q = \mathrm{diag}(10, 10, 10, 10, 0.01, 0.01, 0.01, 0.01))$, $R = \mathrm{diag}(0.1, 0.1)$ and $Q_T = 100Q$. The control forces $u = (u_{1,2}, u_{2,1})$ acts only on the first ball. Moreover, we model constant friction force in the plane with $c_{\mathrm{f}} = 2$ and $\epsilon_{\mathrm{f}} = 0.1$ to regularize the norm at zero and the contact between the balls is frictionless. The constant inertia matrix is $M = \mathrm{diag}(m_1, m_1, m_2, m_2)$ with $m_1 = 2$ and m_1. The gap function reads as $f_c(q) = \|q_1(t) - q_2(t)\|^2 - (r_1 + r_2)^2$,

where the radii of the balls are $r_1 = 0.3$ and $r_2 = 0.2$. We introduce *guiding* box constraints on the position and velocity and bound the control in every direction. The last two constraints model the obstacle which is a ball with a radius of $r_{\text{ob}} = 1$ and located at the origin.

We transform this OCP into a discrete-time time-freezing OCP as described in Section 8.5. The control horizon is $T = 5$ we take $N = 25$ control intervals and use a FESD implicit Euler method with $N_{\text{FE}} = 3$. The problem is reformulated, discretized, and solved with `nosnoc`. Figure 8.18 shows ten frames of the optimal solution. The optimizer finds a creative solution without any hints or sophisticated initial guesses. The first ball goes to the second ball, makes contact with it, pushes it around the obstacle, and brings it to its final position. It breaks the contact and returns to the final position. Figure 8.19 shows the states and optimal controls as a function of physical time. One can see that the second ball is at rest until the first ball touches it and creates a velocity jump. At $t = 4$, the contact breaks, and the first ball slides slowly toward the goal position.

8.7 Conclusions and outlook

This chapter introduced the time-freezing reformulation for transforming complementarity Lagrangian systems (CLS) with state jumps into piecewise smooth systems. The main idea is to define an auxiliary dynamical system in the region that is infeasible for the original system and a clock state. The

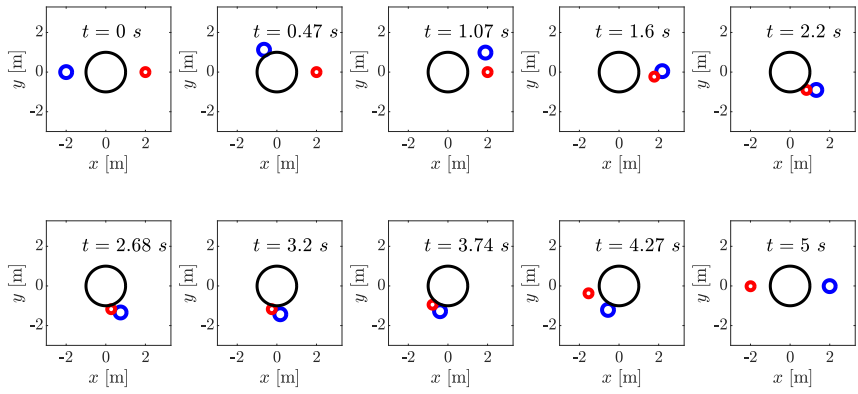


Figure 8.18: Several frames of the optimal solution. The first ball is marked with blue, the second with red, and the obstacle with black.

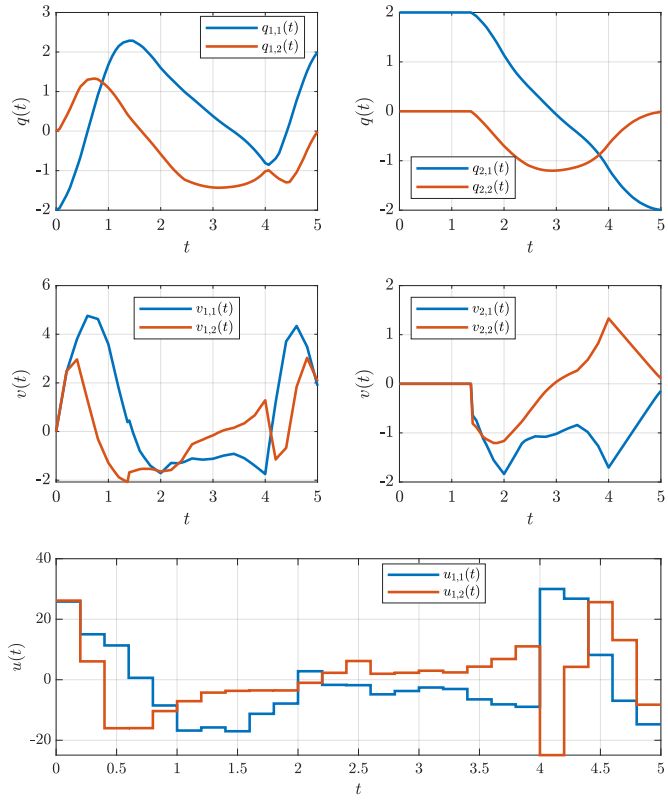


Figure 8.19: The trajectories and optimal controls of the manipulation problem.

endpoints of the trajectory of the auxiliary ODE satisfy the state jump laws, while the evolution of the clock state is frozen during its runtime. By taking only the pieces of the trajectory when time is flowing, we can recover the solution of the original system. To the best of the author's knowledge, this is the first reformulation technique that establishes a strong link between a wide range of practical nonsmooth mechanics problems and Filippov systems, enabling the use of the rich theoretical and computational tools developed for Filippov systems. For example, this enables the seamless application of the FESD method from Chapter 7.

We provide constructive ways for selecting auxiliary ODEs and prove the equivalence of solutions between the original and time-freezing systems. Furthermore, we demonstrate that solving the OCP with time-freezing systems can provide a solution to an OCP with a CLS. We illustrate the practicality

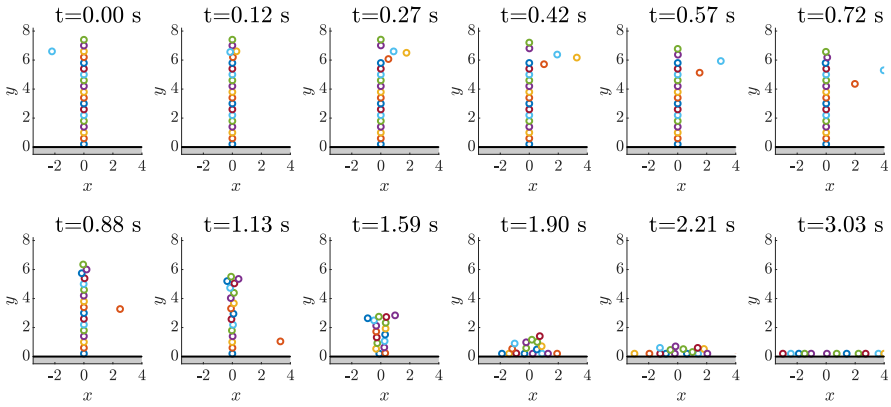


Figure 8.20: Several frames of time-freezing system simulation where a pile of 19 stacked balls is hit by another ball. The tower slowly collapses, and many intermediate contacts take place.

of the discussed methods with several OCP examples. All of the methods described here are implemented in the open-source package `nosnoc` [206].

Several questions need to be addressed in future work. First, a natural and necessary extension is to consider cases with multiple and simultaneous impacts, i.e., models with vector-valued gap functions. In such cases, one also needs to decide which impact model to use. Furthermore, we want to avoid defining separate auxiliary dynamics for every combination of simultaneous or single impacts, as this would result in exponential complexity. A possible solution is to exploit the additive nature of the CLS, where the total contact force is the sum of all contact forces at every contact. This can also be applied to the time-freezing system, for example, by using the sum of Filippov systems concept discussed in Section 6.2.5. The first implementation of this approach is already available in `nosnoc`. However, deriving all the details and developing the equivalence theory is beyond the scope of this thesis. Nevertheless, the first numerical experiments appear to be very promising. Figure 8.20 shows several frames where a pile of 19 balls is hit by another ball. This example has in total 210 gap functions, and the simulation result appears to be physically reasonable. The second question to be addressed is whether there exists a unified time-freezing reformulation for both elastic and inelastic impacts. Currently the two models are somewhat different as they do not use the same number of switching functions. It would be desirable to have a single formulation that depends only on the coefficient of restitution ϵ_{r}. On the one hand, if one tries to create an inelastic impact model from the elastic time-freezing system, the auxiliary dynamical system must become infinitely stiff. Additionally, incorporating

friction in the current elastic case seems to be more difficult. It is not obvious how to define tangential auxiliary dynamics that are compliant with the normal auxiliary dynamics and that enable both stick and slip modes. On the other hand, the inelastic approach was easier to extend to frictional and multiple impacts. However, it is not clear how to treat elastic impacts with the ideas developed for the inelastic case.

Furthermore, to make this approach even more practical, good initialization heuristics and more sophisticated homotopy procedures for the MPCCs would be useful. Exploiting some knowledge and structure in the system might be beneficial. For example, when the system enters R_{aux}, by construction it will leave it or end on its boundary after τ_{jump}. Hence, the second switch may be encoded explicitly and not discovered implicitly via the complementarity conditions.

Chapter 9

The Time-Freezing Reformulation for Nonsmooth Systems with Hysteresis

In this chapter, we extend the time-freezing reformulation to a class of hybrid systems with a hysteresis. Hysteresis is a rate-independent memory effect that results in severe nonsmoothness in the dynamics. These systems are not simply piecewise smooth systems but a more complicated form of hybrid systems. This class is not closely related to complementarity Lagrangian systems. However, we build upon the same core ideas: introducing auxiliary dynamics in infeasible regions to mimic state jumps and freezing the time during their evolution. After the time-freezing reformulation, we end up again with a Filiipov system. From a theoretical standpoint, this reformulation paves the way for studying systems with hysteresis using the sophisticated tools developed for Filippov systems. From a practical perspective, it allows for the use of FESD, which facilitates highly accurate numerical optimal control of hybrid systems with he a hysteresis. To illustrate this, we provide an example of a time-optimal control problem and compare our approach to mixed-integer formulations found in the existing literature. Here a single hysteresis characteristic is considered. An extension to a cascade of such systems was developed in [274].

Outline. Section 9.1 gives some basic definitions of the hybrid systems with a hysteresis. In Section 9.2, we develop the time-freezing reformulation for a class of hybrid systems with hysteresis and provide a simple tutorial example.

Section 9.3 formalizes the relation between time-freezing PSS and hysteresis systems. Finally, Section 9.4 contains a numerical example, and Section 9.5 concludes the chapter. This chapter is mainly based on the article [205].

9.1 Hybrid systems with hysteresis

9.1.1 Introduction

Hysteresis occurs in many physical systems, e.g., ferromagnetism, plasticity, superconductivity, and phase transitions, but also in feedback control, e.g., thermostats [186]. Hysteresis effects in dynamic systems are modeled with nonsmooth differential equations. Here we focus on transforming some classes of systems with hysteresis into the piecewise smooth system (PSS) and numerically solving OCPs subject to such systems. A hybrid system with hysteresis can be represented as a *finite automaton* [186], which has two modes of operation described by $f_{\mathrm{A}}(x)$ and $f_{\mathrm{B}}(x)$, cf. Figure 9.1 (left). If the system operates in mode A with $\dot{x} = f_{\mathrm{A}}(x)$ and if $\psi(x) \geq 1$, it switches to mode B with $\dot{x} = f_{\mathrm{B}}(x)$. On the other hand, if it operates in mode B and if $\psi(x) \leq 0$, it switches to mode A. This is a typical hysteresis behavior given by the characteristic in the right plot of Figure 9.1, which is often called the *delayed relay operator* [51]. The dynamics of the system depend on the value of $w(t)$ and the scalar *switching function* $\psi(x)$. Notably, for $\psi(x) \in [0, 1]$ the function $w(t)$ can be 0 or 1.

There are several other related characteristics, e.g., the dashed lines in Figure 9.1 could be solid, or the resulting polygon in the middle of the plot might be tilted. In all these cases, the characteristic can be readily represented via a linear complementarity problem [275] and the nonsmooth dynamic system recast into a Dynamic Complementarity System (DCS). However, it is an open question if such DCS is a PSS. Another widely used model is the Duhem hysteresis [280]. It can be straightforwardly transformed into a PSS [51, Section 2.6.3] and numerically treated with FESD. In control theory, systems with hysteresis are often studied via the hybrid systems framework, which uses integer state and control variables [186, 31, 20]. Hence, in an optimal control context, this requires solving Mixed-Integer Optimization Problems (MIOP). They can be solved efficiently in the case of discrete-time linear hybrid systems [31]. However, as soon as the junction times need to be determined precisely or nonlinearity is present, e.g., in time optimal control problems, solving MIOP can become arbitrarily difficult. On the other hand, the nonsmoothness can be modeled with complementarity constraints [203], and one must solve only nonsmooth NLPs.

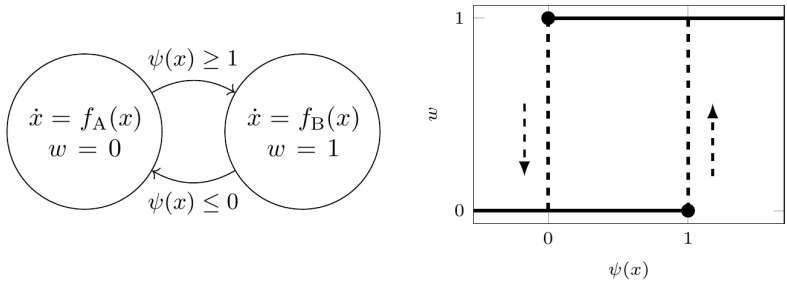


Figure 9.1: Hybrid system with hysteresis.

The *time-freezing* reformulation transforms systems with state jumps into PSS and was first introduced in [212]. This chapter introduces a time-freezing reformulation to transform systems represented with the finite automaton in Figure 9.1 (left) into PSS. Here, the main idea is to regard $w(t)$ as a continuous differential state. However, $w(t)$ exhibits jump discontinuities in time at $(0, 1)$ and $(1, 0)$, which can be interpreted as a *state jump law*. Similar to the approach of the previous chapter, we introduce *auxiliary dynamical systems* and a *clock state*. The auxiliary ODE evolves in regions, which are prohibited for the initial system, and their trajectory endpoints satisfy the state jump law. Additionally, the evolution of the clock state is frozen during the time evolution of the auxiliary systems. By regarding only the parts of $w(\cdot)$ when the clock state was evolving, we recover the original discontinuous solution. Note that the resulting time-freezing system is now a PSS, since the only remaining jump discontinuities are in the system's dynamics but not in the state anymore. For high-accuracy numerical optimal control of PSS, we use the FESD method from Chapter 7.

The contributions of this chapter are as follows. We present a time-freezing reformulation for a class of hybrid systems with hysteresis, which transforms them into PSS. Constructive ways for finding the auxiliary dynamics needed in time-freezing are provided. Solution equivalence between the initial hybrid and time-freezing PSS is proven. From the theoretical side, this contribution enables one to treat hybrid systems with hysteresis with the tools for PSS and Filippov systems [94]. From the practical side, the highlight of this chapter is that we can solve OCP with systems with hysteresis with high accuracy and without the use of any integer variables. A time optimal control problem of a hybrid system with hysteresis and illustrates theoretical and algorithmic developments. We compare the continuous optimization-based FESD method to mixed-integer solution strategies.

9.1.2 Model equations

We consider dynamic systems represented with the finite automaton the left plot of Figure 9.1:

$$\dot{x} = f(x, w) = (1 - w) f_{\mathrm{A}}(x) + w f_{\mathrm{B}}(x), \tag{9.1}$$

where the $(w, \psi(x))$ characteristic is illustrated in the right plot in Figure 9.1. For a uniformly continuous function $x(t)$ on $t \in [0, T]$ and a smooth $\psi(\cdot)$, there can be only finitely many oscillations between 0 and 1. Consequently, the function $w(t)$ is piecewise constant and has only finitely many jumps between 0 and 1 [280].

The system in (9.1) has two modes of operation denoted by A and B. In order to be able to simulate (9.1) for $t \in [0, T]$ with a given $x(0) = x_0$, we must know $w(0)$ as well. This property is typical for systems with a hysteresis. Furthermore, $w(\cdot)$ jumps between 0 and 1, hence we can describe it by an ODE with the state vector $z \coloneqq (x, w) \in \mathbb{R}^{n_x+1}$, which is associated with a state jump law.

$$\dot{z} = (f(x, w), 0), \tag{9.2}$$

accompanied by a state-jump law for $w(\cdot)$ at time-point t_{s}, which covers two scenarios:

1. if $w(t_{\mathrm{s}}^-) = 0$ and $\psi(x(t_{\mathrm{s}}^-)) = 1$, then $x(t_{\mathrm{s}}^+) = x(t_{\mathrm{s}}^-)$ and $w(t_{\mathrm{s}}^+) = 1$,
2. if $w(t_{\mathrm{s}}^-) = 1$ and $\psi(x(t_{\mathrm{s}}^-)) = 0$, then $x(t_{\mathrm{s}}^+) = x(t_{\mathrm{s}}^-)$ and $w(t_{\mathrm{s}}^+) = 0$.

Clearly, due to the state jump law, the ODE (9.2) is not simply a PSS. Throughout this chapter we assume, given $x(0)$ and $w(0)$ that there exists a solution to the Initial Value Problem (IVP) associated with (9.2). A way to define a meaningful notion of solution for a hybrid system as (9.2) is given in e.g., [186, Section 5.4] and sufficient conditions for well-posedness are provided [186, Theorem 5.4].

9.2 The time-freezing reformulation for hybrid systems with hysteresis

This section introduces the time-freezing reformulation for the system (9.2). We define step-by-step the corresponding regions R_i of the time-freezing PSS and give constructive ways to find vector fields associated with them. The section finishes with a tutorial example.

9.2.1 The time-freezing system

The main idea is to transform the state $w(t)$ which is a piecewise constant function of time into a continuous differential state on a different time domain. We call this new time domain the *numerical time* and denote it by τ. Instead of t, τ will now be the time of the time-freezing PSS. Moreover, we introduce a clock state $t(\tau)$ in the time-freezing PSS which we call *physical time*. It grows whenever the systems evolves according to $f_{\mathrm{A}}(x)$ or $f_{\mathrm{B}}(x)$, i.e., we have in these cases $\frac{\mathrm{d}t}{\mathrm{d}\tau}(\tau) = 1$. Otherwise, the physical time is frozen, i.e., we have that $\frac{\mathrm{d}t}{\mathrm{d}\tau}(\tau) = 0$. In other words, the time is frozen whenever $w \notin \{0, 1\}$. Consequently, the $w(\cdot)$ takes only discrete values in physical time, i.e., when $t(\tau)$ is evolving.

The time-freezing PSS has the following state vector $y \coloneqq (x, w, t) \in \mathbb{R}^{n_y}, n_y = n_x + 2$. In the sequel, we define its regions $R_i \subset \mathbb{R}^{n_y}$ and the associated vector fields $f_i(y)$. Some key observations can be made from Figure 9.1. First, everything except the solid curve is prohibited for the system (9.2) in the $(\psi, w)-$ plane. We use this prohibited part of the state space to define auxiliary dynamics. Second, the evolution happens in a lower-dimensional subspace since $\dot{w} = 0$. This corresponds in Filippov's setting to sliding modes, cf. Chapter 6. Hence, we define the regions such that the evolution of the initial system (9.2) corresponds to sliding modes of the time-freezing PSS, i.e., it happens on region boundaries ∂R_i.

A suitable partition of the $(\psi, w)-$ plane can be achieved with *Voronoi regions*. The regions are defined as

$$R_i = \{z \mid \|z - z_i\|^2 < \|z - z_j\|^2,\ j = 1, \ldots, 4, j \neq i\},$$

where $z = (\psi(x), w)$. We select the points: $z_1 = (\frac{1}{4}, -\frac{1}{4}), z_2 = (\frac{1}{4}, \frac{1}{4}), z_3 = (\frac{3}{4}, \frac{3}{4})$ and $z_4 = (\frac{3}{4}, \frac{5}{4})$. An illustration of the regions is given in Figure 9.2, where the black solid lines denote the region boundaries. This choice of z_i defines regions such that their boundaries correspond to the feasible set of the original system (9.2). Moreover, the space is split by the diagonal line between R_2 and R_3 such that we can define different auxiliary dynamics for the state jumps in both directions. One can make other choices for the points z_i with the same properties. The proposed choice partitions the space symmetrically, cf. Figure 9.2. The figure illustrates also the vector fields in the regions R_i whose meaning is detailed below. It is important to note that the original system can only evolve at region boundaries:

$$R_{\mathrm{A}} \coloneqq \{y \in \mathbb{R}^{n_y} \mid w = 0,\ \psi(x) \leq 1\} = \partial R_1 \cap \partial R_2,$$

$$R_{\mathrm{B}} \coloneqq \{y \in \mathbb{R}^{n_y} \mid w = 1,\ \psi(x) \geq 0\} = \partial R_3 \cap \partial R_4.$$

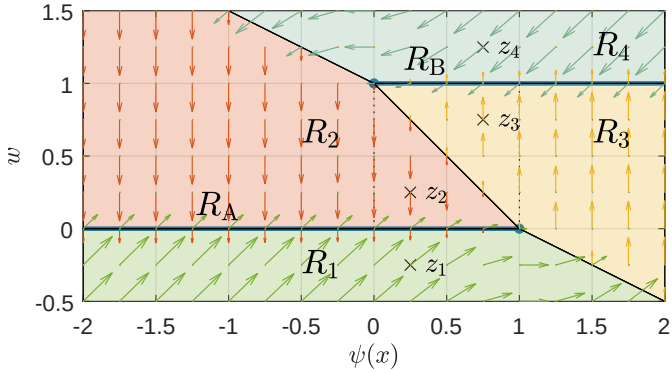


Figure 9.2: Illustration of the partitioning of the state space in $(\psi(x), w)$-plane for the time-freezing PSS via Voronoi regions with the corresponding auxiliary and DAE-forming dynamic's vector fields. The Voronoi points $z_i, i = 1, \ldots, 4$, are marked by the crosses.

We exploit the interior of the regions R_i, $i = 1, \ldots, 4$ to define the needed auxiliary ODEs. In what follows, in the regions R_2 and R_3 we define auxiliary dynamic systems whose trajectory endpoints satisfy the state jump law of (9.2). In the regions R_1 and R_4 we will define so-called DAE-forming ODE, which makes sure that we obtain appropriate sliding modes on R_A and R_B, which are described by index 2 DAE [94] and which match the dynamics of the original system. The next definition formalizes the desired proprieties of an auxiliary ODE.

Definition 9.1 (Auxiliary ODE for hysteresis systems)**.** *The auxiliary ODE in regions R_2 and R_3 are denoted by $y' = f_{\mathrm{aux,A}}(y)$ and $y' = f_{\mathrm{aux,B}}(y)$, respectively. For every initial value $y(\tau_\mathrm{s}) = y_\mathrm{s}$ such that $(w(\tau_\mathrm{s}), \psi(x(\tau_\mathrm{s})) = (1, 0)$, for $y_\mathrm{s} \in R_\mathrm{B}$, (and $(w(\tau_\mathrm{s}), \psi(x(\tau_\mathrm{s})) = (0, 1)$ for $y_\mathrm{s} \in R_\mathrm{A}$, respectively) and for a well-defined and finite time interval $\mathcal{T}_\mathrm{jump} := (\tau_\mathrm{s}, \tau_\mathrm{r})$ with the length $\tau_\mathrm{jump} := \tau_\mathrm{r} - \tau_\mathrm{s}$, the auxiliary ODE satisfy the following properties:*

(i) $w(\tau) \in (0, 1)$, *for all* $\tau \in \mathcal{T}_\mathrm{jump}$,

(ii) $x(\tau_\mathrm{s}) = x(\tau_\mathrm{r})$ *and*

(iii) $w(\tau_\mathrm{r}) = 0$ *(or* $w(\tau_\mathrm{r}) = 1$*).*

In other words, we define an ODE whose trajectory endpoints on $\overline{\mathcal{T}}_\mathrm{jump}$ satisfy the state jump law associated with Eq. (9.2), cf. Figure 9.2. The next proposition provides a constructive way to find an ODE with the above-described properties.

Proposition 9.2 (Auxiliary ODE for hysteresis system)**.** *Given an initial value* $y(\tau_\mathrm{s}) = y_\mathrm{s}$ *such that* $w(\tau_\mathrm{s}) = 1$ *and* $\psi(x(\tau_\mathrm{s})) = 0$*, the ODE given by*

$$y'(\tau) = f_{\mathrm{aux,A}}(y) \coloneqq (\boldsymbol{0}_{n_x,1}, -\gamma(\psi(x) - 1), 0), \tag{9.3}$$

where $\gamma : \mathbb{R} \to \mathbb{R}$ *and* $\gamma(x) = \frac{ax^2}{1+x^2}$ *with* $a > 0$*, is an auxiliary ODE defined in* R_2*. Similarly, for* $y(\tau_\mathrm{s}) = y_\mathrm{s}$ *with* $w(\tau_\mathrm{s}) = 0$ *and* $\psi(x(\tau_\mathrm{s})) = 1$*, the ODE*

$$y'(\tau) = f_{\mathrm{aux,B}}(y) \coloneqq (\boldsymbol{0}_{n_x,1}, \gamma(\psi(x)), 0). \tag{9.4}$$

is an auxiliary ODE in R_3*. In both cases* $\tau_\mathrm{jump} = \frac{1}{\gamma(-1)}$*.*

Proof. We prove the assertion for (9.3), since the second part follows similar lines. Since $x'(\tau) = \mathbf{0}_{n_x,1}$ and $t'(\tau) = 0$ these two variables do not change their value, thus $\psi(x(\tau)) = \psi(x(\tau_\mathrm{s})) = 0$ and $t(\tau) = t(\tau_\mathrm{s})$ for $\tau \geq \tau_\mathrm{s}$. Hence, we have $w'(\tau) = -\gamma(-1) < 0$. By explicitly solving the ODE we obtain $w(\tau_\mathrm{r}) = 0$ for $\tau_\mathrm{r} = \tau_\mathrm{s} + \tau_\mathrm{jump}$, where $\tau_\mathrm{jump} = \frac{1}{\gamma(-1)}$. All conditions of Definition 9.1 are satisfied thus the proof is complete. □

We briefly discuss some of the proprieties of such an auxiliary ODE, since there are several ways to construct a similar ODE. Loosely speaking, in Figure 9.2 in R_2 the vector field should point in the negative w-detection and in R_3 in the positive w-direction, and be zero in all other directions. Note that for $\psi(x) \in (0, 1)$ the vector fields of the auxiliary ODE in both cases point away from the manifold defined $\mathcal{M} = \{y \in \mathbb{R}^{n_y} \mid w + \psi(x) - 1 = 0\}$. In such scenarios, there is usually locally no unique solution to the associated Filippov DI, as the trajectory can leave $\mathcal{M}$ at any point in time [94]. However, the system should never be initialized in this region, since this state is infeasible for the original system. We show later that it can never reach this undesired state if initialized appropriately. Furthermore, the auxiliary ODE from Proposition 9.2 have by construction the favorable property that they do not point away in both directions from $\mathcal{M}$ at the junction points $(0, 1)$ and $(1, 0)$. This is why the function $\gamma(\cdot)$ was introduced in the auxiliary ODE. Another favorable property is if the system is initialized with the wrong value for $w(\cdot)$ for $\psi(x) \notin (0, 1)$ the auxiliary ODE will automatically reinitialize $w(\cdot)$ while the physical time is frozen, cf. Fig 9.2.

We still need to define DAE-forming vector fields for the regions R_1 and R_4. These vector fields should be such that, together with the auxiliary dynamics in their respective regions, they result in sliding modes on R_A and R_B which match the dynamics of the initial system (9.2). In a general PSS, the vector fields are not defined on the region boundaries, thus we use Filippov's convexification

[94] and denote the Filippov set associated to the time-freezing PSS by $F_{\mathrm{TF}}(\cdot)$. The next proposition gives a constructive way to find the desired vector fields.

Proposition 9.3 (DAE-forming ODE)**.** *Suppose the regions R_2 and R_3 are equipped with the vector fields $f_{\mathrm{aux,A}}(\cdot)$ and $f_{\mathrm{aux,B}}(\cdot)$ from Proposition 9.2, respectively. Let the region R_1 be equipped with the ODE*

$$y' = f_{\mathrm{DF,A}}(y) \coloneqq 2(f_{\mathrm{A}}(x), 0, 1) - f_{\mathrm{aux,A}}(y), \tag{9.5}$$

then for $y \in R_{\mathrm{A}}$ it holds that $(f_{\mathrm{A}}(x), 0, 1) \in F_{\mathrm{TF}}(y) = \overline{\mathrm{conv}}\{f_{\mathrm{aux,A}}(y), f_{\mathrm{DF,A}}(y)\}$. Similarly, let the region R_4 be equipped with the following ODE

$$y' = f_{\mathrm{DF,B}}(y) \coloneqq 2(f_{\mathrm{B}}(x), 0, 1) - f_{\mathrm{aux,B}}(y), \tag{9.6}$$

then for $y \in R_{\mathrm{B}}$ it holds that $(f_{\mathrm{B}}(x), 0, 1) \in F_{\mathrm{TF}}(y) = \overline{\mathrm{conv}}\{f_{\mathrm{aux,B}}(y), f_{\mathrm{DF,B}}(y)\}$.

Proof. We prove the assertion for Eq. (9.5) and the second part follows similar lines. Note that for $y \in R_{\mathrm{A}} = \{y \mid c(y) \coloneqq w = 0, \psi(x) < 1\}$ we have that $\nabla c(y)^\top f_{\mathrm{aux,A}}(y) < 0$ and $\nabla c(y)^\top f_{\mathrm{DF,A}}(y) > 0$. Hence, we have a sliding mode on $w = 0$ with $\frac{\mathrm{d}w}{\mathrm{d}\tau} = 0$ [94]. From the definition of a Filippov set, we have that $F_{\mathrm{TF}}(y) = \{\theta_1(2(f_{\mathrm{A}}(x), 0, 1) - f_{\mathrm{aux,A}}(y)) + \theta_2 f_{\mathrm{aux,A}}(y) \mid \theta_1 + \theta_2 = 1, \theta_1, \theta_2 \geq 0\}$. From this relation and $w' = 0$ we obtain that $\theta_1 - \theta_2 = 0$. Thus we can solve for θ_1 and θ_2, i.e., $\theta_1 = \theta_2 = \frac{1}{2}$, which yields $(f_{\mathrm{A}}(x), 0, 1) \in F_{\mathrm{TF}}(y)$. This completes the proof. □

Note that by construction the two sliding modes on R_{A} and R_{B} agree with the r.h.s. of Eq. (9.2), augmented by the dynamics of the clock state. Now we have defined vector fields in all regions of the time-freezing PSS which corresponds to the original system (9.2). Another favorable property of the chosen auxiliary and DAE forming ODE is: since $w'(\tau)$ is bounded by $a > 0$ it cannot make the sliding mode DAE arbitrarily stiff, especially if constraint drift happens.

9.2.2 A tutorial example

To illustrate the theoretical development we construct a time-freezing PSS for a thermostat system with hysteresis. The source code of the example is available in the repository of `nosnoc` [2]. The system has a single state $x(\cdot)$ which models the temperature of a room that should stay inside the interval $x \in [18, 20]$. As soon as the temperature drops below $x = 18$ the heater is switched on and when the temperature grows above $x = 20$ it is switched off. The two modes of operation are given by $\dot{x} = f_{\mathrm{A}}(x) = -0.2x + 5$, when the heater is on and $\dot{x} = f_{\mathrm{B}}(x) = -0.2x$, when the heater is off. One can see that for $\psi(x) = 0.5(x - 18)$, we have a hybrid system that matches the finite automaton

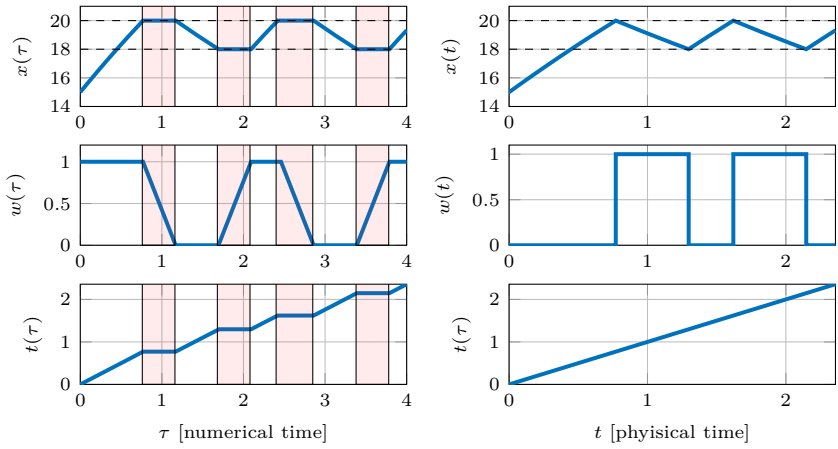


Figure 9.3: Trajectories of the time-freezing PSS for a thermostat example in numerical time τ (left plot) and physical time t (right plot).

in Figure 9.1. For a time-freezing PSS, we define the regions R_i via the Voronoi points as in the previous section. The auxiliary ODE's r.h.s. according to Proposition 9.2 read as $f_{\text{aux,A}}(y) = (0, -\gamma(0.5(x-18)-1), 0)$ and $f_{\text{aux,B}}(y) = (0, \gamma(0.5(x-18), 0)$ with $a = 1$. Similarly, the DAE-forming ODE r.h.s. according to Proposition 9.3 read as $f_{\text{DF,A}}(y) = (-0.4x + 10, \gamma(0.5(x-18)-1), 2)$ and $f_{\text{aux,B}}(y) = (-0.4x, -\gamma(0.5(x-18)), 2)$.

We simulate now the time-freezing PSS with a FESD Radau IIA integrator of order 3 with $x(0) = 15$ and $w(0) = 0$. The left plot in Figure 9.3 illustrates the evolution of the time-freezing PSS in numerical time. The red shaded areas indicate the phases when the auxiliary ODE is active with $w \notin \{0, 1\}$ while the time is frozen, cf. bottom left plot. In the middle left plot, we can see that $w(\tau)$ is now a continuous function in numerical time. The right plot in Figure 9.3 shows the differential state in physical time $t(\tau)$. We can see in the middle right plot that $w(t(\tau))$ is now a discontinuous function, hence the state jumps are successfully recovered in physical time.

9.3 Solution equivalence

From the developments in the last section, the solution equivalence is nearly apparent. We formalize it in the next theorem.

Theorem 9.4. *Regarding the IVP corresponding to:*

(i) the Filippov DI of the time-freezing PSS equipped with the vector fields from Proposition 9.2 and 9.3 with an initial value $y(0) = (z_0, 0)$ *with* $z_0 = (x_0, w_0)$ *and* $w_0 \in \{0, 1\}$*, on a time interval* $[0, \tau_\mathrm{f}]$*,*

(ii) the ODE with state jumps from Eq. (9.2) *with* $z(0) = z_0$ *on a time interval* $[0, t_\mathrm{f}] = [0, t(\tau_\mathrm{f})]$*. Suppose solutions exist to both IVP.*

Then the solutions of the two IVPs $z(t; z_0)$ *and* $y(\tau; y_0)$ *fulfill at any* $\dfrac{dt}{d\tau} = t'(\tau) \neq 0$*:*

$$z(t(\tau); z_0) = My(t(\tau); y_0), \text{ with } M = \begin{bmatrix} I_{n_x+1} & \boldsymbol{0}_{n_x+1,1} \end{bmatrix}. \tag{9.7}$$

Proof. Denote the solution of IVP (i) by $y_1(\tau; y_0)$ for $\tau \in (0, \hat{\tau})$ and for (ii) and $t(\tau) \in (0, t(\hat{\tau}))$ by $z_1(t(\tau); z_0)$. For a given $w(0) = 0$ (or 1) we have from Proposition 9.3 that $y' = (f_\mathrm{A}(x), 0, 1)$ (or $y' = (f_\mathrm{B}(x), 0, 1)$). Note that if there are no $\tau_\mathrm{s} \in (0, \hat{\tau})$ for the IVP (i) such that an auxiliary ODE becomes active, then $t(\tau) = \int_0^\tau \mathrm{d}\tau_1 = \tau$. Since $(f_\mathrm{A}(x), 0) = M(f_\mathrm{A}(x), 0, 1)$, $(f_\mathrm{B}(x), 0) = M(f_\mathrm{B}(x), 0, 1)$ and $z_0 = My_0$ by setting $\hat{\tau} = \tau_\mathrm{f}$, it follows that (9.7) holds.

Suppose now that we have a $\tau_\mathrm{s} \in (0, \tau_\mathrm{f})$ such that for $w(\tau_\mathrm{s}) = 1$ the auxiliary ODE $y' = f_{\mathrm{aux,A}}(y)$ becomes active (or similarly for $w(\tau_\mathrm{s}) = 0$, $y' = f_{\mathrm{aux,B}}(y)$ becomes active). From the first part of the proof we have that (9.7) holds for $\tau \in (0, \tau_\mathrm{s})$ and hence for all $t(\tau) \in (0, t_\mathrm{s}^-)$, where $t_\mathrm{s}^- = t(\tau_\mathrm{s})$. From Proposition 9.2 we have that the solution satisfies $x(\tau_\mathrm{s}) = x(\tau_\mathrm{r})$ and $w(\tau_\mathrm{r}) = 0$ (or $w(\tau_\mathrm{r}) = 1$) with $t'(\tau) = 0$ for $\tau \in [\tau_\mathrm{s}, \tau_\mathrm{r}]$. Hence, we have also $t(\tau_\mathrm{r}) = t_\mathrm{s}^+ = t(\tau_\mathrm{s})$. Denote by $y_\mathrm{s} = (x(\tau_\mathrm{r}), w(\tau_\mathrm{r}), t(\tau_\mathrm{r}))$. Using this we have $y_1(\tau - \tau_\mathrm{r}, y_\mathrm{s}) = y(\tau, y_0)$ for $\tau \in (\tau_\mathrm{r}, \tilde{\tau})$ and denoting $z_\mathrm{s} = My_\mathrm{s}$ we see that $z_1(t(\tau) - t_s; z_\mathrm{s}) = x(t(\tau), z_0)$ for $t(\tau) \in (t_s^+, \tilde{\tau})$. Assume that a single activation of an auxiliary ODE takes place and set $\tilde{\tau} = \tau_\mathrm{f}$. Since the intervals (t_s, t_f) and $(\tau_\mathrm{r}, \tau_\mathrm{f})$ have the same length and $z_\mathrm{s} = My_\mathrm{s}$ from the definitions of the corresponding IVP, we conclude that relation (9.7) holds. If the auxiliary ODE becomes active multiple times we simply apply the same argument on the corresponding sub-intervals. This completes the proof. □

The last theorem opens the door to study the regarded hybrid system with hysteresis as a Filippov system and to apply their rich theory e.g., solution existence results [94]. From the practical side, we can use numerical methods for Filippov systems, which allows us to avoid using integer variables.

9.4 Numerical example: time-optimal problem of a car with turbo charger

In this section, we apply the theoretical developments in a numerical example of a time optimal control problem of a car with a turbo from [20]. We consider a double-integrator car model equipped with a turbo accelerator that follows a hysteresis characteristic as in Figure 9.1. This makes the seemingly simple model severely nonlinear and nonsmooth. The source code of the OCP example is available within `nosnoc`.

The car is described by its position $q(t)$, velocity $v(t)$, and turbo charger state $w(t) \in \{0, 1\}$. The control variable is the car acceleration $u(t)$. The turbo accelerator is activated when the velocity exceeds $v \geq 15$ and is deactivated when it falls below $v \leq 10$. If on, it makes the nominal acceleration $u(t)$ three times greater. One can see that $\psi(x) = \frac{v-10}{5}$. In summary, the state vector reads as $z = (q, v, w) \in \mathbb{R}^3$ with two modes of operation described by $f_{\mathrm{A}}(z, u) = (v, u, 0)$ and $f_{\mathrm{B}}(z, u) = (v, 3u, 0)$. The acceleration is bounded by $|u| \leq \bar{u}$, $\bar{u} = 5$ and the velocity by $|v| \leq \bar{v}$, $\bar{v} = 25$.

In the OCP we consider the time-freezing PSS associated with the car model on a numerical time interval $\tau \in [0, \tau_{\mathrm{f}}]$. The car should reach the goal $q(t(\tau_{\mathrm{f}})) = q_{\mathrm{f}} = 150$ with $v(t(\tau_{\mathrm{f}})) = v_{\mathrm{f}} = 0$, whereby $z(0) = z_0 = \mathbf{0}_{3,1}$. The auxiliary and DAE-forming dynamics are chosen according to Propositions 9.2 (with $a = 1$) and 9.3, respectively. The OCP reads as:

$$\min_{y(\cdot), u(\cdot), s(\cdot)} \quad t(\tau_{\mathrm{f}}) \tag{9.8a}$$

$$\text{s.t.} \quad y(0) = (z_0, 0), \tag{9.8b}$$

$$y'(\tau) \in s(\tau) F_{\mathrm{TF}}(y(\tau), u(\tau)),\ \tau \in [0, \tau_{\mathrm{f}}], \tag{9.8c}$$

$$-\bar{u} \leq u(\tau) \leq \bar{u},\ \tau \in [0, \tau_{\mathrm{f}}], \tag{9.8d}$$

$$\bar{s}^{-1} \leq s(\tau) \leq \bar{s},\ \tau \in [0, \tau_{\mathrm{f}}], \tag{9.8e}$$

$$-\bar{v} \leq v(\tau) \leq \bar{v},\ \tau \in [0, \tau_{\mathrm{f}}], \tag{9.8f}$$

$$(q(\tau_\mathrm{f}), v(\tau_\mathrm{f})) = (q_\mathrm{f}, v_\mathrm{f}). \tag{9.8g}$$

The objective consists of minimizing the final physical time. Since a time optimal control problem is considered, we introduce the scalar *speed-of-time* control variable $s(\cdot)$ which introduces a time-transformation and enables to have a variable terminal physical time $T_\mathrm{f} = t(\tau_\mathrm{f})$, cf. Section 8.5. It is bounded by (9.8e) with $\bar{s} = 10$.

The OCP is discretized with a FESD Radau IIA scheme of order 3 with $N = 10$ control intervals and $N_\mathrm{FE} = 3$ additional integration steps on every control interval, with $\tau_\mathrm{f} = 5$. The controls are taken to be piecewise constant over the control intervals. The OCP discretization and MPCC homotopy is carried out via `nosnoc`, which has `IPOPT` [281] and `CasADi` [9] as a back-end.

Additionally, we compare our approach to the mixed-integer formulation of [20]. We take the same control and state discretization as in `nosnoc`, which results in 56 binary variables. Switches in the integer formulation are allowed only at the control interval boundaries, as a switch detection formulation requires significantly more integer variables and introduces more nonlinearity.

The problem is solved with the dedicated mixed-integer nonlinear programming (MINLP) solver `Bonmin` [45]. Observe that the only nonlinearity in the MINLP is due to the time transformation for the optimal time T_f. Therefore, in a second experiment, we fix T_f and solve the resulting MILP with the commercial solver `Gurobi`. We make a bisection-type search in T_f. The MILP with the smallest T_f that is still feasible, delivers the optimal time T_f. In this experiment, 22 MILPs were solved for an accuracy of 10^{-6}. To determine the solution quality, we additionally perform a high-accuracy solution with the computed optimal controls and obtain $x_\mathrm{sim}(t)$. We compare the terminal constraint satisfactions: $E(T_\mathrm{f}) = \|x_\mathrm{sim}(T_\mathrm{f}) - (q_\mathrm{f}, v_\mathrm{f})\|_2$. The source code for the simulation and the two MIOP approaches are provided in `nosnoc`'s repository [2].

The results are summarized in Table 9.1. All three approaches provide a similar objective value. `Gurobi` is the fastest solver, `nosnoc` is only slightly slower and `Bonmin` is significantly slower. The smallest terminal error is achieved via `nosnoc`, which is due to the underlying FESD discretization. `Gurobi` and `Bonmin` have the same discretization without switch detection and result in the same terminal error. On the other hand, `Gurobi` provides the most robust approach, as `nosnoc`(i.e., `IPOPT` as underlying NLP solver) fails to converge in some variations of the discretization. The results computed by `nosnoc` is depicted in Figure 9.4. One can see an intuitive behavior as the car uses the turbo accelerator as much as possible to reach the goal time optimally, with $T_\mathrm{f} = 10.26$.

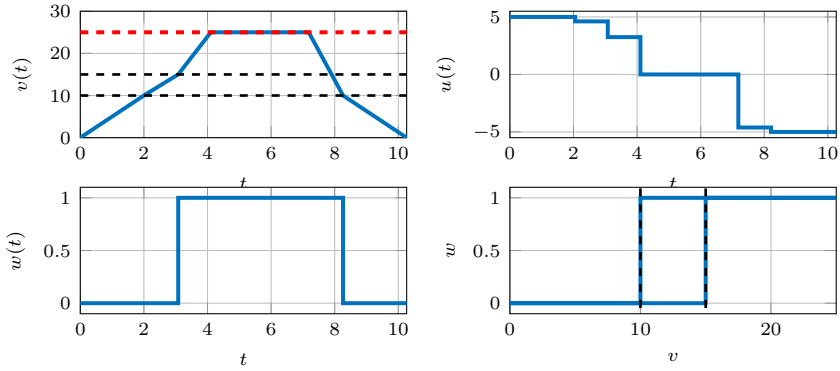


Figure 9.4: Solutions of the OCP (9.8) in physical time. The top left and right plots show the velocity $v(t)$ and optimal controls $u(t)$, respectively. The bottom left and right plots show the hysteresis state $w(t)$ and the solution trajectory in the (v, w)-plane, respectively.

Solver	T_f	CPU Time [s]	$E(T_\mathrm{f})$
`nosnoc`	**10.26**	8.87	**9.49e-02**
`Gurobi` with bisection	11.21	**5.31**	7.88e+01
`Bonmin`	11.28	1481.58	7.88e+01

Table 9.1: Comparison of `nosnoc` to mixed-integer formulations.

9.5 Conclusion

In this chapter, we extend the time-freezing reformulation to a class of hybrid systems with a hysteresis. It transforms the systems with state jumps into PSS, for which we leverage the FESD method from Chapter 7. Thus, we can avoid the use of computationally expensive mixed-integer strategies in numerical optimal control and obtain quickly good and accurate nonsmooth solutions. In the theoretical part, constructive ways to find auxiliary and DAE-forming ODE are provided and solution equivalence is proven. In the future time-freezing for other types of finite automata and hysteresis systems, as e.g., described in the introduction should be investigated as well. A first step in this direction was already taken in [274], where the ideas from this chapter are extended to a cascade of hysteresis systems.

Chapter 10

The Advanced Step Real-Time Iteration for Nonlinear Model Predictive Control

This chapter is thematically isolated from most of the content of this thesis since it regards optimal control problems subject to smooth dynamical systems. The standard definitions, discretization methods, and solution strategies for this class of problems are treated in Chapter 3. In this chapter, we develop fast and accurate numerical methods for Nonlinear Model Predictive Control (NMPC). NMPC is an advanced feedback control strategy that requires the repeated solution of discrete-time parametric OCPs in real-time. At every sampling time, the first control input is passed to the system and given the next state estimate, then next OCP is solved.

In this chapter, we introduce an algorithm for the fast (approximate) solution of such OCPs. We introduce the Advanced Step Real-Time Iteration (AS-RTI) scheme, which is an extension to the well-known Real-Time Iteration (RTI) scheme introduced by Diehl [75, 76]. We combine algorithmic ideas of the RTI, Advanced Step Controller (ASC) [290] and Multi-Level Iterations (MLI) [42], and obtain thereby a family of new algorithms. The AS-RTI allows one to trade control performance for computational efficiency in a flexible way. The main idea is to improve the linearization point for a new iteration by making cheap iterations with a new initial parameter prediction.

Outline. In Section 10.1, we provide an introduction to the problem we wish to solve. We review related work and present the algorithmic building blocks, which are needed for the AS-RTI scheme. Section 10.3 presents the AS-RTI method. Afterwards, we study the convergence properties and error bounds of the proposed methods in Section 10.4. Section 10.5 shows a numerical example, and Section 10.6 concludes this chapter. This chapter is based on [214, 215, 204].

10.1 Introduction to real-time NMPC

Nonlinear Model Predictive Control (NMPC) is increasingly becoming a standard tool in academia and industry [229]. NMPC enables one to incorporate nonlinear system dynamics and constraints directly into an Optimal Control Problem (OCP). When using NMPC to control a system, one has to solve online a sequence of parametric OCPs with different initial states. In each of these OCPs, the latest information about the system state is incorporated.

Solving optimization problems online is generally a computationally intensive task. However, in the last two decades, progress both in software [81, 82, 140, 278] and numerical algorithms [78, 170, 204, 226] made it possible to achieve computation times in the range of milli- and micro-second time scales for various kinds of applications.

In NMPC, at every sampling time, we have to solve the following OCP:

$$\min_{w} \quad \sum_{i=0}^{N-1} L_i(s_i, u_i) + L_N(s_N) \tag{10.1a}$$

$$\text{s.t.} \quad s_0 - x = 0, \tag{10.1b}$$

$$s_{i+1} - \psi(s_i, u_i) = 0, \qquad i = 0, \dots, N-1, \tag{10.1c}$$

$$g_{\mathrm{p}}(s_i, u_i) \leq 0, \qquad i = 0, \dots, N-1, \tag{10.1d}$$

$$g_{\mathrm{t}}(s_N) \leq 0, \tag{10.1e}$$

where N is the horizon length, the vectors $s_i \in \mathbb{R}^{n_s}$, and $u_i \in \mathbb{R}^{n_u}$ are the predicted states and inputs of the controlled system. The vector $w \coloneqq (w_0, w_1, \dots w_N)$ with $w_i \coloneqq (s_i, u_i)$, $i = 0, \dots, N-1$ and $w_N \coloneqq s_N$, collects all optimization variables. The objective terms $L_i : \mathbb{R}^{n_s} \times \mathbb{R}^{n_u} \to \mathbb{R}$ and $L_N : \mathbb{R}^{n_s} \to \mathbb{R}$ are the running and terminal costs, respectively. The function $\psi : \mathbb{R}^{n_s} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_s}$ describes the discrete-time system dynamics, cf. Section 3.1. The functions $g_{\mathrm{p}} : \mathbb{R}^{n_s} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_p}$ and $g_{\mathrm{t}} : \mathbb{R}^{n_s} \to \mathbb{R}^{n_t}$ define

the direct-time path and terminal constraints, respectively. The parameter x represents the initial state of the system. We assume that all functions are twice continuously differentiable. We denote the optimal solution of the parametric OCP as $\bar{w}(x)$, which can be interpreted as an implicit function of the parameter x. In NMPC, at every sampling instant a new state estimate x^k corresponding to time t^k is received. Then, after (approximately) solving the OCP, the first control input $\bar{u}_0(x^k)$ is passed to the system and held constant for a sampling time T_s.

Since feedback delays can degrade control performance, in many online algorithms, the computations are divided into an expensive and long *preparation phase*, where calculations can be performed without the knowledge of the current measurement, and a short *feedback phase* [78]. In the feedback phase, just a few calculations are performed to take into account the new measurement, such that the feedback delay can be reduced. Moreover, to reduce the computation times, many NMPC algorithms seek approximate solutions. These algorithms aim to closely track $\bar{w}(x)$ with a high sampling rate in real-time, cf. [78] for a survey.

The key to the success of real-time NMPC algorithms is that the OCP does not have to be solved to convergence but to keep the numerical error bounded over time [287, 289]. Examples of real-time NMPC algorithms are the Advanced Step Controller (ASC) [290] and the C/GMRES algorithm [216]. Among others, Sequential Quadratic Programming (SQP) based algorithms are the Real-Time Iteration (RTI) scheme [75], the Multi-Level Iteration (MLI) [42]. Many other variants of the RTI and MLI schemes can be found in the literature, cf. [204, 286, 238]. An augmented Lagrangian tracking scheme is presented in [289], and general predictor-corrector algorithms for sampled-data NMPC are described in e.g., [17].

In this chapter, we present a new variant of the RTI, which we denote as the AS-RTI. The main idea is to do inexact Newton steps, or only predictor steps on an OCP with a predicted initial value to improve the linearization point for the next RTI. We provide a convergence analysis and sufficient conditions for the boundedness of the numerical error for the AS-RTI. The benefits of the new methods are illustrated in a numerical example.

10.2 NMPC and continuation methods

The NLP (10.1) can be written in the following compact form:

$$\min_{w} \quad \phi(w) \tag{10.2a}$$

$$\text{s.t.} \quad b(w) + \hat{\Lambda} x = 0, \tag{10.2b}$$

$$c(w) \geq 0, \tag{10.2c}$$

where $\hat{\Lambda} = [-I, 0, \dots]^T$ is a suitable matrix that embeds the parameter $x \in X$ linearly and X is the set of all possible parameter values. The function $b(\cdot)$ and $c(\cdot)$ collect the equality and inequality constraints in (10.1), respectively. The function $\phi(w)$ collects the objective terms. The Lagrangian of the NLP (10.2) reads as

$$\mathcal{L}(w, \lambda, \mu) = \phi(w) - \lambda^\top b(w) - \lambda^\top \hat{\Lambda} x - \mu^\top c(w), \tag{10.3}$$

where $\lambda \in \mathbb{R}^{(N+1)n_x}$ and $\mu \in \mathbb{R}^{Nn_p+n_t}$ denote the vectors containing the Lagrange multipliers.

The NLP (10.2) can be solved to local optimality with a standard NLP algorithm, cf. Chapter 2. The AS-RTI is based on Sequential Quadratic Programming (SQP), hence we recall some basic algorithmic ingredients. Assuming we start with a primal-dual guess (w^0, λ^0, μ^0) close enough to the solution, a full SQP step is performed as

$$w^{k+1} = w^k + \Delta w^k, \;\; \lambda^{k+1} = \lambda^k_{\text{QP}}, \;\; \mu^{k+1} = \mu^k_{\text{QP}}, \tag{10.4}$$

where $(\Delta w^k, \lambda^k_{\text{QP}}, \mu^k_{\text{QP}})$ corresponds to the primal-dual solution of the QP:

$$\min_{\Delta w} \quad \frac{1}{2} \Delta w^\top A^k \Delta w + (a^k)^\top \Delta w \tag{10.5a}$$

$$\text{s.t.} \quad B^k \Delta w + b(w^k) + \hat{\Lambda} x = 0, \tag{10.5b}$$

$$C^k \Delta w + c(w^k) \geq 0, \tag{10.5c}$$

where $A^k \in \mathbb{R}^{n_w \times n_w}$ is a symmetric matrix representing the exact Hessian of the Lagrangian (10.3) or an approximation of it at the current iterate (w^k, λ^k, μ^k), $a^k = \nabla_w \phi(w^k)$ is the gradient of the cost function and B^k and C^k are the Jacobians of the constraints $b(\cdot)$ and $c(\cdot)$ at the current iterate w^k.

10.2.1 Predictor-corrector path-following methods

Let us take a closer look at the parametric NLP parametric (10.2). For ease of exposition, we assume in this section to have only equality constraints. Generalizations can be found in e.g., [85]. The KKT conditions (cf. Theorem

2.14) of this problem can be written in compact form as a parametric root-finding problem:

$$F(z,x) = \hat{F}(z) + \Lambda x = 0, \text{ with } \hat{F}(z) = \begin{bmatrix} \nabla_w \mathcal{L}(w,\lambda,\mu) \\ b(w) \end{bmatrix}, \ \Lambda = \begin{bmatrix} \mathbf{0}_{n_w,n_x} \\ \hat{\Lambda} \end{bmatrix}, \tag{10.6}$$

and where $z := (w,\lambda) \in \mathbb{R}^{n_z}$ collects the primal-dual variables. A solution for a given parameter x will be denoted as $\bar{z}(x)$. A full exact Newton step for this problem reads as:

$$z^{k+1} = z^k - \left[\frac{\partial F}{\partial z}(z^k,x)\right]^{-1}(\hat{F}(z^k) + \Lambda x). \tag{10.7}$$

Next, we regard the case where the parameter x changes with every iteration. Following the presentation in [269], if the parameter x enters F linearly, which can always be achieved via an intermediate variable [77], one step of the path-following predictor-corrector method reads as

$$\begin{aligned} z^{k+1} =& z^k - \left[\frac{\partial F}{\partial z}(z^k,x^k)\right]^{-1}(\hat{F}(z^k) + \Lambda x^k) + \left[\frac{\partial F}{\partial z}(z^k,x^k)\right]^{-1}\Lambda(x^{k+1} - x^k) \\ =& z^k - \left[\frac{\partial F}{\partial z}(z^k,x^k)\right]^{-1}(\hat{F}(z^k) + \Lambda x^{k+1}), \end{aligned} \tag{10.8}$$

where $z^{k+1} \approx \bar{z}(x^{k+1})$ is now an approximate solution for the new parameter value x^{k+1}, given $z^k \approx \bar{z}(x^k)$. This corresponds to a standard Newton step with the new parameter x^{k+1}. Note that if we keep the parameter fixed, i.e., $x^{k+1} = x^k$, the last equation reduces to a standard full Newton step (10.7), often referred to as *corrector step*, and if $z^k = \bar{z}(x^k)$ holds, then equation (10.8) reduces to

$$z^{k+1} = z^k - \left[\frac{\partial F}{\partial z}(z^k,x^k)\right]^{-1}\Lambda(x^{k+1} - x^k), \tag{10.9}$$

which is denoted as a *predictor step*, i.e., a first-order approximation of $\bar{z}(x^{k+1})$, where the sensitivities are in general obtained via the implicit function theorem. Therefore, equation (10.8) has predictive and corrective capabilities. If we consider the inequality constraints in (10.2), the map $\bar{z}(x)$ is nonsmooth but the solution manifold has smooth parts where the active set does not change and nondifferentiable points whenever the active set changes [118]. Furthermore, a *generalized tangential predictor* can be obtained by solving the QP (10.5). Such a predictor is piecewise linear, i.e., the QP can "jump" over active set changes, cf. [75, 269].

10.2.2 Algorithmic ingredients

In this section, we review the three algorithmic approaches that are the basis for the new algorithm presented in this chapter.

Real-time iterations

A widely used algorithm based on the predictor-corrector method in the SQP framework is the RTI scheme, first introduced by M. Diehl in [75]. The RTI does not distinguish between OCPs with different parameters x and iterates while the problem changes. Only one full SQP iteration is done per sampling time so that the algorithm never iterates to convergence for a fixed value of x, which ensures that it always works with the most recent state estimate and does not lose time by working on outdated information. Stability and convergence for a stable active set have been proven in [76, 79] and with active-set changes in [181, 288].

In general, the value of the current state x will not be equal to the optimization variable s_0, but since (10.5b) is linear in s_0, the constraint is satisfied after the first full Newton step. The idea to linearly embed the initial value into the NLP is known as *initial Value embedding* [75]. Moreover, in the RTI framework, each SQP iteration is divided into a longer *preparation phase* and a shorter *feedback phase*. This reordering of the computations does not create any additional overhead per iteration. In those two phases, the following calculations are performed:

- *Preparation phase*: Functions and derivatives are evaluated at the available linearization point $z^k = (w^k, \lambda^k, \mu^k)$. Sometimes the linearization point for a new iterate is adapted, e.g., with a shifting strategy [78]. Since the new measurement x enters the OCP linearly, the Hessian of the Lagrangian A^k, the gradient of the cost function a^k, and the Jacobians of the constraints B^k and C^k do not depend on x, hence they can be evaluated before a new measurement is available.
- *Feedback phase*: When the current state of the system x is available, the possibly condensed QP (10.5) is solved and the new control input $u_0^{k+1} = u_0^k + \Delta u_0^k$ can be passed to the system. Thereby, the feedback delay is reduced to solving a single QP.

Multi-level iterations

Diehl and coworkers introduced in [42] the MLI scheme, which is an extension of the RTI. The main idea is to update the matrices and vectors of the QP (10.5) at different time scales since different calculations have different computational loads. The fixed reference values for the constraint vectors and Jacobians and the objective terms at some level are provided by higher levels of the MLI. The modes or levels of the MLI can be run in parallel, and the levels can exchange information in various ways, cf. [286]. The feedback rate is determined by the fastest level. We briefly explain the original version presented in [42], and its extensions from [204].

Level A iterations. For the lowest level, denoted as level A, we assume that a reference QP (10.5) with fixed $\hat{A}, \hat{a}, \hat{B}, \hat{b}, \hat{C}$ and $\hat{c}$ is given. At every iteration, the following QP is solved:

$$\min_{\Delta w} \quad \frac{1}{2}\Delta w^\top \hat{A}\Delta w + \hat{a}^\top \Delta w \tag{10.10a}$$

$$\text{s.t.} \quad \hat{B}^k \Delta w + \hat{b} + \Lambda x = 0, \tag{10.10b}$$

$$\hat{C}^k \Delta w + \hat{c} \geq 0, \tag{10.10c}$$

The iterations start from a reference solution $(\hat{w}, \hat{\lambda}, \hat{\mu})$, where the reference QP is provided by higher levels of the MLI. The goal of the lowest level of the MLI is to provide feedback as fast as possible and to take at least the active set changes into account. All matrices and vectors of the reference QP are held unchanged, only a new initial value x is embedded, and the feedback $\hat{u}_0 + \Delta u_0^k$ is sent to the system. In a predictor-corrector setting this level has just a predictor part and is equivalent to linear Model Predictive Control (MPC) [75, 116]. In combination with higher levels of the MLI scheme, the reference QP changes and level A can be interpreted as an adaptive linear MPC [42].

Level B iterations. In this level, all constraint vectors in the QP (10.5) are updated, namely $\hat{b}$, $\hat{c}$. That is, new function evaluations are performed, and $\hat{a}$ is possibly updated in an approximate fashion:

$$a^{k+1} = \hat{a} + \hat{A}(w^k - \hat{w}). \tag{10.11}$$

These iterations converge to a suboptimal, but feasible solution of the original NLP (10.2).

Level	Necessary evaluations				Update formula for QP data			
	b^k, c^k	a^k	B^k, C^k	A^k	$b,\ c$	a	B, C	A
D2	✓	✓	✓	✓	b^k, c^k	a^k	$B^k,\ C^k$	A^k
D1	✓	✓	✓	✓	b^k, c^k	a^k	B^k, C^k	A^k_{QN}
D0	✓	✓	✓	✗	b^k, c^k	a^k	B^k, C^k	$A^k_{\mathrm{GN}}/\hat{A}$
C	✓	✓	✓	✗	b^k, c^k	(10.12)	$\hat{B}, \hat{C}$	$\hat{A}$
B	✓	✗	✗	✗	b^k, c^k	(10.11)	$\hat{B}, \hat{C}$	$\hat{A}$
A	✗	✗	✗	✗	$\hat{b}, \hat{c}$	$\hat{a}$	$\hat{B}, \hat{C}$	$\hat{A}$

Table 10.1: Computations and update formulas for the QP data for the different MLI levels.

Level C iterations. In addition to the level B computations, here, the objective gradient a^k is updated as well. Level C iterations are based on an adjoint SQP algorithm [42] and a^k is calculated via

$$a^k = \nabla_w \mathcal{L}(w^k, \lambda^k, \mu^k) + \hat{B}^\top \lambda^k + \hat{C}^\top \mu^k. \tag{10.12}$$

Here, the Jacobians of the constraints do not need to be evaluated, $\nabla_w \mathcal{L}(w^k, \lambda^k, \mu^k)$ can be computed efficiently with the reverse mode of automatic differentiation with a cost approximately five times higher than for the evaluation of (10.3) [115]. Level C iterations can be shown to converge to optimal solutions [42].

Level D iterations. This level is essentially the RTI, where all QP data (except possibly the Hessian) is updated. We distinguish between D0 iterations where new constraints Jacobians (or their approximations) are computed, and the Hessian is kept constant. For a quadratic cost function, which is usually the case in tracking NMPC formulations, the Gauss-Newton approximations result in a constant Hessian matrix, hence the GN-based RTI variant falls also in this level. In D1, we update the Hessian matrix via a quasi-Newton update formula, e.g., SR1 or BFGS [201]. In level D2, we compute the exact Hessian. The different MLI levels are summarized in Table 10.1. This table is inspired by [239]. Thereby, we use the compact notation $b^k = b(w^k), c^= c(w^k), B^k = \nabla b(w^k)^\top, C^k = \nabla c(w^k)^\top, a^k = \nabla \phi(w^k)$ and $A^k = \nabla^2_{ww} \mathcal{L}(w^k, \lambda^k, \mu^k)$. The matrix A^k_{GN} denotes a Gauss-Newton approximation of the Hessian (if possible) and A^k_{QN} a quasi-Newton Hessian approximation.

The advanced step controller

To avoid the possible convergence issues of a predictor-corrector algorithm performing just one iteration, in the Advanced Step Controller (ASC) by Zavala and Biegler [290] the NLP (10.2) is solved to convergence with an interior-point method. This is computationally more expensive but yields an accurate locally optimal solution. To take the feedback delay into account, this online algorithm solves an *advanced problem* in the preparation phase with a predicted state $\tilde{x}$ as the initial value. Furthermore, the solution is not applied directly to the system, but an additional linear system solution based on the last Newton iteration's matrix factorization is performed in the feedback phase. This provides a *tangential predictor* to correct for the mismatch between the predicted $\tilde{x}$ and actual measurement x. While such a tangential predictor can not "jump" over active set changes solving a linear system is often cheaper than solving a QP. Jäschke et al. [152] introduce an extension for the ASC, to handle active-set changes and nonunique Lagrange multipliers within a path-following algorithm.

10.3 The advanced step real-time iteration

The RTI and MLI schemes perform a single (inexact) SQP iteration per sampling time, which might lead to convergence issues and larger numerical errors. The ASC solves an OCP to local optimality, which might be computationally too expensive for a given sampling rate. The MLI scheme contains all the ingredients one needs to refine a solution while still keeping the computational burden low. Instead of solving the advanced problem to convergence, we propose to use some level of the MLI to iterate on this problem to obtain an improved guess $\tilde{z}^{k+1}$. We call this method the Advanced Step Real-Time Iteration (AS-RTI).

The AS-RTI enables one to trade off computational load for controller performance in a flexible way. This approach is computationally more expensive than the standard RTI but alleviates the possible convergence issues and suboptimality of solutions and delivers a predictor that can "jump" over active set changes. The limiting cases would be: (a) doing just one level A iteration in the preparation phase and (b) the full convergent SQP as explained above. In between, there is a wide family of possible algorithms as we can now assemble the preparation phase differently. We call the last QP to solve *outer iteration* and the calculations in the preparation phase *inner iterations*. This approach is summarized in Algorithm 2. It is reasonable to assume that, if $\tilde{x}^{k+1}$ is close to x^{k+1}, we might have fewer active set changes between the corresponding parametric NLPs. Hence, with AS-RTI we determine the active set of the advanced problem, so QP solvers that can be warm-started as `qpOASES` [90] or

Algorithm 2 Single Advanced step real-time iteration

1: **Input**: z^k, QP data at iteration k, state estimate x^{k+1} **Output:** z^{k+1}
Preparation Phase:
2: Predict $\tilde{x}^{k+1}$ with $\tilde{x}^{k+1} = f(x^k, u_0^k)$
3: Predict optimal solution (approximation) $\tilde{z}^{k+1}$ for $\tilde{x}^{k+1}$ by iterating with some mode of the MLI on (10.2) parameterized by $\tilde{x}^{k+1}$
4: Evaluate all functions and derivatives at $\tilde{z}^{k+1}$ needed for the QP (10.5)
5: Possibly condense the QP (10.5)
Feedback phase:
6: Embed current state estimate x^{k+1} into the QP (10.5)
7: Solve QP, compute next iterate z^{k+1} via (10.4) and send first control u_0^{k+1} to the system

first-order methods as `OSQP` [22] will have a better initial guess and the feedback delay is reduced.

We proceed with analyzing the simplest case of Algorithm 2, namely performing just another QP solve (level A iteration) with respect to the standard RTI. Note that in this variant of the AS-RTI, two QPs are solved per sampling time. The next proposition shows that in the simplest variant of the AS-RTI with a perfect prediction of the parameter, the RTI and this AS-RTI variant have the same linearization points. However, the AS-RTI achieves better tracking of the solution manifold. Of course, a perfect prediction will never be available in practice, but we want to show that the reordering of calculations brings us closer to the solution manifold. To simplify the analysis, we assume a fixed active set.

Proposition 10.1. *Suppose a single level A iteration is carried out at line 3 of Algorithm 2 and a perfect prediction of the parameter x is available, then the standard RTI and 2 have the same linearization points.*

Proof. A perfect prediction means $\tilde{x}^k = x^k$, for all $k \geq 0$. Let $\tilde{z}^k$ be the linearization point at iteration k, then the output of the feedback phase of the AS-RTI obtained via a predictor-corrector iteration (10.8) reads as

$$\begin{aligned} z^k =& \tilde{z}^k - \left[\frac{\partial F}{\partial z}(\tilde{z}^k, x^k)\right]^{-1}(\hat{F}(\tilde{z}^k) + \Lambda x^k) - \left[\frac{\partial F}{\partial z}(\tilde{z}^k, x^k)\right]^{-1}\Lambda(\tilde{x}^k - x^k) \\ =& \tilde{z}^k - \left[\frac{\partial F}{\partial z}(\tilde{z}^k, x^k)\right]^{-1}(\hat{F}(\tilde{z}^k) + \Lambda x^k), \end{aligned} \tag{10.13}$$

i.e., we have just the corrector step since the prediction is perfect. At the next iteration $(k+1)$, in the preparation phase of Algorithm 2 we have the

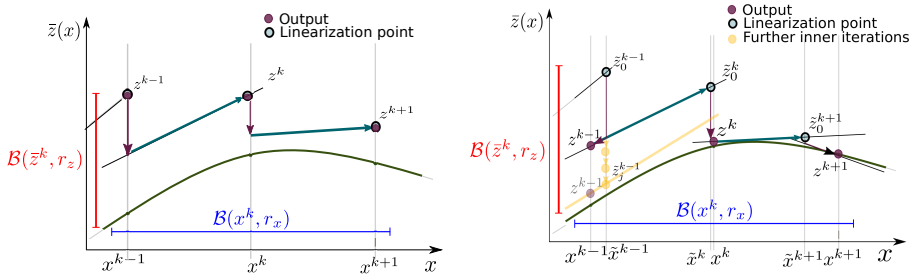


Figure 10.1: Tangential predictors and solution manifold tracking with the RTI (left plot) and AS-RTI (right plot). The linearization point is due to the inner iterations refined.

linearization point prediction (line 3) using (10.9)

$$\begin{aligned}\tilde{z}^{k+1} &= z^k - \left[\frac{\partial F}{\partial z}(\tilde{z}^k, x^k)\right]^{-1} \Lambda(\tilde{x}^{k+1} - x^k) \\ &\overset{(10.13)}{=} \tilde{z}^k - \left[\frac{\partial F}{\partial z}(\tilde{z}^k, x^k)\right]^{-1} (\hat{F}(\tilde{z}^k) + \Lambda(x^k + \tilde{x}^{k+1} - x^k)) \\ &= \tilde{z}^k - \left[\frac{\partial F}{\partial z}(\tilde{z}^k, x^k)\right]^{-1} (\hat{F}(\tilde{z}^k) + \Lambda x^{k+1}),\end{aligned}$$

which is a standard RTI. Using induction, this holds for all $k \geq 0$. □

When we iterate on a problem with a fixed parameter x^k we equip the variables with a second index z_j^k, $j \geq 0$, which counts the number of iterations for a fixed x^k. Figure 10.1 illustrates the linearization points, outputs, and tangential predictors of the RTI (left plot) scheme and the AS-RTI (right plot) with an extra QP solve in the preparation phase, respectively. In the AS-RTI case, we use the tangential predictor from the previous iteration and compute $z(\tilde{x}^{k+1})$ as the new linearization point for the output iteration. Observe that $\tilde{z}_0^k$ is on the same tangent as the previous output z^{k-1}. When implementing this, one has to take care not to add the same corrector twice. The QP solution $(\Delta w^k, \lambda_{\text{QP}}^k, \mu_{\text{QP}}^k)$ has both predictor (moving along the tangent) and corrector (getting closer to the manifold) properties. Now in the AS-RTI solving an extra QP (10.5), with $\tilde{x}^{k+1}$ one obtains $(\Delta\tilde{w}^{k+1}, \tilde{\lambda}_{\text{QP}}^{k+1}, \tilde{\mu}_{\text{QP}}^{k+1})$. This should be added to $(w^{k-1}, \lambda^{k-1}, \mu^{k-1})$ and not to (w^k, λ^k, μ^k), otherwise we add the same corrector twice. Doing further iterations with optimality-improving levels of the MLI will bring the solution approximation closer to $\bar{z}(x^{k+1})$.

10.4 Contraction theory for the AS-RTI

In this section, we study how the numerical errors evolve depending on how many iterations are carried out. To make the analysis more general, we consider now parametric NLPs with inequality constraints. To carry out the convergence analysis we first introduce some tools using generalized equations. Afterwards, the AS-RTI-specific results are presented.

10.4.1 Contraction estimate for abstract real-time algorithms

We rewrite the parametric NLP (10.2) in the following more abstract form

$$\min_{w} \quad \psi(w) \tag{10.14a}$$

$$\text{s.t.} \quad b(w) + \hat{\Lambda} x = 0, \tag{10.14b}$$

$$w \in \Omega, \tag{10.14c}$$

with $w \in \mathbb{R}^n$ and where $\Omega = \{w \mid c(w) \geq 0\}$ is a nonempty, closed and convex set. Note that set Ω can always be made convex by introducing slack variables and isolating the nonconvex parts into the equality constraints.

Following [269], the KKT conditions of the NLP (10.14) can be written as

$$\nabla \psi(w) + \nabla b(w)^\top \lambda + \mathcal{N}_\Omega(w) \ni 0, \tag{10.15a}$$

$$b(w) + \hat{\Lambda} x = 0. \tag{10.15b}$$

The set-valued function $\mathcal{N}_\Omega(w)$ is the normal cone to Ω at w, cf. Definition 2.6. Now, introducing $z \coloneqq (w, \lambda)$, $K = \Omega \times \mathbb{R}^{n_g}$, we have that

$$F(z) \coloneqq \begin{bmatrix} \nabla \psi(w) + \nabla b(w)^\top \lambda \\ b(w) \end{bmatrix},$$

and equation (10.15) can be rewritten as a generalized equation

$$F(z) + \Lambda x + \mathcal{N}_K(z) \ni 0. \tag{10.16}$$

We define its solution mapping as

$$\bar{Z}(x) \coloneqq \{z \mid F(z) + \Lambda x + \mathcal{N}_K(z) \ni 0\},$$

which is the set of KKT points of (10.14) for a given x. We define the following concept, which somewhat generalizes the condition of invertibility of the Jacobian of single to set-valued maps [85].

Definition 10.2 (Strong Regularity [231]). *Let $\bar{z}(x) \in \bar{Z}(x)$ and let $\nabla F(z)^\top$ be the Jacobian of $F(z)$. We say that* (10.16) *is strongly regular at $\bar{z}(x)$, if there exist neighborhoods $\mathcal{B}(0, \bar{r}_\delta)$ and $\mathcal{B}(0, \bar{r}_z)$ such that the linearized generalized equations*

$$F(\bar{z}(x)) + \nabla F(\bar{z}(x))^\top \Delta z + \Lambda x + \mathcal{N}_K(\bar{z}(x) + \Delta z) \ni \delta,$$

with unknown variable Δz, has a unique solution in $\mathcal{B}(0, \bar{r}_z)$ and its solution is Lipschitz continuous in $\mathcal{B}(0, \bar{r}_\delta)$ with a Lipschitz constant σ:

$$\|\Delta\bar{z}(\delta') - \Delta\bar{z}(\delta)\| \leq \sigma\|\delta' - \delta\|, \quad \forall \delta', \delta \in \mathcal{B}(0, \bar{r}_\delta).$$

In the context of optimization, a point $\bar{z}(x)$ is strongly regular if it satisfies the LICQ and the strong second-order sufficient condition, cf. [149, Proposition 1.28]. More details about this concept can be found in the seminal paper of [231], and its application in real-time optimization can be found in e.g. [17, 269, 287, 289]. We make the following regularity assumption.

Assumption 10.3. *The set $\bar{Z}(x)$ is nonempty, and the corresponding generalized equation* (10.16) *is strongly regular at $\bar{z}(x)$ for all $x \in X$.*

The following lemma provides conditions under which the solution manifold $\bar{z}(x)$ is Lipschitz continuous.

Lemma 10.4 (Lemma 3.3, [269]). *Let Assumption 10.3 hold. Then, for any x in X, there exist neighborhoods $\mathcal{B}(x, \tilde{r}_x)$ of x and $\mathcal{B}(\bar{z}(x), \tilde{r}_z)$ of $\bar{z}(x)$, such that the generalized equation* (10.16) *has a unique solution in $\mathcal{B}(\bar{z}(x), \tilde{r}_z)$ for any $x' \in \mathcal{B}(x, \tilde{r}_x)$. Moreover, there exists a positive constant $\gamma \geq 0$ such that the following holds:*

$$\|\bar{z}(x') - \bar{z}(x)\| \leq \gamma\|x' - x\|, \ \forall x', x \in \mathcal{B}(x, \tilde{r}_x). \tag{10.17}$$

Denote by $\bar{z}^k := \bar{z}(x^k)$, and the associated error by $e^k := z^k - \bar{z}^k$. When we solve repeatedly (10.16) for some fixed x^k, we will equip the iterations with a second index z_j^k, $j \geq 0$, which counts the number of iterations for a fixed x^k. In AS-RTI scheme this corresponds to the *inner iterations* and the corresponding error reads as $e_j^k := z_j^k - \bar{z}^k$. If we use a predicted parameter $\tilde{x}^{k+1}$, we denote the corresponding iterate as $\tilde{z}_j^{k+1}$ and the error as $\tilde{e}_j^{k+1} := \tilde{z}_j^{k+1} - \bar{z}(\tilde{x}^{k+1})$, $j \geq 0$.

We recall some results from [287], which we use to analyze the contraction properties of the AS-RTI when using Q-linearly convergent algorithms. We can analyze real-time methods for the parametric NLP (10.14) independently of the concrete numerical scheme used to solve it. For this, we require the use of

algorithms that calculate an element of $\bar{Z}(x)$ for some x that have at least local Q-linear convergence. The properties of a Q-linearly convergent algorithm are summarized in the following assumption.

Assumption 10.5 (Q-linear Convergence)**.** *There exists a radius $\hat{r}_z$ such that, for any given $\bar{z}(x^k) \in \bar{Z}(x^k)$, any $x^k \in X$, and any z_j^k in $\mathcal{B}(\bar{z}(x^k), \hat{r}_z)$, the algorithm used to solve* (10.16) *can produce z_{j+1}^k such that*

$$\|e_{j+1}^k\| \leq \left(\kappa + \frac{\omega}{2}\|e_j^k\|\right)\|e_j^k\|, \tag{10.18}$$

for some positive constants $0 \leq \kappa < 1$ and $0 \leq \omega < \infty$.

From now on, we denote $\alpha_j^k := (\kappa + \frac{\omega}{2}\|e_j^k\|)$, and analogously for the predicted parameter we adapt the notation $\tilde{\alpha}_j^k := (\kappa + \frac{\omega}{2}\|\tilde{e}_j^k\|)$. This assumption covers many standard algorithms for solving the NLP (10.14).

The next lemma provides a bound on the numerical error due to performing a single iteration of a Q-linearly convergent algorithm for solving (10.16) for each new parameter x^k.

Lemma 10.6 (Lemma 1, [287])**.** *Suppose Assumptions 10.3 and 10.5 hold. Then there exist some constants $r_z > 0$ and $r_x > 0$, such that, for any z^k in $\mathcal{B}(\bar{z}^k, r_z)$, and any x^{k+1} in $\mathcal{B}(x^k, r_x)$, the following inequality holds*

$$\begin{aligned}\|e^{k+1}\| \leq \kappa\|e^k\| + c_1\|e^k\|\|x^{k+1} - x^k\| + c_2\|e^k\|^2 \\ + c_3\|x^{k+1} - x^k\| + c_4\|x^{k+1} - x^k\|^2,\end{aligned} \tag{10.19}$$

with $c_1 := \omega\gamma$, $c_2 := \frac{\omega}{2}$, $c_3 := \kappa\gamma$, $c_4 := \frac{\omega\gamma^2}{2}$.

Now we have all the tools to state sufficient conditions for a bounded numerical error for tracking the optimal solution manifold $\bar{z}(x)$.

Theorem 10.7 (Theorem 1, [287])**.** *Suppose that Assumptions 10.3 and 10.5 hold. There exists a positive constant $0 < r_x^s < r_x$ (same as in* Lemma *10.6), such that, if $\|x^{k+1} - x^k\| \leq r_x^s$ for all $k \geq 0$ and $\|e^0\| \leq r_z$, then*

$$\|e^{k+1}\| \leq r_z, \ \forall k \geq 0, \tag{10.20}$$

where

$$r_x := \begin{cases} \frac{\sqrt{(c_3 + r_z c_1)^2 - 4c_4(\kappa - 1 - c_2 r_z)r_z} - (c_3 + r_z c_1)}{2c_4} & \text{if } c_4 > 0, \\ \frac{(1 - \kappa - c_2 r_z)r_z}{c_3 + r_z c_1} & \text{if } c_4 = 0. \end{cases} \tag{10.21}$$

10.4.2 Contraction properties of the AS-RTI scheme

We have now introduced all tools to present the main theoretical results of this chapter. We derive sufficient conditions for the contraction of the iterates of the AS-RTI scheme if the inner and outer iterations are carried out with Q-linearly convergent algorithms. Furthermore, we provide sufficient conditions for the boundedness of the numerical error.

First, we will assume the initialization of a Q-linearly convergent algorithm. This will ensure that the numerical error e_j^k for a fixed x^k gets smaller if we perform further iterations of the algorithm. This is needed in the proof of the next theorem.

Assumption 10.8. *(Initialization) Suppose that the following condition holds at an initial point z^0 and a solution $\bar{z}^0$:*

$$\|z^0 - \bar{z}^0\| \leq \hat{r}_z < \hat{r}_z^s \coloneqq 2(1-\kappa)/\omega. \tag{10.22}$$

Furthermore, we assume to always have a reasonably good parameter prediction, so that we can use the results of Theorem 10.7.

Assumption 10.9. *(Predicted Parameter) In all iterations of Algorithm 2 the parameter predictions $\tilde{x}^{k+1}$ satisfy:*

$$\|x^{k+1} - \tilde{x}^{k+1}\| \leq r_x^s \text{ and } \|\tilde{x}^{k+1} - x^k\| \leq r_x^s, \quad \forall k \geq 0, \tag{10.23}$$

where the positive constant r_x^s is the same as in Theorem 10.7.

When we use different Q-linearly convergent algorithms for inner and outer iterations, we equip the corresponding constants from Assumption 10.5 or Lemma 10.6 with a superscript *in* and *out*, respectively. For example, in Assumption 10.5 we distinguish between κ^{in} and κ^{out} for inner and outer iterations, respectively. With the next theorem, we provide a general contraction estimate for an iteration of the AS-RTI scheme.

Theorem 10.10. *Suppose that Assumptions 10.3, 10.5, 10.8 and 10.9 hold. Moreover, assume that we make $j \geq 0$ inner iterations on the NLP* (10.14) *parametrized by $\tilde{x}^{k+1}$ in the preparation phase of AS-RTI and that $r_z = \min(r_z^{out}, r_z^{in})$. Then, for the sequence of errors $\{e^k\}$, the following inequality holds:*

$$\|e^{k+1}\| \leq (\tilde{\alpha}_0^{k+1})^j \left[\nu^k \|e^k\| + \zeta^k \|\tilde{x}^{k+1} - x^k\|\right] + \eta^k \|x^{k+1} - \tilde{x}^{k+1}\|, \tag{10.24}$$

where we have defined the positive constants $\hat{\nu}^k$, ν^k, ζ^k, η^k, respectively, as:

$$\hat{\nu}^k \coloneqq \kappa^{out} + c_1^{out} \|x^{k+1} - \tilde{x}^{k+1}\| + c_2^{out} (\tilde{\alpha}_0^{k+1})^j \|\tilde{e}_0^{k+1}\|, \tag{10.25}$$

$$\nu^k := \hat{\nu}^k\big(\kappa^{in} + c_1^{in}\|\tilde{x}^{k+1} - x^k\| + c_2^{in}\|e^k\|\big), \tag{10.26}$$

$$\zeta^k := \hat{\nu}^k\big(c_3^{in} + c_4^{in}\|\tilde{x}^{k+1} - x^k\|\big), \tag{10.27}$$

$$\eta^k := c_3^{out} + c_4^{out}\|x^{k+1} - \tilde{x}^{k+1}\|. \tag{10.28}$$

Proof: To use the same r_z for inner and outer iterations we set $r_z = \min\{r_z^{\text{out}}, r_z^{\text{in}}\}$. For the first inner iteration, due to Assumptions 10.3 and 10.5 there exist some constants $r_z > 0$ and $r_x > 0$, such that, for any z^k in $\mathcal{B}(\bar{z}^k, r_z)$. Moreover, due to Assumption 10.9, it holds that $\tilde{x}^{k+1}$ in $\mathcal{B}(x^k, r_x)$. Using Lemma 10.6, it holds that

$$\begin{aligned}\|\tilde{e}_0^{k+1}\| \leq \big(\kappa^{\text{in}} + c_1^{\text{in}}\|\tilde{x}^{k+1} - x^k\| + c_2^{\text{in}}\|e^k\|\big)\|e^k\| \\ + \big(c_3^{\text{in}} + c_4^{\text{in}}\|\tilde{x}^{k+1} - x^k\|\big)\|\tilde{x}^{k+1} - x^k\|.\end{aligned} \tag{10.29}$$

Due to Assumption 10.9 and Theorem 10.7, it holds that $\|\tilde{e}_0^{k+1}\| \leq r_z \leq \hat{r}_z$. Assuming we made $j \geq 0$ iterations, due to Assumptions 10.5 and 10.8 the following holds

$$\tilde{\alpha}_1^{k+1} = \kappa^{\text{in}} + c_1^{\text{in}}\|\tilde{e}_1^{k+1}\| \overset{(10.18)}{\leq} \kappa^{\text{in}} + c_1^{\text{in}}\underbrace{\tilde{\alpha}_0^{k+1}}_{<1}\|\tilde{e}_0^{k+1}\| < \kappa^{\text{in}} + c_1^{\text{in}}\|\tilde{e}_0^{k+1}\| = \tilde{\alpha}_0^{k+1}.$$

Moreover, applying this inductively, we have

$$\|\tilde{e}_j^{k+1}\| \leq (\tilde{\alpha}_0^{k+1})^j\|\tilde{e}_0^{k+1}\| < r_z. \tag{10.30}$$

Furthermore, due to the last inequality it holds that $\tilde{z}_j^{k+1} \in \mathcal{B}(\bar{z}(\tilde{x}^{k+1}), r_z)$ and x^{k+1} in $\mathcal{B}(\tilde{x}^{k+1}, r_x)$ (holds due to Assumption 10.9). Therefore, we can use Lemma 10.6 for the *outer iteration* which yields

$$\begin{aligned}\|e^{k+1}\| \leq \big(\kappa^{\text{out}} + c_1^{\text{out}}\|x^{k+1} - \tilde{x}^{k+1}\| + c_2^{\text{out}}\|\tilde{e}_j^{k+1}\|\big)\|\tilde{e}_j^{k+1}\| \\ + \underbrace{\big(c_3^{\text{out}} + c_4^{\text{out}}\|x^{k+1} - \tilde{x}^{k+1}\|\big)}_{\overset{(10.28)}{=}\eta^k}\|x^{k+1} - \tilde{x}^{k+1}\|.\end{aligned}$$

Using the estimate for $\|\tilde{e}_j^{k+1}\|$ from (10.30), from the last equation we get

$$\begin{aligned}\|e^{k+1}\| \leq \underbrace{\big(\kappa^{\text{out}} + c_1^{\text{out}}\|x^{k+1} - \tilde{x}^{k+1}\| + c_2^{\text{out}}(\tilde{\alpha}_0^{k+1})^j\|\tilde{e}_0^{k+1}\|\big)}_{\overset{(10.25)}{=}\hat{\nu}^k} \\ \cdot (\tilde{\alpha}_0^{k+1})^j\|\tilde{e}_0^{k+1}\| + \eta^k\|x^{k+1} - \tilde{x}^{k+1}\|.\end{aligned}$$

Now, if we replace $\|\tilde{e}_0^{k+1}\|$ with its upper bound (10.29), we obtain

$$\|e^{k+1}\| \leq (\tilde{\alpha}_0^{k+1})^j \hat{\nu}^k \big[\big(\kappa^{\text{in}} + c_1^{\text{in}} \|\tilde{x}^{k+1} - x^k\| + c_2^{\text{in}} \|e^k\|\big) \|e^k\| + \big(c_3^{\text{in}} + c_4^{\text{in}} \|\tilde{x}^{k+1} - x^k\|\big) \|\tilde{x}^{k+1} - x^k\| \big] + \eta^k \|x^{k+1} - \tilde{x}^{k+1}\|.$$

Using the definitions of ν^k in (10.26) and ζ^k in (10.27), the inequality (10.24) follows from the last inequality. This completes the proof. □

Similar to *Theorem* 10.7, we give sufficient conditions for the boundedness of the numerical error of the AS-RTI.

Proposition 10.11. *Suppose that Assumptions 10.3, 10.5, 10.8 and 10.9 hold. Moreover, assume that we perform $j \geq 0$ inner iterations on the NLP* (10.14) *parametrized by $\tilde{x}^{k+1}$ in the preparation phase of AS-RTI. Then there exists a positive constant $0 < r_x^s < r_x$ (same as in Theorem 10.7), such that, if $\|x^{k+1} - x^k\| \leq r_x^s$ for all $k \geq 0$ and $\|e^0\| \leq r_z$, then*

$$\|e^{k+1}\| \leq r_z, \ \forall k \geq 0, \tag{10.31}$$

where r_x^s is given by (10.21) *and $r_z = \min(r_z^{out}, r_z^{in})$.*

Proof: Taking $r_z = \min(r_z^{\text{out}}, r_z^{\text{in}})$ we can use the same r_z for both inner and outer iterations. Since the assumptions of Theorem 10.7 are satisfied, for the first inner iteration corresponding to the NLPs (10.14) parametrized by x^0 and $\tilde{x}^1$ we conclude that $\|\tilde{e}_0^1\| \leq r_z \leq \hat{r}_z$. Using this inequality and Assumptions 10.5 and 10.8, we conclude that for a fixed parameter (further inner iterations) $\tilde{\alpha}_j^1 < 1$ for all $j \geq 0$ (the errors shrinks), i.e., it holds that $\|\tilde{e}_j^1\| < r_z$, $\forall j \geq 0$. For the outer iteration corresponding to the NLPs (10.14) parametrized by $\tilde{x}^1$ and x^1, since $\|\tilde{e}_j^1\| < r_z$ and $\|x^1 - \tilde{x}^1\| \leq r_x^s$, by applying Theorem 10.7 we obtain that $\|e^1\| \leq r_z$. Applying this argument inductively, we conclude that (10.31) holds $\forall k \geq 0$. □

From the results of Theorem 10.10 we can make several observations regarding a new iterate z^{k+1}:

1. having a better parameter guess, i.e., having smaller $\|x^{k+1} - \tilde{x}^{k+1}\|$, decreases the distance to $\bar{z}^{k+1}$,
2. being closer to the solution in the previous iterate, i.e., smaller $\|e^k\|$, also improves the solution,
3. increasing the number of inner iterations j further decreases the error e^{k+1},

4. the distance between the two parameters also $\|x^{k+1} - x^k\|$ affects the numerical error,

5. the value of the constant κ^{in} for the inner iterations affects the solution since for smaller κ^{in} the term $(\tilde{\alpha}_0^{k+1})^j$ shrinks faster.

Several inner iterations will make the first term on the r.h.s of (10.24) become very small, which implies a smaller error. Furthermore, having $j \to \infty$ in the limit and $\tilde{x}^{k+1} = x^{k+1}$ we obtain ideal NMPC in the nominal case.

10.5 Numerical example

In this section, we investigate the performance of the AS-RTI method on a numerical example. Further and more excessive numerical comparisons on the example of electric microgrids and wind turbine control can be found in [204, 210, 211].

Problem description

We consider a diesel generator (DG), which is connected with a power line to a time-varying load. A similar example, with an additional photo-voltaic source, was considered in [239]. A typical DG consists of a synchronous generator (SG) with a governor (GOV) and an automatic voltage regulator (AVR), as depicted in Figure 10.2. We consider an SG model with five differential and 11 algebraic states [173]. The goal of the GOV is to control the power generation P_1 of the DG by controlling the diesel engine. The input of the GOV is the generator frequency ω, as well as the reference power P_{ref}. We use a standard IEEE DEGOV1 model, which consists of 8 differential and 2 algebraic states. The AVR controls the terminal voltage of the generator through the field winding voltage E_{fd} from the exciter. The inputs to the AVR are the reference voltage V_{ref} and the DG voltage V_1. For the AVR, we use the standard IEEE AC5A model, which consists of 5 differential and one algebraic state. For further details on DG and general microgrid modeling, we refer to [211].

The DG has a nominal power of $S_N = 325$ kVA and the control variables are $u(t) = (P_{\text{ref}}(t), V_{\text{ref}}(t))$. The admittance of the power line is $Y_{12} = 137.93 - 344.83i\ \Omega^{-1}$, and it connects the DG at node 1 with the load at node 2. The load is modeled as a time-varying parameter. The DG, the power line, and load, where the connection is modeled via power-flow, cf. [173], result in a Differential Algebraic Equation (DAE) of index-1 with 17 differential states

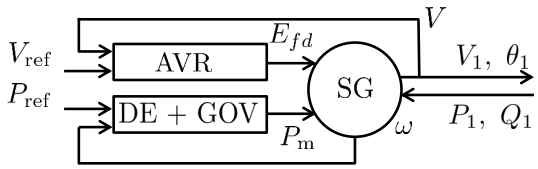


Figure 10.2: Outline of a DG model. It consists of a synchronous generator (SG), automatic voltage regulator and exciter (AVR), and a diesel engine (DE) and governor model (GOV).

$x(t)$ and 18 algebraic states $z(t)$. Our goal is to regulate the voltage at the load V_2 and the electric frequency ω of the DG at 1 per unit (p.u.). We express this with the objective

$$L(x(t), z(t), u(t)) = \|\omega(t) - 1\|^2 + \|V_2(t) - 1\|^2. \tag{10.32}$$

Moreover, we require the voltages at the DG (V_1) and load (V_2), and the frequency ω to be in specific ranges:

$$0.9 \text{ p.u.} \leq V_i \leq 1.1 \text{ p.u.}, \quad i = 1, 2, \tag{10.33a}$$

$$0.95 \text{ p.u.}, \leq \omega \leq 1.05 \text{ p.u.}. \tag{10.33b}$$

Additionally, the production of active and reactive power by the DGs is limited by its nominal power

$$P_1^2 + Q_1^2 \leq S_N^2. \tag{10.34}$$

Together with the objective (10.32), the discretized DAE model of the DG, and the constraints (10.33) and (10.34) evaluated at the discretization grid we obtain an OCP of the form (10.1). To discretize the continuous time dynamics, we use direct multiple shooting [44] with the Gauss-Legendre Implicit Runge-Kutta scheme of order four with a fixed step-size $h = T/N$. For the NMPC prediction horizon, we chose $T = 10$ s. We perform the numerical benchmark with two different discretization grids, where the trajectories are discretized using $N = 50$ and $N = 40$ multiple shooting nodes, which results in a sampling time of $T_s = 200$ ms and $T_s = 250$ ms, respectively. We use a time-varying load profile at node 2 with $P_2 = 300\,\text{kW}$ and $Q_2 = 100,$ kVAr with a scheduled load increase at $t = 4$ s to $P_2 = 305$ kW and $Q_2 = 100$ kVAr, which is also in the NMPC prediction. At $t = 1$ s an unforeseen load drop to $P_2 = 30$kW and $Q_2 = 10$ kVAr occurs until the scheduled load increases. After noticing the load drop, the prediction is adapted to the new value after one sampling time.

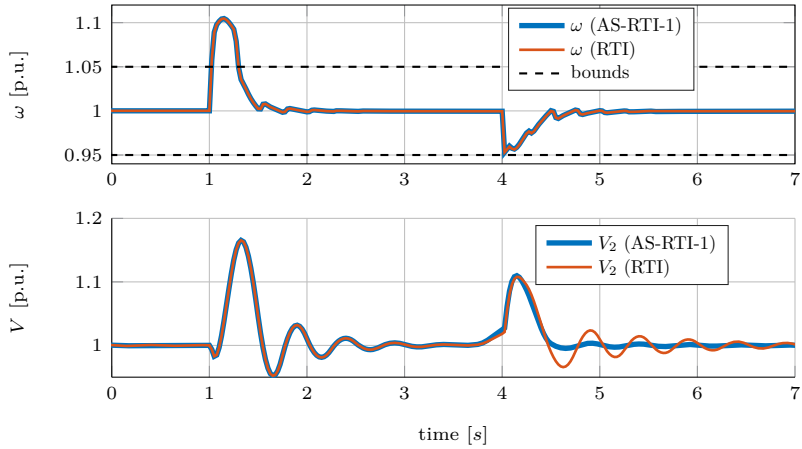


Figure 10.3: Frequency of the DG (top) and voltage at the load (bottom) for two different schemes with $T_s = 250$ ms: 1) AS-RTI with one inner iteration (blue), 2) RTI (red).

Simulation results

We implement the AS-RTI in `acados` through its MATLAB interface [278]. We use `HPIPM` [105], an interior-point-based QP solver for the SQP subproblems. In all experiments, we use a Gauss-Newton (GN) Hessian approximation. In the simulation, we compare the following different schemes: 1) AS-RTI with k inner GN-SQP iterations denoted as AS-RTI-k and, 2) the RTI. We observed that the solution does not improve with further inner iterations, even when solving the advanced problem to convergence. The simulation results for the two schemes with $T_s = 250$ ms are depicted in Fig. 10.3. Both schemes can stabilize the system and bring ω and V_2 to 1. Observe that the overshoot at $t = 1$ s remains the same for all schemes since the load drop is not predicted, and the NMPC controller can react only after noticing it, i.e., after the time T_s has passed. The load changes both unpredicted ($t = 1$s) and predicted ($t = 4$ s) for more than 90% compared to the initial load value. Compared to the AS-RTI, the voltage oscillations with the RTI last long after the predicted load change. The CPU times of all schemes in this experiment are provided in Table 10.2. We compare the schemes by the resulting running cost, defined as

$$J(T_\mathrm{sim}) \coloneqq \int_0^{T_\mathrm{sim}} L(x(t), z(t), u^*_\mathrm{NMPC}(t))\ \mathrm{d}t,$$

where $u^*_\mathrm{NMPC}(t)$ is the resulting NMPC closed-loop input fed back to the system over the simulation time T_sim. All considered schemes are real-time feasible,

Algorithm	N	J	Preparation phase			Feedback phase		
			max	min	mean	max	min	mean
RTI	50	8.04	16.01	11.00	12.19	4.00	1.96	2.50
AS-RTI-1	50	7.69	40.97	24.03	26.85	4.00	1.97	2.53
AS-RTI-2	50	7.69	58.99	37.00	41.42	3.02	2.00	2.61
AS-RTI-3	50	7.69	67.03	50.98	55.18	4.00	1.97	2.55
RTI	40	12.28	12.03	8.03	9.94	2.96	0.99	1.76
AS-RTI-1	40	11.76	23.04	19.00	20.30	2.04	0.97	1.70
AS-RTI-2	40	11.77	41.01	29.99	32.44	3.23	0.98	1.85
AS-RTI-3	40	11.77	54.02	38.99	43.20	2.03	0.93	1.68

Table 10.2: CPU times of different NMPC schemes in milliseconds.

however, with the AS-RTI with few additional computations we improve the running cost $J(\cdot)$ and the improvement is larger with a larger sampling time. With more inner iterations, the computational load in the preparation phase is increasing, however, in the feedback phase we still solve only a single QP and thus the feedback delay stays small. All simulations are run on an HP Z-book equipped with an Intel i7-6820HQ CPU with 2.70 GHz and 16 GB RAM under Windows 10. The computation times for the two experiments are reported in Table 10.2.

10.6 Conclusion

In this chapter, we have presented a new family of algorithms for real-time Nonlinear Model Predictive Control(NMPC). Thereby, the algoritmic ideas of the Real-Time Iteration [77], Multi-Level Iteration [42], and the Advanced Step Controller [290] are combined. We prove in Theorem 10.10 contraction of the new algorithms under standard assumptions. Our result holds for general algorithms with at least Q-linear convergence, including the setting where different algorithms for inner and outer iterations are used. Furthermore, we provided sufficient conditions for the boundedness of the numerical error of the AS-RTI. Numerical examples confirm our theoretical results, and it shows that with few and cheap additional iterations, we get significantly closer to ideal NMPC behavior.

In future work, it would be interesting to investigate theoretically and numerically the influence of suboptimality iterations, such as level B, which improve feasibility, but not optimality, on the overall performance of the AS-RTI scheme. Moreover, it would be useful to have an efficient implementation of the AS-RTI scheme within an open-source package such as `acados` [278].

Chapter 11

Conclusions and Future Research

The last chapter of this thesis summarizes the results and proposes several future research directions.

11.1 Summary and conclusions

This thesis has proposed several new algorithms and reformulations for numerically solving optimal control problems subject to nonsmooth dynamical systems. This resulted in a toolchain for solving several classes of nonsmooth optimal control problems in a unified way. Furthermore, a detailed theoretical analysis of the novel algorithms is provided. For a given accuracy, compared to standard direct methods, smoothing approaches, and mixed-integer formulations, we have achieved computational speed ups to several orders of magnitude. All algorithms developed in thesis are implemented in the open-source software package `nosnoc`. Moreover, this thesis introduced a family of new algorithms for real-time and accurate nonlinear model predictive control for smooth dynamical systems.

We summarize some of the main insights and new ideas introduced in this thesis. A detailed overview of the specific contributions is given in Section 1.2.

First, in Chapter 5, we highlighted some fundamental limitations of standard direct methods, including time-stepping discretizations, smoothing, mixed-

integer reformulations, and mathematical programs with complementarity constraints reformulation. All these approaches led to nonsmooth optimization problems, where the nonsmoothness is possibly smoothed explicitly or implicitly, before or after the time-discretization. If algorithms do not treat the discontinuities explicitly by detecting switches, they will all suffer from the same limitations, namely, they achieve only first-order accuracy and the discrete-time numerical sensitivities do not converge to the correct values. Consequently, the algorithms might converge to spurious solutions or make almost no progress from a given initial guess. These insights are inspired by [259], where it was shown that with a time-stepping discretization numerical sensitivities are wrong, no matter how small the step size is, and that smoothing works only if the step size is sufficiently smaller than the smoothing parameter. We have shown that in a smoothing homotopy approach, the methods can still make reasonable progress towards optimality, as the sensitivities are correct in the early iterations. Moreover, these methods still converge to feasible (even though not even locally optimal) solutions, which explains their occasional practical success. Nevertheless, to obtain an accurate solution with a standard approach, an enormous computational effort is needed. We concluded that it is necessary to develop new tailored algorithms that treat nonsmoothness more explicitly.

Second, this thesis developed high-accuracy discretization methods for Piecewise Smooth Systems (PSS). In Chapter 6, we investigated Stewart's and the Heaviside step reformulation to pass from a Filippov convexification of the PSS to an equivalent Dynamic Complementarity System (DCS). It turns out that these DCSs have favorable properties, e.g., the Lagrange multipliers in the DCS are continuous functions of time even across active set changes. The ODEs and DAEs obtained for a fixed active set have unique solutions under mild conditions. We exploited the properties of the DCS, which enabled us to derive the method of Finite Elements with Switch Detection (FESD) in Chapter 7. The FESD method is based on three main ideas. Starting with a standard Runge-Kutta (RK) discretization for the DCS, inspired by [28], one first lets the integration step sizes be degrees of freedom. Exploiting the continuity of the Lagrange multipliers, we introduced the *cross complementarity* conditions, which enabled implicit and exact switch detection. Finally, to remove spurious degrees of freedom when no switches occur, we introduced the *step equilibration* conditions. We have provided a detailed theoretical analysis of the FESD method and have shown that it is superior to time-stepping methods in terms of computation times and accuracy. FESD enabled us to generalize time-stepping methods and to overcome fundamental limitations of first-order accuracy and wrong numerical sensitivities.

Third, we introduced the time-freezing reformulation in Chapters 8 and 9. It enabled us to transform systems with state jumps into PSS. In particular,

we regarded complementarity Lagrangian systems with frictional impacts and hybrid automata with hysteresis. The main ideas of time-freezing are to define an auxiliary ODE in the prohibited regions of the state space of the initial system and a nonsmooth clock state. The trajectory endpoints of the auxiliary ODE satisfy the point-wise state jump law of the initial system. Moreover, the clock state's evolution is *frozen* during the runtime of the auxiliary ODE. By taking only the piece of the trajectory when the time was evolving, one can recover the solution of the original system with state jumps. We studied the theoretical properties of time-freezing systems and provided constructive ways to build auxiliary dynamics. Interestingly, it turned out that the dynamics of the initial system after a state jump often correspond to a sliding mode of the Filippov embedding of the time-freezing system. Time-freezing opened the possibility to apply the rich theoretical tools and numerical methods for Filippov systems to systems with state jumps.

Fourth, the reformulation of piecewise smooth systems into dynamic complementarity systems via Stewart's or the Heaviside step approach, the FESD method (and all its variations), fully automated time-freezing reformulations of systems with state jumps, and homotopy methods for solving MPCCs are all implemented in the open-source software package `nosnoc`. This makes the results of this thesis reproducible and accessible to other researchers.

Finally, in Chapter 10, we introduced several new real-time algorithms for nonlinear MPC. We extended the well-known Real-Time Iteration (RTI) [75], which computes only one Sequential Quadratic Programming (SQP) step per sampling time. Moreover, we added new variants (called levels) to the Multi-Level Iterations (MLI) [42], which essentially consist of different inexact SQP variations of the RTI. To improve the current linearization point in an RTI setting, we proposed using an MLI variant to iterate on an *advanced problem* with a predicted state while waiting for the next state estimate. This idea of solving an advanced problem is an essential ingredient of the Advanced Step Controller (ASC) [290]. When the new state estimate becomes available, we performed the final SQP step to generate a new control input. This algorithm is called the Advanced Step Real-Time Iteration (AS-RTI). The AS-RTI bridges the gap between two well-established algorithmic paradigms: the ASC that solves the OCP to convergence at every sampling time, and the RTI that performs only one Newton-type iteration. We studied the convergence properties and error bounds of the proposed methods.

In summary, this thesis highlighted some fundamental limitations of the standard methods, proposed new methods with a sound theory that overcomes these limitations, making it possible to treat several classes of nonsmooth optimal control problems in a unified way.

11.2 Future research directions

Next, we comment on some limitations of our toolchain's key parts and discuss future research directions.

Mathematical Programs with Complementarity Constraints (MPCCs). The main computational burden in solving a nonsmooth optimal control problem is solving the MPCCs. Currently, we solve a sequence of Nonlinear Programs (NLPs), parameterized by a homotopy parameter σ and obtained from relaxation, smoothing, elastic mode or ℓ_1 penalty reformulation of the initial NLP, cf. Section 2.4. We solve the NLPs for a fixed σ to convergence with `IPOPT` [281]. However, to solve the overall MPCC, it may not be needed to solve the NLPs in the early phases of the homotopy loop always to convergence. A reasonable approach is to update the barrier parameter τ in the interior-point method and σ simultaneously, as done in [227]. A similar approach was taken in the non-interior point method by Lin and Othsuka [183]. Moreover, we do not exploit the specific block diagonal sparsity structure in the discretized OCP. Using a tailored Riccati recursion [183] instead of an off-the-shelf linear solver as in `IPOPT` would likely speed up the computations. Next, a solver further tailored to MPCCs, with a specialized globalization strategy is also an interesting future research direction [180]. It would be also interesting to consider combinatorial MPCC methods that identify the active set of the complementarity conditions at every step [70, 71, 169, 180]. Note that due to the cross complementarity conditions, the active sets are the same over the whole finite element, which reduces the combinatorial complexity. Furthermore, one should exploit problem structure in an active set strategy. For example, the trajectory can only enter into neighboring regions, which already limits the possible number of active set changes. Similarly, in time-freezing systems, it is clear that once the system enters the auxiliary ODE mode, it must leave this region by construction.

Furthermore, we do not provide any specific initial guess. The homotopy starts with a rather large σ, and one could make computationally cheap simulations of systems smoothed with the same σ to get at least a dynamically feasible initial guess. We observed often in practice that even in very early phases of the homotopy the solver seems to find solutions with the correct switching sequence. It would be interesting to find a projection method to extract the active set of the complementarity conditions from these approximate solutions. Such an approach was studied [247]. After fixing the active set, the FESD MPCC reduces to a smooth discrete-time multi-stage OCP with variable stage lengths. Thus, after only a single further NLP one would obtain a very accurate solution.

We observed in practice that the homotopy loop usually converges to S-stationary

points of the MPCC, cf. Definition 2.29. From a theoretical point of view, it would be interesting to rigorously establish conditions on the discrete-time OCP that lead to convergence to S-stationary points. A starting point could be the analysis in [137], where it was proven that all stationary points of MPCCs originating from discrete-time OCPs with linear complementarity systems are S-stationary points.

Finite Elements with Switch Detection. The FESD method has superior theoretical properties over standard time-steeping problems. However, having the step size as degrees of freedom introduces another nonlinearity and nonconvexity to the problem. We observed in numerical experiments that the time-stepping discretization with a fixed h usually converges faster (however to spurious solutions and only with low accuracy). In the early phases of the homotopy when $h \ll \sigma$, time-stepping methods still have correct sensitivities. We could solve the time-stepping problems in the early homotopy iterations to generate a good initial guess and switch to the FESD problems in later iterations.

In this thesis, we have developed FESD for two different types of DCS reformulations. One of the crucial steps was to exploit the continuity of the Lagrange multipliers in the complementarity conditions and to formulate the cross complementarity conditions to make the switch detection possible. A natural question is: can FESD be extended to DCS formulations of projected dynamical systems and first-order sweeping processes if the variables entering the complementarity conditions have similar properties?

In complementarity Lagrangian systems, studied in Chapter 8, one has complementarity conditions between gap functions and normal contact forces, i.e., $0 \leq f_c(q) \perp \lambda_{\mathrm{n}} \geq 0$. The gap functions $f_c(q)$ are continuous function of time, which resembles the structure and continuity properties of the multipliers in the DCS obtained with Stewart's or the Heaviside step reformulation. Using similar ideas as in Chapter 7, we developed in [209] a FESD method for complementarity Lagrangian systems, which is called FESD-J. It would be interesting to extensively compare FESD-J to FESD applied to time-freezing systems.

Currently, we use either Stewart's or the Heaviside step reformulation for a given PSS. It would be interesting to explore alternative convex or linear programming formulations for computing θ, which may be even more efficient.

Another useful development would be a FESD tailored to smooth approximations of nonsmooth systems, which does not suffer from the usual fundamental limitations in terms of accuracy and convergence of sensitivities. We have

observed in Section 7.3.6, that with FESD the sensitivity converges to the correct value even if $h \gg \sigma$, but depending on the relaxation and smoothing strategy, the solution of the smoothed subproblems may not always be unique. Such a method would be useful for algorithms using smooth approximation of nonsmooth systems or in the context of stochastic optimal control, as discussed below.

Time-freezing. Time-freezing enables an exact reformulation of systems with state jumps into Filippov systems. This enabled us to treat this class of systems directly with FESD. Several questions should be addressed in the future. First, it would be interesting to derive a time-freezing reformulation that treats both elastic and inelastic impacts with friction in a unified way. Second, a natural extension to be considered are systems with multiple and simultaneous impacts, i.e., models with vector-valued gap functions. This is partially done, and the working version with promising results is implemented in `nosnoc`. We described some details in Section 8.7. It is left to perform a detailed theoretical analysis of the proposed approach. Third, it would be interesting to extend time-freezing to more general hybrid automata than the one from Chapter 9. The first step would be identifying usual guard functions, state jump laws, and infeasible regions in the state space for general hybrid automata [188, 245]. Ideally, one would end up with a set of rules that enable an automatic reformulation for every well-posed hybrid automata.

Stochastic nonsmooth optimal control. In control applications, the model parameters and initial states are never known exactly. To account for these uncertainties, in practice, robust and stochastic optimal control problem formulations are used. It would be interesting to derive robust and stochastic optimal control problem formulations for nonsmooth dynamical systems, and to develop tailored numerical methods for these problems, possibly by relying on some of the ideas used in this thesis. In a recent study by Messerer et al. [194], it was shown that the mean dynamics of a bimodal PSS resembles the smoothed dynamics of such a system. It would be interesting to extend this to general PSS and time-freezing systems. An interesting question is: how to interpret in a stochastic time-freezing system the scenario when, at a given point in numerical time, some trajectories are in the auxiliary dynamics mode and others are in the nominal mode? What can we conclude about the initial system?

Real-time NMPC. Real-time NMPC for smooth systems is already a very mature field. One drawback of the AS-RTI with full Newton steps is that if the inner iterations use an MLI level with a smaller contraction radius than the MLI

level in the outer iterations, then the inner iterations might diverge and increase the error. This can be overcome by using globalization strategies [163]. It would be interesting to investigate further algorithmic variations of the AS-RTI, e.g., where the focus is on having at least feasible outer iterations. From a practical point of view, it would be nice to have a proper open-source implementation of the AS-RTI, e.g., in `acados` [278].

Bibliography

[1] Society for industrial and applied mathematics style manual. https://www.siam.org/Portals/0/Books%20-For%20Authors/SIAM%20Style%20Manual.pdf?ver=2017-11-21-112335-700, 2017.

[2] NOSNOC. https://github.com/nurkanovic/nosnoc, 2022.

[3] Acary, V., Bonnefon, O., and Brogliato, B. *Nonsmooth modeling and simulation for switched circuits*, vol. 69. Springer Science & Business Media, 2010.

[4] Acary, V., and Brogliato, B. *Numerical methods for nonsmooth dynamical systems: applications in mechanics and electronics*. Springer Science & Business Media, 2008.

[5] Acary, V., and Brogliato, B. Implicit euler numerical scheme and chattering-free implementation of sliding mode systems. *Systems & Control Letters 59*, 5 (2010), 284–293.

[6] Acary, V., De Jong, H., and Brogliato, B. Numerical simulation of piecewise-linear models of gene regulatory networks using complementarity systems. *Physica D: Nonlinear Phenomena 269* (2014), 103–119.

[7] Albersmeyer, J. *Adjoint-based algorithms and numerical methods for sensitivity generation and optimization of large scale dynamic systems*. PhD thesis, University of Heidelberg, 2010.

[8] Albersmeyer, J., and Diehl, M. The lifted Newton method and its application in optimization. *SIAM Journal on Optimization 20*, 3 (2010), 1655–1684.

[9] Andersson, J. A. E., Gillis, J., Horn, G., Rawlings, J. B., and Diehl, M. CasADi – a software framework for nonlinear optimization and

optimal control. *Mathematical Programming Computation 11*, 1 (2019), 1–36.

[10] Andrés-Martínez, O., Biegler, L. T., and Flores-Tlacuahuac, A. An indirect approach for singular optimal control problems. *Computers & Chemical Engineering 139* (2020), 106923.

[11] Anitescu, M. On solving mathematical programs with complementarity constraints as nonlinear programs. *Preprint ANL/MCS-P864-1200, Argonne National Laboratory, Argonne, IL 3* (2000).

[12] Anitescu, M. On using the elastic mode in nonlinear programming approaches to mathematical programs with complementarity constraints. *SIAM Journal on Optimization 15*, 4 (2005), 1203–1236.

[13] Anitescu, M. Optimization-based simulation of nonsmooth rigid multibody dynamics. *Mathematical Programming 105* (2006), 113–143.

[14] Anitescu, M., and Hart, G. D. A constraint-stabilized time-stepping approach for rigid multibody dynamics with joints, contact and friction. *International Journal for Numerical Methods in Engineering 60*, 14 (2004), 2335–2371.

[15] Anitescu, M., and Potra, F. A. Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems. *Nonlinear Dynamics 14* (1997), 231–247.

[16] Anitescu, M., Tseng, P., and Wright, S. J. Elastic-mode algorithms for mathematical programs with equilibrium constraints: global convergence and stationarity properties. *Mathematical programming 110*, 2 (2007), 337–371.

[17] Anitescu, M., and Zavala, V. M. Mpc as a dvi: Implications on sampling rates and accuracy. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)* (2017), IEEE, pp. 1933–1938.

[18] Ascher, U., and Petzold, L. *Computer Methods for Ordinary Differential Equations and Differential–Algebraic Equations.* SIAM, Philadelphia, 1998.

[19] Aubin, J. P., and Cellina, A. *Differential Inclusions: Set-Valued Maps and Viability Theory.* Springer-Verlag, 1984.

[20] Avraam, M. *Modelling and optimisation of hybrid dynamic processes.* PhD thesis, Imperial College London (University of London), 2000.

[21] Ban, X. J., Pang, J.-S., Liu, H. X., and Ma, R. Continuous-time point-queue models in dynamic network loading. *Transportation Research Part B: Methodological 46*, 3 (2012), 360–380.

[22] Banjac, G., Stellato, B., Moehle, N., Goulart, P., Bemporad, A., and Boyd, S. Embedded code generation using the OSQP solver. In *Proceedings of the IEEE Conference on Decision and Control (CDC)* (2017).

[23] Bard, J. F., and Moore, J. T. A branch and bound algorithm for the bilevel programming problem. *SIAM Journal on Scientific and Statistical Computing 11*, 2 (1990), 281–292.

[24] Barton, P. *The modelling and simulation of combined discrete/continuous processes*. PhD thesis, Department of Chemical Engineering, Imperial College of Science, Technology and Medicine, London, 1992.

[25] Barton, P., and Lee, C. Modeling, Simulation, Sensitivity Analysis and Optimization of Hybrid Systems. *ACM Transactions on Modeling and Computer Simulation 12*, 4 (2002), 256–289.

[26] Barton, P., and Pantelides, C. Modelling of Combined Discrete/Continuous Processes. *AIChE Journal 40* (1994), 966–979.

[27] Bastien, J., and Schatzman, M. Numerical precision for differential inclusions with uniqueness. *ESAIM: Mathematical Modelling and Numerical Analysis 36*, 3 (2002), 427–460.

[28] Baumrucker, B. T., and Biegler, L. T. Mpec strategies for optimization of a class of hybrid dynamic systems. *Journal of Process Control 19*, 8 (2009), 1248–1256.

[29] Bellman, R. *Dynamic programming*. Princeton University Press, 1957.

[30] Belotti, P., Kirches, C., Leyffer, S., Linderoth, J., Luedtke, J., and Mahajan, A. Mixed-integer nonlinear optimization. *Acta Numerica 22* (2013), 1–131.

[31] Bemporad, A., and Morari, M. Control of systems integrating logic, dynamics, and constraints. *Automatica 35*, 3 (1999), 407–427.

[32] Bemporad, A., and Morari, M. Predictive Control of Constrained Hybrid Systems. In *Nonlinear Predictive Control* (Basel Boston Berlin, 2000), F. Allgöwer and A. Zheng, Eds., vol. 26 of *Progress in Systems Theory*, Birkhäuser, pp. 71–98.

[33] Benson, H., Sen, A., Shanno, D., and Vanderbei, R. Interior-Point Algorithms, Penalty Methods and Equilibrium Problems. *Computational Optimization and Applications 34* (2006), 155–182.

[34] Bernardo, M., Budd, C., Champneys, A. R., and Kowalczyk, P. *Piecewise-smooth dynamical systems: theory and applications*, vol. 163. Springer Science & Business Media, 2008.

[35] Bertsekas, D. *Dynamic Programming and Optimal Control*, 3rd ed., vol. 2. Athena Scientific, 2007.

[36] Betts, J. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, 2nd ed. SIAM, 2010.

[37] Biegler, L. T. Solution of dynamic optimization problems by successive quadratic programming and orthogonal collocation. *Computers and Chemical Engineering 8*, 3–4 (1984), 243–248.

[38] Biegler, L. T. *Nonlinear Programming*. MOS-SIAM Series on Optimization. SIAM, 2010.

[39] Bock, H. Numerical treatment of inverse problems in chemical reaction kinetics. In *Modelling of Chemical Reaction Systems*, K. Ebert, P. Deuflhard, and W. Jäger, Eds., vol. 18 of *Springer Series in Chemical Physics*. Springer, Heidelberg, 1981, pp. 102–125.

[40] Bock, H. *Randwertproblemmethoden zur Parameteridentifizierung in Systemen nichtlinearer Differentialgleichungen*, vol. 183 of *Bonner Mathematische Schriften*. Universität Bonn, Bonn, 1987.

[41] Bock, H. G. Recent advances in parameter identification techniques for ODE. In *Numerical Treatment of Inverse Problems in Differential and Integral Equations*. Birkhäuser, 1983, pp. 95–121.

[42] Bock, H. G., Diehl, M., Kostina, E. A., and Schlöder, J. P. Constrained optimal feedback control of systems governed by large differential algebraic equations. In *Real-Time and Online PDE-Constrained Optimization*. SIAM, 2007, pp. 3–22.

[43] Bock, H. G., Kirches, C., Meyer, A., and Potschka, A. Numerical solution of optimal control problems with explicit and implicit switches. *Optimization Methods and Software 33*, 3 (2018), 450–474.

[44] Bock, H. G., and Plitt, K. J. A multiple shooting algorithm for direct solution of optimal control problems. In *Proceedings of the IFAC World Congress* (1984), Pergamon Press, pp. 242–247.

[45] Bonami, P., Biegler, L. T., Conn, A. R., Cornuéjols, G., Grossmann, I. E., Laird, C. D., Lee, J., Lodi, A., Margot, F., Sawaya, N., et al. An algorithmic framework for convex mixed integer nonlinear programs. *Discrete optimization 5*, 2 (2008), 186–204.

[46] Branicky, M. S. Multiple lyapunov functions and other analysis tools for switched and hybrid systems. *IEEE Transactions on automatic control 43*, 4 (1998), 475–482.

[47] Brenan, K., Campbell, S., and Petzold, L. *Numerical solution of initial-value problems in differential-algebraic equations*. SIAM, Philadelphia, 1996. Classics in Applied Mathematics 14.

[48] Brogliato, B. Some perspectives on the analysis and control of complementarity systems. *IEEE Transactions on Automatic Control 48*, 6 (2003), 918–935.

[49] Brogliato, B. *Nonsmooth Mechanics: Models, Dynamics and Control*. Springer, 2016.

[50] Brogliato, B., Daniilidis, A., Lemarechal, C., and Acary, V. On the equivalence between complementarity systems, projected systems and differential inclusions. *Systems & Control Letters 55*, 1 (2006), 45–51.

[51] Brogliato, B., and Tanwani, A. Dynamical systems coupled with monotone set-valued operators: Formalisms, applications, well-posedness, and stability. *Siam Review 62*, 1 (2020), 3–129.

[52] Brogliato, B., and Thibault, L. Existence and uniqueness of solutions for non-autonomous complementarity dynamical systems. *Journal of Convex Analysis 17*, 3&4 (2010), 961–990.

[53] Bürger, A. *Nonlinear mixed-integer model predictive control of renewable energy systems*. PhD thesis, University of Freiburg, 2020.

[54] Bürger, A., Bull, D., Sawant, P., Bohlayer, M., Klotz, A., Beschütz, D., Altmann-Dieses, A., Braun, M., and Diehl, M. Experimental operation of a solar-driven climate system with thermal energy storages using mixed-integer nonlinear model predictive control. *Optimal Control Applications and Methods* (2021), 1–27.

[55] Butcher, J. On the implementation of implicit Runge-Kutta methods. *BIT Numerical Mathematics 16*, 3 (1976), 237–240.

[56] Butcher, J. C. *Numerical Methods for Ordinary Differential Equations*. John Wiley & Sons, Ltd, 2003.

[57] Byrd, R. H., Lopez-Calva, G., and Nocedal, J. A line search exact penalty method using steering rules. *Mathematical Programming 133*, 1 (2012), 39–73.

[58] Byrd, R. H., Nocedal, J., and Waltz, R. A. KNITRO: An integrated package for nonlinear optimization. In *Large Scale Nonlinear Optimization* (2006), G. Pillo and M. Roma, Eds., Springer Verlag, pp. 35–59.

[59] Çamlibel, M. K., Heemels, W., and Schumacher, J. On linear passive complementarity systems. *European Journal of Control 8*, 3 (2002), 220–237.

[60] Camlibel, M. K., and Schumacher, J. On the zeno behavior of linear complementarity systems. In *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No. 01CH37228)* (2001), vol. 1, IEEE, pp. 346–351.

[61] Cao, Y., Li, S., and Petzold, L. Adjoint sensitivity analysis for differential-algebraic equations: algorithms and software. *Journal of Computational and Applied Mathematics 149* (2002), 171–191.

[62] Caracotsios, M., and Stewart, W. Sensitivity analysis of initial value problems with mixed ODEs and algebraic equations. *Computers and Chemical Engineering. 9* (1985), 359–365.

[63] Carius, J., Ranftl, R., Koltun, V., and Hutter, M. Trajectory optimization with implicit hard contacts. *IEEE Robotics and Automation Letters 3*, 4 (2018), 3316–3323.

[64] Caspari, A., Lüken, L., Schäfer, P., Vaupel, Y., Mhamdi, A., Biegler, L. T., and Mitsos, A. Dynamic optimization with complementarity constraints: Smoothing for direct shooting. *Computers & Chemical Engineering 139* (2020), 106891.

[65] Chen, W., Ren, Y., Zhang, G., and Biegler, L. T. A simultaneous approach for singular optimal control based on partial moving grid. *AIChE Journal 65*, 6 (2019), e16584.

[66] Chen, W., Wang, K., Shao, Z., and Biegler, L. T. Chapter 11: Direct transcription with moving finite elements. In *Control and Optimization with Differential-Algebraic Constraints*. SIAM, 2012, pp. 233–252.

[67] Cojocaru, M.-G. Dynamic equilibria of group vaccination strategies in a heterogeneous population. *Journal of Global Optimization 40*, 1 (2008), 51–63.

[68] Cortes, J. Discontinuous dynamical systems. *IEEE Control systems magazine 28*, 3 (2008), 36–73.

[69] Cottle, R. W., Pang, J.-S., and Stone, R. E. *The linear complementarity problem*. SIAM, 2009.

[70] De Marchi, A. Implicit augmented lagrangian and generalized optimization. *arXiv preprint arXiv:2302.00363* (2023).

[71] De Marchi, A., and Themelis, A. Proximal gradient algorithms under local lipschitz gradient continuity: A convergence and robustness analysis of panoc. *Journal of Optimization Theory and Applications 194*, 3 (2022), 771–794.

[72] DeMiguel, V., Friedlander, M. P., Nogales, F. J., and Scholtes, S. A two-sided relaxation scheme for mathematical programs with equilibrium constraints. *SIAM Journal on Optimization 16*, 2 (2005), 587–609.

[73] Dieci, L., and Lopez, L. Sliding motion in filippov differential systems: theoretical results and a computational approach. *SIAM Journal on Numerical Analysis 47*, 3 (2009), 2023–2051.

[74] Dieci, L., and Lopez, L. Sliding motion on discontinuity surfaces of high co-dimension. a construction for selecting a filippov vector field. *Numerische Mathematik 117*, 4 (2011), 779–811.

[75] Diehl, M. *Real-Time Optimization for Large Scale Nonlinear Processes*. PhD thesis, University of Heidelberg, 2001.

[76] Diehl, M., Bock, H. G., and Schlöder, J. P. A real-time iteration scheme for nonlinear optimization in optimal feedback control. *SIAM Journal on Control and Optimization 43*, 5 (2005), 1714–1736.

[77] Diehl, M., Bock, H. G., Schlöder, J. P., Findeisen, R., Nagy, Z., and Allgöwer, F. Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. *Journal of Process Control 12*, 4 (2002), 577–585.

[78] Diehl, M., Ferreau, H. J., and Haverbeke, N. Efficient numerical methods for nonlinear MPC and moving horizon estimation. In *Nonlinear model predictive control*, L. Magni, M. Raimondo, and F. Allgöwer, Eds., vol. 384 of *Lecture Notes in Control and Information Sciences*. Springer, 2009, pp. 391–417.

[79] Diehl, M., Findeisen, R., Allgöwer, F., Bock, H. G., and Schlöder, J. P. Nominal stability of the real-time iteration scheme for nonlinear model predictive control. *IEE Proc.-Control Theory Appl. 152*, 3 (2005), 296–308.

[80] Diehl, M., and Gros, S. *Numerical Optimal Control*. –, expected to be published in 2018.

[81] Diehl, M., Leineweber, D., and Schäfer, A. MUSCOD-II Users' Manual. IWR-Preprint 2001-25, University of Heidelberg, 2001.

[82] Domahidi, A. *Methods and Tools for Embedded Optimization and Control*. PhD thesis, ETH Zürich, 2013.

[83] Dontchev, A., and Lempio, F. Difference methods for differential inclusions: A survey. *SIAM review 34*, 2 (1992), 263–294.

[84] Dontchev, A. L., and Rockafellar, R. T. *Implicit Functions and Solution Mappings*. Springer, 2009.

[85] Dontchev, A. L., and Rockafellar, R. T. *Implicit Functions and Solution Mappings: A View from Variational Analysis*. Springer, 2014.

[86] Doshi, N., Hogan, F. R., and Rodriguez, A. Hybrid differential dynamic programming for planar manipulation primitives. In *2020 IEEE International Conference on Robotics and Automation (ICRA)* (2020), IEEE, pp. 6759–6765.

[87] Edmond, J. F., and Thibault, L. Relaxation of an optimal control problem involving a perturbed sweeping process. *Mathematical programming 104*, 2 (2005), 347–373.

[88] Facchinei, F., Jiang, H., and Qi, L. A smoothing method for mathematical programs with equilibrium constraints. *Mathematical Programming 85* (1999), 107–134.

[89] Facchinei, F., and Pang, J.-S. *Finite-dimensional variational inequalities and complementarity problems*, vol. 1-2. Springer-Verlag, 2003.

[90] Ferreau, H. J., Kirches, C., Potschka, A., Bock, H. G., and Diehl, M. qpOASES: a parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation 6*, 4 (2014), 327–363.

[91] Ferris, M., and Tin-Loi, F. On the solution of a minimum weight elastoplastic problem involving displacement and complementarity constraints. *Computer Methods in Applied Mechanics and Engineering 174*, 1-2 (1999), 108–120.

[92] Filippov, A. On certain questions in the theory of optimal control. *Journal of the Society for Industrial and Applied Mathematics, Series A: Control 1*, 1 (1962), 76–84.

[93] Filippov, A. Differential Equations with discontinuous right hand side. *AMS Transl. 42* (1964), 199–231.

[94] Filippov, A. *Differential Equations with Discontinuous Righthand Sides: Control Systems*, vol. 18. Springer Science & Business Media, 1988.

[95] Flegel, M., and Kanzow, C. A Fritz John approach to first order optimality conditions for mathematical programs with equilibrium constraints. *Optimization 52* (2003), 277–286.

[96] Flegel, M. L., and Kanzow, C. Abadie-type constraint qualification for mathematical programs with equilibrium constraints. *Journal of Optimization Theory and Applications 124*, 3 (2005), 595–614.

[97] Flegel, M. L., and Kanzow, C. On m-stationary points for mathematical programs with equilibrium constraints. *Journal of Mathematical Analysis and Applications 310*, 1 (2005), 286–302.

[98] Flegel, M. L., and Kanzow, C. On the guignard constraint qualification for mathematical programs with equilibrium constraints. *Optimization 54*, 6 (2005), 517–534.

[99] Fletcher, R. A New Low Rank Quasi-Newton Update Scheme for Nonlinear Programming. Tech. Rep. NA/223, University of Dundee, 2005.

[100] Fletcher, R. *Practical methods of optimization*. John Wiley & Sons, 2013.

[101] Fletcher, R., and Leyffer, S. Nonlinear programming without a penalty function. *Mathematical Programming 91* (2002), 239–269.

[102] Fletcher, R., and Leyffer, S. Numerical experience with solving mpecs as nlps. In *Department of Mathematics and Computer Science, University of Dundee, Dundee* (2002), Citeseer.

[103] Fletcher*, R., and Leyffer, S. Solving mathematical programs with complementarity constraints as nonlinear programs. *Optimization Methods and Software 19*, 1 (2004), 15–40.

[104] Fletcher, R., Leyffer, S., Ralph, D., and Scholtes, S. Local Convergence of SQP Methods for Mathematical Programs with Equilibrium Constraints. *SIAM Journal on Optimization 17* (2006), 259–286.

[105] Frison, G., Sartor, T., Zanelli, A., and Diehl, M. The BLAS API of BLASFEO: Optimizing performance for small matrices. *ACM Transactions on Mathematical Software (TOMS) 46*, 2 (2020), 15:1–15:36.

[106] Fukushima, M., and Lin, G.-H. Smoothing methods for mathematical programs with equilibrium constraints. In *International Conference on Informatics Research for Development of Knowledge Society Infrastructure, 2004. ICKS 2004.* (2004), IEEE, pp. 206–213.

[107] Fukushima, M., Luo, Z.-Q., and Pang, J.-S. A globally convergent sequential quadratic programming algorithm for mathematical programs with linear complementarity constraints. *Computational optimization and applications 10*, 1 (1998), 5–34.

[108] Fukushima, M., and Tseng, P. 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 (2002), 724–739.

[109] Gehring, C. Operational space control of single legged hopping. Master's thesis, Eidgenössische Technische Hochschule Zürich, Autonomous Systems Lab, 2011.

[110] Giallombardo, G., and Ralph, D. Multiplier convergence in trust-region methods with application to convergence of decomposition methods for mpecs. *Mathematical Programming 112*, 2 (2008), 335–369.

[111] Gill, P., Murray, W., and Saunders, M. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Review 47*, 1 (2005), 99–131.

[112] Gill, P. E., and Wong, E. Methods for convex and general quadratic programming. *Mathematical Programming Computation 7*, 1 (2015), 71–112.

[113] Goebel, R., Sanfelice, R., and Teel, A. Hybrid Dynamical Systems. *IEEE Control Systems Magazines 29*, 2 (April 2009), 28–93.

[114] Grammel, G. Maximum principle for a hybrid system via singular perturbations. *SIAM journal on control and optimization 37*, 4 (1999), 1162–1175.

[115] Griewank, A., and Walther, A. *Evaluating Derivatives*, 2 ed. SIAM, 2008.

[116] Gros, S., Zanon, M., Quirynen, R., Bemporad, A., and Diehl, M. From linear to nonlinear MPC: bridging the gap via the real-time iteration. *International Journal of Control* (2016).

[117] Grossmann, I. Review of Nonlinear Mixed-Integer and Disjunctive Programming Techniques. *Optimization and Engineering 3* (2002), 227–252.

[118] Guddat, J., Vasquez, F. G., and Jongen, H. *Parametric Optimization: Singularities, Pathfollowing and Jumps*. Teubner, Stuttgart, 1990.

[119] Guglielmi, N., and Hairer, E. An efficient algorithm for solving piecewise-smooth dynamical systems. *Numerical Algorithms 89*, 3 (2022), 1311–1334.

[120] Guo, L., and Deng, Z. A new augmented lagrangian method for mpccs—theoretical and numerical comparison with existing augmented lagrangian methods. *Mathematics of Operations Research 47*, 2 (2022), 1229–1246.

[121] Guo, L., and Ye, J. J. Necessary optimality conditions for optimal control problems with equilibrium constraints. *SIAM Journal on Control and Optimization 54*, 5 (2016), 2710–2733.

[122] Hairer, E., Lubich, C., and Roche, M. *The numerical solution of differential-algebraic systems by Runge-Kutta methods*. No. 1409 in Lecture Notes in Mathematics. Springer, Heidelberg, 1989.

[123] Hairer, E., Lubich, C., and Wanner, G. *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*. Springer, 2006.

[124] Hairer, E., Nørsett, S., and Wanner, G. *Solving Ordinary Differential Equations I*, 2nd ed. Springer Series in Computational Mathematics. Springer, Berlin, 1993.

[125] Hairer, E., Nørsett, S., and Wanner, G. *Solving Ordinary Differential Equations II – Stiff and Differential-Algebraic Problems*, 2nd ed. Springer Series in Computational Mathematics. Springer, Berlin, 1996.

[126] Hairer, E., and Wanner, G. *Solving Ordinary Differential Equations II – Stiff and Differential-Algebraic Problems*, 2nd ed. Springer, Berlin Heidelberg, 1991.

[127] Hall, J., Nurkanović, A., Messerer, F., and Diehl, M. A sequential convex programming approach to solving quadratic programs and optimal control problems with linear complementarity constraints. *IEEE Control Systems Letters 6* (2022), 536–541.

[128] Hall, J., Nurkanović, A., Messerer, F., and Diehl, M. Lcqpow– a solver for linear complementarity quadratic programs. *Mathematical Programming Computation (accepted for publication)* (2023).

[129] Halm, M., and Posa, M. Set-valued rigid body dynamics for simultaneous frictional impact. *arXiv preprint arXiv:2103.15714* (2021).

[130] Harzer, J., Schutter, J. D., and Diehl, M. Efficient numerical optimal control for highly oscillatory systems. *arXiv preprint* (2022).

[131] Hatz, K., Leyffer, S., Schlöder, J. P., and Bock, H. G. Regularizing bilevel nonlinear programs by lifting. *Argonne National Laboratory, USA* (2013).

[132] Hauswirth, A., Bolognani, S., Hug, G., and Dörfler, F. Optimization algorithms as robust feedback controllers. *arXiv preprint arXiv:2103.11329* (2021).

[133] Heemels, W., and Brogliato, B. The complementarity class of hybrid dynamical systems. *European Journal of Control 9*, 2-3 (2003), 322–360.

[134] Heemels, W., Schumacher, J. M., and Weiland, S. Linear complementarity systems. *SIAM journal on applied mathematics 60*, 4 (2000), 1234–1269.

[135] Heemels, W., Schumacher, J. M., and Weiland, S. Projected dynamical systems in a complementarity formalism. *Operations Research Letters 27*, 2 (2000), 83–91.

[136] Hempel, A. B., Goulart, P. J., and Lygeros, J. Every continuous piecewise affine function can be obtained by solving a parametric linear program. In *2013 European Control Conference (ECC)* (2013), IEEE, pp. 2657–2662.

[137] Hempel, A. B., Goulart, P. J., and Lygeros, J. Strong stationarity conditions for optimal control of hybrid systems. *IEEE Transactions on Automatic Control 62*, 9 (2017), 4512–4526.

[138] Hoang, N. D., and Mordukhovich, B. S. Extended euler–lagrange and hamiltonian conditions in optimal control of sweeping processes with controlled moving sets. *Journal of Optimization Theory and Applications 180* (2019), 256–289.

[139] Hoheisel, T., Kanzow, C., and Schwartz, A. Theoretical and numerical comparison of relaxation methods for mathematical programs with complementarity constraints. *Mathematical Programming 137*, 1 (2013), 257–288.

[140] Houska, B., Ferreau, H. J., and Diehl, M. ACADO toolkit – an open source framework for automatic control and dynamic optimization. *Optimal Control Applications and Methods 32*, 3 (2011), 298–312.

[141] Howell, T. A., Cleac'h, S. L., Kolter, J. Z., Schwager, M., and Manchester, Z. Dojo: A differentiable simulator for robotics. *arXiv preprint arXiv:2203.00806* (2022).

[142] Howell, T. A., Le Cleac'h, S., Singh, S., Florence, P., Manchester, Z., and Sindhwani, V. Trajectory optimization with optimization-based dynamics. *IEEE Robotics and Automation Letters 7*, 3 (2022), 6750–6757.

[143] HSL. A collection of Fortran codes for large scale scientific computation., 2011.

[144] Hu, W., Long, J., Zang, Y., Han, J., et al. Solving optimal control of rigid-body dynamics with collisions using the hybrid minimum principle. *arXiv preprint arXiv:2205.08622* (2022).

[145] Hu, X., and Ralph, D. Convergence of a penalty method for mathematical programming with complementarity constraints. *Journal of Optimization Theory and Applications 123*, 2 (2004), 365–390.

[146] Izmailov, A. F. Mathematical programs with complementarity constraints: regularity, optimality conditions, and sensitivity. *Computational Mathematics and Mathematical Physics 44*, 7 (2004), 1145–1164.

[147] Izmailov, A. F., Pogosyan, A., and Solodov, M. V. Semismooth newton method for the lifted reformulation of mathematical programs with complementarity constraints. *Computational Optimization and Applications 51*, 1 (2012), 199–221.

[148] Izmailov, A. F., and Solodov, M. V. Stabilized sqp revisited. *Mathematical programming 133* (2012), 93–120.

[149] Izmailov, A. F., and Solodov, M. V. *Newton-Type Methods for Optimization and Variational Problems*, 1 ed. Springer, 2014.

[150] Izmailov, A. F., Solodov, M. V., and Uskov, E. Global convergence of augmented lagrangian methods applied to optimization problems with degenerate constraints, including problems with complementarity constraints. *SIAM Journal on Optimization 22*, 4 (2012), 1579–1606.

[151] Jane, J. Y. Necessary and sufficient optimality conditions for mathematical programs with equilibrium constraints. *Journal of Mathematical Analysis and Applications 307*, 1 (2005), 350–369.

[152] Jäschke, J., Yang, X., and Biegler, L. T. Fast economic model predictive control based on nlp-sensitivities. *Journal of Process Control 24*, 8 (2014), 1260–1272.

[153] Jean, M. The non-smooth contact dynamics method. *Computer methods in applied mechanics and engineering 177*, 3-4 (1999), 235–257.

[154] Jean, M., and Moreau, J. J. Unilaterality and dry friction in the dynamics of rigid body collections. In *1st Contact Mechanics International Symposium* (1992), pp. 31–48.

[155] Jeffrey, M. R., Jeffrey, M. R., and Chernyk. *Hidden dynamics*. Springer, 2018.

[156] Johnson, A. M., Burden, S. A., and Koditschek, D. E. A hybrid systems model for simple manipulation and self-manipulation systems. *The International Journal of Robotics Research 35*, 11 (2016), 1354–1392.

[157] Kadrani, A., Dussault, J.-P., and Benchakroun, A. A new regularization scheme for mathematical programs with complementarity constraints. *SIAM Journal on Optimization 20*, 1 (2009), 78–103.

[158] Kanzow, C., and Schwartz, A. A new regularization method for mathematical programs with complementarity constraints with strong convergence properties. *SIAM Journal on Optimization 23*, 2 (2013), 770–798.

[159] Kanzow, C., and Schwartz, A. The price of inexactness: convergence properties of relaxation methods for mathematical programs with complementarity constraints revisited. *Mathematics of Operations Research 40*, 2 (2015), 253–275.

[160] Karush, W. Minima of Functions of Several Variables with Inequalities as Side Conditions. Master's thesis, Department of Mathematics, University of Chicago, 1939.

[161] Kastner-Maresch, A. Implicit runge-kutta methods for differential inclusions. *Numerical functional analysis and optimization 11*, 9-10 (1990), 937–958.

[162] Katayama, S., Doi, M., and Ohtsuka, T. A moving switching sequence approach for nonlinear model predictive control of switched systems with state-dependent switches and state jumps. *International Journal of Robust and Nonlinear Control 30*, 2 (2020), 719–740.

[163] Kiessling, D., Zanelli, A., Nurkanović, A., Gillis, J., Diehl, M., Zeilinger, M., Pipeleers, G., and Swevers, J. A feasible sequential linear programming algorithm with application to time-optimal path planning problems. In *2022 IEEE 61st Conference on Decision and Control (CDC)* (2022), IEEE, pp. 1196–1203.

[164] Kim, J., Cho, H., Shamsuarov, A., Shim, H., and Seo, J. H. State estimation strategy without jump detection for hybrid systems using gluing function. In *53rd IEEE International Conference on Decision and Control (CDC)* (2014), IEEE, pp. 139–144.

[165] Kim, Y., Leyffer, S., and Munson, T. Mpec methods for bilevel optimization problems. In *Bilevel Optimization*. Springer, 2020, pp. 335–360.

[166] Kirches, C. A Numerical Method for Nonlinear Robust Optimal Control with Implicit Discontinuities and an Application to Powertrain Oscillations. Diploma thesis, University of Heidelberg, October 2006.

[167] Kirches, C. *Fast Numerical Methods for Mixed-Integer Nonlinear Model-Predictive Control*. PhD thesis, University of Heidelberg, 2010.

[168] Kirches, C., Kostina, E., Meyer, A., and Schlöder, M. Numerical solution of optimal control problems with switches, switching costs and jumps. *Optimization Online 6888* (2018).

[169] Kirches, C., Larson, J., Leyffer, S., and Manns, P. Sequential linearization method for bound-constrained mathematical programs with complementarity constraints. *SIAM Journal on Optimization 32*, 1 (2022), 75–99.

[170] Kouzoupis, D., Frison, G., Zanelli, A., and Diehl, M. Recent advances in quadratic programming algorithms for nonlinear model predictive control. *Vietnam Journal of Mathematics 46*, 4 (2018), 863–882.

[171] Krasovskii, N. N. *Stability of motion: applications of Lyapunov's second method to differential systems and equations with delay*. Stanford university press Stanford, 1963.

[172] Kuhn, H. W., and Tucker, A. W. Nonlinear programming. In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability* (Berkeley, 1951), J. Neyman, Ed., University of California Press.

[173] Kundur, P. S., and Malik, O. P. *Power system stability and control*. McGraw-Hill Education, 2022.

[174] Leine, R. I., and Van de Wouw, N. *Stability and convergence of mechanical systems with unilateral constraints*, vol. 36. Springer Science & Business Media, 2007.

[175] Lenders, F. J. M. *Numerical methods for mixed-integer optimal control with combinatorial constraints*. PhD thesis, Heidelberg University, 2018.

[176] Letourneau, M., and Sharp, J. W. *AMS Style Guide - Journals*, 2017.

[177] Leyffer, S. Macmpec: Ampl collection of mpecs,". Argonne National Laboratory, 2000.

[178] Leyffer, S. Complementarity constraints as nonlinear equations: Theory and numerical experience. In *Optimization with Multivalued Mappings*. Springer, 2006, pp. 169–208.

[179] Leyffer, S., López-Calva, G., and Nocedal, J. Interior methods for mathematical programs with complementarity constraints. *SIAM Journal on Optimization 17*, 1 (2006), 52–77.

[180] Leyffer, S., and Munson, T. S. A globally convergent filter method for mpecs. *Preprint ANL/MCS-P1457-0907, Argonne National Laboratory, Mathematics and Computer Science Division* (2007).

[181] Liao-McPherson, D., Nicotra, M., and Kolmanovsky, I. Time distributed sequential quadratic programming for model predictive control: Stability and robustness. *arXiv preprint arXiv:1903.02605* (2019).

[182] Lin, G.-H., and Fukushima, M. A modified relaxation scheme for mathematical programs with complementarity constraints. *Annals of Operations Research 133*, 1 (2005), 63–84.

[183] Lin, K., and Ohtsuka, T. A non-interior-point continuation method for the optimal control problem with equilibrium constraints. *arXiv preprint arXiv:2210.10336* (2022).

[184] Lions, P. *Generalized Solutions of Hamilton-Jacobi Equations*. Pittman, 1982.

[185] Liu, X., and Sun, J. Generalized stationary points and an interior-point method for mathematical programs with equilibrium constraints. *Mathematical Programming 101*, 1 (2004), 231–261.

[186] Lunze, J., and Lamnabhi-Lagarrigue, F. *Handbook of hybrid systems control: theory, tools, applications*. Cambridge University Press, 2009.

[187] Luo, Z., Pang, J., and Ralph, D. *Mathematical Programs with Equilibrium Constraints*. Cambridge University Press, Cambridge, 1996.

[188] Lygeros, J., Johansson, K. H., Simic, S. N., Zhang, J., and Sastry, S. S. Dynamical properties of hybrid automata. *IEEE Transactions on automatic control 48*, 1 (2003), 2–17.

[189] Lygeros, J., Tomlin, C., and Sastry, S. Controllers for reachibility specifications for hybrid systems. *Automatica 35* (1999), pp. 349–370.

[190] Machina, A., and Ponosov, A. Filippov solutions in the analysis of piecewise linear models describing gene regulatory networks. *Nonlinear Analysis: Theory, Methods & Applications 74*, 3 (2011), 882–900.

[191] Manchester, Z., Doshi, N., Wood, R. J., and Kuindersma, S. Contact-implicit trajectory optimization using variational integrators. *The International Journal of Robotics Research 38*, 12-13 (2019), 1463–1476.

[192] Matsaglia, G., and PH Styan, G. Equalities and inequalities for ranks of matrices. *Linear and multilinear Algebra 2*, 3 (1974), 269–292.

[193] McAllister, R. D., and Rawlings, J. B. Advances in mixed-integer model predictive control. In *2022 American Control Conference (ACC)* (2022), IEEE, pp. 364–369.

[194] Messerer, F., Baumgärtner, K., Nurkanović, A., and Diehl, M. Approximate propagation of normal distributions for stochastic optimal control of nonsmooth systems. *arXiv preprint arXiv:2308.03431* (2023).

[195] Michelsen, M. L. Semi-implicit Runge-Kutta methods for stiff systems: program description and application examples. *Inst. f. Kemiteknik, Danmarks tekniske Højskole, Lynby* (1976).

[196] Moreau, J. J. Rafle par un convexe variable (premiere partie). *Lecture notes, séminaire d'analyse convexe 15* (1971).

[197] Moreau, J. J. Evolution problem associated with a moving convex set in a hilbert space. *Journal of differential equations 26*, 3 (1977), 347–374.

[198] Moreau, J. J. Unilateral contact and dry friction in finite freedom dynamics. In *Nonsmooth mechanics and Applications*. Springer, 1988, pp. 1–82.

[199] Moreau, J. J. Numerical aspects of the sweeping process. *Computer methods in applied mechanics and engineering 177*, 3-4 (1999), 329–349.

[200] Nguyen, N. S., and Brogliato, B. Comparisons of multiple-impact laws for multibody systems: Moreau's law, binary impacts, and the LZB approach. In *Advanced Topics in Nonsmooth Dynamics*. Springer, 2018, pp. 1–45.

[201] Nocedal, J., and Wright, S. J. *Numerical Optimization*, 2 ed. Springer Series in Operations Research and Financial Engineering. Springer, 2006.

[202] Nurkanović, A., Albrecht, S., Brogliato, B., and Diehl, M. The time-freezing reformulation for numerical optimal control of complementarity lagrangian systems with state jumps. *Automatica 158* (2023), 111295.

[203] Nurkanović, A., Albrecht, S., and Diehl, M. Limits of MPCC Formulations in Direct Optimal Control with Nonsmooth Differential Equations. In *2020 European Control Conference (ECC)* (2020), pp. 2015–2020.

[204] Nurkanović, A., Albrecht, S., and Diehl, M. *Multi-level Iterations for Economic Nonlinear Model Predictive Control*. Springer International Publishing, Cham, 2021, pp. 65–105.

[205] Nurkanović, A., and Diehl, M. Continuous optimization for control of hybrid systems with hysteresis via time-freezing. *IEEE Control Systems Letters* (2022).

[206] Nurkanović, A., and Diehl, M. NOSNOC: A software package for numerical optimal control of nonsmooth systems. *IEEE Control Systems Letters* (2022).

[207] Nurkanović, A., Frey, J., Pozharskiy, A., and Diehl, M. Finite elements with switch detection for numerical optimal control of nonsmooth dynamical systems with set-valued heaviside step functions. *arXiv preprint arXiv:2307.03482 - under review in Nonlinear analysis: hybrid systems* (2023).

[208] Nurkanović, A., Frey, J., Pozharskiy, A., and Diehl, M. Finite elements with switched detection for direct optimal control of nonsmooth systems with set-valued step functions. *Proceedings of the IEEE Conference on Decision and Control (CDC) 2023* (2023).

[209] Nurkanović, A., Frey, J., Pozharskiy, A., and Diehl, M. Fesd-j: Finite elements with switch detection for numerical optimal control of rigid bodies with impacts and coulomb friction. *Nonlinear Analysis: Hybrid Systems 52* (2024), 101460.

[210] Nurkanović, A., Mešanović, A., Sperl, M., Albrecht, S., Münz, U., Findeisen, R., and Diehl, M. Optimization-based primary and secondary control of microgrids. *at - Automatisierungstechnik 68*, 12 (2020), 1044–1058.

[211] Nurkanović, A., Mešanović, A., Zanelli, A., Frey, J., Frison, G., Albrecht, S., and Diehl, M. Real-time nonlinear model predictive control for microgrid operation. In *Proceedings of the American Control Conference (ACC)* (Denver, USA, July 2020). (accepted).

[212] Nurkanović, A., Sartor, T., Albrecht, S., and Diehl, M. A Time-Freezing Approach for Numerical Optimal Control of Nonsmooth Differential Equations with State Jumps. *IEEE Control Systems Letters 5*, 2 (2021), 439–444.

[213] Nurkanović, A., Sperl, M., Albrecht, S., and Diehl, M. Finite Elements with Switch Detection for Direct Optimal Control of Nonsmooth Systems. *Submitted to Numerische Mathematik - under review* (2022).

[214] Nurkanović, A., Zanelli, A., Albrecht, S., and Diehl, M. The Advanced Step Real Time Iteration for NMPC. In *Proceedings of the IEEE Conference on Decision and Control (CDC)* (Nice, France, December 2019).

[215] Nurkanović, A., Zanelli, A., Frison, G., Albrecht, S., and Diehl, M. Contraction properties of the advanced step real-time iteration. In *Proceedings of the IFAC World Congress* (2020), vol. 51. (accepted).

[216] Ohtsuka, T. A continuation/GMRES method for fast computation of nonlinear receding horizon control. *Automatica 40*, 4 (2004), 563–574.

[217] Outrata, J. Optimality Conditions for a Class of Mathematical Programs with Equilibrium Constraints. *Math. Oper. Res. 24*, 3 (1999), 627–644.

[218] Outrata, J., Kocvara, M., and Zowe, J. *Nonsmooth approach to optimization problems with equilibrium constraints: theory, applications and numerical results*, vol. 28. Springer Science & Business Media, 2013.

[219] Pang, J.-S., and Fukushima, M. Complementarity constraint qualifications and simplified b-stationarity conditions for mathematical programs with equilibrium constraints. *Computational Optimization and Applications 13*, 1 (1999), 111–136.

[220] Pang, J.-S., and Stewart, D. E. Differential variational inequalities. *Mathematical programming 113*, 2 (2008), 345–424.

[221] Paoli, L., and Schatzman, M. A numerical scheme for impact problems i: The one-dimensional case. *SIAM Journal on Numerical Analysis 40*, 2 (2002), 702–733.

[222] Paoli, L., and Schatzman, M. A numerical scheme for impact problems ii: The multidimensional case. *SIAM journal on numerical analysis 40*, 2 (2002), 734–768.

[223] Patterson, M. A., Hager, W. W., and Rao, A. V. A ph mesh refinement method for optimal control. *Optimal Control Applications and Methods 36*, 4 (2015), 398–421.

[224] Pontryagin, L. S., Boltyanski, V. G., Gamkrelidze, R. V., and Miscenko, E. F. *The Mathematical Theory of Optimal Processes*. Wiley, Chichester, 1962.

[225] Posa, M., Cantu, C., and Tedrake, R. A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research 33*, 1 (2014), 69–81.

[226] Quirynen, R. *Numerical Simulation Methods for Embedded Optimization*. PhD thesis, KU Leuven and University of Freiburg, 2017.

[227] Raghunathan, A. U., and Biegler, L. T. An interior point method for mathematical programs with complementarity constraints (mpccs). *SIAM Journal on Optimization 15*, 3 (2005), 720–750.

[228] Ralph, D., and Wright, S. J. Some properties of regularization and penalization schemes for mpecs. *Optimization Methods and Software 19*, 5 (2004), 527–556.

[229] Rawlings, J. B., Mayne, D. Q., and Diehl, M. M. *Model Predictive Control: Theory, Computation, and Design*, 2nd ed. Nob Hill, 2017.

[230] Reich, S. On an existence and uniqueness theory for nonlinear differential-algebraic equations. *Circuits Syst. Signal Process. 10*, 3 (May 1991), 343–359.

[231] Robinson, S. M. Strongly Regular Generalized Equations. *Mathematics of Operations Research, Vol. 5, No. 1 (Feb., 1980), pp. 43-62 5* (1980), 43–62.

[232] Rockafellar, R. Fundamentals of optimization. *Lecture Notes 2007* (2006).

[233] Rockafellar, R., and Wets, R. *Variational Analysis*, vol. 317 of *Grundlehren der mathematischen Wissenschaften*. Springer, 2004.

[234] Rockafellar, R., and Wets, R. J.-B. *Variational Analysis*. Springer-Verlag, 1997.

[235] Sager, S. *Numerical methods for mixed–integer optimal control problems*. Der andere Verlag, Tönning, Lübeck, Marburg, 2005.

[236] Scheel, H., and Scholtes, S. Mathematical programs with complementarity constraints: Stationarity, optimality, and sensitivity. *Math. Oper. Res. 25* (2000), 1–22.

[237] Scholtes, S. Convergence properties of a regularization scheme for mathematical programs with complementarity constraints. *SIAM Journal on Optimization 11*, 4 (2001), 918–936.

[238] Scholz, R. *Model-Based Optimal Feedback Control For Microgrids*. PhD thesis, University of Heidelberg, 2022.

[239] Scholz, R., Nurkanović, A., Mešanović, A., Gutekunst, J., Potscka, A., Bock, H. G., and Kostina, E. Model-based Optimal Feedback Control for Microgrids with Multi-Level Iterations. *Operations Reserach 2019* (2019).

[240] Schultz, G., and Mombaur, K. Modeling and optimal control of human-like running. *IEEE/ASME Transactions on mechatronics 15*, 5 (2009), 783–792.

[241] Schwartz, A. *Mathematical programs with complementarity constraints: Theory, methods and applications*. PhD thesis, Universität Würzburg, 2011.

[242] Shaikh, M. S., and Caines, P. E. On the hybrid optimal control problem: theory and algorithms. *IEEE Transactions on Automatic Control 52*, 9 (2007), 1587–1603.

[243] Shen, J., and Pang, J.-S. Linear complementarity systems: Zeno states. *SIAM Journal on Control and Optimization 44*, 3 (2005), 1040–1066.

[244] Shen, J., and Pang, J.-S. Semicopositive linear complementarity systems. *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal 17*, 15 (2007), 1367–1386.

[245] Simic, S. N., Johansson, K. H., Lygeros, J., and Sastry, S. Towards a geometric theory of hybrid systems. *Dynamics of Continuous, Discrete and Impulsive Systems Series B: Applications and Algorithms 12*, 5-6 (2005), 649–687.

[246] Smirnov, G. V. *Introduction to the Theory of Differential Inclusions*, vol. 41. American Mathematical Soc., 2002.

[247] Sperl, M. Numerical methods for optimal control problems involving differential complementarity systems. Master's thesis, University of Passau, 2021.

[248] Steffensen, S., and Ulbrich, M. A new relaxation scheme for mathematical programs with equilibrium constraints. *SIAM Journal on Optimization 20*, 5 (2010), 2504–2539.

[249] Stein, O. Lifting mathematical programs with complementarity constraints. *Mathematical programming 131*, 1 (2012), 71–94.

[250] Stewart, D. A high accuracy method for solving odes with discontinuous right-hand side. *Numerische Mathematik 58*, 1 (1990), 299–328.

[251] Stewart, D. E. *High accuracy numerical methods for ordinary differential equations with discontinuous right-hand side.* PhD thesis, University of Queensland, St. Lucia, Queensland 4072, Australia, 1990.

[252] Stewart, D. E. A numerical method for friction problems with multiple contacts. *The ANZIAM Journal 37*, 3 (1996), 288–308.

[253] Stewart, D. E. Existence of solutions to rigid body dynamics and the painlevé paradoxes. *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics 325*, 6 (1997), 689–693.

[254] Stewart, D. E. Convergence of a time-stepping scheme for rigid-body dynamics and resolution of painlevé's problem. *Archive for Rational Mechanics and Analysis 145*, 3 (1998), 215–260.

[255] Stewart, D. E. Rigid-body dynamics with friction and impact. *SIAM review 42*, 1 (2000), 3–39.

[256] Stewart, D. E. Uniqueness for index-one differential variational inequalities. *Nonlinear Analysis: Hybrid Systems 2*, 3 (2008), 812–818.

[257] Stewart, D. E. Uniqueness for solutions of differential complementarity problems. *Mathematical programming 118*, 2 (2009), 327–345.

[258] Stewart, D. E. *Dynamics with Inequalities: impacts and hard constraints.* SIAM, 2011.

[259] Stewart, D. E., and Anitescu, M. Optimal control of systems with discontinuous differential equations. *Numerische Mathematik 114*, 4 (2010), 653–695.

[260] Stewart, D. E., and Trinkle, J. C. An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction. *International Journal for Numerical Methods in Engineering 39*, 15 (1996), 2673–2691.

[261] Studer, C. W. *Augmented time-stepping integration of non-smooth dynamical systems*. PhD thesis, ETH Zurich, 2008.

[262] Sussmann, H. J. A maximum principle for hybrid optimal control problems. In *Proceedings of the 38th IEEE conference on decision and control (Cat. No. 99CH36304)* (1999), vol. 1, IEEE, pp. 425–430.

[263] Tassa, Y., Erez, T., and Todorov, E. Synthesis and stabilization of complex behaviors through online trajectory optimization. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems* (2012), IEEE, pp. 4906–4913.

[264] Taubert, K. Converging multistep methods for initial value problems involving multivalued maps. *Computing 27*, 2 (1981), 123–136.

[265] Tedrake, R., and the Drake Development Team. Drake: Model-based design and verification for robotics, 2019.

[266] Thierry, D. *Nonlinear Optimization based frameworks for Model Predictive Control, State-Estimation, Sensitivity Analysis, and Ill-posed Problems*. PhD thesis, Carnegie Mellon University, 2019.

[267] Thierry, D., and Biegler, L. The ℓ_1—exact penalty-barrier phase for degenerate nonlinear programming problems in ipopt. *IFAC-PapersOnLine 53*, 2 (2020), 6496–6501.

[268] Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems* (2012), IEEE, pp. 5026–5033.

[269] Tran-Dinh, Q., Savorgnan, C., and Diehl, M. Adjoint-based predictor-corrector sequential convex programming for parametric nonlinear optimization. *SIAM J. Optimization 22*, 4 (2012), 1258–1284.

[270] Ulbrich, M. Mathematical modeling with variational inequalities.

[271] Valadier, M. Lipschitz approximation of the sweeping (or moreau) process. *Journal of Differential Equations 88*, 2 (1990), 248–264.

[272] Van Der Schaft, A. J., and Schumacher, H. *An introduction to hybrid dynamical systems*, vol. 251. springer, 2007.

[273] van der Schaft, A. J., and Schumacher, J. M. The complementary-slackness class of hybrid systems. *Mathematics of control, signals and systems 9*, 3 (1996), 266–301.

[274] Van Roy, W., Nurkanović, A., Abbasi-Esfeden, R., Frey, J., Pozharskiy, A., Swevers, J., and Diehl, M. Continuous optimization for control of finite-state machines with cascaded hysteresis via time-freezing. *IEEE Conference on Decision and Control* (2023).

[275] Vandenberghe, L., De Moor, B., and Vandewalle, J. The generalized linear complementarity problem applied to the complete analysis of resistive piecewise-linear circuits. *IEEE transactions on circuits and systems 36*, 11 (1989), 1382–1391.

[276] Vanderbei, R. J. Loqo: An interior point code for quadratic programming. *Optimization methods and software 11*, 1-4 (1999), 451–484.

[277] Veelken, S. *A new relaxation scheme for mathematical programs with equilibrium constraints: Theory and numerical experience.* PhD thesis, Technische Universität München, 2009.

[278] Verschueren, R., Frison, G., Kouzoupis, D., Frey, J., van Duijkeren, N., Zanelli, A., Novoselnik, B., Albin, T., Quirynen, R., and Diehl, M. acados – a modular open-source framework for fast embedded optimal control. *Mathematical Programming Computation* (Oct 2021).

[279] Vieira, A., Brogliato, B., and Prieur, C. Quadratic optimal control of linear complementarity systems: First-order necessary conditions and numerical analysis. *IEEE Transactions on Automatic Control 65*, 6 (2019), 2743–2750.

[280] Visintin, A. *Differential models of hysteresis*, vol. Appl. Math. Sci. 111. Springer-Verlag, Berlin, 1994.

[281] Wächter, A., and Biegler, L. T. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming 106*, 1 (2006), 25–57.

[282] Walther, A. Automatic differentiation of explicit Runge-Kutta methods for optimal control. *Computational Optimization and Applications 36*, 1 (2006), 83–108.

[283] Wensing, P. M., Posa, M., Hu, Y., Escande, A., Mansard, N., and Del Prete, A. Optimization-based control for dynamic legged robots. *arXiv preprint arXiv:2211.11644* (2022).

[284] Werling, K., Omens, D., Lee, J., Exarchos, I., and Liu, C. K. Fast and feature-complete differentiable physics for articulated rigid bodies with contact. *arXiv preprint arXiv:2103.16021* (2021).

[285] Westenbroek, T., Xiong, X., Ames, A. D., and Sastry, S. S. Optimal control of piecewise-smooth control systems via singular perturbations. In *2019 IEEE 58th Conference on Decision and Control (CDC)* (2019), IEEE, pp. 3046–3053.

[286] Wirsching, L. *Multi-Level Iteration Schemes with Adaptive Level Choice for Nonlinear Model Predictive Control*. PhD thesis, Ruprecht-Karls-Universität Heidelberg, 2018.

[287] Zanelli, A., Tran-Dinh, Q., and Diehl, M. Contraction estimates for abstract real-time algorithms for NMPC. In *Proceedings of the IEEE Conference on Decision and Control* (Nice, France, Dec 2019).

[288] Zanelli, A., Tran-Dinh, Q., and Diehl, M. A lyapunov function for the combined system-optimizer dynamics in inexact model predictive control. *Automatica 134* (2021), 109901.

[289] Zavala, V., and Anitescu, M. Real-Time Nonlinear Optimization as a Generalized Equation. *SIAM J. Control Optim. 48*, 8 (2010), 5444–5467.

[290] Zavala, V. M., and Biegler, L. T. The advanced step NMPC controller: Optimality, stability and robustness. *Automatica 45* (2009), 86–93.

[291] Zhang, J., Johansson, K. H., Lygeros, J., and Sastry, S. Zeno hybrid systems. *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal 11*, 5 (2001), 435–451.

[292] Zhong, Y. D., Han, J., and Brikis, G. O. Differentiable physics simulations with contacts: Do they have correct gradients wrt position, velocity and control? In *ICML 2022 2nd AI for Science Workshop* (2022).

[293] Zhu, F., and Antsaklis, P. J. Optimal control of hybrid switched systems: A brief survey. *Discrete Event Dynamic Systems 25* (2015), 345–364.

[294] Zhuravlev, V. F. Equations of motion of mechanical systems with ideal one-sided constraints. *Prikladnaia Matematika i Mekhanika 42* (1978), 781–788.

Curriculum Vitae

Armin Nurkanović was born in 1992 in Zagreb, Croatia. In 2015, he graduated from the University of Tuzla in Bosnia and Herzegovina with a Bachelor's degree in Electrical Engineering. He was awarded with the Golden Plaque of the University of Tuzla for his outstanding performance during his studies (GPA 10.0/10.0). He completed his Master's degree in Electrical Engineering and Information Technology at the Technical University of Munich in 2018. For his master studies he received a DAAD scholarship for master studies for all academic disciplines. Since 2018, Armin is a PhD student at the University of Freiburg (first external-industrial, now internal) under the supervision of Prof. Moritz Diehl. Until October 2021 he was an industrial PhD student at Siemens Technology in the research group Autonomous Systems and Control, under the supervision of Dr. Sebastian Albrecht. Together with his co-authors, he received the IEEE Control Systems Letters Outstanding Paper Award for 2022. In September 2023, he was the main organizer of the summer school on Direct Methods for Optimal Control of Nonsmooth Systems, which covered the topics developed in this thesis. The school was attended by about 70 participants, both from industry and academia, from eight different European countries.

FACULTY OF ENGINEERING
DEPARTMENT OF MICROSYSTEMS ENGINEERING
SYSTEMS CONTROL AND OPTIMIZATION LABORATORY
Georges-Köhler-Allee 102
79110 Freiburg in Br.