Advances in Industrial Control

Thomas J. Böhme
Benjamin Frank

Hybrid Systems, Optimal Control and Hybrid Vehicles

Theory, Methods and Applications

AIC
Springer

Advances in Industrial Control

Series editors

Michael J. Grimble, Glasgow, UK
Michael A. Johnson, Kidlington, UK

More information about this series at http://www.springer.com/series/1412

Thomas J. Böhme · Benjamin Frank

Hybrid Systems, Optimal Control and Hybrid Vehicles

Theory, Methods and Applications

Springer

Thomas J. Böhme
IAV GmbH Ingenieurgesellschaft Auto und Verkehr
Gifhorn
Germany

Benjamin Frank
IAV GmbH Ingenieurgesellschaft Auto und Verkehr
Gifhorn
Germany

ISSN 1430-9491 ISSN 2193-1577 (electronic)
Advances in Industrial Control
ISBN 978-3-319-51315-7 ISBN 978-3-319-51317-1 (eBook)
DOI 10.1007/978-3-319-51317-1

Library of Congress Control Number: 2016960557

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Series Editors' Foreword

The series *Advances in Industrial Control* aims to report and encourage technology transfer in control engineering. The rapid development of control technology has an impact on all areas of the control discipline. New theory, new controllers, actuators, sensors, new industrial processes, computer methods, new applications, new design philosophies…, new challenges. Much of this development work resides in industrial reports, feasibility study papers and the reports of advanced collaborative projects. The series offers an opportunity for researchers to present an extended exposition of such new work in all aspects of industrial control for wider and rapid dissemination.

Over the last two decades, there has been resurgent interest from the automotive industries in using advanced control methods. New technological developments, increased competitiveness, enhanced comfort and safety issues for drivers and passengers, better fuel usage and lower emission of pollutants have all played a role in reassessing how advanced control can help companies in the development of their latest models. This interest has spawned frequent special sessions at international control conferences and from these and the published literature, automotive control monographs have appeared in the *Advances in Industrial Control* series. The most recent of these include:

- *Dry Clutch Control for Automotive Applications* by Pietro J. Dolcini, Carlos Canudas de Wit and Hubert Béchart (ISBN 978-1-84996-067-0, 2010)
- *Active Braking Control Systems Design for Vehicles* by Sergio M. Savaresi and Mara Tanelli (ISBN 978-1-84996-349-7, 2010)
- *Nonlinear Control of Vehicles and Robots* by Béla Lantos and Lőrinc Márton (ISBN 978-1-84996-121-9, 2011)
- *Optimal Control of Hybrid Vehicles* by Bram de Jager, Thijs van Keulen and John Kessels (ISBN 978-1-4471-5075-6, 2015)
- *Robust Control Design for Active Driver Assistance Systems* by Péter Gáspár, Zoltán Szabó, József Bokor and Balázs Németh (ISBN 978-3-319-46124-3, 2017)

The series editors are pleased to continue this sequence of monographs with *Hybrid Systems, Optimal Control and Hybrid Vehicles: Theory, Methods and Applications* by authors Thomas J. Böhme and Benjamin Frank from the IAV GmbH (Department of Automotive and Traffic Engineering), Gifhorn, Germany. The concept of hybrid systems really began to fascinate control theorists from the 1990s when publications and papers started to appear using the term "hybrid system" and a related term "switched systems". To avoid confusion between the terms "hybrid system" and "hybrid vehicle", the authors state quite clearly 'whenever a system has continuous control inputs but at the same time can make discrete decisions or switch between different subsystems the system can be modelled as a 'hybrid system'". The authors then go on to say, "it should be pointed out, that the term 'hybrid' in 'hybrid vehicle' does not necessarily refer to the existence of discrete phenomena but to the fact that at least two energy storages and converters exist" in the vehicle.

The monograph has two strong themes:

a. Hybrid systems and the solution of optimal control of hybrid systems. These topics occupy the eight chapters of Parts I–III. The ultimate outcome of these chapters in the monograph is a valuable chapter of implementable algorithms for solving the optimal hybrid control problems.
b. Hybrid vehicles and the application of the optimal control of hybrid systems to several application problems arising in hybrid vehicle technology. These topics occupy the four chapters of Parts IV and V, and cover modeling hybrid vehicles along with three hybrid vehicle applications.

These two themes are presented and treated in a very comprehensive and thorough manner. The authors have provided careful algorithm descriptions to assist the user to recreate the numerical routine should they wish to. Each chapter is provided with an interesting Bibliographical Notes section and a very generous reference list. The monograph also has a short appendix on the graph theory used with sparse matrices.

The contents of the monograph are an excellent mix of theory, implementation practice, and applications. In the context of the *Advances in Industrial Control* monograph series it complements and adds to the set of volumes on control for automotive vehicles. It is a self-contained presentation of hybrid systems and hybrid vehicles and consequently its readership should be quite wide-ranging from mathematicians and control theorists to industrial automotive engineers and is an excellent entry to the series.

Michael J. Grimble
Michael A. Johnson
Industrial Control Centre
University of Strathclyde
Glasgow, Scotland, UK

Preface

The hybridization of the European car fleet should be an attempt for a positive contribution to meeting regional CO_2 and emission (CO, NOx, particles, and total hydrocarbons) targets, respectively. This has led to enormous research and development efforts in academia and industry to make the necessary technology mature for series production in large numbers.

The scope of the book is dedicated to the optimization of passenger hybrid vehicles. Recent results in the area of optimal control and hybrid vehicle design (powertrain architecture and size of the components) are presented.

The author's intention is to provide a complete overview to the field of optimal control of hybrid vehicles by studying one of the key elements that have a significant impact on the performance: energy management. The book is written from a mathematical viewpoint but takes care of the mixed audience. Much effort has been put into balancing the level of the presentation of topics of control, optimization, and automotive technology. Theoretical results in the field of applied optimal control are stated and commented to provide the reader with more insight but the proofs—with some exceptions—are omitted.

The prerequisites for this book are as follows: the reader should be familiar with dynamic systems in general and their representation in the state space in particular, as covered in standard undergraduate control courses. Especially, the reader should have already gained some experience in modeling and simulating mechanical and electrical systems. Furthermore, the reader should have a fairly sound understanding of differential calculus and some rudimentary understanding of functional analysis, optimization, and optimal control theory.

Intended Readership

The book has been written for master students, researchers, and practitioners in the fields of control engineering, automotive technology, and applied mathematics that are interested in techniques that provide the minimum energy consumption under

further restrictions by taking advantage of the control freedoms provided by hybridization. It is intendedly written from a practical point of view to be attractive to:

- students from various disciplines who envisage a career in control in the automotive industry. They find a transfer of theory to applications;
- applied mathematicians to find some nonstandard algorithms for solving large-scale optimal control problems;
- engineers being involved in specifying, developing, or calibrating energy management systems to get an introduction into a mathematical field of optimization and control which is not easily accessible and some hints how to model the system appropriately;
- researchers who are interested to see how energy management problems are specified and solved in industry; and
- managers or decision makers to get an inspiration of the potential of mathematical tools in this field.

A part of the topics has been taught at the University of Rostock and at the Ruhr University of Bochum and numerous final year students have been mentored under our responsibility during the past 5 years. The comments received from the students have been beneficial in the selection and preparation of the book's topics.

What are the Contributions of This Book

Energy management problems in practice can be large which means simply that the number of controls, states, and time horizon is large. This enforces hard conditions that have to be satisfied by good optimization candidates. This book proposes to solve such problems as hybrid optimal control problems.

Many problems encountered in practical hybrid vehicle applications seem on the first view not easily accessible for mathematical optimization theory, mainly arising from the difficulty that discrete decisions appear in the problem formulations, which causes considerable difficulties to many optimizers. Reconsidering of the underlying systems as hybrid systems and the control problem as hybrid optimal control problems can simplify the way to find a solution. This book supports this methodology and gives a selection of real-world problems, which are tackled as optimal control problems of hybrid systems.

We decided not to rely on the use of a specific third-party nonlinear programming solver but to induce modifications to well-known SQP algorithms in order to improve convergence and numerical stability. We list some well-proven algorithms in detail to give the readers a deeper insight into relevant implementation aspects, which are indispensable for the assessment of third-party nonlinear programming software packages or for writing one's own software code. A major contribution is the presentation of a sparse SQP framework based on sparse quasi-Newton updates to solve discretized optimal control problems for many controls, states, and discretization points. A new, very efficient, and robust sparse SQP algorithm will be

presented that decomposes the Hessian update mechanism into many subproblems of small dimension but with less numerical deficiencies.

To the authors' best knowledge, there is almost no book in the market which covers a complete spectrum of relevant stages to obtain optimal energy management (in a mathematical sense) including a discussion of theoretical aspects, a comprehensive treatment of algorithm implementation, and diverse application scenarios. The first obstacle for practitioners is the fact that many information and algorithms are widely scattered between various disciplines, which generates an initial barrier to enter this field. Many important algorithms, e.g., from the mathematical field of graph theory, are not easily accessible either to engineers or to applied mathematicians. It is therefore a time-consuming and demanding task to develop efficient algorithms for large hybrid vehicle problems. Our intention is, however, not to give blueprints for all possible problems (this is by far not possible) but to encourage the reader to use the provided information in this book including cited literature, proposed algorithms, etc., as basic kit for solving their own problems.

What is Not Covered in This Book

The book deals exclusively with time-invariant process descriptions which means that the process parameters remain constant over the complete time. Process types with time-varying parameters, e.g., battery aging, are not covered in this book. However, the proposed algorithms can serve as an initial tool set to be adapted to this problem class.

Structure of the Book

The book is modular structured and organized into six parts:

- Part I—Theory and Formulations;
- Part II—Methods for Optimal Control;
- Part III—Numerical Implementations;
- Part IV—Modeling of Hybrid Vehicles for Control;
- Part V—Applications; and
- Part VI—Appendix.

Part I of the book can be skipped if they wish to continue directly with the description of methods for obtaining numerical solutions of optimal control problems.

Chapter 1 discusses the challenges of designing and calibrating hybrid vehicles nowadays and motivates the usage of optimal control theory. It gives a general problem statement as an orientation for the following chapters and discusses the

most important control strategy of hybrid vehicles—energy management—and their algorithmic challenges.

Part I—*Theory and Formulations*. In Chap. 2, the theory of nonlinear programming is reviewed. The widely used sequential quadratic programming is presented for the solution of constrained nonlinear minimization problems, which is fundamental in our optimization framework to the solution of optimal control problems. A compact treatment of sensitivity analysis is presented as a tool for studying parameter changes of a system.

In Chap. 3, a general definition for hybrid and switched systems is introduced. Some important formulations for hybrid optimal control problems of dynamic processes described by systems of ordinary differential equations are discussed. The focus is on switched systems, a subclass of hybrid systems that switch between subsystems only in response to a command. This subclass already covers a great range of technical problems.

Chapter 4 discusses the Pontryagin's minimum principle. This important result is briefly approached from the classical calculus of variation. The Hamilton–Jacobi–Bellman method is discussed as an alternative approach to gain first-order necessary conditions for optimality. It is shown that both approaches correspond to each other under restrictive assumptions. The original Pontryagin's minimum principle for continuous optimal control problems is not suitable for hybrid optimal control problems. However, a quite natural reformulation of the hybrid optimal control problem admits the classical theory for deduction of first-order necessary conditions in the sense of Pontryagin. The charm of this methodology is its comprehensible derivation.

Part II—*Methods for Optimal Control*. This part starts the important topic of discretizations, since all numerical procedures rely on numerical integration schemes. In Chap. 5, the famous Runge–Kutta discretizations is presented. The determination of the Runge–Kutta order is briefly discussed and order conditions up to the fourth order are given including the additional conditions for solving optimal control problems. Regarding optimal control problems only explicit and implicit Runge–Kutta discretizations which satisfy additional conditions for the adjoint differential equations are discussed.

In Chap. 6, the Hamilton–Jacobi–Bellman principle is used to introduce the dynamic programming algorithm. Dynamic programming is an appealing approach for the solution of optimal control problems in many situations. The theoretical foundation is relatively easy to understand compared with the much more involved indirect methods. The general algorithm can be stated in a simple form and is easy to apply to continuous optimal control problems and with some minor reformulations, it is also well suited for switched optimal control problems.

In Chap. 7, indirect methods to solve optimal control problems are discussed. Indirect methods rely on first-order necessary conditions, summarized in Pontryagin's minimum principle, and attempt to generate control and state trajectories, which satisfy these conditions. An extension of the indirect shooting method for switched and hybrid systems that yields a solution for systems of low complexity is presented as well.

In Chap. 8, the algorithmic development for optimal control problems of switched systems is considered on the aspect of *First Discretize, then Optimize*. These methods are commonly known as direct methods. Direct methods transform the original problem via a discretization of the control and the state functions on a time grid to a nonlinear constrained optimization problem. This procedure is known as direct transcription of an optimal control problem and refers to the method of approximating the infinite-dimensional problem by a finite-dimensional one and to solve it with nonlinear programming algorithms.

Part III—*Numerical Implementations*. Optimal control problems formulated with direct transcription methods lead to large-scale nonlinear programming problems. One suitable framework for the solution of this type of optimization problems is sequential quadratic programming. But for an efficient implementation of the SQP algorithm it is crucial to take into account the particular properties and structure of the objective and constraint functions. This leads to *Karush–Kuhn–Tucker* (KKT) matrices, which occur in the subproblems to be sparse. Chapter 9 deals with techniques for the determination of the structure of the involved matrices, the calculation of the numerical derivatives, and the implementation of a sparse Quasi-Newton update.

Part IV—*Modeling of Hybrid Vehicles for Control*. In Chap. 10, the main hybrid vehicle configurations are presented including all relevant mechatronic subsystems. Models are derived with respect of optimization which imposes additional restriction in terms of complexity and smoothness. Several powertrain models of different depth for parallel and power-split hybrid vehicle configurations are formulated as hybrid systems. The easiest model includes representations of the electrical and the mechanical subsystem, whereas the most complex model also incorporates a detailed thermodynamic model of the internal combustion engine and the exhaust system as well as an emissions model.

Part V—*Applications*. In Chap. 11, the calibration process for hybrid vehicles is treated, which can be a cumbersome task if no systematic calibration approach is applied. Therefore, the fuel optimal operation of hybrid vehicles is formulated as switched optimal control problems and solved using dynamic programming, indirect shooting, and direct solution methods. Control parameters are derived for the calibration process as well as for the development of new functional approaches for improving the vehicle's performance. The step of transferring the solution of optimal control problems into calibration parameters for the electronic control unit is non-trivial. It is shown that look-up tables for rule-based energy managements can be derived directly from the solution that reduces significantly the time required for a high-quality calibration.

In Chap. 12, the theory of optimal control also suggests new functional approaches. Predictive energy management for minimizing wheel-to-meters energy losses are new functional candidates. Three different predictive control strategies are discussed for battery electric vehicles, full hybrid vehicles, and plug-in hybrid vehicles, which make the solution of a (switched) optimal control problem amenable for real-time implementation in the electronic control unit. This is only possible, if a profile of the driving route is a priori known. A prediction based on data

from modern navigation systems is made to obtain an estimation of this profile. It is demonstrated that predictive control strategies for energy management can significantly achieve fuel saving in real-world test drives.

Engineers aiming to find efficient hybrid powertrain configurations can benefit greatly from the seamless interaction of multi-objective optimization and optimal control methods. In Chap. 13, the simultaneous optimization of design parameters and energy management for a fixed parallel hybrid powertrain structure is discussed.

Gifhorn, Germany
September 2016

Thomas J. Böhme
Benjamin Frank

Acknowledgements

We wish to thank Profs. Dr. Bernhard Lampe and Dr. Torsten Jeinsch (from the University of Rostock), Dr. Kemal Yildiztekin (from the Helmut Schmidt University of Hamburg) and Dr. Leo Dostal (from the Technical University of Hamburg-Harburg) for their interest, advice, and very valuable comments, which cleared some mistakes and really improved the quality of this book. We also thank the many students, who were involved in this research project over the last years. We are also grateful to Matthias Schultalbers (division manager of mechatronics gasoline engines), who has made this publication possible in the first place. Last but not least, big thanks to your families for their encouragement and support over the last years.

Contents

Abbreviations and Symbols

Abbreviations

ADAS	Advanced driver assistance systems
AER	All-electric-range
ARTEMIS	Assessment and reliability of transport emission models and inventory systems
AS	Active-set
ASM	Asynchronous machine
BB	Branch-and-bound
BEV	Battery electric vehicles
BSFC	Brake specific fuel consumption
BFGS	Broyden, Fletcher, Goldfarb, and Shanno
BSOCP	Binary switched optimal control problem
BVP	Boundary value problem
BX	Branch-and-X
CD	Charge-depleting
CPS	Compound power-split
CS	Charge-sustaining
CVT	Continuously variable transmission
DFP	Davidon, Fletcher, and Powell
DOH	Degree of hybridization
DOF	Degree of freedom
DM	Direct methods
DP	Dynamic programming
EA	Evolutionary algorithm
ECMS	Equivalent consumption minimization strategy
ECU	Electronic control unit
ECVT	Electrical continuously variable transmission
EFS	Externally forced switching
EM	Electric machine
EOCP	Embedded optimal control problem

FTP	Federal test procedure
GHG	Greenhouse gas
GIS	Geographic information system
HEV	Hybrid electric vehicle
HIS	Homogeneous-split injection scheme
HJB	Hamilton–Jacobi–Bellman
HMP	Hybrid minimum principle
HOCP	Hybrid optimal control problem
IC	Internal combustion
ICE	Internal combustion engine
IDM	Intelligent driver model
IFS	Internally forced switching
IM	Indirect methods
IP	Interior-point
IPMSM	Interior permanent magnet synchronous machines
IPS	Input power-split
IPS2	Input power-split second generation
IVP	Initial value problem
KKT	Karush–Kuhn–Tucker
LICQ	Linear independence constraint qualification
LP	Linear programming
LUT	Look-up table
MFCQ	Mangasarian–Fromowitz constraint qualification
MG	Motor/Generator
MINLP	Mixed-integer nonlinear programming
MIOCP	Mixed-integer optimal control problem
MOEA	Multi-objective evolutionary algorithm
MPBVP	Multi-point boundary value problem
MPC	Model predictive control
MPCC	Mathematical program with complementary constraints
MVEG	Motor vehicle emission group
NCP	Nonlinear complementarity problem
NEDC	New European driving cycle
NLP	Nonlinear programming
NSGA	Non-dominated search genetic algorithm
OCP	Optimal control problem
OCV	Open circuit voltage
ODE	Ordinary differential equation
OPS	Output power-split
PEMS	Portable emissions measurement systems
PEO	Perfect elimination ordering
PG	Planetary gearbox
PHEV	Plug-in hybrid electric vehicle
PM	Permanent magnet
PMP	Pontryagins minimum principle

PMSM	Permanent magnet synchronous machines
QP	Quadratic programming
QSP	Quadratic subproblem
RB	Rule based
RCS	Row compressed storage
RHS	Right-hand side
RK	Runge–Kutta
SBX	Simulated binary crossover
SI	Spark-ignition
SIS	Standard injection scheme
SOCP	Switched optimal control problem
SOSC	Second-order sufficient conditions
SQP	Sequential quadratic programming
SR1	Symmetrical rank 1
STO	Switching time optimization
THS	Toyota hybrid system
TMPS	Two-mode power-split
TM	Trip management
TPBVP	Two-point boundary value problem
TWC	Three-way catalytic converter
UDDS	Urban dynometer driving schedule
US06	Supplemental federal test procedure
WLTP	World harmonized light test procedures

Symbols

Roman Uppercase

$\mathbf{A}$	Jacobian matrix of equality constraints (–)
$\mathbf{A}_{cps}$	Kinematic speed constraints of compound-split (–)
$\mathbf{A}_{ips}$	Kinematic speed constraints of input-split (–)
$\mathbf{A}_{ops}$	Kinematic speed constraints of output-split (–)
$\mathbf{A}_{se}$	Kinematic speed constraints of serial hybrid (–)
$\mathbf{A}_{tm}$	Kinematic speed constraints of two-mode power-split (–)
A_{sec}	Vehicle cross-sectional area (m^2)
$\mathbf{A}_y$	Jacobian matrix of all active constraints (–)
B	Ball around a point (–)
$\mathbf{B}$	Compressed matrix (–)
$\mathbf{B}_k$	Approximated Hessian (–)
$C(\cdot)$	Cycle of graph (–)
$\mathbf{D}_{cps}$	Kinematic constraints of compound power-split (–)
$\mathbf{D}_{ips}$	Kinematic constraints of input power-split (–)

$\mathbf{D}_{ops}$	Kinematic constraints of output power-split (–)
F	Faraday constant (C/mol)
$\mathbf{F}$	Matrix for obtaining KKT-system (–)
$\mathbf{F}(\cdot)$	Generalized system (–)
$F_a(\cdot)$	Acceleration resistance force (N)
$F_g(\cdot)$	Force due to gravity (N)
$F_{drag}(\cdot)$	Air-drag resistance force (N)
$F_{roll}(\cdot)$	Rolling resistance force (N)
$F_w(\cdot)$	Total friction force (N)
$F_{wh}(\cdot)$	Wheel force (N)
$G(\cdot)$	Graph (–)
$\mathbf{G}(\cdot)$	Canonical equations of continuous systems (–)
$\tilde{\mathbf{G}}(\cdot)$	BVP of continuous systems (–)
$\mathbf{H}_k$	Approximated of the inverse Hessian (–)
$\mathbf{H}_{red}$	Reduced Hessian (–)
H_l	Fuel's lower heating value (MJ/kg)
$\mathbf{I}$	Unity matrix (–)
$I_{bat}(\cdot)$	Battery current (A)
I_{bev}	Inertia of battery electric vehicle (kgm^2)
I_c	Inertia of engine (kgm^2)
$\mathbf{I}_{cps}$	Inertia matrix of compound power-split (kgm^2)
$I_d^r(\cdot)$	d-axis current (A)
I_{fd}	Inertia of final drive (kgm^2)
I_{gbx}	Inertia of gearbox (kgm^2)
I_{ice}	Inertia of ring (kgm^2)
$\mathbf{I}_{ips}$	Inertia matrix of input power-split (kgm^2)
I_{mg}	Inertia of rotor (kgm^2)
$I_q^r(\cdot)$	q-axis current (A)
I_r	Inertia of ring (kgm^2)
$\mathbf{I}_{ops}$	Inertia matrix of output power-split (kgm^2)
I_s	Inertia of sun (kgm^2)
$\mathbf{I}_{se}$	Inertia matrix of serial hybrid (kgm^2)
I_{veh}	Inertia of vehicle (kgm^2)
I_{wh}	Inertia of the wheel (kgm^2)
$\mathbf{K}(\cdot)$	Canonical equations of switched systems (–)
$\tilde{\mathbf{K}}(\cdot)$	BVP of switched systems (–)
N_a	Number of outputs of the associated function (–)
N_{c_u}	Number of control constraints for a continuous system (–)
N_{c_x}	Number of state constraints for a continuous system (–)
$N_{c_{x,q}}$	Number of state constraints for a hybrid system (–)
$N_{c_{x,u}}$	Number of mixed control-state constraints for a continuous system (–)
$N_{c_{u,q}}$	Number of control constraints for a hybrid system (–)

$N_{c_{x,u,q}}$	Number of mixed control-state constraints for a hybrid system (–)
N_e	Number of emission components (–)
N_{ent}	Number of entry time points for state constraints (–)
N_{ex}	Number of exit time points for state constraints (–)
N_{gbx}	Number of gears (–)
N_j	Number of switchings (–)
N_q	Number of discrete state variables (–)
$N_{\overline{q}}$	Size of discretized discrete state vector (–)
N_{swt}	Number of switchings (–)
N_t	Size of time grid (–)
N_u	Number of control-valued variables (–)
N_x	Number of continuous-valued state variables (–)
N_y	Number of optimization variables (–)
$N_{\overline{y}}$	Size of optimization vector (–)
$N_{\mathcal{G}_x}$	Number of discretization points in the grid $\mathcal{G}_x$ (–)
$N_{\mathcal{I}}$	Number of active indices (–)
N_ψ	Number of coupled boundary constraints (–)
N_{ψ_0}	Number of boundary constraints for the initial states (–)
N_{ψ_f}	Number of boundary constraints for the final states (–)
$L^1([t_0, t_f])$	Space of Lebesgue integrable functions (–)
$L^\infty([t_0, t_f])$	Space of Lebesgue essentially bounded functions (–)
L_d	d-axis inductance (H)
L_q	q-axis inductance (H)
$P(\cdot)$	Path of a graph (–)
$\mathbf{P}$	Orthogonal basis (–)
$P_{aux}(\cdot)$	Electrical on-board power (W)
$P_{bat}(\cdot)$	Battery power (W)
$\mathbf{P}_{cps}$	System matrix of compound power-split (–)
$\mathbf{P}, \mathbf{P}_f$	Permutation matrices (–)
$P_{gbx,1}(\cdot)$	Input gearbox power (W)
$P_{gbx,2}(\cdot)$	Output gearbox power (W)
$P_{ice}(\cdot)$	Engine power (W)
P_{ice}^{max}	Maximum engine power (W)
$\mathbf{P}_{ips}$	System matrix of input power-split (–)
$P_{mg}(\cdot)$	Motor power (W)
P_{mg}^{max}	Rated motor power (W)
P_{mg}^{n}	Nominal motor power (W)
$\mathbf{P}_{ops}$	System matrix of output power-split (–)
$\mathbf{Q}$	Orthogonal matrix (–)
Q_{bat}	Battery capacity (Ah)
$Q_{fuel}(\cdot)$	Integral of enthalpy flow (kWh)
R	Universal gas constant (J/molK)

$R(\cdot)$	Stability function (–)
$\mathbf{R}$	Regular upper triangular matrix (–)
R_{bat}	Battery resistance (Ohm)
$\mathbf{S}$	Seed matrix (–)
$T(\cdot)$	Torque (Nm)
$\mathbf{T}(\cdot)$	Transformation matrix (Nm)
$T_{brk}(\cdot)$	Mechanical brake torque (Nm)
$T_{clth}(\cdot)$	Clutch torque (Nm)
$T_{clth1}(\cdot)$	Input torque of clutch (Nm)
$T_{clth2}(\cdot)$	Output torque of clutch (Nm)
$T_{gbx}(\cdot)$	Gearbox torque (Nm)
$T_{gbx1}(\cdot)$	Input torque of the gearbox (Nm)
$T_{gbx2}(\cdot)$	Output torque of the gearbox (Nm)
$T_l(\cdot)$	Temperature-dependent frictional torque (Nm)
$T_m(\cdot)$	Rotor torque (Nm)
$T_{mg}(\cdot)$	Motor torque (Nm)
$T_{mg1}(\cdot)$	Torque of first motor (Nm)
$T_{mg2}(\cdot)$	Torque of second motor (Nm)
$T_n(\cdot)$	Nominal motor torque (Nm)
$T_{roll}(\cdot)$	Roll friction torque (Nm)
$T_{wh}(\cdot)$	Wheel torque (Nm)
$T_\chi(\cdot)$	Inner engine torque (Nm)
$V(\cdot)$	Value function (–)
$V_d^r(\cdot)$	d-axis voltage (V)
$V_q^r(\cdot)$	q-axis voltage (V)
$V_{oc}(\cdot)$	Open circuit voltage (V)
$\mathcal{A}$	Arc set (–)
$\mathcal{AC}([t_0, t_f], \mathbb{R}^{N_x})$	Space of absolutely continuous functions $f : [t_0, t_f] \to \mathbb{R}^{N_x}$ (–)
$\mathcal{B}$	Edge set (–)
$\mathcal{B}_{fill}$	Chordal extended edge set (–)
$\mathcal{C}^k([t_0, t_f], \mathbb{R}^{N_x})$	Space of k-times differentiable functions $f : [t_0, t_f] \to \mathbb{R}^{N_x}$ (–)
$\hat{\mathcal{C}}^k([t_0, t_f], \mathbb{R}^{N_x})$	Space of k-times piecewise differentiable functions $f : [t_0, t_f] \to \mathbb{R}^{N_x}$ (–)
$\mathcal{C}_{cl}$	Clique (–)
$\mathcal{D}$	Hybrid trajectory of discrete state functions (–)
$\hat{\mathcal{E}}$	Event set to change discrete state (–)
$\mathcal{F}$	Indexed collection of vector fields (–)
$\mathbf{G}(\cdot)$	RHS of the canonical equations (–)
$\mathcal{G}$	Discrete mode structure of system (–)
$\mathcal{G}_{sh}$	Multiple shooting grid (–)
$\mathcal{G}_t$	Discretization grid (–)

$\mathcal{G}_{t_2}$	Main discretization grid for switching time optimization (–)
$\mathcal{G}_{t_3}$	Minor discretization grid for switching time optimization (–)
$\mathcal{GP}(\cdot)$	Generalized plant (–)
$\mathcal{H}(\cdot)$	Hamiltonian (–)
$\mathcal{I}$	Set of active indices (–)
$\mathcal{J}$	Vertex coloring of a graph (–)
$\mathcal{K}(\cdot)$	Controller (–)
$\mathcal{L}(\cdot)$	Lagrangian (–)
$\mathcal{M}(\cdot)$	Sparsity index set for Jacobian matrix (–)
$\mathcal{N}(\cdot)$	Sparsity index set for Hessian matrix (–)
$\mathcal{O}(\cdot)$	Order (–)
$\mathcal{P}$	Neighborhood of $\mathbf{p}_0$ (–)
$\hat{\mathcal{P}}$	Feasible set for relaxed controls (–)
$\mathcal{Q}$	Feasible function space for discrete states (–)
$\hat{\mathcal{Q}}$	Feasible set for discrete states (–)
$\overline{\mathcal{Q}}$	Discretized set for discrete controls (–)
$\overline{\mathcal{Q}}_r$	Relaxed discretized set for discrete controls (–)
$\mathcal{R}^t(\cdot)$	Reachable set (–)
$\mathcal{S}$	Hybrid trajectory of continuous-valued state functions and feasible set (–)
$\mathcal{S}_{rk}$	Stability domain of the Dahlquist test function (–)
$\mathcal{T}$	Hybrid time trajectory and critical cone (s)
$\mathcal{T}_{ent}$	Set of entry points (s)
$\mathcal{T}_{ex}$	Set of exit points (s)
$\mathcal{U}$	Feasible function space for continuous-valued controls (–)
$\hat{\mathcal{U}}$	Feasible set for continuous-valued controls (–)
$\overline{\mathcal{U}}$	Discretized set for continuous-valued controls (–)
$\mathcal{V}$	Vertice set (–)
$\mathcal{W}$	Vertice set (–)
$\mathcal{X}$	Feasible function space for continuous-valued states (–)
$\hat{\mathcal{X}}$	Feasible set for continuous-valued states (–)
$\mathcal{Y}$	Neighborhood of $\mathbf{y}_0^*$ (–)
$\mathcal{Z}$	Hybrid continuous control trajectory (–)
$\mathcal{M}_1$	Quasi-static model of parallel hybrids (–)
$\mathcal{M}_2$	Thermodynamic model of parallel hybrids (–)
$\mathcal{M}_3$	Quasi-static model of power-split hybrids (–)
$\mathcal{M}_4$	Extended quasi-static model of parallel hybrids (–)
$\mathcal{P}_1$	Problem formulation without state jumps (–)
$\mathcal{P}_2$	Problem formulation without state jumps including gear sequence (–)
$\mathcal{P}_3$	Problem formulation with state jumps (–)

$\mathcal{P}_4$	Problem formulation for optimal catalytic converter heating (–)
A	Collection of reset functions (–)
$\mathcal{B}$	Collection of admissible discrete control spaces (–)
C	Collection of switching manifolds (–)
D	Collection of system constraints (–)
U	Continuous-valued control space (–)
X	Continuous-valued state space (–)

Roman Lowercase

$a(\cdot)$	Acceleration (m/s^2)
a_0	Rolling resistance (N)
a_1	Velocity dependent drivetrain resistance (Ns/m)
a_2	Aerodynamic resistance (Ns^2/m^2)
$c(\cdot)$	Color map (–)
c_w	Drag coefficient (–)
$\mathbf{c}_u(\cdot)$	Control constraints for continuous systems (–)
$\mathbf{c}_{u,q}(\cdot)$	Control constraints for hybrid systems (–)
$\mathbf{c}_x(\cdot)$	State constraints (–)
$\mathbf{c}_{x,u,q}(\cdot)$	Mixed control-state constraints (–)
d_f	Viscous damping coefficient of final drive (Nm/(rad/s))
d_{gbx}	Viscous damping coefficient of gearbox (Nm/(rad/s))
$\mathbf{d}_k$	Search direction (–)
$\mathbf{e}$	Unit vector (–)
$e_{gbx}(\cdot)$	Gearbox efficiency without lost (–)
$e_{ice}(\cdot)$	Engine efficiency without lost (–)
$e_{mg}(\cdot)$	Motor efficiency without lost (–)
$\mathbf{f}(\cdot)$	System's vector field (–)
$\mathbf{g}(\cdot)$	Inequality constraint function (–)
g	Acceleration due to gravity (m/s^2)
$g_1(\cdot)$	Engine map (rad)
$g_2(\cdot)$	Engine map (rad)
$g_3(\cdot)$	Engine map (Nm)
$g_4(\cdot)$	Engine map (–)
$g_5(\cdot)$	Engine map (°C)
$g_6(\cdot)$	Engine map (–)
$g_7(\cdot)$	Engine map (–)
$g_8(\cdot)$	Engine map (°C)
$\mathbf{h}(\cdot)$	Equality constraint function (–)
i_{01}	Stationary gear ratio of planetary gear 1 (–)
i_{02}	Stationary gear ratio of planetary gear 2 (–)
i_{fd}	Final drive gear ratio (–)

$i_{gbx}(\cdot)$	Time-based function of gear ratios (–)
$\mathbf{i}_{gbx}$	Vector of gear ratios (–)
i_{rd}	Reduction gear ratio (–)
i_{sr}	Stationary gear ratio (–)
$i_t(\cdot)$	Time-based function of transmission gear ratio (–)
$\mathbf{i}_t$	Vector of transmission ratios (–)
$l(\cdot)$	Lagrange-type objective (–)
m	Vehicle mass (kg)
$m(\cdot)$	Mayer-type objective (–)
$m_{air}(\cdot)$	Absolute cylinder charge (kg)
$m_{cyl}(\cdot)$	Relative cylinder charge (%)
m_{wh}	Wheel mass (kg)
n_e	Number of free electrons (–)
p	Pole pairs (–)
$p_1, p_2, p_3, p_4, p_5, p_6$	Engine temperature model parameters (–)
p_7	Engine temperature model parameters (1/kg)
p_8	Engine temperature model parameters (1/s)
$q(\cdot)$	Discrete state of the current subsystem (–)
$q^*(\cdot)$	Optimal discrete state of the current subsystem (–)
$q^\circ(\cdot)$	Discrete state of the next subsystem (–)
q^-	Discrete state at time t_j^- (–)
q^+	Discrete state at time t_j^+ (–)
r_r	Radius of the ring (m)
r_s	Radius of the sun (m)
r_{wh}	Wheel radius (m)
t	Time (s)
t_0	Initial time (s)
t_f	End time (s)
$\mathbf{u}(\cdot)$	Controls (–)
$\hat{\mathbf{u}}(\cdot)$	Relaxed controls (–)
$\mathbf{u}^*(\cdot)$	Optimal controls (–)
$\mathbf{w}$	Discrete variable and mathematical variable (–)
$v(\cdot)$	Velocity (m/s)
$v_{crp}(\cdot)$	Creep velocity (m/s)
$\mathbf{x}(\cdot)$	Continuous-valued states (–)
$\mathbf{x}^*(\cdot)$	Optimal continuous-valued states (–)
$\mathbf{y}$	Optimization vector (–)

Greek Lowercase

$\alpha(\cdot)$	Road slope (rad)
α_k	Step-size (–)

$\beta(\cdot)$	Fuel consumption (l)
$\delta(\cdot)$	State-jump (–)
ϵ	Perturbation step-size (–)
$\chi(\cdot)$	Ignition angle (°CA)
$\xi(\cdot)$	State of charge (–)
$\eta_{bat,g}$	Global battery efficiency (–)
$\eta_{bat,l}(\cdot)$	Local battery efficiency (–)
$\eta_{gbx}(\cdot)$	Gearbox efficiency (–)
$\eta_{ice}(\cdot)$	Engine efficiency (–)
$\eta_{tot}(\cdot)$	Total efficiency of power-split (–)
$\eta_{twc}(\cdot)$	Conversion efficiency of three-way catalytic converter (–)
$\eta_{\mathrm{d}\chi}(\cdot)$	Combustion efficiency (–)
γ_m	Mass factor (–)
γ_f	Product of natural constants $(\cdot)$
$\boldsymbol{\lambda}$	Lagrange multiplier (–)
$\boldsymbol{\lambda}(\cdot)$	Costates (–)
$\hat{\boldsymbol{\lambda}}$	Guessed initial costates (–)
$\hat{\boldsymbol{\lambda}}^{max}$	Upper bound of guessed initial costates (–)
$\hat{\boldsymbol{\lambda}}^{min}$	Lower bound of guessed initial costates (–)
$\boldsymbol{\mu}$	Lagrange multiplier (–)
$\boldsymbol{\pi}$	Elimination ordering (–)
$\boldsymbol{\pi}_f$	Fill-in reduced elimination ordering (–)
$\varpi_q(\cdot)$	Discrete control (–)
$\theta_{cw}(\cdot)$	Temperature of coolant water (°C)
$\theta_{cyl}(\cdot)$	Temperature of cylinder (°C)
$\theta_{exh}(\cdot)$	Temperature of raw exhaust (°C)
$\theta_{twc}(\cdot)$	Temperature of three-way catalytic converter (°C)
ρ	Specific density of air (kg/m^3)
$\boldsymbol{\rho}(\cdot)$	Relaxed controls (–)
$\boldsymbol{\sigma}(\cdot)$	Binary controls (–)
$\hat{\boldsymbol{\sigma}}(\cdot)$	Relaxed binary controls (–)
$\varsigma(\cdot)$	Time transformation function (–)
ω	Mathematical variable (–)
$\omega_c(\cdot)$	Angular speed of carrier (rad/s)
$\omega_{clth}(\cdot)$	Angular speed of clutch (rad/s)
$\omega_{fd1}(\cdot)$	Input angular speed of final drive (rad/s)
$\omega_{fd2}(\cdot)$	Output angular speed of final drive (rad/s)
$\omega_{gbx1}(\cdot)$	Input angular speed of gearbox (rad/s)
$\omega_{gbx2}(\cdot)$	Output angular speed of gearbox (rad/s)
ω_{ice}^{max}	Maximum angular speed of engine (rad/s)
$\omega_{ice}(\cdot)$	Angular speed of engine (rad/s)
$\omega_{mg}(\cdot)$	Motor/Generator angular speed (rad/s)
$\omega_n(\cdot)$	Nominal M/G angular speed (rad/s)

$\omega_r(\cdot)$	Angular speed of ring (rad/s)
$\omega_s(\cdot)$	Angular speed of sun (rad/s)
$\omega_{wh}(\cdot)$	Wheel angular speed (rad/s)
ϵ	Power-splitting factor (–)
$\psi(\cdot)$	General boundary constraints (–)
$\psi_0(\cdot)$	Initial boundary constraints (–)
$\psi_f(\cdot)$	Terminal boundary constraints (–)

Greek Uppercase

$\boldsymbol{\Gamma}_f(\cdot)$	Increment function of the RHS $\mathbf{f}(\cdot)$ of the ODE (–)
$\boldsymbol{\Gamma}_G(\cdot)$	Increment function of the RHS $\mathbf{G}(\cdot)$ of the canonical equations (–)
$\Gamma_l(\cdot)$	Increment function of the RHS $l(\cdot)$ of the Lagrangian (–)
$\boldsymbol{\Lambda}(\cdot)$	Matching conditions for indirect multiple shooting methods (–)
Ψ_m	Peak flux linkage (Wb)
Θ	Switching sequence (–)
Θ_t	Switching time sequence (–)
$\boldsymbol{\Upsilon}(\cdot)$	Final conditions for indirect single shooting methods (–)

Notation

$b(\cdot)$, $T(\cdot)$	b and T are functions
$\mathbf{b}$	$\mathbf{b}$ is a vector
$\mathbf{B}$	$\mathbf{B}$ is a matrix
$\mathbf{b}_{[i]}$	The i-th element of the vector $\mathbf{b}$
$\overline{\mathbf{b}}$	$\overline{\mathbf{b}}$ is a vector and obtained by a discretization process of $b(\cdot)$
$\overline{\mathbf{T}}$	$\overline{\mathbf{T}}$ is a vector and obtained by a discretization of a variable with physical meaning; here torque

Chapter 1
Introduction

1.1 Motivation, Challenges, and Objectives

Individual mobility has become an inherent part in people's life since it has been available to large parts of the society, due to falling production cost and higher living standards. The automotive industry as well as the corresponding industry sector have a significant impact on economy as well as on ecology. Worldwide, the number of vehicles is steadily growing, which causes significant environmental problems. About 19% of the worldwide, carbon dioxide (CO_2) emissions are attributable to passenger cars and trucks (IEA [28]).

Locally, because noxious emissions (hydrocarbons, particles, and many others) endanger the health and quality of people living especially in large urban areas; globally, even the emission of on first sight harmless combustion products such as carbon dioxide constitutes one of the biggest challenges of our time, *global warming*. Reflecting this, the reduction of CO_2 emissions and other *greenhouse gas* (GHG) emissions that are responsible for global warming is one of the major challenges of our time and has therefore become part of the legislation in most parts of the world. It has been worldwide recognized that limiting the rise in global mean temperature to 2 °C is a central climate goal (IEA [30]). Therefore, the European Union has made substantial efforts in tightening of the CO_2 target from 130 g CO_2/km by 2015 to 95 g CO_2/km by 2020.

On the other side, low oil prices over the last decades and a demand for growing individual mobility with increased comfort and installed power have resulted in reduced interests in fuel economy optimized cars. *Battery electric vehicles* (BEV) are currently seen as a way to inspire enthusiasm to new vehicle technologies. BEVs allow to drive locally with zero emissions and to increase at the same time the installed power for propelling the vehicle without bad conscience. If the electric energy used for propelling can be derived from renewable energy sources, this vehicle technology is a promising way to reduce global warming. However, the biggest challenge for BEVs is still the storage of electric energy. Modern batteries have improved significantly in efficiency and capacity but to cover driving distances (500 km as

T.J. Böhme and B. Frank, *Hybrid Systems, Optimal Control and Hybrid Vehicles*,
Advances in Industrial Control, DOI 10.1007/978-3-319-51317-1_1

minimum for today's vehicle, cf. Tanaka et al. [68]) comparable to those of vehicles with conventional combustion engines, a battery pack with an additional mass of several hundred kilograms is required. If with repeated deep discharges, the battery capacity will need to be at least 75 kWh, as it was outlined by Tanaka et al. [68].

The prospects for commercially competitive BEVs are highly dependent on the battery costs. One says that the point of commercialization is reached at the cost breakthrough of 150 USD/kWh (Nykvist and Nilsson [47]). However, the estimation of the current and future battery costs is not a straightforward task. Many different cost estimations exist in the literature, since it depends on the battery chemistry and battery application. Noteworthy are the cost estimation by Dinger et al. [17] and Nykvist and Nilsson [47]. Dinger et al. [17] estimated the battery cost of automotive lithium-ion battery packs in 2010 to 1100 USD/kWh. Even at estimated high-volume battery prices for lithium-ion technology of approximately 410 USD/kWh (Nykvist and Nilsson [47]) for the industry as a whole and 300 USD/kWh for market-leading manufacturers, the battery alone would cost 30750–22500 USD per vehicle, respectively, which is still too costly. Thus, to make BEVs affordable in the near-term, most recently announced models have shorter driving ranges. Despite the increased performance batteries are still energy intensively manufactured using materials which are harmful to the environment. This is certainly a big disadvantage and might be regarded in the design process as an initial negative energy offset.

Despite their problems, however, batteries are very interesting "medium-term" energy storage devices. Their potential comes positive in appearance in the collaboration with at least one electric energy converter, which are added to a conventional powertrain with combustion engine and fuel tank. Such a powertrain setup constitute to *hybrid electric vehicles* (HEV). In case, the battery can be charged externally from a power grid, the hybrid vehicle is called *plug-in hybrid electric vehicles* (PHEV).

Hybrid vehicles, in general, are considered as a bridge technology to BEVs. Their powertrain setups provide additional degrees of freedom that can be exploited to reduce the fuel consumption and to avoid at least partly local emissions while extending the driving range of BEVs significantly.

The PHEV/BEV market is of major importance for the automotive manufactures. Despite the growing numbers of passenger cars, there are certain signs that the whole market for cars will stagnate or even shrink in the future. It is therefore of strategic relevance to access this market that clearly brings the advantage of *green economy labeling*. Surveys have shown that there exists a considerable market for PHEVs and BEVs (cf. Tanaka et al. [68], Trigg et al. [69]) as shown in Fig. 1.1.

To fulfill the ambitious aims, it is important that all renowned automotive manufactures offers a steadily growing number of PHEV/BEV types which meet various customer concerns such as driving range, energy efficiency, acquisition cost, operating cost, noise pollution, and environment-friendly production. For PHEVs, specific homologation requirements in each country such as GHG emissions, nitrogen oxide emissions, and fine particle emissions must also be satisfied as well. Certainly, it also depends strongly on the behavioral change of the customers to reduce energy use and to trust such technologies. Nonetheless, factors that are controlled by political

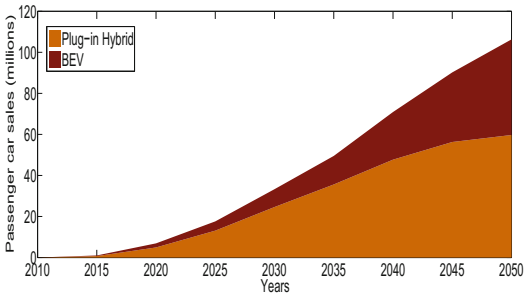


Fig. 1.1 Annual global BEV and PHEV sales in BLUE Map scenarios (Tanaka et al. [68]). Assumptions are vehicle model types are steadily growing, BEVs have an average all-electric range of 150 km, and PHEVs have a minimum all-electric range of 40 km

initiatives such as the share of low-carbon electricity on the energy-mix play a major role as well.

In order to cope with this huge challenge, attractive and efficient hybrid powertrain technologies must be developed in the coming years. This leads to inexorably increasing number of components in the layout of hybrid powertrain systems. Heuristic methods become then very rapidly limited in their performance. To facilitate this enormous challenge the main objectives of this text are to introduce mathematical models of the powertrain components and large-scale optimal control optimization methods that permit engineers and scientist a systematic analysis and minimization of the vehicle's energy consumption.

1.2 Vehicle Design Aspects

The following objectives can usually be found in a technical specification sheet before the design and calibration of a vehicle begins:

- reduction of monetary energy cost (e.g., mix of fuel and electrical energy consumption);
- minimization of CO_2;
- minimization of nitrogen oxide(s) (NO_x);
- minimization of further GHG emissions; and
- improvement of driving comfort and performance.

Design goals like CO_2, NO_x, and GHG emissions are dictated by local regulation authorities, whereas monetary energy cost, driving performance, and comfort goals are formed by the customer's expectations.

Unfortunately, these design goals are contradictory because high driving performance does not result in low fuel consumption or emissions do not have their minimum at the same operating conditions. This forces engineers to make trade-offs in their design procedure, which leads naturally to a *multi-objective* problem as depicted in Fig. 1.2.

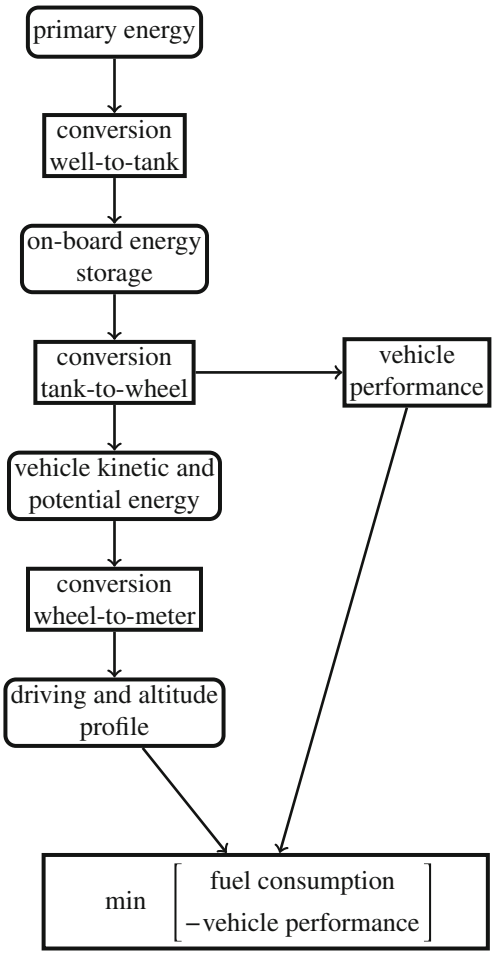


Fig. 1.2 Elements of a multi-objective vehicle design

The chain from the primary energy source to the covered distance is characterized by the efficiencies of the energy stages, which are discussed in the next section. The problem formulation can certainly include more objectives. Additional design targets could be the reduction of component wear, component aging, and the reduction of energy cost during production.

1.2.1 Stages of Energy Conversion

In a modern view of vehicle design the complete energy flow is decomposed into three energy conversion steps. This procedure describes the complete energy evolution

including all losses and begins from the primary energy source up to the driven meters:

- well-to-tank;
- tank-to-wheel; and
- wheel-to-meter.

In the *well-to-tank* conversion step, the primary energy source (e.g., fossil hydrocarbons, renewable energy sources, uranium, and so on) is converted to an energy carrier (e.g., by refineries, power plants, etc.) that is suitable for on-board storage, e.g., gasoline, electric energy. The on-board energy is then converted to mechanical energy in the *tank-to-wheel* conversion step. The mechanical energy is stored via the *wheel-to-meter* conversion as kinetic and potential energy in the vehicle movement. Figure 1.3 exemplifies the conversion steps for an HEV.

Let us view some prominent actions to reduce the energy conversion steps. A general action for the improvement of the well-to-tank efficiency for all combustion-based powertrains is the usage of fuel that causes less CO_2 when combusted. Vehicles that use traction power from battery units profit from increasing the charger efficiency. PHEVs and BEVs can drive some distance only battery powered. This distance is denoted as *all-electric-range* (AER). In this operating mode the vehicles do not exhaust any emissions. One says, the vehicles produce zero-emissions locally.

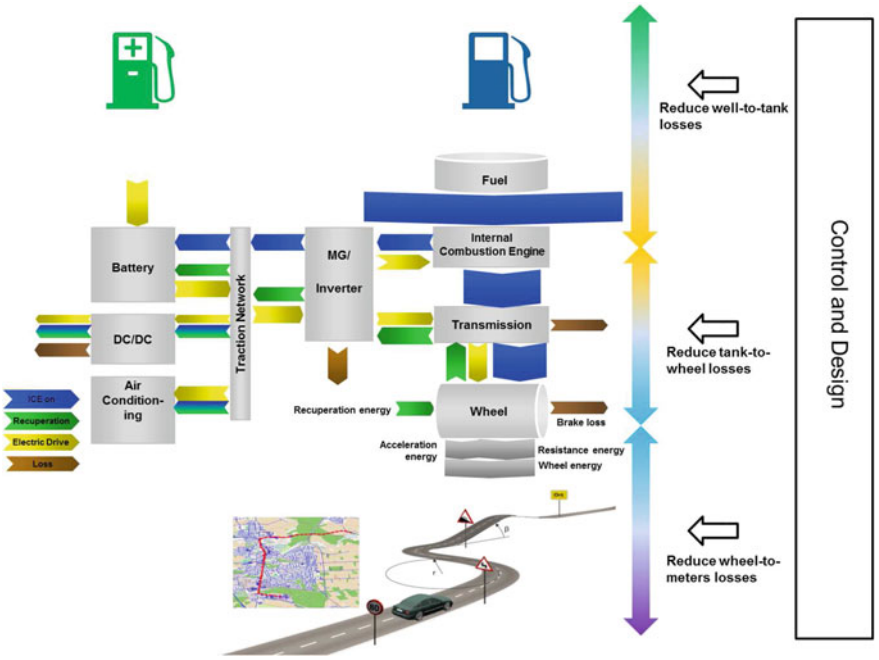


Fig. 1.3 Exemplified well-to-meter conversion steps of an HEV and the aim of a control and design procedure to minimize their losses

Globally, however, the exhaust emissions depend on the energy-mix provided from coal, nuclear, and renewable energy power plants. In order to make such cars competitive with cars with conventional powertrains, it is important to reduce the CO_2 contribution in grid electricity below that level of conventional cars emitted locally by combusted fossil fuel. This goal is called low-carbon electricity.

Actions to improve the tank-to-wheel conversion are well researched, including increase of average engine peak efficiency, increase of mean pressure when engine is on, increase of electric motor/generator efficiency, and so on. The arrangement of the components using an appropriate powertrain plays also an important role and is still a demanding topic. PHEVs are a potentially important technology for reducing the fuel consumption and CO_2 emissions because they can run on electricity for a certain distance after each recharge, depending on their battery's energy storage capacity—expected to be typically between 20 and 80 km. For example, in Europe, 50% of the trips are less than 10 km and 80% of the trips are less than 25 km daily (cf. Tanaka et al. [68]). Today, BEVs are designed to run an AER of approximately 200 km. Both technologies have motivated many countries around the world to promote these vehicle types. The aggregated goal for all countries with known deployment targets is 7.2 million PHEV/BEV sales (Trigg et al. [69]) by the end of 2020. For instance, a huge electric mobility development plan has been initiated by the German Government to increase the number of BEVs on German roads.

An important advantage of BEVs over conventional powertrain vehicles is the very high efficiency and the relative low cost of the electric traction motor. The disadvantages of BEVs are

- traction batteries are expensive, temperature prone, and their energy densities are small compared with fossil energy carriers (e.g., gasoline);
- driver's worry of abrupt stopping due to unanticipated exhausted battery power in a region without power grids is high; and
- relative long charging times compared with fuel-refilling.

These issues are still challenging problems for the automotive manufactures and their suppliers to convince potential customers. For safety reasons, the battery capacities are oversized and much greater than the battery capacities of PHEVs in order to guarantee a minimum acceptable driving range and peak power. This reveals the question of proper component sizing to the customer's expected driving habits.

The *wheel-to-meter* energy conversion depends strongly on the drive profile that the vehicle follows. In terms of energy efficiency, it is in many scenarios beneficial to decrease the vehicle speed to obtain better energy efficiency per kilometer. Lower vehicle speed trajectories can be generated by optimization procedures for recommended target speeds as shown by Boehme et al. [11]. Such eco-driving techniques can be a result of driving assistance systems such as *adaptive cruise control* (van Keulen et al. [34]). A challenge of such systems is to provide recommendations where the drivers do feel comfortable and safe (Charalampidis and Gillet [15]). The same applies for the generation of less-energy consuming acceleration profiles. These trajectories are usually employed to limit automatically the traction power of the vehicles, but in emergency scenarios the full power must be available.

Many research activities have been undertaken in the field of *intelligent traffic control*. Such systems may control the traffic flow and traffic density to achieve a minimum of total time spent, braking and stop durations, emitted emissions, etc., as reported in Liu et al. [43].

Energy consumption also depends on the road topology and therefore on the road slope $\alpha(\cdot)$. This inspires the choice of a less energy-demanding route by avoiding mountain or hilly roads. Predictive energy management strategies (Back [3], Boehme et al. [12], Schori et al. [58]) take advantage of nowadays available geographic information system and can reduce the energy consumption considerably. In this context, Chap. 12 presents some predictive control strategies to minimize the tank-to-wheel/tank-to-meter energy consumption.

The following design parameters have further impact on the wheel-to-meter efficiency. Reducing the rolling resistance can be achieved by applying wheels with lower rolling coefficient. This coefficient increases nonlinearly with the vehicle speed and depends strongly on the tire pressure. Engine downsizing can lead to smaller engine inertia. Some engines employed in hybrid powertrains are totally without a dual mass flywheel but require that the motor/generator has to be connected to the engine for idling. Light-weight vehicle construction methods using aluminum space frames or carbon fiber reinforced composites are popular methods to reduce the vehicle's total mass. Unfortunately, the masses from the electrified components, e.g., motor/generator or high-voltage battery, have a negative impact on the vehicle's total mass. This is especially painful if the components are not properly designed or even oversized. Figure 1.4 illustrates this exemplarily for a vehicle with and without recuperation device.

It can be seen from the second subfigure that the saving due to the recuperation device is negligible. The additional mass increases the forces due to acceleration and roll resistance such that the corresponding energy is nearly balanced with the energy recovered during braking.

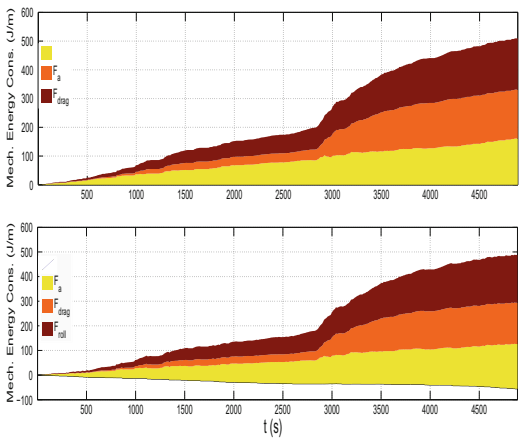


Fig. 1.4 Energy losses for a real-world drive cycle of as follows: **a** conventional vehicle with 1500 kg total mass, **b** vehicle with an additional recuperation device of 150 kg

In the past decades, a growing awareness has been developed in the automotive community that for a comprehensive analysis of the energy consumption the primary energy choice and all three energy conversion steps have to be considered. The impact of the primary energy choice and the well-to-tank efficiency on the vehicle design are definitely of great relevance but is beyond the scope of this book. We will concentrate in this text to technical devices and solutions, which are well-recognized to minimize the substantial energy losses in the tank-to-wheel/tank-to-meter energy conversion.

1.2.2 Real-World Driving Profile, Consumption, and Emissions

It is known from various studies that vehicle designs and calibrations strongly depend on the drivers specific behavior, habits, and environment. These important properties are reasonable represented by real-world drive cycles and the corresponding altitude profiles. For instance, the real-world drive cycles have a big impact on the sizing and aging of battery capacities of BEVs and PHEVs. Such effects have been investigated in Kwon et al. [40], Fellah et al. [19] by classifying cycles with different aggressiveness and driving ranges. The optimal battery capacity may also vary by regional market (e.g., in North America larger minimum trip-ranges are required compared with Europe and Japan) and consumer group. The real-world drive cyles do not only influence the size of the battery but also its usage. In terms of batteries this is the duty (recharge/discharge) cycle. Batteries for PHEVs and BEVs have different duty cycles. PHEV batteries are subject to deep discharge cycles, in addition to frequent shallow cycles for power assist and regenerative braking when the engine is in hybrid mode. Batteries for BEVs are more likely to be subject to repeated deep discharge cycles without as many intermediate or shallow cycles (Tanaka et al. [68]). In both cases, the demands differ from those on batteries used in conventional hybrid electric vehicles, which experience almost exclusively shallow discharge/recharge cycling.

For manufacturers, it is very interesting to analyze components to over-proportional stress by generating agressive real-world speed and acceleration profiles. This allows engineers to conclude at an early stage of the design the weaknesses of the components.

Today, the calibration of vehicles to achieve type-approval fuel economy is based on prescribed homologation test cycles. These tests are conducted under official authority and confirm that the fuel consumption of vehicle production samples will meet the homologation requirements. The test cycles are performed under predefined conditions in a chassis dynamometer laboratory as reported in Franco et al. [20, 21] that clearly do not capture some relevant aspects of realistic driving scenarios. To compensate this lack, the vehicles driving performance is calibrated consecutively under real-world driving conditions. It is therefore not unusual that automotive manufacturer define their own additional real-world test cycles to capture relevant regional characteristics (e.g., varying topologies, stop probabilities, etc.). The problem of this

two-stage calibration process is the decoupling effect, which means that issues of the driving performance are not regarded for the type-approval fuel economy and vice versa.

This has led to a controversial discussion in recent years, since substantial evidence has been provided that evaluated CO_2 values under real-world scenarios with more demanding driving conditions are much higher compared to the laboratory evaluated test cycles. This caused great uncertainty to customers. As a reaction, real-world driving scenarios moved more and more in the focus, since powertrain designs and calibrations with respect to less representative test cycles are not meaningful.

Substantial efforts have been made in the European Union over the recent years to update the current type-approval procedure, called *motor vehicle emission group* (MVEG), by more realistic drive conditions with higher load and speed scenarios. An attempt is to introduce the *worldwide harmonized light vehicles test procedures* (WLTP) approach that aims to better represent actual vehicle operations on the road. WLTP has been in development since 2007 within the activities of the United Nations Economic Commission for Europe Working Party on Pollution and Energy and aims at providing a worldwide harmonized method to determine the levels of gaseous and particulate emissions, CO_2 emissions, fuel consumption, electric energy consumption, and electric range from light-duty vehicles in a repeatable and reproducible manner designed to be representative of real-world vehicle operation. The WLTP seems to be a reasonable step toward real-world emissions but this regulation will enter into force 2017 at the earliest.

Already 2011 and probably earlier some European institutes, among them the *Joint Research Center*, investigated the difference in CO_2 emissions when measured using real-world and type-approval approaches. They proposed a correlation-based formula for estimation of the real-world (also known as in-use) CO_2 emissions of the passenger vehicle fleet. The data have been collected by EU Member States. The empirical models were constructed based upon linear combinations of key variables including the vehicle mass, engine displacement, rated power, and power to mass ratio (Mellios et al. [44]). With a simplified set of quantities (cf. IEA [29]) one obtains

$$\beta_{fuel,iu}(t) = 1.15 + 0.000392 \cdot V_d + 0.00119 \cdot m + 0.643 \cdot \beta_{fuel,tp}(t) \qquad (1.1)$$

where $\beta_{fuel,tp}(\cdot)$ is the type-approval fuel consumption in l/100 km, m is the vehicle reference mass in kg (empty weight + 75 kg for driver and 20 kg for fuel), and V_d is the total displacement of the internal combustion engine in cm^3. The in-use correction formula (1.1) has been derived for gasoline ICE based-vehicles and serves as estimation formula for real-world fleet consumption.

Chassis dynamometer studies like the WLTP are typically more precise and repeatable than those obtained from real-world scenarios. However, in the past few years, *portable emissions measurement systems* (PEMS) have experienced a remarkable technological development as reported in Franco et al. [20]. PEMS are complete sets of emission measurement instruments that can be carried on-board the vehicle and measure in real time the emitted emissions under dynamic load.

1.3 Process Model, Control Strategy, and Optimization

Let us collect the challenges from the last sections:

- improved performance and fuel economy can only be obtained with complex powertrain ensembles;
- multiple contradictory design goals available;
- satisfying real-world drive cycles, which may consist of many time discretization points;
- a control strategy may be unknown prior to the design stage (structure and control parameters).

It seems to be obvious that these challenges can not be solved with heuristic models and methods. A common approach is to model the powertrain using first principles and to simulate the closed-loop system. This simulation-based approach, however, works only satisfactorily if the control strategy is a priori known or can easily obtained from prior control design results.

A more general approach that is discussed in this book is to find optimal control and state trajectories using numerical optimizations, which are the basis for determining a proper control structure.

1.3.1 General Problem Statement

All of the design aspects mentioned before are quite complex in nature and need to be structured. An elegant way is to cast the design problems into a modern control view—an uniform abstraction level. In general, design problems have parts without and with time dependencies. The latter one needs some control strategy to influence the dynamics in such a way that the specified aims of the system are achieved. This result in a pair of plant and controller which can be connected as shown in Fig. 1.5. Here, $\mathcal{GP}(\cdot)$ is the *generalized plant* and $\mathcal{K}(\cdot)$ is the control strategy.
Here, the vector $\mathbf{w}(t) \in \mathbb{R}^{N_w}$ represents external inputs to the control system, such as

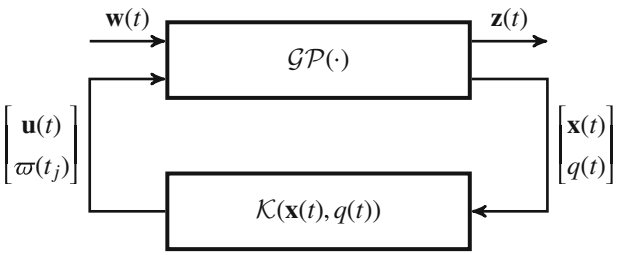


Fig. 1.5 Modern control loop with generalized plant

$$\mathbf{w}(t) = \begin{bmatrix} \text{references}(t) \\ \text{disturbances}(t) \\ \text{parameters} \end{bmatrix}.$$

$\mathbf{u}(t) \in \mathbb{R}^{N_u}$ is a vector of continuous-valued controls that assumes real values within a given convex set with nonempty interior and is called for short *continuous-valued controls*. $\varpi(\cdot)$ is a function that defines events at time instances t_j as

$$\begin{cases} \varpi(t) \in \{1, 2, \ldots\}, & t = t_j \\ \varpi(t) = 0, & t \neq t_j \end{cases}$$

and is called *discrete control*. $\mathbf{x}(t) \in \mathbb{R}^{N_x}$ is a vector of continuous-valued states that assumes real values and is called for short *continuous-valued states*. $q(\cdot)$ is called *discrete state* and is defined on a finite set of values $q(t) \in \{1, 2, \ldots\}$. $\mathbf{z}(t) \in \mathbb{R}^{N_z}$ is a *fictitious output vector* that is used to express design specifications.

The general process $\mathcal{GP}(\cdot)$ represents the (modeled or real) vehicle and contains many different information, for instance

- **references** specify the set points of the control system;
- **disturbances** may be unknown a priori. Measurable or observable examples are road slope and curves, road friction, traffic flow;
- **parameters** are time independent. They influence the vehicle design and consequently the efficiency of the energy conversion for each energy converter in the system. Examples are vehicle mass, aggregated vehicle inertia, wheel radius, rolling coefficient, air drag coefficient, maximum vehicle cross section area, efficiency of the power converters, efficiency of the power source, efficiency of the power grid, and so forth;
- **continuous-valued states** represent measurable or observable operating conditions of continuously acting physical units. Examples are vehicle velocity, battery energy, and so forth;
- **discrete state** represents the operating state of discrete acting physical units like valves, thyristoren, etc. and of control software structures, which change their discrete values at time t_j;
- **continuous-valued controls and discrete control**, both, influence the energy flow. Examples are torque of the converters, vehicle speed, and on/off commands for power switches and valves;
- **fictitious outputs** calculate or measure important auxiliary variables. Examples are fuel consumption, emissions, number of on/off switchings; and
- **design specifications** enforce constraints to the fictitious outputs. Examples are low fuel consumption, low frequency of switchings.

The parameters—more precisely *design parameters*—span the *configuration space* and influence the energy flow in a static but global way. The time-dependent variables of a specific vehicle configuration are the controls and states. The evolution of the continuous-valued states is described by a set of partial and/or ordinary differ-

ential equations. While the evolution of the discrete state is described by transition equations. Thus, the controls drive the mixed states consisting of continuous-valued states and discrete state and influence the energy flow of a vehicle configuration in a dynamic but local way.

It is important to understand the relationship between global and local energy efficiencies, which should not be mistaken with global and local minima of functions. The first one determines the time independent global efficiency for the entire drive mission, whereas the latter one determines the instantaneous efficiency. Both together compose the total efficiency of the vehicle

$$\eta_{veh,tot}(t) = \eta_{veh,g} \cdot \eta_{veh,l}(t).$$

This reveals the importance that for a good vehicle design both efficiencies have to be maximized. Moreover, these efficiencies are cross-coupled which makes the design procedure a challenging task. A separation into two distinct design procedures is the preferred methodology in practice but requires a full enumeration of the design variants. Consequently, for each vehicle configuration a feasible control strategy $\mathcal{K}(\cdot)$ has to be found. A control strategy $\mathcal{K}(\cdot)$ links the continuous-valued states and the discrete state and maybe further information to generate the feasible continuous-valued controls and discrete control. Feasibility means the controls and states satisfy constraints implied from the vehicle configuration. If not, then the complete vehicle configuration is infeasible. This reveals issues that are commonly affected in control system design. It is good practice to employ the following conditions on the design of the control strategy $\mathcal{K}(\cdot)$ such that the control law generates efficient and robust feedback control:

- **control and state constraints**. One encounters on real physical systems constraints on continuous-valued controls and continuous-valued states, which limits the performance of the closed-loop system;
- **feasibility**. The control policy must satisfy technical, economical, and environmental constraints;
- **action**. Acting on systems requires energy and has to be considered in the control design. This can be a challenging modeling task, which usually needs some simplifications; and
- **uncertainty**. Assuming a complete characterization of the behavior of the system is highly unrealistic, which requires that the control design must perform well in the presence of uncertainty.

1.3.2 Energy Management

Energy management strategies $\mathcal{K}(\cdot)$ have an huge impact on the local energy consumption of hybrid vehicles, since the newly gained degrees of freedom can be controlled to improve the overall system efficiency.

In general, it is desirable to provide optimal controls in order to minimize fuel consumption and the related emissions without compromising the performance of the vehicle. This task is dedicated to *energy management* of a hybrid vehicle, which is located on the *electronic control unit* (ECU) and consists of interfaces, software codes, and many calibration parameters. The complexity of energy management depends on the degree-of-freedom of the hybrid powertrain. For example, the repartition of torque, which depends on the number of electric machines installed in the powertrain. The hybrid powertrain is usually controlled by a control policy that consists of several layers. The upper layer constitutes a *supervision* control structure that is connected to energy management and is responsible for determining the set point values for the underlying control loops.

The energy management controller $\mathcal{K}(\cdot)$ from Fig. 1.5 can be realized using different strategies. Let us review the most important ones.

Heuristic and *rule-based* (RB) strategies as discussed by Schouten et al. [59] have the advantage of being completely causal and hence directly applicable to given hybrid powertrain configurations. These approaches to find the suboptimal calibration parameters are well established in practice. It is obvious, because of the wide number of parameters, such energy managements result in hard calibration tasks and no evidence can be given that the solution is even nearly optimal. Therefore, heuristic design processes are time-consuming and cumbersome. The results are usually limited to one specific vehicle configuration.

In the literature, a well-recognized solution method to tackle this problem is the *equivalent consumption minimization strategy* (ECMS) (Serrao et al. [62]). The ECMS strategy has been first introduced by Paganelli et al. [48] as a method to reduce an optimization problem to an instantaneous minimization problem. The method is based on an instantaneous minimization of a sum of fuel mass flow and weighted electrical power. The equivalent fuel consumption can be formulated as

$$\min_{\mathbf{u}(t)} \phi_{fuel}(\mathbf{u}(t)) = \int_{t_0}^{t_f} \dot{m}_{eq}(\mathbf{u}(t), s(t)) = \int_{t_0}^{t_f} \dot{m}_{fuel}(\mathbf{u}(t)) + \frac{s}{H_l} \cdot P_{bat}(\mathbf{u}(t)) \, \mathrm{d}t \tag{1.2}$$

where $\dot{m}_{eq}(\cdot)$ and $\dot{m}_{fuel}(\cdot)$ are the equivalent fuel mass flow of the hybrid system and the fuel mass flow of the IC engine, respectively, both in kg/s, $P_{bat}(\cdot)$ is the electrical battery power, and H_l is the fuel's lower heating value (energy content per unit of mass). s is the equivalence factor which converts the electrical battery power to an equivalent fuel mass flow and has a major impact on the success of this methodology.

The strength of this method is certainly the low computation time, which makes it a candidate for the implementation as an online control strategy. However, the weighting factor s has a significant influence on the value of the equivalent fuel consumption. In practical applications, the determination of a meaningful equivalence factor can be a hard task, since it strongly depends on future drive cycles which are not a priori known. Its value affects the effectiveness of the control strategy to maintain the charge of the battery. If it is too high, more penalties are given to the consumption of electric energy, which prevents electric driving. If it is too low, electric energy is cheap for propelling the vehicle and therefore the charge of the battery

is likely to be completely depleted. The operation to maintain the battery's charge is called *charge sustaining*. The operation of this simple control strategy leads to a nonrobust behavior because the objective function (1.2) does not explicitly include information about the state of charge $\xi(\cdot)$. Thus, the instantaneous minimization of (1.2) cannot maintain the charge of the battery within a predefined range. To overcome this drawback, a correction $p(\xi(\cdot))$-term (Paganelli et al. [49], Serrao et al. [62]), which is a function of the state of charge, can be introduced to compensate deviations from the set-point. This extends the ECMS strategy to

$$\min_{\mathbf{u}(t)} \phi_{fuel}(\mathbf{u}(t)) = \int_{t_0}^{t_f} \dot{m}_{fuel}(\mathbf{u}(t)) + \frac{s}{H_l} \cdot P_{bat}(\mathbf{u}(t)) \cdot p(\xi(t)) \, \mathrm{d}t. \tag{1.3}$$

The performance of (1.3) can be further improved by adaptive tuning of the equivalence ratio s (Musardo et al. [45]), which then becomes a function of time namely $s(t)$. Frequent choices are the tuning of the ratio according to charging/discharging conditions of the battery and vehicle accelerations.

A further step is the consideration of the last term of the objective function as a differential equation $\dot{\xi}(t) = g_1(\xi(t), \mathbf{u}(t))$. This promotes the idea of formulating the fuel-optimal operation of an hybrid vehicle over a *known cycle* as an *optimal control problem* (OCP) given as

$$\min_{\mathbf{u}(t)} \phi_{fuel}(\mathbf{u}(t)) = \int_{t_0}^{t_f} \dot{m}_{fuel}(\mathbf{u}(t)) \, \mathrm{d}t \tag{1.4}$$

subject to the battery's *state of charge* differential equation

$$\dot{\xi}(t) = g_1(\xi(t), \mathbf{u}(t)), \quad \forall t \in [t_0, t_f] \tag{1.5}$$

and then solving this problem using an appropriate numerical method.

There is a close relationship between ECMS (1.2)–(1.3) and the optimal control problem (1.4)–(1.5). The key element is to interpret the term under the integral in (1.2)–(1.3) as a Hamiltonian function, if the model is set up correspondingly. The equivalence factor for the electrical power is usually assumed to be constant. Using the Hamiltonian interpretation, the equivalence factor can be interpreted as a costate. Then, an optimal control method can be applied to determine the equivalence factor. A more rigorous explanation of the close relationship between ECMS and optimal control was described by Kim et al. [37]. Hamiltonians play an important role in optimal control as will be shown in the coming chapters.

In the majority of problem statements, the cost function will resemble fuel consumption over a test cycle but is not limited to this. From an emission point of view, the IC engine produces exhaust-gas products which might also be relevant. For instance, the minimization of the total energy of a diesel hybrid is just allowed if the increase in nitrogen oxide(s) keeps below a certain threshold. Therefore, under certain circumstances, it can be attractive to impose additional emission constraints to address some engine-out emissions. One common approach is to treat the emissions

as additional constraints to the basic problem (1.4)–(1.5)

$$\min_{\mathbf{u}(t)} \phi_{\mathrm{CO}_2}(\mathbf{u}(t)) = \int_{t_0}^{t_f} r_1 \dot{m}_{fuel}(\mathbf{u}(t))\, \mathrm{d}t$$

subject to

$$\begin{aligned}
\dot{\xi}(t) &= g_1(\xi(t), \mathbf{u}(t)) \\
\dot{m}_{\mathrm{CO}}(t) &= g_2(\mathbf{x}(t), \mathbf{u}(t)) \\
\dot{m}_{\mathrm{NO}_x}(t) &= g_3(\mathbf{x}(t), \mathbf{u}(t)) \\
\dot{m}_{\mathrm{HC}}(t) &= g_4(\mathbf{x}(t), \mathbf{u}(t)) \\
m_{\mathrm{CO}}(t) &\leq Z_1^{max} \\
m_{\mathrm{NO}_x}(t) &\leq Z_2^{max} \\
m_{\mathrm{HC}}(t) &\leq Z_3^{max} \\
\mathbf{x}(t) &= [\xi(t),\ m_{\mathrm{CO}}(t),\ m_{\mathrm{NO}_x}(t),\ m_{\mathrm{HC}}(t)]^T \\
\mathbf{z}(t) &= [m_{\mathrm{CO}}(t),\ m_{\mathrm{NO}_x}(t),\ m_{\mathrm{HC}}(t)]^T
\end{aligned}$$

where Z_1^{max}, Z_2^{max}, and Z_3^{max} are the upper bounds of the corresponding emissions: $m_{\mathrm{CO}}(\cdot)$ *carbon monoxide* (CO), $m_{\mathrm{NO}_x}(\cdot)$ nitrogen oxide(s), and $m_{\mathrm{HC}}(\cdot)$ *hydrocarbon* (HC). Carbon dioxide emissions have a direct relation to the fuel consumption by a constant conversion factor r_1. This factor expresses, how much CO_2 is emitted for every liter of fuel burned. This is known as the engine brake specific CO_2 (Stockar et al. [66]).

A major difficulty in controlling such systems which is often ignored in the literature is the fact that hybrid vehicles comprise continuous-valued controls as well as discrete controls and therefore the overall system constitutes a hybrid system and should be modeled as such. Examples are rare in the literature, noteworthy is the text of Zhu et al. [71]. Hybrid systems occur naturally in many technical applications as well as in applications from natural sciences such as biology or chemistry. Whenever a system has continuous-valued control inputs, but at the same time can make discrete decisions or switch between different subsystems, the system can be modeled as a hybrid system. A prominent example of a discrete decision is the on/off command for the internal combustion engine. More complex decisions in automotive applications are gear choices or different drive modes.

This adds new degrees of freedom in the control of the powertrain. It should be pointed out, that the term "hybrid" in "hybrid vehicle" does not necessarily refer to the existence of discrete phenomena but to the fact that at least two energy storages and converters exist. In most cases, the additional energy storage will be a high-voltage battery and the converter an electrical motor/generator.

The task of finding the controls of a hybrid system is called *hybrid optimal control problem* (HOCP). The terminology hybrid optimal control problem has been established by the control engineering community. The optimization community uses instead the terminology of *mixed-integer optimal control problem* (MIOCP) to highlight the discrete character in the optimization problem.

The solution of the optimal control problem (1.4)–(1.5) results in an open-loop control law $\mathbf{u} = \mathcal{K}$, which requires the knowledge of the test cycle a priori. From classical control theory, it is well known that open-loop controls are not robust against disturbances and model mismatch which prevents the direct application as an online strategy. Nevertheless, the solution of optimal control problems will often be very helpful in the calibration of RB control laws, as will be shown in Chap. 11 with less heuristics. Another strategy is to solve the OCP over a shortened time horizon $t_p \leq t_f$ called prediction time horizon and to retrigger the calculation if a certain threshold based upon the continuous-valued states is exceeded. Such a control strategy is closed-loop and known as *model predictive control* (MPC) (see Chap. 12). MPC has been widely used in process control because of its ability to incorporate control and state constraints. A lot of specialized solution methods are developed for MPC of hybrid systems. For example, in Passenberg et al. [52] it is demonstrated that the speed of a truck and the selected gear can be controlled using a hybrid MPC such that the fuel consumption and travel time are optimized simultaneously for maximizing the profit. Much experience has been gathered in the last decades to setup MPC strategies efficiently in automotive problems (see for instant, Fritzsche and Dünow [22], Behrendt et al. [7]), but, there are still some obstacles which makes the application to energy management a challenging task:

- the prediction time horizon for solving energy management has to be sufficiently large which requires enormous computing capacity on ECUs; and
- MPC strategies rely on accurate future information about constraints, boundary conditions, disturbances, etc. (Back [3, 4], Borhan et al. [13], and Passenberg [51]), but this information is for long prediction horizons not easy and reliable to obtain.

Nevertheless, predictive control policies have the potential to make the gap between laboratory evaluated controller tests and tests performed under real-world conditions smaller. Compared with heuristic and ECMS approaches, the use of optimal control theory can significantly reduce the calibration burden, reduce heuristics, and yield much better results under "real" optimality conditions. Consequently, a growing research area has been established, which focuses on optimal control of hybrid vehicles. Among many others, these works were performed by Paganelli et al. [50], Sciarretta [26], Liu and Peng [42], Stockar et al. [66], and Sivertsson et al. [64].

1.3.3 Numerical Solutions

The determination of optimal controls for hybrid systems is complicated by the fact that the system inputs include continuous-valued controls as well as a discrete control and because of the strong interaction between these inputs, a separate optimization will yield inferior results than a combined approach. Even though optimal control of nonhybrid systems is well researched and many powerful algorithms exist, the methods cannot readily be transferred to the hybrid context. Likewise, the well-established

methods for discrete optimization are not suitable, when the discrete decisions interact with continuous-valued controls. This has initiated a growing interest on efficient algorithms for solving HOCPs. Algorithmic development was treated, among others, in the works of Hedlund and Rantzer [27], Bengea and DeCarlo [8], Alamir and Attia [1], Shaikh [63], Sager [55], Axelsson et al. [2], and Passenberg [51].

The solution approaches of (hybrid) optimal control problems can be categorized in three main types as shown in Fig. 1.6. The assignment of the methods to the three main categories is not necessarily unique since some methods combine characteristics from several categories. Hybrid systems are characterized by various subclasses, e.g., hybrid systems with a fixed or free number of switchings and hybrid systems with autonomous or controlled transition mechanism from one discrete state to another one. Switched systems may be obtained from hybrid systems as a further subclass by neglecting the discrete dynamic behavior and instead applying the switching command directly. The distinction is important because the main development branch over the recent years has led to efficient algorithms dedicated to only one property of these subclasses. Only a few algorithms are able to handle several of these properties

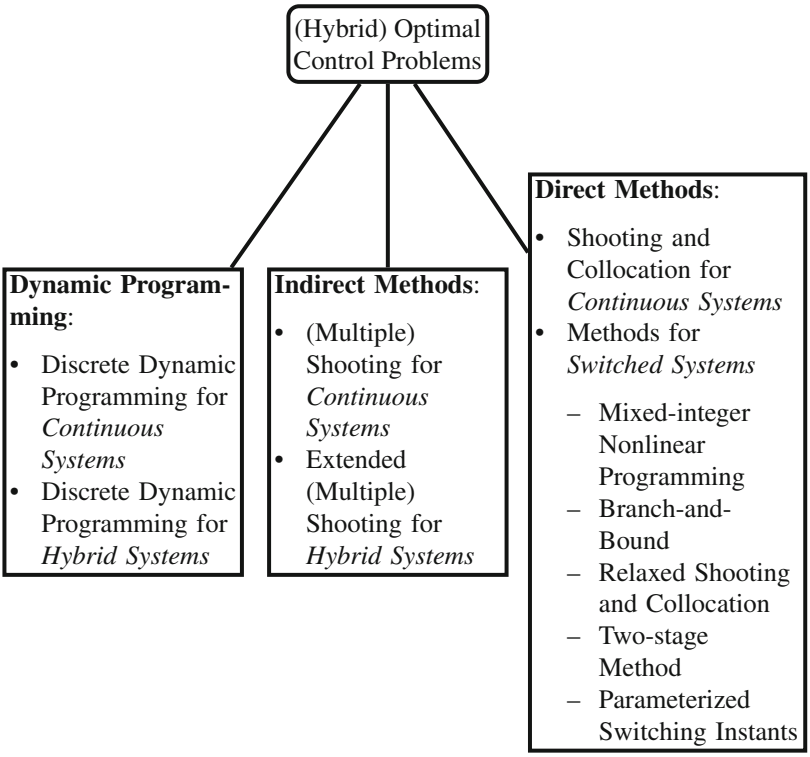


Fig. 1.6 Overview of the most prominent methods for the solution of (hybrid) optimal control problems

simultaneously. Thus, the choice of the applied algorithm must be made specifically for each hybrid system subclass and requires that the problem is well analyzed and classified.

Very accurate methods of solving optimal control problems are *indirect methods* (IM). Indirect methods solve a *multi-point boundary value problem* (MPBVP) that originates from first-order necessary conditions for local extrema that an optimal solution has to satisfy. Common methods to solve MPBVPs for purely continuous optimal control problems are gradient methods (Kirk [38], Bryson and Ho [14], and Stengel [65]) and indirect (multiple) shooting (Bock and Plitt [10], Betts [9]). Multiple shooting can be extended for hybrid systems with controlled and autonomous switching as proposed by Riedinger et al. [53], where the trajectory of the hybrid system is decomposed into a fixed number of phases (also called arcs) with constant discrete state. Indirect solution methods are described in Chap. 7.

Another solution class are *direct methods* (DM). Direct methods can be advantageous over indirect methods due to their larger *domain of convergence*, which reduces the difficulty of initialization, and their direct applicability to optimal control problems without or less knowledge of optimal control theory. Direct methods have the advantage to incorporate state constraints without requiring a predefined sequence of constrained/unconstrained arcs (von Stryk and Bulirsch [67]). They became popular to a broad class of high-dimensional optimal control problems with the progress made in nonlinear programming with sparse Quasi-Newton update rules. Algorithms for purely continuous systems can be adapted for the switched counterpart by convexification. Convexification approaches allow to cast *switched optimal control problems* (SOCP) into a much larger but continuous OCP class where the binary controls are relaxed. Specifically tailored methods for solving SOCPs are branching techniques, two-stage approaches, and mixed-integer nonlinear programming methods (Grossmann [23, 24]). These approaches are computationally very demanding.

A frequent argumentation for direct methods is the lower level of knowledge required in optimal control theory compared with indirect methods. Indeed, direct methods employ no optimality conditions directly and are therefore more easily to apply to practical problems. However, to exploit the full potential of direct methods, numerical aspects become more in focus and take a large part of the effort (see Chap. 9).

The solutions of indirect and direct methods provide only open-loop optimal controls, whereas an optimal feedback control law is obtained from the *dynamic programming* (DP) principle. This methodology can be extended for theoretically all subclasses of hybrid systems including nondifferentiable state-jumps and switching costs. However, it must be stressed that even standard problem formulations with free final time t_f cause some algorithmic challenges. Nevertheless, the DP paradigm is well recognized in academia and industry due to its fairly simple implementation and the undemanding dealing with modeling data. The latter one makes it feasible to use even nonsmoothed model data which may cause in direct and indirect methods catastrophic results due to nondifferentiabilities. This unproblematic dealing is definitely a major strength of this approach but is still only applicable for small-scale problems.

Table 1.1 Qualitative comparison of dynamic programming, direct methods, and indirect methods

Category	Dynamic programming	Direct methods	Indirect methods
Optimality of solution	Global	Local	Local
Domain of convergence	Global	Larger than IM	Smaller than DM
Ease of initialization	Good	Medium	Worse than DM
Ease of applicability	Good	Good, but worse than DP	Worse than DM
Ease of dealing with constraints	Medium	Good	Worse than DP
Accuracy	Low	Medium-high	High
Control solution	Closed-loop	Open-loop	Open-loop

Switching costs are important to penalize frequent switchings which may causes mechanical wearing. Dealing with switching costs is also possible with direct methods as shown in Chap. 8 but requires a special formulation.

Table 1.1 summarizes the main characteristics of the three methodologies.

1.4 Bibliographical Notes

Rule-based energy management strategies are discussed in Lin et al. [41], Schouten et al. [59], and Khayyam et al. [35]. Fuzzy Logic-based energy management strategies can also be classified as rule-based and are employed in Baumann et al. [6], and Farrokhi and Mohebbi [18].

The ECMS strategy has been initially proposed by Paganelli et al. [48, 50] and enhanced by Sciarretta et al. [60], Chen and Salman [16], Musardo et al. [45], Liu and Peng [42] among others. According to its low computational requirements it is implementable as online strategy. In the works of Paganelli et al. [49], Serrao et al. [62], and Jia [32], the state of charge penalty function $p(\xi(\cdot))$ were somehow different, but share the same idea: setting a higher penalty factor when the state of charge is low, in order to prevent it being over-discharged, while setting lower penalty factor when the value of the state of charge is high, in order to utilize the full potential of electric energy.

Optimal control problems are mainly solved offline. Online strategies can be implemented in the framework of model predictive control. A widely used strategy to solve OCPs is to use the *Pontryagin's minimum principle* (PMP). This approach is discussed by Rousseau et al. [54], Serrao and Rizzoni [61], Kim et al. [36], Stockar et al. [66], Kim et al. [37], and others. These works share the common approach to model the operation of a hybrid vehicle with a model similar to a quasi-steady model and reduce the problem to finding the initial value of the costate. In Serrao and Rizzoni [61], battery aging is additionally included in the problem formulation.

Model predictive control strategies are well recognized for automotive topics and have been investigated by Fritzsche and Dünow [22], Behrendt et al. [7], and Back [3].

Dynamic programming is also widely applied to solve problems with quasi-steady models, often to compare the results of ECMS/PMP with the global optimum with respect to a chosen discretization (Karbowski et al. [33], Rousseau et al. [54], and de Jager et al. [31]). In the work of Kum et al. [39], DP is used to solve a much more complex optimal control problem considering thermodynamic and emission constraints.

Some authors proposed nonlinear control techniques to solve the energy management problem. For instant, Barbarisi et al. [5] used a nonlinear decoupling strategy.

Switched or hybrid optimal control problems are reformulated to *mixed-integer nonlinear programming* (MINLP) problems and thoroughly investigated by many authors, noteworthy are Sager [55–57] and Grossmann [23–25]. The use of embedding for solving a system with two modes is encouraged by Uthaichana et al. [70]. Gear changes as well as engine starts and their respective costs are included in the two-stage algorithm proposed by Nüesch et al. [46].

References

1. Alamir M, Attia S (2004) On solving optimal control problems for switched hybrid nonlinear systems by strong variations algorithms. In: 6th IFAC symposium on nonlinear control systems (NOLCOS), Stuttgart, Germany, pp 558–563
2. Axelsson H, Wardi Y, Egerstedt M, Verriest E (2008) Gradient descent approach to optimal mode scheduling in hybrid dynamical systems. J Optim Theory Appl 136. doi:10.1007/s10957-007-9305-y
3. Back M (2005) Prädiktive Antriebsregelung zum energieoptimalen Betrieb von Hybridfahrzeugen. PhD thesis, Universität Stuttgart
4. Back M (2005) Prädiktive Antriebsregelung zum energieoptimalen Betrieb von Hybridfahrzeugen. PhD thesis, Universität Karlsruhe
5. Barbarisi O, Westervelt ER, Vasca F, Rizzoni G (2005) Power management decoupling control for a hybrid electric vehicle. In: Proceedings of the 44th decision and control conference. IEEE, pp 2012–2017
6. Baumann BM, Washington G, Glenn BC, Rizzoni G (2000) Mechatronic design and control of hybrid electric vehicles. IEEE/ASME Trans Mechatron 5(1):58–72
7. Behrendt S, Dünow P, Lampe BP (2011) An application of model predictive control to a gasoline engine. In: 18th international conference on process control, pp 57–63
8. Bengea S, DeCarlo R (2003) Optimal and suboptimal control of switching systems. In: Proceedings of the 42nd IEEE conference on decision and control, pp 5295–5300
9. Betts JT (1998) Survey of numerical methods for trajectory optimization. J Guid Control Dyn 21(2):193–207
10. Bock H, Plitt K (1984) A multiple shooting algorithm for direct solution of optimal control problems. In: Proceedings of the 9th world congress of the international federation of automatic control, vol 9
11. Boehme TJ, Held F, Schultalbers M, Lampe B (2013) Trip-based energy management for electric vehicles: an optimal control approach. Proceedings of the 2013 American control conference (ACC 2013). IEEE, Washington, pp 5998–6003

12. Boehme TJ, Schori M, Frank B, Schultalbers M, Drewelow W (2013) A predictive energy management for hybrid vehicles based on optimal control theory. Proceedings of the 2013 American control conference (ACC 2013). IEEE, Washington, pp 6004–6009
13. Borhan HA, Vahidi A, Phillips AM, Kuang ML (2009) Predictive energy management of a power-split hybrid electric vehicle. Proceedings of the 2009 American control conference, St. Louis. IEEE, pp 3970–3976
14. Bryson A, Ho YC (1975) Applied optimal control—optimization, estimation and control. Taylor & Francis Inc., New York
15. Charalampidis AC, Gillet D (2014) Speed profile optimization for vehicles crossing an intersection under a safety constraint. In: European control conference (ECC), 2014. IEEE, pp 2894–2901
16. Chen JS, Salman M (2005) Learning energy management strategy for hybrid electric vehicles. In: IEEE conference on vehicle power and propulsion, pp 68–73
17. Dinger A, Martin R, Mosquet X, Rabl M, Rizoulis D, Russo M, Sticher G (2010) Batteries for electric cars: challenges, opportunities, and the outlook to 2020
18. Farrokhi M, Mohebbi M (2005) Optimal fuzzy control of parallel hybrid electric vehicles. ICCAS 2005 June pp 2–5
19. Fellah M, Singh G, Rousseau A, Pagerit S, Nam E, Hoffman G (2009) Impact of real-world drive cycles on PHEV battery requirements. In: SAE world congress, technical Paper 2009-01-1383
20. Franco V, Kousoulidou M, Muntean M, Ntziachristos L, Hausberger S, Dilara P (2013) Road vehicle emission factors development: a review. Atmos Environ 70:84–97
21. Franco et al.(2014) Franco C, Sánchez FP, German J, Mock P (2014) Real-world exhaust emissions from modern diesel cars
22. Fritzsche C, Dünow H (2008) Advanced torque control. In: Aschemann H (ed) New approaches in automation and robotics. INTECH Open Access Publisher, pp 239–260
23. Grossmann IE (2002) Review of nonlinear mixed-integer and disjunctive programming techniques. Optim Eng 3(3):227–252
24. Grossmann IE, Kravanja Z (1993) Mixed-integer nonlinear programming: a survey of algorithms and applications. IMA Vol Math Appl 93:73–100
25. Grossmann IE, Kravanja Z (1995) Mixed-integer nonlinear programming techniques for process systems engineering. Comput Chem Eng 19:189–204
26. Guzzella L, Sciarretta A (2005) Vehicle propulsion systems. Introduction to modeling and optimization. Springer, Berlin
27. Hedlund S, Rantzer A (2002) Convex dynamic programming for hybrid systems. IEEE Trans Autom Control 47:1536–1540
28. IEA (2009) Transport, energy and CO2. Technical report, International Energy Agency
29. IEA (2014) Monitoring CO2 emissions from passenger cars and vans in 2013. Technical report, International Energy Agency, technical report 19/2014. doi:10.2800/23352
30. IEA (2015) Energy and climate change. Technical report, International Energy Agency
31. de Jager B, van Keulen T, Kessels J (2013) Optimal control of hybrid vehicles. Springer
32. Jia Y (2014) Powertrain design for power split hybrid vehicles with global optimization procedures applied for test cycles. Master's thesis, Universität RWTH Aachen
33. Karbowski D, Rousseau A, Pagerit S, Sharer P (2006) Plug-in vehicle control strategy: from global optimization to real time application. In: 22nd electric vehicle symposium, EVS22, Yokohama, Japan
34. van Keulen T, Naus G, de Jager B, van de Molengraft R, Steinbuch M, Aneke E (2009) Predictive cruise control in hybrid electric vehicles. World Electr Veh J 3(1)
35. Khayyam H, Kouzani A, Nahavandi S, Marano V, Rizzoni G (2010) Intelligent energy management in hybrid electric vehicles. InTech
36. Kim N, Lee D, Cha SW, Peng H (2009) Optimal control of a plug-in hybrid electric vehicle (PHEV) based on driving patterns. In: International battery, hybrid and fuel cell electric vehicle symposium
37. Kim N, Cha S, Peng H (2011) Optimal control of hybrid electric vehicles based on Pontryagin's minimum principal. IEEE Trans Control Syst Technol 1279–1287

38. Kirk D (1970) Optimal control theory: an introduction. Englewood Cliffs, Prentice-Hall
39. Kum D, Peng H, Bucknor N (2011) Supervisory control of parallel hybrid electric vehicles for fuel and emission reduction. ASME J Dyn Syst Meas Control 133(6):4498–4503
40. Kwon J, Kim J, Fallas E, Pagerit S, Rousseau A (2008) Impact of drive cycles on PHEV component requirements. In: SAE world congress, technical Paper 2008-01-1337
41. Lin CC, Kang JM, Grizzle JW, Peng H (2001) Energy management strategy for a parallel hybrid electric truck. In: Proceedings of the 2001 American control conference, Arlington, vol 4, pp 2878–2883
42. Liu J, Peng H (2006) Control optimization for a power-split hybrid vehicle. In: Proceedings of the 2006 American control conference, Minneapolis, pp 466–471
43. Liu S, De Schutter B, Hellendoorn H (2014) Integrated traffic flow and emission control based on FASTLANE and the multi-class VT-macro model. In: European control conference (ECC), 2014. IEEE, pp 2908–2913
44. Mellios G, Hausberger S, Keller M, Samaras C, Ntziachristos L, Dilara P, Fontaras G (2011) Parameterisation of fuel consumption and CO2 emissions of passenger cars and light commercial vehicles for modelling purposes. Technical report, Joint Research Centre (JRC) Science and Technical Reports. doi:10.2788/58009
45. Musardo C, Rizzoni G, Staccia B (2005) A-ECMS: an adaptive algorithm for hybrid electric vehicle energy management. In: Proceedings of the 44th IEEE conference on decision and control, pp 1816–1823
46. Nüesch T, Elbert P, Flankl M, Onder C, Guzzella L (2014) Convex optimization for the energy management of hybrid electric vehicles considering engine start and gearshift costs. Energies 7:834–856
47. Nykvist B, Nilsson M (2015) Rapidly falling costs of battery packs for electric vehicles. Nat Clim Change 5(4):329–332. doi:10.1038/NCLIMATE2564
48. Paganelli G, Ercole G, Brahma A, Guezennec Y, Rizzoni G (2001) General supervisory control policy for the energy optimization of charge-sustaining hybrid electric vehicles. JSAE Rev 22(4):511–518
49. Paganelli G, Tateno M, Brahma A, Rizzoni G, Guezennec Y (2001) Control development for a hybrid-electric sport-utility vehicle: strategy, implementation and field test results. In: Proceedings of the 2001 American control conference (ACC), vol 6. IEEE, pp 5064–5069
50. Paganelli G, Guezennec Y, Rizzoni G (2002) Optimizing control strategy for hybrid fuel cell vehicle. In: SAE world congress, technical Paper 2002-01-0102. doi:10.4271/2002-01-0102
51. Passenberg B (2012) Theory and algorithms for indirect methods in optimal control of hybrid systems. PhD thesis, Technischen Universität München
52. Passenberg B, Kock P, Stursberg O (2009) Combined time and fuel optimal driving of trucks based on a hybrid model. In: European control conference (ECC). IEEE, pp 4955–4960
53. Riedinger P, Daafouz J, Iung C (2005) About solving hybrid optimal control problems. IMACS05
54. Rousseau G, Sinoquet D, Rouchon P (2007) Constrained optimization of energy management for a mild-hybrid vehicle. Oil Gas Sci Technol Rev IFP 62(4):623–634
55. Sager S (2005) Numerical methods for mixed-integer optimal control problems. PhD thesis, Universität Heidelberg
56. Sager S, Bock HG, Reinelt G (2009) Direct methods with maximal lower bound for mixed-integer optimal control problems. Math Program 118(1):109–149
57. Sager S, Claeys M, Messine F (2014) Efficient upper and lower bounds for global mixed-integer optimal control. J Global Optim 61(4):721–743
58. Schori M, Boehme TJ, Frank B, Lampe B (2014) Optimal calibration of map-based energy management for plug-in parallel hybrid configurations: a hybrid optimal control approach. IEEE Trans Veh Technol 64(9):3897–3907. doi:10.1109/TVT.2014.2363877
59. Schouten NJ, Salman MA, Kheir NA (2003) Energy management strategies for parallel hybrid vehicles using fuzzy logic. Control Eng Pract 11:171–177
60. Sciarretta A, Back M, Guzzella L (2004) Optimal control of parallel hybrid electric vehicles. IEEE Trans Control Systems Technol 12(3):352–363

61. Serrao L, Rizzoni G (2008) Optimal control of power split for a hybrid refuse vehicle. In: Proceedings of the American control conference. IEEE, pp 4498–4503
62. Serrao L, Onori S, Rizzoni G (2011) A comparative analysis of energy management strategies for hybrid electric vehicles. J Dyn Syst Meas Control 133(3):031,012. doi:10.1115/1.4003267
63. Shaikh MS (2004) Optimal control of hybrid systems: Theory and algorithms. PhD thesis, Department of Electrical and Computer Engineering, McGill University, Montreal
64. Sundstroem S, Eriksson SM, Sundstroem C, Eriksson L (2011) Adaptive control of a hybrid powertrain with map-based ecms. In: Proceedings of the IFAC world congress, pp 357–362
65. Stengel RF (1994) Optimal control and estimation. Dover Publications
66. Stockar S, Marano V, Canova M, Rizzoni G, Guzzella L (2011) Energy-optimal control of plug-in hybrid electric vehicles for real-world driving cycles. IEEE Trans Veh Control 60(7):2949–2962
67. von Stryk O, Bulirsch R (1992) Direct and indirect methods for trajectory optimization. Ann Oper Res 37:357–373
68. Tanaka N et al (2011) Technology roadmap: eectric and plug-in hybrid electric vehicles. Technical report, International Energy Agency
69. Trigg T, Telleen P, Boyd R, Cuenot F, DAmbrosio D, Gaghen R, Gagné J, Hardcastle A, Houssin D, Jones A, et al. (2013) Global EV outlook: understanding the electric vehicle landscape to 2020. Technical report, International Energy Agency
70. Uthaichana K, Bengea S, Decarlo R, Pekarek S, Zefran M (2008) Hybrid model predictive control tracking of a sawtooth driving profile for an HEV. In: American control conference, 2008. IEEE, pp 967–974
71. Zhu Y, Chen Y, Tian G, Wu H, Chen Q (2004) A four-step method to design an energy management strategy for hybrid vehicles. In: Proceedings of the 2004 American control conference, vol 1. IEEE, pp 156–161

Part I
Theory and Formulations

Chapter 2
Introduction to Nonlinear Programming

2.1 Introduction

In this chapter, we review some important theory on mathematical optimization (also known as mathematical programming) which provides direct motivation for some numerical algorithms. However, highly efficient algorithms must also account for particular properties and structures of the problems. There are many important special cases for which specialized algorithms are available. Figure 2.1 depicts some of them.

The central problem of such mathematical programming problems is that of minimizing or maximizing a given function of a finite number of variables subject to a finite set of equality and/or inequality constraints. In the first part of this chapter we concentrate on *nonlinear programming* (NLP) which can be viewed as a part of mathematical optimization. Nonlinear programming deals with optimization problems, where the objective function or some of the constraints are nonlinear. This contrasts with: linear programming, which frames algorithms for the solution of optimization problems with linear objectives and constraints; quadratic programming, which frames algorithms for the solution of optimization problems with quadratic objective and linear constraints; and convex programming, which frames algorithms for the solution of optimization problems with convex objective and concave inequality constraints. During the course of this chapter we will see that nonlinear programming also makes use of quadratic programming solution techniques.

In the second part the nonlinear programming problem formulation is extended by a disturbance parameter. The study of the influence of this disturbance parameter on the optimal solution introduces the concept of sensitivity, which motivates some advanced algorithmic extensions.

The last part of this chapter extends the single-objective formulations to multi-objective formulations.

We begin our discussion by introducing some definitions regarding rates of convergence of iterative solvers. These definitions are quite helpful in giving us a metric

T.J. Böhme and B. Frank, *Hybrid Systems, Optimal Control and Hybrid Vehicles*,
Advances in Industrial Control, DOI 10.1007/978-3-319-51317-1_2

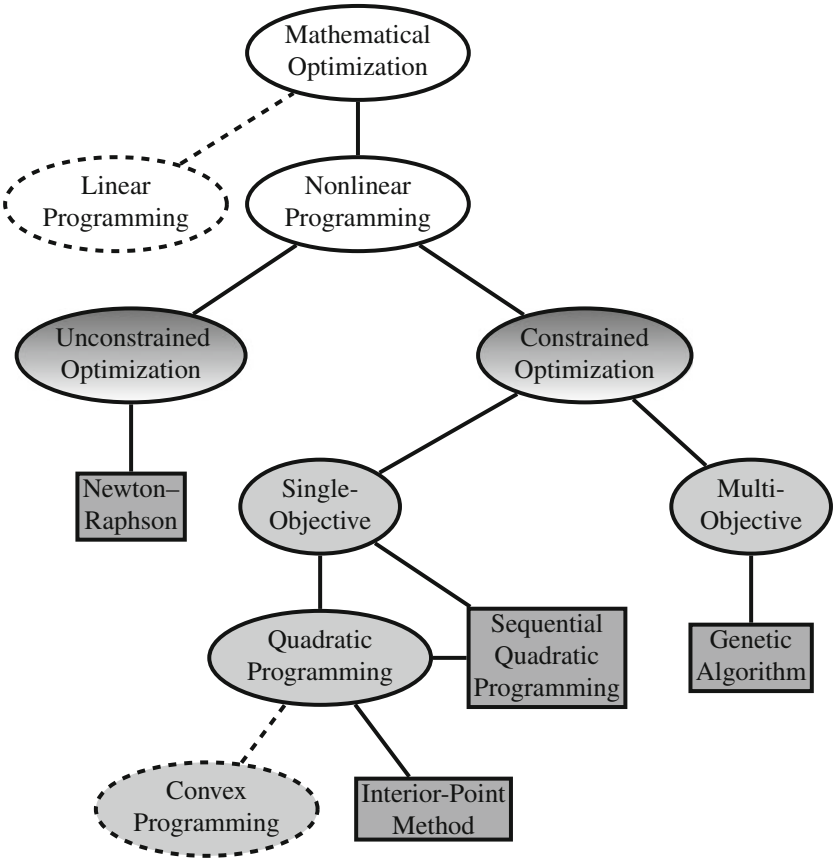


Fig. 2.1 Classes and methods of numerical optimization. *Elliptical nodes* indicate optimization classes; *rectangular nodes* indicate optimization methods; and *dashed nodes* indicate optimization classes not covered in this book

to assess algorithms for solving nonlinear problems. The following results can be found in Ortega and Rheinboldt [62].

Definition 2.1 (*Q-linear Convergence*) A sequence $\{\mathbf{y}_i\} \subset \mathbb{R}^{N_y}$ converging to a fix point $\{\mathbf{y}^*\}$ is said to converge Q-linearly if a constant $0 < c_{ql} < 1$ exists such that

$$\| \mathbf{y}_{i+1} - \mathbf{y}^* \| \leq c_{ql} \| \mathbf{y}_i - \mathbf{y}^* \|$$

holds for all indices i that are sufficiently large.

△

Definition 2.2 (*Q-superlinear Convergence*) The sequence is said to converge Q-superlinearly if a sequence of positive factors $c_{qs,i} \to 0$ exists such that

$$\| \mathbf{y}_{i+1} - \mathbf{y}^* \| \leq c_{qs,i} \| \mathbf{y}_i - \mathbf{y}^* \|$$

holds for all indices i that are sufficiently large.

△

Definition 2.3 (*Q-quadratic Convergence*) Finally, the sequence is said to converge Q-quadratically if a constant $0 < c_{qq} < 1$ exists such that

$$\| \mathbf{y}_{i+1} - \mathbf{y}^* \| \leq c_{qq} \| \mathbf{y}_i - \mathbf{y}^* \|^2$$

holds for all indices i that are sufficiently large.

△

These convergence rates are called *Q-rates*, because the convergence factor c_{qx} in the definitions above is a quotient. Similarly defined are the *R-rates* for which the convergence factor c_{rx} is a root. If in one of three Definitions 2.1, 2.2, and 2.3, the index $i+1$ on the left-hand side is replaced by $i+m$ the corresponding convergence rate is denoted as the m-step convergence rate.

The R-rates are defined as:

Definition 2.4 (*R-linear Convergence*) A sequence $\{\mathbf{y}_i\} \subset \mathbb{R}^{N_y}$ converging to a fix point $\{\mathbf{y}^*\}$ is said to converge R-linearly if a constant $0 < c_{rl} < 1$ exists such that

$$\sqrt[i]{\| \mathbf{y}_i - \mathbf{y}^* \|} \leq c_{rl}$$

holds for all indices i that are sufficiently large.

△

Definition 2.5 (*R-superlinear Convergence*) The sequence is said to converge R-superlinearly if a sequence of positive factors $c_{rs,i} \to 0$ exists such that

$$\sqrt[i]{\| \mathbf{y}_i - \mathbf{y}^* \|} \leq c_{rs,i}$$

holds for all indices i that are sufficiently large.

△

Definition 2.6 (*R-quadratic Convergence*) The sequence is said to converge R-quadratically if a constant $0 < c_{rq} < 1$ exists such that

$$\sqrt[2^i]{\| \mathbf{y}_i - \mathbf{y}^* \|} \leq c_{rq}$$

holds for all indices i that are sufficiently large.

△

For linear and superlinear convergence the R-convergence rates are always smaller than the Q-convergence rates. Therefore, the R-linear convergence is generally slower than the Q-linear convergence; the same applies for superlinear convergence. For quadratic convergence this relation does not hold.

Most convergence analyses of optimization algorithms are concerned with Q-convergence.

2.2 Unconstrained Nonlinear Optimization

We begin our short introduction to numerical optimization with the fundamentals of unconstrained optimization. These are needed to understand the more complex algorithms for constrained optimization, which are later used to solve the optimal control problems arising from the problem formulations.

The simplest nonlinear programming problem is that of minimizing or maximizing a function $f : \mathbb{R}^{N_y} \to \mathbb{R}$. We restrict our analysis only to the minimum problem, as the problem $\min f(\mathbf{y})$ is equivalent to the problem $\max(-f(\mathbf{y}))$.

Definition 2.7 (*Unconstrained Nonlinear Programming Problem*) An *unconstrained nonlinear programming problem* (also known as a *free mathematical programming problem*) is given by

$$\min_{\mathbf{y}\in\mathbb{R}^{N_y}} f(\mathbf{y}) \tag{2.1}$$

where $f : \mathbb{R}^{N_y} \to \mathbb{R}$ is a smooth and real-valued *objective function* of the vector $\mathbf{y} \in \mathbb{R}^{N_y}$.

△

The vector

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{N_y} \end{bmatrix}$$

contains all the *decision variables* of the unconstrained NLP.

Definition 2.8 (*Extremals*) A point $\mathbf{y}^* \in \mathbb{R}^{N_y}$ is called a *local minimum* of (2.1), if a local region U_ϵ with $\epsilon > 0$ exists such that

$$f(\mathbf{y}^*) \leq f(\mathbf{y}) \tag{2.2}$$

is satisfied $\forall \mathbf{y} \in U_\epsilon(\mathbf{y}^*)$. If instead of (2.2),

$$f(\mathbf{y}^*) < f(\mathbf{y}) \tag{2.3}$$

applies, then $\mathbf{y}^*$ is called a *strict local minimum* of (2.1). For the case that (2.2) is satisfied $\forall \mathbf{y} \in \mathbb{R}^{N_y}$, $\mathbf{y}^*$ is a *global minimum* of (2.1). If (2.3) also holds for a global minimum, then (2.1) has an *unique global minimum*.

△

In the remainder of this chapter it is assumed that f is twice continuously differentiable on $\mathbb{R}^{N_y}$ with respect to $\mathbf{y}$, so that the Hessian for $f(\cdot)$ is defined.

2.2.1 Necessary and Sufficient Conditions for Optimality

Every local minimum $\mathbf{y}^*$ of Problem (2.1) satisfies the first-order necessary condition, which is the vanishing of the gradient of $f(\mathbf{y}^*)$ (Fermat's condition)

$$\nabla f(\mathbf{y}^*) = \mathbf{0} \tag{2.4}$$

and the second-order necessary condition

$$\mathbf{v}^T \nabla^2 f(\mathbf{y}^*)\mathbf{v} \geq 0, \quad \forall \mathbf{v} \in \mathbb{R}^{N_y}, \mathbf{v} \neq \mathbf{0}, \tag{2.5}$$

which means that the Hessian $\nabla^2 f(\mathbf{y}^*)$ must be positive semi-definite.

But these necessary conditions can also apply to local maxima or saddle points. So, a point that satisfies these conditions is only guaranteed to be a *stationary or critical point* of Problem (2.1).

If in addition to the necessary conditions the second-order sufficient condition for the point $\mathbf{y}^*$ holds that $f(\mathbf{y}^*)$ has a positive definite Hessian (i.e., $f(\mathbf{y}^*)$ is locally strict convex)

$$\mathbf{v}^T \nabla^2 f(\mathbf{y}^*)\mathbf{v} > 0, \quad \forall \mathbf{v} \in \mathbb{R}^{N_y}, \mathbf{v} \neq \mathbf{0},$$

then the point $\mathbf{y}^*$ is guaranteed to be a local minimum and is called *strict local minimum*.

In summary, every local minimum satisfies the two necessary conditions, but must not necessarily satisfy the sufficient condition. Vice versa, every point that satisfies the necessary conditions and the sufficient condition is guaranteed to be a local minimum.

2.2.2 Newton–Raphson Method

To find a minimum of (2.1) the system of equations (2.4), which is generally nonlinear, must be solved. A commonly used method for this problem is the Newton–Raphson

method. Treating the current point $\mathbf{y}$ as fixed and introducing a new variable $\mathbf{d}$ as deviation from $\mathbf{y}$, the idea now is to construct an iterative procedure by using a linear Taylor approximation of $\nabla f(\mathbf{y}+\mathbf{d})$ around the point $\mathbf{y}$

$$\nabla f(\mathbf{y}+\mathbf{d}) \approx \nabla f(\mathbf{y}) + \nabla^2 f(\mathbf{y})\mathbf{d} \tag{2.6}$$

where $\mathbf{d}$ denotes the *search direction*,

$$\nabla f(\mathbf{y}) = \begin{bmatrix} \frac{\partial f}{\partial y_1} \\ \vdots \\ \frac{\partial f}{\partial y_{N_y}} \end{bmatrix}$$

is the N_y-dimensional gradient vector with respect to $\mathbf{y}$, and

$$\nabla^2 f(\mathbf{y}) = \begin{pmatrix} \frac{\partial^2 f}{\partial y_1^2} & \frac{\partial^2 f}{\partial y_1 \partial y_2} & \cdots & \frac{\partial^2 f}{\partial y_1 \partial y_{N_y}} \\ \frac{\partial^2 f}{\partial y_2 \partial y_1} & \frac{\partial^2 f}{\partial y_2^2} & \cdots & \frac{\partial^2 f}{\partial y_2 \partial y_{N_y}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial y_{N_y} \partial y_1} & \frac{\partial^2 f}{\partial y_n \partial y_2} & \cdots & \frac{\partial^2 f}{\partial y_{N_y}^2} \end{pmatrix}$$

is the symmetric $N_y \times N_y$ *Hessian matrix*.

By simply setting (2.6) to zero, one obtains the Newton's search direction with

$$\mathbf{d} = -(\nabla^2 f(\mathbf{y}))^{-1} \nabla f(\mathbf{y}). \tag{2.7}$$

Using the Newton's search direction (2.7), we can generate a sequence $\{\mathbf{y}_k\}_{k\in\mathbb{N}}$ by applying the update rule

$$\mathbf{y}_{k+1} = \mathbf{y}_k + \mathbf{d}_k. \tag{2.8}$$

For the sake of notational simplicity we introduce the following abbreviations $\nabla f_k := \nabla f(\mathbf{y}_k)$ and $\nabla^2 f_k := \nabla^2 f(\mathbf{y}_k)$.

The proof and conditions for local convergence of the method to a stationary point $\mathbf{y}^*$ were first stated in a general form by Kantorovich [46]. We use an adapted formulation of the Kantorovich's theorem for the problem (2.1), similar to the statement in Ferreira and Svaiter [25].

Theorem 2.1 (Kantorovich's Theorem) *Let $f(\cdot)$ be twice continuously differentiable on $int(\mathbf{C})$ (the interior of $\mathbf{C}$) with respect to $\mathbf{y}$ where $\mathbf{C} \subseteq \mathbb{R}^{N_y}$, $\mathbf{y}_0 \in int(\mathbf{C})$, $L > 0$, $\omega > 0$ and suppose that*

1. *$\nabla^2 f_0$ is nonsingular;*
2. $\| \left(\nabla^2 f_0\right)^{-1} [\nabla^2 f_1 - \nabla^2 f_2] \| \leq L \| \mathbf{y}_1 - \mathbf{y}_2 \|, \quad \forall \mathbf{y}_1, \mathbf{y}_2 \in \mathbf{C}$;

3. $\| \left(\nabla^2 f_0\right)^{-1} \nabla f_0 \| \leq \omega$; *and*
4. $\omega L \leq \dfrac{1}{2}$

holds. Define the radii

$$\mathbf{r}_1 := \frac{1 - \sqrt{1 - 2\omega L}}{L}$$

and

$$\mathbf{r}_2 := \frac{1 + \sqrt{1 - 2\omega L}}{L}.$$

If there exists a ball $B[\mathbf{y}_0, \mathbf{r}_1] \subset \mathbf{C}$*, then the sequence* $\{\mathbf{y}_k\}_{k \in \mathbb{N}}$ *generated by the Newton–Raphson method with the starting point* $\mathbf{y}_0$ *is contained in* $B(\mathbf{y}_0, \mathbf{r}_1)$*, converges to a unique zero of* $\mathbf{y}^*$ *(2.4) in* $B[\mathbf{y}_0, \mathbf{r}_1]$ *and*

$$\| \mathbf{y}^* - \mathbf{y}_{k+1} \| \leq \frac{1}{2} \| \mathbf{y}^* - \mathbf{y}_k \|, \quad \forall k \in \mathbb{N}.$$

If $\omega L < 1/2$ *also holds, then the sequence* $\{\mathbf{y}_k\}_{k \in \mathbb{N}}$ *converges Q-quadratically with*

$$\| \mathbf{y}^* - \mathbf{y}_{k+1} \| \leq \frac{L}{2\sqrt{1 - 2\omega L}} \| \mathbf{y}^* - \mathbf{y}_k \|^2, \quad \forall k \in \mathbb{N}$$

and $\mathbf{y}^*$ *is the unique zero of (2.4) in* $B[\mathbf{y}_0, \boldsymbol{\rho}]$ *for any* $\boldsymbol{\rho}$ *such that*

$$\mathbf{r}_1 \leq \boldsymbol{\rho} < \mathbf{r}_2 \text{ and } B[\mathbf{y}_0, \boldsymbol{\rho}] \subset \mathbf{C}.$$

Proof Proofs of this theorem can be found in Kantorovich [46, 47], and Ortega [61]. □

Under the assumptions (1–3), the theorem of Kantorovich gives sufficient conditions for Q-linear convergence and, with the additional assumption $\omega L < 1/2$, for Q-quadratic convergence to a stationary point $\mathbf{y}^*$; However, it neither guarantees that the method converges to a local minimum nor that the method converges at all if the starting point is not in the neighborhood of a stationary point.

Even though the theorem of Kantorovich does not explicitly state the radius r_c, for which the sequence $\{\mathbf{y}_k\}_{k \in \mathbb{N}}$ with $\| \mathbf{y}_k - \mathbf{y}^* \| \leq r_c$ is guaranteed to converge to $\mathbf{y}^*$, it does guarantee the existence of $r_c > 0$. r_c is denoted as the *convergence radius* of the Newton–Raphson method and $B[\mathbf{y}^*, r_c]$ as the *sphere of convergence*.

If the starting point $\mathbf{y}_0$ lies inside a sphere of convergence and the Hessian $\nabla^2 f_0$ is positive definite, the theorem of Kantorovich guarantees the convergence to a local minimum, because for a positive definite Hessian $\nabla^2 f$ the inverse is also positive definite. Then, one can state that

$$\nabla f(\mathbf{y})^T (\nabla^2 f(\mathbf{y}))^{-1} \nabla f(\mathbf{y}) > 0, \tag{2.9}$$

and it follows with Equation (2.7) that

$$\mathbf{d}^T \nabla f(\mathbf{y}) < 0 \tag{2.10}$$

which is called a *descent condition* and this guarantees that every step of the Newton–Raphson method decreases the function value of (2.1).

In this case, the Hessian $\nabla^2 f_k$ keeps positive definite for all further iteration steps and therefore the sufficient condition (2.5) for a local minimum is satisfied at the point of convergence $\mathbf{y}^*$.

If the starting point lies inside the sphere of convergence and the Hessian is not positive definite the Newton–Raphson method converge to a stationary point, which can be local minimum, a local maximum, or a saddle point. If the method does not converge to a local minimum, the Newton–Raphson iteration must be restarted with another appropriate starting point or a globalization strategy must be used to be able to find a local minimum.

2.2.3 Globalization of the Newton–Raphson Method

In the previous section, we have seen that if the starting point does not lie inside the sphere of convergence of a local minimum, one obtains no guarantee of convergence. It is therefore an obvious extension of the Newton–Raphson method to demand a minimum decrease of the function values for every iteration to establish global convergence to a stationary point and it is hoped to a local minimum

$$f(\mathbf{y}_{k+1}) < f(\mathbf{y}_k). \tag{2.11}$$

This condition guarantees a monotonously decreasing sequence of function values and aims to reach a sphere of convergence in a finite number of iterations, if the function has a bound on the minimum value.

There are two main types of algorithms that aim at finding a sequence of points that satisfy Condition (2.11) at every iteration: line-search methods and trust-region methods. We will focus on line-search methods only because these methods perform very well for the class of problems discussed in this book.

By using line-search methods we satisfy Condition (2.11) for every iteration by enforcing that the search direction $\mathbf{d}_k$ is a descent direction (2.10) and adding a step-size $\alpha_k \in (0, 1]$ to the iteration rule (2.8) which is then called the line-search iteration rule

$$\mathbf{y}_{k+1} = \mathbf{y}_k + \alpha_k \mathbf{d}_k.$$

Inside a sphere of convergence, the Newton–Raphson method guarantees convergence to a stationary point with the step-size $\alpha_k = 1$. Outside a sphere of convergence α_k can be chosen small enough to satisfy Condition (2.11) because $\mathbf{d}_k$ is a descent

direction. This modification of the Newton–Raphson method is known as *damped Newton–Raphson method*.

The next subsections describe how to choose an adequate step-size and how to enforce the search direction $\mathbf{d}_k$ to be a descent direction.

2.2.3.1 Step-Size Algorithms

An appropriate step-size should at least satisfy Condition (2.11) for the next iterate. Consequently, the descent condition is defined by

$$f(\mathbf{y}_k + \alpha_k \mathbf{d}_k) < f(\mathbf{y}_k). \tag{2.12}$$

If only the descent condition is considered for calculating a step-size, it might be possible that the convergence becomes very slow due to small decreases in function values or that the algorithm does not converge at all.

This requires a modification of (2.12) with a factor $\epsilon \in [0, 1)$. We obtain then the condition

$$f(\mathbf{y}_k + \alpha_k \mathbf{d}_k) < f(\mathbf{y}_k) + \epsilon \alpha_k \nabla f_k^T \mathbf{d}_k. \tag{2.13}$$

This condition was proposed by Armijo [2] in 1966 to simplify the application of deepest descent algorithms and is up to now a major contribution in determining a proper step-size. Thus, Condition (2.13) is known as the *Armijo condition*.

If the factor ϵ is large enough, then the additional term enforces a reduction in $f(\cdot)$ proportional to both the step-size α_k and the directional derivative $\mathbf{g}_k^T \mathbf{d}_k$. Obviously, for $\epsilon = 0$ the Armijo condition is equal to the descent condition (2.12) and this procedure can still lead to very small step-sizes.

To avoid this drawback Wolfe (cf. [71] and [72]) introduced the additional condition

$$\nabla f(\mathbf{y}_k + \alpha_k \mathbf{d}_k)^T \mathbf{d}_k \geq \rho \cdot \nabla f_k^T \mathbf{d}_k \tag{2.14}$$

and restricted the parameters to $\epsilon \in [0, 0.5)$ and $\rho \in (\epsilon, 1)$, which limits the decrease of the step-size to an infimum. Conditions (2.13) and (2.14) together are called *Wolfe conditions*.

But a step-size, which satisfies these conditions, must still not be close to an optimal step-size. Therefore, the condition can be strengthened by the modification

$$\left| \nabla f(\mathbf{y}_k + \alpha_k \mathbf{d}_k)^T \mathbf{d}_k \right| \leq \rho \cdot \left| \nabla f_k^T \mathbf{d}_k \right|,$$

which is then called together with (2.13) *strong Wolfe conditions*.

It remains to clarify how a step-size, which satisfies either the Armijo condition, the Wolfe conditions, or the strong Wolfe conditions can be calculated. A step-size for the Armijo condition can be calculated in a fairly simple way because it is automatically satisfied if the step-size is small enough. For this task the basic backtracking algorithm can be used.

The basic backtracking algorithm described in Nocedal and Wright [59] determines the smallest $j \in \mathbb{N}_{\geq 0}$ with $\beta \in (0, 1)$ such that for $\alpha_k = \beta^j$ Condition (2.12) is satisfied. The usual way is to search for the first index j that satisfies the descent condition (2.12) beginning with $j = 0$.

For the (strong) Wolfe conditions the backtracking algorithm needs not to terminate, because very small step-sizes are not acceptable. Instead, the line-search algorithm described in Moré and Thuente [57] can be used.

2.2.3.2 Descent Search Direction

If the Hessian is positive definite then the Newton search direction (2.7) is a descent direction (see (2.9) and (2.10)). In general, one can expect a positive definite Hessian only in the neighborhood of a local minimum. From a global scope it is unlikely that the Hessian $\nabla^2 f_k$ is positive definite for all iterates $\mathbf{y}_k$. In order to enforce a descent search direction one can replace the Hessian $\nabla^2 f_k$ with any other positive definite matrix $\mathbf{B}_k$ in Condition (2.10) and calculate the search direction thereby

$$\mathbf{d}_k = -\mathbf{B}_k^{-1} \nabla f_k. \tag{2.15}$$

If one chooses the unity matrix $\mathbf{B}_k = \mathbf{I}$, the search direction $\mathbf{d}_k$ is exactly the steepest descent search direction.

This makes the strategy clear, if the iterates of the minimization process have not reached a sphere of convergence with a positive definite Hessian, the search direction (2.15) is determined by applying a modified matrix $\mathbf{B}_k$ and an adequate step-size until a sphere of convergence with a positive definite Hessian is reached. Hereafter, the Newton search direction with a step-size of 1 is used to exploit the fast quadratic convergence from the Newton–Raphson method.

A common method to realize a blending between a positive definite matrix and the exact Hessian is to apply a scaled updating with

$$\mathbf{B}_k = \kappa_k \mathbf{I} + \nabla^2 f_k$$

where κ_k must be chosen large enough to ensure positive definiteness of $\mathbf{B}_k$. With increasing k the weighting factor κ_k must tend to zero such that inside a sphere of convergence the exact Hessian can be used. To guarantee $\mathbf{B}_k$ to be positive definite κ_k must be chosen larger than the absolute value of the most negative Eigenvalue of $\nabla^2 f_k$. Betts [7] assumed the choice of

$$\kappa_k = \tau_k \cdot (|\sigma_k| + 1)$$

with the weighting factor $\tau_k \in [0, 1]$ and the *Gershgorin bound* for the most negative Eigenvalue

$$\sigma_k = \min_{1 \le i \le N_y} \left\{ h_{ii} - \sum_{i \neq j}^{N_y} |h_{ij}| \right\}.$$

Here, h_{ij} denotes the element of the i-th row and j-th column of the Hessian $\nabla^2 f_k$.

The choice of the weighting factor τ_k essentially affects the performance of the minimization process; in the neighborhood of a local minimum it should be chosen as small as possible. Therefore, one can use the backtracking algorithm to find the smallest τ_k for which the matrix $\mathbf{B}_k$ is still positive definite. An effective algorithm to check the positive definiteness of $\mathbf{B}_k$ is presented in Chap. 9.

2.2.4 Quasi-Newton Method

For the methods for nonlinear unconstrained optimization previously introduced the calculation of the Hessian in every iteration is needed. Because this calculation is numerically an expensive task and in general not beneficial outside a sphere of convergence, one can try to approximate the Hessian or the inverse of the Hessian with an update rule, which only requires first-order derivatives of the objective function.

The approximation of the Hessian is denoted by $\mathbf{B}_k$ and the approximation of the inverse Hessian is denoted by $\mathbf{H}_k$ for every iteration number k.

Davidon first introduced the *variable metric method* in 1966, which is known today as Quasi-Newton method [17]. Davidon used the *symmetrical rank 1* (SR1) update

$$\mathbf{H}_{k+1} = \mathbf{H}_k + \frac{\left(\boldsymbol{\delta}_k - \mathbf{H}_k \boldsymbol{\gamma}_k\right)\left(\boldsymbol{\delta}_k - \mathbf{H}_k \boldsymbol{\gamma}_k\right)^T}{\left(\boldsymbol{\delta}_k - \mathbf{H}_k \boldsymbol{\gamma}_k\right)^T \boldsymbol{\gamma}_k}$$

and the rank 2 update

$$\mathbf{H}_{k+1} = \mathbf{H}_k + \frac{\boldsymbol{\delta}_k \boldsymbol{\delta}_k^T}{\boldsymbol{\delta}_k^T \boldsymbol{\gamma}_k} - \frac{\mathbf{H}_k \boldsymbol{\gamma}_k \boldsymbol{\gamma}_k^T \mathbf{H}_k}{\boldsymbol{\gamma}_k^T \mathbf{H}_k \boldsymbol{\gamma}_k} \tag{2.16}$$

to approximate the inverse of the Hessian. Herein, $\boldsymbol{\delta}_k = \mathbf{y}_{k+1} - \mathbf{y}_k$ is the change made in $\mathbf{y}$ and $\boldsymbol{\gamma}_k = \nabla f_{k+1} - \nabla f_k$ the change made in the gradient $\nabla f(\mathbf{y})$ at the k-th iteration. The advantage of the approximation of the inverse of the Hessian is that no linear system of equations need to be solved to calculate the search direction. Instead, a simple matrix–vector multiplication is used. A proof of convergence of this method is given by Fletcher and Powell [31]. The update process (2.16) is known as the DFP update formula (named by Davidon, Fletcher, and Powell).

The SR1 formula for the approximation of the Hessian is

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \frac{\left(\boldsymbol{\gamma}_k - \mathbf{B}_k\boldsymbol{\delta}_k\right)\left(\boldsymbol{\gamma}_k - \mathbf{B}_k\boldsymbol{\delta}_k\right)^T}{\left(\boldsymbol{\gamma}_k - \mathbf{B}_k\boldsymbol{\delta}_k\right)^T\boldsymbol{\delta}_k}$$

and the DFP formula is

$$\mathbf{B}_{k+1} = \left(\mathbf{I} - \frac{\boldsymbol{\gamma}_k\boldsymbol{\delta}_k^T}{\boldsymbol{\delta}_k^T\boldsymbol{\gamma}_k}\right)\cdot\mathbf{B}_k\cdot\left(\mathbf{I} - \frac{\boldsymbol{\delta}_k\boldsymbol{\gamma}_k^T}{\boldsymbol{\delta}_k^T\boldsymbol{\gamma}_k}\right) + \frac{\boldsymbol{\gamma}_k\boldsymbol{\gamma}_k^T}{\boldsymbol{\delta}_k^T\boldsymbol{\gamma}_k}. \tag{2.17}$$

The characteristic feature of the SR1 and DFP formulas is the validity of the secant equations

$$\boldsymbol{\gamma}_k^T\mathbf{H}_{k+1} = \boldsymbol{\delta}_k^T \tag{2.18}$$

and

$$\mathbf{B}_{k+1}\boldsymbol{\delta}_k = \boldsymbol{\gamma}_k. \tag{2.19}$$

Thereby, the SR1 and DFP updates track the curvature of the object function along the search path and store these informations in the updated matrices. In order to guarantee that the updated matrix keeps positive definite the curvature condition

$$\boldsymbol{\delta}_k^T\boldsymbol{\gamma}_k > 0 \tag{2.20}$$

must be satisfied in every iteration. It can be seen that this condition preserves the positive definiteness, if Equation (2.19) is multiplied from the left-hand side with $\boldsymbol{\delta}_k^T$. Obviously, the same applies to the secant equation of the inverse Hessian if (2.18) is multiplied from the right-hand side with $\boldsymbol{\gamma}_k^T$. The SR1 update has the drawback that the denominator can vanish. In this case the update cannot be applied.

A further update formula for the inverse of the Hessian which also fulfills the secant Eq. (2.18) is

$$\mathbf{H}_{k+1} = \left(\mathbf{I} - \frac{\boldsymbol{\delta}_k\boldsymbol{\gamma}_k^T}{\boldsymbol{\delta}_k^T\boldsymbol{\gamma}_k}\right)\cdot\mathbf{H}_k\cdot\left(\mathbf{I} - \frac{\boldsymbol{\gamma}_k\boldsymbol{\delta}_k^T}{\boldsymbol{\delta}_k^T\boldsymbol{\gamma}_k}\right) + \frac{\boldsymbol{\delta}_k\boldsymbol{\delta}_k^T}{\boldsymbol{\delta}_k^T\boldsymbol{\gamma}_k}.$$

This update formula is known as the BFGS update rule (named by Broyden, Fletcher, Goldfarb, and Shanno). The inverse BFGS formula is

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \frac{\boldsymbol{\gamma}_k\boldsymbol{\gamma}_k^T}{\boldsymbol{\delta}_k^T\boldsymbol{\gamma}_k} - \frac{\mathbf{B}_k\boldsymbol{\delta}_k\boldsymbol{\delta}_k^T\mathbf{B}_k}{\boldsymbol{\delta}_k^T\mathbf{B}_k\boldsymbol{\delta}_k}. \tag{2.21}$$

On the basis of the two update formulas (2.17) and (2.21), a convex class of update rules, known as the *Broyden class*, can be stated with

$$\mathbf{B}_{k+1} = (1 - \phi_k)\mathbf{B}_{k+1}^{BFGS} + \phi_k \mathbf{B}_{k+1}^{DFP}.$$

The explicit formula can be stated as

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \frac{\boldsymbol{\gamma}_k \boldsymbol{\gamma}_k^T}{\boldsymbol{\delta}_k^T \boldsymbol{\gamma}_k} - \frac{\mathbf{B}_k \boldsymbol{\delta}_k \boldsymbol{\delta}_k^T \mathbf{B}_k}{\boldsymbol{\delta}_k^T \mathbf{B}_k \boldsymbol{\delta}_k} + \phi_k \boldsymbol{v}_k \boldsymbol{v}_k^T$$

where

$$\boldsymbol{v}_k = \left(\boldsymbol{\delta}_k^T \mathbf{B}_k \boldsymbol{\delta}_k\right)^{1/2} \cdot \left(\frac{\boldsymbol{\gamma}_k}{\boldsymbol{\delta}_k^T \boldsymbol{\gamma}_k} - \frac{\mathbf{B}_k \boldsymbol{\delta}_k}{\boldsymbol{\delta}_k^T \mathbf{B}_k \boldsymbol{\delta}_k}\right).$$

Dennis and Moré [20] showed that a Quasi-Newton method converges Q-superlinearly if the condition

$$\lim_{k \to \infty} \frac{\| \left[\mathbf{B}_k - \nabla^2 f(\mathbf{y}^*)\right](\mathbf{y}_{k+1} - \mathbf{y}_k) \|}{\| \mathbf{y}_{k+1} - \mathbf{y}_k \|} = 0$$

is satisfied. They also showed that this condition is always fulfilled, if any of the above-described Quasi-Newton update methods is used and the curvature condition (2.20) is fulfilled. If a damped Quasi-Newton method is used the step-size must become 1 in the neighborhood of the solution $\mathbf{y}^*$ to hold for a Q-superlinear convergence rate.

2.3 Constrained Nonlinear Optimization

This section is about minimizing at least two times continuously differentiable functions subject to constraints. The following definitions and theorems can be found in numerous textbooks including Nocedal and Wright [59], Fletcher [29], McCormick [54], Avriel [4], and Gill et al. [36].

Definition 2.9 (*Constrained Nonlinear Programming Problem*) The *constrained nonlinear programming problem* with equality and inequality constraints is given by

$$\begin{aligned} \min_{\mathbf{y} \in \mathbb{R}^{N_y}} \quad & f(\mathbf{y}) \\ \text{subject to} \quad & \mathbf{g}(\mathbf{y}) \leq \mathbf{0} \\ & \mathbf{h}(\mathbf{y}) = \mathbf{0} \end{aligned} \tag{2.22}$$

where $f : \mathbb{R}^{N_y} \to \mathbb{R}$, $\mathbf{g} : \mathbb{R}^{N_y} \to \mathbb{R}^{N_g}$, and $\mathbf{h} : \mathbb{R}^{N_y} \to \mathbb{R}^{N_h}$ are all assumed to be twice continuously differentiable and real-valued. The constraints are defined as

$$\mathbf{g}(\mathbf{y}) := \begin{bmatrix} g_1(\mathbf{y}) \\ \vdots \\ g_{N_g}(\mathbf{y}) \end{bmatrix} \quad \text{and} \quad \mathbf{h}(\mathbf{y}) := \begin{bmatrix} h_1(\mathbf{y}) \\ \vdots \\ h_{N_h}(\mathbf{y}) \end{bmatrix}. \qquad \triangle$$

For a compact representation the optimization problem (2.22) may be rewritten to

$$\min_{\mathbf{y} \in \mathcal{S}} f(\mathbf{y}) \tag{2.23}$$

where $\mathcal{S}$ is the feasible set of points.

Definition 2.10 (*Feasible Set*) A point $\mathbf{y} \in \mathbb{R}^{N_y}$ is feasible, if all constraints of (2.22) are satisfied. The feasible set contains all feasible points

$$\mathcal{S} := \{\mathbf{y} \in \mathbb{R}^{N_y} \mid g_i(\mathbf{y}) \leq 0,\ i = 1, \ldots, N_g \ \wedge\ h_j(\mathbf{y}) = 0,\ j = 1, \ldots, N_h\}.$$

The feasible set $\mathcal{S} \subset \mathbb{R}^{N_y}$ is closed.

$\triangle$

Remark 2.1 The set $\mathcal{S}$ is also known by the terms: the *feasible region* or the *opportunity set*.

Remark 2.2 We assume that the feasible set is given by the solutions of a finite number of equations. *Abstract constraints* or *variational inequalities* are not considered.

Before we derive the optimality conditions of Problem (2.23) some further definitions are required.

Definition 2.11 (*Set of Active Indices*) An inequality constraint $g_i(\mathbf{y})$ is said to be active if $g_i(\mathbf{y}) = 0$. Then, the set of active indices $\mathcal{I}$ of the inequality constraints is defined by

$$\mathcal{I}(\mathbf{y}) := \left\{i = 1, \ldots, N_g \mid g_i(\mathbf{y}) = 0\right\}. \tag{2.24}$$

The number of active indices is defined by

$$N_{\mathcal{I}} := \#\mathcal{I}(\mathbf{y})$$

where the symbol # means the number of elements.

$\triangle$

The set of active indices characterizes the relevance of the inequality constraints to the optimal solution $\mathbf{y}^*$. That means, an inactive inequality constraint has no influence on the optimal solution.

Definition 2.12 (*Extremals*) A point $\mathbf{y}^* \in \mathbb{R}^{N_y}$ is said to be a *local minimum* of (2.23), if $\mathbf{y}^* \in \mathcal{S}$ is satisfied and $\epsilon > 0$ exists, such that

$$f(\mathbf{y}^*) \leq f(\mathbf{y}) \tag{2.25}$$

is satisfied $\forall \mathbf{y} \in \mathcal{S} \cap U_\epsilon(\mathbf{y}^*)$. If instead of (2.25),

$$f(\mathbf{y}^*) < f(\mathbf{y}) \tag{2.26}$$

applies, then $\mathbf{y}^*$ is called a *strict local minimum* of (2.23). For the case that (2.26) is satisfied $\forall \mathbf{y} \in \mathcal{S}$, $\mathbf{y}^*$ is a *global minimum* of (2.23).

△

2.3.1 Necessary and Sufficient Conditions for Optimality

One can find many necessary conditions with different strong assumptions in the literature, but only a few of them are directly applicable to practical optimization problems. The two fundamental results for (2.23), which have an impact on modern nonlinear programming algorithms, are the Fritz John and the Karush–Kuhn–Tucker conditions.

A prerequisite for stating the necessary conditions for the Problem (2.23) is the definition of the Lagrangian function, named after the mathematician Lagrange who treated equality-constrained optimization problems in the second half of the eighteenth century.

Definition 2.13 (*Lagrangian Function (Nocedal and Wright* [59]*)*) The mapping $\mathcal{L} : \mathbb{R}^{N_y} \times \mathbb{R} \times \mathbb{R}^{N_g} \times \mathbb{R}^{N_h} \to \mathbb{R}$ is defined by

$$\mathcal{L}(\mathbf{y}, l_0, \boldsymbol{\lambda}, \boldsymbol{\mu}) := l_0 f(\mathbf{y}) + \sum_{i=1}^{N_g} \lambda_i g_i(\mathbf{y}) + \sum_{i=1}^{N_h} \mu_i h_i(\mathbf{y}) \tag{2.27}$$

and is called the *Lagrangian function* for the problem definition (2.23). l_0 and the components of the vectors

$$\boldsymbol{\lambda}^T := \left[\lambda_1, \ldots, \lambda_{N_g}\right] \quad \text{and} \quad \boldsymbol{\mu}^T := \left[\mu_1, \ldots, \mu_{N_h}\right] \tag{2.28}$$

are called Lagrange multipliers.

△

Theorem 2.2 (First-Order Necessary Conditions, Fritz John Conditions) *Suppose that the functions* $f(\cdot)$, $\mathbf{g}(\cdot)$, *and* $\mathbf{h}(\cdot)$ *are continuously differentiable with respect to* $\mathbf{y}$. *Then, the following* Fritz John conditions *are satisfied at a local minimum* $(\mathbf{y}^*, \boldsymbol{\lambda}, \boldsymbol{\mu})$*:*

1. ***sign condition:***

$$l_0 \geq 0; \tag{2.29}$$

2. ***optimality condition:***

$$\nabla_y \mathcal{L}(\mathbf{y}^*, l_0, \boldsymbol{\lambda}, \boldsymbol{\mu}) = l_0 \nabla f(\mathbf{y}^*) + \nabla \mathbf{g}^T(\mathbf{y}^*)\boldsymbol{\lambda} + \nabla \mathbf{h}^T(\mathbf{y}^*)\boldsymbol{\mu} = \mathbf{0}. \tag{2.30}$$

where the Jacobian matrices of $\mathbf{g}(\cdot)$ *and* $\mathbf{h}(\cdot)$ *are of size* $\nabla \mathbf{g}(\mathbf{y}^*) \in \mathbb{R}^{N_g \cdot N_y}$ *and* $\nabla \mathbf{h}(\mathbf{y}^*) \in \mathbb{R}^{N_h \cdot N_y}$;

3. ***complementarity conditions:***

$$\boldsymbol{\lambda} \geq \mathbf{0}, \quad , \mathbf{g}(\mathbf{y}^*) \leq \mathbf{0}, \quad \boldsymbol{\lambda}^T \mathbf{g}(\mathbf{y}^*) = 0; \tag{2.31}$$

and

4. ***equality constraints:***

$$\mathbf{h}(\mathbf{y}^*) = \mathbf{0}. \tag{2.32}$$

Proof Proofs can be found in John [45], Mangasarian [51], and Bertsekas [6]. □

Each vector $(\mathbf{y}^*, l_0, \boldsymbol{\lambda}, \boldsymbol{\mu}) \in \mathbb{R}^{N_y+1+N_g+N_h}$ which satisfies the conditions (2.29)–(2.32) is called a Fritz John stationary point of (2.23). It can be shown that for a local minimum $\mathbf{y}^*$ a point $(l_0, \boldsymbol{\lambda}, \boldsymbol{\mu}) \neq 0$ with $l_0 > 0$ only exists if an additional condition, called a *regularity condition* or a *constraint qualification*, is satisfied. If such a constraint qualification holds, $l_0 = 1$ can be chosen without loss of generality.

In textbooks on optimization one can find numerous different constraint qualifications for nonconvex problems. In general, it is desirable to use the weakest constraint qualification in order to restrict the allowed constraints as little as possible. But it is a major drawback of the weaker constraint qualifications, e.g., the *quasiregularity condition*, that they cannot be evaluated in a simple manner. Therefore, stronger but much easier evaluable constraint qualifications like the Mangasarian–Fromowitz constraint qualification or the even stronger linear independence constraint qualification are often used. They are defined as follows.

Definition 2.14 (*Mangasarian–Fromowitz Constraint Qualification*) Let a point $\mathbf{y} \in \mathcal{S}$ be feasible for (2.23) and the active set $\mathcal{I}(\mathbf{y})$ be defined by (2.24). Then, the *Mangasarian–Fromowitz constraint qualification* (MFCQ) holds if the gradients of the equality constraints $\nabla h_j(\mathbf{y}),\ j = 1, \ldots, N_h$ are linearly independent (i.e., the constraints in this case are said to be *regular*) and there exists a vector $\mathbf{b} \in \mathbb{R}^{N_y}$ such that

$$\nabla h_j^T(\mathbf{y})\mathbf{b} = 0 \ (j = 1, \ldots, N_h) \text{ and } \nabla g_i^T(\mathbf{y})\mathbf{b} < 0 \ (\forall i \in \mathcal{I}(\mathbf{y})).$$

△

Definition 2.15 (*Linear Independence Constraint Qualification*) Let a point $\mathbf{y} \in \mathcal{S}$ be feasible for (2.23) and the active set $\mathcal{I}(\mathbf{y})$ be defined by (2.24). Then, the *linear independence constraint qualification* (LICQ) holds if the gradients $\nabla g_i(\mathbf{y}),\ \forall i \in \mathcal{I}(\mathbf{y})$ and $\nabla h_j(\mathbf{y}),\ j = 1, \ldots, N_h$ are linearly independent.

△

Remark 2.3 One can realize that these constraint qualifications are strongly restrictive. One can even show that these constraint qualifications are over restrictive just with a simple test. If an equality constraint $h_i(\mathbf{y}) = 0$ is replaced by two equivalent inequality constraints, e.g., with $g_i(\mathbf{y}) <= 0$ and $g_i(\mathbf{y}) >= 0$, neither the MFCQ nor the LICQ holds.

Theorem 2.3 (Karush–Kuhn–Tucker Conditions) *Assume that the functions* $f(\cdot)$, $\mathbf{g}(\cdot)$, *and,* $\mathbf{h}(\cdot)$ *are continuously differentiable with respect to* $\mathbf{y}$ *and a constraint qualification holds. Then the following conditions at a local minimum* $(\mathbf{y}^*, \boldsymbol{\lambda}, \boldsymbol{\mu})$

$$\nabla_y \mathcal{L}(\mathbf{y}^*, 1, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \mathbf{0} \tag{2.33}$$
$$\nabla_\mu \mathcal{L}(\mathbf{y}^*, 1, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \mathbf{h}(\mathbf{y}^*) = \mathbf{0} \tag{2.34}$$
$$\nabla_\lambda \mathcal{L}(\mathbf{y}^*, 1, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \mathbf{g}(\mathbf{y}^*) \leq \mathbf{0} \tag{2.35}$$
$$\boldsymbol{\lambda}^T \mathbf{g}(\mathbf{y}^*) = 0 \tag{2.36}$$
$$\boldsymbol{\lambda} \geq \mathbf{0} \tag{2.37}$$

are satisfied and are known as Karush–Kuhn–Tucker (KKT) *conditions.*

△

Proof Proofs of Theorem 2.3 can be found in Karush [48], Kuhn and Tucker [49], Mangasarian [51], Bertsekas [6], Fiacco and McCormick [27] and Fletcher [29]. □

Remark 2.4 The Fritz John conditions with $l_0 = 1$ are also called Karush–Kuhn–Tucker conditions.

Analogous to the Fritz John conditions, each vector $(\mathbf{y}^*, \boldsymbol{\lambda}, \boldsymbol{\mu}) \in \mathbb{R}^{N_y + N_g + N_h}$ which satisfies the KKT conditions (2.33)–(2.37) is called a stationary KKT point of (2.23).

Condition (2.36) can be tightened with the strict complementarity condition which excludes the case $\lambda_i = 0$ and $g_i = 0$ for any $i = 1, \ldots, N_g$. This stronger assumption is needed in the sensitivity analysis.

Definition 2.16 (*Strict Complementarity Condition*) The *strict complementarity condition* is satisfied, if

$$\boldsymbol{\lambda} - \mathbf{g}(\mathbf{y}) > \mathbf{0} \tag{2.38}$$

holds. △

In order to ensure that any stationary KKT point $(\mathbf{y}^*, \boldsymbol{\lambda}, \boldsymbol{\mu})$ is indeed an optimal solution of problem definition (2.23), *second-order sufficient conditions* (SOSC) are needed. SOSC also play an important role in the sensitivity analysis discussed in Sect. 2.4.

For the formulation of the SOSC we need the definition of the *critical cone*.

Definition 2.17 (*Critical Cone*) Suppose $\mathbf{y}$ is a feasible point, then the *critical cone* is defined by

$$\begin{aligned} \mathcal{T}_y := \{\, \mathbf{v} \in \mathbb{R}^{N_y} \;\mid\; & \nabla g_i^T(\mathbf{y})\mathbf{v} \le 0, \quad i \in \mathcal{I}(\mathbf{y}), \quad \lambda_i = 0, \\ & \nabla g_i^T(\mathbf{y})\mathbf{v} = 0, \quad i \in \mathcal{I}(\mathbf{y}), \quad \lambda_i > 0 \\ & \nabla h_j^T(\mathbf{y})\mathbf{v} = 0, \quad j \in \{1, \ldots, N_h\} \,\}. \end{aligned}$$

△

Theorem 2.4 (Second-Order Sufficient Conditions) *Let $\mathbf{y}^* \in \mathcal{S}$ be a feasible point for the problem definition (2.23) and let the functions $f(\cdot)$, $\mathbf{g}(\cdot)$, and $\mathbf{h}(\cdot)$ be twice continuously differentiable with respect to $\mathbf{y}$. Assume that Lagrange vectors $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ exist, Theorem 2.3 holds and that the Hessian of the Lagrangian is positive definite on the critical cone*

$$\mathbf{v}^T \nabla_y^2 \mathcal{L}(\mathbf{y}^*, \boldsymbol{\lambda}, \boldsymbol{\mu})\mathbf{v} > 0, \quad \forall \mathbf{v} \in \mathcal{T}_{\mathbf{y}^*}, \mathbf{v} \neq \mathbf{0}. \tag{2.39}$$

Then, $\mathbf{y}^*$ satisfies the SOSC and is a strict local minimum of (2.23).

△

Proof The proof can be found in Karush [48], Fletcher [29], and Bertsekas [6]. □

2.3.2 Projected Hessian

Because of the critical cone the evaluation of Theorem 2.4 is quite difficult in practice. Therefore, we project the Hessian of the Lagrangian to the kernel of the constraints to obtain numerically evaluable conditions.

Suppose, we found a triple $(\mathbf{y}, \boldsymbol{\lambda}, \boldsymbol{\mu})$ of (2.23) which is feasible and the strict complementarity condition of the Lagrange multipliers and a constraint qualification holds, i.e., the tangent cone is a vector space. Then, with the Jacobian matrix of all active constraints

$$\mathbf{A}_y := \left(\nabla g_i(\mathbf{y})\ \nabla h_j(\mathbf{y})\right)^T,$$

where the indices are given by $i \in \mathcal{I}(\mathbf{y})$ and $j = 1, \ldots, N_h$, we can restate the critical cone as

$$\mathcal{T}_y = \left\{\mathbf{v} \in \mathbb{R}^{N_y} \mid \mathbf{A}_y\mathbf{v} = \mathbf{0}\right\} = \ker(\mathbf{A}_y).$$

The set $\ker(\mathbf{A}_y)$ denotes the *kernel of the matrix* $\mathbf{A}_y$. It should be noted that the dimension of the matrix depends on the number of active constraints $\mathbf{A}_y \in \mathbb{R}^{N_y \times N_s}$ with $N_s := N_{\mathcal{I}} + N_h$.

Using the kernel $\ker(\mathbf{A}_y)$, Condition (2.39) is equivalent to

$$\mathbf{v}^T \nabla_y^2 \mathcal{L}(\mathbf{y}^*, \boldsymbol{\lambda}, \boldsymbol{\mu})\mathbf{v} > 0 \quad \forall \mathbf{v} \in \ker\left(\mathbf{A}_y\right), \mathbf{v} \neq \mathbf{0}.$$

Introducing the matrix $\mathbf{P} \in \mathbb{R}^{N_y \times (N_y - N_s)}$ with full column rank whose columns span the kernel $\ker(\mathbf{A}_y)$ such that

$$\mathbf{A}_y \mathbf{P} = \mathbf{0}_{N_s \times (N_y - N_s)},$$

any vector $\mathbf{v} \in \ker(\mathbf{A}_y)$ can be uniquely reexpressed as

$$\mathbf{v} = \mathbf{P}\mathbf{w}$$

for some vector $\mathbf{w} \in \mathbb{R}^{N_y - N_s}$. Then, Condition (2.39) may be restated as

$$\mathbf{w}^T \mathbf{P}^T \nabla_y^2 \mathcal{L}(\mathbf{y}^*, \boldsymbol{\lambda}, \boldsymbol{\mu})\mathbf{P}\mathbf{w} > 0, \quad \forall \mathbf{w} \in \mathbb{R}^{N_y - N_s}, \mathbf{w} \neq \mathbf{0} \tag{2.40}$$

under the assumption that the strict complementarity condition (2.38) holds and $\mathbf{P}$ forms an orthogonal basis for $\ker(\mathbf{A}_y)$.

The matrix

$$\mathbf{B}_{red} := \mathbf{P}^T \nabla_y^2 \mathcal{L}(\mathbf{y}^*, \boldsymbol{\lambda}, \boldsymbol{\mu})\mathbf{P} \in \mathbb{R}^{(N_y - N_s) \times (N_y - N_s)} \tag{2.41}$$

is called the *projected or reduced Hessian*. From (2.40) it can be concluded that the positive definiteness of the reduced Hessian is equivalent to the positive definiteness of the full Hessian $\nabla_y^2 \mathcal{L}(\mathbf{y}^*, \boldsymbol{\lambda}, \boldsymbol{\mu})$ on the critical cone. Thus, the check for SOSC is reduced by showing that the projected Hessian is positive definite. The matrix $\mathbf{P}$ can be obtained by a QR-factorization of $\mathbf{A}_y^T$ with

$$\mathbf{A}_y^T = \mathbf{Q}\mathbf{R} = \begin{pmatrix} \mathbf{Q}_1 & \mathbf{Q}_2 \end{pmatrix} \begin{pmatrix} \mathbf{R}_1 \\ \mathbf{0}_{(N_y - N_s) \times N_s} \end{pmatrix} \tag{2.42}$$

where $\mathbf{Q} \in \mathbb{R}^{N_y \times N_y}$ is an orthogonal matrix and $\mathbf{R}_1 \in \mathbb{R}^{N_s \times N_s}$ is a regular upper triangular matrix. Then $\mathbf{P} := \mathbf{Q}_2$ forms an orthogonal basis for $\ker(\mathbf{A}_y)$, which can be shown if Eq. (2.42) is multiplied from the left-hand side by $\mathbf{Q}^T$

$$\begin{pmatrix} \mathbf{Q}_1^T \\ \mathbf{Q}_2^T \end{pmatrix} \mathbf{A}_y(\mathbf{y}) = \begin{pmatrix} \mathbf{R}_1 \\ \mathbf{0}_{(N_y - N_s) \times N_s} \end{pmatrix} \Rightarrow \mathbf{A}_y \mathbf{Q}_2 = \mathbf{0}_{N_s \times (N_y - N_s)}.$$

2.3.3 Sequential Quadratic Programming

A group of very popular methods to solve the constrained minimization problem (2.23) are *sequential quadratic programming* (SQP) methods. SQP algorithms are iterative and in every iteration they intend to calculate a suitable search direction. Han [43] and Powell [65] showed that such methods converge to a solution (KKT point) of the NLP problem using local curvature information by approximating the Lagrangian as a quadratic function and the nonlinear constraints as linearized constraints. They also showed that the local convergence rate of these methods is Q-superlinear, if an update method from the Broyden class (i.e., BFGS or DFP) is used to obtain a convex approximation of the Lagrangian. A local minimizer is found by solving a sequence of these convex quadratic subproblems.

There are two common variants of the SQP method, *line-search* and *filter SQP* methods. Line-search SQP methods extend the Quasi-Newton methods of unconstrained optimization with line-search globalization strategy for constrained problems. The step-length is calculated for a suitable merit function, which accounts for the object function and the constraints. Filter SQP methods extend the Quasi-Newton methods of unconstrained optimization with trust-region globalization strategy to constrained problems.

As for the unconstrained case we will concentrate only on line-search methods for constrained optimization, because the numerical results for both types of SQP algorithms are nearly identical for the problem set considered in this book. Furthermore, only the Quasi-Newton based SQP methods are discussed.

The construction of line-search SQP methods is analogous to Newton–Raphson methods. The gradient of the Lagrangian function (2.27) is approximated by a first-order Taylor expansion, which is supposed to vanish in a local minimum according to the necessary conditions. We obtain the first-order Taylor expansion of the Lagrangian function as

$$\nabla_y \mathcal{L}(\mathbf{y}+\mathbf{d},\boldsymbol{\lambda},\boldsymbol{\mu}) \approx \nabla f(\mathbf{y}) + \nabla \mathbf{g}^T(\mathbf{y})\boldsymbol{\lambda} + \nabla \mathbf{h}^T(\mathbf{y})\boldsymbol{\mu} + \mathbf{B}(\mathbf{y},\boldsymbol{\lambda},\boldsymbol{\mu})\mathbf{d} = \mathbf{0}. \quad (2.43)$$

The matrix $\mathbf{B}(\mathbf{y},\boldsymbol{\lambda},\boldsymbol{\mu})$ accounts for second-order derivatives of the object function and the constraints and is updated in every iteration with a suitable Quasi-Newton update strategy.

The constraints $\mathbf{g}(\cdot)$ and $\mathbf{h}(\cdot)$ are also approximated with a first-order Taylor expansion in every iteration step by

$$\mathbf{g}(\mathbf{y}+\mathbf{d}) \approx \mathbf{g}(\mathbf{y}) + \nabla \mathbf{g}(\mathbf{y})\mathbf{d} \leq \mathbf{0} \quad (2.44)$$

and

$$\mathbf{h}(\mathbf{y}+\mathbf{d}) \approx \mathbf{h}(\mathbf{y}) + \nabla \mathbf{h}(\mathbf{y})\mathbf{d} = \mathbf{0}. \quad (2.45)$$

The search direction can be calculated by solving (2.43)–(2.45). If the active set of the inequality constraints is known, only a system of linear equations must be solved to obtain the search direction. In general, however, the active set is not known a priori.

Notice that the solution of the system of Eqs. (2.43) and (2.45) and the inequalities (2.44) is equivalent to the minimization of a convex *quadratic subproblem* (QSP) with linear constraints.

Definition 2.18 (*Quadratic Subproblem*) A QSP with linear constraints is defined by

$$\begin{aligned} \min_{\mathbf{d}\in\mathbb{R}^{N_y}} \quad & f(\mathbf{y}) + \nabla f^T(\mathbf{y})\mathbf{d} + \frac{1}{2}\mathbf{d}^T\mathbf{B}\mathbf{d} \\ \text{subject to} \quad & \mathbf{g}(\mathbf{y}) + \nabla \mathbf{g}^T(\mathbf{y})\mathbf{d} \leq \mathbf{0}, \\ & \mathbf{h}(\mathbf{y}) + \nabla \mathbf{h}^T(\mathbf{y})\mathbf{d} = \mathbf{0}. \end{aligned} \tag{2.46}$$

We regard the solution to the quadratic subproblem (2.46) as search direction $\mathbf{d}_k$ that will be a descent direction for some merit function. Analogous to the globalized Quasi-Newton method, we can generate a sequence $\{\mathbf{y}_k\}_{k\in\mathbb{N}}$ by applying the update rule

$$\mathbf{y}_{k+1} = \mathbf{y}_k + \alpha_k \mathbf{d}_k$$

where α_k is a positive step-length and $\mathbf{d}_k$ is the feasible search direction calculated for the k-th iteration of the SQP algorithm by solving the quadratic subproblem (2.46).

In the following three subsections, some methods are presented for solving the quadratic subproblem to obtain the search direction, to calculate the step-length, and to calculate appropriate Quasi-Newton updates for the Hessian approximation.

2.3.3.1 Solution of the Quadratic Subproblem

For the solution of the QSP (2.46) *active set* (AS) and *interior-point* (IP) methods are particularly suitable. The theory of solving convex quadratic problems is quite complex and would take a lot of space in this book. Therefore, only the interior-point algorithm in its basic form will be introduced. Interested readers may consult the textbooks of Nocedal and Wright [59], Fletcher [29], and Wright [73] to get more information about this type of problem.

Before reviewing the IP algorithm we start with a QSP, which is only equality constrained, and rename the variables as $\boldsymbol{\omega} := \mathbf{y}_k$, $\mathbf{x} := \mathbf{d}_k$, $\mathbf{y} := \boldsymbol{\mu}_k$, and $\mathbf{z} := \boldsymbol{\lambda}_k$ for a more compact problem formulation.

Definition 2.19 (*Convex Quadratic Subproblem with Equality Constraints*) The convex QSP with equality constraints can be compactly stated as

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^{N_y}} \quad & f + \mathbf{x}^T\mathbf{c} + \frac{1}{2}\mathbf{x}^T\mathbf{G}\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \end{aligned} \tag{2.47}$$

where $\mathbf{G} := \mathbf{B}_k$ is the symmetric $N_y \times N_y$ Hessian matrix, $\mathbf{A} := \nabla \mathbf{h}^T(\boldsymbol{\omega})$ is the Jacobian of the equality constraints, $\mathbf{c} := \nabla f(\boldsymbol{\omega})$ is the gradient of the objective function, $\mathbf{b} := \mathbf{h}(\boldsymbol{\omega})$ are the equality constraints, and $f := f(\boldsymbol{\omega})$ is the objective function value—all at the k-th iteration. Notably, the term to be minimized is quadratic and the constraints are linear.

△

From (2.27) we know that the Lagrangian for the quadratic problem (2.47) is given by

$$\mathcal{L}(\mathbf{x}, \mathbf{y}) = f + \mathbf{x}^T\mathbf{c} + \frac{1}{2}\mathbf{x}^T\mathbf{G}\mathbf{x} - \mathbf{y}^T(\mathbf{A}\mathbf{x} - \mathbf{b}).$$

The first-order KKT conditions (see Theorem 2.3) of the equality-constrained problem (2.47) can be written as

$$\mathbf{F}(\mathbf{x}, \mathbf{y}) = \begin{bmatrix} \nabla_x \mathcal{L}(\mathbf{x}, \mathbf{y}) \\ \nabla_y \mathcal{L}(\mathbf{x}, \mathbf{y}) \end{bmatrix} = \begin{bmatrix} \mathbf{c} + \mathbf{G}\mathbf{x} - \mathbf{A}^T\mathbf{y} \\ -\mathbf{A}\mathbf{x} + \mathbf{b} \end{bmatrix} = \mathbf{0}_{(N_x+N_h)\times 1}. \tag{2.48}$$

The Jacobian of (2.48) with respect to $\mathbf{x}$ and $\mathbf{y}$ is given by

$$\frac{\partial \mathbf{F}}{\partial(\mathbf{x}, \mathbf{y})}(\mathbf{x}, \mathbf{y}) = \begin{pmatrix} \nabla_x^2 \mathcal{L}(\mathbf{x}, \mathbf{y}) & \nabla_{xy} \mathcal{L}(\mathbf{x}, \mathbf{y}) \\ \nabla_{yx} \mathcal{L}(\mathbf{x}, \mathbf{y}) & \nabla_y^2 \mathcal{L}(\mathbf{x}, \mathbf{y}) \end{pmatrix} = \begin{pmatrix} \mathbf{G} & -\mathbf{A}^T \\ -\mathbf{A} & \mathbf{0}_{N_h \times N_h} \end{pmatrix} \tag{2.49}$$

and is known as the so-called *Karush–Kuhn–Tucker*-matrix (KKT matrix for short).

The solution of (2.48) can be calculated by

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = -\begin{pmatrix} \mathbf{G} & -\mathbf{A}^T \\ -\mathbf{A} & \mathbf{0}_{N_h \times N_h} \end{pmatrix}^{-1} \begin{bmatrix} \mathbf{c} \\ \mathbf{b} \end{bmatrix}, \tag{2.50}$$

which is equivalent to the calculation of Newton's search direction.

Remark 2.5 Note that the sign of equality constraints together with the Lagrange parameters $\mathbf{y}^T(\mathbf{A}\mathbf{x} - \mathbf{b})$ coupled to the Lagrangian can be chosen arbitrarily.

Having in mind the quadratic problem formulation (2.46) and using the new variables one obtains a general convex quadratic problem formulation.

Definition 2.20 (*Convex Quadratic Subproblem*) In addition to Definition 2.19, the general convex QSP can be stated as

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^{N_y}} \quad & f + \mathbf{x}^T\mathbf{c} + \frac{1}{2}\mathbf{x}^T\mathbf{G}\mathbf{x} \\ \text{subject to} \quad & \mathbf{C}\mathbf{x} \geq \mathbf{d}, \\ & \mathbf{A}\mathbf{x} = \mathbf{b} \end{aligned}$$

where $\mathbf{C} := -\nabla \mathbf{g}^T(\boldsymbol{\omega})$ is the negative Jacobian of the inequality constraints and $\mathbf{d} := \mathbf{g}(\boldsymbol{\omega})$ are the inequality constraints.

△

Interior-Point Method

We apply Mehrotra's predictor-corrector IP method described in [55] to problem definition 2.20. We first introduce slack variables $\mathbf{s}$ to convert the inequality constraints to equality constraints with additional box constraints for the slack variables. Then the first-order necessary conditions for this problem formulation can be stated as

$$\begin{aligned} \mathbf{G}\mathbf{x} - \mathbf{A}^T\mathbf{y} - \mathbf{C}^T\mathbf{z} &= -\mathbf{c}, \\ \mathbf{A}\mathbf{x} &= \mathbf{b}, \\ \mathbf{C}\mathbf{x} - \mathbf{s} &= \mathbf{d}, \\ \mathbf{z} \geq \mathbf{0}, \quad \mathbf{s} \geq \mathbf{0}, \quad \mathbf{z}^T\mathbf{s} &= 0. \end{aligned} \tag{2.51}$$

A feasible starting point for this method can be directly denoted, because every point $(\mathbf{x}_0, \mathbf{y}_0, \mathbf{z}_0, \mathbf{s}_0)$ with $\mathbf{z}_0 > \mathbf{0}$ and $\mathbf{s}_0 > \mathbf{0}$ is feasible. The iterates of the interior-point method are always feasible, because $\mathbf{z}_j > \mathbf{0}$ and $\mathbf{s}_j > \mathbf{0}$ will be guaranteed due to a suitable step-size calculation. In contrast to feasibility, the initial point will usually not satisfy the complementary condition (2.51). Instead of imposing this condition at every iteration, the algorithm aims at decreasing an appropriately chosen residual of this condition. Therefore, the complementarity measure given by

$$\mu = \frac{\mathbf{z}^T\mathbf{s}}{N_g} \tag{2.52}$$

plays an important role in interior-point methods. Consequently, the violation of condition (2.52) will be forced by the algorithm to become smaller with every iteration.

Given a starting point $(\mathbf{x}_0, \mathbf{y}_0, \mathbf{z}_0, \mathbf{s}_0)$ with $(\mathbf{z}_0, \mathbf{s}_0) > \mathbf{0}$ and a parameter $\tau \in [2, 4]$, the following steps for $j = 1, 2, \ldots$ will be repeated until the convergence check is verified and the algorithm ends with an optimal solution.

Algorithm 2.1 QP Interior-point method (cf. Gertz and Wright [33])

1: $j \leftarrow 1$

2: $\mu_j \leftarrow \dfrac{\mathbf{z}_j^T \mathbf{s}_j}{N_g}$

3: **if** first-order necessary conditions are satisfied up to a given tolerance **then**

4: **stop**

5: **end if**

6: Solve for $(\Delta \mathbf{x}_j^{aff}, \Delta \mathbf{y}_j^{aff}, \Delta \mathbf{z}_j^{aff}, \Delta \mathbf{s}_j^{aff})$:

$$\begin{pmatrix} \mathbf{G} & -\mathbf{A}^T & -\mathbf{C}^T & 0 \\ \mathbf{A} & 0 & 0 & 0 \\ \mathbf{C} & 0 & 0 & -\mathbf{I} \\ 0 & 0 & \mathbf{S}_j & \mathbf{Z}_j \end{pmatrix} \begin{bmatrix} \Delta \mathbf{x}_j^{aff} \\ \Delta \mathbf{y}_j^{aff} \\ \Delta \mathbf{z}_j^{aff} \\ \Delta \mathbf{s}_j^{aff} \end{bmatrix} = - \begin{bmatrix} \mathbf{r}_{Gj} \\ \mathbf{r}_{Aj} \\ \mathbf{r}_{Cj} \\ \mathbf{Z}_j \mathbf{S}_j \mathbf{e} \end{bmatrix},$$

where $\mathbf{S}_j = \text{diag}(s_{j1}, s_{j2}, \ldots, s_{jN_g})$, $\mathbf{Z}_j = \text{diag}(z_{j1}, z_{j2}, \ldots, z_{jN_g})$, $\mathbf{r}_{Gj} = \mathbf{G}\mathbf{x}_j - \mathbf{A}^T \mathbf{y}_j - \mathbf{C}^T \mathbf{z}_j + \mathbf{c}$, $\mathbf{r}_{Aj} = \mathbf{A}\mathbf{x}_j - \mathbf{b}$, and $\mathbf{r}_{Cj} = \mathbf{C}\mathbf{x}_j - \mathbf{s}_j - \mathbf{d}$

7: Compute

$$\alpha_j^{aff} = \arg \max_{\alpha \in (0,1]} \left\{ (\mathbf{z}_j, \mathbf{s}_j) + \alpha \left(\Delta \mathbf{z}_j^{aff}, \Delta \mathbf{s}_j^{aff} \right) \geq 0 \right\}$$

8:

$$\mu_j^{aff} \leftarrow \frac{\left(\mathbf{z}_j + \alpha_j^{aff} \Delta \mathbf{z}_j^{aff} \right)^T \left(\mathbf{s}_j + \alpha_j^{aff} \Delta \mathbf{s}_j^{aff} \right)}{N_g}$$

9:

$$\sigma_j \leftarrow \left(\frac{\mu_j^{aff}}{\mu_j} \right)^{\tau}$$

10: Solve for $(\Delta \mathbf{x}_j, \Delta \mathbf{y}_j, \Delta \mathbf{z}_j, \Delta \mathbf{s}_j)$:

$$\begin{pmatrix} \mathbf{G} & -\mathbf{A}^T & -\mathbf{C}^T & 0 \\ \mathbf{A} & 0 & 0 & 0 \\ \mathbf{C} & 0 & 0 & -\mathbf{I} \\ 0 & 0 & \mathbf{S}_j & \mathbf{Z}_j \end{pmatrix} \begin{bmatrix} \Delta \mathbf{x}_j \\ \Delta \mathbf{y}_j \\ \Delta \mathbf{z}_j \\ \Delta \mathbf{s}_j \end{bmatrix} = - \begin{bmatrix} \mathbf{r}_{Gj} \\ \mathbf{r}_{Aj} \\ \mathbf{r}_{Cj} \\ \mathbf{Z}_j \mathbf{S}_j \mathbf{e} - \sigma_j \mu_j \mathbf{e} + \Delta \mathbf{Z}_j^{aff} \Delta \mathbf{S}_j^{aff} \mathbf{e} \end{bmatrix},$$

where $\Delta \mathbf{Z}_j^{aff}$ and $\Delta \mathbf{S}_j^{aff}$ are defined in the same way as $\mathbf{Z}_j$ and $\mathbf{S}_j$ from step 6

11: Compute

$$\alpha_j^{max} = \arg \max_{\alpha \in (0,1]} \{ (\mathbf{z}_j, \mathbf{s}_j) + \alpha (\Delta \mathbf{z}_j, \Delta \mathbf{s}_j) \geq \mathbf{0} \}$$

12: Choose $\alpha_j \in (0, \alpha^{max})$ according to Mehrotra's heuristic

13: Update $\mathbf{x}_{j+1} \leftarrow \mathbf{x}_j + \alpha_j \Delta \mathbf{x}_j$, $\mathbf{y}_{j+1} \leftarrow \mathbf{y}_j + \alpha_j \Delta \mathbf{y}_j$, $\mathbf{z}_{j+1} \leftarrow \mathbf{z}_j + \alpha_j \Delta \mathbf{z}_j$, $\mathbf{s}_{j+1} \leftarrow \mathbf{s}_j + \alpha_j \Delta \mathbf{s}_j$, and $j \leftarrow j + 1$

14: Return to step 2.

For details on the implementation of convergence criteria, Mehrotra's heuristic for step-size determination, simplifications of the linear system of equations, and higher order corrections the interested reader may refer to Mehrotra [55], Gondzio [40], and Gertz and Wright [33].

2.3.3.2 Quasi-Newton Update for Constrained Problems

The Quasi-Newton update formulas from Sect. 2.2.4 can also be used for the SQP method, if one replaces the definition of $\boldsymbol{\gamma}_k$ with

$$\boldsymbol{\gamma}_k = \nabla_y \mathcal{L}(\mathbf{y}_{k+1}, \boldsymbol{\lambda}_{k+1}, \boldsymbol{\mu}_{k+1}) - \nabla_y \mathcal{L}(\mathbf{y}_k, \boldsymbol{\lambda}_{k+1}, \boldsymbol{\mu}_{k+1}).$$

The proof of the local convergence of this Quasi-Newton update for constrained optimization can be transferred to the unconstrained case, if the projected Hessian (2.41) is used. Powell showed in [65] that

$$\lim_{k\to\infty} \frac{\| \mathbf{P}_k^T \cdot [\mathbf{B}_k - \nabla_y^2 \mathcal{L}(\mathbf{y}^*, \boldsymbol{\lambda}, \boldsymbol{\mu})] \cdot \mathbf{P}_k \cdot (\mathbf{y}_{k+1} - \mathbf{y}_k) \|}{\| \mathbf{y}_{k+1} - \mathbf{y}_k \|} = 0$$

implies

$$\frac{\mathbf{y}_{k+1} - \mathbf{y}^*}{\mathbf{y}_{k-1} - \mathbf{y}^*} \to 0 \tag{2.53}$$

where $\mathbf{P}_k$ is the same projection matrix as in Eqs. (2.41) and (2.53) implies a two-step superlinear convergence rate. It should be reminded that the strict complementary condition must hold for the existence of the projected Hessian and is therefore required for this result about the convergence rate.

A practical drawback of the Quasi-Newton update for constrained optimization is the fact that the curvature condition (2.20) tends to be violated more frequently and therefore the Quasi-Newton update matrix $\mathbf{B}_{k+1}$ is no longer positive definite. To avoid this drawback Powell suggests the use of a modified BFGS update with

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \frac{\boldsymbol{\eta}_k \boldsymbol{\eta}_k^T}{\boldsymbol{\delta}_k^T \boldsymbol{\eta}_k} - \frac{\mathbf{B}_k \boldsymbol{\delta}_k \boldsymbol{\delta}_k^T \mathbf{B}_k}{\boldsymbol{\delta}_k^T \mathbf{B}_k \boldsymbol{\delta}_k} \tag{2.54}$$

where $\boldsymbol{\gamma}_k$ is replaced with $\boldsymbol{\eta}_k$, which is defined as

$$\boldsymbol{\eta}_k = \theta_k \boldsymbol{\gamma}_k + (1 - \theta_k) \mathbf{B}_k \boldsymbol{\delta}_k. \tag{2.55}$$

The factor θ_k is given by

$$\theta_k = \begin{cases} 1, & \boldsymbol{\delta}_k^T \boldsymbol{\gamma}_k \geq 0.2 \boldsymbol{\delta}_k^T \mathbf{B}_k \boldsymbol{\delta}_k \\ \dfrac{0.8 \boldsymbol{\delta}_k^T \mathbf{B}_k \boldsymbol{\delta}_k}{\boldsymbol{\delta}_k^T \mathbf{B}_k \boldsymbol{\delta}_k - \boldsymbol{\delta}_k^T \boldsymbol{\gamma}_k}, & \boldsymbol{\delta}_k^T \boldsymbol{\gamma}_k < 0.2 \boldsymbol{\delta}_k^T \mathbf{B}_k \boldsymbol{\delta}_k \end{cases}.$$

With this modification, the stronger condition

$$\boldsymbol{\delta}_k^T \boldsymbol{\eta}_k > 0.2 \boldsymbol{\delta}_k^T \mathbf{B}_k \boldsymbol{\delta}_k$$

is satisfied in every iteration of the SQP method and the BFGS matrix remains positive definite. The update formula (2.54) has proven to perform well in many applications (often even with a Q-superlinear rate of convergence) and Powell proved that the convergence rate is at least R-superlinear if some additional assumptions for the modified BFGS updates hold (see Powell [65]). However, there is, as far as we know, no general proof of local convergence for this modified update formula.

2.3.3.3 Step-Size Calculation with Merit Functions

After the calculation of the search direction $\mathbf{d}_k$ a suitable step-size α_k must be determined. To accomplish this task a merit function can be defined, which weights the decrease in the object function and the violation of the constraints. A prerequisite is that the search direction $\mathbf{d}_k$ obtained from the QP is also a descent direction of the merit function.

The necessity of a merit function can be best imagined for a given pair of points. If the two points are feasible one can determine which is best by comparing the objective function values. However, this becomes problematic if the points are allowed to be infeasible. Then, it is not apparent which of the two points offers the best approximation to the solution.

In their line-search SQP algorithms, Han and Powell introduced the l_1 merit function

$$\Psi(\mathbf{y}, \varrho, \nu) = f(\mathbf{y}) + \varrho^T \cdot \max(0, \mathbf{g}(\mathbf{y})) + \nu^T \cdot |\mathbf{h}(\mathbf{y})|,$$

where the maximum must be evaluated element-wise, and Powell suggests updating the penalty parameters ϱ and ν according to the update rules

$$\varrho_k = \begin{cases} |\boldsymbol{\lambda}_k|, & k = 0 \\ \max\left(|\boldsymbol{\lambda}_k|, \frac{1}{2}\left(\varrho_{k-1} + |\boldsymbol{\lambda}_k|\right)\right), & k > 0 \end{cases} \tag{2.56}$$

and

$$\nu_k = \left|\boldsymbol{\mu}_k\right|. \tag{2.57}$$

That means, the merit function assigns a positive penalty for increasing constraint violations. For the penalty parameters (2.56) and (2.57) it can be shown that the search direction $\mathbf{d}_k$ is a descent direction of the l_1 merit function. Because of the non-differentiability of the l_1 merit function Powell recommends using the backtracking algorithm with a modification of the Armijo condition (2.13) as step-size procedure

$$f(\mathbf{y}_k + \alpha_k \mathbf{d}_k) < f(\mathbf{y}_k) + \epsilon \alpha_k (f(\mathbf{y}_k + \mathbf{d}_k) - f(\mathbf{y}_k))$$

with $\epsilon = 0.1$.

A further drawback of the l_1 merit function is that the iterations can cycle, such that the SQP algorithm does not converge at all or converges only very slowly. This effect is known as the *Maratos effect* [52].

To avoid the Maratos effect some modifications to the algorithm can be applied. One possible modification to overcome the problem is the Watchdog technique proposed by Chamberlain, Powell, Lemarechal, and Pedersen [14]. Another approach is a second-order correction of the QSP solution as described in Mayne and Polak [53], Fletcher [28], and Coope [16].

The aim of a second-order correction is to replace the update $\mathbf{y}_{k+1} = \mathbf{y}_k + \alpha_k \mathbf{d}_k$ by a corrected update

$$\mathbf{y}_{k+1} = \mathbf{y}_k + \alpha_k \mathbf{d}_k + \alpha_k^2 \tilde{\mathbf{d}}_k,$$

where the correction step $\tilde{\mathbf{d}}_k$ will be obtained by a modified equality or modified inequality-constrained quadratic subproblem.

Definition 2.21 (*Equality-Constrained Quadratic Subproblem for the Second-Order Correction (cf. Coope* [16]*)*) The modified equality-constrained quadratic subproblem for obtaining second-order corrections can be formulated as

$$\begin{aligned} \min_{\tilde{\mathbf{d}} \in \mathbb{R}^{N_y}} \quad & \tilde{\mathbf{d}}^T \left(\nabla f(\mathbf{y}) + \mathbf{d}\right) + \frac{1}{2} \tilde{\mathbf{d}}^T \tilde{\mathbf{d}} \\ \text{subject to} \quad & \mathbf{g}_{i \in \mathcal{I}(\mathbf{y})}(\mathbf{y} + \mathbf{d}) + \nabla \mathbf{g}^T_{i \in \mathcal{I}(\mathbf{y})}(\mathbf{y}) \tilde{\mathbf{d}} = \mathbf{0}, \\ & \mathbf{h}(\mathbf{y} + \mathbf{d}) + \nabla \mathbf{h}^T(\mathbf{y}) \tilde{\mathbf{d}} = \mathbf{0}. \end{aligned}$$

△

Definition 2.22 (*Inequality-Constrained Quadratic Subproblem for Fletcher's Second-Order Correction (cf. Fletcher* [28]) The modified inequality-constrained quadratic subproblem for obtaining second-order corrections can be formulated as

$$\begin{aligned} \min_{\tilde{\mathbf{d}} \in \mathbb{R}^{N_y}} \quad & \tilde{\mathbf{d}}^T \left(\nabla f(\mathbf{y}) + \mathbf{B}\mathbf{d}\right) + \frac{1}{2} \tilde{\mathbf{d}}^T \mathbf{B} \tilde{\mathbf{d}} \\ \text{subject to} \quad & \mathbf{g}(\mathbf{y} + \mathbf{d}) + \nabla \mathbf{g}^T(\mathbf{y}) \tilde{\mathbf{d}} \leq \mathbf{0}, \\ & \mathbf{h}(\mathbf{y} + \mathbf{d}) + \nabla \mathbf{h}^T(\mathbf{y}) \tilde{\mathbf{d}} = \mathbf{0} \end{aligned}$$

△

The solutions $\tilde{\mathbf{d}}$ of these quadratic subproblems account for a second-order approximation of the constraints curvature and can reduce the l_1 merit function and the Lagrangian, if $\| \tilde{\mathbf{d}} \|$ is sufficiently small.

Motivated by the conditions for applying a correction step described by Mayne [53], Coope [16], and Fletcher [28] we calculate a correction step only if

$$\mathcal{L}(\mathbf{y} + \mathbf{d}, \boldsymbol{\lambda}_k, \boldsymbol{\mu}_k) < \mathcal{L}(\mathbf{y}, \boldsymbol{\lambda}_k, \boldsymbol{\mu}_k)$$

applies and finally use it only if

$$\mathcal{L}(\mathbf{y}+\mathbf{d}+\tilde{\mathbf{d}},\boldsymbol{\lambda}_k,\boldsymbol{\mu}_k) < \mathcal{L}(\mathbf{y}+\mathbf{d},\boldsymbol{\lambda}_k,\boldsymbol{\mu}_k)$$

applies.

2.4 Sensitivity Analysis

In the last section we studied the local optimal solution $\mathbf{y}^*$ of problem definition 2.9 with fixed boundary parameters. In the case that some of these parameters change, the solution $\mathbf{y}^*$ can no longer be regarded as locally optimal. But we are highly interested in solutions for different parameters to be able to find the best parameter set for a prescribed NLP. The probably most simple and accurate method to deal with such parameter variation problems is the reoptimization of the problem for several parameter combinations selected by a *brute force* approach (Beltracchi and Gabriele [5] and Büskens [10]). The brute force approach can handle arbitrary large disturbances in the parameters, but the interpretation of the impact of varying parameters on the results can be difficult due to the overwhelming amount of datasets generated, which makes this method not only computationally demanding but also provides no clear systematics in dealing with such problems.

To get a more systematic approach for such problems, we ask in a first step only for solutions of a NLP with little disturbances in the parameter set, which can be expressed as a first-order correction of the nominal solution. Such problems are known as *sensitivity problems* and widely applied in the optimization theory. We consider this problem class in a systematic fashion using sensitivity methods based on the Kuhn–Tucker conditions (2.33)–(2.37). These methods avoid the computational expense of reoptimization but are only local approximations of the disturbed solutions. In so doing, we introduce an parameter vector $\mathbf{p} = [p_1, p_2, \ldots, p_{N_p}]^T \in \mathcal{P}$ where $\mathcal{P} \subset \mathbb{R}^{N_p}$ is an open set of finite-dimensional parameters and define a modified problem definition of the standard form of the nonlinear programming problem (2.22).

Definition 2.23 (*Parametric Nonlinear Programming Problem (cf. Büskens and Maurer* [12])) The *parametric nonlinear programming problem* (NLP($\mathbf{p}$)) with equality and inequality constraints is given

$$\begin{aligned} \min_{\mathbf{y}\in\mathbb{R}^{N_y}} \quad & f(\mathbf{y},\mathbf{p}) \\ \text{subject to} \quad & \mathbf{g}(\mathbf{y},\mathbf{p}) \leq \mathbf{0} \\ & \mathbf{h}(\mathbf{y},\mathbf{p}) = \mathbf{0} \end{aligned} \tag{2.58}$$

where $f : \mathbb{R}^{N_y} \times \mathcal{P} \to \mathbb{R}$, $\mathbf{g} : \mathbb{R}^{N_y} \times \mathcal{P} \to \mathbb{R}^{N_g}$, and $\mathbf{h} : \mathbb{R}^{N_y} \times \mathcal{P} \to \mathbb{R}^{N_h}$ are real-valued and assumed to be twice continuously differentiable w.r.t. $\mathbf{y}$. The constraints are defined as

$$\mathbf{g}(\mathbf{y}, \mathbf{p}) := \begin{bmatrix} g_1(\mathbf{y}, \mathbf{p}) \\ \vdots \\ g_{N_g}(\mathbf{y}, \mathbf{p}) \end{bmatrix} \quad \text{and} \quad \mathbf{h}(\mathbf{y}, \mathbf{p}) := \begin{bmatrix} h_1(\mathbf{y}, \mathbf{p}) \\ \vdots \\ h_{N_h}(\mathbf{y}, \mathbf{p}) \end{bmatrix}.$$

△

Remark 2.6 For a reference parameter vector $\mathbf{p} = \mathbf{p}_0$ the problem formulation (2.58) is called an undisturbed or nominal nonlinear programming problem NLP($\mathbf{p}_0$).

The aim of the sensitivity analysis is the calculation of sensitivities of the parametric problem definition 2.23. The sensitivities are total differentials of the optimal solution dependent on the parameter vector $\mathbf{p}$. Based on the second-order sufficient conditions the main result of the sensitivity analysis is the sensitivity theorem which gives conditions for the existence and the properties of an optimal solution under presence of disturbing parameters. A perturbed optimization problem is given if the parameter vector $\mathbf{p}$ from (2.58) differs from the reference values $\mathbf{p}_0$ of the nominal optimization problem (2.22).

The cornerstone of the sensitivity analysis is laid by Fiacco [26]. We follow the definitions of this work and present the main results which are used in Chap. 13. The following assumptions are made: the functions $f(\cdot)$, $\mathbf{g}(\cdot)$, and $\mathbf{h}(\cdot)$ can be either linear or nonlinear functions of the NLP-variables $\mathbf{y}$ and the parameters $\mathbf{p}$ are held fixed during the optimization. Then, based on the problem formulation (2.58), the Lagrangian function is modified to $\mathcal{L} : \mathbb{R}^{N_y} \times \mathbb{R}^{N_g} \times \mathbb{R}^{N_h} \times \mathcal{P} \to \mathbb{R}$

$$\mathcal{L}(\mathbf{y}, \boldsymbol{\lambda}, \boldsymbol{\mu}, \mathbf{p}) := f(\mathbf{y}, \mathbf{p}) + \sum_{i=1}^{N_g} \lambda_i g_i(\mathbf{y}, \mathbf{p}) + \sum_{i=1}^{N_h} \mu_i h_i(\mathbf{y}, \mathbf{p})$$

where the Lagrange multipliers are defined as (2.28).

Remark 2.7 The necessary and sufficient conditions derived for the standard nonlinear programming problem (2.22) apply to the modified problem formulation (2.58) as well. Each constantly disturbed NLP($\mathbf{p}$) can be transformed to a standard nonlinear programming problem.

Analogous to the nominal problem case, the first-order KKT conditions (2.33)–(2.37) for the disturbed problem must hold. Thus, applying the KKT conditions for the problem formulation (2.58) we obtain a system

$$\mathbf{F}(\mathbf{y}, \boldsymbol{\lambda}, \boldsymbol{\mu}, \mathbf{p}) := \begin{pmatrix} \nabla_y \mathcal{L}(\mathbf{y}, \boldsymbol{\lambda}, \boldsymbol{\mu}, \mathbf{p}) \\ \boldsymbol{\Lambda} \cdot \mathbf{g}(\mathbf{y}, \mathbf{p}) \\ \mathbf{h}(\mathbf{y}, \mathbf{p}) \end{pmatrix} = \mathbf{0}_{(N_y + N_g + N_h) \times 1} \tag{2.59}$$

where $\boldsymbol{\Delta} := \text{diag}(\lambda_1, \ldots, \lambda_{N_g})$. At the nominal solution $\mathbf{F}(\mathbf{y}_0^*, \boldsymbol{\lambda}_0, \boldsymbol{\mu}_0, \mathbf{p}_0)$, the Jacobian of $\mathbf{F}(\cdot)$ with respect to the arguments $\mathbf{y}$, $\boldsymbol{\lambda}$, and $\boldsymbol{\mu}$ is given by

$$\frac{\partial \mathbf{F}}{\partial(\mathbf{y}, \boldsymbol{\lambda}, \boldsymbol{\mu})}(\mathbf{y}_0^*, \boldsymbol{\lambda}_0, \boldsymbol{\mu}_0, \mathbf{p}_0) = \begin{pmatrix} \nabla_y^2 \mathcal{L}(\mathbf{y}_0^*, \boldsymbol{\lambda}_0, \boldsymbol{\mu}_0, \mathbf{p}_0) & \nabla_y \mathbf{g}^T(\mathbf{y}_0^*, \mathbf{p}_0) & \nabla_y \mathbf{h}^T(\mathbf{y}_0^*, \mathbf{p}_0) \\ \boldsymbol{\Delta} \cdot \nabla_y \mathbf{g}(\mathbf{y}_0^*, \mathbf{p}_0) & \boldsymbol{\Gamma} & \mathbf{0}_{N_g \times N_h} \\ \nabla_y \mathbf{h}(\mathbf{y}_0^*, \mathbf{p}_0) & \mathbf{0}_{N_h \times N_g} & \mathbf{0}_{N_h \times N_h} \end{pmatrix} \tag{2.60}$$

where the matrix $\boldsymbol{\Gamma}$ is defined by $\boldsymbol{\Gamma} := \text{diag}(g_1(\mathbf{y}_0^*, \mathbf{p}_0), \ldots, g_{N_g}(\mathbf{y}_0^*, \mathbf{p}_0))$. The matrix (2.60) is the KKT matrix. Since we assume from SOSC (2.39) that $\nabla_y^2 \mathcal{L}(\mathbf{y}_0^*, \boldsymbol{\lambda}_0, \boldsymbol{\mu}_0, \mathbf{p}_0)$ is positive definite on a nonvanishing $\ker(\mathbf{A}_y^T(\mathbf{y}_0^*, \mathbf{p}_0))$ the KKT matrix is regular and therefore invertible. Hence, the classical implicit function theorem can be applied to (2.60) to obtain differentiable functions $\mathbf{y} : \mathcal{P} \to \mathbb{R}^{N_y}$, $\boldsymbol{\lambda} : \mathcal{P} \to \mathbb{R}^{N_g}$, and $\boldsymbol{\mu} : \mathcal{P} \to \mathbb{R}^{N_h}$ in a neighborhood of $\mathbf{p}_0$ such that $\lim_{\mathbf{p} \to \mathbf{p}_0} \mathbf{y}(\mathbf{p}) = \mathbf{y}_0^*$, $\lim_{\mathbf{p} \to \mathbf{p}_0} \boldsymbol{\lambda}(\mathbf{p}) = \boldsymbol{\lambda}_0$, and $\lim_{\mathbf{p} \to \mathbf{p}_0} \boldsymbol{\mu}(\mathbf{p}) = \boldsymbol{\mu}_0$ applies.

Theorem 2.5 (Implicit Function Theorem (Fiacco [26])) *Let the pair* $(\mathbf{y}_0^*, \mathbf{p}_0) \in \mathbb{R}^{N_y} \times \mathcal{P}$ *be given. Suppose the function* $\mathbf{K} : \mathbb{R}^{N_y} \times \mathcal{P} \to \mathbb{R}^{N_y}$ *with* $\mathbf{K}(\mathbf{y}_0^*, \mathbf{p}_0) = \mathbf{0}$ *is a continuously differentiable mapping at the point* $(\mathbf{y}_0^*, \mathbf{p}_0)$ *and its Jacobian with respect to* $\mathbf{y}$*, i.e.,* $\nabla_y \mathbf{K}(\mathbf{y}_0^*, \mathbf{p}_0)$*, is nonsingular and thus invertible. Then, there exists a neighborhood of* $\mathcal{Y} \subset \mathbb{R}^{N_y}$ *with* $\mathbf{y}_0^* \in \mathcal{Y}$ *and* $\mathcal{P} \subset \mathbb{R}^{N_p}$ *with* $\mathbf{p}_0 \in \mathcal{P}$ *and a differentiable and unique function* $\mathbf{y} : \mathcal{P} \to \mathcal{Y}$*, which satisfies the condition* $\mathbf{K}(\mathbf{y}(\mathbf{p}), \mathbf{p}) = \mathbf{0}$ *for all* $\mathbf{p} \in \mathcal{P}$ *with the unique solution* $\mathbf{y} = \mathbf{y}(\mathbf{p})$.

△

Proof The proof can be found in Oliveira [60]. □

In order to obtain explicit formulae for the sensitivity derivatives of the optimal solutions and Lagrange multipliers let us apply the Theorem 2.5.

Theorem 2.6 (Differentiability of Optimal Solutions (Büskens [10])) *Consider the parametric nonlinear problem (2.58) under the following assumptions:*

- *the functions* $f(\cdot)$, $\mathbf{g}(\cdot)$, *and* $\mathbf{h}(\cdot)$ *are twice continuously differentiable with respect to* $\mathbf{y}$ *in a neighborhood of* $\mathbf{y}_0$*. Also, let the gradients* $\nabla_y f$, $\nabla_y \mathbf{g}$, $\nabla_y \mathbf{h}$ *and the functions* $\mathbf{g}(\cdot)$ *and* $\mathbf{h}(\cdot)$ *be continuously differentiable with respect to* $\mathbf{p}$ *in the neighborhood of* $\mathbf{p}_0$*;*
- $\mathbf{y}_0^*$ *is a feasible solution such that the LICQ holds; and*
- *the triple* $(\mathbf{y}_0^*, \boldsymbol{\lambda}_0, \boldsymbol{\mu}_0)$ *is an optimal solution for the nominal problem formulation NLP*$(\mathbf{p}_0)$ *with the reference parameter vector* $\mathbf{p} = \mathbf{p}_0$ *which satisfies the SOSC of Theorem 2.4 and the strict complementarity condition (2.38).*

Then, the following applies:

- *there exists a neighborhood* $\mathcal{P} \subset \mathbb{R}^{N_p}$ *of* $\mathbf{p}_0$*, i.e.,* $\mathbf{p}_0 \in \mathcal{P}$*, and once continuously differentiable functions* $\mathbf{y} : \mathcal{P} \to \mathbb{R}^{N_y}$, $\boldsymbol{\lambda} : \mathcal{P} \to \mathbb{R}^{N_g}$, *and* $\boldsymbol{\mu} : \mathcal{P} \to \mathbb{R}^{N_h}$ *with the following properties:*

1. $\mathbf{y}(\mathbf{p}_0) = \mathbf{y}_0^*,\ \boldsymbol{\lambda}(\mathbf{p}_0) = \boldsymbol{\lambda}_0,\ \boldsymbol{\mu}(\mathbf{p}_0) = \boldsymbol{\mu}_0$;
2. *for all* $\mathbf{p} \in \mathcal{P}$, *the triple* $(\mathbf{y}(\mathbf{p}), \boldsymbol{\lambda}(\mathbf{p}), \boldsymbol{\mu}(\mathbf{p}))$ *satisfies the SOSC of Theorem 2.4 and the strict complementarity condition (2.38) for the perturbed NLP*$(\mathbf{p})$ *problem.* $\mathbf{y}(\mathbf{p})$ *is a unique local minimum of the NLP*$(\mathbf{p})$ *problem with the Lagrange multipliers* $\boldsymbol{\lambda}(\mathbf{p})$ *and* $\boldsymbol{\mu}(\mathbf{p})$.

- *for all* $\mathbf{p} \in \mathcal{P}$ *the following holds:*

1. *the set of active constraints is constant in* $\mathcal{P}$, *i.e.,*

$$\mathcal{I}(\mathbf{y}(\mathbf{p}), \mathbf{p}) \equiv \mathcal{I}(\mathbf{y}_0^*, \mathbf{p}_0), \quad \forall \mathbf{p} \in \mathcal{P};\ \textit{and}$$

2. *the Jacobian matrix of active constraints*

$$\mathbf{A}_y(\mathbf{y}(\mathbf{p}), \mathbf{p}) := \left(\nabla_y g_i(\mathbf{y}(\mathbf{p}), \mathbf{p}) \quad \nabla_y h_j(\mathbf{y}(\mathbf{p}), \mathbf{p})\right),$$

where the indices are given by $i \in \mathcal{I}(\mathbf{y}_0^*, \mathbf{p})$ *and* $j = 1, \ldots, N_h$, *has full rank, i.e.,*

$$\operatorname{rank}(\mathbf{A}_y(\mathbf{y}(\mathbf{p}), \mathbf{p})) = N_s, \quad \forall \mathbf{p} \in \mathcal{P}.$$

△

Proof A compact proof which exclusively considers the active constraints can be found in Büskens [10]. The original proof with the consideration of all constraints is presented in the work of Fiacco [26]. □

Remark 2.8 The requirement that the active sets of the nominal and disturbed problems keep unchanged can be quite restrictive. It is sometimes advantageous to remove some constraints, if possible, to enlarge the neighborhood $\mathcal{P}$ of $\mathbf{p}_0$.

Corollary 2.1 (Sensitivity Differentials) *Suppose Theorem 2.5 holds. Then, the sensitivity differentials are obtained by differentiation of the identity* $\mathbf{K}(\mathbf{y}(\mathbf{p}), \mathbf{p}) \equiv \mathbf{0}$ *at the nominal parameter* $\mathbf{p}_0$ *with respect to* $\mathbf{p}$ *which yields*

$$\nabla_y \mathbf{K}(\mathbf{y}_0^*, \mathbf{p}_0) \cdot \nabla_p \mathbf{y}(\mathbf{p}_0) + \nabla_p \mathbf{K}(\mathbf{y}_0^*, \mathbf{p}_0) = \mathbf{0}. \tag{2.61}$$

Rearranging (2.61) yields the explicit formulae

$$\nabla_p \mathbf{y}(\mathbf{p}_0) = -(\nabla_y \mathbf{K}(\mathbf{y}_0^*, \mathbf{p}_0))^{-1} \nabla_p \mathbf{K}(\mathbf{y}_0^*, \mathbf{p}_0).$$

△

Analogous to Corollary 2.1, we obtain the sensitivity differentials of the parametric nonlinear programming problem (2.58) by differentiation of its optimal solution (2.59) at the nominal parameter $\mathbf{p}_0$ with respect to all parameters $\mathbf{p}$. This yields the linear equations

$$\begin{pmatrix} \nabla_y^2 \mathcal{L}_0 & \nabla_y \mathbf{g}_0^T & \nabla_y \mathbf{h}_0^T \\ \boldsymbol{\Lambda} \cdot \nabla_y \mathbf{g}_0 & \boldsymbol{\Gamma} & \mathbf{0}_{N_g \times N_h} \\ \nabla_y \mathbf{h}_0 & \mathbf{0}_{N_h \times N_g} & \mathbf{0}_{N_h \times N_h} \end{pmatrix} \begin{pmatrix} \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) \\ \frac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) \\ \frac{\mathrm{d}\boldsymbol{\mu}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) \end{pmatrix} + \begin{pmatrix} \nabla_{yp} \mathcal{L}_0 \\ \boldsymbol{\Lambda} \cdot \nabla_p \mathbf{g}_0 \\ \nabla_p \mathbf{h}_0 \end{pmatrix} = \mathbf{0}_{(N_y + N_g + N_h) \times N_p}$$

where $\mathcal{L}_0 := \mathcal{L}(\mathbf{y}_0^*, \boldsymbol{\lambda}_0, \boldsymbol{\mu}_0, \mathbf{p}_0)$, $\mathbf{g}_0 := \mathbf{g}(\mathbf{y}_0^*, \mathbf{p}_0)$, and $\mathbf{h}_0 := \mathbf{h}(\mathbf{y}_0^*, \mathbf{p}_0)$ are defined for the sake of a better readability.

Hence, the explicit formulae for the sensitivity differentials are obtained by the following corollary.

Corollary 2.2 (Sensitivity Differentials of the Optimal Solution) *Suppose Theorem 2.6 holds. Then, the sensitivity differentials of the optimal solution (2.59) at the nominal parameter* $\mathbf{p}_0$ *with respect to all parameters* $\mathbf{p}$ *are given by*

$$\begin{pmatrix} \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) \\ \frac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) \\ \frac{\mathrm{d}\boldsymbol{\mu}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) \end{pmatrix} = - \begin{pmatrix} \nabla_y^2 \mathcal{L}_0 & \nabla_y \mathbf{g}_0^T & \nabla_y \mathbf{h}_0^T \\ \boldsymbol{\Lambda} \cdot \nabla_y \mathbf{g}_0 & \boldsymbol{\Gamma} & \mathbf{0}_{N_g \times N_h} \\ \nabla_y \mathbf{h}_0 & \mathbf{0}_{N_h \times N_g} & \mathbf{0}_{N_h \times N_h} \end{pmatrix}^{-1} \begin{pmatrix} \nabla_{yp} \mathcal{L}_0 \\ \boldsymbol{\Lambda} \cdot \nabla_p \mathbf{g}_0 \\ \nabla_p \mathbf{h}_0 \end{pmatrix}.$$

△

One might think that the calculation of the sensitivity differentials is a byproduct of all modern iterative solution algorithms that apply Newton's method to the KKT system, e.g., SQP methods, and hence the appearance of the Kuhn–Tucker matrix. However, the exact computation of the Hessian of the Lagrangian $\nabla_y^2 \mathcal{L}_0$ of (2.60) in every iteration is far too expensive for large-scale problems. In general, the direct approximation of (2.60) by the wealth of Quasi-Newton methods is not possible because they do not converge to the KKT matrix due to their low rank approximation of the Hessian of the Lagrangian. This makes the embedding of the sensitivity analysis in modern solution algorithms a challenging task.

A more straightforward way is proposed by Büskens and Maurer [12] using the calculation of the sensitivity differentials as a *post-optimal* analysis.

2.4.1 Sensitivity Analysis of the Objective Function and Constraints

Beside the sensitivity of the optimal solution $\mathbf{y}^*$ the sensitivity of the objective function and constraints are also interesting. The calculation of these sensitivities can be accomplished by simply extending Corollary 2.2 to *associated functions*. Associated functions can be interpreted as functions which provide information about the solution of disturbed NLP($\mathbf{p}$).

Definition 2.24 (*Associated Function*) Let $\tilde{\mathbf{a}} : \mathbb{R}^{N_y} \times \mathbb{R}^{N_g} \times \mathbb{R}^{N_h} \times \mathcal{P} \to \mathbb{R}^{N_a}$ be a continuously differentiable function and let the assumptions of the sensitivity Theorem 2.6 hold. Hence, there exists a neighborhood of $\mathbf{p}_0$, $\mathcal{P} \subset \mathbb{R}^{N_p}$, to define differentiable functions $\mathbf{y} : \mathcal{P} \to \mathbb{R}^{N_y}$, $\boldsymbol{\lambda} : \mathcal{P} \to \mathbb{R}^{N_g}$, and $\boldsymbol{\mu} : \mathcal{P} \to \mathbb{R}^{N_h}$. Then, the function

$$\mathbf{a}(\mathbf{p}) := \tilde{\mathbf{a}}(\mathbf{y}(\mathbf{p}), \boldsymbol{\lambda}(\mathbf{p}), \boldsymbol{\mu}(\mathbf{p}), \mathbf{p})$$

is denoted as an associated function to NLP($\mathbf{p}$).

△

According to Definition 2.24, if the function $f(\cdot)$, $\mathbf{g}(\cdot)$, and $\mathbf{h}(\cdot)$ are continuously differentiable with respect to the parameter vector $\mathbf{p}$, then the Lagrangian $\mathcal{L}(\cdot)$ is an associated function. Moreover, the objective function $\mathbf{f}(\cdot)$ and the constraints $\mathbf{g}(\cdot)$ and $\mathbf{h}(\cdot)$ are associated functions too. Using Corollary 2.2 and the chain rules of differentiation we obtain the sensitivity theorem for the associated function.

Theorem 2.7 (Sensitivity of Associated Functions) *Let $\mathbf{a} : \mathcal{P} \to \mathbb{R}^{N_a}$ be an associated function. Further, let the assumptions of the sensitivity Theorem 2.6 for the disturbed NLP($\mathbf{p}$) with the optimal solution $\mathbf{y}_0^*$, $\boldsymbol{\lambda}_0$, $\boldsymbol{\mu}_0$ for $\mathbf{p} = \mathbf{p}_0$ hold. Then*

$$\frac{\mathrm{d}\mathbf{a}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) = -\begin{pmatrix}\nabla_y \mathbf{a} & \nabla_\lambda \mathbf{a} & \nabla_\mu \mathbf{a}\end{pmatrix}\left[\begin{pmatrix}\nabla_y^2 \mathcal{L}_0 & \nabla_y \mathbf{g}_0^T & \nabla_y \mathbf{h}_0^T \\ \boldsymbol{\Lambda} \cdot \nabla_y \mathbf{g}_0 & \boldsymbol{\Gamma} & \mathbf{0}_{N_g \times N_h} \\ \nabla_y \mathbf{h}_0 & \mathbf{0}_{N_h \times N_g} & \mathbf{0}_{N_h \times N_h}\end{pmatrix}^{-1}\begin{pmatrix}\nabla_{yp} \mathcal{L}_0 \\ \boldsymbol{\Lambda} \cdot \nabla_p \mathbf{g}_0 \\ \nabla_p \mathbf{h}_0\end{pmatrix}\right] + \nabla_p \mathbf{a}$$

is the sensitivity differential of the associated function.

△

Proof The function $\mathbf{a}(\cdot)$ is a differentiable mapping with respect to the argument $\mathbf{y}(\cdot)$, $\boldsymbol{\lambda}(\cdot)$, $\boldsymbol{\mu}(\cdot)$, and $\mathbf{p}$. Together with Corollary 2.2 one obtains the result of Theorem 2.7 straightforwardly. □

With this result it is easy to state the sensitivities of the objective function and the constraints.

Corollary 2.3 (Sensitivity Differentials of the Constraints) *Suppose Theorem 2.6 holds. Then, the sensitivity differentials of constraints are given by*

$$\begin{aligned}\frac{\mathrm{d}\mathbf{g}}{\mathrm{d}\mathbf{p}}(\mathbf{y}_0^*, \mathbf{p}_0) &= -\begin{pmatrix}\nabla_y \mathbf{g}_0 & \mathbf{0}_{N_g \times N_g} & \mathbf{0}_{N_g \times N_h}\end{pmatrix}\left[\begin{pmatrix}\nabla_y^2 \mathcal{L}_0 & \nabla_y \mathbf{g}_0^T & \nabla_y \mathbf{h}_0^T \\ \boldsymbol{\Lambda} \cdot \nabla_y \mathbf{g}_0 & \boldsymbol{\Gamma} & \mathbf{0}_{N_g \times N_h} \\ \nabla_y \mathbf{h}_0 & \mathbf{0}_{N_h \times N_g} & \mathbf{0}_{N_h \times N_h}\end{pmatrix}^{-1}\begin{pmatrix}\nabla_{yp} \mathcal{L}_0 \\ \boldsymbol{\Lambda} \cdot \nabla_p \mathbf{g}_0 \\ \nabla_p \mathbf{h}_0\end{pmatrix}\right] + \nabla_p \mathbf{g}_0 \\ &= \nabla_y \mathbf{g}_0 \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) + \nabla_p \mathbf{g}_0 \end{aligned} \tag{2.62}$$

$$\begin{aligned}\frac{\mathrm{d}\mathbf{h}}{\mathrm{d}\mathbf{p}}(\mathbf{y}_0^*, \mathbf{p}_0) &= -\begin{pmatrix}\nabla_y \mathbf{h}_0 & \mathbf{0}_{N_h \times N_g} & \mathbf{0}_{N_h \times N_h}\end{pmatrix}\left[\begin{pmatrix}\nabla_y^2 \mathcal{L}_0 & \nabla_y \mathbf{g}_0^T & \nabla_y \mathbf{h}_0^T \\ \boldsymbol{\Lambda} \cdot \nabla_y \mathbf{g}_0 & \boldsymbol{\Gamma} & \mathbf{0}_{N_g \times N_h} \\ \nabla_y \mathbf{h}_0 & \mathbf{0}_{N_h \times N_g} & \mathbf{0}_{N_h \times N_h}\end{pmatrix}^{-1}\begin{pmatrix}\nabla_{yp} \mathcal{L}_0 \\ \boldsymbol{\Lambda} \cdot \nabla_p \mathbf{g}_0 \\ \nabla_p \mathbf{h}_0\end{pmatrix}\right] + \nabla_p \mathbf{h}_0 \\ &= \nabla_y \mathbf{h}_0 \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) + \nabla_p \mathbf{h}_0 = \mathbf{0}. \end{aligned} \tag{2.63}$$

In particular

$$\frac{\mathrm{d}g_{0_i}}{\mathrm{d}\mathbf{p}}(\mathbf{y}_0^*, \mathbf{p}_0) = \mathbf{0}, \quad \forall i \in \mathcal{I}(\mathbf{y}_0^*, \mathbf{p}_0)$$

holds.

△

Proof Equations (2.62)–(2.63) are obtained directly from Theorem 2.7. It remains to show that the sensitivity differentials are zero for active constraints. It follows for every g_{0_i} with $i \in \mathcal{I}(\mathbf{y}_0^*, \mathbf{p}_0)$ from Corollary 2.2 that

$$\nabla_y g_{0_i} \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) = -\nabla_p g_{0_i}$$

and therefore

$$\frac{\mathrm{d}g_i}{\mathrm{d}\mathbf{p}}(\mathbf{y}_0^*, \mathbf{p}_0) = -\nabla_p g_{0_i} + \nabla_p g_{0_i} = \mathbf{0}_{1\times N_p}.$$

The proof can be applied to the equality constraints in the same manner. □

Corollary 2.4 (Sensitivity Differential of the Objective Function) *Suppose Theorem 2.6 holds. Additionally, let the function $f(\cdot)$ be once continuously differentiable with respect to* $\mathbf{p}$. *Then, the sensitivity differential of the objective function is given by*

$$\begin{aligned}\frac{\mathrm{d}f}{\mathrm{d}\mathbf{p}}(\mathbf{y}_0^*, \mathbf{p}_0) &= -\begin{pmatrix}\nabla_y f_0 & \mathbf{0}_{1\times N_g} & \mathbf{0}_{1\times N_h}\end{pmatrix}\left[\begin{pmatrix}\nabla_y^2\mathcal{L}_0 & \nabla_y\mathbf{g}_0^T & \nabla_y\mathbf{h}_0^T\\ \boldsymbol{\Delta}\cdot\nabla_y\mathbf{g}_0 & \boldsymbol{\Gamma} & \mathbf{0}_{N_g\times N_h}\\ \nabla_y\mathbf{h}_0 & \mathbf{0}_{N_h\times N_g} & \mathbf{0}_{N_h\times N_h}\end{pmatrix}^{-1}\begin{pmatrix}\nabla_{yp}\mathcal{L}_0\\ \boldsymbol{\Delta}\cdot\nabla_p\mathbf{g}_0\\ \nabla_p\mathbf{h}_0\end{pmatrix}\right] + \nabla_p f_0\\ &= \nabla_y f_0 \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) + \nabla_p f_0 \qquad (2.64)\\ &= \nabla_p \mathcal{L}_0 \qquad (2.65)\end{aligned}$$

with $f_0 := f(\mathbf{y}_0^*, \mathbf{p}_0)$.

△

Proof Equation (2.64) follows analogously from Corollary 2.3. With the first-order necessary conditions from Theorem 2.3 it follows that

$$\frac{\mathrm{d}f}{\mathrm{d}\mathbf{p}}(\mathbf{y}_0^*, \mathbf{p}_0) \overset{\text{Thm. 2.3}}{=} -\boldsymbol{\lambda}_0^T\nabla_y\mathbf{g}_0\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) - \boldsymbol{\mu}_0^T\nabla_y\mathbf{h}_0\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) + \nabla_p f_0.$$

Because of the strict complementarity condition (2.38), the inactive constraints can be disregarded. Finally, it follows from Corollary 2.3 and the definition of the Lagrangian 2.13 that

$$\begin{aligned}\frac{\mathrm{d}f}{\mathrm{d}\mathbf{p}}(\mathbf{y}_0^*, \mathbf{p}_0) &= -\boldsymbol{\lambda}_0^T \nabla_y \mathbf{g}_0 \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) - \boldsymbol{\mu}_0^T \nabla_y \mathbf{h}_0 \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) + \nabla_p f_0 \\ &\overset{\text{Coroll. 2.3}}{=} -\boldsymbol{\lambda}_0^T(-\nabla_p \mathbf{g}_0) - \boldsymbol{\mu}_0^T(-\nabla_p \mathbf{h}_0) + \nabla_p f_0 \\ &= \nabla_p \left(f_0 + \boldsymbol{\lambda}_0^T \mathbf{g}_0 + \boldsymbol{\mu}_0^T \mathbf{h}_0\right) \\ &\overset{\text{Def. 2.13}}{=} \nabla_p \mathcal{L}_0.\end{aligned}$$

□

Remark 2.9 One can realize that the sensitivity differential of the objective function using (2.64) needs second-order information to calculate $(\mathrm{d}\mathbf{y}/\mathrm{d}\mathbf{p})(\mathbf{p}_0)$. In contrast, (2.65) requires only first-order information. This has the consequence that for the evaluation of the objective function sensitivities the solution of the linear equation system and the differential of second order are not required which allows for a more efficient computation.

A further differentiation of this identity yields the second-order sensitivities of the objective function.

Corollary 2.5 (Second-Order Sensitivity Differential of the Objective Function) *Suppose Theorem 2.6 holds. Additionally, let the functions $f(\cdot)$, $\mathbf{g}(\cdot)$, and $\mathbf{h}(\cdot)$ be twice continuously differentiable with respect to $\mathbf{p}$. Then, the second-order sensitivity differential of the objective function is given by*

$$\frac{\mathrm{d}^2 f}{\mathrm{d}\mathbf{p}^2}(\mathbf{y}_0^*, \mathbf{p}_0) = \left(\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_{yp} \mathcal{L}_0 + \left(\frac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_p \mathbf{g}_0 + \left(\frac{\mathrm{d}\boldsymbol{\mu}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_p \mathbf{h}_0 + \nabla_p^2 \mathcal{L}_0 \tag{2.66}$$

$$= 2\left(\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_{yp} \mathcal{L}_0 + \left(\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_y^2 \mathcal{L}_0 \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) + \nabla_p^2 \mathcal{L}_0. \tag{2.67}$$

△

Proof Using $\nabla_p \mathcal{L}_0 = \nabla_p f_0 + \boldsymbol{\lambda}_0^T \nabla_p \mathbf{g}_0 + \boldsymbol{\mu}_0^T \nabla_p \mathbf{h}_0$, Eq. (2.66) is obtained by differentiation of $(\mathrm{d}f/\mathrm{d}\mathbf{p})(\mathbf{p}_0)$. We obtain

$$\begin{aligned}\frac{\mathrm{d}^2 f}{\mathrm{d}\mathbf{p}^2}(\mathbf{y}_0^*, \mathbf{p}_0) &= \frac{\mathrm{d}\left(\nabla_p f_0 + \boldsymbol{\lambda}_0^T \nabla_p \mathbf{g}_0 + \boldsymbol{\mu}_0^T \nabla_p \mathbf{h}_0\right)}{\mathrm{d}\mathbf{p}} \\ &= \left(\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_{yp} f_0 + \nabla_p^2 f_0 + \left(\frac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_p \mathbf{g}_0 \\ &+ \left(\frac{\mathrm{d}\boldsymbol{\mu}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_p \mathbf{h}_0 + \left(\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \left[\boldsymbol{\lambda}_0^T, \nabla_p \mathbf{g}_0\right] \\ &+ \left(\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \left[\boldsymbol{\mu}_0^T, \nabla_p \mathbf{h}_0\right] + \left[\boldsymbol{\lambda}_0^T, \nabla_p \mathbf{g}_0\right] + \left[\boldsymbol{\mu}_0^T, \nabla_p \mathbf{h}_0\right] \\ &= \left(\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_{yp} \mathcal{L}_0 + \left(\frac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_p \mathbf{g}_0 + \left(\frac{\mathrm{d}\boldsymbol{\mu}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_p \mathbf{h}_0 + \nabla_p^2 \mathcal{L}_0\end{aligned}$$

where the operator $[\mathbf{b}(\mathbf{x}), \mathbf{A}(\mathbf{x})]$ with the arguments $\mathbf{b}(\mathbf{x}) \in \mathbb{R}^{N_m}$ and $\mathbf{A}(\mathbf{x}) \in \mathbb{R}^{N_m+N_p}$ is defined by

$$[\mathbf{b}(\mathbf{x}), \mathbf{A}(\mathbf{x})] := \left[\sum_{i=1}^{N_m} b_i(\mathbf{x}) \nabla_x a_{i,1}(\mathbf{x}), \ldots, \sum_{i=1}^{N_m} b_i(\mathbf{x}) \nabla_x a_{i,p}(\mathbf{x}) \right].$$

where the variable $\mathbf{x}$ of the operator is a substituted variable for $\mathbf{y}(\cdot)$ or $\mathbf{p}$, respectively.

This completes the proof of the first Eq. (2.66) of the second-order sensitivities of the objective function.

Now follows the proof of (2.67). Using $\nabla_y \mathcal{L}_0 = \nabla_y f_0 + \boldsymbol{\lambda}^T \nabla_y \mathbf{g}_0 + \boldsymbol{\mu}^T \nabla_y \mathbf{h}_0$ and applying Theorem 2.7 yields the following first equation straightforwardly. The second and third equations follow from Corollary 2.3.

$$\nabla_y^2 \mathcal{L}_0 \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) + \nabla_y \mathbf{g}_0^T \frac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) + \nabla_y \mathbf{h}_0^T \frac{\mathrm{d}\boldsymbol{\mu}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) = -\nabla_{zp} \mathcal{L}_0 \tag{2.68}$$

$$\boldsymbol{\Delta} \cdot \nabla_y \mathbf{g}_0 \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) = -\boldsymbol{\Delta} \cdot \nabla_p \mathbf{g}_0 \tag{2.69}$$

$$\nabla_y \mathbf{h}_0 \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) = -\nabla_p \mathbf{h}_0. \tag{2.70}$$

Multiplying the first Eq. (2.68) from the right-hand side by $(\mathrm{d}\mathbf{y}/\mathrm{d}\mathbf{p})(\mathbf{p}_0)$, the second equation (2.69) from the left-hand side by $(\mathrm{d}\boldsymbol{\lambda}/\mathrm{d}\mathbf{p})^T(\mathbf{p}_0)$, and the third equation (2.70) from the left-hand side by $(\mathrm{d}\boldsymbol{\mu}/\mathrm{d}\mathbf{p})^T(\mathbf{p}_0)$ and summing them up yields the following relationships with

$$\left(\frac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_y \mathbf{g}_0 \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) = -\left(\frac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_p \mathbf{g}_0 \tag{2.71}$$

$$\left(\frac{\mathrm{d}\boldsymbol{\mu}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_y \mathbf{h}_0 \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) = -\left(\frac{\mathrm{d}\boldsymbol{\mu}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_p \mathbf{h}_0. \tag{2.72}$$

Putting (2.71)–(2.72) into the first sensitivity Eq. (2.66), then rearranging yields

$$\begin{aligned}
&\frac{\mathrm{d}^2 f}{\mathrm{d}\mathbf{p}^2}(\mathbf{y}_0^*, \mathbf{p}_0) = \\
&= \left(\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_{yp} \mathcal{L}_0 + \left(\frac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_p \mathbf{g}_0 + \left(\frac{\mathrm{d}\boldsymbol{\mu}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_p \mathbf{h}_0 + \nabla_p^2 \mathcal{L}_0 \\
&= \left(\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_{yp} \mathcal{L}_0 - \left(\left(\frac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_y \mathbf{g}_0 + \left(\frac{\mathrm{d}\boldsymbol{\mu}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_y \mathbf{h}_0 \right) \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) + \nabla_p^2 \mathcal{L}_0 \\
&= \left(\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_{yp} \mathcal{L}_0 + \left(\nabla_y^2 \left[\left(\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \mathcal{L}_0 \right] + \nabla_{yp} \mathcal{L}_0^T \right) \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) + \nabla_p^2 \mathcal{L}_0 \\
&= 2 \left(\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_{yp} \mathcal{L}_0 + \left(\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}\right)^T (\mathbf{p}_0) \cdot \nabla_y^2 \left[\mathcal{L}_0 \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) \right] + \nabla_p^2 \mathcal{L}_0
\end{aligned}$$

with the assumption that

$$\left(\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}\right)^T(\mathbf{p}_0)\nabla_{yp}\mathcal{L}_0 = \left(\left(\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}\right)^T(\mathbf{p}_0)\nabla_{yp}\mathcal{L}_0\right)^T = \nabla_{yp}\mathcal{L}_0^T\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0).$$

□

The numerical advantage of (2.67) in contrast with (2.66) is its independence from the Jacobian of the constraints and the Jacobian of the Lagrange multipliers.

2.4.2 Linear Perturbations

A special case of parameter disturbance is the linear perturbation in the constraints. If the objective function is independent of the parameter vector $\mathbf{p}$ and the constraints involve linear perturbation we obtain an optimization problem of the form

$$\begin{aligned} \min_{\mathbf{y}\in\mathbb{R}^{N_y}} \quad & f(\mathbf{y}) \\ \text{subject to} \quad & g_i(\mathbf{y}) - \mathbf{p}_{[i]} \leq 0, \quad i = 1,\ldots,N_g \\ & h_j(\mathbf{y}) - \mathbf{p}_{[N_g+j]} = 0, \quad j = 1,\ldots,N_h \end{aligned} \tag{2.73}$$

where $\mathbf{p} \in \mathbb{R}^{N_g+N_h}$ is the linear perturbation vector. Then, we obtain immediately the sensitivities of the optimal solution, the constraints, and the objective function with the following corollary.

Corollary 2.6 (Sensitivity Differentials for Linear Perturbation in the Constraints) *Let the problem formulation (2.73) with linear perturbation in the constraints be given where $\mathbf{p}_0 \in \mathbb{R}^{N_g+N_h}$ is the reference parameter vector. Assume that the implicit function Theorem 2.5 for the problem formulation (2.73) holds. Then, the sensitivity differentials with respect to $\mathbf{p}$ are obtained by:*

1. ***optimal solution**:*

$$\begin{pmatrix} \dfrac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}_{[i]}}(\mathbf{p}_0) \\ \dfrac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}\mathbf{p}_{[i]}}(\mathbf{p}_0) \\ \dfrac{\mathrm{d}\boldsymbol{\mu}}{\mathrm{d}\mathbf{p}_{[i]}}(\mathbf{p}_0) \end{pmatrix} = \begin{pmatrix} \nabla_y^2\mathcal{L}_0 & \nabla_y\mathbf{g}_0^T & \nabla_y\mathbf{h}_0^T \\ \boldsymbol{\Lambda}\cdot\nabla_y\mathbf{g}_0 & \boldsymbol{\Gamma} & 0 \\ \nabla_y\mathbf{h}_0 & 0 & 0 \end{pmatrix}^{-1}_{[\cdot,N_y+i]}$$

2. ***constraints***:

$$\frac{\mathrm{d}\mathbf{g}}{\mathrm{d}\mathbf{p}}(\mathbf{y}_0^*, \mathbf{p}_0) = \nabla_y \mathbf{g}_0 \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0) - \mathbf{I}_{N_g};$$

3. ***objective function***:

$$\frac{\mathrm{d}f}{\mathrm{d}\mathbf{p}}(\mathbf{y}_0^*, \mathbf{p}_0) = \nabla_p \mathcal{L}_0 = \begin{pmatrix} \boldsymbol{\lambda}_0 \\ \boldsymbol{\mu}_0 \end{pmatrix}.$$

In particular $\frac{\mathrm{d}f}{\mathrm{d}\mathbf{p}_{[i]}}(\mathbf{y}_0^*, \mathbf{p}_0) = 0$ *for* $i \notin \mathcal{I}(\mathbf{y}_0^*, \mathbf{p}_0)$ *holds; and*

4. ***second-order sensitivity of the object function***:

$$\frac{\mathrm{d}^2 f}{\mathrm{d}\mathbf{p}^2}(\mathbf{y}_0^*, \mathbf{p}_0) = \begin{pmatrix} \frac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}\mathbf{p}}(\mathbf{y}_0^*, \mathbf{p}_0) \\ \frac{\mathrm{d}\boldsymbol{\mu}}{\mathrm{d}\mathbf{p}}(\mathbf{y}_0^*, \mathbf{p}_0) \end{pmatrix}.$$

In particular $\frac{\mathrm{d}^2 f}{\mathrm{d}\mathbf{p}_{[i]}^2}(\mathbf{y}_0^*, \mathbf{p}_0) = 0$ *for* $i \notin \mathcal{I}(\mathbf{y}_0^*, \mathbf{p}_0)$ *holds.*

△

Proof The proof follows directly by insertion. □

Remark 2.10 The same procedure can be applied to the objective function of the form $f(\mathbf{y}) - \mathbf{r}^T\mathbf{y}$. The interested reader may refer to Büskens [10] for more details.

2.4.3 Approximation of the Perturbed Solution

The calculated sensitivity differentials can now be used to approximate the perturbed solutions of the optimization problem, if the perturbation of the nominal parameters $\Delta\mathbf{p} := \mathbf{p} - \mathbf{p}_0$ is small enough, such that the set of active constraints does not change. How to calculate the confidence region for the active set will be the topic of Sect. 2.4.4.

The approximated solution $(\tilde{\mathbf{y}}(\mathbf{p}), \tilde{\boldsymbol{\lambda}}(\mathbf{p}), \tilde{\boldsymbol{\mu}}(\mathbf{p}))$ can be calculated very fast by the use of the sensitivity differentials and therefore this method is suitable for an online optimization strategy, which means that the approximated solutions can be calculated in real-time even on the electronic control unit of the vehicle, if the optimal solution $\mathbf{y}_0^*$ and the sensitivity differentials $(\mathrm{d}\mathbf{y}/\mathrm{d}\mathbf{p})(\mathbf{p}_0)$ are calculated offline before.

Instead of restarting the optimization procedure again, as it is done by the brute force method, the perturbed solution triple $(\mathbf{y}(\mathbf{p}), \boldsymbol{\lambda}(\mathbf{p}), \boldsymbol{\mu}(\mathbf{p}))$ is approximated by a first-order Taylor series expansion according to

$$\mathbf{y}(\mathbf{p}) \approx \tilde{\mathbf{y}}(\mathbf{p}) := \mathbf{y}_0^* + \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0)\Delta\mathbf{p} \tag{2.74}$$

where $\tilde{\boldsymbol{\lambda}}(\cdot)$ and $\tilde{\boldsymbol{\mu}}(\cdot)$ can be obtained analogously to (2.74). The computation of (2.74) is very fast because the first-order Taylor approximation uses only matrix-vector multiplication and vector addition.

For the objective function a Taylor approximation of second-order is even possible with

$$f(\mathbf{y}(\mathbf{p}), \mathbf{p}) \approx f(\tilde{\mathbf{y}}(\mathbf{p}), \mathbf{p}) := f_0 + \frac{\mathrm{d}f}{\mathrm{d}\mathbf{p}}(\mathbf{y}_0^*, \mathbf{p}_0)\Delta\mathbf{p} + \frac{1}{2}\Delta\mathbf{p}^T \frac{\mathrm{d}^2 f}{\mathrm{d}\mathbf{p}^2}(\mathbf{y}_0^*, \mathbf{p}_0)\Delta\mathbf{p} \tag{2.75}$$

where $(\mathrm{d}f/\mathrm{d}\mathbf{p})(\mathbf{y}_0^*, \mathbf{p}_0)$ and $(\mathrm{d}^2 f/\mathrm{d}\mathbf{p}^2)(\mathbf{y}_0^*, \mathbf{p}_0)$ are calculated by (2.65) and (2.67), respectively. The approximated solution of the disturbed problem can deviate from the optimal solution of the disturbed problem. The following theorem shows that the error of the disturbed solution $\tilde{\mathbf{y}}(\mathbf{p})$, the functions $f(\tilde{\mathbf{y}}(\mathbf{p}), \mathbf{p})$, $\mathbf{g}(\tilde{\mathbf{y}}(\mathbf{p}), \mathbf{p})$, $\mathbf{h}(\tilde{\mathbf{y}}(\mathbf{p}), \mathbf{p})$, the multipliers $\tilde{\boldsymbol{\lambda}}(\mathbf{p})$, $\tilde{\boldsymbol{\mu}}(\mathbf{p})$, and the Lagrangian $\mathcal{L}(\tilde{\mathbf{y}}(\mathbf{p}), \tilde{\boldsymbol{\lambda}}(\mathbf{p}), \tilde{\boldsymbol{\mu}}(\mathbf{p}), \mathbf{p})$ is of second-order with respect to the disturbance $\Delta\mathbf{p}$. Again, only the active constraints of the inequality set are important. Therefore, we define the vector of active inequality constraints as

$$\mathbf{g}_a := (g_i)_{i \in \mathcal{I}(\mathbf{y}_0^*, \mathbf{p}_0)}. \tag{2.76}$$

Theorem 2.8 (Error Estimation) *Let the assumptions from the sensitivity Theorem 2.6 hold. Let the functions $f(\cdot)$, $\mathbf{g}(\cdot)$, and $\mathbf{h}(\cdot)$ be three times continuously differentiable with respect to $\mathbf{y}(\cdot)$ and $\mathbf{p}$. Then, there exists a neighborhood of $\mathbf{p}_0$ with $\mathcal{P}$ such that the following error estimations hold for all $\mathbf{p} = \mathbf{p}_0 + \Delta\mathbf{p} \in \mathcal{P}$:*

$$\begin{aligned}
\|\mathbf{y}(\mathbf{p}) - \tilde{\mathbf{y}}(\mathbf{p})\| &= \mathcal{O}(\|\Delta\mathbf{p}\|^2) \\
\|f(\mathbf{y}(\mathbf{p}), \mathbf{p}) - f(\tilde{\mathbf{y}}(\mathbf{p}), \mathbf{p})\| &= \mathcal{O}(\|\Delta\mathbf{p}\|^2) \\
\|\mathbf{g}_a(\tilde{\mathbf{y}}(\mathbf{p}), \mathbf{p})\| &= \mathcal{O}(\|\Delta\mathbf{p}\|^2) \\
\|\mathbf{h}(\tilde{\mathbf{y}}(\mathbf{p}), \mathbf{p})\| &= \mathcal{O}(\|\Delta\mathbf{p}\|^2) \\
\left\|\boldsymbol{\lambda}(\mathbf{p}) - \tilde{\boldsymbol{\lambda}}(\mathbf{p})\right\| &= \mathcal{O}(\|\Delta\mathbf{p}\|^2) \\
\|\boldsymbol{\mu}(\mathbf{p}) - \tilde{\boldsymbol{\mu}}(\mathbf{p})\| &= \mathcal{O}(\|\Delta\mathbf{p}\|^2) \\
\left\|\nabla_y \mathcal{L}(\tilde{\mathbf{y}}(\mathbf{p}), \tilde{\boldsymbol{\lambda}}(\mathbf{p}), \tilde{\boldsymbol{\mu}}(\mathbf{p}), \mathbf{p})\right\| &= \mathcal{O}(\|\Delta\mathbf{p}\|^2).
\end{aligned}$$

Proof The proof can be found in Büskens [11]. □

2.4.4 Approximation of the Confidence Region

The real-time strategy presented before can only be applied to perturbations that keep the set of active indices unchanged. This restriction is necessary since a change in the active set of constraints might result in a violation of the regularity condition and thus render the sensitivity differentials incorrect or even invalid. This means that the Eqs. (2.74) and (2.75) are only bounded by the assumption that the active set remains the same. An estimate of when the active set will change can be made by examining the Lagrange multipliers of the active inequality constraints (2.76) and linear approximations of the inactive constraints. In order to obtain linear approximations of the inactive constraints we need linear approximations of the Lagrange multiplier $\boldsymbol{\lambda}(\mathbf{p})$ and inequality constraint $\mathbf{g}(\mathbf{p})$ with

$$\begin{aligned} \boldsymbol{\lambda}(\mathbf{p}) &\approx \tilde{\boldsymbol{\lambda}}(\mathbf{p}) := \hat{\boldsymbol{\lambda}} + \frac{\mathrm{d}\boldsymbol{\lambda}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0)\Delta\mathbf{p} \\ \mathbf{g}(\mathbf{p}) &\approx \tilde{\mathbf{g}}(\mathbf{p}) := \hat{\mathbf{g}} + \frac{\mathrm{d}\mathbf{g}}{\mathrm{d}\mathbf{p}}(\mathbf{y}_0^*, \mathbf{p}_0)\Delta\mathbf{p}. \end{aligned} \tag{2.77}$$

An inactive constraint ($g_i \neq 0, \lambda_i = 0$) becomes active if g_i vanishes under the influence of disturbance. Using the Taylor series expansion from (2.77), this means

$$0 = g_i(\mathbf{p}) \approx \hat{g}_i + \frac{\mathrm{d}g_i}{\mathrm{d}\mathbf{p}}(\mathbf{y}_0^*, \mathbf{p}_0)\Delta\mathbf{p}, \qquad i \notin \mathcal{I}(\mathbf{y}_0^*, \mathbf{p}_0).$$

Accordingly, the disturbance $\Delta\mathbf{p}_{[j]}$ which causes g_i to enter the set of active indices can be approximated by using the linear prediction

$$\Delta\mathbf{p}_{[j]} \approx -\frac{\hat{g}_i}{\dfrac{\mathrm{d}g_i}{\mathrm{d}\mathbf{p}_{[j]}}(\mathbf{y}_0^*, \mathbf{p}_0)}, \qquad i \notin \mathcal{I}(\mathbf{y}_0^*, \mathbf{p}_0), \quad j \in \{1, \ldots, N_p\}. \tag{2.78}$$

We assume that $(\mathrm{d}g_i/\mathrm{d}\mathbf{p}_{[j]})(\mathbf{y}_0^*, \mathbf{p}_0) \neq 0$ holds. For the case $(\mathrm{d}g_i/\mathrm{d}\mathbf{p}_{[j]})(\mathbf{y}_0^*, \mathbf{p}_0) = 0$ the inequality constraint g_i cannot become active and the region of confidence is not restricted by $\Delta\mathbf{p}_{[j]}$.

Analogous to the previous case, an active constraint ($g_i = 0, \lambda_i \neq 0$) becomes inactive if λ_i vanishes under the influence of disturbance. Using the Taylor series expansion from (2.77), this means

$$0 = \lambda_i(\mathbf{p}) \approx \hat{\lambda}_i + \frac{\mathrm{d}\lambda_i}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0)\Delta\mathbf{p}, \qquad i \in \mathcal{I}(\mathbf{y}_0^*, \mathbf{p}_0).$$

Thus, the disturbance $\Delta\mathbf{p}_{[j]}$ which causes g_i to leave the set of active indices can be approximated by using the linear prediction

$$\Delta \mathbf{p}_{[j]} \approx -\frac{\hat{\lambda}_i}{\frac{\mathrm{d}\lambda_i}{\mathrm{d}\mathbf{p}_{[j]}}(\mathbf{p}_0)}, \qquad i \in \mathcal{I}(\mathbf{y}_0^*, \mathbf{p}_0), \quad j \in \{1, \ldots, N_p\}. \tag{2.79}$$

We assume again that $(\mathrm{d}\lambda_i / \mathrm{d}\mathbf{p}_{[j]})(\mathbf{p}_0) \neq 0$ holds.

This enables us to predict a change in the set of active indices to occur at the smallest value of $\Delta \mathbf{p}_{[j]}$ obtained from applying (2.78) and (2.79) to all inequality constraints.

2.5 Multi-Objective Optimization

Many real-world optimization problems involve multiple objectives which often conflict with each other. This becomes apparent if an improvement of one objective may lead to a deterioration of another. Thus, a single solution, which can optimize all objectives simultaneously, does not exist. Logically, we need to introduce another concept for optimization of such problems which finds the best trade-off solutions.

Let us define a multi-objective optimization problem.

Definition 2.25 (*Constrained Multi-Objective Optimization*) A constrained *multi-objective optimization problem* can be formulated as

$$\min \quad \mathbf{f}(\mathbf{y}) = \begin{bmatrix} f_1(\mathbf{y}) \\ \vdots \\ f_{N_f}(\mathbf{y}) \end{bmatrix} \tag{2.80}$$

where $\mathbf{y}$ is taken from the constrained decision space Ω which is defined by

$$\Omega := \left\{ \mathbf{y} \in \mathbb{R}^{N_y} \;\middle|\; \mathbf{g}(\mathbf{y}) \leq \mathbf{0} \wedge \mathbf{h}(\mathbf{y}) = \mathbf{0} \right\} \tag{2.81}$$

and the constraints are defined as

$$\mathbf{g}(\mathbf{y}) := \begin{bmatrix} g_1(\mathbf{y}) \\ \vdots \\ g_{N_g}(\mathbf{y}) \end{bmatrix} \quad \text{and} \quad \mathbf{h}(\mathbf{y}) := \begin{bmatrix} h_1(\mathbf{y}) \\ \vdots \\ h_{N_h}(\mathbf{y}) \end{bmatrix}.$$

The objective functions and constraints $\mathbf{f} : \mathbb{R}^{N_y} \to \mathbb{R}^{N_f}$, $\mathbf{g} : \mathbb{R}^{N_y} \to \mathbb{R}^{N_g}$, and $\mathbf{h} : \mathbb{R}^{N_y} \to \mathbb{R}^{N_h}$ are real-valued.

△

Remark 2.11 To enlarge the potential algorithmic class that deals with problem (2.80) and (2.81) we do not enforce the requirement that the functions $\mathbf{f}(\cdot)$, $\mathbf{g}(\cdot)$, and $\mathbf{h}(\cdot)$ have to be all continuously differentiable.

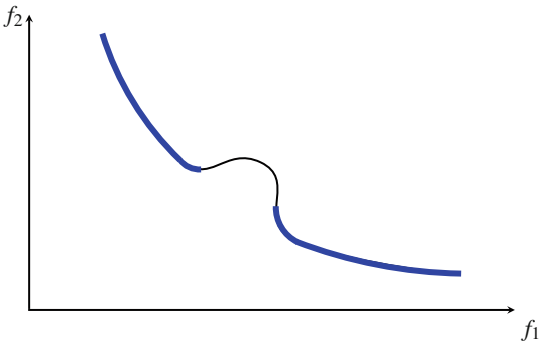


Fig. 2.2 Pareto front: The *bold blue lines* show a Pareto front. The points on the *thin black line* are dominated by points on the *blue lines* and are therefore not contained in the Pareto front. Nonconvexity of Pareto fronts may result from active inequality constraints

The minimize operator in Definition 2.25 means, that we want to minimize all objectives simultaneously. If there exists a single solution $\mathbf{y}^*$, which minimizes each of the objectives $f_1(\mathbf{y}^*), \ldots, f_{N_f}(\mathbf{y}^*)$, we call it a trivial solution. But, in general, a minimum of one objective will not be optimal for other objectives. Therefore, we need to define another optimality concept: the concept of *Pareto optimality*. This concept was first proposed by V. Pareto and is formally defined in terms of nondominated decision vectors as follows (cf. Miettinen [56] and Zhou et al. [74]).

Definition 2.26 (*Dominated Decision Vector*) A decision vector $\mathbf{y}_1 \in \Omega$ is said to *dominate* another decision vector $\mathbf{y}_2 \in \Omega$, $(\mathbf{y}_1 \prec \mathbf{y}_2)$ if $f_i(\mathbf{y}_1) \leq f_i(\mathbf{y}_2)$ for all $i = 1, \ldots, N_f$ and $f_j(\mathbf{y}_1) < f_j(\mathbf{y}_2)$ for at least one index j.

△

Definition 2.27 (*Pareto Optimality*) A decision vector $\mathbf{y}^* \in \Omega$ is called *Pareto optimal* or *nondominated* if there does not exist another decision vector $\mathbf{y} \in \Omega$ which dominates $\mathbf{y}^*$.

△

The set of all Pareto optimal solutions is called a *Pareto optimal set*, which can be nonconvex and disconnected. So, the solution of problem definition 2.25 leads in general to a Pareto optimal set. The image of the Pareto optimal set is called its *Pareto front*, which is illustrated in Fig. 2.2.

2.5.1 Elitist Multi-Objective Evolutionary Algorithm

The elitist *multi-objective evolutionary algorithm* (MOEA) is categorized as a stochastic search technique which maintains and manipulates a population of solutions and implements a survival of the fittest strategy in its search for better solutions

(Davis [18]). These solutions are clustered into nondominated solution fronts, which are used together with the best solutions found so far (elitism concept) to generate an offspring generation by selection, crossover, and mutation. There are several elitist MOEAs (Zhou et al. [74] gives a good overview), but we will concentrate only on the elitist *nondominated search genetic algorithm* (NSGA), more precisely the second generation NSGA-II, by Deb et al. [19].

The principle working scheme of NSGA-II is described in Algorithm 2.2.

Algorithm 2.2 Elitist Multi-objective Evolutionary Algorithm NSGA-II (Deb et al. [19])

1: $k \leftarrow 0$
2: Choose a population size N_{pop}
3: Generate a random initial parent population P_k of size N_{pop}
4: $Q_k \leftarrow$ MAKENEWPOPULATION(P_k)
5: **if** prescribed number of generations N_{gen} is reached **then**
6: **stop**
7: **end if**
8: Generate a combined population $R_k \leftarrow P_k \cup Q_k$
9: Calculate $F \leftarrow$ FASTNONDOMINATEDSORT(R_k)
10: Set $P_{k+1} \leftarrow \emptyset$ and $l \leftarrow 1$
11: **while** $|P_{k+1}| + |F_l| \leq N_{pop}$ **do**
12: $P_{k+1} \leftarrow P_{k+1} \cup F_l$
13: $l \leftarrow l + 1$
14: **end while**
15: $I_{dist} \leftarrow$ CROWDINGDISTANCEASSIGNMENT(F_l)
16: $F_l = $ SORT(F_l, I_{dist})
17: $P_{k+1} \leftarrow P_{k+1} \cup F^{[l]}_{1,..,N_{pop}-|P_{k+1}|}$
18: $k \leftarrow k + 1$
19: Return to step 4.

The first population P_0 is randomly generated. Then, for each iteration $k = 0, \ldots, N_{gen}$ the population P_k is used to generate an offspring population Q_k with the same number of members N_{pop} as the parent population P_k (step 4 of Algorithm 2.2). Therefore, a tournament selection algorithm is applied to select some of the parents, which are then used to generate the children for the offspring population by crossover or mutation. A simple tournament selection algorithm which selects the fitter of two randomly selected members from the population P_k as parents is called a *binary tournament selection*. An algorithm MAKENEWPOPULATION which implements a binary tournament selection for the selection of N_{pool} parents (step 3 of Algorithm 2.3) and which performs a *simulated binary crossover* (SBX) of these parents with a chance of 90% and a polynomial mutation with a chance of 10% as it is implemented by Seshadri [69] can be stated as follows:

Algorithm 2.3 MAKENEWPOPULATION(P)

1: $Q \leftarrow \emptyset$
2: $N_q \leftarrow 0$
3: Perform a binary tournament selection for population P to determine a set of parents P_{pool} with a given size N_{pool}.
4: **for** $i \leftarrow 1$ **to** N_{pop} **do**
5: Get a random number $r \in [0, 1]$
6: **if** $r < 0.9$ **then**
7: Calculate c_1 and c_2 by SBX from randomly selected parents $p_1, p_2 \in P_{pool}$
8: $Q_{[N_q+1]} \leftarrow c_1$
9: $Q_{[N_q+2]} \leftarrow c_2$
10: $N_q \leftarrow N_q + 2$
11: **else**
12: Calculate c_3 by polynomial mutation from a randomly selected parent $p \in P_{pool}$
13: $Q_{[N_q+1]} \leftarrow c_3$
14: $N_q \leftarrow N_q + 1$
15: **end if**
16: **end for**

For each decision variable the SBX (step 7 of Algorithm 2.3) selects a random number $w \in [0, 1]$ and calculates

$$\beta_j(w) = \begin{cases} (2w)^{\frac{1}{\eta_c+1}}, & if\, w \le 0.5 \\ \dfrac{1}{[2(1-w)]^{\frac{1}{\eta_c+1}}}, & otherwise \end{cases}$$

for all $j = 1, \ldots, N_y$. The j-th decision variable of the two children c_1 and c_2 are then calculated from the two parents p_1 and p_2 by the calculation rule

$$c_{j,1} = \frac{1}{2} \cdot [(1 - \beta_j)p_{j,1} + (1 + \beta_j)p_{j,2}]$$
$$c_{j,2} = \frac{1}{2} \cdot [(1 + \beta_j)p_{j,1} + (1 - \beta_j)p_{j,2}].$$

If a child's decision variable violates a constraint, it is set to the extremum.

For each decision variable, the polynomial mutation (step 12 of Algorithm 2.3) selects a random number $w \in [0, 1]$ and calculates

$$\delta_j(w) = \begin{cases} (2w)^{\frac{1}{\eta_m+1}} - 1, & if\, w < 0.5 \\ 1 - [2(1-w)]^{\frac{1}{\eta_m+1}}, & otherwise \end{cases}$$

for all $j = 1, \ldots, N_y$. The j-th decision variable of the child c_3 is then calculated from the parent p by

$$c_{j,3} = p_j + (p_j^{max} - p_j^{min})\delta_j,$$

where p_j^{min} and p_j^{max} are the lower and upper bounds of the j-th decision variable from the parent p, respectively.

The parent population P_k and the offspring population Q_k are then combined into a common population R_k of double size. From R_k the best entries are selected as members for the next parent population P_{k+1}. The selection from the combined population R_k is a task of the elitism concept, which guarantees that the best members of all generations are preserved.

To select the best members from the combined population R_k, the set is first sorted by the fast nondominated sorting algorithm. This algorithm assigns each member to a front F_r, where r is the domination rank of the member, i.e., the number of dominating members. The algorithm can be stated as follows:

Algorithm 2.4 FASTNONDOMINATEDSORT(P) (cf. Deb et al. [19])

```
1: for all p ∈ P do
2:     S_p ← ∅
3:     N_p ← 0
4:     for all q ∈ P do
5:         if p ≺ q then
6:             S_p ← S_p ∪ {q}
7:         else if q ≺ p then
8:             N_p ← N_p + 1
9:         end if
10:    end for
11:    if N_p = 0 then
12:        p_rank ← 1
13:        F_1 ← F_1 ∪ {p}
14:    end if
15: end for
16: r ← 1
17: while F_r ≠ ∅ do
18:    Q ← ∅
19:    for all p ∈ F_r do
20:        for all q ∈ S_p do
21:            N_q ← N_q − 1
22:            if N_q = 0 then
23:                q_rank ← r + 1
24:                Q ← Q ∪ {q}
25:            end if
26:        end for
27:    end for
28:    r ← r + 1
29:    F_r ← Q
30: end while
```

The fronts F_r beginning with $r = 1$ are added to the next population P_{k+1} until an index s is reached for which the addition of the front F_s would exceed the population size. For this front F_s the crowding distances are calculated according to Algorithm 2.5.

Algorithm 2.5 CROWDINGDISTANCEASSIGNMENT(F) (cf. Deb et al. [19])

1: $j \leftarrow |F|$
2: **for** $i \leftarrow 1$, **to** j **do**
3: $\quad I_i^{dist} \leftarrow 0$
4: **end for**
5: **for all** objectives $m \in [1, \ldots, N_f]$ **do**
6: $\quad F \leftarrow \text{sort}(F, m)$
7: $\quad I_1^{dist} \leftarrow \infty$
8: $\quad I_j^{dist} \leftarrow \infty$
9: $\quad$ **for** $i \leftarrow 2$ **to** $j - 1$ **do**
10: $\quad\quad I_i^{dist} \leftarrow I_i^{dist} + \dfrac{F_{i+1}^{[m]} - F_{i-1}^{[m]}}{\mathbf{f}_{[m]}^{max} - \mathbf{f}_{[m]}^{min}}$
11: $\quad$ **end for**
12: **end for**

In Algorithm 2.5, $F_i^{[m]}$ refers to the m-th objective function value of the i-th individual in the set F and $\mathbf{f}_{[m]}^{min}$ and $\mathbf{f}_{[m]}^{max}$ are the minimum and maximum values of the m-th objective function.

The members of F_s with greatest crowding distances are added to the population P_{k+1} until the desired population size is reached. The selection of the greatest crowding distances leads to a greater diversity of the members.

2.5.2 Remarks for MOGAs

Although the underlying mechanisms of *evolutionary algorithms* (EA) are simple, these algorithms have a some advantages compared with smooth nonlinear programming techniques such as:

- high robustness; and
- natural ability to cope with discontinuous and non-differentiable problems.

On the other hand is the none deterministic convergence of these algorithms which makes them slow and only applicable in offline optimizations, whereas SQP can even have a superlinear convergence property. However, SQP can perform badly on problems with non-convex and discontinuous behavior because of inaccurate gradient information used to determine the search direction.

Another popular, but more naive, approach is to transcribe the multi-objective problem into a single-objective problem by focusing on one particular Pareto optimal solution at a time (Johannesson et al. [44]). Such methods have the difficulty that this procedure has to be applied many times, hopefully finding different solutions which approximate the exact Pareto optimal front as good as possible.

2.6 Bibliographical Notes

The history of nonlinear programming is vast and diverse. A good historical overview is provided by Giorgi and Kjeldsen [38]. The first activity, involving finding a local minimizer of a nonlinear function, can be traced back to the years of the Second World War and the years immediately following the war. The term "nonlinear programming" was first mentioned in the fifties in the paper of Kuhn and Tucker [49]. It was later discovered that Karush and John [58] published similar results earlier. The emergence of Kuhn and Tucker [49], however, can be seen as a starting point of nonlinear programming as an autonomous field of research which has strong relationships to other disciplines like mathematical analysis, numerical and nonsmooth analysis, linear algebra, operations research, etc.

Some further subjects of nonlinear programming can be found in the following citations. Arrow et al. [3] weakened the KKT conditions and provided some analysis of the various constraint qualifications. An overview of constraint qualifications for the KKT conditions can be found in Eustaquio et al. [24]. Further examinations and special applications of Kantorovich's theorem can be found in Polyak [63], Ferreira and Svaiter [25], and Potra [64].

A wide variety of algorithms exist for solving NLPs, none of which can be considered preferable for all problems, but there is a consensus in the literature that SQP is one of the most effective methods for solving constrained NLPs.

The first SQP method, which used the exact Hessian of the Lagrangian, was introduced in 1963 by Wilson [70].

There is a wealth of good surveys and overview papers of SQP methods. For instance, Eldersveld [23], Boggs and Tolle [8], Gould and Toint [41], Schittkowski and Yuan [68], and Gill and Wong [35], to name but a few. A comprehensive introduction to nonlinear programming and SQP can be found in the very good textbook of Nocedal and Wright [59]. The SQP approach can be employed in both line-search and trust-region frameworks. The trust-region methods are completely disregarded in this book but can be found in many textbooks of nonlinear programming. Interested readers are recommended to consult Conn et al. [15] for a comprehensive introduction to trust-region methods and Fletcher and Leyffer [30] and Fletcher et al. [32] for details regarding the filter SQP methods omitted from this book.

Alternative approaches for SQP which use an augmented Lagrangian

$$\Psi(\mathbf{y}, \boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\varrho}, \boldsymbol{\nu}) = f(\mathbf{y}) + \boldsymbol{\lambda}^T \mathbf{g}(\mathbf{y}) + \boldsymbol{\mu}^T \mathbf{h}(\mathbf{y}) + \frac{1}{2} \boldsymbol{\varrho}^T \mathbf{g}^2(\mathbf{y}) + \boldsymbol{\nu}^T \mathbf{h}^2(\mathbf{y})$$

as a merit function are described in Boggs and Tolle [8] and Gill et al. [37]. These methods have the advantage that the merit function is differentiable and thus allows for more accurate line-search methods. Moreover, by construction they avoid the Maratos effect. Slightly modified augmented Lagrangian were also proposed by Rockafellar [66], Di Pillo and Grippo [22], Schittkowski [67], Byrd et al. [13], Anitescu [1], Gill and Robinson [34], and Bertsekas [6] as merit functions for constrained nonlinear programming.

More details about line-search methods can be found in Dennis Jr and Schnabel [21], Lemaréchal [50], and Hager and Zhang [42].

More information about the convergence of Quasi-Newton updates can be found in Boggs et al. [9].

In commercial solvers, the quadratic subproblems arising in SQP methods are often solved by AS methods, which were completely disregarded in this book because this solver class for quadratic subproblems is especially successful for small- and medium-scale problems but not for large-scale problems. A very good and stable AS algorithm is described in Goldfarb and Idnani [39].

Sensitivity analysis has been widely used in linear programming. Fiacco [26] made some substantial contributions to the theory of sensitivity analysis for nonlinear programming problems and laid the cornerstone for further research activities. For instance, the transfer of sensitivities to real-time applications, which was extensively investigated by Büskens [10–12].

Since the 1970s several evolutionary methodologies have been proposed, mainly genetic algorithms, evolutionary programming, and evolution strategies. All work on a population of solutions and are therefore blueprints for dealing with conflicting objectives (multi-objectives). An excellent introduction to multi-objective evolutionary algorithms is presented by Zitzler et al. [75] and a great survey paper is given by Zhou et al. [74].

References

1. Anitescu M (2002) On the rate of convergence of sequential quadratic programming with nondifferentiable exact penalty function in the presence of constraint degeneracy. Math Program 92(2):359–386
2. Armijo L (1966) Minimization of functions having Lipschitz continuous first partial derivatives. Pacific J Math 16(1):1–3
3. Arrow KJ, Hurwicz L, Uzawa H (1961) Constraint qualifications in maximization problems. Naval Res Logist Q 8(2):175–191
4. Avriel M (2003) Nonlinear programming: analysis and methods. Courier Corporation
5. Beltracchi TJ, Gabriele GA (1988) An investigation of new methods for estimating parameter sensitivities. Technical Report NASA-CR-183195, NASA
6. Bertsekas DP (2014) Constrained optimization and Lagrange multiplier methods. Academic press
7. Betts JT (2010) Practical methods for optimal control and estimation using nonlinear programming, 2nd edn. Society for industrial and applied mathematics. doi:10.1137/1.9780898718577
8. Boggs PT, Tolle JW (1995) Sequential quadratic programming. Acta Numer 4:1–51
9. Boggs PT, Tolle JW, Wang P (1982) On the local convergence of quasi-newton methods for constrained optimization. SIAM J Control Optim 20(2):161–171
10. Büskens C (1998) Optimierungsmethoden und Sensitivitätsanalyse für optimale Steuerprozesse mit Steuer- und Zustandsbeschränkungen. PhD thesis, Universität Münster
11. Büskens C (2002) Real-time optimization and real-time optimal control of parameter-perturbed problems. Universität Bayreuth, Habilschrift
12. Büskens C, Maurer H (2001) Sensitivity analysis and real-time optimization of parametric nonlinear programming problems. In: Online optimization of large scale systems. Springer, pp 3–16

13. Byrd R, Tapia R, Zhang Y (1992) An SQP augmented lagrangian BFGS algorithm for constrained optimization. SIAM J Optim 2(2):210–241
14. Chamberlain R, Powell M, Lemarechal C, Pedersen H (1982) The watchdog technique for forcing convergence in algorithms for constrained optimization. In: Algorithms for constrained minimization of smooth nonlinear functions. Springer, pp 1–17
15. Conn A, Gould N, Toint P (2000) Trust region methods. Society for industrial and applied mathematics, MPS-SIAM series on optimization
16. Coope I (1985) The Maratos effect in sequential quadratic programming algorithms using the l1 exact penalty function. Research report (University of Waterloo. Faculty of Mathematics), University of Waterloo, Computer Science Department
17. Davidon WC (1966) Variable metric method for minimization. [in Fortran for IBM 704]. Technical report, Argonne National Lab., IL (USA)
18. Davis L (1991) The handbook of genetic algorithms. Van Nostrand Reingold, New York
19. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6:182–197
20. Dennis J, Moré JJ (1974) A characterization of superlinear convergence and its application to Quasi-Newton methods. Math Comput 28(126):549–560
21. Dennis Jr JE, Schnabel RB (1996) Numerical methods for unconstrained optimization and nonlinear equations, vol 16. Siam
22. Di Pillo G, Grippo L (1979) A new class of augmented Lagrangians in nonlinear programming. SIAM J Control Optim 17(5):618–628
23. Eldersveld SK (1992) Large-scale sequential quadratic programming algorithms. Technical Report SOL 92–4, DTIC Document
24. Eustaquio RG, Karas EW, Ribeiro AA (2008) Constraint qualifications for nonlinear programming. Federal University of Parana http://pessoal.utfpr.edu.br/eustaquio/arquivos/kkt.pdf
25. Ferreira OP, Svaiter BF (2012) Kantorovich's theorem on Newton's method. ArXiv e-prints 1209:5704
26. Fiacco AV (1983) Introduction to sensitivity and stability analysis in nonlinear programming. Elsevier
27. Fiacco AV, McCormick GP (1990) Nonlinear programming: sequential unconstrained minimization techniques, vol 4. Siam
28. Fletcher R (1982) Second order corrections for non-differentiable optimization. Springer
29. Fletcher R (2013) Practical methods of optimization. Wiley
30. Fletcher R, Leyffer S (2002) Nonlinear programming without a penalty function. Math Program 91(2):239–269. doi:10.1007/s101070100244
31. Fletcher R, Powell MJ (1963) A rapidly convergent descent method for minimization. Comput J 6(2):163–168
32. Fletcher R, Leyffer S, Toint PL, et al. (2006) A brief history of filter methods. Preprint ANL/MCS-P1372-0906. Argonne National Laboratory, Mathematics and Computer Science Division
33. Gertz EM, Wright SJ (2003) Object-oriented software for quadratic programming. ACM Trans Math Softw 29(1):58–81. doi:10.1145/641876.641880
34. Gill PE, Robinson DP (2012) A primal-dual augmented Lagrangian. Comput Optim Appl 51(1):1–25
35. Gill PE, Wong E (2012) Sequential quadratic programming methods. In: Lee J, Leyffer S (eds) Mixed Integer nonlinear programming, The IMA Volumes in Mathematics and its Applications, vol 154. Springer, New York, pp 147–224. doi:10.1007/978-1-4614-1927-3_6
36. Gill PE, Murray W, Wright MH (1981) Practical optimization. Academic press
37. Gill PE, Murray W, Saunders MA, Wright MH (1986) Some theoretical properties of an augmented Lagrangian merit function. Technical Report SOL 86-6R, Stanford Univ., CA (USA). Systems Optimization Lab
38. Giorgi G, Kjeldsen TH (2014) A historical view of nonlinear programming: traces and emergence. In: Traces and emergence of nonlinear programming. Springer, pp 1–43

39. Goldfarb D, Idnani A (1983) A numerically stable dual method for solving strictly convex quadratic programs. Math program 27(1):1–33
40. Gondzio J (1996) Multiple centrality corrections in a primal-dual method for linear programming. Comput Optim Appl 6(2):137–156. doi:10.1007/BF00249643
41. Gould NI, Toint PL (2000) SQP methods for large-scale nonlinear programming. Springer
42. Hager WW, Zhang H (2005) A new conjugate gradient method with guaranteed descent and an efficient line search. SIAM J Optim 16(1):170–192
43. Han SP (1976) Superlinearly convergent variable metric algorithms for general nonlinear programming problems. Math Program 11(1):263–282
44. Johannesson L, Murgovski N, Ebbesen S, Egardt B, Gelso E, Hellgren J (2013) Including a battery state of health model in the HEV component sizing and optimal control problem. Proceedings of the 7th IFAC symposium on advances in automotive control. Tokyo, Japan, pp 388–393
45. John F (2014) Extremum problems with inequalities as subsidiary conditions. In: traces and emergence of nonlinear programming. Springer, pp 197–215
46. Kantorovich L (1948) Functional analysis and applied mathematics. Uspekhi Mat Nauk 3(6(28)):89–185
47. Kantorovich L, Akilov G (1964) Functional analysis in normed spaces. International series of monographs in pure and applied mathematics, Pergamon Press; [distributed in the Western Hemisphere by Macmillan, New York]
48. Karush W (2014) Minima of functions of several variables with inequalities as side conditions. Springer
49. Kuhn HW, Tucker AW (2014) Nonlinear programming: a historical view. In: traces and emergence of nonlinear programming. Springer, pp 393–414
50. Lemaréchal C (1981) A view of line-searches. In: Optimization and optimal control. Springer, pp 59–78
51. Mangasarian OL (1993) Nonlinear programming, vol 10. siam
52. Maratos N (1978) Exact penalty function algorithms for finite dimensional and control optimization problems. PhD thesis, Imperial College London (University of London)
53. Mayne DQ, Polak E (1982) A surperlinearly convergent algorithm for constrained optimization problems. In: Buckley A, Goffin JL (eds) Algorithms for constrained minimization of smooth nonlinear functions, mathematical programming studies, vol 16, Springer, Berlin Heidelberg, pp 45–61. doi:10.1007/BFb0120947
54. McCormick ST (1983) Optimal approximation of sparse Hessians and its equivalence to a graph coloring problem. Math Program 26(2):153–171
55. Mehrotra S (1992) On the implementation of a primal-dual interior point method. SIAM J Optim 2(4):575–601
56. Miettinen K (1999) Nonlinear multiobjective optimization. International series in operations research and management science, vol 12
57. Moré JJ, Thuente DJ (1994) Line search algorithms with guaranteed sufficient decrease. ACM Trans Math Softw 20(3):286–307. doi:10.1145/192115.192132
58. Moser J (ed) (1985) Fritz John collected papers, vol 1. Birkhäuser-Verlag, Basel-Boston
59. Nocedal J, Wright S (2006) Numerical optimization. Springer series in operations research and financial engineering, 2nd edn. Springer, Science+Business
60. de Oliveira ORB (2012) The implicit and the inverse function theorems: easy proofs. arXiv preprint arXiv:12122066
61. Ortega J (1968) The Newton-Kantorovich theorem. The Am Math Monthly 75(6):658–660
62. Ortega J, Rheinboldt W (2000) Iterative solution of nonlinear equations in several variables. Society for Industrial and Applied Mathematics, Classics in Applied Mathematics
63. Polyak BT (2006) Newton-Kantorovich method and its global convergence. J Math Sci 133(4):1513–1523
64. Potra FA (2005) The Kantorovich theorem and interior point methods. Math program 102(1):47–70

65. Powell MJ (1978) The convergence of variable metric methods for non-linearly constrained optimization calculations. Nonlinear programming 3
66. Rockafellar RT (1974) Augmented Lagrange multiplier functions and duality in nonconvex programming. SIAM J Control 12(2):268–285
67. Schittkowski K (1982) On the convergence of a sequential quadratic programming method with an augmented Lagrangian line search functions. Technical Report SOL 82–4, DTIC Document
68. Schittkowski K, Yuan YX (2011) Sequential quadratic programming methods. Wiley encyclopedia of operations research and management science
69. Seshadri A (2007) A fast elitist multi-objective genetic algorithm NSGA-II
70. Wilson RB (1963) A simplicial algorithm for concave programming. PhD thesis, Graduate School of Business Administration, George F. Baker Foundation, Harvard University
71. Wolfe P (1969) Convergence conditions for ascent methods. SIAM Rev 11(2):226–235
72. Wolfe P (1971) Convergence conditions for ascent methods. II: some corrections. SIAM Rev 13(2):185–188
73. Wright S (1997) Primal-dual interior-point methods. Society for industrial and applied mathematics
74. Zhou A, Qu BY, Li H, Zhao SZ, Suganthan PN, Zhang Q (2011) Multiobjective evolutionary algorithms: a survey of the state of the art. Swarm Evol Comput 1(1):32–49
75. Zitzler E, Laumanns M, Bleuler S (2004) A tutorial on evolutionary multiobjective optimization. In: Metaheuristics for multiobjective optimisation. Springer, pp 3–37

Chapter 3
Hybrid Systems and Hybrid Optimal Control

3.1 Introduction

A hybrid system is a generalization of a continuous dynamical system and a discrete-event system that incorporates the behavior of both system types. An electrical circuit, where current and voltage can change continuously over time but can also change discontinuously when a switch is opened or closed (Goebel et al. [31]), may serve as an example. Both system classes, continuous and discrete-event systems, have their specific ways of representation, such as discrete automata for discrete-event systems and ordinary differential equations for purely continuous systems. Many physical systems exhibit both kinds of dynamical behavior and the respective model needs to incorporate both types of dynamics as shown by Riedinger and Kratz [48]. Hybrid systems can therefore be seen as a generalization of conventional systems. It is due to the frequent appearance in technical systems and daily life that the research interest in hybrid systems is steadily growing.

There exist many different notations for describing hybrid systems, mainly driven by the viewpoint of the corresponding discipline, e.g., system and control theory (Van Der Schaft et al. [58], Goebel et al. [31], and many more) and computer science and software engineering (Stauner [55], Zhu and Antsaklis [67]). There are two common approaches to build up hybrid systems. The first approach is motivated by extending the classical theory of time-driven continuous systems. The continuous dynamics given by ordinary differential or difference equations is generalized to a hybrid system by incorporating discrete phenomena. The second method extends completely discrete-event-driven modeling schemes like finite-state machines, finite automata, or Petri nets by adding timing and continuous dynamics. The second method leads to more general hybrid system descriptions and is mainly driven by computer science and queuing theory.

We adopt the first approach by describing the subsystems of interest by ordinary differential equations and formulate optimal control problems to numerically determine optimal feed-forward trajectories or feedback control laws.

T.J. Böhme and B. Frank, *Hybrid Systems, Optimal Control and Hybrid Vehicles*,
Advances in Industrial Control, DOI 10.1007/978-3-319-51317-1_3

3.2 System Definition

In this section, we describe continuous systems, hybrid systems, and switched systems which are of major interest in this book. We start by classifying these nonlinear systems and provide conditions for existence and uniqueness of their continuous-valued control and state trajectories.

3.2.1 Continuous Systems

Definition 3.1 (*Continuous System*) A continuous system consists of a 4-tuple

$$\mathbb{C} := (\mathsf{X}, \mathsf{U}, \mathcal{F}, \mathsf{D})$$

where X is a continuous-valued state space, U is a continuous-valued control space, $\mathcal{F}$ is a collection of vector fields to describe the continuous dynamics, and D is a collection of constraint functions. △

The dynamics of a continuous nonlinear system $\mathbb{C}$ is described by a set of nonlinear *ordinary differential equations* (ODE)

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \quad \forall t \in [t_0, t_f] \tag{3.1}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{3.2}$$

where $t \in [t_0, t_f] \subset \mathbb{R}$ is the time; t_0 and t_f are the initial and final times, respectively; and $\mathbf{x}_0$ is the initial state. The continuous-valued state space is the complete $\mathbb{R}^{N_x}$ and is defined as $\mathsf{X} := \mathbb{R}^{N_x}$. The same applies for the continuous-valued control space and is defined as $\mathsf{U} := \mathbb{R}^{N_u}$. $\mathbf{x} : [t_0, t_f] \to \mathsf{X}$ is the N_x-dimensional continuous-valued state vector (for short *continuous-valued states*), $\mathbf{u} : [t_0, t_f] \to \mathsf{U}$ is the N_u-dimensional continuous-valued control vector (for short *continuous-valued controls*), and $\mathbf{f} : \mathsf{X} \times \mathsf{U} \to \mathsf{X}$, $\mathcal{F} = \mathbf{f}$ is the vector field which describes the dynamics of the continuous system. The system (3.1)–(3.2) is classified as *autonomous* because the system does not explicitly depend on the time t, whereas, for *non-autonomous* systems the right-hand side function $\mathbf{f}(\cdot, t)$ of the ODE explicitly depends on the time.

Next, we will insist that for every choice of the initial state $\mathbf{x}_0$ and every admissible control $\mathbf{u}(\cdot)$, the system (3.1)–(3.2) has a unique solution $\mathbf{x}(\cdot)$ on the time interval $[t_0, t_f]$. Such systems are called *well-posed*. In order to guarantee that, we have to impose some regularity conditions on the right-hand side function $\mathbf{f}(\cdot)$ and on the admissible controls $\mathbf{u}(\cdot)$.

First, we specify the regularity of $\mathbf{f}(\mathbf{x}(t), \cdot)$ with respect to $\mathbf{u}(\cdot)$ by assuming the weakest property, namely *measurability*. Readers who wish to gain a deeper understanding about the measurability concept are advised to consult the works of Adams and Fournier [2] and Clarke [22]. Since we have a space of functions $\mathbf{u}(\cdot)$ (we come back to this point in the next chapter in more detail) we denote the space

of measurable functions in general with $L^p([t_0, t_f], \mathsf{U})$. The infinite-dimensional function space $L^p([t_0, t_f], \mathsf{U})$ of real-valued measurable functions $\mathbf{u}(t)$ is endowed with the norm

$$||\mathbf{u}(\cdot)||_p = \left(\int_{t_0}^{t_f} |\mathbf{u}(t)|^p \, \mathrm{d}t \right)^{\frac{1}{p}} < \infty.$$

$L^p([t_0, t_f], \mathsf{U})$ spaces are Banach spaces, i.e., complete normed vector spaces, which are defined by the Lebesgue integral. We use in the sequel, the space of all essentially bounded measurable maps that is denoted by $L^\infty([t_0, t_f], \mathsf{U})$. This space $L^\infty([t_0, t_f], \mathsf{U})$ is equipped with the norm

$$||\mathbf{u}(\cdot)||_\infty := \operatorname{ess} \sup_{t \in [t_0, t_f]} |\mathbf{u}(t)|.$$

The function $\mathbf{u}(\cdot)$ is essentially bounded, if $||\mathbf{u}(\cdot)||_\infty < \infty$, i.e., $|\mathbf{u}(t)| < \infty$ for a.e. $t \in [t_0, t_f]$. The abbreviation "a.e." means *almost everywhere*.

We use this rather complex form of regularity to be consistent with the necessary conditions and existence and uniqueness results established in the majority of the literature. However, the handling with measurable functions can be greatly facilitated by substituting intellectually *piecewise continuous functions* instead of "measurable functions." Piecewise continuous functions have only a finite number of discontinuities on every bounded interval and possess finite limits from the right and from the left at each of these discontinuities. This substitution does even not violate the necessary conditions since solutions with Zeno-behavior are excluded from our examples. For clarification, we speak from Zeno-behavior, if the solution have infinitely many switchings, which cannot be represented by piecewise continuous functions.

Second, we specify the regularity of $\mathbf{f}(\cdot, \mathbf{u}(t))$ with respect to $\mathbf{x}(\cdot)$ for all $t \in [t_0, t_f]$. In so doing, we assume $\mathbf{x}(\cdot)$ to be absolutely continuous. That means, $\mathbf{x}(\cdot)$ is continuous everywhere, continuously differentiable almost everywhere, and satisfies the corresponding integral equation

$$\mathbf{x}(t) = \mathbf{x}_0 + \int_{t_0}^{t} \mathbf{f}(\mathbf{x}(s), \mathbf{u}(s)) \, \mathrm{d}s, \quad t \in [t_0, t_f]. \tag{3.3}$$

A function with these properties is called *absolutely continuous* and the derivative of such a function is always measurable. That means, the terminology "almost everywhere" stems from the derivative of the absolutely continuous function that can be discontinuous on a countable set of points that has measure zero.

We denote by $\mathcal{AC}^\infty([t_0, t_f], \mathsf{X})$ the class of absolutely continuous functions $\mathbf{x}(\cdot)$ which admits a representation of the form (3.3) on the interval $[t_0, t_f]$ with $\mathbf{f}(\cdot) \in L^\infty([t_0, t_f], \mathsf{X})$. The norm on $\mathcal{AC}^\infty([t_0, t_f], \mathsf{X})$ (Clarke [22]) is defined by

$$||\mathbf{x}(\cdot)||_{\mathcal{AC}^\infty} := |\mathbf{x}_0| + ||\dot{\mathbf{x}}(\cdot)||_\infty.$$

If $\mathbf{f}(\cdot)$ satisfies both regularity assumptions, then on the interval $[t_0, t_f]$ there exists a solution $\mathbf{x}(\cdot)$ of the initial value problem (3.1)–(3.2) according to Carathéodory's existence theorem (see Coddington and Levinson [23]). However, this solution is not unique. For uniqueness we have to tighten the space of possible functions by inducing a condition that limits the slope of $\mathbf{x}(\cdot)$ w.r.t. time.

In so doing, we insist that $\mathbf{f}(\cdot, \mathbf{u}(t))$ is *Lipschitz continuous* with respect to $\mathbf{x}(\cdot)$ for all t. For Lipschitz continuity, there must exist a *Lipschitz constant* $0 \leq L_f < \infty$ such that

$$||\mathbf{f}(\mathbf{x}_1, \mathbf{u}(t)) - \mathbf{f}(\mathbf{x}_2, \mathbf{u}(t))|| \leq L_f \cdot ||\mathbf{x}_1 - \mathbf{x}_2||, \quad \forall \mathbf{x}_1, \mathbf{x}_2 \in \mathsf{X}, \ \forall \mathbf{u}(t) \in \mathsf{U}, \ \forall t \in [t_0, t_f] \tag{3.4}$$

holds.

If $\mathbf{f}(\cdot)$ satisfies additionally the Lipschitz regularity, then the solution $\mathbf{x}(\cdot)$ of the initial value problem (3.1)–(3.2) is unique according to the Lemma of Bellman–Gronwall (cf. Sontag [54] and Ziebur [68]). In fact, we can be more generous by assuming $\mathbf{f}(\cdot, \mathbf{u}(\cdot))$ to be continuously differentiable in $\mathbf{x}(\cdot)$ for each fixed $\mathbf{u}(\cdot) \in L^\infty([t_0, t_f]), \mathsf{U})$. This is a stronger hypothesis and excludes some systems. It is noteworthy that in either case differentiability of $\mathbf{f}(\mathbf{x}(\cdot), \cdot)$ with respect to $\mathbf{u}(\cdot)$ is not assumed. Furthermore it is to mention that every function, which satisfies the Lipschitz condition (3.4), is also absolutely continuous.

In most cases, $\mathbf{u}(\cdot)$ and $\mathbf{x}(\cdot)$ will be subject to a set of constraints D. The set $\mathsf{D} = \{\mathbf{c}_{x,u}(\cdot), \mathbf{c}_x(\cdot), \mathbf{c}_u(\cdot)\}$ is a collection of different constraint types that applies to the considered system which may include mixed control-state constraints $\mathbf{c}_{x,u}(\cdot)$, pure state constraints $\mathbf{c}_x(\cdot)$, and pure control constraints $\mathbf{c}_u(\cdot)$. Hence, it is assumed that the functions $\mathbf{u}(\cdot)$ and $\mathbf{x}(\cdot)$ belong to *admissible function spaces* $\mathcal{U}$ and $\mathcal{X}$.

An interesting special case of continuous nonlinear systems is systems *affine in controls* (or *control-affine*). For such systems, $\mathbf{f}(\mathbf{x}(\cdot), \cdot)$ is affine in the continuous-valued controls $\mathbf{u}(\cdot) \in L^\infty([t_0, t_f], \mathsf{U})$, such that its equations take the form

$$\dot{\mathbf{x}}(t) = \mathbf{f}_0(\mathbf{x}(t)) + \sum_{i=1}^{N_u} \mathbf{f}_i(\mathbf{x}(t)) \cdot u_i(t), \quad \text{for a.e. } t \in [t_0, t_f] \tag{3.5}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{3.6}$$

where $\mathcal{F} = \left\{\mathbf{f}_i : \mathsf{X} \to \mathsf{X} \;\middle|\; i = 0, \ldots, N_u\right\}$ is an indexed collection of $N_u + 1$ vector fields.

Such systems are very common in technical applications, because many components encountered in the vehicles show input linear behavior due to implemented low-level control schemes. Affine system will play an important role for the representations of switched systems.

3.2.2 Hybrid Systems

In contrast to continuous systems, hybrid systems are those that evolve on the continuous state space and involve continuous controls as well as discrete states. The description of hybrid systems is far more complicated compared with the continuous counterpart. We start by describing the general hybrid system and then eliminate some conditions that are not needed for the problems discovered in this book.

Definition 3.2 (*Hybrid System* (*Passenberg* [44])) A hybrid system consists of a 9-tuple

$$\mathbb{H} := (\hat{\mathcal{Q}}, \mathsf{X}, \mathsf{U}, \mathcal{F}, \Pi, \mathsf{A}, \mathcal{B}, \mathsf{C}, \mathsf{D}),$$

where $\hat{\mathcal{Q}}$ denotes a set of discrete states, X is the continuous-valued state space, U is a collection of continuous-valued control spaces, $\mathcal{F}$ is a collection of vector fields to describe the continuous dynamics, Π is a discrete transition function, A is a collection of reset maps, $\mathcal{B}$ is a set of admissible discrete control edges, C is a collection of switching surfaces, and D is a collection of constraint functions. △

The hybrid system can be imagined as a piecewise decomposition into $N_q \in \mathbb{N}_{>0}$ stages by adding a discrete state $q : [t_0, t_f] \to \hat{\mathcal{Q}} = \{1, 2, \ldots, N_q\}$ to the system description (3.1)–(3.2), which defines the activity of a subsystem or mode (Riedinger et al. [48, 49]). In the literature, $q(\cdot)$ as element from the set of discrete states $\hat{\mathcal{Q}}$ is also often referred to as location.

In general, the continuous-valued control space and continuous-valued state space of hybrid systems are partitioned into several control and state spaces of equal or different dimensions depending on the number of controls and states in use for the active subsystem. However, throughout this book, we assume that the continuous-valued control space and the continuous-valued state space keep their dimension constant, i.e., $\mathsf{U} := \mathbb{R}^{N_u}$ and $\mathsf{X} := \mathbb{R}^{N_x}$.

For each discrete state $q \in \hat{\mathcal{Q}}$, one vector field $\mathbf{f}_q : \mathsf{X} \times \mathsf{U} \to \mathsf{X}$ is defined. All together, we obtain an indexed collection of vector fields $\mathcal{F} = \left\{ \mathbf{f}_q : \mathsf{X} \times \mathsf{U} \to \mathsf{X} \;\middle|\; q \in \hat{\mathcal{Q}} \right\}$ of N_q-subsystems. The overall system shares a common state $\mathbf{x}(\cdot)$ but the vector fields, which govern the state's evolution, are switched depending on the current value of the discrete state $q(\cdot)$. The overall system's differential equation can be written as

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)), \quad \text{for a.e. } t \in [t_0, t_f] \tag{3.7}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0, \; q(t_0) = q_0 \tag{3.8}$$

where the controls and states are again restricted to the admissible function spaces, $\mathbf{u}(\cdot) \in \mathcal{U}(q(\cdot))$ and $\mathbf{x}(\cdot) \in \mathcal{X}$, respectively, and q_0 is the initial discrete state.

The times, where the discrete state $q(\cdot)$, and therefore the right-hand side term in (3.7) changes, are denominated as t_j^- and t_j^+, where the former refers to the time

right before a change and the latter to the time right after a change. The subscript j enumerates the number of switchings and the total number of discontinuous changes in $q(\cdot)$ is denoted as N_{swt}. A system, starting at the initial hybrid state $(\mathbf{x}(t_0), q(t_0))$, will then evolve according to the differential equation $\dot{\mathbf{x}}(t) = \mathbf{f}_{q(t_0)}(\mathbf{x}(t), \mathbf{u}(t))$, until a switching occurs at a switching time t_j. The switching is induced by a transition function $\Pi(\cdot)$ to the piecewise constant discrete state function $q(\cdot)$. This transition function can capture four different kinds of characteristic behavior in hybrid systems: *autonomous switching*, *autonomous jumps*, *controlled switching*, and *controlled jumps* (Branicky et al. [17], Passenberg [44]).

At an autonomous or controlled switching, the successive discrete state $q(\cdot)$ is chosen by the discrete control $\varpi(\cdot)$, which is explicitly known in the case of controlled switching and implicitly defined in the case of autonomous switching, as will be seen later. The set of admissible transitions $\mathcal{B}_q$ for the discrete control is defined as a collection $\mathcal{B}_q = \{(q, q^\circ) \mid (q, q^\circ) \in \hat{\mathcal{Q}} \times \hat{\mathcal{Q}}\}$ which consists of all tuples (q, q°), for which the switching from only the current subsystem q to the next subsystem q° is allowed. The edge set $\mathcal{B} = \{\mathcal{B}_q \mid \forall q \in \hat{\mathcal{Q}}\}$ describes all admissible tuples for the discrete control. To enumerate the entries of these tuples a function $g : \hat{\mathcal{Q}} \to \mathbb{N}$ is defined. Then, the discrete control $\varpi(t_j)$ is determined from the set of admissible control edges $\mathcal{B}_q$ by the relation $\varpi(t_j) = g(q^\circ) - g(q)$ for each $t_j \in [t_0, t_f]$, $j = 1, \ldots, N_{swt}$ and is otherwise zero. In a more transparent definition, let us define the set $\hat{\mathsf{B}}_q = \{g(q^\circ) - g(q) \mid (q, q^\circ) \in \mathcal{B}_q\}$ of signed values and

$$\begin{cases} \varpi(t) \in \hat{\mathsf{B}}_q, & t = t_j,\ t_j \in [t_0, t_f],\ j = 1, \ldots, N_{swt} \\ \varpi(t) = 0, & t \neq t_j. \end{cases}$$

The reader should note that the admissible discrete control edge set $\mathcal{B}$ contains no self-invocations.

On autonomous switching, the discrete state $q(\cdot)$ changes at the time t_j in a predefined way, when the states encounter a switching manifold $\mathsf{C}_{(q(t_j^-),q(t_j^+))}(\cdot)$ from the set of switching manifolds $\mathsf{C} = \{\mathsf{C}_{(q(t_j^-),q(t_j^+))}(\cdot) \mid q \in \hat{\mathcal{Q}}\}$. A switching manifold $\mathsf{C}_{(q(t_j^-),q(t_j^+))}(\cdot)$ of $\mathbb{R}^{N_x}$ has codimension 1 and can be thought as a nonlinear surface. Switching manifolds can be locally expressed by $\mathsf{C}_{(q(t_j^-),q(t_j^+))}(t) := \{\mathbf{x}(t) \mid c_{(q(t_j^-),q(t_j^+))}(\mathbf{x}(t), t) = 0,\ \mathbf{x}(t) \in \mathsf{X},\ q(t) \in \hat{\mathcal{Q}}\}$, where the equations $c_{(q(t_j^-),q(t_j^+))}(\mathbf{x}(t), t) = 0$ have to be specified by the designer for all transitions as shown in Table 3.1.

The switching manifold is at least once continuously differentiable w.r.t. $\mathbf{x}(\cdot)$ and continuous w.r.t. time t. If the states hit the switching manifold, i.e., $\mathbf{x}(t_j) \in \mathsf{C}_{(q(t_j^-),q(t_j^+))}(t_j)$, then the discrete control $\varpi(t_j) \in \hat{\mathsf{B}}_q$ is triggered such that the *discrete transition function* $\Pi_{ifs} : \mathsf{X} \times \hat{\mathcal{Q}} \times \hat{\mathsf{B}}_q \to \hat{\mathcal{Q}}$ (Shaikh [53])

Table 3.1 Switching manifolds for given continuous-valued states $\mathbf{x}(\cdot)$ and time t arranged in a table

$q(\cdot)$	1	2	$\cdots$	N_q
1	0	$c_{(1,2)}(\mathbf{x}(t),t)=0$	$\cdots$	$c_{(1,N_q)}(\mathbf{x}(t),t)=0$
2	$c_{(2,1)}(\mathbf{x}(t),t)=0$	0	$\cdots$	$c_{(2,N_q)}(\mathbf{x}(t),t)=0$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
N_q	$c_{(N_q,1)}(\mathbf{x}(t),t)=0$	$c_{(N_q,2)}(\mathbf{x}(t),t)=0$	$\cdots$	0

$$q(t_j^+)=\Pi_{ifs}\left(\mathbf{x}(t_j^-),q(t_j^-),\varpi(t_j)\right),\quad \mathbf{x}(t_j^-)\in\mathsf{X},\quad q(t_j^-),\,q(t_j^+)\in\hat{\mathcal{Q}},\quad \varpi(t_j)\in\hat{\mathsf{B}}_q \tag{3.9}$$

is executed. The transition (3.9) is called an *internally forced switching* (IFS) (Xu and Antsaklis [65]).

For example, when the subsystem $q(t)=2$ is active and the state trajectory intersects the switching manifold $\mathbf{x}(t_j)\in\mathsf{C}_{(2,3)}(t_j)$ at time t_j, $\varpi(t_j)=q_2q_3=1$, $\varpi(t)=0,\ t\neq t_j$ is autonomously triggered and the system is forced to switch from subsystem $q(t_j^-)=2$ to subsystem $q(t_j^+)=3$.

For instance, if a transition from subsystem $q(t)=2$ to subsystem $q(t)=3$ is consciously chosen then a controlled switching (also known as *externally forced switching* (EFS)) must be performed. Here, $q(\cdot)$ changes according to the discrete transition function $\Pi_{efs}:\hat{\mathcal{Q}}\times\hat{\mathsf{B}}_q\to\hat{\mathcal{Q}}$

$$q(t_j^+)=\Pi_{efs}\left(q(t_j^-),\varpi(t_j)\right),\quad q(t_j^-),\,q(t_j^+)\in\hat{\mathcal{Q}},\quad \varpi(t_j)\in\hat{\mathsf{B}}_q \tag{3.10}$$

in response to a commanded change in the discrete control $\varpi(\cdot)$.

From a point of control, autonomous switching is completely described by the discrete transition function $\Pi_{ifs}(\cdot)$ whereas controlled switchings need external commands $\varpi(\cdot)$ to initiate switchings between the subsystems which are in general unknown. The discrete transition function $\Pi_{efs}(\cdot)$ defines which subsystem changes are allowed. One can interpret the transition functions as memory functions which makes it easier to understand $q(\cdot)$ as a discrete state.

Often, the continuous-valued states $\mathbf{x}(\cdot)$ are continuous. In hybrid systems the state(s) $\mathbf{x}(\cdot)$ may exhibit discontinuities, when a switching occurs.

If we assume that the transient between the subsystems is fast, the discontinuities can be modeled with the help of reset (jump) functions $\boldsymbol{\delta}_{(q(t_j^-),q(t_j^+))}:\mathsf{X}\to\mathsf{X},\ q(t_j^-)\neq q(t_j^+)$:

$$\mathbf{x}(t_j^+)=\mathbf{x}(t_j^-)+\boldsymbol{\delta}_{(q(t_j^-),q(t_j^+))}(\mathbf{x}(t_j^-)). \tag{3.11}$$

Table 3.2 Height of the jump $\Delta\mathbf{x}$ is given by the function values $\boldsymbol{\delta}_{(q(t_j^-),q(t_j^+))}(\cdot)$ for given continuous-valued states $\mathbf{x}(\cdot)$ at time instant t_j arranged in a table

$q(\cdot)$	1	2	$\cdots$	N_q
1	$\mathbf{0}$	$\boldsymbol{\delta}_{(1,2)}(\mathbf{x}(t_j^-))$	$\cdots$	$\boldsymbol{\delta}_{(1,N_q)}(\mathbf{x}(t_j^-))$
2	$\boldsymbol{\delta}_{(2,1)}(\mathbf{x}(t_j^-))$	$\mathbf{0}$	$\cdots$	$\boldsymbol{\delta}_{(2,N_q)}(\mathbf{x}(t_j^-))$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
N_q	$\boldsymbol{\delta}_{(N_q,1)}(\mathbf{x}(t_j^-))$	$\boldsymbol{\delta}_{(N_q,2)}(\mathbf{x}(t_j^-))$	$\cdots$	$\mathbf{0}$

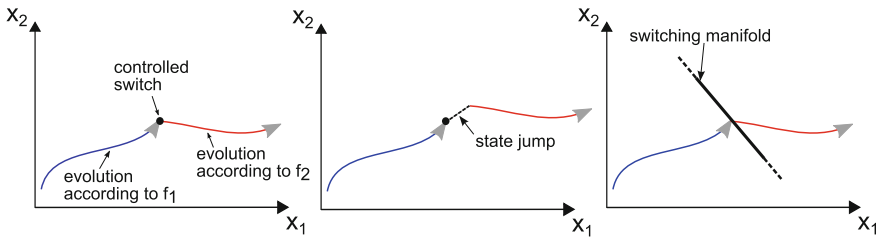


Fig. 3.1 Hybrid phenomena, from *left* to *right*: controlled switching, controlled state jump, and autonomous switching (Passenberg [44])

The collection of reset functions is then defined by $\mathsf{A} = \{\boldsymbol{\delta}_{(q^-,q^+)}(\cdot) \mid \forall q(t_j^-), q(t_j^+) \in \hat{\mathcal{Q}}$ with $q(t_j^-) \neq q(t_j^+)\}$.

The reset functions $\boldsymbol{\delta}_{(q(t_j^-),q(t_j^+))}(\cdot)$ can be arranged in a table as shown in Table 3.2.

State jumps are often the result of simplifications made in the modeling of complex process parts, e.g., to account for energy costs incurred by operating a discrete switch. Sketches of the discrete phenomena, controlled switching, autonomous switching, and state jump, are depicted in Fig. 3.1.

In Fig. 3.1, the thick straight line denotes the switching manifold, the colored curves with arrows denote the continuous portions of the trajectory, and the dashed line symbolizes the state jump. The instantaneous jumps of the continuous-valued state are sometimes referred to as *impulse effects*.

3.2.3 *Controlled Hybrid Systems and Switched Systems*

A class of system that is of particular interest in this book is hybrid systems that exhibit only controlled switching. This class can be obtained from Definition 3.2 by simply setting the switching manifolds for autonomous switching to an empty set, i.e., $\mathsf{C} = \emptyset$. We obtain then the definition for a hybrid system with controlled switching as follows:

Definition 3.3 (*Hybrid System with Controlled Switching*) A hybrid system with controlled switching and without state jumps consists of a 6-tuple

$$\mathbb{H}_1 := (\hat{\mathcal{Q}}, \mathsf{X}, \mathsf{U}, \mathcal{F}, \mathcal{B}, \mathsf{D}).$$

△

Definition 3.4 (*Hybrid System with Controlled Switching and State Jumps*) A hybrid system with controlled switching and state jumps consists of a 7-tuple

$$\mathbb{H}_2 := (\hat{\mathcal{Q}}, \mathsf{X}, \mathsf{U}, \mathcal{F}, \mathsf{A}, \mathcal{B}, \mathsf{D}).$$

△

In System Definitions 3.3 and 3.4 the admissible discrete control edge set $\mathcal{B}$ can be incomplete (cf. Definition 14.5). A further simplification according to Xu and Antsaklis [63, 64] implies that the admissible discrete control edge set is defined as $\mathcal{B} := \left(\hat{\mathcal{Q}} \times \hat{\mathcal{Q}}\right) \setminus \left\{(q, q) \mid q \in \hat{\mathcal{Q}}\right\}$, which is a complete graph. This implies that the subsets $\mathcal{B}_q \subset \mathcal{B}$ are identical for all admissible values of the discrete state $q \in \hat{\mathcal{Q}}$. This simplification leads to a subclass of hybrid systems, where the discrete control $\varpi(\cdot)$ can be neglected and the discrete state $q(\cdot)$ can be chosen freely at any time from the admissible set $\hat{\mathcal{Q}}$. This particular class, the so-called *switched systems*, covers a great range of technical problems. From this definition we can deduce two important classes of switched systems which are frequently considered in this book:

Definition 3.5 (*Switched System without State Jumps*) A switched system without state jumps consists of a 5-tuple

$$\mathbb{S}_1 := (\hat{\mathcal{Q}}, \mathsf{X}, \mathsf{U}, \mathcal{F}, \mathsf{D}).$$

△

Definition 3.6 (*Switched System with State Jumps*) A switched system with state jumps consists of a 6-tuple

$$\mathbb{S}_2 := (\hat{\mathcal{Q}}, \mathsf{X}, \mathsf{U}, \mathcal{F}, \mathsf{A}, \mathsf{D}).$$

△

The major difference between a hybrid system with controlled switching and a switched system can be best illustrated using a finite-state machine interpretation. Let us take $G = (\hat{\mathcal{Q}}, \mathcal{B})$ as a directed graph indicating the discrete mode structure of the system. Observing, the left graph in Fig. 3.2 is complete which allows to choose every discrete state at any time whereas the right graph is incomplete which prevents direct changes at any time from $q = 1$ to $q = 3$. As a result, the left graph constitutes a switched system and the right graph constitutes a hybrid system.

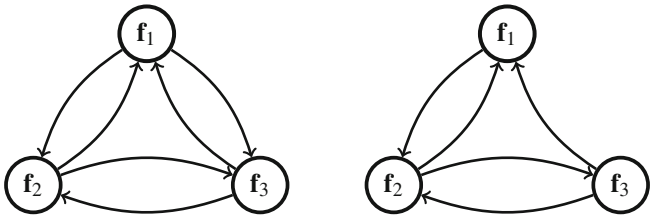


Fig. 3.2 *Subfigure left* Switched system with $\hat{\mathcal{Q}} = \{1, 2, 3\}$ and $\mathcal{B} = \{q_1q_2, q_1q_3, q_2q_1, q_2q_3, q_3q_1, q_3q_2\}$; *subfigure right* none switched system with $\hat{\mathcal{Q}} = \{1, 2, 3\}$ and $\mathcal{B} = \{q_1q_2, q_2q_1, q_2q_3, q_3q_1, q_3q_2\}$

3.2.4 Existence and Uniqueness of Admissible States and Controls

For the establishment of existence and uniqueness statements for admissible continuous-valued states and continuous-valued controls, we assume only hybrid systems without reset functions, i.e., $\mathsf{A} = \emptyset$, which let us proceed in a similar way to the continuous systems. But beforehand, we need to introduce the following definitions.

Definition 3.7 (*Hybrid Execution (Shaikh* [53])) An execution of a hybrid (switched) system is defined by the tuple $e_{\mathbb{H}} := (\mathcal{T}, \mathcal{D}, \mathcal{S}, \mathcal{Z})$. Herein

- a hybrid time trajectory is a strictly increasing sequence of times, $\mathcal{T} = (t_0, t_1, t_2, \ldots, t_{N_{exe}})$ with $0 \leq N_{exe} < \infty$ and $t_{N_{exe}} = t_f$, of finitely many arcs $([t_0, t_1), [t_1, t_2), \ldots, [t_{N_{exe}-1}, t_{N_{exe}}))$ of different active subsystems within the interval $[t_0, t_f]$;
- an associated sequence to $\mathcal{T}$ of discrete state functions is given by $\mathcal{D} = (q_0, q_1, q_2 \ldots, q_{N_{exe}-1})$ with the functions $q_j : [t_j, t_{j+1}) \to \hat{\mathcal{Q}}$;
- an associated sequence to $\mathcal{T}$ of continuous-valued state functions is given by $\mathcal{S} = (\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_{exe}-1})$ with functions $\mathbf{x}_j : [t_j, t_{j+1}) \to \mathsf{X}$ which evolve according to

$$\dot{\mathbf{x}}_j(t) = \mathbf{f}_{q_j(t)}(\mathbf{x}_j(t), \mathbf{u}_j(t))$$

  over the interval $[t_j, t_{j+1})$ and fulfill the state jump condition $\mathbf{x}(t_j^+) = \mathbf{x}(t_j^-),\ j = 1, \ldots, N_{exe}$; and
- an associated sequence to $\mathcal{T}$ of continuous-valued control functions is given by $\mathcal{Z} = (\mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2 \ldots, \mathbf{u}_{N_{exe}-1})$ with the control functions $\mathbf{u}_j : [t_j, t_{j+1}) \to \mathsf{U}$.

△

Definition 3.8 (*Switching Sequence and Switching Time*) A switching sequence is a time-based sequence that indicates when the system switches to a different mode and is defined as the tuple

$$\Theta := ((t_1, q_1), (t_2, q_2), \ldots, (t_{N_{exe}-1}, q_{N_{exe}-1})), \tag{3.12}$$

where the switching values t_j and q_j are taken pairwise from the sequences $\mathcal{T}$ and $\mathcal{D}$, respectively (Definition 3.7). Consequently, a switching time sequence is defined as the tuple

$$\Theta_t := (t_1, t_2, \ldots, t_{N_{exe}-1}). \tag{3.13}$$

The number of switchings is then given as $N_{swt} = N_{exe} - 1$. △

Example of a Hybrid Execution (Schori [50]):

Considering a system of the form $\dot{\mathbf{x}}(t) = \mathbf{f}_{q(t)}(\mathbf{x}(t))$ with the hybrid time trajectory $\mathcal{T} = (t_0, t_1, t_2) = (0\text{s}, 10\text{s}, 20\text{s})$ and the associated discrete state sequence $\mathcal{D} = (q_0, q_1, q_2) = (1, 2, 2)$. The system has two modes and the corresponding two vector fields are represented by

$$\mathbf{f}_1(\mathbf{x}(t)) = \begin{bmatrix} -3x_1 + 0.2x_2 \\ 2x_1 - 0.5x_2 \end{bmatrix} \tag{3.14}$$

$$\mathbf{f}_2(\mathbf{x}(t)) = \begin{bmatrix} -3 \cdot (x_1 + 1) + 0.2 \cdot (x_2 + 1) \\ 2 \cdot (x_1 + 1) - 0.5 \cdot (x_2 + 1) \end{bmatrix}. \tag{3.15}$$

The hybrid execution of the described system is depicted in Fig. 3.3. Starting with the active subsystem $\mathbf{f}_1(\cdot)$ the continuous-valued states evolve from $\mathbf{x}(t_0) = [-1, -1]^T$ to the stationary point $\mathbf{x}(t_1) = [0, 0]^T$. According to Definition 3.8, the hybrid execution

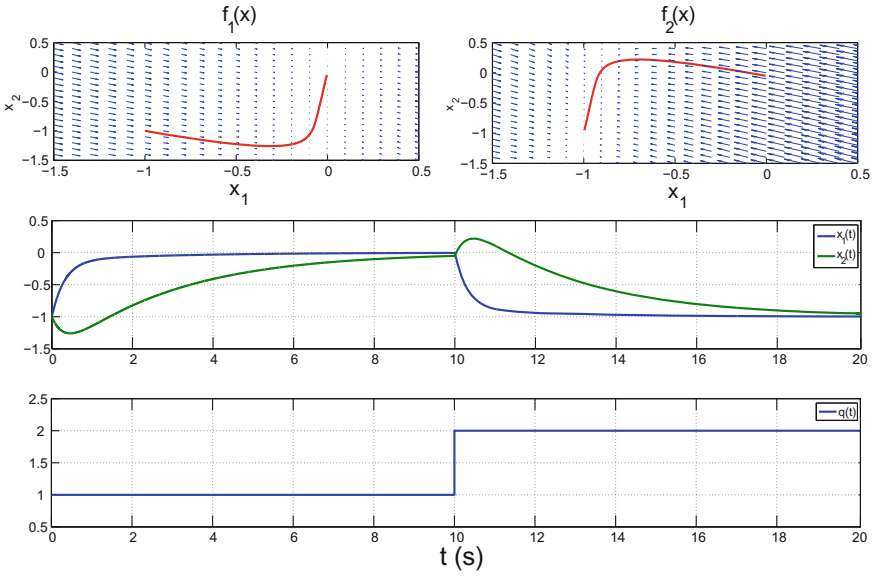


Fig. 3.3 Trajectories of the switched system represented by (3.14)–(3.15) (cf. Schori [50])

reveals one commanded switching at $t_1 = 10$s where the discrete state switches from $q(t_1^-) = 1$ to $q(t_1^+) = 2$ and consequently, the active vector field switches from $\mathbf{f}_1(\cdot)$ to $\mathbf{f}_2(\cdot)$. Please note that no state jump is implied with the switching ($\boldsymbol{\delta}_{(q(t_j^-),q(t_j^+))} = \mathbf{0} \quad \forall q(t_j^-), q(t_j^+) \in \hat{\mathcal{Q}}$) and therefore the states $\mathbf{x}(\cdot)$ remain continuous but clearly not continuously differentiable at t_1. After the switching, the continuous-valued states evolve to the stationary point $\mathbf{x}(t_2) = [-1, -1]^T$ of subsystem $\mathbf{f}_2(\cdot)$.

Now, we utilize the weakest assumptions to establish existence and uniqueness conditions. The following properties are assumed for all switched systems and hybrid systems without state discontinuities.

Let us assume the decomposition of the hybrid system into a hybrid trajectory.

1. since the right-hand side function $\mathbf{f}_{q_j}(\mathbf{x}_j(\cdot), \mathbf{u}_j(\cdot))$ is continuously differentiable w.r.t. $\mathbf{x}_j(\cdot)$ and $\mathbf{u}_j(\cdot)$ and since the control functions $\mathbf{u}_j(\cdot) \in L^\infty([t_j, t_{j+1}), \mathsf{U})$ are bounded and measurable with respect to t, the function $\mathbf{f}_{q_j}(\mathbf{x}_j(\cdot), \mathbf{u}_j(t))$ is measurable in $t \in [t_j, t_{j+1})$ for each fixed $\mathbf{x}_j(\cdot) \in \mathcal{AC}^\infty([t_j, t_{j+1}), \mathsf{X})$; and
2. the function $\mathbf{f}_{q_j}(\cdot, \mathbf{u}_j(t))$ is Lipschitz continuous with respect to each fixed $\tilde{\mathbf{x}} = \mathbf{x}_j(\cdot)$ for all $t \in [t_j, t_{j+1})$. That means, the Lipschitz condition must be fulfilled

$$||\mathbf{f}_{q_j}(\tilde{\mathbf{x}}_1, \mathbf{u}_j(t)) - \mathbf{f}_{q_j}(\tilde{\mathbf{x}}_2, \mathbf{u}_j(t))|| \leq L_f \cdot ||\tilde{\mathbf{x}}_1 - \tilde{\mathbf{x}}_2||, \quad \forall \tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2 \in \mathsf{X} \tag{3.16}$$

for each fixed $\mathbf{u}_j(\cdot) \in L^\infty([t_j, t_{j+1}), \mathsf{U})$ for all $t \in [t_j, t_{j+1})$ and $q_j \in \mathcal{D}$.

These two regularity conditions guarantee the existence of unique solutions to the N_{swt}-IVPs

$$\begin{aligned} \dot{\mathbf{x}}_j(t) &= \mathbf{f}_{q_j(t)}(\mathbf{x}_j(t), \mathbf{u}_j(t)), \quad \text{for a.e. } t \in \left[t_j, t_{j+1}\right) \\ \mathbf{x}_j(t_j^+) &= \mathbf{x}_{j-1}(t_j^-) \end{aligned}$$

for a fixed $q(t) \in \hat{\mathcal{Q}}$ on the interval $[t_j, t_{j+1})$. We can therefore conclude that an unique solution to the ODE $\dot{\mathbf{x}}(t) = \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t))$ evolving through the given initial hybrid state condition $(\mathbf{x}_0, q_0)$ over the entire interval $[t_0, t_f]$ exists. Using these regularity conditions we are able to state the existence and uniqueness theorem for hybrid systems with continuous states.

Theorem 3.1 (Existence and Uniqueness of Hybrid Systems with Absolutely Continuous State Trajectory (Shaikh [53])) *Given a switched system according to Definition 3.5, the continuous-valued state trajectory of the hybrid execution sequence (Definition 3.7) is assumed to be continuous. Then, the switched system possesses a unique hybrid execution sequence, passing through an initial hybrid state* $(\mathbf{x}_j(t_j^+), q_j(t_j^+))$ *up to the next controlled switching* (t_{j+1}, q_{j+1}) *of the switching sequence (Definition 3.8) or the final time* t_f. △

Proof The proof can be found in Shaikh [53]. □

Remark 3.1 Theorem 3.1 provides no conditions for the existence and uniqueness of a hybrid execution sequence for a continuous-valued state trajectory with discontinuities at the switchings.

3.2.5 Control and State Constraints, Admissible Sets, and Admissible Function Spaces

Many real continuous and hybrid systems may have imposed constraints D on the system's controls and states. These constraints can occur as mixed constraints or as separable constraints for the states and controls, which leads to slightly different optimality conditions. It is important to distinguish the structure of these different constraints.

Definition 3.9 (*Mixed Control-State Constraints, State Constraints, and Control Constraints*) A hybrid system $\mathbb{H}$ may possess a set D with different constraint types.

$N_{c_{x,u,q}}$-inequalities ($N_{c_{x,u,q}} \in \mathbb{N}_{\geq 0}$) which depend on $\mathbf{x}(\cdot)$ and $\mathbf{u}(\cdot)$, i.e.,

$$\mathbf{c}_{x,u,q}(\mathbf{x}(t), \mathbf{u}(t)) \leq \mathbf{0}, \quad \mathbf{x}(t) \in \mathsf{X}, \quad \mathbf{u}(t) \in \mathsf{U}, \quad q(t) \in \hat{\mathcal{Q}}, \quad \forall t \in [t_0, t_f] \tag{3.17}$$

are called *mixed control-state constraints*.

$N_{c_{x,q}}$-inequalities ($N_{c_{x,q}} \in \mathbb{N}_{\geq 0}$) which depend explicitly on $\mathbf{x}(\cdot)$, i.e.,

$$\mathbf{c}_{x,q}(\mathbf{x}(t)) \leq \mathbf{0}, \quad \mathbf{x}(t) \in \mathsf{X}, \quad q(t) \in \hat{\mathcal{Q}}, \quad \forall t \in [t_0, t_f] \tag{3.18}$$

are called *state constraints*.

$N_{c_{u,q}}$-inequalities ($N_{c_{u,q}} \in \mathbb{N}_{\geq 0}$) which depend explicitly on $\mathbf{u}(\cdot)$, i.e.,

$$\mathbf{c}_{u,q}(\mathbf{u}(t)) \leq \mathbf{0}, \quad \mathbf{u}(t) \in \mathsf{U}, \quad q(t) \in \hat{\mathcal{Q}}, \quad \forall t \in [t_0, t_f] \tag{3.19}$$

are called *control constraints*.

For continuous systems $\mathbb{C}$ the mixed control-state constraints (3.17) reduce to $N_{c_{x,u}}$-inequalities of the form

$$\mathbf{c}_{x,u}(\mathbf{x}(t), \mathbf{u}(t)) \leq \mathbf{0}, \quad \mathbf{x}(t) \in \mathsf{X}, \quad \mathbf{u}(t) \in \mathsf{U}, \quad \forall t \in [t_0, t_f],$$

the state constraints (3.18) reduce to N_{c_x}-inequalities of the form

$$\mathbf{c}_x(\mathbf{x}(t)) \leq \mathbf{0}, \quad \mathbf{x}(t) \in \mathsf{X}, \quad \forall t \in [t_0, t_f],$$

and the control constraints (3.19) reduce to N_{c_u}-inequalities of the form

$$\mathbf{c}_u(\mathbf{u}(t)) \leq \mathbf{0}, \quad \mathbf{u}(t) \in \mathsf{U}, \quad \forall t \in [t_0, t_f]. \qquad \triangle$$

Examples are

$$\begin{aligned} x(t) &\leq 5, && \text{box constraint} \\ x(t) &\leq u(t), && \text{mixed control-state constraint} \\ x_1(t) &\leq x_2(t), && \text{state constraint.} \end{aligned}$$

Definition 3.10 (*Admissible Control Sets and Admissible State Set*) For a continuous system $\mathbb{C}$, an admissible control set which depends on the control constraints $\mathbf{c}_u(\cdot)$ is defined by

$$\hat{\mathcal{U}}(t) = \Big\{\mathbf{u}(t) \in \mathsf{U} \;\Big|\; \mathbf{c}_u(\mathbf{u}(t)) \leq \mathbf{0}\Big\}.$$

An admissible state set which depends on the state constraints $\mathbf{c}_x(\cdot)$ is defined by

$$\hat{\mathcal{X}}(t) = \Big\{\mathbf{x}(t) \in \mathsf{X} \;\Big|\; \mathbf{c}_x(\mathbf{x}(t)) \leq \mathbf{0}\Big\}.$$

An admissible state-dependent control set which depends on the mixed control-state constraints $\mathbf{c}_{x,u}(\cdot)$ is defined by

$$\hat{\mathcal{U}}(\mathbf{x}(t), t) = \Big\{\mathbf{u}(t) \in \mathsf{U} \;\Big|\; \mathbf{c}_{x,u}(\mathbf{x}(t), \mathbf{u}(t)) \leq \mathbf{0}\Big\}.$$

For a hybrid system $\mathbb{H}$, an admissible control set which depends on the control constraints $\mathbf{c}_{u,q}(\cdot)$ is defined by

$$\hat{\mathcal{U}}(q(t), t) = \Big\{\mathbf{u}(t) \in \mathsf{U} \;\Big|\; \mathbf{c}_{u,q}(\mathbf{u}(t)) \leq \mathbf{0}\Big\}.$$

An admissible state set which depends on the state constraints $\mathbf{c}_{x,q}(\cdot)$ is defined by

$$\hat{\mathcal{X}}(q(t), t) = \Big\{\mathbf{x}(t) \in \mathsf{X} \;\Big|\; \mathbf{c}_{x,q}(\mathbf{x}(t)) \leq \mathbf{0}\Big\}.$$

The union of the admissible control sets $\bigcup_{1 \leq q \leq N_q} \hat{\mathcal{U}}(q(t), t)$ are not necessarily be connected.

△

For the special case that the nonlinear constraints $\mathbf{c}_u(\cdot)$ and $\mathbf{c}_x(\cdot)$ degrade to linear ones we denote this constraint type as *box constraints*.

Definition 3.11 (*Box Constraints for Controls and States*) If the controls satisfy simple box constraints of the form

$$u_{i,q}^{min} \leq u_i(t) \leq u_{i,q}^{max}, \quad \forall t \in [t_0, t_f], \quad i \in \{1, \ldots, N_{u,q}\}, \quad \forall q \in \hat{\mathcal{Q}}$$

then the set of admissible controls is defined as

$$\hat{\mathcal{U}}(q(t)) = \left[u_{1,q}^{min}, u_{1,q}^{max}\right] \times \cdots \times \left[u_{N_u,q}^{min}, u_{N_u,q}^{max}\right].$$

If the states satisfy simple box constraints of the form

$$x_i^{min} \leq x_i(t) \leq x_i^{max}, \quad \forall t \in [t_0, t_f], \quad i \in \{1, \ldots, N_x\}$$

then the set of admissible states is defined as

$$\hat{\mathcal{X}} = \left[x_1^{min}, x_1^{max}\right] \times \cdots \times \left[x_{N_x}^{min}, x_{N_x}^{max}\right].$$

The bounds $u_{1,q}^{min}, \ldots, u_{N_u,q}^{min}$ and $x_1^{min}, \ldots, x_{N_x}^{min}$ are called *lower bounds* and the bounds $u_{1,q}^{max}, \ldots, u_{N_u,q}^{max}$ and $x_1^{max}, \ldots, x_{N_x}^{max}$ are called *upper bounds*. The lower and upper bounds must satisfy $u_{i,q}^{min} \leq u_{i,q}^{max}$ and $x_i^{min} \leq x_i^{max}$. △

Since control constraints with lower and upper bounds can be expressed as $u_{i,q}^{min} - u_i(\cdot) \leq 0$ and $u_i(\cdot) - u_{i,q}^{max} \leq 0$, respectively, all box control constraints can be stated in the form $\mathbf{c}_u(\cdot) \leq \mathbf{0}$. Please note that $u_{i,q}^{min} = -\infty$ and $u_{i,q}^{max} = \infty$ are explicitly allowed and that such constraints can be omitted in the problem formulation. The same applies to the box state constraints.

The admissible sets in Definition 3.10 are defined pointwise for each $t \in [t_0, t_f]$. The corresponding function spaces for the admissible state and control functions are defined as follows.

Definition 3.12 (*Admissible Control Function Spaces and Admissible State Function Space*) For continuous systems $\mathbb{C}$, the continuous-valued control function $\mathbf{u}(\cdot)$ is admissible on the time interval $[t_0, t_f]$, if

$$\mathbf{u}(\cdot) \in \mathcal{U} := \left\{\mathbf{u}(\cdot) \in L^\infty([t_0, t_f], \mathsf{U}) \;\middle|\; \mathbf{c}_u(\mathbf{u}(t)) \leq \mathbf{0},\ \forall t \in [t_0, t_f]\right\}$$

applies, where $\mathcal{U}$ is the admissible function space for the continuous-valued controls.

The continuous-valued state function $\mathbf{x}(\cdot)$ is admissible on the time interval $[t_0, t_f]$, if

$$\mathbf{x}(\cdot) \in \mathcal{X} := \left\{\mathbf{x}(\cdot) \in \mathcal{AC}^\infty([t_0, t_f], \mathsf{X}) \;\middle|\; \mathbf{c}_x(\mathbf{x}(t)) \leq \mathbf{0},\ \forall t \in [t_0, t_f]\right\}$$

applies, where $\mathcal{X}$ is the admissible function space for continuous-valued states.

For mixed control-state constraints $\mathbf{c}_{x,u}(\cdot)$, the continuous-valued control function $\mathbf{u}(\cdot)$ is admissible on the time interval $[t_0, t_f]$, if

$$\mathcal{U}(\mathbf{x}(\cdot)) := \left\{\mathbf{u}(\cdot) \in L^\infty([t_0, t_f], \mathsf{U}) \;\middle|\; \mathbf{c}_{x,u}(\mathbf{x}(t), \mathbf{u}(t)) \leq \mathbf{0},\ \forall t \in [t_0, t_f]\right\}$$

applies, where $\mathcal{U}(\mathbf{x}(\cdot))$ is the admissible function space for the continuous-valued controls which depends on the continuous-valued states $\mathbf{x}(\cdot) \in L^\infty([t_0, t_f], \mathsf{X})$.

For hybrid systems $\mathbb{H}$, the continuous-valued control function $\mathbf{u}(\cdot)$ is admissible on the time interval $[t_0, t_f]$, if

$$\mathbf{u}(\cdot) \in \mathcal{U}(q(\cdot)) := \left\{\mathbf{u}(\cdot) \in L^\infty([t_0, t_f], \mathsf{U}) \;\middle|\; \mathbf{c}_{u,q}(\mathbf{u}(t)) \leq \mathbf{0},\ \forall t \in [t_0, t_f]\right\}$$

applies, where $\mathcal{U}(q(\cdot))$ is the admissible function space for the continuous-valued controls which depends on the discrete state $q(\cdot) \in L^\infty([t_0, t_f], \hat{\mathcal{Q}})$.

The continuous-valued state function $\mathbf{x}(\cdot)$ is admissible on the time interval $[t_0, t_f]$, if

$$\mathbf{x}(\cdot) \in \mathcal{X}(q(\cdot)) := \left\{\mathbf{x}(\cdot) \in L^\infty([t_0, t_f], \mathsf{X}) \;\middle|\; \mathbf{c}_{x,q}(\mathbf{x}(t)) \leq \mathbf{0},\ \forall t \in [t_0, t_f]\right\}$$

applies, where $\mathcal{X}(q(\cdot))$ is the admissible function space for the continuous-valued states which depends on the discrete state $q(\cdot) \in L^\infty([t_0, t_f], \hat{\mathcal{Q}})$.

The discrete control function $\varpi(\cdot)$ is admissible on the time interval $[t_0, t_f]$, if

$$\varpi(\cdot) \in \mathsf{B}(q(\cdot)) := \left\{\varpi(\cdot) \in L^\infty([t_0, t_f], \hat{\mathsf{B}}_q) \;\middle|\; \varpi(t) \in \hat{\mathsf{B}}_q,\ \forall t \in [t_0, t_f]\right\}$$

applies, where $\mathsf{B}(q(\cdot))$ is the admissible function space for the discrete control.

The discrete state function $q(\cdot)$ is admissible on the time interval $[t_0, t_f]$, if

$$q(\cdot) \in \mathcal{Q} := \left\{q(\cdot) \in L^\infty([t_0, t_f], \hat{\mathcal{Q}}) \;\middle|\; q(t) \in \hat{\mathcal{Q}},\ \forall t \in [t_0, t_f]\right\}$$

applies, where $\mathcal{Q}$ is the admissible function space for the discrete state. △

3.2.6 Reformulation of Switched Systems

A reformulation of a switched system $\mathbb{S}$ can be helpful. The discrete state $q(\cdot)$ is re-interpreted as piecewise constant binary controls $\boldsymbol{\sigma} : [t_0, t_f] \to \hat{\Omega}$ that defines which subsystem $\mathbf{f}_{q(t)}(\cdot)$ is active at time t. $\hat{\Omega} := \{0, 1\}^{N_q}$ is the admissible set for the binary controls. $\boldsymbol{\sigma}(\cdot)$ is called an admissible binary control function on the time interval $[t_0, t_f]$, if

$$\boldsymbol{\sigma}(\cdot) \in \Omega := \left\{\boldsymbol{\sigma}(\cdot) \in L^\infty([t_0, t_f], \hat{\Omega}) \;\middle|\; \boldsymbol{\sigma}(t) \in \hat{\Omega},\ \forall t \in [t_0, t_f]\right\}$$

applies, where Ω is the admissible function space.

To ensure that only one subsystem is active at any time, the constraint

$$\begin{aligned} &\sum_{q=1}^{N_q} \sigma_q(t) = 1, \quad \forall t \in [t_0, t_f], \\ &\sigma_1, \ldots, \sigma_{N_q} \in \{0, 1\} \end{aligned} \tag{3.20}$$

must be fulfilled and is added to the system description, where σ_q refers to the q-th entry in the binary controls $\boldsymbol{\sigma}(\cdot)$. The system's differential equation can then be re-written in affine form (Xu [62], Alamir and Attia [3]) as

$$\dot{\mathbf{x}}(t) = \mathbf{F}(\mathbf{x}(t), \boldsymbol{\sigma}(t), \mathbf{u}(t)) = \sum_{q=1}^{N_q} \sigma_q(t) \cdot \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)), \quad \text{for a.e. } t \in [t_0, t_f] \tag{3.21}$$

where the right-hand side function $\mathbf{F} : \mathsf{X} \times \hat{\Omega} \times \mathsf{U} \to \mathsf{X}$ is measurable in t. With a concatenated control vector $\boldsymbol{\rho}(\cdot) \in \mathcal{U} \times \Omega$, which contains the continuous-valued as well as the binary controls,

$$\boldsymbol{\rho}(t) = [\mathbf{u}(t), \boldsymbol{\sigma}(t)], \quad \forall t \in [t_0, t_f] \tag{3.22}$$

the system can be written in the conventional form

$$\dot{\mathbf{x}}(t) = \mathbf{F}(\mathbf{x}(t), \boldsymbol{\rho}(t)), \quad \text{for a.e. } t \in [t_0, t_f]. \tag{3.23}$$

Definition 3.13 (*Binary Switched System*) The dynamics of the switched system (3.7)–(3.8) can be equivalently reformulated to a binary switched system using (3.20)–(3.23) and represented by

$$\begin{aligned}
\dot{\mathbf{x}}(t) &= \mathbf{F}(\mathbf{x}(t), \boldsymbol{\rho}(t)) = \sum_{q=1}^{N_q} \sigma_q(t) \cdot \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)), \quad \text{for a.e. } t \in [t_0, t_f], \\
\sum_{q=1}^{N_q} \sigma_q(t) &= 1, \quad \forall t \in [t_0, t_f], \\
\mathbf{x}(t_0) &= \mathbf{x}_0.
\end{aligned}$$

The collection of vector fields reduces to one single entry $\mathcal{F} = \mathbf{F}(\cdot)$.

A binary switched system without state jumps is then defined by

$$\mathbb{S}_3 := (\hat{\mathcal{Q}}, \mathsf{X}, \mathsf{U}, \mathcal{F}, \hat{\Omega}, \mathsf{D})$$

where $\hat{\Omega}$ is the set of admissible binary controls.

△

It should be noted that the binary character of $\boldsymbol{\sigma}(\cdot)$ makes the admissible control set $\hat{\mathcal{U}} \times \hat{\Omega}$ disjoint. More specifically, the admissible set is split into N_q-subsets which

are not connected to each other and in particular the admissible set is nonconvex, which is problematic for the convergence of iterative optimal control solvers, as will be seen later.

3.3 Optimal Control Problem Formulations

It is often of major interest, to define a control law for the continuous controls as well as for the discrete automaton, that leads to a desired behavior. Engineers and scientists often describe such problems in the form of a cost functional that can represent time, energy, and money costs or their respective combinations. The task of finding the controls relates to the minimization of the cost functional. A respective formulation is denominated as an optimal control problem, or more specifically, as a switched optimal control problem, when the underlying system is a switched system. In this section we consider important problem formulations.

3.3.1 Functionals

The finite-dimensional optimization, which is the topic of Chap. 2, aims at finding optimal points in the Euclidean vector space $\mathbb{R}^{N_y}$, which minimizes a given cost function. Now, in the setting of hybrid optimal control we are seeking for functions of time in an infinite-dimensional function space, which minimizes a given cost functional. Hereby, a functional is a mapping of a function to a single real value. We might also say, a functional is a function of a function.

In optimal control theory a distinction between three types of cost functionals is usually made (Ioffe and Tihomirov [33]), where all three types can be transformed into each other, as will be outlined later. Therefore, we will only use one symbol $\phi(\cdot)$ for a general cost functional.

Functionals of purely integral-type are of the following form $\phi : \mathcal{X} \times \mathcal{U}(\cdot) \to \mathbb{R}$:

$$\phi(\mathbf{x}(\cdot), \mathbf{u}(\cdot)) = \int_{t_0}^{t_f} l_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)) \, \mathrm{d}t \tag{3.24}$$

where $\mathcal{X} \times \mathcal{U}(\cdot)$ is the cartesian product of the function space for the states and the function space for the controls. The functionals of type (3.24) are called *integral functionals* or Lagrange functionals. The function $l_q : \mathsf{X} \times \mathsf{U} \to \mathbb{R}$ under the integral is real-valued and is called either *Lagrangian*, *running cost*, or *instantaneous cost*. The reader should be aware of the notation. On the one hand, $\phi(\cdot)$ means a functional of the functions $\mathbf{x}(\cdot)$ and $\mathbf{u}(\cdot)$. On the other hand, the function $l_q(\cdot)$ assigns a real value to a single point of the trajectory $(\mathbf{x}(t), \mathbf{u}(t))$ in the Euclidean space. Then, for a given initial set $(t_0; \mathbf{x}_0)$, the dynamic behaviors are parameterized by control

functions $\mathbf{u}(\cdot)$. Thus, functionals of type (3.24) assign a cost value to each feasible control $\mathbf{u}(\cdot)$.

Endpoint functionals with the mapping $\phi : \mathcal{X} \to \mathbb{R}$ depend only on the terminal values of the state functions, i.e., $\mathbf{x}(t_f)$, and assign a cost value to them:

$$\phi(\mathbf{x}(\cdot)) = m(\mathbf{x}(t_f)),$$

where $m(\cdot)$ are called *endpoint functionals* or *Mayer functionals*. Finally, combinations of both types are called *mixed functionals* or *Bolza functionals*:

$$\phi(\mathbf{x}(\cdot), \mathbf{u}(\cdot)) = m(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} l_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)) \, \mathrm{d}t.$$

Different functional types are necessary because some numerical solution methods require different formulations. For example, the solution method *dynamic programming* requires an optimal control problem to be formulated as Bolza-type.

3.3.2 Boundary Conditions

To obtain a well-posed optimal control problem boundary conditions are needed. Boundary conditions define initial and/or final values of the desired trajectories. The solution of a system of ordinary differential equations usually needs the prescription of one boundary value per differential equation to get a unique solution. For OCPs up to $2N_x$ boundary values can be prescribed, because the necessary conditions introduce N_x adjoint differential equations additionally to the N_x differential equations of the dynamic system, as will be seen in Chap. 4. If fewer boundary conditions are prescribed, the necessary conditions will define the missing boundary conditions by the transversality conditions.

Definition 3.14 (*Coupled and Decoupled Boundary Conditions*) The boundary conditions $\boldsymbol{\psi}_0(\cdot)$ impose N_{ψ_0}-equality restrictions on the initial states $\mathbf{x}(t_0)$ only with

$$\boldsymbol{\psi}_0(\mathbf{x}(t_0)) = \mathbf{0}.$$

The boundary conditions $\boldsymbol{\psi}_f(\cdot)$ impose N_{ψ_f}-equality restrictions on the final states $\mathbf{x}(t_f)$ only with

$$\boldsymbol{\psi}_f(\mathbf{x}(t_f)) = \mathbf{0}.$$

Such conditions are classified as *decoupled boundary conditions* (Kirches [35]) because they are separable over the entire time horizon.

The more general case is *coupled boundary conditions* $\boldsymbol{\psi}(\cdot)$ with

$$\boldsymbol{\psi}(\mathbf{x}(t_0), \mathbf{x}(t_f)) = \mathbf{0}$$

which impose N_ψ-equality restrictions that couple the states at the endpoints. △

The boundary conditions $\boldsymbol{\psi}_0(\cdot)$, $\boldsymbol{\psi}_f(\cdot)$, and $\boldsymbol{\psi}(\cdot)$ are usually nonlinear in $\mathbf{x}(t_0)$ and $\mathbf{x}(t_f)$. For the special case that the boundary conditions degrade to a linear form, then the initial and final states can be separately stated as

$$\begin{aligned}\mathbf{x}(t_0) &= \mathbf{x}_0\\ \mathbf{x}_{[\mathcal{I}_f]}(t_f) &= \mathbf{x}_f,\end{aligned}$$

where the set $\mathcal{I}_f$ specifies which final states $\mathbf{x}_f$ are given explicitly. In contrast, the initial states $\mathbf{x}_0$ are assumed to be all given. This type of boundary condition is of major importance because it appears in many practical applications. Since it is considered exclusively in the examples given we restrict the problem formulations and theoretical derivations to this boundary type only.

3.3.3 Continuous Optimal Control Problem

In an optimal control problem for a continuous system $\mathbb{C}$, the task is finding an admissible control function $\mathbf{u}(\cdot) \in L^\infty([t_0, t_f], \mathsf{U})$ generating the corresponding admissible state function $\mathbf{x}(\cdot) \in \mathcal{AC}^\infty([t_0, t_f], \mathsf{X})$ such that all imposed constraints from the set $\mathsf{D} = \{\mathbf{c}_{x,u}(\cdot), \mathbf{c}_x(\cdot), \mathbf{c}_u(\cdot)\}$ and boundary conditions are satisfied and the cost functional $\phi(\mathbf{u}(\cdot))$ is minimized.

Definition 3.15 (*Continuous Optimal Control Problem*) Let a continuous system $\mathbb{C}$ be given by Definition 3.1. A continuous *optimal control problem* (OCP) is a (constrained) infinite-dimensional optimization problem where the optimal control functions $\mathbf{u}^*(\cdot)$ have to be chosen, such that a given functional

$$\phi\left(\mathbf{u}^*(\cdot)\right) = \min_{\mathbf{u}(\cdot)\in\mathcal{U}} \phi(\mathbf{u}(\cdot)) \tag{3.25}$$

subject to the continuous system

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \quad \text{for a.e. } t \in [t_0, t_f], \tag{3.26}$$

imposed boundaries

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{3.27}$$

$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f, \tag{3.28}$$

and imposed constraints on the controls and states

$$\mathbf{u}(\cdot) \in \mathcal{U} \tag{3.29}$$
$$\mathbf{x}(\cdot) \in \mathcal{X} \tag{3.30}$$

is minimized.

△

If the functional $\phi(\cdot)$ is integral-type, then the problem (3.25)–(3.30) is called *Lagrange problem*; if the functional is endpoint, then the problem is called *Mayer problem*; and, finally, if the functional is mixed, then the problem is called *Bolza problem* (Ioffe and Tihomirov [33]). It should be noted that a minimization problem can always be converted into a maximization problem by changing the sign of $\phi(\cdot)$.

For all problem definitions encountered in this book, the initial condition $\mathbf{x}_0$ is assumed to be known at the fixed initial time t_0. The final time t_f can be fixed or free. Problems with free final time require an extension of the formulation, see Sect. 3.3.7.

We say the control and state functions are feasible, if the system (3.26) is forced to satisfy all imposed constraints (3.27), (3.28), (3.29), and (3.30).

Optimal controls of control-affine systems have continuous-valued optimal control trajectories of bang–bang type. That means, the trajectories of $u_i(\cdot)$ may take values only from the set $\{u_i^{min}, u_i^{max}\}$ with the exception of some singular arcs. This fact will be used in the later chapters to mimic binary switches by the embedding approach. For control-affine systems the Lagrangian must have the following affine structure:

$$l_0(\mathbf{x}(t)) + \mathbf{l}_1^T(\mathbf{x}(t)) \cdot \mathbf{u}(t), \tag{3.31}$$

where $l_0 : \mathsf{X} \to \mathbb{R}$ and $\mathbf{l}_1 : \mathsf{X} \to \mathsf{U}$.

Definition 3.16 (*Affine Optimal Control Problem*) Given a continuous system $\mathbb{C}$ by Definition 3.1, an optimal control problem with linear controls (3.5) and affine Lagrangian (3.31) can be stated as

$$\phi\left(\mathbf{u}^*(\cdot)\right) = \min_{\mathbf{u}(\cdot)\in\mathcal{U}} m(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} l_0(\mathbf{x}(t)) + \mathbf{l}_1^T(\mathbf{x}(t)) \cdot \mathbf{u}(t)\, \mathrm{d}t \tag{3.32}$$

subject to

$$\dot{\mathbf{x}}(t) = \mathbf{f}_0(\mathbf{x}(t)) + \sum_{i=1}^{N_u} \mathbf{f}_i(\mathbf{x}(t)) \cdot u_i(t), \quad \text{for a.e. } t \in [t_0, t_f] \tag{3.33}$$
$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{3.34}$$
$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f \tag{3.35}$$
$$\mathbf{x}(\cdot) \in \mathcal{X}.$$

The formulation (3.32)–(3.35) is called an *affine optimal control problem*. △

3.3.4 Hybrid Optimal Control Problem

For a hybrid optimal control problem, the goal is to find the continuous-valued controls $\mathbf{u}(\cdot) \in \mathcal{U}(q(\cdot))$ and the discrete control $\varpi(\cdot) \in \mathsf{B}(q(\cdot))$ for a hybrid system $\mathbb{H}$, such that desired characteristics are met.

The higher complexity accounts for the selection of a subsystem from the set $\mathcal{F}$ such that all imposed constraints from the set $\mathsf{D} = \{\mathbf{c}_{x,u,q}(\cdot), \mathbf{c}_{x,q}(\cdot), \mathbf{c}_{u,q}(\cdot)\}$ and boundary conditions are satisfied and the cost functional is still minimized.

The optimal control problem for a hybrid system without state jumps can be stated as follows:

Definition 3.17 (*Hybrid Optimal Control Problem without Reset Functions*) Given a hybrid system $\mathbb{H}_1$ by Definition 3.3. Then, an optimal control problem can be stated as Bolza problem

$$\phi\left(\mathbf{u}^*(\cdot), \varpi^*(\cdot)\right) = \min_{\mathbf{u}(\cdot)\in\mathcal{U}(q(\cdot)),\ \varpi(\cdot)\in\mathsf{B}(q(\cdot))} m(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} l_{q(t)}(\mathbf{x}(t), \mathbf{u}(t))\ \mathrm{d}t \tag{3.36}$$

subject to

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)), \quad \text{for a.e. } t \in [t_0, t_f] \tag{3.37}$$

$$q(t_j^+) = q(t_j^-) + \varpi(t_j), \quad t_j \in \Theta_t \tag{3.38}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{3.39}$$

$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f \tag{3.40}$$

$$q(\cdot) \in \mathcal{Q},\ \mathbf{x}(\cdot) \in \mathcal{X}(q(\cdot)).$$

The formulation (3.36)–(3.40) is called a *hybrid optimal control problem* (HOCP) with controlled switching and without state jumps. △

According to the additional degree-of-freedom, we define feasibility of an HOCP as follows:

Definition 3.18 (*Feasibility of HOCPs*) A tuple $(\mathbf{x}(\cdot), q(\cdot), \mathbf{u}(\cdot), \varpi(\cdot))$ of functions is said to be feasible if the ODE, boundary conditions, and all constraints of the HOCP formulation are satisfied. △

The consideration of a hybrid system $\mathbb{H}_2$ with reset requires an extension of the hybrid optimal control problem from above.

Definition 3.19 (*Hybrid Optimal Control Problem with Reset Functions*) Given a hybrid system $\mathbb{H}_2$ by Definition 3.4. Then, an optimal control problem can be stated as Bolza problem

$$\phi\left(\mathbf{u}^*(\cdot), \varpi^*(\cdot)\right) = \min_{\mathbf{u}(\cdot)\in\mathcal{U}(q(\cdot)),\ \varpi(\cdot)\in\mathsf{B}(q(\cdot))} m(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} l_{q(t)}(\mathbf{x}(t), \mathbf{u}(t))\,\mathrm{d}t \tag{3.41}$$

subject to

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)), \qquad \text{for a.e. } t \in [t_0, t_f] \tag{3.42}$$
$$q(t_j^+) = q(t_j^-) + \varpi(t_j), \qquad t_j \in \Theta_t \tag{3.43}$$
$$\mathbf{x}(t_j^+) = \mathbf{x}(t_j^-) + \boldsymbol{\delta}_{(q(t_j^-),q(t_j^+))}\left(\mathbf{x}(t_j^-)\right) \tag{3.44}$$
$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{3.45}$$
$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f \tag{3.46}$$
$$q(\cdot) \in \mathcal{Q},\ \mathbf{x}(\cdot) \in \mathcal{X}(q(\cdot)).$$

The formulation (3.41)–(3.46) is called a *hybrid optimal control problem with state jumps*. △

In general, the addition of the reset condition (3.55) leads to a nonsmooth optimal control problem.

3.3.5 Switched Optimal Control Problem

According to the assumption made in Sect. 3.2.3, we can directly use the discrete state $q(\cdot)$ as control variable for a switched system $\mathbb{S}$. Thus, for a switched optimal control problem, the goal is to find the continuous-valued controls $\mathbf{u}(\cdot) \in \mathcal{U}(q(\cdot))$ and discrete states $q(\cdot) \in \mathcal{Q}$, such that desired characteristics are met.

The consideration of an optimal control problem for a switched system without state jumps $\mathbb{S}_1$ requires some modifications in the formulation:

Definition 3.20 (*Switched Optimal Control Problem without Reset Functions*) Given a switched system $\mathbb{S}_1$ by Definition 3.5. Then, an optimal control problem can be stated as Bolza problem

$$\phi\left(q^*(\cdot), \mathbf{u}^*(\cdot)\right) = \min_{q(\cdot)\in\mathcal{Q},\ \mathbf{u}(\cdot)\in\mathcal{U}(q(\cdot))} m(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} l_{q(t)}(\mathbf{x}(t), \mathbf{u}(t))\,\mathrm{d}t \tag{3.47}$$

subject to

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)), \qquad \text{for a.e. } t \in [t_0, t_f] \tag{3.48}$$
$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{3.49}$$
$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f \tag{3.50}$$
$$\mathbf{x}(\cdot) \in \mathcal{X}(q(\cdot)).$$

The formulation (3.47)–(3.50) is called a *switched optimal control problem* (SOCP) without state jumps.

△

Let us adapt the definition of feasibility for a SOCP as follows:

Definition 3.21 (*Feasibility of SOCPs*) A tuple $(\mathbf{x}(\cdot), q(\cdot), \mathbf{u}(\cdot))$ of functions is said to be feasible if the ODE, boundary conditions, and all constraints of the SOCP formulation are satisfied. △

Definition 3.22 (*Optimal Control Function, Optimal Continuous State Function, and Optimal Discrete State Function*) For any *optimal tuple* $(\mathbf{x}^*(\cdot), q^*(\cdot), \mathbf{u}^*(\cdot))$ satisfying

$$(\mathbf{x}^*(\cdot), q^*(\cdot), \mathbf{u}^*(\cdot)) = arg \min_{\mathbf{x}(\cdot)\in\mathcal{X}(q(\cdot)),\ q(\cdot)\in\mathcal{Q},\ \mathbf{u}(\cdot)\in\mathcal{U}(q(\cdot))} \phi(\mathbf{x}(\cdot), q(\cdot), \mathbf{u}(\cdot)),$$

$\mathbf{u}^*(\cdot)$ is called an *optimal control function*, $\mathbf{x}^*(\cdot)$ is called an *optimal continuous state function*, and $q^*(\cdot)$ is called an *optimal discrete state function*. △

The consideration of a switched system $\mathbb{S}_2$ with reset requires an extension of the switched optimal control problem from above.

Definition 3.23 (*Switched Optimal Control Problem with Reset Functions*) Given a switched system $\mathbb{S}_2$ by Definition 3.6. Then, an optimal control problem can be stated as Bolza problem

$$\phi\left(q^*(\cdot), \mathbf{u}^*(\cdot)\right) = \min_{q(\cdot)\in\mathcal{Q},\ \mathbf{u}(\cdot)\in\mathcal{U}(q(\cdot))} m(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} l_{q(t)}(\mathbf{x}(t), \mathbf{u}(t))\ \mathrm{d}t \tag{3.51}$$

subject to

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)), \qquad \text{for a.e. } t \in [t_0, t_f] \tag{3.52}$$
$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{3.53}$$
$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f \tag{3.54}$$
$$\mathbf{x}(t_j^+) = \mathbf{x}(t_j^-) + \boldsymbol{\delta}_{(q(t_j^-),q(t_j^+))}\left(\mathbf{x}(t_j^-)\right) \tag{3.55}$$
$$\mathbf{x}(\cdot) \in \mathcal{X}(q(\cdot)).$$

The formulation (3.51)–(3.55) is called a *switched optimal control problem with state jumps*. △

3.3.6 Binary Switched Optimal Control Problem

Applying the reformulation of Sect. 3.2.6 to the discrete state $q(\cdot)$ one obtains piecewise constant binary functions $\boldsymbol{\sigma} : [t_0, t_f] \to \hat{\Omega}$. An additional constraint (3.20)

assures that only one subsystem is active. The SOCP from Definition 3.20 can then equivalently be reformulated to an optimal control problem for binary switched systems. The numerical solution of binary switched optimal control problems plays an essential part of this book.

Definition 3.24 (*Binary Switched Optimal Control Problem*) Let a switched system $\mathbb{S}_3$ be given by Definition 3.13. Then, an optimal control problem formulation can be stated as Bolza problem

$$\phi\left(\boldsymbol{\sigma}^*(\cdot), \mathbf{u}^*(\cdot)\right) = \min_{\boldsymbol{\sigma}(\cdot)\in\Omega,\ \mathbf{u}(\cdot)\in\mathcal{U}(q(\cdot))} m(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} \sum_{q=1}^{N_q} \sigma_q(t) \cdot l_q(\mathbf{x}(t), \mathbf{u}(t))\, \mathrm{d}t \tag{3.56}$$

subject to

$$\dot{\mathbf{x}}(t) = \sum_{q=1}^{N_q} \sigma_q(t) \cdot \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)), \quad \text{for a.e. } t \in [t_0, t_f] \tag{3.57}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{3.58}$$

$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f \tag{3.59}$$

$$1 - \sum_{q=1}^{N_q} \sigma_q(t) = 0, \qquad \forall t \in [t_0, t_f] \tag{3.60}$$

$$\mathbf{x}(\cdot) \in \mathcal{X}(q(\cdot)).$$

The formulation (3.56)–(3.60) is called a *binary switched optimal control problem* (BSOCP). △

3.3.7 Transformations of Optimal Control Problems

Several useful transformation techniques are discussed that allow to transform fairly general optimal control problems to standard form.

3.3.7.1 Transformation of Bolza-, Lagrange-, and Mayer-type Problems

Optimal control problems of the Bolza-, Lagrange-, and Mayer-type problems can be equivalently transformed into each other. This is often necessary because different numerical procedures require that the problem formulations are tailored to the algorithms. We start by transforming Bolza-type problems to Mayer-type problems.

Again, the Bolza-type problem consists of a final state cost $m(\mathbf{x}(t_f))$ and a set of N_q-Lagrange terms:

$$\phi(q(\cdot), \mathbf{u}(\cdot)) = m(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} l_{q(t)}(\mathbf{x}(t), \mathbf{u}(t))\, \mathrm{d}t. \tag{3.61}$$

By introducing an additional state $x_{N_x+1}(\cdot)$ with the time derivative $\dot{x}_{N_x+1}(t) = l_{q(t)}(\mathbf{x}(t), \mathbf{u}(t))$ and the initial condition $x_{N_x+1}(t_0) = 0$, the cost functional (3.61) can be easily formulated as endpoint functional. Then, the functional (3.61) becomes

$$\phi(q(\cdot), \mathbf{u}(\cdot)) = m(\mathbf{x}(t_f)) + x_{N_x+1}(t_f)$$

and the dimension of the dynamical system is increased by 1

$$\dot{\hat{\mathbf{x}}}(t) = \begin{bmatrix} \dot{x}_1(t) \\ \vdots \\ \dot{x}_{N_x}(t) \\ \hline \dot{x}_{N_x+1}(t) \end{bmatrix} = \left[\begin{array}{c} \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)) \\ \hline l_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)) \end{array}\right]. \tag{3.62}$$

Analogously, Lagrange-type problems

$$\phi(q(\cdot), \mathbf{u}(\cdot)) = \int_{t_0}^{t_f} l_{q(t)}(\mathbf{x}(t), \mathbf{u}(t))\, \mathrm{d}t \tag{3.63}$$

can be converted into Mayer-type problems by introducing an additional state $x_{N_x+1}(\cdot)$ with the differential equation $\dot{x}_{N_x+1}(t) = l_{q(t)}(\mathbf{x}(t), \mathbf{u}(t))$ and the initial condition $x_{N_x+1}(t_0) = 0$. Then, the functional (3.63) becomes

$$\phi(q(\cdot), \mathbf{u}(\cdot)) = x_{N_x+1}(t_f)$$

with the extended dynamical system (3.62).

Mayer-type problems

$$\phi(q(\cdot), \mathbf{u}(\cdot)) = m(\mathbf{x}(t_f)) \tag{3.64}$$

can be transformed to Lagrange-type problems by introducing an additional state $x_{N_x+1}(\cdot)$ with the differential equation $\dot{x}_{N_x+1}(t) = 0$ and the initial condition $x_{N_x+1}(t_0) = (t_f - t_0)^{-1} \cdot m(\mathbf{x}(t_f))$. Then, the functional (3.64) becomes

$$\phi(q(\cdot), \mathbf{u}(\cdot)) = \int_{t_0}^{t_f} x_{N_x+1}(t)\, \mathrm{d}t$$

with the extended dynamical system

$$\dot{\hat{\mathbf{x}}}(t) = \begin{bmatrix} \dot{x}_1(t) \\ \vdots \\ \dot{x}_{N_x}(t) \\ \hline \dot{x}_{N_x+1}(t) \end{bmatrix} = \left[\begin{array}{c} \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)) \\ \hline 0 \end{array}\right]. \tag{3.65}$$

Bolza-type problems (3.61) can be analogously transformed to Lagrange-type problems by introducing an additional state $x_{N_x+1}(\cdot)$ with the differential equation $\dot{x}_{N_x+1}(t) = 0$ and the initial condition $x_{N_x+1}(t_0) = (t_f - t_0)^{-1} \cdot m(\mathbf{x}(t_f))$. Then, the functional (3.61) becomes

$$\phi(q(\cdot), \mathbf{u}(\cdot)) = \int_{t_0}^{t_f} \left[l_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)) + x_{N_x+1}(t) \right] \mathrm{d}t$$

with the extended dynamical system (3.65).

Thus, it can be said that Bolza-, Lagrange-, and Mayer-type problems are theoretically equivalent. A deeper discussion of this topic is given in Cesari [21].

3.3.7.2 Transformation to Fixed Time Interval

Some problem formulations require a free final time. For this case, the final time t_f becomes then a freedom in the OCP formulation. To transform this into an equivalent problem let us use the following linear time transformation:

$$t(\tau) = t_0 + \tau \cdot (t_f - t_0), \quad \tau \in [0, 1] \tag{3.66}$$

which maps the time interval $[t_0, t_f]$ onto $[0, 1]$. Since t_0 is assumed to be zero, (3.66) simplifies to $t(\tau) = \tau t_f$. Substitution of t with $t(\tau)$ gives the continuous-valued states, the continuous-valued controls, and the discrete state over the relative time interval $\tau \in [0, 1]$

$$\begin{aligned} \tilde{\mathbf{x}}(\tau) &:= \mathbf{x}(t(\tau)) = \mathbf{x}(\tau t_f) \\ \tilde{\mathbf{u}}(\tau) &:= \mathbf{u}(t(\tau)) = \mathbf{u}(\tau t_f) \\ \tilde{q}(\tau) &:= q(t(\tau)) = q(\tau t_f). \end{aligned}$$

The final time can be treated as an additional but constant state $\tilde{x}_{N_x+1}(\tau) = t_f$ with the time derivative given as

$$\frac{\mathrm{d}\tilde{x}_{N_x+1}}{\mathrm{d}\tau}(\tau) = 0. \tag{3.67}$$

The evolution of the transformed states is now described by

$$\frac{\mathrm{d}\tilde{\mathbf{x}}}{\mathrm{d}\tau}(\tau) = \dot{\mathbf{x}}(t(\tau)) \cdot \frac{\mathrm{d}t}{\mathrm{d}\tau}(\tau) = \mathbf{f}_{\tilde{q}(\tau)}(\tilde{\mathbf{x}}(\tau), \tilde{\mathbf{u}}(\tau)) \cdot \tilde{x}_{N_x+1}(\tau). \tag{3.68}$$

According to (3.67) and (3.68), the enhanced state vector $\hat{\tilde{\mathbf{x}}}(\cdot)$ is then given by

$$\frac{\mathrm{d}\hat{\tilde{\mathbf{x}}}}{\mathrm{d}\tau}(\tau) = \begin{bmatrix} \frac{\mathrm{d}\tilde{x}_1}{\mathrm{d}\tau}(\tau) \\ \vdots \\ \frac{\mathrm{d}\tilde{x}_{N_x}}{\mathrm{d}\tau}(\tau) \\ \hline \frac{\mathrm{d}\tilde{x}_{N_x+1}}{\mathrm{d}\tau}(\tau) \end{bmatrix} = \begin{bmatrix} \mathbf{f}_{\tilde{q}(\tau)}(\tilde{\mathbf{x}}(\tau), \tilde{\mathbf{u}}(\tau)) \cdot \tilde{x}_{N_x+1}(\tau) \\ \hline 0 \end{bmatrix}.$$

One obtains the equivalent transformed SOCP as

$$\phi\left(\tilde{q}^*(\cdot), \tilde{\mathbf{u}}^*(\cdot)\right) = \min_{\tilde{q}(\cdot)\in\tilde{\mathcal{Q}},\, \tilde{\mathbf{u}}(\cdot)\in\tilde{\mathcal{U}}(\tilde{q}(\cdot))} \phi(\tilde{q}(\cdot), \tilde{\mathbf{u}}(\cdot)) \tag{3.69}$$

subject to

$$\frac{\mathrm{d}\hat{\tilde{\mathbf{x}}}}{\mathrm{d}\tau}(\tau) = \begin{bmatrix} \mathbf{f}_{\tilde{q}(\tau)}(\tilde{\mathbf{x}}(\tau), \tilde{\mathbf{u}}(\tau)) \cdot \tilde{x}_{N_x+1}(\tau) \\ \hline 0 \end{bmatrix}, \qquad \text{for a.e. } \tau \in [0,1] \tag{3.70}$$

$$\hat{\tilde{\mathbf{x}}}_{[1:N_x]}(0) = \mathbf{x}_0 \tag{3.71}$$

$$\hat{\tilde{\mathbf{x}}}_{[\mathcal{I}_f]}(1) = \mathbf{x}_f \tag{3.72}$$

$$\hat{\tilde{\mathbf{x}}}(\cdot) \in \hat{\tilde{\mathcal{X}}}(\tilde{q}(\cdot))$$

where the admissible sets are defined as

$$\tilde{\mathbf{u}}(\cdot) \in \tilde{\mathcal{U}}(\tilde{q}(\cdot)) := \left\{\tilde{\mathbf{u}}(\cdot) \in L^\infty([0,1], \mathsf{U}) \;\middle|\; \mathbf{c}_{\tilde{u},\tilde{q}}(\tilde{\mathbf{u}}(\tau)) \leq \mathbf{0},\ \forall \tau \in [0,1]\right\},$$

$$\tilde{q}(\cdot) \in \tilde{\mathcal{Q}} := \left\{\tilde{q}(\cdot) \in L^\infty([0,1], \hat{\mathcal{Q}}) \;\middle|\; \tilde{q}(\tau) \in \hat{\mathcal{Q}},\ \forall \tau \in [0,1]\right\},$$

and

$$\hat{\tilde{\mathbf{x}}}(\cdot) \in \hat{\tilde{\mathcal{X}}}(\tilde{q}(\cdot)) := \left\{\hat{\tilde{\mathbf{x}}}(\cdot) \in \mathcal{AC}^\infty([0,1], \mathsf{X}) \;\middle|\; \mathbf{c}_{\hat{\tilde{x}},\tilde{q}}\left(\hat{\tilde{\mathbf{x}}}(\tau)\right) \leq \mathbf{0},\ \forall \tau \in [0,1]\right\}.$$

The equivalent problem (3.69)–(3.72) can be extend for a free initial time t_0 in the same manner as shown by Gerdts [30].

3.3.7.3 Transformation into Hybrid Execution Phases with Fixed Time Intervals

The principle of hybrid execution of a hybrid system promotes the idea to virtually breaking the hybrid optimal control problem into multiple phases depending on the current active subsystem. Then, the breaks occur such that the discrete state $q(\cdot)$ takes on constant values. The smaller subintervals of the hybrid trajectory are defined on the hybrid time trajectory $\mathcal{T}$ as

$$\begin{aligned}\mathbf{x}_j &: [t_j, t_{j+1}] \to \mathsf{X} \\ \mathbf{u}_j &: [t_j, t_{j+1}] \to \mathsf{U} \\ q_j &: [t_j, t_{j+1}] \to \hat{\mathcal{Q}}.\end{aligned}$$

This concept modifies the Bolza-type objective functional to multiple stages

$$m(\mathbf{x}_{N_{exe}}(t_f)) + \sum_{j=0}^{N_{exe}-1} \int_{t_j}^{t_{j+1}} l_{q_j(t)}(\mathbf{x}_j(t), \mathbf{u}_j(t))\, \mathrm{d}t. \tag{3.73}$$

Using linear time transformations

$$t_j(\tau) = t_j + \tau \cdot (t_{j+1} - t_j), \quad \tau \in [0, 1], \quad j = 0, \ldots, N_{exe} - 1 \tag{3.74}$$

that map each interval $[t_j, t_{j+1}]$ onto $[0, 1]$, let us transform problems (3.73) into equivalent problem formulations. The derivation of (3.74) yields

$$\frac{\mathrm{d}t_j}{\mathrm{d}\tau}(\tau) = (t_{j+1} - t_j) = \varsigma_j, \quad j = 0, \ldots, N_{exe} - 1,$$

where ς_j can be interpreted as the stage-length of stage j. Substitution of t with $t_j(\tau)$ yields the new continuous-valued states, the continuous-valued controls, and the discrete state over the relative time interval $\tau \in [0, 1]$

$$\begin{aligned}\tilde{\mathbf{x}}_j(\tau) &:= \mathbf{x}_j(t_j(\tau)) \\ \tilde{\mathbf{u}}_j(\tau) &:= \mathbf{u}_j(t_j(\tau)) \\ \tilde{q}_j(\tau) &:= q_j(t_j(\tau))\end{aligned}$$

and the transformed jump conditions

$$\tilde{\mathbf{x}}_j(0) = \tilde{\mathbf{x}}_{j-1}(1) + \boldsymbol{\delta}_{(q_{j-1}, q_j)}(\tilde{\mathbf{x}}_{q_{j-1}}(1)), \quad j = 1, \ldots, N_{exe} - 1.$$

With the usage of the continuous variable $t_j(\tau)$, the evolution of the transformed states can now be described by

$$\begin{aligned}\frac{\mathrm{d}\tilde{\mathbf{x}}_j}{\mathrm{d}\tau}(\tau) &= \frac{\mathrm{d}\mathbf{x}_j}{\mathrm{d}t_j}(t_j(\tau)) \cdot \frac{\mathrm{d}t_j}{\mathrm{d}\tau}(\tau) = \dot{\mathbf{x}}_j(t_j(\tau)) \cdot \frac{\mathrm{d}t_j}{\mathrm{d}\tau}(\tau) \\ &= \varsigma_j \mathbf{f}_{\tilde{q}_j(\tau)}(\tilde{\mathbf{x}}_j(\tau), \tilde{\mathbf{u}}_j(\tau)), \quad \text{for a.e. } \tau \in [0,1]\end{aligned}$$

for each stage $j = 0, \ldots, N_{exe} - 1$.

The transformed problem is then given as follows:

$$\min_{\tilde{\mathbf{u}}_j(\cdot) \in \tilde{\mathcal{U}}(\tilde{q}_j(\cdot)),\ \tilde{q}_j(\cdot) \in \tilde{\mathcal{Q}},\ \varsigma \in \mathbb{R}^{N_\varsigma},\ N_\varsigma \in \mathbb{N}_{\geq 0}} m(\tilde{\mathbf{x}}_{N_\varsigma}(1)) + \sum_{j=0}^{N_\varsigma} \int_0^1 \varsigma_j \cdot l_{\tilde{q}_j(\tau)}(\tilde{\mathbf{x}}_j(\tau), \tilde{\mathbf{u}}_j(\tau))\, \mathrm{d}\tau \tag{3.75}$$

subject to

$$\frac{\mathrm{d}\tilde{\mathbf{x}}_j}{\mathrm{d}\tau}(\tau) = \varsigma_j \mathbf{f}_{\tilde{q}_j(\tau)}(\tilde{\mathbf{x}}_j(\tau), \tilde{\mathbf{u}}_j(\tau)), \qquad j = 0, \ldots, N_\varsigma, \quad \text{for a.e. } \tau \in [0,1] \tag{3.76}$$

$$\tilde{\mathbf{x}}_0(0) = \mathbf{x}_0 \tag{3.77}$$

$$\tilde{\mathbf{x}}_{N_\varsigma}^{[\mathcal{I}_f]}(1) = \mathbf{x}_f \tag{3.78}$$

$$\left(\sum_{i=0}^{N_\varsigma} \varsigma_j\right) = (t_f - t_0) \tag{3.79}$$

$$-\varsigma_j \leq 0, \qquad j = 0, \ldots, N_\varsigma \tag{3.80}$$

$$\tilde{\mathbf{x}}_j(0) = \tilde{\mathbf{x}}_{j-1}(1) + \boldsymbol{\delta}_{(q_{j-1}, q_j)}(\tilde{\mathbf{x}}_{j-1}(1)), \qquad j = 1, \ldots, N_\varsigma \tag{3.81}$$

$$\tilde{\mathbf{x}}_j(\cdot) \in \tilde{\mathcal{X}}(\tilde{q}_j(\cdot)), \qquad j = 0, \ldots, N_\varsigma.$$

The problem formulation (3.75)–(3.81) can be interpreted as a multistage optimization problem with the stage number $N_\varsigma = N_{exe} - 1$ and the vector $\boldsymbol{\varsigma}$ of stage-lengths and is equivalent to the problem (3.51)–(3.55). An important special case arises if the number of stages N_ς and the mode sequence $\mathcal{D} = (\tilde{q}_0, \tilde{q}_1, \tilde{q}_2 \ldots, \tilde{q}_{N_{exe}})$ is a priori known. Then, the switching times remain as freedoms of the discrete decisions and the problem is then called *switching time optimization*.

3.3.7.4 Transformation of Switchings Costs into Reset Functions

In many technical scenarios, a switching requires a certain amount of energy and thus frequent switching should be avoided. To account for this, switching cost functions of the form $\varrho_{(q_{j-1}, q_j)} : \mathsf{X} \to \mathbb{R}$ for $q_{j-1}, q_j \in \hat{Q}$ and $q_{j-1} \neq q_j$ can be added to the cost functional

$$\phi(\mathbf{u}_j(\cdot), q_j(\cdot)) = m(\mathbf{x}_{N_{exe}}(t_f)) + \sum_{j=1}^{N_{exe}-1} \varrho_{(q_{j-1},q_j)}(\mathbf{x}_{j-1}(t_j^-)) + \sum_{j=0}^{N_{exe}-1} \int_{t_j}^{t_{j+1}} l_{q_j}(\mathbf{x}_j(t), \mathbf{u}_j(t))\, \mathrm{d}t. \tag{3.82}$$

The switching cost functions are assumed to be at least once continuously differentiable with respect to t.

The switching costs in (3.82) can be transformed into an additional state $\tilde{x}_j(\cdot)$ with the Bolza to Mayer transformation rule given in Sect. 3.3.7.1. The final value of the new state $\tilde{x}_{N_{exe}}(t_f)$ is then added to the Mayer term. The time derivative and the initial value of $\tilde{x}_j(\cdot)$ are given as

$$\dot{\tilde{x}}_j(t) = 0$$
$$\tilde{x}_j(t_0) = 0.$$

Then, the system evolves over time according to

$$\dot{\breve{\mathbf{x}}}_j(t) = \begin{bmatrix} \dot{\mathbf{x}}_j^{[1]}(t) \\ \vdots \\ \dot{\mathbf{x}}_j^{[N_x]}(t) \\ \hline \dot{\tilde{x}}_j(t) \end{bmatrix} = \begin{bmatrix} \mathbf{f}_{q_j}(\mathbf{x}_j(t), \mathbf{u}_j(t)) \\ \hline 0 \end{bmatrix} \tag{3.83}$$

with the reset condition

$$\begin{aligned} \breve{\mathbf{x}}_j(t_j^+) &= \breve{\mathbf{x}}_{j-1}(t_j^-) + \hat{\boldsymbol{\delta}}_{(q_{j-1},q_j)}\left(\mathbf{x}_j(t_j^-)\right) \\ &= \breve{\mathbf{x}}_{j-1}(t_j^-) + \begin{bmatrix} \mathbf{0}_{N_x \times 1} \\ \varrho_{(q_{j-1},q_j)}(\mathbf{x}_{j-1}(t_j^-)) \end{bmatrix}, \quad j = 1, \ldots, N_{swt} \end{aligned}$$

where $\breve{\mathbf{x}}_j(t)$ is the extended state vector. Transformation of the switching costs into reset functions yields the standard form of a SOCP according to Definition 3.23 as

$$\phi\left(q^*(\cdot), \mathbf{u}^*(\cdot)\right) =$$
$$\min_{q_j(\cdot)\in\mathcal{Q},\, \mathbf{u}_j(\cdot)\in\mathcal{U}(q_j(\cdot)),\, N_{swt}\in\mathbb{N}_{\geq 0}} m(\mathbf{x}_{N_{exe}}(t_f)) + \tilde{x}_{N_{exe}}(t_f) + \sum_{j=0}^{N_{swt}} \int_{t_j}^{t_{j+1}} l_{q_j}(\mathbf{x}_j(t), \mathbf{u}_j(t))\, \mathrm{d}t$$

subject to

$$
\begin{aligned}
\dot{\breve{\mathbf{x}}}_j(t) &= \left[\frac{\mathbf{f}_{q_j}(\mathbf{x}_j(t), \mathbf{u}_j(t))}{0}\right], \qquad \text{for a.e. } t \in [t_0, t_f] \\
\breve{\mathbf{x}}_0(t_0) &= [\mathbf{x}_0, 0]^T \\
\breve{\mathbf{x}}_{N_{exe}}^{[\mathcal{I}_f]}(t_f) &= \mathbf{x}_f \\
\breve{\mathbf{x}}_j(t_j^+) &= \breve{\mathbf{x}}_{j-1}(t_j^-) + \begin{bmatrix} \mathbf{0}_{N_x \times 1} \\ \varrho_{(q_{j-1},q_j)}(\mathbf{x}_{j-1}(t_j^-)) \end{bmatrix}, \quad j = 1, \ldots, N_{swt} \\
\mathbf{x}_j(\cdot) &\in \mathcal{X}(q_j(\cdot)).
\end{aligned}
$$

For SOCPs with state jumps, the reset condition is extended as follows:

$$
\begin{aligned}
\breve{\mathbf{x}}_j(t_j^+) &= \breve{\mathbf{x}}_{j-1}(t_j^-) + \hat{\boldsymbol{\delta}}_{(q_{j-1},q_j)}(\mathbf{x}_{j-1}(t_j^-)) \\
&= \breve{\mathbf{x}}_{j-1}(t_j^-) + \begin{bmatrix} \boldsymbol{\delta}_{(q_{j-1},q_j)}(\mathbf{x}_{j-1}(t_j^-)) \\ \varrho_{(q_{j-1},q_j)}(\mathbf{x}_{j-1}(t_j^-)) \end{bmatrix} \quad j = 1, \ldots, N_{swt}.
\end{aligned}
$$

Certainly, it makes sometimes sense to apply the opposite transformation, which yields switching cost terms from a reset function.

3.4 Bibliographical Notes

The area of applications for hybrid systems is vast and multifaceted. A majority of examples come from the field of robotics (Ding et al. [25] and Egerstedt [28]), process engineering (Avraam et al. [5]), automotive control (Balluchi et al. [6–11], Liu et al. [40], Vasak et al. [60], and Kirches et al. [36]), power systems, and traffic control systems (De Schutter and De Moor [24]) gives the impression that hybrid systems are everywhere. And indeed, many processes encountered in engineering, science, and, daily life can be effectively modeled as hybrid systems.

Many textbooks on hybrid systems have been published recently. Some examples are the books Van Der Schaft et al. [58], Lunze and Lamnabhi-Lagarrigue [41], Cassandras and Lafortune [19], and Sun and Ge [56].

The work of Branicky et al. [15–17] introduced a general hybrid dynamical system description. This work is based on the general abstract definition of a dynamical system developed by Sontag [54]. A further treatment in hybrid modeling is given in Labinaz et al. [38].

A special class of hybrid systems which capture both characteristics, time-driven and event-driven dynamics, is considered for modeling queuing structures that can be found in many manufacturing systems (Cassandras et al. [20]). An interesting fact of this class is the non-differentiabilities in the event-driven state dynamics. This problem leads to a nonsmooth optimization problem which limits the applicability of classical nonlinear programming methodologies.

Purely discrete-event-driven systems have also attracted much attention in the recent literature: Notably, Cassandras and Lafortune [19], Eqtami et al. [29], Donkers [27], and Varutti [59]. They have some similarities to hybrid systems; however, the focus of discrete-event systems is the occurrence of only asynchronously generated discrete events that force state transitions instantaneously.

Hybrid systems with set-valued dynamical systems (known as differential inclusion form) are considered to account for time-varying and perturbations effects (Goebel et al. [31]) and to describe nonsmooth systems (Acary and Brogliato [1]).

Xu and Antsaklis [65] defined switched systems without state reset functions. In contrast, controlled state jumps, where the height of the jump is a control variable, are reported by Attia [4].

A practically important branch is switched (linear) systems, e.g., piecewise affine systems (Johansson [34], Rantzer and Johansson [47]), linear complementarity systems (Van Der Schaft et al. [58], Lunze and Lamnabhi-Lagarrigue [41]), extended linear complementarity systems, and mixed logical dynamical systems (Bemporad and Morari [13], Morari et al. [43]).

Complementary systems are dynamic systems with discrete modes determined by pairs of signals which are complementary to each other. Two signals $\mathbf{a} \geq \mathbf{0}$ and $\mathbf{b} \geq \mathbf{0}$ are said to be subject to a complementary condition if the following holds:

$$\mathbf{0} \leq \mathbf{a} \perp \mathbf{b} \geq \mathbf{0}. \tag{3.84}$$

The notation $\perp$ represents complementarity and means that either a component of signal $a_k = 0$ or $b_k = 0$ is satisfied. Formulation (3.84) can be re-expressed as

$$\mathbf{a}^T \mathbf{b} = 0, \quad \mathbf{a} \geq \mathbf{0}, \ \mathbf{b} \geq \mathbf{0}.$$

Mixed logical dynamical systems use reformulations of logical decisions. The logic decision part, typically represented by Boolean values, is embedded into affine state equations by simply transforming the Boolean decision variables into {0, 1} integers and translating logic relations into mixed-integer linear inequalities as shown by Morari et al. [43]. Heemels et al. [32] have shown that under assumptions related to well-posedness and boundedness of input, state and output that piecewise affine, linear complementary, extended linear complementarity, and mixed logical dynamical hybrid systems are equivalent.

The hybrid phenomena not encountered in either purely time-driven continuous nor discrete-event-driven systems resulted in new challenges for proving system stability. This problem class opened up extensive research activities. Consequently, the classical stability theory has been unified and extended to provide necessary and sufficient Lyapunov conditions for asymptotic stability in hybrid systems (Xu [62], Lygeros et al. [42], and Goebel et al. [31]). The survey paper from Lin and Antsaklis [39] presents recent stability and stabilization results for switched linear systems. Existence and uniqueness of hybrid automata solutions is given in Lygeros et al. [42].

The main focus of hybrid systems in the literature is on optimal control. There are many papers around dealing with this topic; however, the majority of the excellent textbooks discuss only the continuous optimal control counterpart (Kirk [37], Bryson and Ho [18], and Sethi and Thompson [52]).

Optimal control problems governed by partial differential equations or differential-algebraic equations are also a wide field in research and is discussed in Gerdts [30].

A survey paper on optimal control of hybrid and switched systems is presented by Zhu and Antsaklis [67]. Early works on optimal control of hybrid systems were already performed by Witsenhausen [61] and Seidman [51] but especially over the last two decades theoretical and computational achievements have been obtained. The research efforts in the area of theoretical analysis of hybrid optimal control problems include the derivation of necessary conditions for optimal control (Seidman [51], Sussmann [57], Piccoli [46], Shaikh [53], Riedinger et al. [48, 49], Dmitruk and Kaganovich [26], and Passenberg et al. [45]). The computational achievements take advantage of various optimization techniques and high-performance computers to develop efficient numerical methods, for e.g., switched linear systems (Xu and Antsaklis [65], Xu and Antsaklis [66]); switched systems (Bengea et al. [14]); and switched linear system with linear performance index (Baotic et al. [12]).

References

1. Acary V, Brogliato B (2008) Numerical methods for nonsmooth dynamical systems. Applications in mechanics and electronics. Lecture notes in applied and computational mechanics, vol 35. Springer, Berlin
2. Adams RA, Fournier JJ (2003) Sobolev spaces, vol 140, 2nd edn. Academic Press
3. Alamir M, Attia S (2004) On solving optimal control problems for switched hybrid nonlinear systems by strong variations algorithms. In: 6th IFAC symposium on nonlinear control systems (NOLCOS), Stuttgart, Germany, pp 558–563
4. Attia SA, Azhmyakov VZ Jr (2007) State jump optimization for a class of hybrid autonomous systems. In: Proceedings of the 16th IEEE international conference on control applications, pp 1408–1413
5. Avraam M, Shah N, Pantelides C (1998) Modelling and optimisation of general hybrid systems in the continuous time domain. Comput Chem Eng 22:S221–S228
6. Balluchi A, Di Benedetto M, Pinello C, Rossi C, Sangiovanni-Vincentelli A (1997) Cut-off in engine control: a hybrid system approach. In: Proceedings of the 36th IEEE conference on decision and control, 1997, vol 5. IEEE, pp 4720–4725
7. Balluchi A, Bicchi A, Caterini C, Rossi C, Sangiovanni-Vincentelli AL (2000) Hybrid tracking control for spark-ignition engines. In: Proceedings of the 39th IEEE conference on decision and control, 2000, vol 4. IEEE, pp 3126–3131
8. Balluchi A, Benvenuti L, Lemma C, Murrieri P, Sangiovanni-Vincentelli AL (2004a) Hybrid models of an automotive driveline. Technical report, PARADES, Rome, I
9. Balluchi A, Di Natale F, Sangiovanni-Vincentelli A, Van Schuppen JH (2004b) Synthesis for idle speed control of an automotive engine. In: Hybrid systems: computation and control. Springer, pp 80–94
10. Balluchi A, Zoncu M, Villa T, Sangiovanni-Vincentelli AL (2004c) A nonlinear hybrid model of a 4-cylinder engine for idle speed control. Test case for the computation and control European project

11. Balluchi A, Benvenuti L, Ferrari A, Sangiovanni-Vincentelli A (2006) Hybrid systems in automotive electronics design. Int J Control 79(05):375–394
12. Baotic M, Christophersen FJ, Morari M (2006) Constrained optimal control of hybrid systems with a linear performance index. IEEE Trans Autom Control 51(12):1903–1919
13. Bemporad A, Morari M (1999) Control of systems integrating logic, dynamics, and constraints. Automatica 35(3):407–427
14. Bengea S, Uthaichana K, Žefran M, DeCarlo RA (2011) The control system handbook optimal control of switching systems via embedding into continuous optimal control problems, 2nd edn. CRC Press
15. Branicky M, Borkar V, Mitter S (1998) A unified framework for hybrid control: model and optimal control theory. IEEE Trans Autom Control 43(1):31–45
16. Branicky MS (1995) Studies in hybrid systems: modeling, analysis, and control. PhD thesis, Massachusetts Institute of Technology
17. Branicky MS, Borkar VS, Mitter SK (1994) A unified framework for hybrid control. Proceedings of the 33rd conference on decision and control (CDC), Lake Buena Vista. IEEE, pp 4228–4234
18. Bryson A, Ho YC (1975) Applied optimal control—optimization, estimation and control. Taylor & Francis Inc., New York
19. Cassandras CG, Lafortune S (2008) Introduction to discrete event systems, vol 2. Springer Science & Business Media
20. Cassandras CG, Pepyne DL, Wardi Y (2001) Optimal control of a class of hybrid systems. IEEE Trans Autom Control 46(3):398–415
21. Cesari L (2012) Optimization theory and applications: problems with ordinary differential equations, vol 17. Springer Science & Business Media
22. Clarke F (2013) Functional analysis, calculus of variations and optimal control, vol 264. Springer Science & Business Media
23. Coddington EA, Levinson N (1955) Theory of ordinary differential equations. Tata McGraw-Hill Education
24. De Schutter B, De Moor B (1999) The extended linear complementarity problem and the modeling and analysis of hybrid systems. In: Hybrid systems V. Springer, pp 70–85
25. Ding J, Gillulay JH, Huang H, Vitus MP, Zhang W, Tomlin CJ (2011) Hybrid systems in robotics. IEEE Robot Autom Mag 18(3):33–43
26. Dmitruk A, Kaganovich A (2008) The hybrid maximum principle is a consequence of Pontryagin maximum principle. Syst Control Lett 57(11):964–970
27. Donkers M (2011) Networked and event-triggered control systems. PhD thesis, Eindhoven: Technische Universiteit Eindhoven
28. Egerstedt M (2000) Behavior based robotics using hybrid automata. In: Hybrid systems: computation and control. Springer, pp 103–116
29. Eqtami A, Dimarogonas DV, Kyriakopoulos KJ (2011) Novel event-triggered strategies for model predictive controllers. In: Proceedings of the 50th IEEE conference on decision and control and european control conference (CDC-ECC), Orlando, Florida, pp 3392–3397. doi:10.1109/CDC.2011.6161348
30. Gerdts M (2012) Optimal control of ordinary differential equations and differential-algebraic equations. de Gruyter, Berlin
31. Goebel R, Sanfelice R, Teel A (2009) Hybrid dynamical systems. IEEE Control Syst 29(2):28–93
32. Heemels WP, De Schutter B, Bemporad A (2001) Equivalence of hybrid dynamical models. Automatica 37(7):1085–1091
33. Ioffe AD, Tihomirov VM (1979) Theory of extremal problems. North-Holland
34. Johansson M (2003) Piecewise linear control systems. Springer
35. Kirches C (2011) Fast numerical methods for mixed-integer nonlinear model-predictive control. Springer
36. Kirches C, Sager S, Bock HG, Schlöder JP (2010) Time-optimal control of automobile test drives with gear shifts. Optim Control Appl Methods 31(2):137–153. doi:10.1002/oca.892

37. Kirk D (1970) Optimal control theory: an introduction. Englewood Cliffs, Prentice-Hall
38. Labinaz G, Bayoumi MM, Rudie K (1997) A survey of modeling and control of hybrid systems. Annu Rev Control 21:79–92
39. Lin H, Antsaklis PJ (2009) Stability and stabilizability of switched linear systems: a survey of recent results. IEEE Trans Autom control 54(2):308–322
40. Liu Z, Fan G, Chen H (2011) Idle speed control of a 4-cylinder automotive engine using hybrid system method. In: Proceedings of 2011 international conference on modelling, identification and control (ICMIC). IEEE, pp 331–336
41. Lunze J, Lamnabhi-Lagarrigue F (2009) Handbook of hybrid systems control: theory, tools, applications. Cambridge University Press, Cambridge
42. Lygeros J, Johansson KH, Simic SN, Zhang J, Sastry SS (2003) Dynamical properties of hybrid automata. IEEE Trans Autom Control 48(1):2–17
43. Morari M, Baotic M, Borrelli F (2003) Hybrid systems modeling and control. Eur J Control 9(2):177–189
44. Passenberg B (2012) Theory and algorithms for indirect methods in optimal control of hybrid systems. PhD thesis, Technischen Universität München
45. Passenberg B, Leibold M, Stursberg O, M Buss PC (2011) The minimum principle for time-varying hybrid systems with state switching and jumps. In: Proceedings of the IEEE conference on decision and control, pp 6723–6729
46. Piccoli B (1999) Necessary conditions for hybrid optimization. In: Proceedings of the 38th IEEE conference on decision and control, Phoenix, vol 1. IEEE, pp 410–415
47. Rantzer A, Johansson M (2000) Piecewise linear quadratic optimal control. IEEE Trans Autom Control 45(4):629–637
48. Riedinger P, Kratz F (2003) An optimal control approach for hybrid systems. Eur J Control 9:449–458
49. Riedinger P, Kratz F, Iung C, Zannes C (1999) Linear quadratic optimization for hybrid systems. In: Proceedings of the 38th IEEE conference on decision and control, pp 3059–3064
50. Schori M (2015) Solution of optimal control problems for switched systems. Algorithms and applications for hybrid vehicles. PhD thesis, Universität Rostock
51. Seidman T (1987) Optimal control for switching systems. In: Proceedings of the 21st annual conference on information science and systems, pp 485–489
52. Sethi S, Thompson GL (2006) optimal control theory: applications to management science and economics. Springer
53. Shaikh MS (2004) Optimal control of hybrid systems: theory and algorithms. PhD thesis, Department of Electrical and Computer Engineering, McGill University, Montreal
54. Sontag ED (1998) Mathematical control theory: deterministic finite dimensional systems, 2nd edn. Springer Science & Business Media
55. Stauner T (2001) Systematic development of hybrid systems. PhD thesis, TU München
56. Sun Z, Ge SS (2005) Switched linear systems: control and design. Springer Science & Business Media
57. Sussmann HJ (1999) A maximum principle for hybrid optimal control problems. In: Proceedings of the 38th IEEE conference on decision and control, Phoenix, vol 1. IEEE, pp 425–430
58. Van Der Schaft AJ, Schumacher JM, van der Schaft AJ, van der Schaft AJ (2000) An introduction to hybrid dynamical systems, vol 251. Lecture notes in control and information science. Springer, London
59. Varutti P (2013) Model predictive control for nonlinear networked control systems. PhD thesis, Otto-von-Guericke-Universität Magdeburg
60. Vasak M, Baotic M, Petrovic I, Peric N (2007) Hybrid theory-based time-optimal control of an electronic throttle. IEEE Trans Ind Electron 54(3):1483–1494
61. Witsenhausen H (1966) A class of hybrid-state continuous-time dynamic systems. IEEE Trans Autom Control 11:161–167
62. Xu X (2001) Analysis and design of switched systems. PhD thesis, University of Notre Dame
63. Xu X, Antsaklis PJ (2000a) A dynamic programming approach for optimal control of switched systems. In: Proceedings of the 39th IEEE conference on decision and control, 2000, vol 2. IEEE, pp 1822–1827

64. Xu X, Antsaklis PJ (2000b) Optimal control of switched systems: new results and open problems. In: Proceedings of the American control conference, 2000, vol 4. IEEE, pp 2683–2687
65. Xu X, Antsaklis PJ (2003) Results and perspectives on computational methods for optimal control of switched systems. In: Hybrid systems: computation and control. Springer, pp 540–555
66. Xu X, Antsaklis PJ (2004) Optimal control of switched systems based on parameterization of the switching instants. IEEE Trans Autom Control 49. doi:10.1109/TAC.2003.821417
67. Zhu F, Antsaklis PJ (2011) Optimal control of switched hybrid systems: a brief survey. Technical report, ISIS-2013-007, ISIS Group at the University of Notre Dame
68. Ziebur A (1968) On the Gronwall-Bellman lemma. J Math Anal Appl 22(1):92–95

Chapter 4
The Minimum Principle and Hamilton–Jacobi–Bellman Equation

4.1 Introduction

A good understanding of optimal control begins with the study of its ancestor, the calculus of variations. We begin by introducing some important concepts to the classical calculus of variations. Despite its powerful formulations we will see that the classical approach of calculus of variations will yield only unsatisfactory answers to modern control problems and opened a wide area of control activities (some historical marks are given in the bibliography in Sect. 4.6). Nevertheless, we use this brief introduction to motivate *Pontryagin's minimum principle* (PMP), which can be seen as a milestone of a great evolution in the calculus of variation. To conform with the problems discovered in this book, we state only conditions for autonomous problems. These are the problems in which the time does not appear explicitly.

4.1.1 The Calculus of Variations

Numerical methods of optimal control rely on conditions that a control trajectory $\mathbf{u}(\cdot)$ and a state trajectory $\mathbf{x}(\cdot)$ must fulfill in order to be a candidate for the optimal solution and hence those conditions are called necessary conditions for optimality. A powerful mathematical theory for obtaining these conditions is the calculus of variations, which is covered in much more detail in the textbooks Gelfand et al. [24], Liberzon [40], Vinter [55], Kirk [37], and Bryson and Ho [10]. In this section, some basic principles of variational analysis that are needed to derive necessary conditions for optimal control will be explained.

We first consider a simple problem, that aims at finding a trajectory (i.e., a function) $\mathbf{x}^*(\cdot)$ that minimizes a functional $\phi(\mathbf{x}(\cdot))$

$$\phi(\mathbf{x}^*(\cdot)) = \min_{\mathbf{x}(\cdot)\in\mathcal{X}} \phi(\mathbf{x}(\cdot)) = \int_{t_0}^{t_f} g(\mathbf{x}(t))\,\mathrm{d}t, \tag{4.1}$$

T.J. Böhme and B. Frank, *Hybrid Systems, Optimal Control and Hybrid Vehicles*,
Advances in Industrial Control, DOI 10.1007/978-3-319-51317-1_4

where $\mathcal{X}$ is a linear function space with a norm, i.e., a Banach space. The functional $\phi(\cdot)$ returns a scalar value for a trajectory $\mathbf{x}(t)$ evolving from t_0 to t_f, i.e., a functional maps a function to a scalar.

We are interested in the relation to neighboring trajectories $\mathbf{x}(t) + \delta\mathbf{x}(t)$, as we want the trajectory $\mathbf{x}(t)$ to have a lower value of $\phi(\cdot)$ than all neighboring trajectories. We can calculate the difference in the functional between $\mathbf{x}(t)$ and an arbitrary neighbor $\mathbf{x}(t) + \delta\mathbf{x}(t)$ by

$$\Delta\phi(\mathbf{x}, \delta\mathbf{x}) = \phi(\mathbf{x} + \delta\mathbf{x}) - \phi(\mathbf{x}) = \int_{t_0}^{t_f} g(\mathbf{x}(t) + \delta\mathbf{x}(t)) - g(\mathbf{x}(t)) \, \mathrm{d}t.$$

Definition 4.1 (*Local Minimum of a Functional*) A functional $\phi(\mathbf{x}^*(\cdot))$ has a local minimum, if there is a neighborhood $||\delta\mathbf{x}|| = ||\mathbf{x} - \mathbf{x}^*|| < \epsilon$ such that for all neighboring functions $\mathbf{x}^*(\cdot) + \delta\mathbf{x}(\cdot) \in \mathcal{X}$,

$$\Delta\phi(\mathbf{x}^*, \delta\mathbf{x}) \geq 0$$

applies, where the symbol $||\cdot||$ denotes a suitable norm (we will go in more detail soon). If this is the case for an arbitrary large ϵ, then $\phi(\mathbf{x}^*(\cdot))$ is a global minimum.

△

To define first-order necessary conditions for a local minimum of a functional in the sense of infinitesimal calculus, in a similar fashion as it is done in the optimization of finite-dimensional functions (cf. Chap. 2), we need to compare solution candidates with each other. For this task, we need to measure the distance of two functions and that means, we need a norm for the functions. We therefore assume that all solution candidates are defined in a linear function space with a defined norm. Unfortunately, there are many possible choices of linear function spaces with different norms. A very common function space, is the space of continuous functions on an interval $[t_0, t_f]$ which is denoted by $\mathcal{C}^0([t_0, t_f], \mathsf{X})$. One possible and also very common norm for a continuous function $\mathbf{x}(\cdot) \in \mathcal{C}^0([t_0, t_f], \mathsf{X})$ is the 0-norm defined by

$$||\mathbf{x}(\cdot)||_0 = \sup_{t \in [t_0, t_f]} |\mathbf{x}(t)|, \tag{4.2}$$

where $|\cdot|$ is the standard norm for the Euclidean space X.

Another common functional space is the space of continuous differentiable functions $\mathcal{C}^1([t_0, t_f], \mathsf{X})$ for which common norms are the 0-norm and the 1-norm defined by

$$||\mathbf{x}(\cdot)||_1 = \sup_{t \in [t_0, t_f]} |\mathbf{x}(t)| + \sup_{t \in [t_0, t_f]} |\dot{\mathbf{x}}(t)|. \tag{4.3}$$

In an obvious way one can define the k-norm for an arbitrary k for function spaces that consists of at least k times differentiable functions, which in turn is denoted by $\mathcal{C}^k([t_0, t_f], \mathsf{X})$.

With the definition of the 0- and the 1-norm in mind, we can precise the definition of local minima of functionals in the calculus of variations. If we look for optimal trajectories in the function space $\mathcal{C}^1([t_0, t_f], \mathsf{X})$, which is usual in the calculus of variations, and use the 0-norm in Definition 4.1, the minimum is called a strong minimum. If we use the 1-norm, it is called a weak minimum. Obviously, a strong minimum is always a weak minimum, but the converse is not true.

It will turn out that we need to seek solutions in the function space $\mathcal{C}^0([t_0, t_f], \mathsf{X})$ or even in the function space of piecewise continuously differentiable functions for optimal solutions of optimal control problems, which we denote by $\hat{\mathcal{C}}^1([t_0, t_f], \mathsf{X})$. We must therefore rely on conditions for strong local minima. Fortunately, the first-order conditions for a strong and weak minimum are the same and the distinction between them are only challenging for the derivation of second-order conditions. In this book, we only introduce the first-order necessary conditions for an extremum and refer the reader to textbooks about the optimal control theory, e.g., Liberzon [40], for a complete introduction of all necessary conditions. We will assume for this task in a first step, that the solution candidates and all functions involved are continuously differentiable w.r.t. all variables.

To state first-order necessary conditions, we need to introduce a suitable derivative, which can be applied to functionals. This derivative is called Gateaux derivative and is a generalization of the well-known directional derivatives for finite-dimensional functions, as considered in Chap. 2, even for infinite-dimensional functions. This is necessary because a functional is infinite-dimensional, which becomes clear, if e.g., the approximations of functions as power series are considered, for which a basis is the set of all basic polynomials $P = x^{N_x}$ for all $N_x = 0, \ldots, \infty$. It makes therefore no sense to calculate a gradient with an infinite number of entries by partial derivatives and try to find a function for which the gradient vanishes by the solution of an infinite-dimensional system of equations.

Definition 4.2 (*Gateaux derivative for Functionals*) Let $\phi(\cdot)$ be a functional, that maps a function $x(\cdot) \in \mathcal{C}^k([t_0, t_f], \mathbb{R}), k \geq 0$ to a scalar. Then, the Gateaux derivative of the functional $\phi(\cdot)$ at $x(\cdot)$ in the direction $\delta x(\cdot) \in \mathcal{C}^k([t_0, t_f], \mathbb{R})$ is defined by

$$\delta\phi(x(\cdot); \delta x(\cdot)) = \lim_{\tau \to 0} \frac{\phi(x(\cdot) + \tau \delta x(\cdot)) - \phi(x(\cdot))}{\tau} = \frac{\mathrm{d}}{\mathrm{d}\tau} \phi(x(\cdot) + \tau \delta x(\cdot))\Big|_{\tau=0}. \tag{4.4}$$

If the limit exists, the functional is said to be Gateaux differentiable at $x(\cdot)$.

△

The Gateaux derivative $\delta\phi(x(\cdot); \delta x(\cdot))$ defines a function from the same function space $\mathcal{C}^k([t_0, t_f], \mathbb{R})$ as function $x(\cdot)$.

By applying the Gateaux derivative to the functional (4.1), we obtain the so-called *first variation* of the functional

$$\delta\phi(\mathbf{x}(\cdot); \delta\mathbf{x}(\cdot)) = \int_{t_0}^{t_f} \frac{\partial g}{\partial \mathbf{x}}(\mathbf{x}(t)) \cdot \delta\mathbf{x}(t)\, \mathrm{d}t,$$

which has to be interpreted component-wise for the vector of functions $\mathbf{x}(\cdot)$. We use this vector notation for the sake of simplicity in the remainder of this book for Gateaux derivatives and integrals, meaning that the operations have to be applied component-wise.

Now, we can define a necessary condition for $\mathbf{x}(\cdot)$ to be a minimum.

Theorem 4.1 (First-order necessary condition for a minimum of a functional) *If $\mathbf{x}^*(\cdot)$ is a minimum of the functional $\phi(\cdot)$, then the first variation of $\phi(\cdot)$ must vanish for all admissible $\delta\mathbf{x}(\cdot)$:*

$$\delta\phi(\mathbf{x}^*(\cdot); \delta\mathbf{x}(\cdot)) = 0.$$

△

Proof A proof of this theorem can be found in Gelfand et al. [24]. □

4.1.2 Deriving First-Order Necessary Conditions for an Extremum of an Optimal Control Problem

In the previous sections, the basic principles of the calculus of variations were explained for a simple functional. The same principles also apply for more complicated optimal control problems. In the following, we will regard an optimal control problem formulated as Mayer problem:

$$\phi\left(\mathbf{u}^*(\cdot)\right) = \min_{\mathbf{u}(\cdot)\in\mathcal{C}^0([t_0,t_f],\mathsf{U})} \phi(\mathbf{u}(\cdot)) = m(\mathbf{x}^*(t_f)) \tag{4.5}$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \quad \forall t \in [t_0, t_f] \tag{4.6}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{4.7}$$

$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f. \tag{4.8}$$

The continuous-valued states $\mathbf{x}(\cdot)$ are assumed to be continuously differentiable and the continuous-valued controls $\mathbf{u}(\cdot)$ are assumed to be continuous. The functions $m(\cdot)$ and $\mathbf{f}(\cdot)$ are assumed to be continuously differentiable with respect to all arguments. The endpoint boundaries $\mathbf{x}_{[\mathcal{I}_f]}(t_f)$ are prescribed for some or all continuous-valued states identified by the index set $\mathcal{I}_f \subseteq \{1, \ldots, N_x\}$. The number of prescribed endpoints is denoted by $N_f = \#\mathcal{I}_f$ and the complement of the index set is defined by $\mathcal{I}_f^c = \{1, \ldots, N_x\} \setminus \mathcal{I}_f$.

We first augment the functional to incorporate the constraints. The differential equation is appended with the vector of continuously differentiable costates $\boldsymbol{\lambda}(\cdot)$

$$\phi(\mathbf{u}(\cdot), \mathbf{x}(\cdot), \dot{\mathbf{x}}(\cdot), \boldsymbol{\lambda}(\cdot)) = m(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} \boldsymbol{\lambda}^T(t) \cdot \left(\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) - \dot{\mathbf{x}}(t)\right) \mathrm{d}t,$$

which is no problem, because the additional term $\int_{t_0}^{t_f} \boldsymbol{\lambda}^T \left(\mathbf{f}(\mathbf{x}, \mathbf{u}) - \dot{\mathbf{x}}\right) \mathrm{d}t$ is zero for all admissible control and state vectors. Next, the state boundary constraints $\mathbf{x}(t_0) - \mathbf{x}_0$ and $\mathbf{x}_{[\mathcal{I}_f]}(t_f) - \mathbf{x}_f$ are attached with the help of additional Lagrange multipliers $\boldsymbol{\mu}_0 \in \mathbb{R}^{N_x}$, $\boldsymbol{\mu}_f \in \mathbb{R}^{N_f}$ which gives

$$\begin{aligned}\phi_1(\mathbf{u}(\cdot), \mathbf{x}(\cdot), \dot{\mathbf{x}}(\cdot), \boldsymbol{\lambda}(\cdot), \boldsymbol{\mu}_0, \boldsymbol{\mu}_f) = m(\mathbf{x}(t_f)) + \boldsymbol{\mu}_0^T \cdot [\mathbf{x}(t_0) - \mathbf{x}_0] + \boldsymbol{\mu}_f^T \cdot \left[\mathbf{x}_{[\mathcal{I}_f]}(t_f) - \mathbf{x}_f\right] \\ + \int_{t_0}^{t_f} \left[\boldsymbol{\lambda}^T(t)\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) - \boldsymbol{\lambda}^T(t)\dot{\mathbf{x}}(t)\right] \mathrm{d}t. \qquad (4.9)\end{aligned}$$

Please note that the vectors of multipliers $\boldsymbol{\mu}_0$ and $\boldsymbol{\mu}_f$ are not time dependent. We call (4.9) the Lagrangian of problem (4.5)–(4.8).

At this point, we have augmented all conditions of the OCP (4.5)–(4.8) to the functional and can now move on to calculate a stationary point of this functional, which means that the derivatives for all arguments must vanish. According to Theorem 4.1 we must calculate the Gateaux derivatives for all arguments, which are functions, and demand the vanishing of them in all directions to derive the desired first-order necessary conditions. For simplification we write the derivatives in vector notation, even though the derivatives have to be calculated component-wise.

The Gateaux derivatives of $\delta\phi_1(\cdot, \delta\boldsymbol{\lambda}(\cdot))$ and $\delta\phi_1(\cdot, \delta\mathbf{u}(\cdot))$ must vanish, i.e.,

$$\delta\phi_1(\mathbf{u}(\cdot), \mathbf{x}(\cdot), \dot{\mathbf{x}}(\cdot), \boldsymbol{\lambda}(\cdot), \boldsymbol{\mu}_0, \boldsymbol{\mu}_f; \delta\boldsymbol{\lambda}(\cdot)) = \int_{t_0}^{t_f} \left[\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) - \dot{\mathbf{x}}(t)\right] \delta\boldsymbol{\lambda}(t) \, \mathrm{d}t = 0 \qquad (4.10)$$

$$\delta\phi_1(\mathbf{u}(\cdot), \mathbf{x}(\cdot), \dot{\mathbf{x}}(\cdot), \boldsymbol{\lambda}(\cdot), \boldsymbol{\mu}_0, \boldsymbol{\mu}_f; \delta\mathbf{u}(\cdot)) = \int_{t_0}^{t_f} \left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right)^T (\mathbf{x}(t), \mathbf{u}(t))\boldsymbol{\lambda}(t) \, \delta\mathbf{u}(t) \, \mathrm{d}t = 0. \qquad (4.11)$$

Before the Gateaux derivative w.r.t. $\mathbf{x}(\cdot)$ can be performed, we have to get rid of $\dot{\mathbf{x}}(\cdot)$. Integration by parts gives

$$\int_{t_0}^{t_f} \boldsymbol{\lambda}^T(t)\dot{\mathbf{x}}(t) \, \mathrm{d}t = \boldsymbol{\lambda}^T(t_f)\mathbf{x}(t_f) - \boldsymbol{\lambda}^T(t_0)\mathbf{x}(t_0) - \int_{t_0}^{t_f} \dot{\boldsymbol{\lambda}}^T(t)\mathbf{x}(t) \, \mathrm{d}t. \qquad (4.12)$$

Putting (4.12) into the Lagrangian (4.9) yields

$$\begin{aligned}\phi_2(\mathbf{u}(\cdot), \mathbf{x}(\cdot), \boldsymbol{\lambda}(\cdot), \boldsymbol{\mu}_0, \boldsymbol{\mu}_f) = m(\mathbf{x}(t_f)) + \boldsymbol{\mu}_0^T \cdot [\mathbf{x}(t_0) - \mathbf{x}_0] + \boldsymbol{\mu}_f^T \cdot \left[\mathbf{x}_{[\mathcal{I}_f]}(t_f) - \mathbf{x}_f\right] \\ - \boldsymbol{\lambda}^T(t_f)\mathbf{x}(t_f) + \boldsymbol{\lambda}^T(t_0)\mathbf{x}(t_0) \\ + \int_{t_0}^{t_f} \left[\boldsymbol{\lambda}^T(t)\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) + \dot{\boldsymbol{\lambda}}^T(t)\mathbf{x}(t)\right] \mathrm{d}t.\end{aligned}$$

Clearly, $\phi_1(\cdot) = \phi_2(\cdot)$ holds, but we use the enumeration to make clear, which variant of the Lagrangian is used for the derivation of the different parts of the necessary conditions. Then, the Gateaux derivative of $\delta\phi_2(\cdot, \delta\mathbf{x}(\cdot))$ must vanish

$$\delta\phi_2(\mathbf{u}(\cdot), \mathbf{x}(\cdot), \boldsymbol{\lambda}(\cdot), \boldsymbol{\mu}_0, \boldsymbol{\mu}_f; \delta\mathbf{x}(\cdot)) = \int_{t_0}^{t_f} \left[\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right)^T (\mathbf{x}(t), \mathbf{u}(t))\boldsymbol{\lambda}(t) + \dot{\boldsymbol{\lambda}}(t) \right] \delta\mathbf{x}(t)\, \mathrm{d}t = 0. \tag{4.13}$$

Additionally, the derivatives w.r.t. the endpoints $\mathbf{x}(t_0)$ and $\mathbf{x}(t_f)$ must vanish too:

$$\frac{\partial \phi_2}{\partial \mathbf{x}(t_0)}(\mathbf{u}(\cdot), \mathbf{x}(\cdot), \boldsymbol{\lambda}(\cdot), \boldsymbol{\mu}_0, \boldsymbol{\mu}_f) = \boldsymbol{\mu}_0 + \boldsymbol{\lambda}(t_0) = 0 \tag{4.14}$$

$$\frac{\partial \phi_2}{\partial \mathbf{x}(t_f)}(\mathbf{u}(\cdot), \mathbf{x}(\cdot), \boldsymbol{\lambda}(\cdot), \boldsymbol{\mu}_0, \boldsymbol{\mu}_f) = \frac{\partial m}{\partial \mathbf{x}(t_f)}(\mathbf{x}(t_f)) + \hat{\boldsymbol{\mu}}_f - \boldsymbol{\lambda}(t_f) = 0, \tag{4.15}$$

where $\hat{\boldsymbol{\mu}}_f \in \mathbb{R}^{N_x}$ is defined as

$$\hat{\boldsymbol{\mu}}_f^{[\mathcal{I}_f]} = \boldsymbol{\mu}_f$$
$$\hat{\boldsymbol{\mu}}_f^{[\mathcal{I}_f^c]} = \mathbf{0}.$$

The variations $\delta x(\cdot)$, $\delta u(\cdot)$, and $\delta\lambda(\cdot)$ of the trajectories $x(\cdot)$, $u(\cdot)$, and $\lambda(\cdot)$ as depicted in Fig. 4.1 are assumed to be continuous and need to vanish at t_0 and t_f.

To evaluate the first variations (4.10), (4.11), and (4.13) we make use of the following lemma:

Lemma 4.1 (Fundamental Lemma of Calculus of Variations) *In order for a continuous function* $\mathbf{h}(\mathbf{x}(\cdot))$ *to fulfill*

$$\int_{t_0}^{t_f} \mathbf{h}^T(\mathbf{x}(t)) \cdot \delta\mathbf{x}(t)\, \mathrm{d}t = 0$$

for arbitrary continuous $\delta\mathbf{x}(\cdot)$ *with* $\delta\mathbf{x}(t_0) = \delta\mathbf{x}(t_f) = \mathbf{0}$*, the function* $\mathbf{h}(\mathbf{x}(\cdot))$ *must vanish for all* $t \in [t_0, t_f]$.

Thus, we obtain the necessary conditions

$$\begin{aligned}
\dot{\mathbf{x}}(t) &= \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), && \forall t \in [t_0, t_f] \\
\dot{\boldsymbol{\lambda}}(t) &= -\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right)^T (\mathbf{x}(t), \mathbf{u}(t)) \cdot \boldsymbol{\lambda}(t), && \forall t \in [t_0, t_f] \\
\mathbf{0} &= \left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}} \right)^T (\mathbf{x}(t), \mathbf{u}(t)) \cdot \boldsymbol{\lambda}(t), && \forall t \in [t_0, t_f] \\
\boldsymbol{\lambda}(t_0) &= -\boldsymbol{\mu}_0 \\
\boldsymbol{\lambda}(t_f) &= \frac{\partial m}{\partial \mathbf{x}(t_f)}(\mathbf{x}(t_f)) + \hat{\boldsymbol{\mu}}_f.
\end{aligned}$$

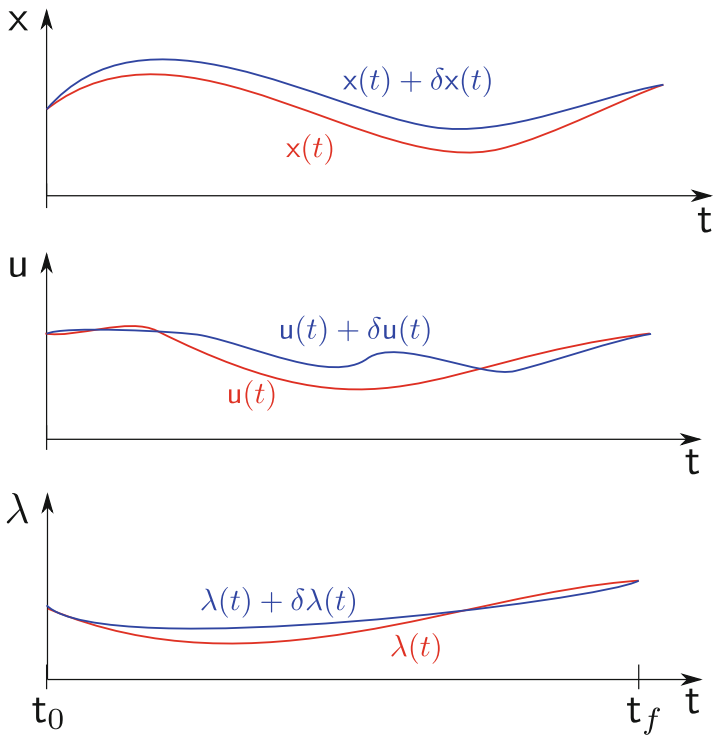


Fig. 4.1 Exemplary variations of the trajectories of a first-order system, i.e., $x(t), u(t), \lambda(t) \in \mathbb{R}$, with fixed final time t_f

The conditions (4.14) and (4.15) are called transversality conditions. Since $\boldsymbol{\mu}_0$ and $\hat{\boldsymbol{\mu}}_f$ can be chosen arbitrarily, we can ignore condition (4.14) and parts of condition (4.15). It remains a transversality condition for continuous-valued states with a free endpoint, which gives further endpoint conditions for the costates

$$\boldsymbol{\lambda}_{[\mathcal{I}_f^c]}(t_f) = \frac{\partial m}{\partial \mathbf{x}_{[\mathcal{I}_f^c]}(t_f)}(\mathbf{x}(t_f)).$$

With the definition of a function known as Hamiltonian

$$\mathcal{H}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)) := \boldsymbol{\lambda}^T(t)\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \tag{4.16}$$

the conditions from above can be written in a more compact form that will allow for a very elegant formulation of necessary conditions and is therefore commonly applied in optimal control theory

$$\dot{\mathbf{x}}(t) = \frac{\partial \mathcal{H}}{\partial \boldsymbol{\lambda}}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)), \qquad \forall t \in [t_0, t_f] \tag{4.17}$$

$$\dot{\boldsymbol{\lambda}}(t) = -\frac{\partial \mathcal{H}}{\partial \mathbf{x}}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)), \qquad \forall t \in [t_0, t_f] \tag{4.18}$$

$$\mathbf{0} = \frac{\partial \mathcal{H}}{\partial \mathbf{u}}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)), \qquad \forall t \in [t_0, t_f] \tag{4.19}$$

$$\begin{aligned} \mathbf{x}(t_0) &= \mathbf{x}_0 \\ \mathbf{x}_{[\mathcal{I}_f]}(t_f) &= \mathbf{x}_f \\ \boldsymbol{\lambda}_{[\mathcal{I}_f^c]}(t_f) &= \frac{\partial m}{\partial \mathbf{x}_{[\mathcal{I}_f^c]}(t_f)}(\mathbf{x}(t_f)). \end{aligned} \tag{4.20}$$

Now, let us collect equations (4.17)–(4.20) and state the first-order necessary conditions for an extremum.

Theorem 4.2 (Necessary Conditions for Continuous Optimal Control Problems) *Let* $\mathbf{u}^*(\cdot) \in \mathcal{C}^0([t_0, t_f], \mathsf{U})$ *and* $\mathbf{x}^*(\cdot) \in \mathcal{C}^1([t_0, t_f], \mathsf{X})$ *be an optimal pair for the OCP (4.5)–(4.8) over the fixed time interval* $[t_0, t_f]$. *The final time* t_f *is specified. Then, there exists a continuously differentiable costate trajectory* $\boldsymbol{\lambda}(\cdot) \in \mathcal{C}^1([t_0, t_f], \mathbb{R}^{N_x})$ *satisfying the nontriviality condition* $\boldsymbol{\lambda}(t) \neq \mathbf{0}$ *and a Hamiltonian be defined as (4.16). Then, the following conditions hold:*

1. *the states* $\mathbf{x}^*(\cdot)$ *and the costates* $\boldsymbol{\lambda}(\cdot)$ *satisfy the canonical equations with respect to the Hamiltonian (4.16)*

$$\dot{\mathbf{x}}^*(t) = \frac{\partial \mathcal{H}}{\partial \boldsymbol{\lambda}}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \mathbf{u}^*(t)) = \mathbf{f}(\mathbf{x}^*(t), \mathbf{u}^*(t)) \tag{4.21}$$

$$\dot{\boldsymbol{\lambda}}(t) = -\frac{\partial \mathcal{H}}{\partial \mathbf{x}}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \mathbf{u}^*(t)) = -\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)^T (\mathbf{x}^*(t), \mathbf{u}^*(t)) \cdot \boldsymbol{\lambda}(t), \tag{4.22}$$

   for all $t \in [t_0, t_f]$ *with the boundary conditions* $\mathbf{x}^*(t_0) = \mathbf{x}_0$ *and* $\mathbf{x}^*_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f$;
2. *the Hamiltonian minimum condition*

$$\mathbf{0} = \frac{\partial \mathcal{H}}{\partial \mathbf{u}}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \mathbf{u}^*(t)) \tag{4.23}$$

   holds for all $t \in [t_0, t_f]$;
3. *at the final time* t_f *the transversality condition*

$$\boldsymbol{\lambda}_{[\mathcal{I}_f^c]}(t_f) = \frac{\partial m}{\partial \mathbf{x}_{[\mathcal{I}_f^c]}(t_f)}(\mathbf{x}^*(t_f)) \tag{4.24}$$

   is fulfilled.

△

The time derivation of the Hamiltonian yields an important property

$$\dot{\mathcal{H}}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)) = \dot{\boldsymbol{\lambda}}^T(t)\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) + \boldsymbol{\lambda}^T(t) \cdot \left[\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right)^T (\mathbf{x}(t), \mathbf{u}(t))\dot{\mathbf{x}}(t) + \left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}} \right)^T (\mathbf{x}(t), \mathbf{u}(t)\dot{\mathbf{u}}(t) \right] = 0.$$

Hence, the Hamiltonian is constant. For the case that the final time t_f is free, the derivative $\frac{\partial \phi_1}{\partial t_f}(\cdot)$ must vanish

$$\frac{\partial m}{\partial \mathbf{x}(t_f)}(\mathbf{x}(t_f))\dot{\mathbf{x}}(t_f) + \boldsymbol{\mu}_f^T \dot{\mathbf{x}}_{[\mathcal{I}_f]}(t_f) = 0.$$

Hence, $\mathcal{H}(t_f) = 0$ applies and the Hamiltonian $\mathcal{H}(t) \equiv 0$ must be identically zero:

We can weaken the smoothness assumptions of Theorem 4.2, if we use a generalization of the fundamental Lemma 4.1.

Lemma 4.2 (General Form of the Fundamental Lemma of Calculus of Variations) *In order for a measurable function* $\mathbf{h}(\mathbf{x}(\cdot)) \in L^1([t_0, t_f])$ *to fulfill*

$$\int_{t_0}^{t_f} \mathbf{h}^T(\mathbf{x}(t)) \cdot \delta\mathbf{x}(t)\, \mathrm{d}t = 0$$

for arbitrary continuous $\delta\mathbf{x}(\cdot)$ *with* $\delta\mathbf{x}(t_0) = \delta\mathbf{x}(t_f) = \mathbf{0}$*, the function* $\mathbf{h}(\mathbf{x}(\cdot))$ *must vanish for a.e.* $t \in [t_0, t_f]$.

By applying this lemma, the continuous-valued states $\mathbf{x}(\cdot)$ and the costates $\boldsymbol{\lambda}(\cdot)$ only need to be absolutely continuous, the continuous-valued controls $\mathbf{u}(\cdot)$ only need to be measurable w.r.t. time, and the conditions of Theorem 4.2 needs to hold "for almost every time" rather than "for every time". A rigorous explanation can be found in Sussmann [54]. The elimination of the continuity assumption for the continuous-valued controls is especially important for the consideration of control affine systems.

4.2 Minimum Principle

So far, we assumed that the control $\mathbf{u}(\cdot)$ can be chosen freely. In most technical applications, there will be restraints on the control variable. For example, the electrical motor/generator of a hybrid vehicle cannot provide an infinite torque but the torque is restricted to a certain range. This additional restriction can be adapted to the optimal control problem formulation as

$$\phi\left(\mathbf{u}^*(\cdot)\right) = \min_{\mathbf{u}(\cdot)\in\mathcal{U}} \phi(\mathbf{u}(\cdot)) = m(\mathbf{x}^*(t_f)) + \int_{t_0}^{t_f} l(\mathbf{x}^*(t), \mathbf{u}^*(t))\,\mathrm{d}t \tag{4.25}$$

subject to

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \qquad \text{for a.e. } t \in [t_0, t_f] \tag{4.26}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{4.27}$$

$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f \tag{4.28}$$

with continuous-valued controls to be constrained to the admissible function space $\mathcal{U} = \left\{\mathbf{u}(\cdot) \in L^\infty\left([t_0, t_f], \mathsf{U}\right) \;\middle|\; \mathbf{c}_u(\mathbf{u}(t)) \leq \mathbf{0},\ \forall t \in [t_0, t_f]\right\}$ where the controls are taken from the set of essentially bounded measurable functions $L^\infty\left([t_0, t_f], \mathsf{U}\right)$. The abbreviation "a.e." stands for "almost every" or "almost everywhere".

The variational approach presented in the last section has led us to the necessary conditions for an extremum expressed by the canonical equations and the Hamiltonian minimization property by vanishing of $\partial\mathcal{H}/\partial\mathbf{u}$. While the relative cumbersome derivative helps us to gain some intuition about the minimum principle, the variational approach has several limitations which prohibits the application to the OCP (4.25)–(4.28). Among them are (cf. Liberzon [40])

- the classical variational approach assumes that $\mathbf{u}^*(\cdot)$ are functions of an unbounded control set. However, if the admissible control set $\hat{\mathcal{U}}(t)$ has a boundary then $\partial\mathcal{H}/\partial\mathbf{u}$ need not to be $\mathbf{0}$ when the minimum is achieved at a boundary point;
- the Hamiltonian minimization property derived by the variational approach is overrestrictive concerning the existence and uniqueness assumptions of the dynamical system since it relies on differentiability of the Hamiltonian with respect to $\mathbf{u}(\cdot)$ and thus implies differentiability of $l(\cdot)$ and $\mathbf{f}(\cdot)$ with respect to $\mathbf{u}(\cdot)$ too. The reader should note that the existence and uniqueness of the dynamical system $\mathbf{f}(\cdot)$ is based on Lipschitz-continuity, and therefore differentiability of $\mathbf{f}(\cdot)$ with respect to $\mathbf{u}(\cdot)$ was not assumed. However, as stated in the introductory Sect. 4.1 for practical solution we require stronger regularity assumptions.

To state the minimum principle the concept of the Hamiltonian is needed. We already employed the Hamiltonian in the previous paragraph but now it is time to introduce this concept in more detail.

Definition 4.3 (*Hamiltonian*) The Hamiltonian $\mathcal{H} : \mathcal{AC}^\infty([t_0, t_f], \mathsf{X}) \times \mathcal{AC}^\infty([t_0, t_f], \mathbb{R}^{N_x}) \times \mathbb{R} \times L^\infty([t_0, t_f], \mathsf{U}) \to \mathbb{R}$ of a continuous optimal control problem (4.25)–(4.28) is given by

$$\mathcal{H}(\mathbf{x}(\cdot), \boldsymbol{\lambda}(\cdot), \lambda_0, \mathbf{u}(\cdot)) := \lambda_0 l(\mathbf{x}(\cdot), \mathbf{u}(\cdot)) + \boldsymbol{\lambda}^T(\cdot)\mathbf{f}(\mathbf{x}(\cdot), \mathbf{u}(\cdot)), \tag{4.29}$$

where $\boldsymbol{\lambda}(\cdot) \in \mathcal{AC}^\infty([t_0, t_f], \mathbb{R}^{N_x})$ are the absolutely continuous costates.

△

Remark 4.1 The Hamiltonian in Definition 4.3 is a functional depending on the functions $\mathbf{x}(\cdot)$, $\boldsymbol{\lambda}(\cdot)$, $\mathbf{u}(\cdot)$, and on the scalar λ_0. For fixed $\mathbf{x}(\cdot) \in \mathcal{AC}^\infty([t_0, t_f], \mathsf{X})$,

$\boldsymbol{\lambda}(\cdot) \in \mathcal{AC}^{\infty}([t_0, t_f], \mathbb{R}^{N_x})$, $\mathbf{u}(\cdot) \in L^{\infty}([t_0, t_f], \mathsf{U})$, and $\lambda_0 \in \mathbb{R}$ the Hamiltonian is a function only depending on the time

$$\mathcal{H}(t) := \lambda_0 l(\mathbf{x}(t), \mathbf{u}(t)) + \boldsymbol{\lambda}^T(t)\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)).$$

We use both interpretations of the Hamiltonian in the following chapters.

The minimum principle in its basic form, originally referred to as maximum principle, is an extension of the classical variational approach and goes back to the early fifties and the works of Hestenes [33], and of course Pontryagin et al. [46]. Whether we speak about a minimum principle or a maximum principle is determined just by the sign convention of Hamiltonian. The minimum principle states the existence of costates $\boldsymbol{\lambda}(\cdot)$ that satisfy adjoint differential equations and transversality conditions. The optimal controls $\mathbf{u}^*(\cdot)$ are characterized as an implicit function of the states $\mathbf{x}^*(\cdot)$ and the costates $\boldsymbol{\lambda}(\cdot)$.

Theorem 4.3 (Pontryagin's Minimum Principle) *Let* $\mathbf{u}^*(\cdot) \in L^{\infty}\left([t_0, t_f], \mathsf{U}\right)$ *be a measurable and essentially bounded strong optimal control function and let* $\mathbf{x}^*(\cdot) \in \mathcal{AC}^{\infty}\left([t_0, t_f], \mathsf{X}\right)$ *be the corresponding absolutely continuous strong optimal state function. Then, there exist absolutely continuous costates* $\boldsymbol{\lambda}(\cdot) \in \mathcal{AC}^{\infty}\left([t_0, t_f], \mathbb{R}^{N_x}\right)$ *and a constant* λ_0 *satisfying the nontrivial solution* $(\lambda_0, \boldsymbol{\lambda}(t)) \neq \mathbf{0}$ *for every* $t \in [t_0, t_f]$*, for which the following conditions hold:*

1. *the states* $\mathbf{x}^*(\cdot)$ *and the costates* $\boldsymbol{\lambda}(\cdot)$ *satisfy the canonical equations with respect to the Hamiltonian (4.29)*

$$\dot{\mathbf{x}}^*(t) = \frac{\partial \mathcal{H}}{\partial \boldsymbol{\lambda}}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}^*(t))$$
$$\dot{\boldsymbol{\lambda}}(t) = -\frac{\partial \mathcal{H}}{\partial \mathbf{x}}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}^*(t))$$

   for almost every $t \in [t_0, t_f]$ *with the boundary conditions* $\mathbf{x}^*(t_0) = \mathbf{x}_0$ *and* $\mathbf{x}^*_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f$*;*
2. *for the Hamiltonian function the inequality*

$$\mathcal{H}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}^*(t)) \leq \mathcal{H}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}(t))$$

   holds for almost every $t \in [t_0, t_f]$ *and all* $\mathbf{u}(t) \in \hat{\mathcal{U}}$*. Then, the minimum condition*

$$\mathbf{u}^*(t) = \arg \min_{\mathbf{u}(t) \in \hat{\mathcal{U}}(t)} \mathcal{H}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}(t)) \tag{4.30}$$

   applies for almost every $t \in [t_0, t_f]$*;*
3. *since the problem is autonomous, the Hamiltonian has the following important properties:*

- *if the final time t_f is fixed, then the Hamiltonian*

$$\mathcal{H}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}^*(t)) = c, \quad \textit{for a.e. } t \in [t_0, t_f] \tag{4.31}$$

  must be constant; and
- *if the final time t_f is free, then the Hamiltonian*

$$\mathcal{H}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}^*(t)) = 0, \quad \textit{for a.e. } t \in [t_0, t_f]$$

  must be identically zero;

and

4. *at the final time t_f the transversality condition*

$$\boldsymbol{\lambda}_{[\mathcal{I}_f^c]}(t_f) = \frac{\partial m}{\partial \mathbf{x}_{[\mathcal{I}_f^c]}(t_f)}(\mathbf{x}^*(t_f)) \tag{4.32}$$

is fulfilled.

△

Proof We omit the rather complex proof for the PMP but interested readers may refer to Pontryagin et al. [46]. A good introduction into the steps required for establishing the minimum principle is given in Liberzon [40]. □

Remark 4.2 The reader should note that the original result from Pontryagin is stated as "maximum principle". This should not confuse the reader. By a sign convention Pontryagin's result can be applied to a maximization problem (or minimization problem).

One difference to the necessary conditions derived using the classical variational approach in Sect. 4.1.2 is that the vanishing of the Gateaux derivative $\partial\mathcal{H}/\partial\mathbf{u}$ is no longer a necessary condition. The reason is, that we restrained the control set. A further difference is the presence of λ_0. This positive scalar is called the *abnormal multiplier*. Similarly to the abnormal multiplier in Sect. 2.3.1 if a constraint qualification holds then λ_0 can be chosen somehow without loss of generality. Then, the earlier Definition (4.16) can be recovered by normalizing $(\lambda_0, \boldsymbol{\lambda}(t))$ such that $\lambda_0 = 1$ applies. The reader should note that such scaling does not affect any of the properties stated in the minimum principle. As convention, whenever the abnormal multiplier is not explicitly written, it is assumed to be equal to 1.

4.2.1 Necessary Conditions for Optimal Control Problems with Control Restraints

The minimum principle from Theorem 4.3 states that the optimal control $\mathbf{u}^*(\cdot)$ satisfies the necessary condition

$$\mathbf{u}^*(t) = \arg \min_{\mathbf{u}(t) \in \hat{\mathcal{U}}(t)} \mathcal{H}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}(t)), \quad \text{for a.e. } t \in [t_0, t_f]. \tag{4.33}$$

This necessary condition is unpractical in terms of evaluation. Indeed, for practical optimal control implementation a more strict statement is required. In so doing, we make use of augmentation of the Hamiltonian. Hestenes [33] used the classical multiplier rules in the calculus of variations for the control restraint case. The OCP (4.25)–(4.28) can be reformulated as the following optimal control problem:

$$\phi\left(\mathbf{u}^*(\cdot)\right) = \min_{\mathbf{u}(\cdot) \in L^\infty\left(([t_0, t_f], \mathsf{U})\right.} \phi(\mathbf{u}(\cdot)) = m(\mathbf{x}^*(t_f)) + \int_{t_0}^{t_f} l(\mathbf{x}^*(t), \mathbf{u}^*(t))\, \mathrm{d}t \tag{4.34}$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \qquad \text{for a.e. } t \in [t_0, t_f] \tag{4.35}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{4.36}$$

$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f \tag{4.37}$$

$$\mathbf{c}_u(\mathbf{u}(t)) \leq \mathbf{0}, \qquad \forall t \in [t_0, t_f]. \tag{4.38}$$

Then, the Hamiltonian function

$$\mathcal{H}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}(t)) := \lambda_0 l(\mathbf{x}(t), \mathbf{u}(t)) + \boldsymbol{\lambda}^T(t)\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))$$

can be augmented to

$$\begin{aligned}\mathcal{H}_a(\mathbf{x}(t), \boldsymbol{\lambda}(t), \lambda_0, \boldsymbol{\gamma}(t), \mathbf{u}(t)) &:= \lambda_0 l(\mathbf{x}(t), \mathbf{u}(t)) + \boldsymbol{\lambda}^T(t)\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) + \boldsymbol{\gamma}^T(t)\mathbf{c}_u(\mathbf{u}(t)) \\ &= \mathcal{H}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}(t)) + \boldsymbol{\gamma}^T(t)\mathbf{c}_u(\mathbf{u}(t)),\end{aligned} \tag{4.39}$$

where $\gamma_i(t) \geq 0$ are time-dependent multipliers with $\gamma_i(t) = 0$ whenever i-th constraint $\mathbf{c}_u^{[i]}(\mathbf{u}(t)) < 0$ is not active. All active constraints are identified with the set of active indices

$$\mathcal{I}_{c_u}(t) := \left\{ i = 1, \ldots, N_{c_u} \;\middle|\; \mathbf{c}_u^{[i]}(\mathbf{u}(t)) = 0 \right\}.$$

We impose the following rank condition:

$$\operatorname{rank} \left[\left(\frac{\partial \mathbf{c}_u^{[i]}}{\partial \mathbf{u}}(\mathbf{u}(t)) \right)_{i \in \mathcal{I}_{c_u}(t)} \right] = \#\mathcal{I}_{c_u}(t), \tag{4.40}$$

for all $\mathbf{u}(\cdot)$ that could arise along an optimal solution. The constraint qualification (4.40) means that the Jacobian $\partial \mathbf{c}_u / \partial \mathbf{u}$ of all active constraints $\mathbf{c}_u^{[i]}(\cdot) = 0,\ i \in \mathcal{I}_{c_u}(t)$ must be linearly independent.

It is a matter of taste to append the control constraints either to the Hamiltonian or to the Lagrangian. Then, the minimum condition also holds for the augmented Hamiltonian with

$$\mathcal{H}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}^*(t)) + \boldsymbol{\gamma}^T(t)\mathbf{c}_u(\mathbf{u}^*(t)) \leq \mathcal{H}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}(t))$$

satisfying (4.38) for a.e. $t \in [t_0, t_f]$. Hestenes proofed the minimum condition using the augmented Hamiltonian only for $\mathcal{C}^k(\cdot)$ functions.

For the case that we admit differentiability of the Hamiltonian with respect to $\mathbf{u}(\cdot)$ allows us to state the minimum condition (4.23) for the augmented Hamiltonian (4.39) as

$$\begin{aligned}&\frac{\partial \mathcal{H}_a}{\partial \mathbf{u}}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \boldsymbol{\gamma}(t), \mathbf{u}^*(t)) \\ &= \frac{\partial l}{\partial \mathbf{u}}(\mathbf{x}^*(t), \mathbf{u}^*(t)) \cdot \lambda_0 + \left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right)^T (\mathbf{x}^*(t), \mathbf{u}^*(t)) \cdot \boldsymbol{\lambda}(t) + \left(\frac{\partial \mathbf{c}_u}{\partial \mathbf{u}}\right)^T (\mathbf{u}^*(t)) \cdot \boldsymbol{\gamma}(t) = \mathbf{0}\end{aligned}$$

for a.e. $t \in [t_0, t_f]$. This minimum condition is more restrictive than (4.33) due to the differentiability assumption of $l(\cdot)$, $\mathbf{f}(\cdot)$, and $\mathbf{c}_u(\cdot)$ w.r.t. $\mathbf{u}(\cdot)$. However, the stronger minimum condition is required for nonlinear programming implementations as a correspondence to the first-order necessary conditions for discretized optimal control problems.

We are now ready to state first-order necessary conditions for restraint optimal control problems

Theorem 4.4 (First-Order Necessary Conditions for Continuous Optimal Control Problems with Control Constraint) *Let* $\mathbf{u}^*(\cdot) \in L^\infty\left([t_0, t_f], \mathsf{U}\right)$ *be a measurable and essentially bounded optimal control function with right-continuous and left-hand limits and* $\mathbf{x}^*(\cdot) \in \mathcal{AC}^\infty\left([t_0, t_f], \mathsf{X}\right)$ *be an absolutely continuous optimal state function. Then,* $(\mathbf{x}^*(\cdot), \mathbf{u}^*(\cdot))$ *is an optimal pair for the OCP (4.34)–(4.38) over the fixed time interval* $[t_0, t_f]$. *The constraint qualification (4.40) holds for every* $\mathbf{u}(t),\ t \in [t_0, t_f]$ *with* $\mathbf{u}(t) \in \hat{\mathcal{U}}$. *Then, there exist a constant* $\lambda_0 \geq 0$, *absolutely continuous costates* $\boldsymbol{\lambda}(\cdot) \in \mathcal{AC}^\infty\left([t_0, t_f], \mathbb{R}^{N_x}\right)$ *and a piecewise continuous multiplier function* $\boldsymbol{\gamma}(\cdot) \in \hat{\mathcal{C}}^0\left([t_0, t_f], \mathbb{R}^{N_{c_u}}\right)$ *satisfying the nontrivial solution* $(\lambda_0, \boldsymbol{\lambda}(t), \boldsymbol{\gamma}(t))) \neq \mathbf{0}$ *for every* $t \in [t_0, t_f]$ *for which the following conditions hold with the Hamiltonian defined by (4.39):*

1. *the states* $\mathbf{x}^*(\cdot)$ *and the costates* $\boldsymbol{\lambda}(\cdot)$ *satisfy the canonical equations with respect to the Hamiltonian (4.29)*

$$\dot{\mathbf{x}}^*(t) = \frac{\partial \mathcal{H}}{\partial \boldsymbol{\lambda}}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}^*(t)) = \mathbf{f}(\mathbf{x}^*(t), \mathbf{u}^*(t)), \tag{4.41}$$

$$\begin{aligned}\dot{\boldsymbol{\lambda}}(t) &= -\frac{\partial \mathcal{H}}{\partial \mathbf{x}}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}^*(t)) \\ &= -\frac{\partial l}{\partial \mathbf{x}}(\mathbf{x}^*(t), \mathbf{u}^*(t)) \cdot \lambda_0 - \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)^T (\mathbf{x}^*(t), \mathbf{u}^*(t)) \cdot \boldsymbol{\lambda}(t)\end{aligned} \tag{4.42}$$

for almost every $t \in [t_0, t_f]$ *with the boundary conditions* $\mathbf{x}^*(t_0) = \mathbf{x}_0$ *and* $\mathbf{x}^*_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f$*;*

2. *the Hamiltonian minimum conditions*

$$\mathbf{u}^*(t) = \arg \min_{\mathbf{u}(t) \in \hat{\mathcal{U}}(t)} \mathcal{H}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}(t)) \tag{4.43}$$

and

$$\begin{aligned} &\frac{\partial \mathcal{H}_a}{\partial \mathbf{u}}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \boldsymbol{\gamma}(t), \mathbf{u}^*(t)) \\ &= \frac{\partial l}{\partial \mathbf{u}}(\mathbf{x}^*(t), \mathbf{u}^*(t)) \cdot \lambda_0 + \left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right)^T (\mathbf{x}^*(t), \mathbf{u}^*(t)) \cdot \boldsymbol{\lambda}(t) + \left(\frac{\partial \mathbf{c}_u}{\partial \mathbf{u}}\right)^T (\mathbf{u}^*(t)) \cdot \boldsymbol{\gamma}(t) \\ &= \mathbf{0} \end{aligned} \tag{4.44}$$

holds for almost every $t \in [t_0, t_f]$*;*

3. $\boldsymbol{\gamma}(t) \in \hat{\mathcal{C}}^0\left([t_0, t_f], \mathbb{R}^{N_{c_u}}\right)$ *are time-dependent multipliers and subject to the complementarity condition*

$$\begin{aligned} &\gamma_i(t) \geq 0, \ \mathbf{c}_u^{[i]}(\cdot) \leq 0, \quad i \in \left\{1, \ldots, N_{c_u}\right\} \\ &\boldsymbol{\gamma}^T(t) \mathbf{c}_u(\mathbf{u}^*(t)) = 0 \end{aligned} \tag{4.45}$$

that holds for almost every $t \in [t_0, t_f]$*; and*

4. *at the final time* t_f *the transversality condition evaluated on the terminal Lagrangian*

$$\boldsymbol{\lambda}_{[\mathcal{I}_f^c]}(t_f) = \frac{\partial m}{\partial \mathbf{x}_{[\mathcal{I}_f^c]}(t_f)}(\mathbf{x}^*(t_f)) \tag{4.46}$$

is fulfilled.

△

Proof An early proof is given in Hestenes [33] for $\mathbf{u}(\cdot) \in \mathcal{C}^0([t_0, t_f], \mathsf{U})$ and $\mathbf{x}(\cdot) \in \mathcal{C}^1([t_0, t_f], \mathsf{X})$. More general proofs can be found in the references of the survey paper of Hartl et al. [30]. □

4.2.2 Necessary Conditions for Optimal Control Problems with State Constraints

State constraints are a natural feature in many practical applications. Necessary conditions of optimality for optimal control problems with state constraints have been

studied since the very beginning of optimal control theory. However, the derivation of these necessary conditions is more difficult than for control constrained problems.

We have to distinguish between purely state constraints of the form

$$\mathbf{c}_x(\mathbf{x}(t)) \leq \mathbf{0}, \qquad \forall t \in [t_0, t_f] \tag{4.47}$$

and mixed control-state constraints of the form

$$\mathbf{c}_{x,u}(\mathbf{x}(t), \mathbf{u}(t)) \leq \mathbf{0}, \qquad \forall t \in [t_0, t_f].$$

An optimal control problem with pure control and state constraints (4.47) can be expressed as

$$\phi\left(\mathbf{u}^*(\cdot)\right) = \min_{\mathbf{u}(\cdot)\in L^\infty([t_0,t_f],\mathsf{U})} \phi(\mathbf{u}(\cdot)) = m(\mathbf{x}^*(t_f)) + \int_{t_0}^{t_f} l(\mathbf{x}^*(t), \mathbf{u}^*(t))\, \mathrm{d}t \tag{4.48}$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \qquad \text{for a.e. } t \in [t_0, t_f] \tag{4.49}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{4.50}$$

$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f \tag{4.51}$$

$$\mathbf{c}_u(\mathbf{u}(t)) \leq \mathbf{0}, \qquad \forall t \in [t_0, t_f] \tag{4.52}$$

$$\mathbf{c}_x(\mathbf{x}(t)) \leq \mathbf{0}, \qquad \forall t \in [t_0, t_f]. \tag{4.53}$$

Next, let us discuss some important features of state constraints. Therefore, we take a closer look to the time instants when the trajectory enters or leaves the boundary of the state constraints. Let us denote $[t_1, t_2) \subset [t_0, t_f]$ as an *interior interval* of a state trajectory if $\mathbf{c}_x(\mathbf{x}(t)) < \mathbf{0},\ t \in [t_1, t_2)$ applies. Similar, let us denote $[t_1, t_2) \subset [t_0, t_f]$ as a *boundary interval* if $\mathbf{c}_x^{[i]}(\mathbf{x}(t)) = 0,\ t \in [t_1, t_2)$ applies for an arbitrary i-th constraint. Then, a time instance t_j is called an *entry time point* if there is an interior subinterval ending at $t = t_j$ and a boundary interval beginning at $t = t_j$. Consequently, t_l is called an *exit time point* if a boundary subinterval ends and an interior subinterval begins at t_l. Figure 4.2 illustrates this concept on one boundary interval.

For the special case, that the entry and exit time point coincide, this time point is called *contact time point*. We call altogether, *junction times*. In order to generalize this concept, let us introduce vectors $\mathbf{t}_{ent} = \left[t_{j_1}, t_{j_2}, \ldots, t_{j_{N_{ent}}}\right]^T$ and $\mathbf{t}_{ex} = \left[t_{l_1}, t_{l_2}, \ldots, t_{l_{N_{ex}}}\right]^T$ of all entry and exit time points, respectively. Then, a time point on the boundary interval is given as $t_b \in [\mathbf{t}_{ent}^{[b]}, \mathbf{t}_{ex}^{[b]}]$, where the number b enumerates the boundary interval.

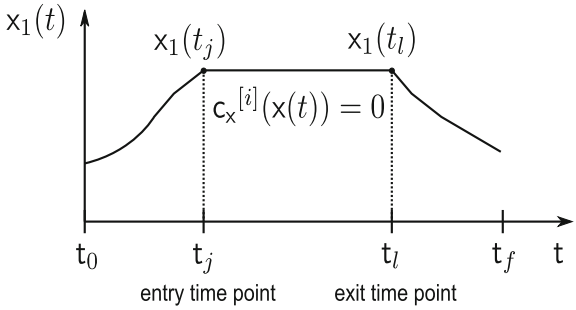


Fig. 4.2 State trajectory $\mathbf{x}(\cdot)$ with one boundary interval $[t_j, t_l)$ on the i-th constraint from $\mathbf{c}_x(\cdot)$. We say, the i-th constraint is active at $[t_j, t_l)$

Let us define all active constraints with the set of indices

$$\mathcal{I}_{c_x}(t) := \left\{ i = 1, \ldots, N_{c_x} \mid \mathbf{c}_x^{[i]}(\mathbf{x}(t)) = 0 \right\}.$$

The total number of constrained state arcs in the time interval $[t_0, t_f]$ is denoted by $N_{c_x}^{arc}$.

There exist different sets of necessary conditions in the literature for the OCP (4.48)–(4.53) depending on the way of adding the constraints to the Hamiltonian and the assumptions for the multipliers. Hartl et al. [30] enumerated three ways to incorporate state constraints

- direct adjoining approach (Maurer [41]);
- indirect adjoining approach with complementary slackness (Bryson and Ho [10]); and
- indirect adjoining approach with continuous costate (Hestenes [33]).

We follow the argumentation of the direct adjoining approach, which is closer to numerical optimization procedures. By this method, the state constraints are directly adjoined to the Hamiltonian resulting in the augmented Hamiltonian

$$\begin{aligned}\mathcal{H}_a(\mathbf{x}(t), \boldsymbol{\lambda}(t), \lambda_0, \boldsymbol{\gamma}(t), \boldsymbol{\rho}(t), \mathbf{u}(t)) := \lambda_0 l(\mathbf{x}(t), \mathbf{u}(t)) + \boldsymbol{\lambda}^T(t)\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \\ + \boldsymbol{\gamma}^T(t)\mathbf{c}_u(\mathbf{u}(t)) + \boldsymbol{\rho}^T(t)\mathbf{c}_x(\mathbf{x}(t)). \quad (4.54)\end{aligned}$$

For the upcoming prerequisite of the theorem, it will be helpful to differentiate the state constraints iteratively w.r.t. time

$$\begin{aligned}\frac{\mathrm{d}\mathbf{c}_x}{\mathrm{d}t}(\mathbf{x}(t)) &= \frac{\partial \mathbf{c}_x}{\partial \mathbf{x}}(\mathbf{x}(t))\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \\ \frac{\mathrm{d}^2\mathbf{c}_x}{\mathrm{d}t^2}(\mathbf{x}(t)) &= \frac{\partial}{\partial \mathbf{x}}\left(\frac{\mathrm{d}\mathbf{c}_x}{\mathrm{d}t}(\mathbf{x}(t))\right)\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \\ &\vdots \\ \frac{\mathrm{d}^p\mathbf{c}_x}{\mathrm{d}t^p}(\mathbf{x}(t)) &= \frac{\partial}{\partial \mathbf{x}}\left(\frac{\mathrm{d}^{p-1}\mathbf{c}_x}{\mathrm{d}t^{p-1}}(\mathbf{x}(t))\right)\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))\end{aligned}$$

until

$$\begin{aligned}
\frac{\partial}{\partial \mathbf{u}}\left(\frac{\mathrm{d}\mathbf{c}_x}{\mathrm{d}t}(\mathbf{x}(t))\right) &= \mathbf{0} \\
\frac{\partial}{\partial \mathbf{u}}\left(\frac{\mathrm{d}^2\mathbf{c}_x}{\mathrm{d}t^2}(\mathbf{x}(t))\right) &= \mathbf{0} \\
&\vdots \\
\frac{\partial}{\partial \mathbf{u}}\left(\frac{\mathrm{d}^p\mathbf{c}_x}{\mathrm{d}t^p}(\mathbf{x}(t))\right) &\neq \mathbf{0}
\end{aligned} \tag{4.55}$$

holds. Then, we impose for the following theorem that the rank condition

$$\operatorname{rank}\left[\frac{\partial}{\partial \mathbf{u}}\left\{\left(\frac{\mathrm{d}^p\mathbf{c}_x}{\mathrm{d}t^p}(\mathbf{x}(t))\right)_{i\in\mathcal{I}_{c_x}(t)}\right\}\right] = \#\mathcal{I}_{c_x}(t), \quad \forall i \in \mathcal{I}_{c_x}(t) \text{ and } t \in \left[\mathbf{t}_{ent}^{[b]}, \mathbf{t}_{ex}^{[b]}\right] \text{ a.e.} \tag{4.56}$$

holds along an optimal solution. We are now ready to state a formulation of the minimum principle with state constraints that is often used for an applied numerical stetting.

Theorem 4.5 (First-Order Necessary Conditions for Continuous Optimal Control Problems with State Inequality Constraints (Hartl et al. [30])) *Let* $\mathbf{u}^*(\cdot) \in L^\infty\left([t_0, t_f], \mathsf{U}\right)$ *be a measurable, essentially bounded and right-continuous optimal control function with left-hand limits and* $\mathbf{x}^*(\cdot) \in \mathcal{AC}^\infty\left([t_0, t_f], \mathsf{X}\right)$ *be an absolutely continuous optimal state function. Then,* $(\mathbf{x}^*(\cdot), \mathbf{u}^*(\cdot))$ *is an optimal pair for the OCP (4.48)–(4.53) over the fixed time interval* $[t_0, t_f]$*. The constraint qualification (4.40) holds for every* $\mathbf{u}(t) \in \hat{\mathcal{U}}(t),\ t \in [t_0, t_f]$ *and the constraint qualification (4.56) holds for every* $\mathbf{x}(t) \in \hat{\mathcal{X}}(t),\ t \in [t_j, t_l]$*. Assume that* $\mathbf{x}^*(\cdot)$ *has only finitely many junction times. Then, there exist constant* $\lambda_0 \geq 0$*, piecewise absolutely continuous costates* $\boldsymbol{\lambda}(\cdot) \in \hat{\mathcal{AC}}^\infty\left([t_0, t_f], \mathbb{R}^{N_x}\right)$*, piecewise continuous multiplier functions* $\boldsymbol{\gamma}(\cdot) \in \hat{\mathcal{C}}^0\left([t_0, t_f], \mathbb{R}^{N_{cu}}\right)$ *and* $\boldsymbol{\rho}(\cdot) \in \hat{\mathcal{C}}^0\left([t_0, t_f], \mathbb{R}^{N_{cx}}\right)$ *and a vector of multipliers* $\boldsymbol{\pi}_b$ *for each point* $t_b \in [\mathbf{t}_{ent}^{[b]}, \mathbf{t}_{ex}^{[b]}]$ *of discontinuity of* $\boldsymbol{\lambda}(\cdot)$ *satisfying the nontrivial solution* $\left(\lambda_0, \boldsymbol{\lambda}(t), \boldsymbol{\gamma}(t), \boldsymbol{\rho}(t), \boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots, \boldsymbol{\pi}_{N_{cx}^{arc}}\right) \neq \mathbf{0}$ *for every* $t \in [t_0, t_f]$ *for which the following conditions hold with the definition of the Hamiltonian (4.54):*

1. *the states* $\mathbf{x}^*(\cdot)$ *and the costates* $\boldsymbol{\lambda}(\cdot)$ *satisfy the canonical equations with respect to the Hamiltonian (4.54)*

$$\begin{aligned}
\dot{\mathbf{x}}^*(t) &= \frac{\partial \mathcal{H}_a}{\partial \boldsymbol{\lambda}}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \boldsymbol{\gamma}(t), \boldsymbol{\rho}(t), \mathbf{u}^*(t)) = \mathbf{f}(\mathbf{x}^*(t), \mathbf{u}^*(t)), \\
\dot{\boldsymbol{\lambda}}(t) &= -\frac{\partial \mathcal{H}_a}{\partial \mathbf{x}}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \boldsymbol{\gamma}(t), \boldsymbol{\rho}(t), \mathbf{u}^*(t))
\end{aligned} \tag{4.57}$$

$$= -\frac{\partial l}{\partial \mathbf{x}}(\mathbf{x}^*(t), \mathbf{u}^*(t)) \cdot \lambda_0 - \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)^T (\mathbf{x}^*(t), \mathbf{u}^*(t)) \cdot \boldsymbol{\lambda}(t) \qquad (4.58)$$
$$- \left(\frac{\partial \mathbf{c}_x}{\partial \mathbf{x}}\right)^T (\mathbf{x}^*(t)) \cdot \boldsymbol{\rho}(t)$$

for almost every $t \in [t_0, t_f]$ *and with the boundary conditions* $\mathbf{x}^*(t_0) = \mathbf{x}_0$ *and* $\mathbf{x}^*_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f$*;*

2. *the Hamiltonian minimum conditions*

$$\mathbf{u}^*(t) = \arg \min_{\mathbf{u}(t) \in \hat{\mathcal{U}}(t)} \mathcal{H}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}(t)) \qquad (4.59)$$

and

$$\begin{aligned} &\frac{\partial \mathcal{H}_a}{\partial \mathbf{u}}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \boldsymbol{\gamma}(t), \boldsymbol{\rho}(t), \mathbf{u}^*(t)) \\ &= \frac{\partial l}{\partial \mathbf{u}}(\mathbf{x}^*(t), \mathbf{u}^*(t)) \cdot \lambda_0 + \left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right)^T (\mathbf{x}^*(t), \mathbf{u}^*(t)) \cdot \boldsymbol{\lambda}(t) + \left(\frac{\partial \mathbf{c}_u}{\partial \mathbf{u}}\right)^T (\mathbf{u}^*(t)) \cdot \boldsymbol{\gamma}(t) \\ &= \mathbf{0} \end{aligned} \qquad (4.60)$$

holds for almost every $t \in [t_0, t_f]$*;*

3. $\boldsymbol{\gamma}(t) \in \hat{\mathcal{C}}^0\left([t_0, t_f], \mathbb{R}^{N_{cu}}\right)$ *are time-dependent multipliers, which satisfy the complementarity condition (4.45) for almost every* $t \in [t_0, t_f]$*;*
4. $\boldsymbol{\rho}(t) \in \hat{\mathcal{C}}^0\left([t_0, t_f], \mathbb{R}^{N_{cx}}\right)$ *are time-dependent multipliers and subject to the complementarity condition*

$$\begin{aligned} &\rho_i(t) \geq 0, \ \mathbf{c}_x^{[i]}(\cdot) \leq 0, \quad i \in \{1, \ldots, N_{c_x}\} \\ &\boldsymbol{\rho}^T(t) \mathbf{c}_x(\mathbf{x}^*(t)) = 0 \end{aligned} \qquad (4.61)$$

that holds for almost every $t \in [t_0, t_f]$*;*

5. *at the final time* t_f *the transversality condition*

$$\boldsymbol{\lambda}_{[\mathcal{I}_f^c]}(t_f) = \frac{\partial m}{\partial \mathbf{x}_{[\mathcal{I}_f^c]}(t_f)}(\mathbf{x}^*(t_f)) + \left(\frac{\partial \mathbf{c}_x}{\partial \mathbf{x}_{[\mathcal{I}_f^c]}}\right)^T_{t=t_f} (\mathbf{x}^*(t_f)) \cdot \boldsymbol{\alpha}_f \qquad (4.62)$$

is fulfilled, where $\boldsymbol{\alpha}_f \in \mathbb{R}^{N_{cx}}$ *is a vector of multipliers, for which the complementary condition*

$$\begin{aligned} &\boldsymbol{\alpha}_f^{[i]} \geq 0, \ \mathbf{c}_x^{[i]}(\mathbf{x}^*(t_f)) \leq 0, \quad i \in \{1, \ldots, N_{c_x}\} \\ &\boldsymbol{\alpha}_f^T \mathbf{c}_x(\mathbf{x}^*(t_f)) = 0 \end{aligned} \qquad (4.63)$$

holds; and

6. *for each boundary interval or contact time, the costates may be discontinuous at any time* $t_b \in [\mathbf{t}_{ent}^{[b]}, \mathbf{t}_{ex}^{[b]}]$*, which means that the following jump conditions hold:*

$$\boldsymbol{\lambda}(t_b^-) = \boldsymbol{\lambda}(t_b^+) + \left(\frac{\partial \mathbf{c}_x}{\partial \mathbf{x}}\right)_{t=t_b}^{T} (\mathbf{x}^*(t_b)) \cdot \boldsymbol{\pi}_b \tag{4.64}$$

$$\mathcal{H}(t_b^-) = \mathcal{H}(t_b^+), \tag{4.65}$$

where $\boldsymbol{\pi}_b \in \mathbb{R}^{N_{c_x}}$ *is a vector of multipliers with* $\boldsymbol{\pi}_b^{[i]} \geq 0$, $\boldsymbol{\pi}_b^T \mathbf{c}_x(\mathbf{x}^*(t_b)) = 0$ *and* t_b^- *and* t_b^+ *denote the left-hand side and the right-hand side limits, respectively.*

△

Proof A proof is given in the references of Hartl et al. [30]. □

Remark 4.3 Please note, that the Hamiltonian (4.65) is continuous, because we are dealing with autonomous problems only.

Remark 4.4 In many practical applications, the state constraints are simple box-constraints, which means simply that each constraint of them is linear in x_l for one index $l \in \{1, \ldots, N_x\}$. The box-constraints may apply for some or all x_l. Let us assume, that $l \in \mathcal{I}_f^c$ is the index for the state and $i \in \{1, \ldots, N_{c_x}\}$ is the index of the corresponding constraint. Then, one obtains the following differentiations:

- if $x_l^*(t_f)$ is not constrained, $\frac{\partial \mathbf{c}_x^{[i]}}{\partial x_l}(\mathbf{x}^*(t_f)) = 0$ holds and the transversality condition simplifies to

$$\lambda_l(t_f) = \frac{\partial m}{\partial x_l(t_f)}(\mathbf{x}^*(t_f)); \tag{4.66}$$

  and
- if $x_l^*(t_f)$ is constrained by $\mathbf{c}_x^{[i]}(x_l^*(t_f))$, we can assume without loss of generality $\frac{\partial \mathbf{c}_x^{[i]}}{\partial x_l}(\mathbf{x}^*(t_f)) = 1$ and we have to distinguish two cases:
  1. if the state constraint is active at t_f, $\boldsymbol{\alpha}_f^{[i]} > 0$ holds according to the complementary condition. $\boldsymbol{\alpha}_f^{[i]}$ can then be chosen arbitrarily to satisfy the transversality condition. Therefore, this transversality condition can be omitted as a boundary condition and has to be replaced by the state constraint $\mathbf{c}_x^{[i]}(x_l^*(t_f)) = 0$; and
  2. if the state constraint is not active at t_f, $\boldsymbol{\alpha}_f^{[i]} = 0$ holds according to the complementary condition and the transversality condition simplifies to (4.66).

Remark 4.5 The necessary conditions of Theorem 4.5 still have some practical handicaps:

- a correlation of the multiplier functions $\boldsymbol{\rho}(\cdot)$ and $\boldsymbol{\pi}_b$ is not directly obvious. However, if one can show that a nonincreasing function of bounded variation $\tilde{\boldsymbol{\rho}}(\cdot)$ with piecewise continuous derivative exists, then, one can set

$$\boldsymbol{\rho}(t) = -\dot{\tilde{\boldsymbol{\rho}}}(t)$$

for every t for which $\dot{\tilde{\boldsymbol{\rho}}}(\cdot)$ exists and

$$\boldsymbol{\pi}_b = \tilde{\boldsymbol{\rho}}(t_b^-) - \tilde{\boldsymbol{\rho}}(t_b^+)$$

at any one time $t_b \in [\mathbf{t}_{ent}^{[b]}, \mathbf{t}_{ex}^{[b]}]$ and for all $b = 1, \ldots, N_{c_x}^{arc}$, where $\tilde{\boldsymbol{\rho}}(\cdot)$ is not differentiable. The proof of Maurer [41] relies on this fact that $\tilde{\boldsymbol{\rho}}(\cdot)$ is piecewise continuously differentiable; and

- the conditions usually require a guess of the beginning and the end of the constrained state arcs. This makes numerical approaches such as indirect methods, which uses these conditions directly, rather inflexible. If state constraints are of major importance, direct methods, which will be described in Chap. 8, usually yield very good results with more flexible algorithms.

Remark 4.6 Piecewise absolutely continuous functions $\mathcal{A}\hat{\mathcal{C}}^{\infty}$ have a fixed number of discontinuities at $t_a \in [t_0, t_f]$. Thus, the elements of $\mathcal{A}\hat{\mathcal{C}}^{\infty}$ are functions which are absolutely continuous on each interval $[t_a, t_{a+1})$, $a = 0, 1, \ldots, m-1$ and $[t_m, t_f]$; and is continuous from the right at the discontinuities $t_1, t_2, \ldots$ (Azbelev and Rakhmatullina [2]).

It is, in general, more difficult to derive the necessary conditions for pure state constraints since $\mathbf{c}_x(\cdot)$ does not explicitly depend on $\mathbf{u}(\cdot)$ and $\mathbf{x}(\cdot)$ can be controlled only indirectly through the ODE (4.49). For mixed control-state problems, we do not state these necessary conditions but interested readers may consult Hartl et al. [30].

4.2.3 Necessary Conditions for Optimal Control Problems with Affine Controls

Considering an affine optimal control problem

$$\min_{\mathbf{u}(\cdot)\in L^{\infty}(([t_0,t_f],\mathsf{U})} \phi(\mathbf{u}(\cdot)) = m(\mathbf{x}^*(t_f)) + \int_{t_0}^{t_f} l_0(\mathbf{x}^*(t)) + \mathbf{l}_1^T(\mathbf{x}^*(t)) \cdot \mathbf{u}^*(t)\, \mathrm{d}t \tag{4.67}$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}_0(\mathbf{x}(t)) + \sum_{i=1}^{N_u} \mathbf{f}_i(\mathbf{x}(t)) \cdot u_i(t), \quad \text{for a.e. } t \in [t_0, t_f] \tag{4.68}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{4.69}$$

$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f \tag{4.70}$$

$$\mathbf{c}_u(\mathbf{u}(t)) \leq \mathbf{0}, \qquad \forall t \in [t_0, t_f] \tag{4.71}$$

the Hamiltonian is given by

$$\mathcal{H}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}(t)) = \lambda_0 l_0(\boldsymbol{x}(t)) + \boldsymbol{\lambda}^T(t)\mathbf{f}_0(\mathbf{x}(t)) + \lambda_0 \mathbf{l}_1^T(\mathbf{x}(t)) \cdot \mathbf{u}(t) + \boldsymbol{\lambda}^T(t) \sum_{i=1}^{N_u} \mathbf{f}_i(\mathbf{x}(t)) \cdot u_i(t). \quad (4.72)$$

The derivation of the Hamiltonian with respect to $\mathbf{u}(\cdot)$ and equating to zero yields the *singularity condition*

$$\frac{\mathrm{d}\mathcal{H}}{\mathrm{d}\mathbf{u}}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}(t)) = \lambda_0 \mathbf{l}_1(\mathbf{x}(t)) + \sum_{i=1}^{N_u} \mathbf{f}_i^T(\mathbf{x}(t))\boldsymbol{\lambda}(t) = \mathbf{0}_{N_u \times 1}. \quad (4.73)$$

Equation (4.73) is independent of the control $\mathbf{u}(\cdot)$. Thus, $\mathbf{u}(\cdot)$ may take any value from the admissible set. However, the stationary condition for the Hamiltonian minimum condition still applies

$$\frac{\mathrm{d}^r}{\mathrm{d}t^r}\left(\frac{\partial \mathcal{H}}{\partial \mathbf{u}}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}(t))\right) = \mathbf{0}.$$

The i-th continuous-valued control $u_i(\cdot)$ is said to have a *degree of singularity* r if $u_i(\cdot)$ appears explicitly for the first time in

$$\frac{\partial}{\partial u_i}\left[\frac{\mathrm{d}^r}{\mathrm{d}t^r}\left(\frac{\partial \mathcal{H}}{\partial \mathbf{u}}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}(t))\right)\right] \neq 0.$$

We denote the function

$$\mathbf{S}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \lambda_0) = \lambda_0 \mathbf{l}_1(\mathbf{x}(t)) + \sum_{i=1}^{N_u} \mathbf{f}_i^T(\mathbf{x}(t))\boldsymbol{\lambda}(t)$$

as *switching function*. Using the switching function the Hamiltonian minimum condition yields

$$\min_{\mathbf{u}(t) \in \hat{\mathcal{U}}} \mathcal{H}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)) = \lambda_0 l_0(\boldsymbol{x}(t)) + \boldsymbol{\lambda}^T(t)\mathbf{f}_0(\mathbf{x}(t)) + \mathbf{S}^T(\mathbf{x}(t), \boldsymbol{\lambda}(t), \lambda_0) \cdot \mathbf{u}(t).$$

Since the affine control is lower and upper bounded with $\mathbf{u}(t) \in \hat{\mathcal{U}} = \left[\mathbf{u}^{min},\ \mathbf{u}^{max}\right]$, we obtain from the minimum condition the optimal control as

$$u_i^*(t) = \begin{cases} u_i^{max} & S_i(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0) < 0 \\ u_i^{min} & S_i(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0) > 0 \\ u_i^0 \in \left(u_i^{min}, u_i^{max}\right) & S_i(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0) = 0 \end{cases} \quad (4.74)$$

for all $i \in \{1, \ldots, N_u\}$. For the case that the switching function $S_i(\mathbf{x}^*, \boldsymbol{\lambda}, \lambda_0)$ on the time interval $[t_1, t_2] \subset [t_0, t_f]$ has only isolated zeros, the optimal control takes values on the control boundaries with $u_i^*(t) \in \left\{u_i^{min}, u_i^{max}\right\}$ and is called a *bang–bang* control on the time interval $[t_1, t_2]$. For time intervals $[t_1, t_2] \subset [t_0, t_f]$ where the switching function is completely zero, i.e., $S_i(\mathbf{x}^*, \boldsymbol{\lambda}, \lambda_0) = 0$, the control u_i is called *singular*.

Theorem 4.6 (First-order necessary Conditions for Affine Optimal Control Problems) *Let* $\mathbf{u}^*(\cdot) \in L^\infty\left([t_0, t_f], \mathsf{U}\right)$ *be a measurable and essentially bounded optimal control function and* $\mathbf{x}^*(\cdot) \in \mathcal{AC}^\infty\left([t_0, t_f], \mathsf{X}\right)$ *be the corresponding absolutely continuous optimal state function. Then,* $(\mathbf{x}^*(\cdot), \mathbf{u}^*(\cdot))$ *is an optimal pair for the affine optimal control problem (4.67)–(4.71) over the fixed time interval* $[t_0, t_f]$. *Then, there exist a constant* $\lambda_0 \geq 0$ *and absolutely continuous costates* $\boldsymbol{\lambda}(\cdot) \in \mathcal{AC}^\infty\left([t_0, t_f], \mathbb{R}^{N_x}\right)$ *satisfying the nontrivial solution* $(\lambda_0, \boldsymbol{\lambda}(t)) \neq \mathbf{0}$ *for every* $t \in [t_0, t_f]$ *for which the following conditions hold with the Hamiltonian defined by (4.72):*

1. *the states* $\mathbf{x}^*(\cdot)$ *and the costates* $\boldsymbol{\lambda}(\cdot)$ *satisfy the canonical equations with respect to the Hamiltonian (4.72)*

$$\begin{aligned}
\dot{\mathbf{x}}^*(t) &= \frac{\partial \mathcal{H}}{\partial \boldsymbol{\lambda}}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}^*(t)) \\
&= \mathbf{f}_0(\mathbf{x}^*(t)) + \sum_{i=1}^{N_u} \mathbf{f}_i(\mathbf{x}^*(t)) \cdot u_i^*(t), \qquad (4.75) \\
\dot{\boldsymbol{\lambda}}(t) &= -\frac{\partial \mathcal{H}}{\partial \mathbf{x}}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}^*(t)) \\
&= -\frac{\partial l_0}{\partial \mathbf{x}}(\mathbf{x}^*(t)) \cdot \lambda_0 - \left(\frac{\partial \mathbf{l}_1}{\partial \mathbf{x}}\right)^T (\mathbf{x}^*(t)) \cdot \mathbf{u}^*(t) \cdot \lambda_0 \qquad (4.76) \\
&\quad - \left(\frac{\partial \mathbf{f}_0}{\partial \mathbf{x}}\right)^T (\mathbf{x}^*(t)) \cdot \boldsymbol{\lambda}(t) + \sum_{i=1}^{N_u} \left(\frac{\partial \mathbf{f}_i}{\partial \mathbf{x}}\right)^T (\mathbf{x}^*(t)) \cdot u_i^*(t) \cdot \boldsymbol{\lambda}(t)
\end{aligned}$$

   for almost every $t \in [t_0, t_f]$ *and with the boundary conditions* $\mathbf{x}^*(t_0) = \mathbf{x}_0$ *and* $\mathbf{x}^*_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f$*;*
2. *the Hamiltonian minimum condition*

$$\mathbf{u}^*(t) = \arg\min_{\mathbf{u}(t) \in \hat{\mathcal{U}}} \mathcal{H}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \mathbf{u}(t)) \qquad (4.77)$$

   holds for almost every $t \in [t_0, t_f]$ *and yields the optimal control (4.74); and*
3. *at the final time* t_f *the transversality condition evaluated on the terminal Lagrangian*

$$\boldsymbol{\lambda}_{[\mathcal{I}_f^c]}(t_f) = \frac{\partial m}{\partial \mathbf{x}_{[\mathcal{I}_f^c]}(t_f)}(\mathbf{x}^*(t_f)) \qquad (4.78)$$

   is fulfilled.

△

4.3 Hamilton–Jacobi–Bellman Equation

As we have learned from the previous section, the maximum principle states the necessary conditions for fixed optimal states of the system. Bellman [4] recognized this weakness and created a new idea called *dynamic programming* right around the time when the maximum principle was being developed with the purpose to determine the optimal controls at any state of the system. This results in a theory for obtaining necessary as well as sufficient conditions for optimality expressed in terms of the so-called Hamilton–Jacobi–Bellman partial differential equation. Despite its difference to the maximum principle, dynamic programming and the maximum principle have their foundations in calculus of variations and there are important connections between the two.

Let us reconsider the optimal control problem (4.25)–(4.28) from Sect. 4.2 with an empty set $\mathcal{I}_f$ and let us assume that the optimal controls $\mathbf{u}(\cdot) \in L^\infty(\cdot, \mathsf{U})$ and the corresponding absolutely continuous optimal states $\mathbf{x}(\cdot) \in \mathcal{AC}^\infty(\cdot, \mathsf{X})$ exist. Instead of minimizing the optimal control problem $\phi(\mathbf{u}(\cdot))$ for given t_0 and $\mathbf{x}_0$, we consider now a series of minimization problems associated with the cost functionals

$$\phi(\mathbf{x}(\cdot), \mathbf{u}(\cdot), t) = m(\mathbf{x}(t_f)) + \int_t^{t_f} l(\mathbf{x}(\tau), \mathbf{u}(\tau))\, \mathrm{d}\tau,$$

where the function $\mathbf{x}(\cdot)$ has to be interpreted as independent from the control function $\mathbf{u}(\cdot)$. This means in other words, we minimize a family of cost functionals $\phi(\mathbf{x}(\cdot), \mathbf{u}(\cdot), t)$ without adjoined differential equations. It is also worth noting that the cost functionals depend now on time.

For these functionals, let us define a function that measures the cost of completing the trajectory denoted as the *value function* or *cost-to-go*. The value function $V : \mathcal{AC}^\infty(\cdot, \mathsf{X}) \times [t_0, t_f] \to \mathbb{R}$ is again a functional depending on the function $\mathbf{x}(\cdot)$ and the time t. For fixed $\mathbf{x}(\cdot) \in \mathcal{AC}^\infty(\cdot, \mathsf{X})$ it becomes a function depending only on the time. We use both interpretations of the value function in the following chapters.

Let us assume that the value function is continuously differentiable w.r.t. $\mathbf{x}(\cdot)$ and t, i.e., $V(\cdot) \in \mathcal{C}^1$. Then, $V(\cdot)$ provides the optimal cost for a trajectory $\mathbf{x}(\cdot)$ starting at time t

$$V(\mathbf{x}(\cdot), t) := \min_{\mathbf{u}(\cdot) \in L^\infty(\cdot, \mathsf{U})} \left\{ m(\mathbf{x}(t_f)) + \int_t^{t_f} l(\mathbf{x}(\tau), \mathbf{u}(\tau))\, \mathrm{d}\tau \right\}. \tag{4.79}$$

Obviously, the value function satisfies the boundary condition

$$V(\mathbf{x}(\cdot), t_f) = m(\mathbf{x}(t_f)). \tag{4.80}$$

In particular, if there is no Mayer term then we have $V(\mathbf{x}(\cdot), t_f) = 0$.

We are now ready for the statement of an important concept known as *principle of optimality*.

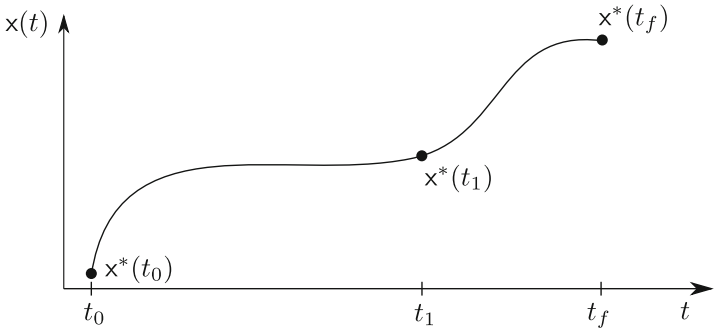


Fig. 4.3 Optimal trajectory $\mathbf{x}^*(t)$ from $\mathbf{x}^*(t_0)$ to $\mathbf{x}^*(t_f)$ and the partial trajectory from $\mathbf{x}^*(t_1)$ to $\mathbf{x}^*(t_f)$ that is also optimal

Theorem 4.7 (Principle of Optimality) *For every pair* $(\mathbf{x}(\cdot), t) \in \mathcal{AC}^\infty(\cdot, \mathsf{X}) \times [t_0, t_f]$ *and every* $\Delta t \in (0, t_f - t)$, *the value function* $V(\cdot)$ *(4.79) satisfies the relation*

$$V(\mathbf{x}(\cdot), t) = \min_{\mathbf{u}(\cdot) \in L^\infty(\cdot, \mathsf{U})} \left\{ \int_t^{t+\Delta t} l(\mathbf{x}(\tau), \mathbf{u}(\tau))\, \mathrm{d}\tau + V(\mathbf{x}(t+\Delta t), t+\Delta t) \right\}. \tag{4.81}$$

△

Proof The proof is given in Bellman [4]. □

Theorem 4.7 states that the search for an optimal control is equivalent to the search for an optimal control over a small time interval that minimizes the cost over this interval, plus the subsequent optimal cost-to-go. For a better understanding, let us split the time interval $[t, t_f]$ into two subintervals: $[t, t_1]$ and $[t_1, t_f]$ for an arbitrary $t_1 \in (t, t_f)$ as shown in Fig. 4.3. Then, the principle of optimality (4.81) can be written as

$$\begin{aligned} V(\mathbf{x}(\cdot), t) &= \min_{\mathbf{u}(\cdot) \in L^\infty(\cdot, \mathsf{U})} \left\{ \int_t^{t_1} l(\mathbf{x}(\tau), \mathbf{u}(\tau))\, \mathrm{d}\tau + \int_{t_1}^{t_f} l(\mathbf{x}(t), \mathbf{u}(t))\, \mathrm{d}t + m(\mathbf{x}(t_f)) \right\} \\ &\overset{\text{Eq. (4.79)}}{=} \min_{\mathbf{u}(\cdot) \in L^\infty(\cdot, \mathsf{U})} \left\{ \int_t^{t_1} l(\mathbf{x}(\tau), \mathbf{u}(\tau))\, \mathrm{d}\tau + V(\mathbf{x}(\cdot), t_1) \right\}. \end{aligned} \tag{4.82}$$

It is obvious from (4.82), if $(\mathbf{x}^*(\cdot), \mathbf{u}^*(\cdot))$ is an optimal solution starting at $\mathbf{x}^*(t)$ passing through $\mathbf{x}^*(t_1)$ and ending at $\mathbf{x}^*(t_f)$, then the partial trajectory from $\mathbf{x}^*(t_1)$ to $\mathbf{x}^*(t_f)$ is also optimal with respect to the same value function and $\mathbf{x}^*(t_1)$ as initial condition. The reader may notice that the value function appears on both side of (4.82). Thus, we can think of (4.82) as describing a dynamic relationship among the optimal values of the costs for different $\mathbf{x}(\cdot)$ and t.

We can rewrite (4.81) into a more compact version, which will take the form of a *partial differential equation* (PDE). Let us express $V(\mathbf{x}^*(t+\Delta t), t+\Delta t)$ using the first-order Taylor series expansion as

$$\begin{aligned} V(\mathbf{x}^*(t+\Delta t), t+\Delta t) &= V(\mathbf{x}^*(t), t) + \frac{\partial V}{\partial t}(\mathbf{x}^*(t), t)\Delta t + \left(\frac{\partial V}{\partial \mathbf{x}}\right)^T (\mathbf{x}^*(t), t)\frac{\partial \mathbf{x}}{\partial t}(t)\Delta t + \mathcal{O}(\Delta t) \\ &= V(\mathbf{x}^*(t), t) + \frac{\partial V}{\partial t}(\mathbf{x}^*(t), t)\Delta t + \left(\frac{\partial V}{\partial \mathbf{x}}\right)^T (\mathbf{x}^*(t), t)\mathbf{f}(\mathbf{x}^*(t), \mathbf{u}(t))\Delta t \\ &\quad + \mathcal{O}(\Delta t). \end{aligned} \tag{4.83}$$

We also have from (4.81)

$$\int_t^{t+\Delta t} l(\mathbf{x}^*(\tau), \mathbf{u}(\tau))\, \mathrm{d}\tau = l(\mathbf{x}^*(t), \mathbf{u}(t))\Delta t + \mathcal{O}(\Delta t). \tag{4.84}$$

Inserting (4.83) and (4.84) into (4.81) yields after some cancelations

$$-\frac{\partial V}{\partial t}(\mathbf{x}^*(t), t) = \min_{\mathbf{u}(t)\in\hat{\mathcal{U}}(t)} \left\{ l(\mathbf{x}^*(t), \mathbf{u}(t)) + \left(\frac{\partial V}{\partial \mathbf{x}}\right)^T (\mathbf{x}^*(t), t) \cdot \mathbf{f}(\mathbf{x}^*(t), \mathbf{u}(t)) \right\}. \tag{4.85}$$

The PDE (4.85) is called the *Hamilton–Jacobi–Bellman* (HJB) equation. The HJB equation holds for almost every $t \in [t_0, t_f)$ and all $\mathbf{x}^*(t) \in \hat{\mathcal{X}}$ where $V(\cdot)$ is continuously differentiable with respect to these arguments. Technically speaking, the significance of the HJB equation is its property to reduce the problem to an optimization at each stage by finding $\mathbf{u}(t)$ that minimizes (4.85) for each fixed $\mathbf{x}^*$ that solve consequently the PDE for $V(\cdot)$. We call this the classical result which can be found in many textbooks. We see that the link between the optimal control problem (4.25)–(4.28) and the HJB equation is provided by the dynamic programming principle.

The adjoining of the right-hand side of the dynamical system to the Lagrangian $l(\cdot)$ by $\partial V/\partial \mathbf{x}$ lets us restate the Hamiltonian as

$$\mathcal{H}(\mathbf{x}^*(t), \partial V/\partial \mathbf{x}, \mathbf{u}^*(t)) = l(\mathbf{x}^*(t), \mathbf{u}^*(t)) + \left(\frac{\partial V}{\partial \mathbf{x}}\right)^T (\mathbf{x}^*(t), t) \cdot \mathbf{f}(\mathbf{x}^*(t), \mathbf{u}^*(t)). \tag{4.86}$$

From (4.86) we see that $\partial V/\partial \mathbf{x}$ corresponds with the costates from the PMP

$$\boldsymbol{\lambda}(t) = \frac{\partial V}{\partial \mathbf{x}}(\mathbf{x}^*(t), t). \tag{4.87}$$

This reveals the Hamiltonian minimum condition from the HJB equation

$$\frac{\partial V}{\partial t}(\mathbf{x}^*(t), t) = - \min_{\mathbf{u}(t)\in\hat{\mathcal{U}}(t)} \mathcal{H}(\mathbf{x}^*(t), \partial V/\partial \mathbf{x}, \mathbf{u}(t)). \tag{4.88}$$

Applying (4.87) to the boundary condition (4.80) yields the transversality condition from the HJB equation

$$\boldsymbol{\lambda}(t_f) = \left(\frac{\partial V}{\partial \mathbf{x}}\right)_{t=t_f} (\mathbf{x}^*(t_f), t_f) = \frac{\partial m}{\partial \mathbf{x}(t_f)}(\mathbf{x}^*(t_f)). \tag{4.89}$$

We see that the right-hand side of (4.88) is analogous to the Hamiltonian minimum condition from the PMP, but both, the PMP and the HJB equation, are derived differently with different assumptions.

The HJB equation (4.85) and the Hamiltonian minimum condition (4.88) constitute first-order necessary conditions for optimality. It can be further shown that these conditions are sufficient too (Liberzon [40]). However, these first-order conditions obtained from the HJB equation are too restrictive to deduce the PMP from Sect. 4.2. The obstacle in preventing that, is the assumption of globally continuously differentiable value functions with respect to $\mathbf{x}(\cdot)$ and t, i.e., $V(\cdot)$ is of class $\mathcal{C}^1([t_0, t_f], \mathsf{X})$. A fact that we cannot expect to be true in general. This lack of smoothness, even in simple problems, was recognized as a severe restriction to the range of applicability of Hamilton–Jacobi theory to OCPs.

However, it would be valuable with respect to the forthcoming section to obtain a relationship to the minimum principle. A satisfactory mathematical foundation has been established by Crandall-Lions [17] notion of *viscosity solutions* of Hamilton–Jacobi equations which is based on first-order *semidifferentials*. Viscosity solutions need not be differentiable anywhere and thus do not suffer from the classical differentiability problem. We introduce this concept in an informal way. For more details about this topic we refer interested readers to the works of Bardi and Capuzzo-Dolcetta [3], Capuzzo-Dolcetta [11].

Let us first formulate the definition of viscosity solutions in an appealing form. We start by assuming a continuous function $v : \mathsf{X} \to \mathbb{R}$. The function $v(\cdot)$ is said to be differentiable at $\mathbf{x} \in \mathsf{X}$ that yields $\nabla v(\mathbf{x}) = \mathbf{r} \in \mathsf{X}$, if we admit

$$v(\mathbf{y}) = v(\mathbf{x}) + \mathbf{r}^T \cdot (\mathbf{y} - \mathbf{x}) + \mathcal{O}(|\mathbf{y} - \mathbf{x}|) \tag{4.90}$$

for all $\mathbf{y} \in \mathbb{R}^{N_x}$ where $\nabla v(\cdot)$ denotes the gradient of $v(\cdot)$. Then (4.90) can be split into the two relations

$$\limsup_{\mathbf{y}\to\mathbf{x}} \frac{v(\mathbf{y}) - v(\mathbf{x}) - \mathbf{r}^T \cdot (\mathbf{y} - \mathbf{x})}{|\mathbf{y} - \mathbf{x}|} \leq 0 \tag{4.91}$$

and

$$\liminf_{\mathbf{y}\to\mathbf{x}} \frac{v(\mathbf{y}) - v(\mathbf{x}) - \mathbf{r}^T \cdot (\mathbf{y} - \mathbf{x})}{|\mathbf{y} - \mathbf{x}|} \geq 0. \tag{4.92}$$

We denote $\mathbf{r}$ in (4.91) as a *supergradient*. Similarly, we denote $\mathbf{r}$ in (4.92) as a *sub-gradient*. These sub- and supergradients $\mathbf{r}$ are in general not unique. Thus, we have a function $D^+v(\cdot)$, called super-differential, which maps $\mathbf{x} \in \mathsf{X}$ to a set of supergradients of $v(\mathbf{x})$ denoted with

$$D^+v(\mathbf{x}) := \left\{ \mathbf{r} \in \mathbb{R}^{N_x} \;\middle|\; \limsup_{\mathbf{y}\to\mathbf{x}} \frac{v(\mathbf{y}) - v(\mathbf{x}) - \mathbf{r}^T \cdot (\mathbf{y} - \mathbf{x})}{|\mathbf{y} - \mathbf{x}|} \leq 0 \right\}$$

and a function $D^-v(\cdot)$, called sub-differential, which maps $\mathbf{x} \in \mathsf{X}$ to a set of sub-gradients of $v(\mathbf{x})$ denoted with

$$D^-v(\mathbf{x}) := \left\{ \mathbf{r} \in \mathbb{R}^{N_x} \;\middle|\; \liminf_{\mathbf{y}\to\mathbf{x}} \frac{v(\mathbf{y}) - v(\mathbf{x}) - \mathbf{r}^T \cdot (\mathbf{y} - \mathbf{x})}{|\mathbf{y} - \mathbf{x}|} \geq 0 \right\}.$$

It is then a trivial consequence that if both $D^+v(\mathbf{x})$ and $D^-v(\mathbf{x})$ are nonempty at some $\mathbf{x}$, then $D^+v(\mathbf{x}) = D^-v(\mathbf{x}) = \{\nabla v(\mathbf{x})\}$ and $v(\cdot)$ is differentiable at $\mathbf{x}$.

Next, we need the concept of a viscosity solution for PDEs. Consider a PDE of the form

$$F(\mathbf{x}, v(\mathbf{x}), \nabla v(\mathbf{x})) = 0, \tag{4.93}$$

where $F : \mathsf{X} \times \mathbb{R} \times \mathbb{R}^{N_x} \to \mathbb{R}$ is a continuous function. The terminology "viscosity solution" is borrowed from the fluid mechanics where the motion of a viscous fluid is described by PDEs.

We may now define the concept of a viscosity solution of (4.93).

Definition 4.4 (*Viscosity Solution (cf. Crandall et al.* [18])) A viscosity solution of

$$F(\mathbf{x}, v(\mathbf{x}), \nabla v(\mathbf{x})) = 0$$

is a continuous function $v(\cdot) \in \mathcal{C}^0$ satisfying

$$F(\mathbf{x}, v(\mathbf{x}), \mathbf{r}) \leq 0, \quad \forall \mathbf{r} \in D^+v(\mathbf{x}), \ \forall \mathbf{x} \tag{4.94}$$

and

$$F(\mathbf{x}, v(\mathbf{x}), \mathbf{r}) \geq 0, \quad \forall \mathbf{r} \in D^-v(\mathbf{x}), \ \forall \mathbf{x}. \tag{4.95}$$

△

This can be restated in an equivalent notion that is more manageable using a continuously differentiable *test function* (or *verification function*) $\varphi : \mathsf{X} \to \mathbb{R}$.

Definition 4.5 (*Viscosity Solution using a Test Function (cf. Crandall et al.* [18])) $v(\cdot) \in \mathcal{C}^0$ is a viscosity solution of

$$F(\mathbf{x}, v(\mathbf{x}), \nabla v(\mathbf{x})) = 0$$

provided that for all $\varphi(\cdot) \in \mathcal{C}^1$,

$$\text{if } \varphi(\mathbf{x}) - v(\mathbf{x}) \text{ attains a local minimum at } \mathbf{x}, \text{ then}$$
$$F(\mathbf{x}, v(\mathbf{x}), \nabla\varphi(\mathbf{x})) \leq 0 \tag{4.96}$$

$$\text{if } \varphi(\mathbf{x}) - v(\mathbf{x}) \text{ attains a local maximum at } \mathbf{x}, \text{ then}$$
$$F(\mathbf{x}, v(\mathbf{x}), \nabla\varphi(\mathbf{x})) \geq 0. \tag{4.97}$$

△

Viscosity solutions satisfying (4.94) or (4.96) are called *viscosity subsolutions* and viscosity solutions satisfying (4.95) or (4.97) are called *viscosity supersolutions*. A proof for the equivalence between Definitions 4.4 and 4.5 can be found in Bardi and Capuzzo-Dolcetta [3].

Now, we are ready to interpret the HJB equation in the viscosity sense

$$-\frac{\partial V}{\partial t}(\mathbf{x}^*(t), t) - \min_{\mathbf{u}(t)\in\hat{\mathcal{U}}(t)} \left\{ l(\mathbf{x}(t), \mathbf{u}(t)) + \left(\frac{\partial V}{\partial \mathbf{x}}\right)^T (\mathbf{x}(t), t) \cdot \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \right\} = 0. \tag{4.98}$$

Let us transform (4.98) to an autonomous PDE by introducing an additional state as a representation of the time. Then, the HJB (4.98) takes the form (4.93), except that $v(\cdot)$ is defined on $\mathbb{R}^{N_x+1}$.

Then, the value function $V(\cdot)$ is a unique viscosity subsolution (also Lipschitz) of the HJB equation (4.85) with the boundary condition (4.80), if for any fixed pair $\tilde{\mathbf{x}}_0 = (\mathbf{x}_0, t_0)$ and for every test function $\varphi(\tilde{\mathbf{x}})$, such that $\varphi(\tilde{\mathbf{x}}) - V(\tilde{\mathbf{x}})$ attains a local minimum at $(\mathbf{x}_0, t_0)$, the inequality

$$-\frac{\partial \varphi}{\partial t}(\tilde{\mathbf{x}}_0) - \min_{\mathbf{u}(t)\in\hat{\mathcal{U}}(t)} \left\{ l(\mathbf{x}_0, \mathbf{u}(t)) + \left(\frac{\partial \varphi}{\partial \mathbf{x}}\right)^T (\tilde{\mathbf{x}}_0) \cdot \mathbf{f}(\mathbf{x}_0, \mathbf{u}(t)) \right\} \leq 0$$

is satisfied. This result holds under the assumptions: $\mathbf{f}(\cdot), l(\cdot)$, and $m(\cdot)$ are uniformly continuous w.r.t. all arguments, $\partial\mathbf{f}/\partial\mathbf{x}$, $\partial l/\partial\mathbf{x}$, and $\partial m/\partial\mathbf{x}$ are bounded, and the set of admissible controls $\hat{\mathcal{U}}(t)$ is compact (cf. Liberzon [40]).

Despite the problem discussed above, the HJB equation admits new interpretations. The minimum principle discussed in Sect. 4.2 states that for a.e. $t \in [t_0, t_f]$, the optimal controls $\mathbf{u}^*(\cdot)$ must satisfy

$$\mathbf{u}^*(t) = \arg \min_{\mathbf{u}(t) \in \hat{\mathcal{U}}(t)} \mathcal{H}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)).$$

The optimal controls $\mathbf{u}^*(\cdot)$ depend not only on the optimal states $\mathbf{x}^*(\cdot)$ but also on the costates $\boldsymbol{\lambda}(\cdot)$. In case of the HJB equation, the optimal controls $\mathbf{u}^*(\cdot)$ must satisfy:

$$\mathbf{u}^*(t) = \arg \min_{\mathbf{u}(t) \in \hat{\mathcal{U}}(t)} \mathcal{H}\left(\mathbf{x}^*(t), \frac{\partial V}{\partial \mathbf{x}}(\mathbf{x}^*(t), t), \mathbf{u}(t)\right).$$

The optimal controls $\mathbf{u}^*(\cdot)$ are completely determined by the optimal states $\mathbf{x}^*(\cdot)$. In the forthcoming chapters we will denote these principles as open-loop and closed-loop strategies, respectively.

4.4 Hybrid Minimum Principle

Due to the growing interest in optimal control of hybrid systems, significant research has been done over the last two decades on investigating first-order necessary conditions. Thus, let us consider the *hybrid optimal control problem* (HOCP) of the form

$$\phi\left(q^*(\cdot), \mathbf{u}^*(\cdot)\right) = \min_{q(\cdot) \in \mathcal{Q},\ \mathbf{u}(\cdot) \in \mathcal{U}(q(\cdot))} \phi(q(\cdot), \mathbf{u}(\cdot)) = m\left(\mathbf{x}^*(t_f)\right) \tag{4.99}$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)), \qquad \text{for a.e. } t \in [t_0, t_f] \tag{4.100}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{4.101}$$

$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f \tag{4.102}$$

$$\boldsymbol{\varphi}(\mathbf{x}(t_j^-), \mathbf{x}(t_j^+)) = \mathbf{x}(t_j^+) - \mathbf{x}(t_j^-) - \boldsymbol{\delta}_{(q(t_j^-), q(t_j^+))}(\mathbf{x}(t_j^-)) = \mathbf{0}, \quad t_j \in \Theta_t, \tag{4.103}$$

where $j = 1, \ldots, N_{swt}$ and $\Theta_t := (t_1, t_2, \ldots, t_{N_{swt}})$ is the switching time sequence. On the one side, the most important result in the study of such systems is the *hybrid minimum principle* (HMP). The derivation of the classical PMP relies on a special class of needle-like control variations, which is rather complex to understand and makes it an inappropriate choice for the derivation of the HMP in the scope of this book. Therefore, we rely for the derivation of the HMP on a different approach, which was introduced by Dmitruk and Kaganovich [20] and avoids the complex variational approach, but takes advantage of the results of the PMP for continuous OCPs as discussed in Sect. 4.2. Dmitruk and Kaganovich showed that after some transformations of the original HOCP, the HMP is a consequence of the classical Pontryagin's minimum principle, if all discontinuities of the continuous-valued states or the costates

occur at transitions of the hybrid system. The procedure requires to consider the evolution of a hybrid system as hybrid trajectory as defined in Sect. 3.2.4 with the corresponding trajectories $(q_0, q_1, q_2 \ldots, q_{N_{exe}-1})$, $(\mathbf{x}_0(t), \mathbf{x}_1(t), \mathbf{x}_2(t), \ldots, \mathbf{x}_{N_{exe}-1}(t))$, $(\boldsymbol{\lambda}_0(t), \boldsymbol{\lambda}_1(t), \boldsymbol{\lambda}_2(t), \ldots, \boldsymbol{\lambda}_{N_{exe}-1}(t))$, and $(\mathbf{u}_0(t), \mathbf{u}_1(t), \mathbf{u}_2(t) \ldots, \mathbf{u}_{N_{exe}-1}(t))$.

For the sake of a clear presentation the HOCP (4.99)–(4.103) is stated without control and state constraints. Additionally, it is convenient for the derivation to consider the problem formulation as Mayer-type problem. Furthermore, we need an extension of the transversality conditions for the case of general boundary conditions

$$\boldsymbol{\psi}(\mathbf{x}(t_0), \mathbf{x}(t_f)) = \mathbf{0}.$$

This is necessary, because for the transformed problem not all initial values will be defined. The transversality conditions for the general boundary conditions are defined by (see Liberzon [40])

$$\boldsymbol{\lambda}(t_0) = -\left(\frac{\partial \boldsymbol{\psi}}{\partial \mathbf{x}(t_0)}\right)^T (\mathbf{x}(t_0), \mathbf{x}(t_f)) \cdot \boldsymbol{\pi} \tag{4.104}$$

$$\boldsymbol{\lambda}(t_f) = \frac{\partial m}{\partial \mathbf{x}(t_f)}(\mathbf{x}(t_f)) + \left(\frac{\partial \boldsymbol{\psi}}{\partial \mathbf{x}(t_f)}\right)^T (\mathbf{x}(t_0), \mathbf{x}(t_f)) \cdot \boldsymbol{\pi}. \tag{4.105}$$

We have now all ingredients to start. Let us assume that the system trajectory is decomposed into a hybrid trajectory. Then, the first step is to transform the hybrid trajectory such that all individual time intervals $[t_j, t_{j+1}]$ are mapped onto the same interval. Usually, all subintervals are mapped onto fixed intervals $[0, 1]$. In so doing, let us introduce a new time variable $\tau \in [0, 1]$, $\varsigma_j = t_{j+1} - t_j$, $j = 0, \ldots, N_{exe} - 1$, and the functions $\tilde{t}_j : [0, 1] \to \mathbb{R}$. With $\tilde{t}_j(\tau) = t_j + \varsigma_j \tau$, we obtain a set of linear initial value problems

$$\frac{\mathrm{d}\tilde{t}_j}{\mathrm{d}\tau}(\tau) = \varsigma_j, \quad \tilde{t}_j(0) = t_j.$$

The functions $\tilde{t}_j(\cdot)$ can be interpreted as an additional state variable which represents the time on the subintervals ς_j with the continuity conditions $\tilde{t}_{j-1}(1) = \tilde{t}_j(0)$, $j = 1, \ldots, N_{swt}$. Realizing that the continuous-valued states $\mathbf{x}_j(t)$, the discrete state $q_j(t)$, the continuous-valued controls $\mathbf{u}_j(t)$, and the costates $\boldsymbol{\lambda}_j(t)$ are now functions depending on τ, i.e., $\mathbf{x}_j(\tilde{t}_j(\tau))$, $q(\tilde{t}_j(\tau))$, $\mathbf{u}_j(\tilde{t}_j(\tau))$, and $\boldsymbol{\lambda}_j(\tilde{t}_j(\tau))$, respectively. For short, let us write $\tilde{\mathbf{x}}_j(\tau)$, $\tilde{q}_j(\tau)$, $\tilde{\mathbf{u}}_j(\tau)$, and $\tilde{\boldsymbol{\lambda}}_j(\tau)$.

The problem (4.99)–(4.103) can then be transformed into the new time domain $\tau \in [0, 1]$ as

$$\phi\left(\tilde{q}_j^*(\cdot), \tilde{\mathbf{u}}_j^*(\cdot)\right) = \min_{\tilde{q}_j(\cdot)\in\tilde{\mathcal{Q}},\, \tilde{\mathbf{u}}_j(\cdot)\in\tilde{\mathcal{U}}(\tilde{q}_j(\cdot))} \phi(\tilde{q}_j(\cdot), \tilde{\mathbf{u}}_j(\cdot)) = m\left(\tilde{\mathbf{x}}_{N_{exe}-1}^*(1)\right) \tag{4.106}$$

$$\frac{\mathrm{d}\tilde{\mathbf{x}}_j}{\mathrm{d}\tau}(\tau) = \varsigma_j \mathbf{f}_{\tilde{q}_j(\tau)}(\tilde{\mathbf{x}}_j(\tau), \tilde{\mathbf{u}}_j(\tau)), \quad j = 0, \ldots, N_{exe} - 1, \quad \text{for a.e. } \tau \in [0, 1] \tag{4.107}$$

$$\tilde{\mathbf{x}}_0(0) = \mathbf{x}_0 \tag{4.108}$$

$$\tilde{\mathbf{x}}_{N_{exe}-1}^{[\mathcal{I}_f]}(1) = \mathbf{x}_f \tag{4.109}$$

$$\boldsymbol{\varphi}(\tilde{\mathbf{x}}_{j-1}(1), \tilde{\mathbf{x}}_j(0)) = \tilde{\mathbf{x}}_j(0) - \tilde{\mathbf{x}}_{j-1}(1) - \boldsymbol{\delta}_{(\tilde{q}_{j-1}(1), \tilde{q}_j(0))}(\tilde{\mathbf{x}}_{j-1}(1)) = \mathbf{0}, \quad j = 1, \ldots, N_{swt} \tag{4.110}$$

$$\frac{\mathrm{d}\tilde{t}_j}{\mathrm{d}\tau}(\tau) = \varsigma_j, \quad j = 0, \ldots, N_{exe} - 1, \quad \text{for all } \tau \in [0, 1] \tag{4.111}$$

$$\tilde{t}_0(0) = t_0 \tag{4.112}$$

$$\tilde{t}_{N_{exe}-1}(1) = t_f \tag{4.113}$$

$$\tilde{t}_{j-1}(1) - \tilde{t}_j(0) = 0, \quad j = 1, \ldots, N_{swt}. \tag{4.114}$$

The general boundary conditions according to the transformed problem is given as

$$\begin{aligned}\boldsymbol{\psi}(\mathbf{x}_0(0), \ldots, \mathbf{x}_{N_{exe}-1}(1), \tilde{t}_0(0), \ldots, \tilde{t}_{N_{exe}-1}(1)) &= \begin{bmatrix} \tilde{\mathbf{x}}_0(0) - \mathbf{x}_0 \\ \boldsymbol{\varphi}(\tilde{\mathbf{x}}_{j-1}(1), \tilde{\mathbf{x}}_j(0)), \ j = 1, \ldots, N_{exe} - 1 \\ \tilde{\mathbf{x}}_{N_{exe}-1}^{[\mathcal{I}_f]}(1) - \mathbf{x}_f \\ \tilde{t}_0(0) - t_0 \\ \tilde{t}_{j-1}(1) - \tilde{t}_j(0), \ j = 1, \ldots, N_{exe} - 1 \\ \tilde{t}_{N_{exe}-1}(1) - t_f \end{bmatrix} \\ &= \mathbf{0}. \end{aligned} \tag{4.115}$$

Defining the Hamiltonian of the transformed problem as

$$\begin{aligned}\tilde{\mathcal{H}}\left(\tilde{\mathbf{x}}(\tau), \tilde{\boldsymbol{\lambda}}(\tau), \tilde{\rho}(\tau), \tilde{\mathbf{u}}(\tau), \varsigma\right) &= \sum_{j=0}^{N_{exe}-1} \tilde{\mathcal{H}}\left(\tilde{\mathbf{x}}_j(\tau), \tilde{\boldsymbol{\lambda}}_j(\tau), \tilde{\rho}_j(\tau), \tilde{\mathbf{u}}_j(\tau), \varsigma_j\right) \\ &= \sum_{j=0}^{N_{exe}-1} \left[\tilde{\boldsymbol{\lambda}}_j^T(\tau)\frac{\mathrm{d}\tilde{\mathbf{x}}_j}{\mathrm{d}\tau}(\tau) + \tilde{\rho}_j(\tau)\frac{\mathrm{d}\tilde{t}_j}{\mathrm{d}\tau}\right] \\ &= \sum_{j=0}^{N_{exe}-1} \varsigma_j \cdot \left[\tilde{\boldsymbol{\lambda}}_j^T(\tau)\mathbf{f}_{\tilde{q}_j(\tau)}(\tilde{\mathbf{x}}_j(\tau), \tilde{\mathbf{u}}_j(\tau)) + \tilde{\rho}_j(\tau)\right], \qquad \text{for a.e. } \tau \in [0, 1]\end{aligned}$$

lets us express the adjoint differential equations in the usual way

$$\begin{aligned}\frac{\mathrm{d}\tilde{\boldsymbol{\lambda}}_j}{\mathrm{d}\tau}(\tau) &= -\frac{\partial\tilde{\mathcal{H}}}{\partial\tilde{\mathbf{x}}_j}(\tilde{\mathbf{x}}_j(\tau), \tilde{\boldsymbol{\lambda}}_j(\tau), \tilde{\rho}_j(\tau), \tilde{\mathbf{u}}_j(\tau), \varsigma_j) \\ \frac{\mathrm{d}\tilde{\rho}_j}{\mathrm{d}\tau}(\tau) &= -\frac{\partial\tilde{\mathcal{H}}}{\partial\tilde{t}_j}(\tilde{\mathbf{x}}_j(\tau), \tilde{\boldsymbol{\lambda}}_j(\tau), \tilde{\rho}_j(\tau), \tilde{\mathbf{u}}_j(\tau), \varsigma_j)\end{aligned}$$

for almost every $\tau \in [0, 1]$, $j = 0, \ldots, N_{exe} - 1$. The reader should note that $\tilde{\rho}_j(\cdot)$ is the costate that adjoints the differential equation of the transformed time $\tilde{t}_j(\cdot)$ to the Hamiltonian.

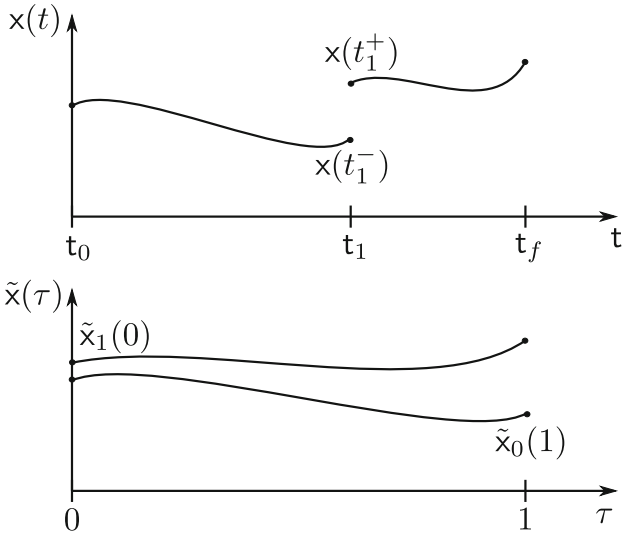


Fig. 4.4 Subfigure **a**: Original trajectory with one switching at t_1 with a state discontinuity. Subfigure **b**: Stacked trajectories $\tilde{x}_0(\tau)$ and $\tilde{x}_1(\tau)$

The transversality conditions of the transformed problem are obtained by applying (4.104) and (4.105) to each *stacked* subinterval from the hybrid trajectory. Stacking means that all subintervals from the hybrid trajectory are treated as if they evolve simultaneously as illustrated in Fig. 4.4.

Applying the principle of stacked transversality conditions to (4.115) yields

$$\tilde{\boldsymbol{\lambda}}_{j-1}(1) = \left(\frac{\partial \boldsymbol{\psi}}{\partial \tilde{\mathbf{x}}_{j-1}(1)}\right)^T (\tilde{\mathbf{x}}_{j-1}(1), \ldots) \cdot \boldsymbol{\pi}_j$$
$$\tilde{\boldsymbol{\lambda}}_{j}(0) = -\left(\frac{\partial \boldsymbol{\psi}}{\partial \tilde{\mathbf{x}}_{j}(0)}\right)^T (\tilde{\mathbf{x}}_{j}(0), \ldots) \cdot \boldsymbol{\pi}_j$$

for all $j = 1, \ldots, N_{swt}$ where $\boldsymbol{\pi}_j \in \mathbb{R}^{N_x}$ are Lagrange multipliers. Using the derivatives

$$\frac{\partial \boldsymbol{\psi}}{\partial \tilde{\mathbf{x}}_{j-1}(1)} = \frac{\partial \boldsymbol{\varphi}}{\partial \tilde{\mathbf{x}}_{j-1}(1)}(\tilde{\mathbf{x}}_{j-1}(1), \tilde{\mathbf{x}}_j(0)) = -\mathbf{I} - \frac{\partial \boldsymbol{\delta}_{(\tilde{q}_{j-1}(1), \tilde{q}_j(0))}}{\partial \tilde{\mathbf{x}}_{j-1}(1)}(\tilde{\mathbf{x}}_{j-1}(1))$$
$$\frac{\partial \boldsymbol{\psi}}{\partial \tilde{\mathbf{x}}_{j}(0)} = \frac{\partial \boldsymbol{\varphi}}{\partial \tilde{\mathbf{x}}_{j}(0)}(\tilde{\mathbf{x}}_{j-1}(1), \tilde{\mathbf{x}}_j(0)) = \mathbf{I}$$

yields the stacked transversality conditions

$$\boldsymbol{\lambda}_{j-1}(1) = -\left(\mathbf{I} + \frac{\partial \boldsymbol{\delta}_{(\tilde{q}_{j-1}(1), \tilde{q}_j(0))}}{\partial \tilde{\mathbf{x}}_{j-1}(1)}(\tilde{\mathbf{x}}_{j-1}(1))\right)^T \cdot \boldsymbol{\pi}_j \tag{4.116}$$

$$\boldsymbol{\lambda}_j(0) = -\boldsymbol{\pi}_j. \tag{4.117}$$

An important feature is obtained if (4.117) is inserted into (4.116), which yields

$$\boldsymbol{\lambda}_{j-1}(1) = \boldsymbol{\lambda}_j(0) + \left(\frac{\partial \boldsymbol{\delta}_{(\tilde{q}_{j-1}(1),\tilde{q}_j(0))}}{\partial \tilde{\mathbf{x}}_{j-1}(1)}\right)^T (\tilde{\mathbf{x}}_{j-1}(1)) \cdot \boldsymbol{\lambda}_j(0). \tag{4.118}$$

The Eq. (4.118) is known as the *switching condition* for a state jump. One obtains the switching condition for a state jump in the original time domain using the correspondences $\tilde{\boldsymbol{\lambda}}_{j-1}(1) = \boldsymbol{\lambda}(t_j^-)$, $\tilde{\boldsymbol{\lambda}}_j(0) = \boldsymbol{\lambda}(t_j^+)$, and $\tilde{\mathbf{x}}_{j-1}(1) = \mathbf{x}(t_j^-)$ as

$$\boldsymbol{\lambda}(t_j^-) = \boldsymbol{\lambda}(t_j^+) + \left(\frac{\partial \boldsymbol{\delta}_{(q(t_j^-),q(t_j^+))}}{\partial \mathbf{x}(t_j^-)}\right)^T (\mathbf{x}(t_j^-)) \cdot \boldsymbol{\lambda}(t_j^+). \tag{4.119}$$

Clearly, a system without state jumps leads to continuous costates. The transversality condition at the final time $\tilde{t}_{N_{exe}-1}(1)$ is completely analogous to the transversality condition for the continuous optimal control problem, as can be easily shown.

The Hamiltonian before and after the transitions is identical

$$\mathcal{H}(\mathbf{x}(t_j^-), q(t_j^-), \boldsymbol{\lambda}(t_j^-), \mathbf{u}(t_j^-)) = \mathcal{H}(\mathbf{x}(t_j^+), q(t_j^+), \boldsymbol{\lambda}(t_j^+), \mathbf{u}(t_j^+)),$$

since we are dealing with autonomous problems only.

After stacking, we can conclude that the costates $\boldsymbol{\lambda}(\cdot)$ are allowed to be discontinuous at the switching times t_j, while in the time interval $[t_j, t_{j+1})$ the adjoint differential equation must be satisfied

$$\dot{\boldsymbol{\lambda}}(t) = -\frac{\partial \mathcal{H}}{\partial \mathbf{x}(t)}(\mathbf{x}(t), q(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)), \quad \text{for a.e. } t \in [t_j, t_{j+1}), \quad j = 0, 1, \ldots, N_{exe} - 1,$$

where the Hamiltonian is defined as

$$\mathcal{H}(\mathbf{x}(t), q(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)) := \boldsymbol{\lambda}^T(t)\mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)). \tag{4.120}$$

For all arcs of the hybrid trajectory, the Hamiltonian minimization condition holds for both, the continuous controls $\mathbf{u}(\cdot)$ and discrete state $q(\cdot)$, in the classical sense of Pontryagin

$$(q^*(t), \mathbf{u}^*(t)) = \arg \min_{\mathbf{u}(t) \in \hat{\mathcal{U}}(q(t),t),\, q(t) \in \hat{\mathcal{Q}}} \mathcal{H}(\mathbf{x}^*(t), q(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)).$$

Collecting all results, we obtain the first-order necessary conditions for hybrid optimal control problems.

4.4.1 Necessary Conditions for Switched Optimal Control Problems Without State Jumps

At first sight, the necessary conditions derived for optimal control problems of switched systems without jumps in the state trajectory are very similar to the necessary conditions for purely continuous systems and are documented among others in the works Sussmann [53], Riedinger et al. [47, 48], and Passenberg et al. [43]. This is not surprising. As demonstrated in Sect. 3.2.6, a switched system can be reformulated as a conventional system. The only difference is, that the set of feasible constraints $\hat{\mathcal{U}}(q(t), t)$ is no longer compact and convex. This fact makes it necessary to split the hybrid trajectory into arcs where the discrete state $q(\cdot)$ changes its location.

Decomposing the original SOCP (4.99)–(4.103) without state jumps into the new time domain, applying the classical Pontryagin's theorem to each arc, and composing the transformed first-order necessary conditions back to the original time domain yields the HMP for (4.99)–(4.103) without state jumps. Let us summarize the results in the following theorem:

Theorem 4.8 (First-order Necessary Conditions for Switched Optimal Control Problems without State Jumps) *Let* $\mathbf{u}^*(\cdot) \in L^\infty([t_0, t_f], \mathsf{U})$, $\mathbf{x}^*(\cdot) \in \mathcal{AC}^\infty([t_0, t_f], \mathsf{X})$, *and* $q^*(\cdot) \in L^\infty([t_0, t_f], \hat{\mathcal{Q}})$ *be an optimal tuple for the SOCP (4.99)–(4.103) without state jumps, i.e.,* $\boldsymbol{\varphi}(\mathbf{x}^*(t_j^-), \mathbf{x}^*(t_j^+)) = \mathbf{x}^*(t_j^+) - \mathbf{x}^*(t_j^-) = \mathbf{0}$, *over the fixed time interval* $[t_0, t_f]$. *Let us assume that the cost functional is given as Mayer problem, which results in the Hamiltonian defined as (4.120). Then, there exist absolutely continuous costates* $\boldsymbol{\lambda}(\cdot) \in \mathcal{AC}^\infty([t_0, t_f], \mathbb{R}^{N_x})$ *satisfying the nontrivial solution* $\boldsymbol{\lambda}(t) \neq \mathbf{0}$, *for which the following conditions hold:*

1. *the states* $\mathbf{x}^*(\cdot)$ *and the costates* $\boldsymbol{\lambda}(\cdot)$ *satisfy the canonical equations with respect to the Hamiltonian (4.120)*

$$\begin{aligned} \dot{\mathbf{x}}^*(t) &= \frac{\partial \mathcal{H}}{\partial \boldsymbol{\lambda}}(\mathbf{x}^*(t), q^*(t), \boldsymbol{\lambda}(t), \mathbf{u}^*(t)) \\ &= \mathbf{f}_{q^*(t)}(\mathbf{x}^*(t), \mathbf{u}^*(t)), \end{aligned} \tag{4.121}$$

$$\begin{aligned} \dot{\boldsymbol{\lambda}}(t) &= -\frac{\partial \mathcal{H}}{\partial \mathbf{x}}(\mathbf{x}^*(t), q^*(t), \boldsymbol{\lambda}(t), \mathbf{u}^*(t)) \\ &= -\left(\frac{\partial \mathbf{f}_{q^*(t)}}{\partial \mathbf{x}}\right)^T (\mathbf{x}^*(t), \mathbf{u}^*(t)) \cdot \boldsymbol{\lambda}(t) \end{aligned} \tag{4.122}$$

   for almost every $t \in [t_j, t_{j+1})$, $\quad j = 0, 1, \ldots, N_{exe} - 1$ *with the boundary conditions* $\mathbf{x}^*(t_0) = \mathbf{x}_0$ *and* $\mathbf{x}^*_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f$;
2. *the Hamiltonian minimum condition*

$$(q^*(t), \mathbf{u}^*(t)) = \arg \min_{\mathbf{u}(t) \in \hat{\mathcal{U}}(q(t), t),\, q(t) \in \hat{\mathcal{Q}}} \mathcal{H}(\mathbf{x}^*(t), q(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)) \tag{4.123}$$

   holds for almost every $t \in [t_j, t_{j+1})$, $\quad j = 0, 1, \ldots, N_{exe} - 1$;

3. *at the final time t_f the transversality condition evaluated on the terminal Lagrangian*

$$\boldsymbol{\lambda}_{[\mathcal{I}_f^c]}(t_f) = \frac{\partial m}{\partial \mathbf{x}_{[\mathcal{I}_f^c]}(t_f)}(\mathbf{x}^*(t_f)) \tag{4.124}$$

is fulfilled; and
4. *for an externally forced switching at time instance t_j, the following jump conditions are satisfied*

$$\boldsymbol{\lambda}(t_j^-) = \boldsymbol{\lambda}(t_j^+) \tag{4.125}$$

$$\mathcal{H}(\mathbf{x}^*(t_j^-), q^*(t_j^-), \boldsymbol{\lambda}(t_j^-), \mathbf{u}^*(t_j^-)) = \mathcal{H}(\mathbf{x}^*(t_j^+), q^*(t_j^+), \boldsymbol{\lambda}(t_j^+), \mathbf{u}^*(t_j^+)). \tag{4.126}$$

△

Proof The proof can be found in Dmitruk and Kaganovich [20]. A similar result is obtained in Shaikh [52]. □

A crucial difference to the first-order necessary conditions formulated in the PMP is that the Hamiltonian now is minimized for a.e. fixed $t \in [t_0, t_f]$ by both, continuous-valued controls and discrete state.

4.4.2 Necessary Conditions for Switched Optimal Control Problems with State Jumps

To account for state jumps, the trajectories of the original SOCP (4.99)–(4.103) with state jumps must be decomposed into the hybrid trajectory in the new time domain in order to apply the stacking procedure. Then, applying the classical Pontryagin theorem to each arc and composing the transformed first-order necessary conditions back to the original time domain yields the HMP for (4.99)–(4.103) with state jumps.

Optimal control problems of switched systems with state jumps have the challenge that the costates are discontinuous too. This requires the exact knowledge of the number of switching arcs and needs some additional work in the evaluation of the jump conditions.

We summarize the results in the following theorem:

Theorem 4.9 (First-order Necessary Conditions for Switched Optimal Control Problems with State Jumps) *Let $\mathbf{u}^*(\cdot) \in L^\infty([t_0, t_f], \mathsf{U})$, $\mathbf{x}^*(\cdot) \in \mathcal{AC}^\infty([t_0, t_f], \mathsf{X})$, and $q^*(\cdot) \in L^\infty([t_0, t_f], \hat{\mathcal{Q}})$ be an optimal tuple for the SOCP (4.99)–(4.103) with state jumps, i.e., $\boldsymbol{\varphi}(\mathbf{x}^*(t_j^-), \mathbf{x}^*(t_j^+)) = \mathbf{x}^*(t_j^+) - \mathbf{x}^*(t_j^-) - \boldsymbol{\delta}_{(q(t_j^-), q(t_j^+))}(\mathbf{x}^*(t_j^-)) = \mathbf{0}$, over the fixed time interval $[t_0, t_f]$. Let us assume that the cost functional is given as Mayer problem, which results in the Hamiltonian defined as (4.120). Then, there*

exist piecewise absolutely continuous costates $\boldsymbol{\lambda}(\cdot) \in \hat{\mathcal{AC}}^{\infty}([t_0, t_f], \mathbb{R}^{N_x})$ *satisfying the nontrivial solution* $\boldsymbol{\lambda}(t) \neq \mathbf{0}$*, for which the following conditions hold:*

1. *the states* $\mathbf{x}^*(\cdot)$ *and the costates* $\boldsymbol{\lambda}(\cdot)$ *satisfy the canonical equations (4.121) and (4.122) with respect to the Hamiltonian (4.120) for almost every* $t \in [t_j, t_{j+1})$, $j = 0, 1, \ldots, N_{exe} - 1$ *with the boundary conditions* $\mathbf{x}^*(t_0) = \mathbf{x}_0$ *and* $\mathbf{x}^*_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f$*;*
2. *the Hamiltonian minimum condition (4.123) holds for almost every* $t \in [t_j, t_{j+1})$, $j = 0, 1, \ldots, N_{exe} - 1$*;*
3. *at the final time* t_f *the transversality condition (4.124) is fulfilled; and*
4. *for an externally forced switching at time instance* t_j*, the following jump conditions must be satisfied*

$$\boldsymbol{\lambda}(t_j^-) = \boldsymbol{\lambda}(t_j^+) + \left(\frac{\partial \boldsymbol{\delta}_{(q(t_j^-), q(t_j^+))}}{\partial \mathbf{x}^*(t_j^-)} \right)^T (\mathbf{x}^*(t_j^-)) \cdot \boldsymbol{\lambda}(t_j^+) \tag{4.127}$$

$$\mathcal{H}(\mathbf{x}^*(t_j^-), q^*(t_j^-), \boldsymbol{\lambda}(t_j^-), \mathbf{u}^*(t_j^-)) = \mathcal{H}(\mathbf{x}^*(t_j^+), q^*(t_j^+), \boldsymbol{\lambda}(t_j^+), \mathbf{u}^*(t_j^+)). \tag{4.128}$$

△

Proof The proof is a simple consequence of the proof given in Dmitruk and Kaganovich [20]. □

4.4.3 *Revisited: Necessary Conditions for a State Constrained Optimal Control Problem*

Theorem 4.5 states first-order necessary conditions for state constrained optimal control problems which are fairly general. Especially, the jump conditions for the costates are difficult to understand and to apply in practice. The re-interpretation of the state constrained optimal control problem as hybrid optimal control problem provides no further information but let us more easily derive the jump conditions for the costates. For the sake of simplicity, let us consider a simple example of an optimal control problem with only one pure state constraint. This example is casted to a hybrid optimal control problem. This approach is naturally motivated if one thinks that the system autonomously execute a transition, when the second state of the system encounters a switching manifold of the type

$$\mathbf{x}(t_j) \in C_{(q(t^-), q(t^+))}(t) = \left\{ \mathbf{x}(t) \;\middle|\; c_x(\mathbf{x}(t)) = 0, \quad \mathbf{x}(t) \in \mathsf{X} \right\}.$$

We assume again that $c_x(\cdot)$ satisfies the rank condition (4.56) on the boundary interval such that $c_x(\cdot)$ is implicitly controllable by $\mathbf{u}(\cdot)$ via the system differential equation.

We assume a system of two states where only the second state admits a constraint, i.e., $c_x(x_2(t))$. The state constrained system is then casted to a hybrid system, where

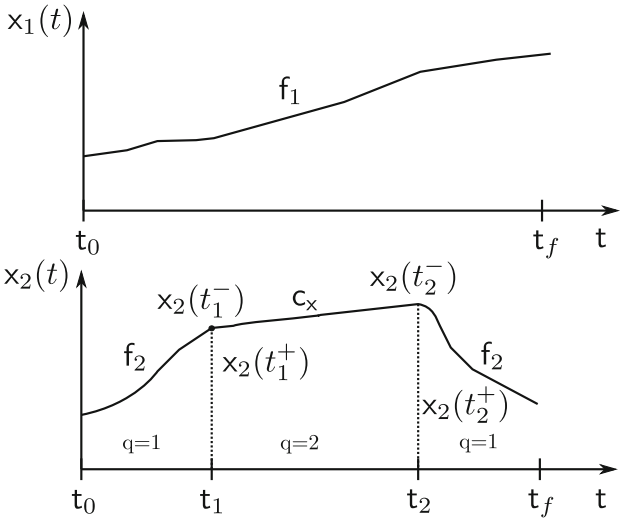


Fig. 4.5 Exemplary state trajectories $x_1(\cdot)$ and $x_2(\cdot)$ of the switched system, where the second state is constrained in the time interval $t \in [t_1, t_2)$

the time evolution can be decomposed into two unconstrained arcs and one constrained arc. This is exemplarily depicted for the case that $\mathbf{x}_1^{[2]}$ is constrained to a ramp as shown in Fig. 4.5. We obtain then a hybrid trajectory given as

$$\begin{aligned}
\dot{\mathbf{x}}_0(t) &= \mathbf{f}_{q_0(t)}(\mathbf{x}_0(t), \mathbf{u}_0(t)), \quad \text{for a.e. } t \in [t_0, t_1) \\
\dot{\mathbf{x}}_1(t) &= \mathbf{f}_{q_1(t)}(\mathbf{x}_1(t), \mathbf{u}_1(t)), \quad \text{for a.e. } t \in [t_1, t_2) \\
\dot{\mathbf{x}}_2(t) &= \mathbf{f}_{q_2(t)}(\mathbf{x}_2(t), \mathbf{u}_2(t)), \quad \text{for a.e. } t \in [t_2, t_f],
\end{aligned}$$

where $[q_0, q_1, q_2] = [1, 2, 1]$. The constrained state $\mathbf{x}_1^{[2]}$ is implicitly described by the equation

$$c_x(\mathbf{x}_1) = 0, \text{ for all } t \in [t_1, t_2).$$

The transformed Mayer problem for the time domain $\tau \in [0, 1]$ is then given as

$$\phi\left(q^*(\cdot), \mathbf{u}^*(\cdot)\right) = \min_{q(\cdot)\in\mathcal{Q},\ \mathbf{u}(\cdot)\in\mathcal{U}(q(\cdot))} \phi(q(\cdot), \quad \mathbf{u}(\cdot)) = m\left(\tilde{\mathbf{x}}_2^*(1)\right) \tag{4.129}$$

$$\frac{\mathrm{d}\tilde{\mathbf{x}}_j}{\mathrm{d}\tau} = \varsigma_j \mathbf{f}_{\tilde{q}_j(\tau)}(\tilde{\mathbf{x}}_j(\tau), \tilde{\mathbf{u}}_j(\tau)), \qquad j = 0, \ldots, 2, \text{ for all } \tau \in [0, 1] \tag{4.130}$$

$$c_x(\tilde{\mathbf{x}}_1) = 0, \qquad \text{for all } \tau \in [0, 1] \tag{4.131}$$

$$\tilde{\mathbf{x}}_0(0) = \mathbf{x}_0 \tag{4.132}$$

$$\tilde{\mathbf{x}}_2^{[\mathcal{I}_f]}(1) = \mathbf{x}_f \tag{4.133}$$

$$\boldsymbol{\varphi}(\tilde{\mathbf{x}}_{j-1}(1), \tilde{\mathbf{x}}_j(0)) = \tilde{\mathbf{x}}_j(0) - \tilde{\mathbf{x}}_{j-1}(1) = \mathbf{0}, \qquad j = 1, 2 \tag{4.134}$$

$$\dot{\tilde{t}}_j(\tau) = \varsigma_j, \qquad j = 0, \ldots, 2, \text{ for all } \tau \in [0, 1] \tag{4.135}$$

$$\tilde{t}_0(0) = t_0 \tag{4.136}$$

$$\tilde{t}_2(1) = t_f \tag{4.137}$$

$$\tilde{t}_{j-1}(1) - \tilde{t}_j(0) = 0, \qquad j = 1, 2. \tag{4.138}$$

The general boundary conditions according to the transformed problem is given as

$$\boldsymbol{\psi}(\mathbf{x}_0(0), \ldots, \mathbf{x}_2(1), \tilde{t}_0(0), \ldots, \tilde{t}_2(1)) = \begin{bmatrix} \tilde{\mathbf{x}}_0(0) - \mathbf{x}_0 \\ \boldsymbol{\varphi}(\tilde{\mathbf{x}}_{j-1}(1), \tilde{\mathbf{x}}_j(0)), \ j = 1, 2 \\ \tilde{\mathbf{x}}_2^{[\mathcal{I}_f]}(1) - \mathbf{x}_f \\ c_x(\tilde{\mathbf{x}}_1) \\ \tilde{t}_0(0) - t_0 \\ \tilde{t}_{j-1}(1) - \tilde{t}_j(0), \ j = 1, 2 \\ \tilde{t}_2(1) - t_f \end{bmatrix} = \mathbf{0}. \tag{4.139}$$

We adjoint the state constraint directly to the Hamiltonian

$$\begin{aligned} &\tilde{\mathcal{H}}\left(\tilde{\mathbf{x}}(\tau), \tilde{\boldsymbol{\lambda}}(\tau), \tilde{\rho}(\tau), \tilde{\gamma}(\tau), \tilde{\mathbf{u}}(\tau), \varsigma\right) \\ &= \sum_{j=0}^{2} \varsigma_j \cdot \left[\tilde{\boldsymbol{\lambda}}_j^T(\tau) \mathbf{f}_{\tilde{q}_j(\tau)}(\tilde{\mathbf{x}}_j(\tau), \tilde{\mathbf{u}}_j(\tau)) + \tilde{\rho}_j(\tau)\right] + \tilde{\gamma}(\tau) \cdot c_x(\tilde{\mathbf{x}}_1(\tau)), \qquad \text{for a.e. } \tau \in [0, 1]. \end{aligned}$$

Then, the adjoint differential equations are

$$\frac{\mathrm{d}\tilde{\boldsymbol{\lambda}}_j}{\mathrm{d}\tau} = -\varsigma_j \cdot \left(\frac{\partial \mathbf{f}_{\tilde{q}(\tau)}}{\partial \tilde{\mathbf{x}}_j}\right)^T (\tilde{\mathbf{x}}_j(\tau), \tilde{\mathbf{u}}_j(\tau)) \tilde{\boldsymbol{\lambda}}_j(\tau), \quad j \in \{0, 2\}, \qquad \text{for a.e. } \tau \in [0, 1]$$

$$\frac{\mathrm{d}\tilde{\boldsymbol{\lambda}}_1}{\mathrm{d}\tau} = -\varsigma_1 \cdot \left(\frac{\partial \mathbf{f}_{\tilde{q}(\tau)}}{\partial \tilde{\mathbf{x}}_1}\right)^T (\tilde{\mathbf{x}}_1(\tau), \tilde{\mathbf{u}}_1(\tau)) \tilde{\boldsymbol{\lambda}}_1(\tau) + \tilde{\gamma}(\tau) \frac{\partial c_x}{\partial \tilde{\mathbf{x}}_1}(\tilde{\mathbf{x}}_1(\tau)), \qquad \text{for a.e. } \tau \in [0, 1]$$

$$\frac{\mathrm{d}\tilde{\rho}_j}{\mathrm{d}\tau} = 0, \quad j = 0, \ldots, 2, \qquad \forall \tau \in [0, 1].$$

Since we adjoined the state constraint directly, we must specify an additional point constraint anywhere on the constrained arc. We choose the entry point

$$c_x(\tilde{\mathbf{x}}_1(0)) = 0.$$

Together with the constraint (4.134) the application of the general transversality conditions (4.104)–(4.105) to (4.139) yields

$$\begin{aligned}\boldsymbol{\lambda}_0(1) &= -\boldsymbol{\alpha}_0\\ \boldsymbol{\lambda}_1(0) &= -\boldsymbol{\alpha}_0 - \frac{\partial c_x}{\partial \tilde{\mathbf{x}}_1(0)}(\tilde{\mathbf{x}}_1(0)) \cdot \pi,\end{aligned}$$

where $\pi \in \mathbb{R}, \pi \geq 0$ is a Lagrange multiplier for the state constraint (4.131) and $\boldsymbol{\alpha}_0 \in \mathbb{R}^2$ are Lagrange multipliers for constraint (4.134) with $j = 0$. Thus, the transversality conditions give the following jump condition for the costate:

$$\boldsymbol{\lambda}_0(1) = \boldsymbol{\lambda}_1(0) + \frac{\partial c_x}{\partial \tilde{\mathbf{x}}_1(0)}(\tilde{\mathbf{x}}_1(0)) \cdot \pi.$$

Back transformation into the original time domain yields the first-order necessary conditions from Sect. 4.2.2 for a jump of the costate at the entry point of the state constraint. The derivation is fully analogously for a jump at the exit point. If the jump shall occur in the interior interval of the state constraint, an additional phase of the hybrid trajectory must be added. Then, the derivation is done according to the same principle.

4.5 Existence

Necessary conditions can be useless or even misleading unless we know that a solution of an optimal control problem exists. But how can we ensure the existence?

We state the existence theorems for continuous optimal control problems and switched optimal control problems with a weak compactness assumption for the control space. In so doing, let us first consider the Mayer problem

$$\phi\left(\mathbf{u}^*(\cdot)\right) = \min_{\mathbf{u}(\cdot)\in L^\infty([t_0,t_f]),\mathsf{U})} \phi(\mathbf{u}(\cdot)) = m(\mathbf{x}^*(t_f)) \tag{4.140}$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \qquad \text{for a.e. } t \in [t_0, t_f] \tag{4.141}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{4.142}$$

$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f \tag{4.143}$$

$$\mathbf{c}_u(\mathbf{u}(t)) \leq \mathbf{0} \tag{4.144}$$

$$\mathbf{c}_x(\mathbf{x}(t)) \leq \mathbf{0}, \tag{4.145}$$

where $\mathbf{x}(\cdot) \in \mathcal{AC}^\infty([t_0, t_f], \mathsf{X})$. The Mayer term is defined as $m : \hat{\mathcal{X}} \to \mathcal{S}$ and $\mathcal{S} \subseteq \mathsf{X}$ is closed.

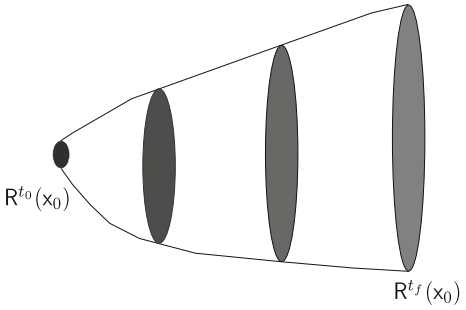


Fig. 4.6 Evolution of the reachable set $\mathcal{R}^t(\mathbf{x}_0)$ over the time span $t_0 \le t \le t_f$

Let us define the reachable sets for the Mayer problem as

$$\mathcal{R}^t(\mathbf{x}_0) := \left\{ \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \;\middle|\; \mathbf{u}(t) \in \hat{\mathcal{U}}(t) \right\} \subset \mathsf{X}, \quad \forall t \in [t_0, t_f]. \tag{4.146}$$

The reachable sets are illustrated in Fig. 4.6. The reachable sets are slabs for each $t \in [t_0, t_f]$ which contains all points $\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))$ that can be reached by any controls taken from the admissible control set $\hat{\mathcal{U}}(t)$.

For convenience of the coming theorems, let us define the set

$$\mathcal{M} = \left\{ (\mathbf{x}(t), \mathbf{u}(t)) \;\middle|\; \mathbf{x}(t) \in \hat{\mathcal{X}}(t),\ \mathbf{u}(t) \in \hat{\mathcal{U}}(t) \right\} \subset \mathbb{R}^{N_x + N_u}$$

and let the function $\mathbf{f}(\cdot)$ be defined on $\mathcal{M}$.

With these definitions, Cesari [13] stated the following existence theorem:

Theorem 4.10 (Filippov's Existence Theorem for Mayer Problems (cf. Cesari [13])) *Let $\hat{\mathcal{X}}(t)$ and $\mathcal{S}$ be closed and assume that for every $N \ge 0$ there exists a set $\mathcal{M}_N = \{(\mathbf{x}(t), \mathbf{u}(t)) \in \mathcal{M} \mid |\mathbf{x}(t)| \le N\}$ that is compact. Also, let the endpoint functional $m(\cdot)$ be lower semicontinuous on $\mathcal{S}$, and the function $\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))$ continuous on $\mathcal{M}$. Let us further assume that the reachable sets $\mathcal{R}^t(\mathbf{x}_0)$ are convex for a.e. $t \in [t_0, t_f]$. Then, the problem (4.140)–(4.145) has an absolute minimum on all feasible pairs $(\mathbf{x}(\cdot), \mathbf{u}(\cdot))$.*

△

Proof The proof is given in Cesari [13]. □

Remark 4.7 The underlined assumption in Theorem 4.10 can be replaced by one of the following alternatives (cf. Cesari [13]):

- there is a constant $C \ge 0$ such that $x_1 f_1 + \ldots + x_{N_x} f_{N_x} \le C \cdot \left(|\mathbf{x}(t)|^2 + 1\right)$ for all $(\mathbf{x}(t), \mathbf{u}(t)) \in \mathcal{M}$ and $t \in [t_0, t_f]$;

- there is a constant $C \geq 0$ such that $|\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))| \leq C \cdot (|\mathbf{x}(t)| + 1)$ for all $(\mathbf{x}(t), \mathbf{u}(t)) \in \mathcal{M}$ and $t \in [t_0, t_f]$; or
- there is a locally integrable scalar function $M(\cdot)$ such that $|\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))| \leq M(t) \cdot (|\mathbf{x}(t)| + 1)$ and $M(t) \geq 0$ for all $(\mathbf{x}(t), \mathbf{u}(t)) \in \mathcal{M}$ and $t \in [t_0, t_f]$.

Remark 4.8 A lower semicontinuous function is locally bounded below.

The existence result from above can be extended for Lagrange and Bolza-type problems as shown in Cesari [13], which requires to redefine the reachable sets accordingly.

For hybrid optimal control problems the existence statement is a trivial consequence from the previous results by using the stacking argumentation. In so doing, let us consider a HOCP formulated as Mayer problem:

$$\phi\left(q^*(\cdot), \mathbf{u}^*(\cdot)\right) = \min_{q(\cdot)\in\mathcal{Q},\, \mathbf{u}(\cdot)\in\mathcal{U}(q(\cdot))} \phi(q(\cdot), \mathbf{u}(\cdot)) = m(\mathbf{x}^*(t_f)) \tag{4.147}$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)), \qquad \text{for a.e. } t \in [t_0, t_f] \tag{4.148}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{4.149}$$

$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f \tag{4.150}$$

$$\boldsymbol{\varphi}(\mathbf{x}(t_j^-), \mathbf{x}(t_j^+)) = \mathbf{x}(t_j^+) - \mathbf{x}(t_j^-) + \boldsymbol{\delta}_{(q^-,q^+)}(\mathbf{x}(t_j^-)) = \mathbf{0}, \quad t_j \in \Theta_t \tag{4.151}$$

$$\mathbf{c}_x(\mathbf{x}(t)) \leq \mathbf{0} \tag{4.152}$$

$$\mathbf{x}(t_j^-), \mathbf{x}(t_j^+) \in \mathcal{S}_j, \quad j = \{1, 2, \ldots, N_{swt}\}. \tag{4.153}$$

Then, endpoint functional

$$\phi(\mathbf{x}(t_f), \mathbf{x}(t_j^-), \mathbf{x}(t_j^+)) = m(\mathbf{x}(t_f)) + \sum_{j=1}^{N_{swt}} \boldsymbol{\pi}_j^T \boldsymbol{\varphi}(\mathbf{x}(t_j^-), \mathbf{x}(t_j^+))$$

can be decomposed into N_{exe} endpoint functionals

$$\begin{aligned}
\phi(\mathbf{x}(t_1^-), \mathbf{x}(t_1^+)) &= \boldsymbol{\pi}_1^T \boldsymbol{\varphi}(\mathbf{x}(t_1^-), \mathbf{x}(t_1^+)) \\
&\vdots \\
\phi(\mathbf{x}(t_{N_{swt}}^-), \mathbf{x}(t_{N_{swt}}^+)) &= \boldsymbol{\pi}_{N_{swt}}^T \boldsymbol{\varphi}(\mathbf{x}(t_{N_{swt}}^-), \mathbf{x}(t_{N_{swt}}^+)) \\
\phi(\mathbf{x}(t_f)) &= m(\mathbf{x}(t_f)).
\end{aligned}$$

Since the jump function $\boldsymbol{\delta}_{(q^-,q^+)}(\cdot)$ is assumed to be continuously differentiable with respect to $\mathbf{x}(\cdot)$, the continuity assumption on the endpoint functionals with respect to their target sets still holds. Also, we assume that $\mathbf{f}_q(\cdot)$ are continuously defined on $\mathcal{M}$. Then, Theorem 4.10 states for each subproblem of (4.147)–(4.153), the existence of an optimal control.

4.6 Bibliography

A good survey paper about optimal control and the some historical background is given by Sargent [50]. Some historical key points: optimal control theory began nearly 380 years ago with the first attempt of calculus of variations and the main actors Galileo, Newton, Euler, to name just a few. In 1755, Lagrange introduced the first analytical approach based on variations of the optimal curve and used undetermined multipliers; later called as "Lagrange multiplier". The subject "calculus of variation" was born. Meanwhile Hamilton [42] adopted the variational principle to the equation of mechanics and introduced the "principal function of motion of a system"—now known as the Hamiltonian function. Hamilton [28, 29] expressed his principle in terms of a pair of partial differential equations, but Jacobi showed in 1838 that it could be more compactly written as the "Hamilton-Jacobi" equation (see Goldstine [26] for more information). Then we make a jump after the end of World War II. At this time there was a great interest in dealing with minimum-time interception problems for fighter aircraft. Due to the increased speed of aircraft, nonlinear terms no longer could be neglected. However, linearisation was not the preferred method. Many famous mathematicians were seeking for a general solution method for this problem type. For instance, Hestenes [31, 32] wrote two famous RAND research memoranda. In particular, Hestenes memorandum [31] includes an early formulation of what later became known as the maximum principle. Hestenes deep affinity to the calculus of variation led him miss the point of radical rethinking of his major problem: the assumption of unbounded controls. Pontryagin et al. [46] realized this problem for the air force and established as first author necessary conditions for control problems with control constraints and hence introduced the famous "maximum principle". He used a function $\mathcal{H}(\cdot)$, which is not a classical Hamiltonian. Later, Clarke [16] termed this function "pseudo-Hamiltonian". However, the canonical equation from Hamilton holds also for this pseudo-Hamiltonian, if and only if the Euler–Lagrange equation holds. In fact, Pontryagin laid the foundation for many other research directions and his maximum principle still holds even for modern hybrid optimal control problems. A good history overview about the maximum principle is presented by Pesch and Plail [44, 45]. Another crucial moment is the development of nonsmooth analysis during the 1970 s and 1980 s. Nonsmooth analysis has triggered a renew interest in optimal control problems and brought new solutions to old problems. One of the main protagonist in this research field is Clarke [16].

There is a wealth of excellent textbooks on the subject of optimal control theory. A first introduction to the theory of variational analysis and a brief description of some basic algorithms is given in the textbook Kirk [37]. Much more detailed are the derivations given in Bryson and Ho [10]. A very good introduction to optimal control theory is given by Liberzon [40], which deals with variational calculus, Pontryagin's maximum principle and its sketch of proof, and some advanced topics. A concise but theoretical treatise are given by Vinter [55], Ioffe and Tihomirov [35], and Girsanov [25]. Also, a concise introduction to optimal control with a strong foundation on functional analysis is given by Clarke [14]. The complex concepts of measurable

and absolutely continuous functions are described in an accessible way. A more technical description of these concepts are presented in Adams and Fournier [1].

The central role of value functions and Hamilton–Jacobi equations in the calculus of variations was recognized by Caratheodory [12]. Similar ideas reappeared under the name of Dynamic Programming in the work of Richard Bellman (Bellman [4]) and became a standard tool for the synthesis of feedback controls for discrete-time systems. The Hamilton–Jacobi–Bellman equation is used to derive necessary conditions for optimal control of hybrid and switched systems. In Riedinger and Kratz [47], dynamic programming argumentation is used to derive the set of necessary conditions for a hybrid optimal control problem. The result is similar to the stacking procedure but implies in silence that the value function belongs to $\mathcal{C}^1$.

A good overview of necessary conditions for optimal control problems with state inequality constraints can be found in Hartl et al. [30]. Further references are Jacobson et al. [36], Biswas and De Pinho [5, 6]. Graichen and Petit [27] proposed a method to transform a constrained to an unconstrained optimal control problem. The method employs an input-output linearization as coordinate transformation of the state constraints. The transformed state constraints appears then linear but are none differentiable on the state-boundaries which requires the assumption that the OCP steers the system only close to the state constraints. Most proofs of the maximum principle for state inequality constraints are not easily accessible. Dmitruk [19] discussed a sliding variation approach originally proposed by Dubovitskii and Milyutin [21] to proof the maximum principle for this problem class on the level of well-known results of functional analysis.

Necessary conditions for the hybrid case were derived in Sussmann [53] for a very general class of systems. More specific results for systems with absolutely continuous states can be found in the works Shaikh [52] and Xu [57] and for systems with discontinuities in the state trajectory in Riedinger et al. [47, 48], Passenberg et al. [43], and Schori et al. [51]. Clarke and De Pinho [15] presented a theory which plays an important role for the derivation of necessary conditions for control problems with control-state constraints. This synthesis approach is called the *nonsmooth analysis approach*. Second-order necessary conditions for OCPs with state inequality constraints are derived by Bonnans and Hermant [8, 9] and Hoehener [34]. Sufficient conditions for optimality are given in Boltyanski and Poznyak [7].

In the context of autonomous systems, HOCPs with jumps in the state were treated in X. Xu [56] and S.A. Attia [49]. A related case are the HOCPs where no jumps in the state occur but additional costs are added to the objective function for every switching. Necessary conditions for this case are stated in Garavello and Piccoli [23].

The existence of optimal controls is treated in-depth in Cesari [13] and Clarke [14]. Further recommended references are Knowles [38], Lee and Markus [39], and Filippov [22].

References

1. Adams RA, Fournier JJ (2003) Sobolev spaces, vol 140, 2nd edn. Academic Press
2. Azbelev N, Rakhmatullina L (2007) Introduction to the theory of functional differential equations: methods and applications. Contemporary mathematics and its applications. Hindawi
3. Bardi M, Capuzzo-Dolcetta I (1997) Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations. Springer Science & Business Media
4. Bellman RE (2003) Dynamic programming, republication of the edition published by Princeton University Press (1957) edn. Dover Publications
5. Biswas HA, De Pinho MDR (2011) Necessary conditions for optimal control problems with and without state constraints: a comparative study. WSEAS Trans Syst Control 6(6):217–228
6. Biswas MHA (2013) Necessary conditions for optimal control problems with state constraints: theory and applications. PhD thesis, University of Porto, Nov 2013
7. Boltyanski VG, Poznyak A (2011) The robust maximum principle: theory and applications. Springer Science & Business Media
8. Bonnans JF, Hermant A (2009) Revisiting the analysis of optimal control problems with several state constraints. Control Cybernet 38(4A):1021–1052
9. Bonnans JF, Dupuis X, Pfeiffer L (2014) Second-order necessary conditions in Pontryagin form for optimal control problems. SIAM J Control Optim 52(6):3887–3916
10. Bryson A, Ho YC (1975) Applied optimal control-optimization. Estimation and control. Taylor & Francis Inc., New York
11. Capuzzo-Dolcetta I (1998) Hamilton-Jacobi-bellman equations and optimal control. Variational calculus, optimal control and applications. Springer, pp 121–132
12. Caratheodory C (1935) Variationsrechnung und partielle Differentialgleichungen erster Ordnung. B. G, Teubner, Leipzig
13. Cesari L (2012) Optimization theory and applications: problems with ordinary differential equations, vol 17. Springer Science & Business Media
14. Clarke F (2013) Functional analysis, calculus of variations and optimal control, vol 264. Springer Science & Business Media
15. Clarke F, De Pinho M (2010) Optimal control problems with mixed constraints. SIAM J Control Optim 48(7):4500–4524
16. Clarke FH (1990) Optimization and nonsmooth analysis, vol 5. Siam
17. Crandall MG, Lions PL (1983) Viscosity solutions of Hamilton-Jacobi equations. Trans Am Math Soc 277(1):1–42
18. Crandall MG, Evans LC, Lions PL (1984) Some properties of viscosity solutions of Hamilton-Jacobi equations. Trans Am Math Soc 282(2):487–502
19. Dmitruk A (1993) Maximum principle for the general optimal control problem with phase and regular mixed constraints. Comput Math Model 4(4):364–377
20. Dmitruk A, Kaganovich A (2008) The hybrid maximum principle is a consequence of Pontryagin maximum principle. Syst Control Lett 57(11):964–970
21. Dubovitskii AY, Milyutin A (1965) Constrained extremum problems. Zh v?chisl Mat mat Fiz 5(3):395–453
22. Filippov A (1962) On certain questions in the theory of optimal control. J Soc Ind Appl Math Ser A: Control 1(1):76–84
23. Garavello M, Piccoli B (2005) Hybrid necessary principle. SIAM J Control Optim 43: 1867–1887
24. Gelfand IM, Silverman RA et al (2000) Calculus of variations. Courier Corporation
25. Girsanov IV (1972) Lectures on mathematical theory of extremum problems, vol 67. Springer
26. Goldstine HH (2012) A history of the calculus of variations from the 17th through the 19th century, vol 5. Springer Science & Business Media
27. Graichen K, Petit N (2009) Incorporating a class of constraints into the dynamics of optimal control problems. Optim Control Appl Methods 30(6):537–561

28. Hamilton WR (1834) On a general method in dynamics; by which the study of the motions of all free systems of attracting or repelling points is reduced to the search and differentiation of one central relation, or characteristic function. Philos Trans R Soc Lond 124:247–308
29. Hamilton WR (1835) Second essay on a general method in dynamics. Philos Trans R Soc Lond 125:95–144
30. Hartl RF, Sethi SP, Vickson RG (1995) A survey of the maximum principles for optimal control problems with state constraints. SIAM Rev 37(2):181–218
31. Hestenes MR (1949) Numerical methods of obtaining solutions of fixed end-point problems in the calculus of variations. Technical report, RAND Corporation
32. Hestenes MR (1950) A general problem in the calculus of variations with applications to paths of least time. Technical report, Rand Corporation
33. Hestenes MR (1966) Calculus of variations and optimal control theory. Wiley
34. Hoehener D (2013) Second-order necessary optimality conditions in state constrained optimal control. In: Proceedings of the 52nd IEEE conference on decision and control (CDC). IEEE, pp 544–549
35. Ioffe AD, Tihomirov VM (1979) Theory of extremal problems. North-Holland
36. Jacobson DH, Lele M, Speyer JL (1971) New necessary conditions of optimality for control problems with state-variable inequality constraints. J Math Anal Appl 35(2):255–284
37. Kirk D (1970) Optimal control theory: an introduction. Englewood Cliffs, N.J., Prentice-Hall
38. Knowles G (1981) An introduction to applied optimal control. Academic Press
39. Lee EB, Markus L (1967) Foundations of optimal control theory. Technical report, DTIC Document
40. Liberzon D (2012) Calculus of variations and optimal control theory: a concise introduction. Princeton University Press
41. Maurer H (1979) On the minimum principle for optimal control problems with state constraints. Rechenzentrum d. Univ.
42. Nakane M, Fraser CG (2002) The early history of Hamilton-Jacobi dynamics 1834–1837. Centaurus 44(3–4):161–227
43. Passenberg B, Leibold M, Stursberg O, Buss M (2011) The minimum principle for time-varying hybrid systems with state switching and jumps. In: Proceedings of the IEEE conference on decision and control, pp 6723–6729
44. Pesch HJ, Plail M (2009) The maximum principle of optimal control: a history of ingenious ideas and missed opportunities. Control Cybern 38(4A):973–995
45. Pesch HJ, Plail M (2012) The cold war and the maximum principle of optimal control. Optimization Stories Documenta Mathematica
46. Pontryagin L, Boltyanskii V, Gamkrelidze R, Mishchenko E (1962) The mathematical theory of optimal processes. Wiley
47. Riedinger P, Kratz F (2003) An optimal control approach for hybrid systems. Eur J Control 9:449–458
48. Riedinger P, Kratz F, Iung C, Zannes C (1999) Linear quadratic optimization for hybrid systems. In: Proceedings of the 38th IEEE conference on decision and control, pp 3059–3064
49. SA Attia Jr, Azhmyakov V (2007) State jump optimization for a class of hybrid autonomous systems. In: Proceedings of the 16th IEEE international conference on control applications, pp 1408–1413
50. Sargent R (2000) Optimal control. J Comput Appl Math 124(1):361–371
51. Schori M, Boehme TJ, Jeinsch T, Lampe B (2015) Switching time optimization for discontinuous switched systems. In: Proceedings of the 2015 European control conference (ECC), 15–17 July. Linz, IEEE, pp 1742–1747
52. Shaikh MS (2004) Optimal control of hybrid systems: theory and algorithms. PhD thesis, Department of Electrical and Computer Engineering, McGill University, Montreal
53. Sussmann HJ (1999) A maximum principle for hybrid optimal control problems. In: Proceedings of the 38th IEEE conference on decision and control, Phoenix, vol 1. IEEE, pp 425–430
54. Sussmann HJ (2000) Transversality, set separation, and the proof of maximum principle, Part II

55. Vinter R (2010) Optimal control. Springer Science & Business Media. doi:10.1007/9780817680862
56. Xu X, Antsaklis P (2003) Optimal control of hybrid autonomous systems with state jumps. In: Proceedings of the American control conference. IEEE, pp 5191–5196
57. Xu X (2001) Analysis and design of switched systems. PhD thesis, University of Notre Dame

Part II
Methods for Optimal Control

Chapter 5
Discretization and Integration Schemes for Hybrid Optimal Control Problems

5.1 Introduction

Let us start the discussion by considering the following hybrid optimal control problem (HOCP)

$$\phi\left(q^*(\cdot), \mathbf{u}^*(\cdot)\right) = \min_{q(\cdot)\in\mathcal{Q},\, \mathbf{u}(\cdot)\in\mathcal{U}(q(\cdot))} m(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} l_{q(t)}(\mathbf{x}(t), \mathbf{u}(t))\, \mathrm{d}t \tag{5.1}$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)) \tag{5.2}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0. \tag{5.3}$$

In Chap. 3, we introduced $\phi(\cdot)$ as a functional where the controls $q(\cdot)$ and $\mathbf{u}(\cdot)$ are measurable and the states $\mathbf{x}(\cdot)$ are absolutely continuous. The terms measurable and absolutely continuous are important to deduce necessary conditions but for practically relevant solutions this concept is far to general, because Zeno solutions are per se excluded as practically desirable solutions. Zeno-behavior describes the occurrence of a switching pattern having infinitely many switchings in finite time (cf. Zhang et al. [29]). It is then a natural choice to take the controls as piecewise continuous and the states as piecewise differentiable.

Moreover, in Chap. 4 we deduced necessary conditions which can in principle be used to obtain explicit state and control functions, which minimize the OCP. However, this way is not feasible for the most practically relevant problems, even for simple problems. In general, we need numerical approximations of these functions. Then, the task is quite clear, we try to find numerically piecewise approximations of the optimal control trajectories $\mathbf{u}^*(t)$ and $\mathbf{x}^*(t)$ taken from the admissible sets $\hat{\mathcal{U}}(q(t), t)$ and $\hat{\mathcal{X}}(q(\cdot), t)$ over the interval $t \in [t_0, t_f]$. This procedure implies that we discretize the problem formulation (5.1)–(5.3) using a time grid. This changes dramatically our viewpoint: an objective functional becomes an objective function and the admissible sets for controls and states shrink from an infinite dimensional space to a finite dimensional space.

T.J. Böhme and B. Frank, *Hybrid Systems, Optimal Control and Hybrid Vehicles*,
Advances in Industrial Control, DOI 10.1007/978-3-319-51317-1_5

5.2 Discretization of the Initial Value Problem

The numerical solution of *initial value problems* (IVP) for ODEs is fundamental to the numerical solution of hybrid optimal control problems (5.1)–(5.3). To allow for more compact and clearer descriptions of the algorithms presented in the coming sections, we adopt the notation $\mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t))$ of the vector field of the hybrid system to $\mathbf{f}(\mathbf{x}(t), q(t), \mathbf{u}(t))$.

Let us assume that $q(\cdot)$ and $\mathbf{u}(\cdot)$ are known for the IVP of the controlled switched system in (5.2)–(5.3)

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), q(t), \mathbf{u}(t)), \quad \forall t \in [t_0, t_f] \tag{5.4}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{5.5}$$

and that all the initial values (5.5) of the differential equation are given. Since we assumed that $\mathbf{f}(\cdot)$ is continuous on the time interval $[t_0, t_f]$ and that $\mathbf{f}(\cdot)$ satisfies the Lipschitz condition from Theorem 3.1, a solution is guaranteed to exist which is also unique.

No loss of generality results by assuming autonomous systems. It is always possible to transform non-autonomous systems into autonomous systems by introducing an additional variable $x_{N_x+1}(\cdot)$

$$x_{N_x+1}(t) = t \tag{5.6}$$

$$\dot{x}_{N_x+1}(t) = 1 \tag{5.7}$$

$$x_{N_x+1}(t_0) = 0 \tag{5.8}$$

which is used to substitute t with (5.6) and thus satisfies the IVP with (5.7)–(5.8).

To make the IVP (5.4)–(5.5) amenable for numerical integration, a certain discretization of the problem is required. The discretization will have a major effect on the quality of the solution and on the computation time and consequently, care has to be taken when choosing the discretization method as well as the discretization parameters. We restrict our considerations to explicit one-step methods of the Runge–Kutta class.

We start by discretizing the time t with the formulae

$$t_k = k \cdot h_k, \quad k = 0, \ldots, N_t$$

where h_k is known as the *step-length*. One obtains a time grid as a strictly increasing sequence of times $\{t_k\}$, $k = 0, \ldots, N_t$

$$0 = t_0 < t_1 < t_2 < \cdots < t_{N_t} = t_f, \qquad \mathcal{G}_t = \{t_0, t_1, t_2, \ldots, t_{N_t}\} \tag{5.9}$$

where

$$\begin{aligned} t_0 &= 0 \\ t_1 &= h_1 \\ &\vdots \\ t_f &= N_t \cdot h_{N_t}. \end{aligned}$$

The grid (5.9) is referred to as an *integration grid* or *discretization grid*. Sometimes the term mesh is used in the literature instead. The step-length h_k defined as

$$\begin{aligned} h_0 &= 0 \\ h_k &= t_k - t_{k-1}, \qquad k = 1, \ldots, N_t \end{aligned}$$

is not necessarily equidistant. Consequently,

$$h^{max} = \max_k h_k$$

is the fineness of the discretization grid $\mathcal{G}_t$.

We are now looking for an accurate and stable numerical approximation of the exact solution $\mathbf{x}(t_k) \approx \mathbf{x}_k$ at the grid points $k = 0, \ldots, N_t$.

Definition 5.1 (*Discretization Scheme*) A discretization scheme to approximate the exact solution $\mathbf{x}(t)$ of the IVP (5.4)–(5.5) is a procedure, which assigns every grid point of $\mathcal{G}_t$ a grid function with $\mathbf{x} : \mathcal{G}_t \to \hat{\mathcal{X}}$.

△

The simplest discretization scheme is the explicit Euler method with

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_k + h_k \cdot \mathbf{f}(\mathbf{x}_k, q_k, \mathbf{u}_k), \quad k = 0, \ldots, N_t - 1, \\ \mathbf{x}_0 &= \mathbf{x}(t_0). \end{aligned} \tag{5.10}$$

This procedure is obtained by approximating $\mathbf{f}(\mathbf{x}(t_k), q(t_k), \mathbf{u}(t_k))$ with $\mathbf{f}(\mathbf{x}_k, q_k, \mathbf{u}_k)$ and $\dot{\mathbf{x}}(t_k)$ with the forward difference quotient $(\mathbf{x}_{k+1} - \mathbf{x}_k) / h_k$.

5.3 Runge–Kutta Integration Scheme

Motivated by the discretization scheme (5.10), our aim is now the construction of a general expression for a single step from t_k to t_{k+1}, which allows us to control the accuracy of the IVP approximation.

Integration of (5.4)–(5.5) yields

$$\begin{aligned}\mathbf{x}_{k+1} &= \mathbf{x}_k + \int_{t_k}^{t_{k+1}} \dot{\mathbf{x}}(t)\ \mathrm{d}t \\ &= \mathbf{x}_k + \int_{t_k}^{t_{k+1}} \mathbf{f}(\mathbf{x}(t), q(t), \mathbf{u}(t))\ \mathrm{d}t, \quad k = 0, \ldots, N_t - 1.\end{aligned}$$

To evaluate the integral, let us first subdivide the integration step into K subintervals

$$\tau_{l,k} = t_k + c_l h_k, \quad \text{for } 1 \le l \le K \tag{5.11}$$

with

$$0 \le c_1 \le c_2 \le \cdots \le c_K \le 1.$$

To obtain a K-stage Runge–Kutta method (K function evaluations per integration step), we apply a quadrature formula and obtain

$$\int_{t_k}^{t_{k+1}} \mathbf{f}(\mathbf{x}(t), q(t), \mathbf{u}(t))\ \mathrm{d}t \approx h_k \cdot \sum_{l=1}^{K} b_l \mathbf{f}_l$$

where $\mathbf{f}_l = \mathbf{f}(\mathbf{x}_{l,k}, \breve{q}_{l,k}, \breve{\mathbf{u}}_{l,k})$ is the value of the right-hand side vector function at intermediate time points $\tau_{l,k}$, $1 \le l \le K$. Of course, this requires the approximation of values for the continuous states $\mathbf{x}_{l,k}$, the discrete state $\breve{q}_{l,k}$, and the continuous-valued controls $\breve{\mathbf{u}}_{l,k}$, which is described next.

The continuous-valued controls $\mathbf{u}(\cdot)$ are discretized using a set of N_t base functions $\Xi_k^u : \mathsf{U} \times \mathsf{U} \times [t_0, t_f] \to \mathsf{U}$

$$\mathbf{u}(t) = \begin{cases} \Xi_k^u\left(\mathbf{u}_k, \mathbf{u}_{k+1}, t\right), & \forall t \in \left[t_k, t_{k+1}\right), \quad k = 0, \ldots, N_t - 2 \\ \Xi_{N_t-1}^u\left(\mathbf{u}_{N_t-1}, \mathbf{u}_{N_t}, t\right), & \forall t \in \left[t_{N_t-1}, t_{N_t}\right] \end{cases} \tag{5.12}$$

that define the control on each time interval. To ensure separability of the discretization the base functions $\Xi_k^u(\cdot)$ shall have local support. Piecewise constant and piecewise linear approximation schemes as described by von Stryk [28] are popular choices for base functions. The piecewise constant parametrization approximates the exact controls by constant control values on each time interval using

$$\mathbf{u}(t) = \begin{cases} \Xi_k^u\left(\mathbf{u}_k, \mathbf{u}_{k+1}, t\right) = \begin{cases} \mathbf{u}_k, & \forall t \in \left[t_k, \dfrac{t_k + t_{k+1}}{2}\right) \\ \mathbf{u}_{k+1}, & \forall t \in \left[\dfrac{t_k + t_{k+1}}{2}, t_{k+1}\right) \end{cases}, \quad k = 0, \ldots, N_t - 2 \\ \Xi_{N_t-1}^u\left(\mathbf{u}_{N_t-1}, \mathbf{u}_{N_t}, t\right) = \begin{cases} \mathbf{u}_{N_t-1}, & \forall t \in \left[t_{N_t-1}, \dfrac{t_{N_t-1} + t_{N_t}}{2}\right) \\ \mathbf{u}_{N_t}, & \forall t \in \left[\dfrac{t_{N_t-1} + t_{N_t}}{2}, t_{N_t}\right] \end{cases} \end{cases} \tag{5.13}$$

whereas the piecewise linear parametrization interpolates linearly between $\mathbf{u}(t_k)$ and $\mathbf{u}(t_{k+1})$ by the base functions

$$\mathbf{u}(t) = \begin{cases} \dfrac{t_{k+1} - t}{t_{k+1} - t_k}\mathbf{u}_k + \dfrac{t - t_k}{t_{k+1} - t_k}\mathbf{u}_{k+1}, & \forall t \in \left[t_k, t_{k+1}\right), \quad k = 0, \ldots, N_t - 2 \\ \dfrac{t_{N_t} - t}{t_{N_t} - t_{N_t-1}}\mathbf{u}_{N_t-1} + \dfrac{t - t_{N_t-1}}{t_{N_t} - t_{N_t-1}}\mathbf{u}_{N_t}, & \forall t \in \left[t_{N_t-1}, t_{N_t}\right] \end{cases}.$$

Let us notice some characteristics of both parametrization types:

- the piecewise constant control parametrization has an error of order $\mathcal{O}(h)$ ($h := \max_{k \in \{1,\ldots,N_t\}} h_k$);
- the piecewise linear control parametrization has an error of order $\mathcal{O}(h^2)$;
- for both parametrization types no additional optimization variables are necessary; and
- sparsity of the Jacobians of the control constraints.

Higher order parametrizations, e.g., cubic interpolation, can be found in Kirches [19] and Büskens [3], but require additional optimization variables.

Thus, the intermediate values of the continuous-valued controls are obtained by

$$\breve{\mathbf{u}}_{l,k} = \Xi_k^u\left(\mathbf{u}_k, \mathbf{u}_{k+1}, \tau_{l,k}\right).$$

Due to the piecewise constant nature of the discrete state trajectory $q(\cdot)$, a respective piecewise constant approximation scheme with N_t base functions $\Xi_k^q : \hat{\mathcal{Q}} \times \hat{\mathcal{Q}} \times [t_0, t_f] \to \hat{\mathcal{Q}}$ is used:

$$q(t) = \begin{cases} \Xi_k^q\left(q_k, q_{k+1}, t\right) = \begin{cases} q_k, & \forall t \in \left[t_k, \dfrac{t_k + t_{k+1}}{2}\right) \\ q_{k+1}, & \forall t \in \left[\dfrac{t_k + t_{k+1}}{2}, t_{k+1}\right) \end{cases}, \quad k = 0, \ldots, N_t - 2 \\ \Xi_{N_t-1}^q\left(q_{N_t-1}, q_{N_t}, t\right) = \begin{cases} q_{N_t-1}, & \forall t \in \left[t_{N_t-1}, \dfrac{t_{N_t-1} + t_{N_t}}{2}\right) \\ q_{N_t}, & \forall t \in \left[\dfrac{t_{N_t-1} + t_{N_t}}{2}, t_{N_t}\right] \end{cases} \end{cases}. \tag{5.14}$$

Thus, the intermediate values of the discrete state are obtained by

$$\breve{q}_{l,k} = \Xi_k^q \left(q_k, q_{k+1}, \tau_{l,k}\right).$$

Then, the intermediate values of the state trajectory are obtained by the following expression:

$$\mathbf{x}_{l,k} = \mathbf{x}_k + \int_{t_k}^{\tau_{l,k}} \mathbf{f}(\mathbf{x}(t), q(t), \mathbf{u}(t))\, \mathrm{d}t \approx \mathbf{x}_k + h_k \cdot \sum_{j=1}^{K} a_{l,j} \mathbf{f}(\mathbf{x}_{j,k}, \breve{q}_{j,k}, \breve{\mathbf{u}}_{j,k})$$

for $1 \leq l \leq K$.

Collecting results, we obtain a family of one-step methods for the solution of initial value problems (5.4)–(5.5) known as *Runge–Kutta* (RK) methods. Generally, a K-stage RK scheme formulated by Kutta [20] is given by the recursive relation

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_k + h_k \cdot \boldsymbol{\Gamma}_f \left(\mathbf{x}_k, \mathbf{x}_{k+1}, q_k, q_{k+1}, \mathbf{u}_k, \mathbf{u}_{k+1}, t_k, h_k\right), \quad k = 0, \ldots, N_t - 1 \\ \mathbf{x}_0 &= \mathbf{x}(t_0) \end{aligned} \tag{5.15}$$

where

$$\boldsymbol{\Gamma}_f \left(\mathbf{x}_k, \mathbf{x}_{k+1}, q_k, q_{k+1}, \mathbf{u}_k, \mathbf{u}_{k+1}, t_k, h_k\right) = \sum_{l=1}^{K} b_l \mathbf{k}_l \tag{5.16}$$

is the *increment function* for the time interval $[t_k, t_k + h_k)$ of the one-step methods.

The functions $\mathbf{k}_l(\cdot)$ are estimations of the function value $\mathbf{f}(\cdot)$ in the interior of the interval $[t_k, t_{k+1})$ and are recursively calculated by

$$\begin{aligned} &\mathbf{k}_l(\mathbf{x}_k, \mathbf{x}_{k+1}, q_k, q_{k+1}, \mathbf{u}_k, \mathbf{u}_{k+1}, t_k, h_k) = \mathbf{f}(\mathbf{x}_{l,k}, \breve{q}_{l,k}, \breve{\mathbf{u}}_{l,k}) \\ &= \mathbf{f}\left(\left[\mathbf{x}_k + h_k \sum_{j=1}^{K} a_{l,j} \mathbf{k}_j\right], \Xi_k^q \left(q_k, q_{k+1}, t_k + c_l h_k\right), \Xi_k^u \left(\mathbf{u}_k, \mathbf{u}_{k+1}, t_k + c_l h_k\right)\right) \end{aligned} \tag{5.17}$$

for all $1 \leq l \leq K$. Since we assume that $\mathbf{f}(\cdot)$ is Lipschitz-continuous, an unique Lipschitz constant L_f for the increment function exists. The coefficients $a_{l,j}$, b_l, and c_l from (5.15)–(5.17) are obtained from the so-called Butcher array $[\breve{\mathbf{c}}, \breve{\mathbf{A}}, \breve{\mathbf{b}}^T]$ (originally described in [5]) of the respective Runge–Kutta method. The Butcher array consists of three sets of parameters. The parameters $\breve{\mathbf{c}} = [c_1, c_2, c_3, \ldots, c_K]^T$, $c_l = \sum_{j=1}^{K} a_{l,j}$ are related to the instances of the integration grid where the RK integration evaluates the right-hand side function of (5.4) and the parameters $\breve{\mathbf{b}} = [b_1, b_2, \ldots, b_K]^T$ are relative weights assigned to each of these evaluations. The tableau parameters $\breve{\mathbf{A}} = [a_{l,j}]$ affect the order of consistence of the RK integration method.

Table 5.1 Butcher array in matrix notation

$\check{\mathbf{c}}$	$\check{\mathbf{A}}$
	$\check{\mathbf{b}}$

Table 5.2 Left table: Butcher array for implicit schemes; right table: Butcher array for explicit schemes

c_1	$a_{1,1}$	$a_{1,2}$	$\cdots$	$a_{1,K-1}$	$a_{1,K}$
c_2	$a_{2,1}$	$a_{2,2}$	$\cdots$	$a_{2,K-1}$	$a_{2,K}$
c_3	$a_{3,1}$	$a_{3,2}$	$\cdots$	$a_{3,K-1}$	$a_{3,K}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
c_K	$a_{K,1}$	$a_{K,2}$	$\cdots$	$a_{K,K-1}$	$a_{K,K}$
	b_1	b_2	$\cdots$	b_{K-1}	b_K

0					
c_2	$a_{2,1}$				
c_3	$a_{3,1}$	$a_{3,2}$			
$\vdots$	$\vdots$	$\vdots$	$\ddots$		
c_K	$a_{K,1}$	$a_{K,2}$	$\cdots$	$a_{K,K-1}$	
	b_1	b_2	$\cdots$	b_{K-1}	b_K

Butcher arrays are often displayed in the form as matrix notation (Table 5.1) or as tableau (Table 5.2).

Schemes with $a_{l,j} = 0$ for $j \geq l$ are called *explicit* and *implicit* otherwise.

The Runge–Kutta method (5.15)–(5.17) is often introduced by collocation methods. Suppose we consider the approximation of the solution of (5.4)–(5.5) by a polynomial of degree K over each time interval $[t_k, t_{k+1})$

$$\tilde{\mathbf{x}}(t) = \mathbf{l}_0 + \mathbf{l}_1 \cdot (t - t_k) + \cdots + \mathbf{l}_K \cdot (t - t_k)^K \tag{5.18}$$

with the coefficients $[\mathbf{l}_0, \mathbf{l}_1, \ldots, \mathbf{l}_K]$ chosen such that the approximation matches at the beginning of the time interval $[t_k, t_{k+1})$, i.e.,

$$\tilde{\mathbf{x}}(t_k) = \mathbf{x}_k,$$

whose derivative coincides at K given points with the vector field of the differential equation at the intermediate points (5.11)

$$\dot{\tilde{\mathbf{x}}}(\tau_{l,k}) = \mathbf{f}\left(\mathbf{x}(\tau_{l,k}), \Xi_k^q\left(q_k, q_{k+1}, \tau_{l,k}\right), \Xi_k^u\left(\mathbf{u}_k, \mathbf{u}_{k+1}, \tau_{l,k}\right)\right). \tag{5.19}$$

The conditions (5.19) are called *collocation* conditions and (5.18) is a *collocation polynomial*. Thus, we call the RK scheme (5.15)–(5.17) a collocation method and the solution produced by the method is a piecewise polynomial (Betts [1]). In Hairer and Wanner [15] it is shown, that the collocation method is equivalent to the K-stage RK method.

5.4 Consistence Order of Runge–Kutta Methods

We need a description of "how well" the Runge–Kutta methods solve the problems compared to the exact solutions. This introduces the concept of *order of consistence*. For short, only order of Runge–Kutta methods.

Butcher [4] developed an algebraic theory for determining the order of Runge–Kutta integration schemes for differential equations. His theory is based on B-series and associated calculus on rooted trees. We give a brief introduction into Butcher's theory which is inspired by the textbook of Strehmel et al. [27] and illustrates the basic concepts and provides the order conditions for RK methods up to order 4.

To obtain the order we expand the exact solution $\mathbf{x}(t+h)$ and the numerical solution $\mathbf{x}_{k+1}$ by a Taylor approximation and compare the terms of the series. If all terms matches for both series up to the order p, then the RK method has the consistence order p. For the sake of simplicity, let us assume a constant step-length $h_k \equiv h$ for the following derivations in this section. Doing so, we expand exemplarily the exact solution up to the order four and obtain

$$\mathbf{x}(t+h) = \mathbf{x}(t) + h\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x}(t) + \frac{h^2}{2}\frac{\mathrm{d}^2}{\mathrm{d}t^2}\mathbf{x}(t) + \frac{h^3}{3!}\frac{\mathrm{d}^3}{\mathrm{d}t^3}\mathbf{x}(t) + \frac{h^4}{4!}\frac{\mathrm{d}^4}{\mathrm{d}t^4}\mathbf{x}(t) + \mathcal{O}(h^5) \tag{5.20}$$

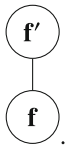
where $\mathcal{O}(h^5)$ accounts for the higher order terms. The next step is to evaluate the differentials. Differentiating the differential equation $\mathrm{d}\mathbf{x}/\mathrm{d}t = \mathbf{f}$ with respect to time yields

$$\frac{\mathrm{d}^2\mathbf{x}}{\mathrm{d}t^2} = \frac{\mathrm{d}\mathbf{f}}{\mathrm{d}t} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}'\dot{\mathbf{x}} = \mathbf{f}'\mathbf{f} \tag{5.21}$$

where $\mathbf{f}' = \mathbf{f}_x$ is a Jacobian matrix of the vector field $\mathbf{f}$. For the sake of simplicity we omit the arguments of the vector field and its derivatives. We denote $\mathbf{f}'\mathbf{f}$ according to Butcher as elementary differential which can be written in element-wise notation yielding

$$\left(\mathbf{f}'\mathbf{f}\right)_{[i]} = \sum_{k=1}^{N_x} \frac{\partial f_i}{\partial x_k} f_k, \quad i = 1, \ldots, N_x$$

where $[i]$ indicates the i-th element of the vector. Butcher proposed the organization of elementary differentials as rooted trees. Applying this idea the elementary differential $\mathbf{f}'\mathbf{f}$ can be conveniently organized as



We can proceed further by differentiating (5.21) which yields

$$\frac{\mathrm{d}^3\mathbf{x}}{\mathrm{d}t^3} = \mathbf{f}''(\mathbf{f},\mathbf{f}) + \mathbf{f}'\mathbf{f}'\mathbf{f}. \tag{5.22}$$

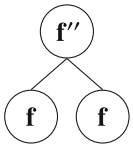
The two terms of (5.22) can be represented element-wise as

$$\begin{aligned}\left(\mathbf{f}''(\mathbf{f},\mathbf{f})\right)_{[i]} &= \sum_{k=1}^{N_x}\sum_{l=1}^{N_x}\frac{\partial^2 f_i}{\partial x_k \partial x_l} f_k f_l \\ &= \sum_{k=1}^{N_x} f_k \cdot \left[\sum_{l=1}^{N_x}\frac{\partial}{\partial x_l}\left(\frac{\partial f_i}{\partial x_k}\right) f_l\right], \quad i = 1,\ldots,N_x\end{aligned}$$

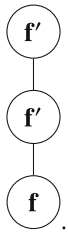
and

$$\begin{aligned}\left(\mathbf{f}'\mathbf{f}'\mathbf{f}\right)_{[i]} &= \sum_{k=1}^{N_x}\sum_{l=1}^{N_x}\frac{\partial f_i}{\partial x_k}\frac{\partial f_k}{\partial x_l} f_l \\ &= \sum_{k=1}^{N_x}\frac{\partial f_i}{\partial x_k} \cdot \left[\sum_{l=1}^{N_x}\frac{\partial f_k}{\partial x_l} f_l\right], \quad i = 1,\ldots,N_x\end{aligned}$$

or conveniently expressed as rooted trees



and



.

Proceeding further by differentiating both terms in (5.22) yields for the first term

$$\begin{aligned}\left(\mathbf{f}''(\mathbf{f},\mathbf{f})\right)' &= \mathbf{f}'''(\mathbf{f},\mathbf{f},\mathbf{f}) + \mathbf{f}''\left(\mathbf{f}'\mathbf{f},\mathbf{f}\right) + \mathbf{f}''\left(\mathbf{f},\mathbf{f}'\mathbf{f}\right) \\ &= \mathbf{f}'''(\mathbf{f},\mathbf{f},\mathbf{f}) + 2\mathbf{f}''\left(\mathbf{f},\mathbf{f}'\mathbf{f}\right)\end{aligned} \tag{5.23}$$

and for the second term

$$\left(\mathbf{f}'\mathbf{f}'\mathbf{f}\right)' = \mathbf{f}''\left(\mathbf{f},\mathbf{f}'\mathbf{f}\right) + \mathbf{f}'\mathbf{f}''(\mathbf{f},\mathbf{f}) + \mathbf{f}'\mathbf{f}'\mathbf{f}'\mathbf{f}. \tag{5.24}$$

The fourth-order differential is then obtained by collecting the terms from (5.23) and (5.24) yielding

$$\frac{\mathrm{d}^4\mathbf{x}}{\mathrm{d}t^4} = \mathbf{f}'''(\mathbf{f},\mathbf{f},\mathbf{f}) + 3\mathbf{f}''\left(\mathbf{f},\mathbf{f}'\mathbf{f}\right) + \mathbf{f}'\mathbf{f}''(\mathbf{f},\mathbf{f}) + \mathbf{f}'\mathbf{f}'\mathbf{f}'\mathbf{f}. \tag{5.25}$$

The terms of (5.25) can be represented element-wise as

$$\begin{aligned}\left(\mathbf{f}'''\left(\mathbf{f},\mathbf{f},\mathbf{f}\right)\right)_{[i]} &= \sum_{k=1}^{N_x}\sum_{l=1}^{N_x}\sum_{j=1}^{N_x}\frac{\partial^3 f_i}{\partial x_k \partial x_l \partial x_j} f_k f_l f_j \\ &= \sum_{k=1}^{N_x} f_k \cdot \left\{\sum_{l=1}^{N_x} f_l \cdot \left[\sum_{j=1}^{N_x}\frac{\partial}{\partial x_j}\left[\frac{\partial}{\partial x_l}\left(\frac{\partial f_i}{\partial x_k}\right)\right]\cdot f_j\right]\right\}, \quad i = 1,\ldots,N_x\end{aligned}$$

and

$$\begin{aligned}\left(\mathbf{f}''\left(\mathbf{f},\mathbf{f}'\mathbf{f}\right)\right)_{[i]} &= \sum_{k=1}^{N_x}\sum_{l=1}^{N_x}\sum_{j=1}^{N_x}\frac{\partial^2 f_i}{\partial x_k \partial x_l} f_k \frac{\partial f_l}{\partial x_j} f_j \\ &= \sum_{k=1}^{N_x} f_k \cdot \left\{\sum_{l=1}^{N_x}\frac{\partial}{\partial x_l}\left(\frac{\partial f_i}{\partial x_k}\right)\cdot\left[\sum_{j=1}^{N_x}\frac{\partial f_l}{\partial x_j} f_j\right]\right\}, \quad i = 1,\ldots,N_x\end{aligned}$$

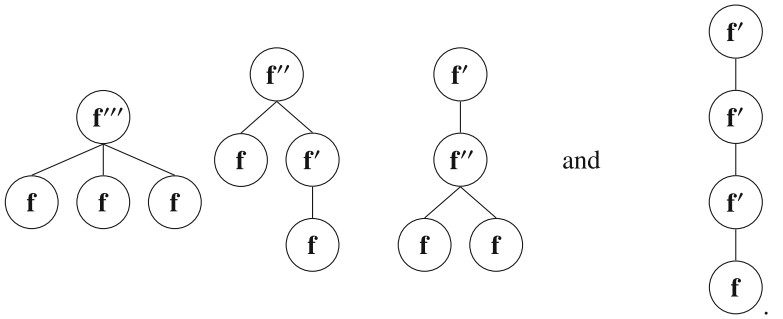
and

$$\begin{aligned}\left(\mathbf{f}'\mathbf{f}''\left(\mathbf{f},\mathbf{f}\right)\right)_{[i]} &= \sum_{k=1}^{N_x}\sum_{l=1}^{N_x}\sum_{j=1}^{N_x}\frac{\partial f_i}{\partial x_k}\frac{\partial^2 f_k}{\partial x_l \partial x_j} f_l f_j \\ &= \sum_{k=1}^{N_x}\frac{\partial f_i}{\partial x_k}\cdot\left\{\sum_{l=1}^{N_x} f_l \cdot \left[\sum_{j=1}^{N_x}\frac{\partial}{\partial x_j}\left(\frac{\partial f_k}{\partial x_l}\right)\cdot f_j\right]\right\}, \quad i = 1,\ldots,N_x\end{aligned}$$

and

$$\begin{aligned}\left(\mathbf{f}'\mathbf{f}'\mathbf{f}'\mathbf{f}\right)_{[i]} &= \sum_{k=1}^{N_x}\sum_{l=1}^{N_x}\sum_{j=1}^{N_x}\frac{\partial f_i}{\partial x_k}\frac{\partial f_k}{\partial x_l}\frac{\partial f_l}{\partial x_j} f_j \\ &= \sum_{k=1}^{N_x}\frac{\partial f_i}{\partial x_k}\cdot\left\{\sum_{l=1}^{N_x}\frac{\partial f_k}{\partial x_l}\cdot\left[\sum_{j=1}^{N_x}\frac{\partial f_l}{\partial x_j} f_j\right]\right\}, \quad i = 1,\ldots,N_x\end{aligned}$$

or conveniently expressed as rooted trees



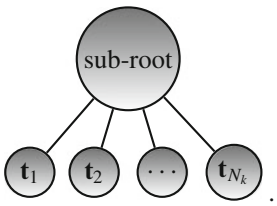
Extensions for higher derivatives $\mathbf{f}^{(N_k)}$ are straightforward. The idea from Butcher becomes now apparent and let us denote the set of rooted trees as

$$\mathbf{T} = \Bigg\{ \underbrace{\bullet}_{\mathbf{T}_1 = root}, \quad \underbrace{\bullet\!\!\bullet}_{\mathbf{T}_2 = \mathbf{t}_1}, \quad \underbrace{\bullet\!\!\bullet\!\!\bullet, \bullet\!\!\bullet\!\!\bullet}_{\mathbf{T}_3 \quad \mathbf{t}_2 \quad \mathbf{t}_3}, \quad \underbrace{\bullet\!\!\bullet\!\!\bullet\!\!\bullet, \bullet\!\!\bullet\!\!\bullet\!\!\bullet, \bullet\!\!\bullet\!\!\bullet\!\!\bullet, \bullet\!\!\bullet\!\!\bullet\!\!\bullet\!\!\bullet}_{\mathbf{T}_4 \quad \mathbf{t}_4 \quad \mathbf{t}_5 \quad \mathbf{t}_6 \quad \mathbf{t}_7}, \quad \dots \Bigg\}$$

$$= \{\text{root}\} \cup \left\{ \left[\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_{N_k} \right] : \mathbf{t}_i \in \mathbf{T} \right\}$$

where root $= \bullet$ is the root of the tree. The bracket $[\cdot]$ is an operator to append arbitrary trees to a sub-root

$$\left[\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_{N_k} \right] :=$$



To generate any rooted tree the $[\cdot]$-operator can be recursively applied, e.g., $\bullet\!\!\bullet\!\!\bullet = [[\bullet]]$ or $\bullet\!\!\bullet\!\!\bullet = \left[\bullet\!\!\bullet\!\!\bullet, \bullet \right] = [[\bullet, \bullet], [\bullet]]$. The relationship between the rooted trees and the elementary differentials can be stated in a recursive way using the $[\cdot]$-operator.

Definition 5.2 (*Elementary Differential associated to the Tree* $\mathbf{t}$) The elementary differential assigned to the rooted tree $\mathbf{t} \in \mathbf{T}$ is given by

$$F(\mathbf{t})(\mathbf{x}(t)) =$$

$$\begin{cases} \mathbf{f}(\mathbf{x}(t), \cdot) & \text{for } \mathbf{t} = \text{root} \\ \mathbf{f}^{(N_k)}(\mathbf{x}(t), \cdot) \left[F(\mathbf{t}_1)(\mathbf{x}(t)), F(\mathbf{t}_2)(\mathbf{x}(t)), \dots, F(\mathbf{t}_{N_k})(\mathbf{x}(t)) \right] & \text{for } \mathbf{t} = \left[\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_{N_k} \right]. \end{cases}$$

△

Now, the calculated differentials are applied to the Taylor series (5.20). According to Definition 5.2 we obtain

$$\mathbf{x}(t+h) = \mathbf{x}(t) + hF(\text{root})(\mathbf{x}(t)) + \frac{h^2}{2}F(\mathbf{t}_1)(\mathbf{x}(t)) + \frac{h^3}{3!}\left[F(\mathbf{t}_2)(\mathbf{x}(t)) + F(\mathbf{t}_3)(\mathbf{x}(t))\right] + \frac{h^4}{4!}\left[F(\mathbf{t}_4)(\mathbf{x}(t)) + 3F(\mathbf{t}_5)(\mathbf{x}(t)) + F(\mathbf{t}_6)(\mathbf{x}(t)) + F(\mathbf{t}_7)(\mathbf{x}(t))\right] + \mathcal{O}(h^5). \quad (5.26)$$

With this terminology, we can generalize the result (5.26) to a formal Taylor series for the solution to the ODE (5.20).

Definition 5.3 (*Order of the rooted tree* $\mathbf{t} \in \mathbf{T}$) The number of nodes of the tree $\mathbf{t} \in \mathbf{T}$ is called the order $\rho(\mathbf{t})$. Then, let us define a set of trees

$$\mathbf{T}_p := \{\mathbf{t} \in \mathbf{T} \mid \rho(\mathbf{t}) \leq p\}$$

for all trees up to order p.

△

Definition 5.4 (*Symmetry and Density*) The symmetry $\beta(\mathbf{t})$ is defined by

$$\beta(root) = 1$$
$$\beta\left(\left[\mathbf{t}_1^{l_1}, \mathbf{t}_2^{l_2}, \ldots, \mathbf{t}_{N_k}^{l_{N_k}}\right]\right) = \prod_{s=1}^{N_k} l_s! \beta(\mathbf{t}_s)^{l_s}.$$

The exponents $l_1, l_2, \ldots, l_{N_k}$ are the numbers of equal sub-trees.
The density $\gamma(\mathbf{t})$ is defined by

$$\gamma(root) = 1$$
$$\gamma(\mathbf{t}) = \rho(\mathbf{t}) \prod_{s=1}^{N_k} \gamma(\mathbf{t}_s), \quad \mathbf{t} = \left[\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_{N_k}\right].$$

△

Theorem 5.1 (Exact Solution by Elementary Differentials) *The exact solution can be represented by*

$$\mathbf{x}(t_k + h) = \mathbf{x}(t_k) + \sum_{\mathbf{t} \in \mathbf{T}_p} \frac{1}{\gamma(\mathbf{t})} \frac{h^{\rho(\mathbf{t})}}{\beta(\mathbf{t})} F(\mathbf{t})(\mathbf{x}(t_k)) + \mathcal{O}(h^{p+1}), \quad k = 0, \ldots, N_t - 1 \quad (5.27)$$

with the pth derivative of the exact solution

$$\frac{\mathrm{d}^p \mathbf{x}}{\mathrm{d}t^p} = \sum_{\mathbf{t}\in\mathbf{T}_p} \frac{1}{\gamma(\mathbf{t})} \frac{1}{\beta(\mathbf{t})} F(\mathbf{t})(\mathbf{x}(t_k))$$

and $\mathbf{x}(t_0)$ *as initial states.*

△

Proof The proof is given in Butcher [4]. □

Analogously to the exact solution a similar expression for the RK discretization is required. It is convenient to define certain functions which bear a (1–1) correspondence to the elementary differentials. Let us define the following elementary weights.

Definition 5.5 (*Elementary Weights of RK Methods*) An elementary weight $\phi(\cdot)$ of a K-stage RK discretization method is defined by

$$\phi(\text{root}) = \sum_{l=1}^{K} b_l \tag{5.28}$$

$$\begin{aligned}\phi\left(\left[\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_{N_k}\right]\right) &= \sum_{l=1}^{K} b_l \prod_{s=1}^{N_k} \tilde{\boldsymbol{\phi}}^{[l]}(\mathbf{t}_s) \\ &= \sum_{l=1}^{K} b_l \cdot \tilde{\boldsymbol{\phi}}^{[l]}(\mathbf{t}_1) \cdot \tilde{\boldsymbol{\phi}}^{[l]}(\mathbf{t}_2) \cdot \ldots \cdot \tilde{\boldsymbol{\phi}}^{[l]}(\mathbf{t}_{N_k})\end{aligned} \tag{5.29}$$

where

$$\tilde{\boldsymbol{\phi}}(\text{root}) = \left(\sum_{j=1}^{K} a_{l,j} = c_l\right)_{l=1,\ldots,K} \tag{5.30}$$

$$\begin{aligned}\tilde{\boldsymbol{\phi}}\left(\left[\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_{N_k}\right]\right) &= \left(\sum_{j=1}^{K} a_{l,j} \prod_{s=1}^{N_k} \tilde{\boldsymbol{\phi}}^{[j]}(\mathbf{t}_s)\right)_{l=1,\ldots,K} \\ &= \left(\sum_{j=1}^{K} a_{l,j} \cdot \tilde{\boldsymbol{\phi}}^{[j]}(\mathbf{t}_1) \cdot \tilde{\boldsymbol{\phi}}^{[j]}(\mathbf{t}_2) \cdot \ldots \cdot \tilde{\boldsymbol{\phi}}^{[j]}(\mathbf{t}_{N_k})\right)_{l=1,\ldots,K}.\end{aligned} \tag{5.31}$$

△

With the help of the elementary weights in Definition 5.5 we can state a correspondence to (5.27).

Theorem 5.2 (RK Solution by Elementary Differentials) *For the solution of a RK discretization, the stage and recursive equations*

$$\tilde{\mathbf{x}}_{l,k+1} = \mathbf{x}(t_k) + \sum_{\mathbf{t}\in\mathbf{T}_p} \tilde{\boldsymbol{\phi}}^{[l]}(\mathbf{t}) \frac{h^{\rho(\mathbf{t})}}{\beta(\mathbf{t})} F(\mathbf{t})(\mathbf{x}(t_k)) + \mathcal{O}(h^{p+1}), \quad l = 1, \ldots, K \quad k = 0, \ldots, N_t - 1 \tag{5.32}$$

$$\tilde{\mathbf{x}}_{k+1} = \mathbf{x}(t_k) + \sum_{\mathbf{t}\in\mathbf{T}_p} \phi(\mathbf{t}) \frac{h^{\rho(\mathbf{t})}}{\beta(\mathbf{t})} F(\mathbf{t})(\mathbf{x}(t_k)) + \mathcal{O}(h^{p+1}), \quad k = 0, \ldots, N_t - 1 \tag{5.33}$$

must hold with $\mathbf{x}(t_0)$ *as initial states.*

△

Proof The proof that equations (5.32) and (5.33) satisfy the stage and recursive equations, respectively, is given in Butcher [4]. □

Comparing the two series (5.27) and (5.33) yields the order of the RK method.

Theorem 5.3 (Consistence Order of RK Methods) *A Runge–Kutta method has the consistence order p, if for all trees* $\mathbf{t} \in \mathbf{T}_p$ *the condition*

$$\frac{1}{\gamma(\mathbf{t})} = \phi(\mathbf{t}) = \sum_{l=1}^{K} b_l \tilde{\boldsymbol{\phi}}^{[l]}(\mathbf{t}), \quad l = 1, \ldots, K$$

is satisfied.

△

Proof The proof is given in Butcher [4]. □

The order of RK methods with linear right-hand side functions might be higher as determined by Theorem 5.3.

Remark 5.1 For explicit integration schemes the order of consistence is directly related with the number of stages K, whereas for implicit integration schemes the consistence orders greater than K may be possible.

The evaluation of the order conditions leads to tensor notations. To avoid these rather complex notations let us define the following helper vectors

$$\check{\mathbf{b}}_{-1} := \begin{bmatrix} \frac{1}{b_1} \\ \frac{1}{b_2} \\ \vdots \\ \frac{1}{b_K} \end{bmatrix} \quad \check{\mathbf{b}}_{-2} := \begin{bmatrix} \frac{1}{b_1^2} \\ \frac{1}{b_2^2} \\ \vdots \\ \frac{1}{b_K^2} \end{bmatrix} \quad \check{\mathbf{d}} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_K \end{bmatrix} := \check{\mathbf{A}}^T \check{\mathbf{b}}$$

$$\check{\mathbf{d}}_2 := \begin{bmatrix} d_1^2 \\ d_2^2 \\ \vdots \\ d_K^2 \end{bmatrix} \quad \check{\mathbf{d}}_3 := \begin{bmatrix} d_1^3 \\ d_2^3 \\ \vdots \\ d_K^3 \end{bmatrix} \quad \mathbb{1}_{(K\times 1)} := \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

and helper diagonal matrices

$$\begin{aligned} \check{\mathbf{C}} &:= diag\,(c_1, c_2, \ldots, c_K) \\ \check{\mathbf{C}}_2 &:= diag\left(c_1^2, c_2^2, \ldots, c_K^2\right) \\ \check{\mathbf{C}}_3 &:= diag\left(c_1^3, c_2^3, \ldots, c_K^3\right) \\ \check{\mathbf{D}} &:= diag\,(d_1, d_2, \ldots, d_K) \\ \check{\mathbf{D}}_2 &:= diag\left(d_1^2, d_2^2, \ldots, d_K^2\right). \end{aligned}$$

Using these helpers the order conditions are stated up to order $p = 4$ and depicted in Table 5.3. For all conditions $b_l > 0$ must hold.

After evaluation of the matrix-vector multiplications, the conditions from Table 5.3 for a fourth order Runge–Kutta process have the following form:

$$\begin{aligned} b_1 + b_2 + b_3 + b_4 &= 1 \\ b_2 a_{21} + b_3 (a_{31} + a_{32}) + b_4 (a_{41} + a_{42} + a_{43}) &= \frac{1}{2} \\ b_3 a_{32} a_{21} + b_4 (a_{42} a_{21} + a_{43} [a_{31} + a_{32}]) &= \frac{1}{6} \\ b_2 a_{21}^2 + b_3 (a_{31} + a_{32})^2 + b_4 (a_{41} + a_{42} + a_{43})^2 &= \frac{1}{3} \\ b_2 a_{21}^3 + b_3 (a_{31} + a_{32})^3 + b_4 (a_{41} + a_{42} + a_{43})^3 &= \frac{1}{4} \end{aligned}$$

Table 5.3 Order of a Runge–Kutta discretization for ordinary differential equations (cf. Strehmel et al. [27])

p	t	$F(\mathbf{t})$	$\beta(\mathbf{t})$	$\gamma(\mathbf{t})$	Conditions
1	root	$\mathbf{f}$	1	1	${}^{T}_{K\times 1}\check{\mathbf{b}} = 1$
2		$\mathbf{f}'\mathbf{f}$	1	2	${}^{T}_{K\times 1}\check{\mathbf{d}} = \frac{1}{2}$
3		$\mathbf{f}''(\mathbf{f},\mathbf{f})$	2	3	${}^{T}_{K\times 1}\check{\mathbf{C}}\check{\mathbf{d}} = \frac{1}{6}$
		$\mathbf{f}'\mathbf{f}'\mathbf{f}$	1	6	${}^{T}_{K\times 1}\check{\mathbf{C}}_2\check{\mathbf{b}} = \frac{1}{3}$
4		$\mathbf{f}'''(\mathbf{f},\mathbf{f},\mathbf{f})$	6	4	${}^{T}_{K\times 1}\check{\mathbf{C}}_3\check{\mathbf{b}} = \frac{1}{4}$
		$\mathbf{f}''(\mathbf{f},\mathbf{f}'\mathbf{f})$	1	8	$\left(\check{\mathbf{C}}\check{\mathbf{b}}\right)^{T}\check{\mathbf{A}}\check{\mathbf{c}} = \frac{1}{8}$
		$\mathbf{f}'\mathbf{f}''(\mathbf{f},\mathbf{f})$	2	12	${}^{T}_{K\times 1}\check{\mathbf{C}}_2\check{\mathbf{d}} = \frac{1}{12}$
		$\mathbf{f}'\mathbf{f}'\mathbf{f}'\mathbf{f}$	1	24	$\check{\mathbf{d}}^{T}\check{\mathbf{A}}\check{\mathbf{c}} = \frac{1}{24}$

$$
\begin{aligned}
b_3\left(a_{31}+a_{32}\right)a_{32}a_{21} + b_4\left(a_{41}+a_{42}+a_{43}\right)\left(a_{42}a_{21}+a_{43}\left[a_{31}+a_{32}\right]\right) &= \frac{1}{8}\\
b_3 a_{32}a_{21}^2 + b_4\left(a_{42}a_{21}^2 + a_{43}\left[a_{31}+a_{32}\right]^2\right) &= \frac{1}{12}\\
b_4 a_{43}a_{32}a_{21} &= \frac{1}{24}.
\end{aligned}
$$

The error of the discrete approximation to the OCP depends both on the smoothness of the solution to the original problem and on the order of the RK scheme used for the discretization. However, the theory developed by Butcher does not apply to discretizations for optimal control problems. Hager [14] showed in his paper that for the solution of optimal control problems additional conditions for the RK discretization must hold. Hager derived these conditions by establishing a connection between the Lagrange multipliers for the discretized problem and the costates associated with Pontryagin's minimum principle (see Sect. 4.4 for details about Pontryagin's minimum principle).

We state these additional order conditions, marked as red entries, in Table 5.4. Interested readers may consult the paper of Hager [14] for more details and proofs.

The conditions in Table 5.4 can be easily evaluated for checking the preferred RK discretization before implementation. Bonnans and Laurent-Varin [2] and Flaig [12] provide additional order conditions up to order six for RK discretization.

Table 5.4 Order conditions of a Runge–Kutta discretization for optimal control problems

Order	Conditions		
1	${}^{T}_{K\times 1}\check{\mathbf{b}} = 1$		
2	${}^{T}_{K\times 1}\check{\mathbf{d}} = \frac{1}{2}$		
3	${}^{T}_{K\times 1}\check{\mathbf{C}}\check{\mathbf{d}} = \frac{1}{6}$,	${}^{T}_{K\times 1}\check{\mathbf{C}}_2\check{\mathbf{b}} = \frac{1}{3}$,	$1^{T}_{K\times 1}\check{\mathbf{D}}_2\check{\mathbf{b}}_{-1} = \frac{1}{3}$
4	${}^{T}_{K\times 1}\check{\mathbf{C}}_3\check{\mathbf{b}} = \frac{1}{4}$,	$\left(\check{\mathbf{C}}\check{\mathbf{b}}\right)^{T}\check{\mathbf{A}}\check{\mathbf{c}} = \frac{1}{8}$,	${}^{T}_{K\times 1}\check{\mathbf{C}}_2\check{\mathbf{d}} = \frac{1}{12}$,
	$\check{\mathbf{d}}^{T}\check{\mathbf{A}}\check{\mathbf{c}} = \frac{1}{24}$,	$\left(\check{\mathbf{C}}\check{\mathbf{d}}_2\right)^{T}\check{\mathbf{b}}_{-1} = \frac{1}{12}$,	$\check{\mathbf{d}}_3^{T}\check{\mathbf{b}}_{-2} = \frac{1}{4}$,
	$\left(\check{\mathbf{C}}\check{\mathbf{b}}\right)^{T}\check{\mathbf{A}}\left(\check{\mathbf{D}}\check{\mathbf{b}}_{-1}\right) = \frac{5}{24}$,	$\check{\mathbf{d}}^{T}\check{\mathbf{A}}\left(\check{\mathbf{D}}\check{\mathbf{b}}_{-1}\right) = \frac{1}{8}$	

5.5 Stability

A numerical method is stable, if it is insensitive to small disturbances. For the case of Runge–Kutta quadrature this means that rounding errors are not sum up during the numerical integration process. In order to characterize the stability behavior of RK methods we are interested in the asymptotic behavior of the numerical solution for $t \to \infty$ with fixed step-length h.

Applying any RK method to the scalar IVP

$$\begin{aligned} \dot{x}(t) &= \lambda x(t), \quad \lambda \in \mathbb{C} \\ x(t_0) &= 1 \end{aligned}$$

yields

$$x_{l,k+1} = x_k + h\lambda \sum_{j=1}^{K} a_{l,j} x_{j,k+1}, \quad l = 1, \ldots, K \tag{5.34}$$

$$x_{k+1} = x_k + h\lambda \sum_{l=1}^{K} b_l x_{l,k+1} \tag{5.35}$$

for all $k = 0, \ldots, N_t - 1$. Collecting all K-stage terms from (5.34) yields

$$\mathbf{X}_{k+1} := \left[x_{1,k+1}, x_{2,k+1}, \ldots, x_{K,k+1}\right]^{T} \in \mathbb{C}^{K}. \tag{5.36}$$

We obtain from (5.34) and (5.36) a linear equation system of the form

$$\mathbf{X}_{k+1} = \mathbb{1}_{K\times 1} x_k + h\lambda \check{\mathbf{A}} \mathbf{X}_{k+1}. \tag{5.37}$$

Reformulation of (5.37) yields

$$(\mathbf{I} - h\lambda\check{\mathbf{A}})\mathbf{X}_{k+1} = \mathbb{1}_{K\times 1}x_k.$$

If $(\mathbf{I} - h\lambda\check{\mathbf{A}})$ is regular, then inversion yields

$$\mathbf{X}_{k+1} = (\mathbf{I} - h\lambda\check{\mathbf{A}})^{-1}\mathbb{1}_{K\times 1}x_k. \tag{5.38}$$

Inserting (5.38) into (5.35) yields

$$\begin{aligned} x_{k+1} &= x_k + h\lambda\check{\mathbf{b}}^T\mathbf{X}_{k+1} \\ &= (1 + z\check{\mathbf{b}}^T(\mathbf{I} - z\check{\mathbf{A}})^{-1}\mathbb{1}_{K\times 1})x_k, \quad z = h\lambda \\ &= R(z)x_k. \end{aligned}$$

Definition 5.6 (*Stability Function*) The complex-valued function

$$R(z) = 1 + z\check{\mathbf{b}}^T(\mathbf{I} - z\check{\mathbf{A}})^{-1}\mathbb{1}_{K\times 1}$$

is called the *stability function* of the RK method for the famous *Dahlquist test equation*

$$\dot{x}(t) = \lambda x(t), \quad \lambda \in \mathbb{C} \tag{5.39}$$

$$x(t_0) = 1. \tag{5.40}$$

The set

$$\mathcal{S}_{rk} = \{z \in \mathbb{C} \mid |R(z)| \leq 1\}$$

is called the *stability domain* of the RK method.

△

Remark 5.2 The following equivalent form of the stability function may be more appropriate for evaluation

$$R(z) = \frac{\det\begin{pmatrix} \mathbf{I} - z\check{\mathbf{A}} & \mathbb{1}_{K\times 1} \\ -z\check{\mathbf{b}}^T & 1 \end{pmatrix}}{\det\left(\mathbf{I} - z\check{\mathbf{A}}\right)}.$$

Despite its simplicity, the Dahlquist equation provides essential characteristics of stiff ODEs and has been well established as test equation for numerical methods.

We know from linear system theory that the exact solution of (5.39) is $x(t_k + h) = e^z x(t_k)$, which implies that $R(z)$ approximates the exponential function.

For a good numerical solution we require for (5.39) with Re $z \leq 0$, $z = h\lambda$, that

$$|x(t_k + h)| \leq |x(t_k)| \quad \text{or} \quad \lim_{\text{Re } z \to -\infty} x(t_k + h) = \lim_{\text{Re } z \to -\infty} e^z x(t_k) = 0$$

holds. This implies, that for the best case the stability domain $\mathcal{S}_{rk}$ is the complete left haft plane of the complex domain.

This leads to the important stability definitions.

Definition 5.7 (*A-Stability*) A Runge–Kutta method is *A-stable*, if

$$|R(z)| \leq 1, \quad \forall z \in \mathbb{C} \text{ with Re } z \leq 0$$

holds.

△

Definition 5.8 (*L-Stability*) A Runge–Kutta method is *L-stable*, if A-stable and the additional condition

$$\lim_{\text{Re } z \to -\infty} R(z) = 0$$

hold.

△

5.6 Some Lower-Order Runge–Kutta Integration Schemes

Runge–Kutta integration schemes can be distinguished in explicit and implicit ones. Explicit formulations have the advantage to be easily evaluable, whereas implicit formulations require in general the solution of a nonlinear equation system of K stage equations at each step-length h_k. The nonlinear equation system results from the appearance of $\mathbf{x}_{k+1}$ on the left-hand and right-hand side. Consequently, the computing effort is higher compared with the explicit schemes but for the advantage of improved numerical properties (e.g., A-stability).

We state in this book only those RK methods which satisfy the additional OCP conditions proposed by Hager [14]. Additionally, only implicit RK schemes which can be easily used in direct collocation transcriptions for optimal control problems are discussed. In particular, only RK schemes are considered, which can be directly implemented as equality constraints without requiring the introduction of additional optimization variables or the additional solution of nonlinear equation systems.

For the sake of compact notations we assume for all RK schemes that the continuous-valued controls $\mathbf{u}(\cdot)$, the continuous states $\mathbf{x}(\cdot)$, and the discrete state $q(\cdot)$ are numerically represented by approximations on the time grid $\mathcal{G}_t$. Then, the elements $\mathbf{u}_k$ are stored in the discretization vector

Table 5.5 Explicit Euler Method, $K = 1$, $p = 1$

$$\begin{array}{c|c} c_1 & a_{1,1} \\ \hline & b_1 \end{array} \quad = \quad \begin{array}{c|c} 0 & 0 \\ \hline & 1 \end{array}.$$

$$\overline{\mathbf{u}} := \left[\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N_t}\right] \in \mathbb{R}^{N_u \cdot (N_t+1)},$$

the elements q_k are stored in the discretization vector

$$\overline{\mathbf{q}} := \left[q_0, q_1, \ldots, q_{N_t}\right] \in \mathbb{R}^{N_q \cdot (N_t+1)},$$

and the elements $\mathbf{x}_k$ are stored in the discretization vector

$$\overline{\mathbf{x}} := \left[\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{N_t}\right] \in \mathbb{R}^{N_x \cdot (N_t+1)}.$$

We mark vectors that are assembled from a discretization process with an "overline" symbol.

5.6.1 Explicit Runge–Kutta Schemes

Explicit Euler Discretization

The most basic solver in the Runge–Kutta family is the *explicit Euler* method. It has the lowest computational demand (stage order $K = 1$) but also the lowest consistence order $p = 1$.

The Butcher array for the explicit Euler method is shown in Table 5.5.

The explicit Euler method has the stability function

$$R(z) = 1 + z. \tag{5.41}$$

For example, in the case of real, negative eigenvalues with $z = h\lambda < -2$, the stability function exceeds the unity circle $|R(z)| > 1$ which indicates instability of the explicit Euler method. The recursion for the computation of the states is given by

$$\begin{aligned} \overline{\mathbf{x}}_{[k+1]} &= \overline{\mathbf{x}}_{[k]} + h_k \cdot \boldsymbol{\Gamma}_f \left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{u}}_{[k]}, t_k, h_k\right) \\ &= \overline{\mathbf{x}}_{[k]} + h_k \cdot \mathbf{f} \left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right), \qquad k = 0, \ldots, N_t - 1. \end{aligned}$$

The increment function is independent on h_k, i.e.,

$$\boldsymbol{\Gamma}_f \left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{u}}_{[k]}, t_k, h_k\right) = \mathbf{f} \left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right).$$

Table 5.6 Explicit Heun Method, $K = 2$, $p = 2$

$$\begin{array}{c|cc} c_1 & & \\ c_2 & a_{2,1} & \\ \hline & b_1 & b_2 \end{array} = \begin{array}{c|cc} 0 & & \\ 1 & 1 & \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}.$$

The lack of stability and accuracy limits its use to well behaved problems.

The accuracy of the approximations $\overline{\mathbf{x}}_{[k]}$ depends strongly on the integration step-length. Certainly, to obtain a satisfactorily quality of the approximation the step-length must be sufficiently small compared with the smallest system time constant of interest.

Explicit Heun Discretization

Heun's method is a method with consistence order $p = 2$ and with two stages (stage order $K = 2$). It is also known as explicit trapezoid rule. Table 5.6 shows the Butcher array of the *explicit Heun* method.

The stability function is

$$R(z) = 1 + z + \frac{1}{2}z^2. \tag{5.42}$$

The two-stage recursion is given by

$$\begin{aligned} \overline{\mathbf{x}}_{[k+1]} &= \overline{\mathbf{x}}_{[k]} + h_k \cdot \boldsymbol{\Gamma}_f \left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h_k\right) \\ &= \overline{\mathbf{x}}_{[k]} + \frac{h_k}{2} \cdot (\mathbf{k}_1 + \mathbf{k}_2), \qquad k = 0, \ldots, N_t - 1 \end{aligned}$$

where

$$\begin{aligned} \mathbf{k}_1 &= \mathbf{f}\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right) \\ \mathbf{k}_2 &= \mathbf{f}\left(\left[\overline{\mathbf{x}}_{[k]} + h_k \mathbf{k}_1\right], \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]}\right). \end{aligned}$$

Here and for all following discretization schemes, the increment functions depends on h_k, i.e.,

$$\begin{aligned} &\boldsymbol{\Gamma}_f \left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h_k\right) \\ &= \frac{1}{2} \cdot \left[\mathbf{f}\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right) + \mathbf{f}\left(\left[\mathbf{x}_{[k]} + h_k \mathbf{k}_1\right], \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]}\right)\right]. \end{aligned}$$

Explicit Hermite-Simpson Discretization

Table 5.7 shows the Butcher array of the *explicit Hermite-Simpson* (stage order $K = 3$, consistence order $p = 3$) method.

Table 5.7 Explicit Hermite-Simpson Method, $K = 3$, $p = 3$

$$\begin{array}{c|ccc} c_1 & & & \\ c_2 & a_{2,1} & & \\ c_2 & a_{3,1} & a_{3,2} & \\ \hline & b_1 & b_2 & b_3 \end{array} = \begin{array}{c|ccc} 0 & & & \\ \frac{1}{2} & \frac{1}{2} & & \\ 1 & -1 & 2 & \\ \hline & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \end{array}.$$

The stability function is

$$R(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3. \tag{5.43}$$

The recursion for the computation of the states is given by

$$\begin{aligned} \overline{\mathbf{x}}_{[k+1]} &= \overline{\mathbf{x}}_{[k]} + h_k \cdot \boldsymbol{\Gamma}_f \left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h_k\right) \\ &= \overline{\mathbf{x}}_{[k]} + \frac{h_k}{6} \cdot (\mathbf{k}_1 + 4\mathbf{k}_2 + \mathbf{k}_3), \qquad k = 0, \ldots, N_t - 1 \end{aligned} \tag{5.44}$$

where

$$\begin{aligned} \mathbf{k}_1 &= \mathbf{f}\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right) \\ \mathbf{k}_2 &= \mathbf{f}\left(\left[\overline{\mathbf{x}}_{[k]} + \frac{h_k}{2}\mathbf{k}_1\right], \Xi_k^q\left(\overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, t_k + \frac{h_k}{2}\right), \Xi_k^u\left(\overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k + \frac{h_k}{2}\right)\right) \\ \mathbf{k}_3 &= \mathbf{f}\left(\left[\overline{\mathbf{x}}_{[k]} + h_k \cdot (-\mathbf{k}_1 + 2\mathbf{k}_2)\right], \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]}\right). \end{aligned}$$

Explicit Classical Runge–Kutta Discretization

Table 5.8 shows the Butcher array of the classical Runge–Kutta (stage order $K = 4$, consistence order $p = 4$) method.

Table 5.8 Explicit classical Runge–Kutta Method, $K = 4$, $p = 4$

$$\begin{array}{c|cccc} c_1 & & & & \\ c_2 & a_{2,1} & & & \\ c_3 & a_{3,1} & a_{3,2} & & \\ c_4 & a_{4,1} & a_{4,2} & a_{4,3} & \\ \hline & b_1 & b_2 & b_3 & b_4 \end{array} = \begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}.$$

The stability function is

$$R(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4. \tag{5.45}$$

The recursion for the computation of the states is given by

$$\begin{aligned}\overline{\mathbf{x}}_{[k+1]} &= \overline{\mathbf{x}}_{[k]} + h_k \cdot \boldsymbol{\Gamma}_f \left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h_k\right) \\ &= \overline{\mathbf{x}}_{[k]} + \frac{h_k}{6} \cdot (\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4), \qquad k = 0, \ldots, N_t - 1 \end{aligned} \tag{5.46}$$

where

$$\begin{aligned}\mathbf{k}_1 &= \mathbf{f}\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right) \\ \mathbf{k}_2 &= \mathbf{f}\left(\left[\overline{\mathbf{x}}_{[k]} + \frac{h_k}{2}\mathbf{k}_1\right], \Xi_k^q\left(\overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, t_k + \frac{h_k}{2}\right), \Xi_k^u\left(\overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k + \frac{h_k}{2}\right)\right) \\ \mathbf{k}_3 &= \mathbf{f}\left(\left[\overline{\mathbf{x}}_{[k]} + \frac{h_k}{2}\mathbf{k}_2\right], \Xi_k^q\left(\overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, t_k + \frac{h_k}{2}\right), \Xi_k^u\left(\mathbf{u}_{[k]}, \mathbf{u}_{[k+1]}, t_k + \frac{h_k}{2}\right)\right) \\ \mathbf{k}_4 &= \mathbf{f}\left(\left[\overline{\mathbf{x}}_{[k]} + h_k\mathbf{k}_3\right], \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]}\right).\end{aligned}$$

For the order of consistence $p \geq 5$ there exist no explicit Runge–Kutta methods of order p with $K = p$ stages.

5.6.2 *Implicit Runge–Kutta Schemes*

Radau Discretization

The simplest *Radau* scheme is the *implicit Euler* method, also known as *backward Euler* method and has the stage order $K = 1$. The implicit Euler method has the stability function

$$R(z) = \frac{1}{1 - z}.$$

Applying the Definition 5.8 implies that the implicit Euler method is L-stable. It has the consistence order $p = 1$, i.e., $\mathcal{O}(h)$.

The Butcher array for the implicit Euler method is shown in Table 5.9.

The recursion for the computation of the states is given by

$$\begin{aligned}\overline{\mathbf{x}}_{[k+1]} &= \overline{\mathbf{x}}_{[k]} + h_k \cdot \boldsymbol{\Gamma}_f \left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h_k\right) \\ &= \overline{\mathbf{x}}_{[k]} + h_k\mathbf{k}_1, \qquad k = 0, \ldots, N_t - 1 \end{aligned} \tag{5.47}$$

Table 5.9 Implicit Euler method, $K = 1$, $p = 1$

$$\begin{array}{c|c} c_1 & a_{1,1} \\ \hline & b_1 \end{array} \quad = \quad \begin{array}{c|c} 1 & 1 \\ \hline & 1 \end{array}.$$

Table 5.10 Implicit Radau method, $K = 2$, $p = 3$

$$\begin{array}{c|cc} c_1 & a_{1,1} & a_{1,2} \\ c_2 & a_{2,1} & a_{2,2} \\ \hline & b_1 & b_2 \end{array} \quad = \quad \begin{array}{c|cc} \frac{1}{3} & \frac{5}{12} & -\frac{1}{12} \\ 1 & \frac{3}{4} & \frac{1}{4} \\ \hline & \frac{3}{4} & \frac{1}{4} \end{array}.$$

where

$$\mathbf{k}_1 = \mathbf{f}\left(\left[\overline{\mathbf{x}}_{[k]} + h_k \mathbf{k}_1\right], \Xi_k^q\left(\overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, t_k + h_k\right), \Xi_k^u\left(\overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k + h_k\right)\right). \tag{5.48}$$

Comparing (5.47) and (5.48) let us conclude that

$$\mathbf{k}_1 = \mathbf{f}\left(\overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]}\right).$$

Compared with the explicit Euler integration scheme, the implicit counterpart requires an additional function evaluation $\mathbf{f}(\overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]})$.

Table 5.10 shows the Butcher array of the Radau IIA method with consistence order $p = 3$. With stage order $K = 2$ one obtains the consistence order $p = 3$, which is not possible with any explicit discretization scheme.

The stability function

$$R(z) = \frac{1 + \frac{1}{3}z}{1 - \frac{2}{3}z + \frac{1}{3}\frac{z^2}{2!}}$$

which implies that the Radau IIA(3) is L-stable.

The recursion formula for computation of the states is given by

$$\begin{aligned} \overline{\mathbf{x}}_{[k+1]} &= \overline{\mathbf{x}}_{[k]} + h_k \cdot \boldsymbol{\Gamma}_f\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h_k\right) \\ &= \overline{\mathbf{x}}_{[k]} + h_k \cdot \left(\frac{3}{4}\mathbf{k}_1 + \frac{1}{4}\mathbf{k}_2\right), \qquad k = 0, \ldots, N_t - 1 \end{aligned} \tag{5.49}$$

where

$$\mathbf{k}_1 = \mathbf{f}\left(\left[\overline{\mathbf{x}}_{[k]} + h_k \cdot \left(\frac{5}{12}\mathbf{k}_1 - \frac{1}{12}\mathbf{k}_2\right)\right], \Xi_k^q\left(\overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, t_k + \frac{h_k}{3}\right), \Xi_k^u\left(\overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k + \frac{h_k}{3}\right)\right) \tag{5.50}$$

$$\mathbf{k}_2 = \mathbf{f}\left(\left[\overline{\mathbf{x}}_{[k]} + h_k \cdot \left(\frac{3}{4}\mathbf{k}_1 + \frac{1}{4}\mathbf{k}_2\right)\right], \Xi_k^q\left(\overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, t_k + h_k\right), \Xi_k^u\left(\overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k + h_k\right)\right). \tag{5.51}$$

Comparing (5.49) and (5.51) we can conclude that $\mathbf{k}_2 = \mathbf{f}(\overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]})$ is evaluable at the end point of the integration interval. Inspection of (5.50) reveals that $\mathbf{k}_1$ is given implicitly. But fortunately, we can resolve $\mathbf{k}_1$ in an explicitly evaluable form by substituting $\mathbf{k}_2$ in the recursive equation (5.49) with $\mathbf{f}(\overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]})$. Isolation with respect to $\mathbf{k}_1$ yields $\mathbf{k}_1 = \frac{4}{3h_k} \cdot \left(\overline{\mathbf{x}}_{[k+1]} - \overline{\mathbf{x}}_{[k]}\right) - \frac{1}{3}\mathbf{f}(\overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]})$. This result and $\mathbf{k}_2 = \mathbf{f}(\overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]})$ are now used in stage equation (5.50) which yields an evaluable expression

$$\mathbf{k}_1 = \mathbf{f}\left(\left[\frac{4}{9}\overline{\mathbf{x}}_{[k]} + \frac{5}{9}\overline{\mathbf{x}}_{[k+1]} - \frac{2h_k}{9}\mathbf{f}\left(\overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]}\right)\right], \overline{\mathbf{q}}_{[k]}, \Xi_k^u\left(\overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k + \frac{h_k}{3}\right)\right).$$

The incremental function can now be restated and an easily evaluable form of the Radau IIA method of order 3 is given by (5.49) and the stage equations

$$\mathbf{k}_1 = \mathbf{f}\left(\left[\frac{4}{9}\overline{\mathbf{x}}_{[k]} + \frac{5}{9}\overline{\mathbf{x}}_{[k+1]} - \frac{2h_k}{9}\mathbf{f}\left(\overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]}\right)\right], \overline{\mathbf{q}}_{[k]}, \Xi_k^u\left(\overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k + \frac{h_k}{3}\right)\right)$$
$$\mathbf{k}_2 = \mathbf{f}\left(\overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]}\right).$$

An unique property of Radau methods is the imposition of the collocation condition at only one end of the time interval, typically at $c_K = 1$.

Lobatto Discretization

Lobatto methods are characterized by the usage of the endpoints of each subinterval of integration $[t_k, t_{k+1}]$ also as collocations points, i.e., $c_1 = 0$ and $c_K = 1$. The symbol III is usually associated to Lobatto methods whereas the symbols I and II being reserved for the two types of Radau methods. There are three families of Lobatto methods, called IIIA, IIIB, and IIIC. All are implicit methods with order $2K - 2$. Therefore, choosing the stage $K \geq 3$, one obtains an integration scheme with order of consistence greater than the number of stages.

Two well-known Lobatto methods are the implicit trapezoidal rule and the implicit Hermite-Simpson rule. These integration scheme are often used in practice because of their simplicity.

Table 5.11 shows the Butcher array of the Lobatto IIIA method of order $p = 2$, also known as *implicit trapezoidal rule*.

Table 5.11 Implicit trapezoidal method, $K = 2$, $p = 2$

$$\begin{array}{c|cc} c_1 & a_{1,1} & a_{1,2} \\ c_2 & a_{2,1} & a_{2,2} \\ \hline & b_1 & b_2 \end{array} \quad = \quad \begin{array}{c|cc} 0 & 0 & 0 \\ 1 & \frac{1}{2} & \frac{1}{2} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}.$$

Table 5.12 Lobatto IIIA methods, $K = 3$, $p = 4$

$$\begin{array}{c|ccc} c_1 & a_{1,1} & a_{1,2} & a_{1,3} \\ c_2 & a_{2,1} & a_{2,2} & a_{2,3} \\ c_2 & a_{3,1} & a_{3,2} & a_{3,3} \\ \hline & b_1 & b_2 & b_3 \end{array} \quad = \quad \begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{5}{24} & \frac{1}{3} & -\frac{1}{24} \\ 1 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \\ \hline & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \end{array}.$$

The stability function is

$$R(z) = \frac{1 + \frac{1}{2}z}{1 - \frac{1}{2}z}$$

which implies that the implicit trapezoidal rule is A-stable.

The recursion formula for computation of the states is given by

$$\begin{aligned} \overline{\mathbf{x}}_{[k+1]} &= \overline{\mathbf{x}}_{[k]} + h_k \cdot \boldsymbol{\Gamma}_f \left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h_k \right) \\ &= \overline{\mathbf{x}}_{[k]} + \frac{h_k}{2} \cdot (\mathbf{k}_1 + \mathbf{k}_2), \qquad k = 0, \ldots, N_t - 1 \end{aligned} \tag{5.52}$$

where

$$\mathbf{k}_1 = \mathbf{f} \left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{u}}_{[k]} \right) \tag{5.53}$$

$$\mathbf{k}_2 = \mathbf{f} \left(\overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]} \right). \tag{5.54}$$

Table 5.12 shows the Butcher array of the Lobatto IIIA method of consistence order $p = 4$, also known as *implicit Hermite-Simpson method*.

The stability function is

$$R(z) = \frac{1 + \frac{1}{2}z + \frac{1}{12}z^2}{1 - \frac{1}{2}z + \frac{1}{12}z^2}$$

which implies that the implicit Hermite-Simpson method is A-stable.

The recursion for the computation of the states is given by

$$\begin{aligned}\overline{\mathbf{x}}_{[k+1]} &= \overline{\mathbf{x}}_{[k]} + h_k \cdot \boldsymbol{\Gamma}_f\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h_k\right)\\ &= \overline{\mathbf{x}}_{[k]} + \frac{h_k}{6} \cdot (\mathbf{k}_1 + 4\mathbf{k}_2 + \mathbf{k}_3), \qquad k = 0, \ldots, N_t - 1 \end{aligned} \tag{5.55}$$

where

$$\mathbf{k}_1 = \mathbf{f}\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right) \tag{5.56}$$

$$\begin{aligned}\mathbf{k}_2 = \mathbf{f}\Bigg(&\left[\overline{\mathbf{x}}_{[k]} + h_k \cdot \left(\frac{5}{24}\mathbf{k}_1 + \frac{1}{3}\mathbf{k}_2 - \frac{1}{24}\mathbf{k}_3\right)\right], \Xi_k^q\left(\overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, t_k + \frac{h_k}{2}\right),\\ &\Xi_k^u\left(\overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k + \frac{h_k}{2}\right)\Bigg)\end{aligned} \tag{5.57}$$

$$\mathbf{k}_3 = \mathbf{f}\left(\left[\overline{\mathbf{x}}_{[k]} + h_k \cdot \left(\frac{1}{6}\mathbf{k}_1 + \frac{2}{3}\mathbf{k}_2 + \frac{1}{6}\mathbf{k}_3\right)\right], \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]}\right). \tag{5.58}$$

Again, comparing (5.55) and (5.58) let us conclude that $\mathbf{k}_3 = \mathbf{f}(\overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]})$. That means, we can evaluate $\mathbf{k}_1$ and $\mathbf{k}_3$ at the end points of the integration interval, t_k and t_{k+1}, respectively. Similar to the Radau IIA method the stage equation (5.57) will be reformulated in an easily evaluable form by substituting $\mathbf{k}_2$ and $\mathbf{k}_3$ in stage equation (5.57) with $\frac{3}{2} \cdot \left[\frac{1}{h_k} \cdot \left(\overline{\mathbf{x}}_{[k+1]} - \overline{\mathbf{x}}_{[k]}\right) - \frac{1}{6}\mathbf{k}_1 - \frac{1}{6}\mathbf{k}_3\right]$ and $\mathbf{f}(\overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]})$, respectively. This leads to a simply evaluable expression in a direct collocation scheme.

$$\begin{aligned}\mathbf{k}_2 = \mathbf{f}\Bigg(&\left\{\overline{\mathbf{x}}_{[k]} + \frac{1}{2} \cdot \left(\overline{\mathbf{x}}_{[k+1]} - \overline{\mathbf{x}}_{[k]}\right) + \frac{h_k}{8} \cdot \left[\mathbf{f}\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right) - \mathbf{f}\left(\overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]}\right)\right]\right\},\\ &\Xi_k^q\left(\overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, t_k + \frac{h_k}{2}\right), \Xi_k^u\left(\overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k + \frac{h_k}{2}\right)\Bigg).\end{aligned}$$

Thus, the incremental function (5.55) can now be restated and an evaluable form of the Lobatto IIIA method of order 4 with the stage equations

$$\mathbf{k}_1 = \mathbf{f}\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right) \tag{5.59}$$

$$\begin{aligned}\mathbf{k}_2 = \mathbf{f}\Bigg(&\left\{\overline{\mathbf{x}}_{[k]} + \frac{1}{2} \cdot \left(\overline{\mathbf{x}}_{[k+1]} - \overline{\mathbf{x}}_{[k]}\right) + \frac{h_k}{8} \cdot \left[\mathbf{f}\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right) - \mathbf{f}\left(\overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]}\right)\right]\right\},\\ &\Xi_k^q\left(\overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, t_k + \frac{h_k}{2}\right), \Xi_k^u\left(\overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k + \frac{h_k}{2}\right)\Bigg)\end{aligned} \tag{5.60}$$

$$\mathbf{k}_3 = \mathbf{f}\left(\overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{u}}_{[k+1]}\right). \tag{5.61}$$

A unique property of Lobatto methods is that mesh points coincides with collocation points.

In general, RK methods with large order of consistence and A or L stability yield the best convergence behavior. The explicit Runge–Kutta methods have stability functions (5.41), (5.42), (5.43), and (5.45), which are not A-stable. Thus, for these RK methods the step-length must be carefully chosen to remain within the unit circle, whereas Radau and Lobatto methods exhibit large asymptotic regions which allow large step-lengths to be used. Without formal proof, A and L stable implicit methods are in practice better suited for stiff differential equations compared with explicit methods.

5.7 Remarks for Integration Schemes for Switched System with Discontinuities

As already pointed out in Chap. 3, some automotive systems in practice are described by models which include discontinuities at the switching points. In system representation (5.4)–(5.5), we have assumed that the state variable $\mathbf{x}(\cdot)$ is Lipschitz-continuous and this smoothness assumption has also been required for the derivation of the RK schemes. For switched systems with discontinuities at the switchings

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), q(t), \mathbf{u}(t)), \quad \forall t \in [t_0, t_f] \tag{5.62}$$

$$\mathbf{x}(t_j^+) = \mathbf{x}(t_j^-) + \boldsymbol{\delta}(q(t_j^-), q(t_j^+), \mathbf{x}(t_j^-)) \tag{5.63}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{5.64}$$

this smoothness assumption does not apply any more. At least, we cannot deduce an uniform Lipschitz constant. Please note, similar to the previous sections we use notation $\boldsymbol{\delta}(q^-, q^+, \mathbf{x})$ instead of $\boldsymbol{\delta}_{(q^-,q^+)}(\mathbf{x})$ to define the state jump at a switching.

This theoretical lack avoids the development of integration schemes without knowledge of the switching time instances. Conversely, this means an accurate switching detection is necessary. Especially, frequent switchings requires integration schemes which are able to deal with these effects at low additional computational costs. This makes the numerical treatment of the IVP (5.62)–(5.64) much more challenging than its counterpart without discontinuities.

Fortunately, this problem is already apparent to the theory of numerical solutions of differential equations for many years, without explicitly paying attention to hybrid systems. According to Hairer and Wanner [15], three numerical computation methods are established in dealing with discontinuities:

1. **ignoring the discontinuity**: these concepts rely on the hope that a variable step-length control will handle the discontinuity appropriately;
2. **singularity detection**: these concepts evaluates comparisons of the local error estimates and the step-length; and

3. **employing the switching function**: these concepts stop the integration at the occurrence of the switching event t_j and restart the integration with the right-hand side function $\mathbf{f}(\cdot)$ with the boundary condition $\mathbf{x}(t_j^+) = \mathbf{x}(t_j^-) + \boldsymbol{\delta}(q(t_j^-), q(t_j^+), \mathbf{x}(t_j^-))$.

Singularity detection is also known as *zero-crossing detection* in some simulation packages, e.g., Simulink®.

The first method may cause the solver to take many small steps in the vicinity of a discontinuity to account for these effects to obtain smaller discretization (truncation) errors (Hairer and Wanner [15]). This leads to excessive computation times. The first two methods are standard techniques of ODE solvers packages. The last method is faster and more reliable compared with the other methods and is implemented in hybrid system solvers.

The implementation of the latter one becomes fairly simple for collocation transcriptions if one assumes that a transition in the discrete state can only take place on a sampling instant t_k of the grid $\mathcal{G}_t$. If this can not be assumed, then an implementation of an appropriate step-length control should be considered.

5.8 Consequences of the Discretization to Optimal Control Problems

We have seen that the discretization of the controls and state dynamics is important for obtaining numerical approximations of the solution functions. But, an important question remains open, whether the discrete approximation of an optimal control problem converges to the continuous formulation for $h_k \to 0,\ k = 0, \ldots, N_t - 1$.

This topic has been addressed by Mordukhovich [21, 22] and Gerdts [13], which have shown detailed convergence analysis for the discrete approximation of some classes of purely continuous optimal control problems. Unfortunately, for the specific types of hybrid optimal control problems regarded in this book, this question becomes even more complex and can therefore not be fully answered.

Nevertheless, let us (loosely speaking) infer the following consequences for our hybrid optimal control problems:

- a discrete approximation of an HOCP may not always converge to its continuous counterpart even if the quantization steps for each variable are made infinitely small;
- a local minimum of a discrete approximation of an HOCP may differ from the local minimum of its continuous counterpart; and
- sensitivity results obtained from a discrete approximation of an HOCP may not agree with the sensitivity results from its continuous counterpart.

At first glance, these statements might introduce some theoretical obstacles for numerically solving HOCPs. However, for practical problems convergence analysis under some hypotheses is rather unlikely to be performed successfully. Consequently,

it is reasonable to assume that the discrete approximations converge to their continuous counterparts and the reader should interpret the statements from above as a warning to not overestimate the quality of numerical approximations.

5.9 Bibliographical Notes

Many different integration methods have been proposed over the past decades in an attempt to solve different types of ordinary differential equations accurately including Adams– Bashforth– Moulton, Backward Differentiation Formulae, and Runge–Kutta methods. There is a wealth of excellent textbooks on this subject, among them Hairer and Wanner [15, 16], and Strehmel et al. [27]. They can be classified as one-step, two-step, and multi-step approaches. Descriptions of multi-step methods can be found in Hairer and Wanner [16].

The vast number of papers (e.g., Henrici [18], Butcher [5], Hairer and Wanner [17], etc.) gives the impression that the family of Runge–Kutta methods is properly the most well-known and used method for numerical integration of ordinary differential equations.

RK integration schemes have been investigated for optimal control problems by many authors including Hager [14], Schwartz [26]. Dontchev et al. [7] developed conditions under which a RK discretization of an OCP with control constraints yields a second-order approximation to the continuous-valued controls.

Extensions has been made to approximate higher order differential equations directly by Runge–Kutta– Nystrom methods as proposed by Dormand et al. [8].

Some historical key-points: Carl Runge [24] published his famous paper in 1895 and extended the approximation method of Euler to a more elaborate scheme which was capable of greater accuracy. The idea of Runge was to approximate the solution of the differential equation using improved formulas as the midpoint and trapezoidal rules. The requirement of evaluating the derivative of the function $\mathbf{f}(\cdot)$ a number of times on an integration interval gave the Runge–Kutta methods their characteristic feature.

The paper by Kutta [20] in 1901 extended the analysis of Runge–Kutta methods as far as order 5 including the famous classical Runge–Kutta method shown in Table 5.8. The fifth-order approximation from Kutta had slight errors which has been corrected by Nyström [23].

One of the pioneers of the embedded approach is Fehlberg [10, 11]. He has developed embedded RK formulas with simplifying assumptions which have a small principal truncation term in the lower order formula. The successful development of automatic step-length control strategies was made possible by Fehlberg's work. Further embedded methods were then derived by Sarafyan [25] and England [9].

Chai [6] proposed a local error estimation procedure based on multi-step methods but with less computing time requirement compared with one-step and two-step methods.

References

1. Betts JT (2010) Practical methods for optimal control and estimation using nonlinear programming, 2nd edn. Society for Industrial and Applied Mathematics. doi:10.1137/1:9780898718577
2. Bonnans JF, Laurent-Varin J (2006) Computation of order conditions for symplectic partitioned Runge-Kutta schemes with application to optimal control. Numerische Mathematik 103(1):1–10
3. Büskens C (1998) Optimierungsmethoden und Sensitivitätsanalyse für optimale Steuerprozesse mit Steuer- und Zustandsbeschränkungen. PhD thesis, Universität Münster
4. Butcher JC (1963) Coefficients for the study of Runge-Kutta integration processes. J Aust Math Soc 3(2):185–201
5. Butcher JC (1987) The numerical analysis of ordinary differential equations: Runge-Kutta and general linear methods. Wiley, New York
6. Chai A (1968) Error estimate of a fourth-order Runge-Kutta method with only one initial derivative evaluation. In: Proceedings of the April 30–May 2, 1968, spring joint computer conference, ACM, pp 467–471
7. Dontchev AL, Hager WW, Veliov VM (2000) Second-order Runge-Kutta approximations in control constrained optimal control. SIAM J Numer Anal 38(1):202–226
8. Dormand J, El-Mikkawy M, Prince P (1987) Families of Runge-Kutta-Nystrom formulae. IMA J Numer Anal 7(2):235–250
9. England R (1969) Error estimates for Runge-Kutta type solutions to systems of ordinary differential equations. Comput J 12(2):166–170
10. Fehlberg E (1968) Classical fifth-, sixth-, seventh-, and eighth-order Runge-Kutta formulas with stepwise control. Tech. rep., NASA TR R-287
11. Fehlberg E (1969) Low-order classical Runge-Kutta formulas with stepwise control and their application to some heat transfer problems. Tech. rep., NASA TR R-315
12. Flaig TG (2013) Implicit Runge-Kutta schemes for optimal control problems with evolution equations. arXiv:13110640
13. Gerdts M (2012) Optimal control of ordinary differential equations and differential-algebraic equations. de Gruyter, Berlin
14. Hager WW (2000) Runge-Kutta methods in optimal control and the transformed adjoint system. Numerische Mathematik 87(2):247–282
15. Hairer E, Wanner G (1993) Solving ordinary differential equations i: nonstiff problems, vol 14. Springer
16. Hairer E, Wanner G (1996) Solving ordinary differential equations ii: stiff and differential-algebraic problems, vol 14. Springer
17. Hairer E, Wanner G (1999) Stiff differential equations solved by Radau methods. J Comput Appl Math 111(1):93–111
18. Henrici P (1962) Discrete variable methods in ordinary differential equations. Wiley, New York, 1962:1
19. Kirches C (2011) Fast numerical methods for mixed-integer nonlinear model-predictive control. Springer
20. Kutta W (1901) Beitrag zur näherungweisen Integration totaler Differentialgleichungen. Z Math Phys 46:435–453
21. Mordukhovich B (1978) On difference approximations of optimal control systems. J Appl Math Mech 42(3):452–461
22. Mordukhovich B (2006) Variational analysis and generalized differentiation II. Applications, Grundlehren der mathematischen Wissenschaften. Springer, Berlin
23. Nyström EJ (1925) Über die numerische Integration von Differentialgleichungen. Societas Scientiarum Fennica 50(13)
24. Runge C (1895) Über die numerische Auflösung von Differentialgleichungen. Mathematische Annalen 46(2):167–178
25. Sarafyan D (1966) Error estimation for Runge-Kutta methods through pseudo-iterative formulas. Tech. Rep. Techn. Rep. No 14, Louisiana State University

26. Schwartz AL (1989) Theory and implementation of numerical methods based on Runge-Kutta integration for solving optimal control problems. PhD thesis, University of California at Berkeley
27. Strehmel K, Weiner R, Podhaisky H (2012) Numerik gewöhnlicher Differentialgleichungen: nichtsteife, steife und differential-algebraische Gleichungen. Springer Science & Business Media
28. von Stryk O (1995) Numerische Lösung optimaler Steuerungsprobleme: Diskretisierung, Parameteroptimierung und Berechnung der adjungierten Variablen. Fortschritt-Berichte VDI-Verlag 8
29. Zhang J, Johansson KH, Lygeros J, Sastry S (2001) Zeno hybrid systems. Int J Robust Nonlinear Control 11(5):435–451. doi:10.1002/mc.592

Chapter 6
Dynamic Programming

6.1 Introduction

Dynamic programming (DP) for nonlinear systems was formulated by Bellman [2] on his principle of optimality. Bellman's principle of optimality states that the solution from any intermediate state of the optimal solution to the final state is also optimal. This important fact is exploited in DP by proceeding backwards in time beginning from the final state and dividing the optimal control problem into many small problems. This makes the theoretical foundation relatively easy to understand compared with the much more involved indirect methods. The general algorithm can be stated in a simple form and is easy to apply to purely continuous *optimal control problems* (OCP) and with some minor reformulations it is also well suited for *hybrid optimal control problems* (HOCP) with underlying systems of non-differentiable inputs and dynamics. That means, even if the system exhibits state jumps on a switching or if a switching between subsystems is to be penalized, this does not impair the use of DP. Additionally, the solution is obtained in closed-loop form and provides naturally a feedback control strategy in contrast with indirect and direct methods for optimal control which yield only open-loop solutions. Considering these aspects, it might appear that DP is ideal for almost any OCP in automotive applications. However, there are serious drawbacks referring to "noncausality" and the "curse of dimensionality". The latter one refers to the fact that the computational demands (computing time and memory requirements) grow exponentially with the number of continuous-valued states N_x and continuous-valued controls N_u. Nevertheless, for certain problems, DP is still a convenient and powerful method to obtain solutions of (H)OCPs. Dynamic programming has often been used for benchmark analysis in order to compare the solution of other optimization methods.

T.J. Böhme and B. Frank, *Hybrid Systems, Optimal Control and Hybrid Vehicles*,
Advances in Industrial Control, DOI 10.1007/978-3-319-51317-1_6

6.2 Optimal Control for Continuous Systems

Let us start by considering the simple continuous OCP without state constraints

$$\begin{aligned}\phi\left(\mathbf{u}^*(\cdot)\right) &= \min_{\mathbf{u}(\cdot)\in\mathcal{U}} \phi\left(\mathbf{u}(\cdot)\right) \\ &= m(\mathbf{x}^*(t_f)) + \int_{t_0}^{t_f} l(\mathbf{x}^*(t), \mathbf{u}^*(t))\, \mathrm{d}t \end{aligned} \tag{6.1}$$

subject to

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \tag{6.2}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0. \tag{6.3}$$

The task is to solve (6.1)–(6.3) with a numerical algorithm derived from the dynamic programming principle rather than trying to find analytical solutions of the functions $\mathbf{u}(\cdot)$, $\mathbf{x}(\cdot)$, and $q(\cdot)$. This requires to work with discretizations and hence we obtain a finite dimensional optimal control problem.

Before we start to develop the algorithmic procedure step by step, we make some technical comments to the problem formulation above. In formulation (6.1)–(6.3), no final state conditions $x_i(t_f) = x_{i,f},\ \forall i \in \mathcal{I}_f$ are imposed, where the set $\mathcal{I}_f$ specifies which state is fixed at the endpoint t_f. Yet, this is not a serious limitation, since final state conditions can easily be implemented as soft constraints. Soft constraints as the name suggests must not be exactly satisfied and are allowed to deviate. Usually they are implemented as penalty term that penalizes deviations from the desired values at the final time t_f. The penalization is controlled by choosing appropriate weights $\mathbf{K}_f$ that scale the different constraints. This suggests to use the endpoint function $m(\mathbf{x}(t_f)) = \mathbf{K}_f \cdot \left(\mathbf{x}_{[\mathcal{I}_f]}(t_f) - \mathbf{x}_f\right)$ in the objective function as final state penalty term. The finite dimensional admissible sets $\hat{\mathcal{U}}$ and $\hat{\mathcal{X}}$ are compact and bounded by box constraints. The restriction to compact sets is quite crucial because numerical procedures cannot search for the whole space of U and X.

The first step towards developing a dynamic programming algorithm is to take the value function $V : \mathsf{X} \times [t_0, t_f] \to \mathbb{R}$ from Sect. 4.3

$$V(\mathbf{x}(\cdot), t) = \min_{\mathbf{u}(\cdot)\in\mathcal{U}} \left\{ m(\mathbf{x}(t_f)) + \int_t^{t_f} l(\mathbf{x}(\tau), \mathbf{u}(\tau))\, \mathrm{d}\tau \right\} \tag{6.4}$$

as a cost measure. Applying the principle of optimality from Theorems 4.7 to (6.4) yields

$$V(\mathbf{x}(\cdot), t) = \min_{\mathbf{u}(\cdot)\in\mathcal{U}} \left\{ \int_t^{t_1} l(\mathbf{x}(\tau), \mathbf{u}(\tau))\, \mathrm{d}\tau + V(\mathbf{x}(\cdot), t_1) \right\}. \tag{6.5}$$

From Sect. 4.3, we have learned that the value function $V(\cdot)$ with an optimal trajectory $\mathbf{x}^*(t)$ must satisfy the *Hamilton–Jacobi–Bellman* (HJB) equation

$$-\frac{\partial V}{\partial t}(\mathbf{x}^*(t), t) = \min_{\mathbf{u}(t)\in\hat{\mathcal{U}}} \left\{ l(\mathbf{x}^*(t), \mathbf{u}(t)) + \left(\frac{\partial V}{\partial \mathbf{x}}\right)^T (\mathbf{x}^*(t), t) \cdot \mathbf{f}(\mathbf{x}^*(t), \mathbf{u}(t)) \right\} \tag{6.6}$$

with the boundary condition

$$V(\mathbf{x}^*(t_f), t_f) = m(\mathbf{x}^*(t_f)). \tag{6.7}$$

The reader should be aware that in most cases, the HJB equation does not admit $V(\cdot) \in \mathcal{C}^1$. Then, the value function $V(\cdot)$ in (6.6) and (6.7) is only locally Lipschitz with respect to $\mathbf{x}(\cdot)$; see therefore Sect. 4.3. Remember, $\mathbf{x}^*(\cdot)$ attains a local minimum if the value function $V(\cdot)$ is a unique viscosity subsolution of the HJB equation. This holds under the technical assumptions on the right-hand side function $\mathbf{f}(\cdot)$ of the ODE, the Lagrange term $l(\cdot)$, and the endpoint function $m(\cdot)$ (cf. Liberzon [15]) with: $\mathbf{f}(\cdot)$, $l(\cdot)$, and $m(\cdot)$ are uniformly continuous w.r.t. all arguments, $\partial\mathbf{f}/\partial\mathbf{x}$, $\partial l/\partial\mathbf{x}$, and $\partial m/\partial\mathbf{x}$ are bounded, and the set of admissible controls $\hat{\mathcal{U}}$ is compact. This fact makes the dynamic programming methodology suitable for dealing with nonlinear systems $\mathbf{f}(\cdot)$ in the optimal control problem (6.2) with non-differentiability assumption w.r.t. the continuous-valued controls $\mathbf{u}(\cdot)$.

It is clear, an analytical solution of the value function which satisfies (6.6) and (6.7) will be hard to find but can be approximated via numerical computations by simply applying the principle of optimality to smaller time steps. Thus, let us introduce an equidistant time grid from Sect. 5.1 with

$$0 = t_0 < t_1 < t_2 < \cdots < t_{N_t} = t_f, \qquad \mathcal{G}_t = \left\{t_0, t_1, \ldots, t_{N_t}\right\},$$

then we obtain from (6.5) the value function for one time instant as

$$V(\mathbf{x}(t_k), t_k) = \min_{\mathbf{u}(t)\in\hat{\mathcal{U}}} \left\{ \int_{t_k}^{t_{k+1}} l(\mathbf{x}(\tau), \mathbf{u}(\tau))\, \mathrm{d}\tau + V(\mathbf{x}(t_{k+1}), t_{k+1}) \right\}. \tag{6.8}$$

Discretizing continuous-valued states $\mathbf{x}(\cdot)$ and continuous-valued controls $\mathbf{u}(\cdot)$ in (6.8) on the grid $\mathcal{G}_t$ using any explicit Runge–Kutta scheme let us obtain

$$V(\mathbf{x}_k, t_k) = \min_{\mathbf{u}_k\in\hat{\mathcal{U}}(kh)} \{h \cdot \Gamma_l(\mathbf{x}_k, \mathbf{u}_k, t_k, h) + V(\mathbf{g}(\mathbf{x}_k, \mathbf{u}_k), t_{k+1})\} \tag{6.9}$$

with the boundary condition

$$V(\mathbf{x}_{N_t}, t_{N_t}) = m(\mathbf{x}_{N_t}). \tag{6.10}$$

The consecutive states $\mathbf{x}_{k+1}$ are given by $\mathbf{g} : \mathsf{X} \times \mathsf{U} \to \mathbb{R}^{N_x}$

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{g}(\mathbf{x}_k, \mathbf{u}_k) \\ \mathbf{g}(\mathbf{x}_k, \mathbf{u}_k) &= \mathbf{x}_k + h \cdot \boldsymbol{\Gamma}_f \left(\mathbf{x}_k, \mathbf{u}_k, t_k, h\right), \end{aligned} \tag{6.11}$$

where $\boldsymbol{\Gamma}_f(\cdot)$ is the increment function of the right-hand side of the ODE in (6.2) and h is a fixed step-length for integration.

Equations (6.9)–(6.10) are called the *discrete Bellman equation* or *dynamic programming equation* which is the basis for computer implementation of dynamic programming. Observe that (6.9)–(6.10) evolves backward in time, a property that can be explored in a numerical algorithm.

The increment function of the Lagrangian $\Gamma_l(\cdot)$ can be defined analogously as

$$\Gamma_l \left(\mathbf{x}_k, \mathbf{u}_k, t_k, h\right) = \sum_{l=1}^{K} b_l k_l,$$

where the functions $k_l(\cdot)$ estimate the value of the Lagrange function $l(\cdot)$ in the interior of the interval $[t_k, t_{k+1})$ and are recursively calculated by

$$k_l \left(\mathbf{x}_k, \mathbf{u}_k, t_k, h\right) = l \left(\mathbf{x}_k + h \cdot \sum_{j=1}^{K} a_{l,j} k_j,\ \Xi_k^u \left(\mathbf{u}_k, t_k + c_l h\right) \right)$$

for all stages $1 \leq l \leq K$, where $\Xi_k^u(\cdot)$ is the control function to evaluate intermediate values. The parameters $a_{l,j}$, b_l, and c_l are Butcher array parameters. The optimal continuous-valued control $\mathbf{u}_k^*$ at time instance k which achieves the minimum of the value function (6.9) is

$$\mathbf{u}_k^* = \arg \min_{\mathbf{u}_k \in \hat{\mathcal{U}}(kh)} \left\{ h \cdot \Gamma_l \left(\mathbf{x}_k, \mathbf{u}_k, t_k, h\right) + V\left(\mathbf{g}(\mathbf{x}_k, \mathbf{u}_k), t_{k+1}\right) \right\}. \tag{6.12}$$

As has been mentioned before, an analytical formulation of the value function will only rarely be available. In order to get a numerical approximation of the value function we quantize the continuous-valued states $\mathbf{x}_k$ on $\mathcal{G}_x^{N_x}$ with

$$x^1 < x^2 < \cdots < x^{N_{\mathcal{G}_x}} \quad x^i \in \mathcal{G}_x = \left\{x^1, x^2, \ldots, x^{N_{\mathcal{G}_x}}\right\}$$

using $N_{\mathcal{G}_x}$ values for each dimension (i.e., for each coordinate). Similarly, we quantizes the continuous-valued controls $\mathbf{u}_k$ on $\mathcal{G}_u^{N_u}$ with

$$u^1 < u^2 < \cdots < u^{N_{\mathcal{G}_u}} \quad u^i \in \mathcal{G}_u = \left\{u^1, u^2, \ldots, u^{N_{\mathcal{G}_u}}\right\}$$

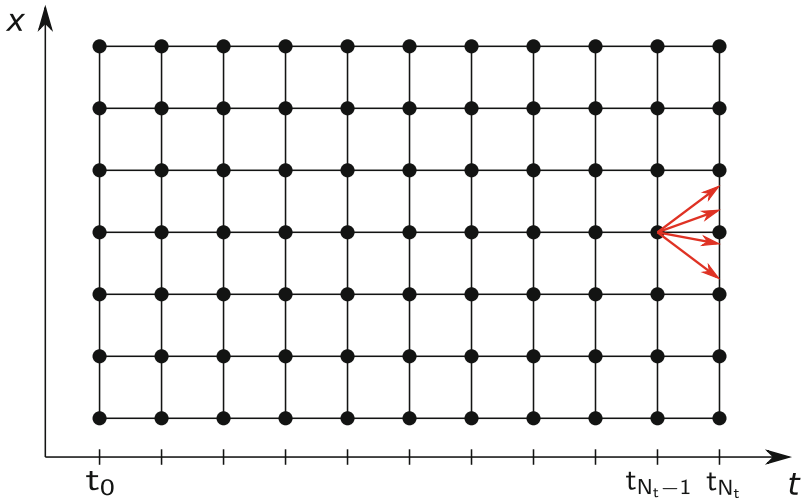


Fig. 6.1 Grid $\mathcal{G}_x^1 \times \mathcal{G}_t$

using $N_{\mathcal{G}_u}$ values for each dimension. The aim of the dynamic programming algorithm is to calculate the value function for every point on the grid $\mathcal{G}_x^{N_x} \times \mathcal{G}_t$ using a backward recursion scheme. For a state with dimension $N_x = 1$, a sketch of this grid is depicted in Fig. 6.1.

The value function at $(\mathbf{x}_{N_t}, t_{N_t})$ can easily be evaluated on the grid $\mathcal{G}_x^{N_x} \times \mathcal{G}_t$, since the boundary condition

$$V\left(\mathbf{x}_{N_t}, t_{N_t}\right) = m\left(\mathbf{x}_{N_t}\right), \quad \forall \mathbf{x}_{N_t} \in \mathcal{G}_x^{N_x}$$

applies. This results in the number of $\#\mathcal{G}_x^{N_x}$ numerical evaluations of the value function $V\left(\mathbf{x}_{N_t}, t_{N_t}\right)$ at time instance t_{N_t}. For the next time instant of the backward calculation, the discrete Bellman equation (6.9) yields

$$V\left(\mathbf{x}_{N_t-1}, t_{N_t-1}\right) = \min_{\mathbf{u}_{N_t-1} \in \hat{\mathcal{U}}((N_t-1)h)} \left\{h \cdot \Gamma_l\left(\mathbf{x}_{N_t-1}, \mathbf{u}_{N_t-1}, t_k, h\right) + V\left(\mathbf{g}(\mathbf{x}_{N_t-1}, \mathbf{u}_{N_t-1}), t_{N_t}\right)\right\}. \tag{6.13}$$

The cost-to-go function $V(\mathbf{x}_{N_t-1}, t_{N_t-1})$ in (6.13) is evaluated on the grid $\mathcal{G}_x^{N_x}$ for all $\mathbf{x}_{N_t-1} \in \mathcal{G}_x^{N_x}$. Herein, the evaluation of the last term $V(\mathbf{g}(\mathbf{x}_{N_t-1}, \mathbf{u}_{N_t-1}), t_{N_t})$ must be performed for all $\mathbf{u}_{N_t-1} \in \mathcal{G}_u^{N_u}$ to each $\mathbf{x}_{N_t-1} \in \mathcal{G}_x^{N_x}$, where the consecutive states $\mathbf{g}(\mathbf{x}_{N_t-1}, \mathbf{u}_{N_t-1})$ are likely to fall between the grid points of $\mathcal{G}_x^{N_x}$. Figure 6.2 illustrates

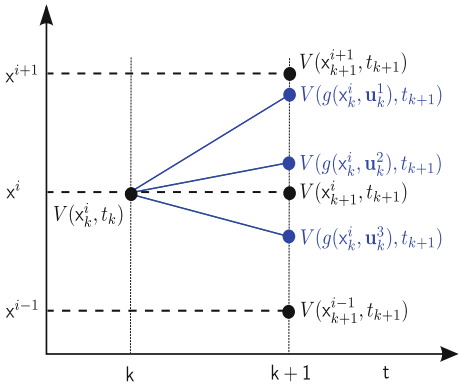


Fig. 6.2 Illustration of the calculation of $V(g(x_k^i, \mathbf{u}_k), t_{k+1})$ on the grid $\mathcal{G}_x^1$ (cf. Guzzella and Sciarretta [11]). The value functions, calculated at the time instance k with x_k^i and the three exemplary continuous-valued controls $\mathbf{u}_k^{1-3}$, displayed with blue filled circles are interpolated using the grid values at $V(x_{k+1}^{i-1}, t_{k+1})$, $V(x_{k+1}^{i}, t_{k+1})$, and $V(x_{k+1}^{i+1}, t_{k+1})$

this problem. Consequently, the term $V(\mathbf{g}(\mathbf{x}_{N_t-1}, \mathbf{u}_{N_t-1}), t_{N_t})$ must be evaluated with an appropriate interpolation scheme, such that the value function can be assumed to be entirely defined over the boundaries of the grid. For an interpolation, we are expecting the value function $V(\mathbf{x}_{k+1}, t_{k+1})$ to be known on the grid points $\mathcal{G}_x^{N_x}$.

There exist several methods of finding the cost-to-go function $V(\mathbf{g}(\mathbf{x}_{N_t-1}, \mathbf{u}_{N_t-1}), t_{N_t})$ on the non-grid point $\mathbf{g}(\mathbf{x}_{N_t-1}, \mathbf{u}_{N_t-1})$: *nearest neighbor interpolation* or *linear interpolation*. The first approach takes the closest value of the state grid $\mathcal{G}_x^{N_x}$ and evaluates the cost-to-go function using this value. The advantage to this approach is its computational speed but it suffers from poor accuracy. The second approach provides good accuracy while the extra computational cost can be made acceptable by an efficient implementation of the interpolation scheme and by choosing an equally spaced state grid $\mathcal{G}_x^{N_x}$. Then, the optimal control $\mathbf{u}_{N_t-1}^*$ can be determined by selecting the continuous-valued controls that minimize $V(\mathbf{g}(\mathbf{x}_{N_t-1}, \mathbf{u}_{N_t-1}), t_{N_t})$ for each point on the grid.

For each $\mathbf{x}_{N_t-1} \in \mathcal{G}_x^{N_x}$, the optimal value of $V(\mathbf{x}_{N_t-1}, t_{N_t-1})$ is saved in the matrix $\mathbf{V}(\mathbf{x}_{N_t-1}, t_{N_t-1})$ of dimension $\mathcal{G}_x^{N_x} \times \mathcal{G}_t$ and the corresponding optimal continuous-valued controls are saved in a similar structure $\mathbf{U}(\mathbf{x}_{N_t-1}, t_{N_t-1})$. With $V(\mathbf{x}_{N_t-1}, t_{N_t-1})$ being defined on the grid, we can move one step backwards to $N_t - 2$ and repeat the same procedure. The algorithm can be summarized as follows:

Algorithm 6.1 Dynamic Programming for OCPs

1: Define $\mathcal{G}_t$, $\mathcal{G}_x$, and $\mathcal{G}_u$
2: **for** $k \leftarrow N_t$ **to** 0 **do**
3: **for all** $\mathbf{x}_j \in \mathcal{G}_x^{N_x}$ **do**
4: **if** $k = N_t$ **then**
5: $\mathbf{V}(\mathbf{x}_j, t_k) \leftarrow m(\mathbf{x}_j)$
6: **else**
7: **for all** $\mathbf{u}_i \in \mathcal{G}_u^{N_u}$ **do**
8: $\mathbf{x}_j^+ \leftarrow \mathbf{g}(\mathbf{x}_j, \mathbf{u}_i)$
9: $C_i \leftarrow h \cdot \Gamma_l\left(\mathbf{x}_j, \mathbf{u}_i, t_k, h\right) + V(\mathbf{x}_j^+, t_{k+1})$
10: **end for**
11: $i^* \leftarrow \arg\min_i C_i$
12: $\mathbf{V}(\mathbf{x}_j, t_k) \leftarrow C_{i^*}$
13: $\mathbf{U}(\mathbf{x}_j, t_k) \leftarrow \mathbf{u}_{i^*}$
14: **end if**
15: **end for**
16: **end for**

The evaluation of $V(\mathbf{x}_j^+, t_{k+1})$ in line 9 of the Algorithm 6.1 requires an interpolation, as it is rather unlikely that the state $\mathbf{x}_j^+$ will be on the grid $\mathcal{G}_x^{N_x}$. If continuous-valued controls and continuous-valued states are constrained by additional conditions different to box constraints, the minimization of the value function must also be further constrained. Therefore, an additional penalty function $P(\mathbf{x}_{k+1}, \mathbf{u}_{k+1})$ for the control and state constraints is introduced and the code in line 9 becomes

$$C_i = h \cdot \Gamma_l\left(\mathbf{x}_j, \mathbf{u}_i, t_k, h\right) + V(\mathbf{x}_j^+, t_{k+1}) + P(\mathbf{x}_j^+, \mathbf{u}_i). \tag{6.14}$$

Once the values and optimal controls $\mathbf{V}(\cdot)$ and $\mathbf{U}(\cdot)$ of the value function on the grid $\mathcal{G}_x^{N_x} \times \mathcal{G}_t$ are defined by Algorithm 6.1. The optimal trajectories $\overline{\mathbf{x}}^*$ and $\overline{\mathbf{u}}^*$ can be recovered for arbitrary starting times $t_{k_{init}} \in \mathcal{G}_t$ and initial states $\overline{\mathbf{x}}^*_{[k_{init}]} \in \text{conv}(\mathcal{G}_x^{N_x})$, where $\text{conv}(\mathcal{G}_x^{N_x})$ denotes the convex hull of the state grid. The vectors $\overline{\mathbf{x}} := \left[\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{N_t}\right] \in \mathbb{R}^{N_x \cdot (N_t+1)}$ and $\overline{\mathbf{u}} := \left[\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N_t-1}\right] \in \mathbb{R}^{N_u \cdot N_t}$ are marked with an overline to indicate the discretization process. The recovering procedure for the optimal trajectories is given by:

Algorithm 6.2 Optimal Trajectory for OCPs

1: Define k_{init}, $\overline{\mathbf{x}}^*_{[k_{init}]}$
2: **for** $k \leftarrow k_{init}$ **to** $N_t - 1$ **do**
3: $\overline{\mathbf{u}}^*_{[k]} \leftarrow \mathbf{U}(\overline{\mathbf{x}}^*_{[k]}, t_k)$
4: $\overline{\mathbf{x}}^*_{[k+1]} \leftarrow \mathbf{g}(\overline{\mathbf{x}}^*_{[k]}, \overline{\mathbf{u}}^*_{[k]})$
5: **end for**

For a given starting point $(\overline{\mathbf{x}}^*_{[k_{init}]}, t_{k_{init}})$, the optimal control trajectory $\overline{\mathbf{u}}^*_{[k]}$ is determined from $\mathbf{U}(\overline{\mathbf{x}}^*_{[k]}, t_k)$. This will usually require an interpolation, as $\overline{\mathbf{x}}^*_{[k]}$ is most likely not on the state grid $\mathcal{G}_x^{N_x}$. Using this control, the consecutive state $\overline{\mathbf{x}}^*_{[k+1]}$ is

computed using (6.11). This is repeated until $k = N_t - 1$. As we can compute an optimal trajectory from arbitrary starting points $(\overline{\mathbf{x}}^*_{[k_{init}]}, t_{k_{init}})$, we obtained a *feedback control law* or *control policy*. Even if, for some reason, the initial states $\overline{\mathbf{x}}^*_{[k_{init}]}$ deviate from the originally planned initial states, then one still knows, how to recover an optimal trajectory from the disturbed initial states, as long as the states are in the interior of the grid $\mathcal{G}_x^{N_x}$.

The dynamic programming methodology is referred to as *deterministic dynamic programming*.

6.3 Optimal Control of Hybrid Systems

Let us now consider a hybrid optimal control problem. To allow for more compact descriptions of the algorithm, we adopt the notation of the vector field $\mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t))$, the Lagrange function $l_{q(t)}(\mathbf{x}(t), \mathbf{u}(t))$, and the jump function $\boldsymbol{\delta}_{(q(t_j^-), q(t_j^+))}(\mathbf{x}(t_j^-))$ to $\mathbf{f}(\mathbf{x}(t), q(t), \mathbf{u}(t))$, $l(\mathbf{x}(t), q(t), \mathbf{u}(t))$, and $\boldsymbol{\delta}(q(t_j^-), q(t_j^+), \mathbf{x}(t_j^-))$, respectively. The HOCP can then be stated as

$$\begin{aligned}
\phi\left(\mathbf{u}^*(\cdot), \varpi^*(\cdot)\right) &= \min_{\mathbf{u}(\cdot)\in\mathcal{U}(q(\cdot)),\ \varpi(\cdot)\in\mathsf{B}(q(\cdot))} \phi\left(\mathbf{u}(\cdot), \varpi(\cdot)\right) \\
&= m(\mathbf{x}^*(t_f)) + \int_{t_0}^{t_f} l(\mathbf{x}^*(t), q^*(t), \mathbf{u}^*(t))\,\mathrm{d}t && (6.15)\\
\dot{\mathbf{x}}(t) &= \mathbf{f}(\mathbf{x}(t), q(t), \mathbf{u}(t)), \quad \text{for a.e. } t \in [t_0, t_f] && (6.16)\\
q(t_j^+) &= q(t_j^-) + \varpi(t_j), \quad t_j \in \Theta_t && (6.17)\\
\mathbf{x}(t_j^+) &= \mathbf{x}(t_j^-) + \boldsymbol{\delta}(q(t_j^-), q(t_j^+), \mathbf{x}(t_j^-)) && (6.18)\\
\mathbf{x}(t_0) &= \mathbf{x}_0. && (6.19)
\end{aligned}$$

The major challenge in optimal control of hybrid systems is the occurrence of the discrete state $q(\cdot)$. Dynamic programming can deal with such complex problems. Analogue to the continuous counterpart dynamic programming must satisfy the HJB equation for the hybrid optimal control problem. To extend the HJB equation to the hybrid case (6.15)–(6.19) let us introduce a family of value functions $V_{q(t)}(\cdot)$, which are viscosity subsolutions of the extended HJB equation

$$\begin{aligned}
&-\frac{\partial V_{q(t)}}{\partial t}(\mathbf{x}^*(t), t) = \\
&\min_{\mathbf{u}(t)\in\hat{\mathcal{U}}(q(t),t),\ \varpi(t)\in\hat{\mathsf{B}}_q} \left\{ l(\mathbf{x}^*(t), q(t), \mathbf{u}(t)) + \left(\frac{\partial V_{q(t)}}{\partial \mathbf{x}}\right)^T (\mathbf{x}^*(t), t) \cdot \mathbf{f}(\mathbf{x}^*(t), q(t), \mathbf{u}(t)) \right\},
\end{aligned} \tag{6.20}$$

the boundary condition

$$V_{q(t)}(\mathbf{x}^*(t_f), t_f) = m(\mathbf{x}^*(t_f)), \tag{6.21}$$

and the state transitions

$$V_{q(t_j^+)}(\mathbf{x}^*(t_j^+), t_j^+) = V_{q(t_j^-)}(\mathbf{x}^*(t_j^-), t_j^-) + \sum_{j=1}^{N_{swt}} \boldsymbol{\pi}_j^T \boldsymbol{\varphi}(\mathbf{x}^*(t_j^-), \mathbf{x}^*(t_j^+)), \tag{6.22}$$

where $\boldsymbol{\varphi}(\mathbf{x}(t_j^-), \mathbf{x}(t_j^+)) = \mathbf{x}(t_j^-) - \mathbf{x}(t_j^+) + \boldsymbol{\delta}(q(t_j^-), q(t_j^+), \mathbf{x}(t_j^-))$ is the rearranged term (6.18) and $\boldsymbol{\pi}_j$ are multipliers. The extension of the HJB equation to hybrid systems follows the stacking principle from Sect. 4.4.

Due to the stacking argumentation the assumptions on the right-hand side function $\mathbf{f}(\mathbf{x}(t), q(t), \mathbf{u}(t))$ of the ODE, the jump function $\boldsymbol{\delta}(q(t_j^-), q(t_j^+), \mathbf{x}(t_j^-))$, the Lagrange term $l(\mathbf{x}(t), q(t), \mathbf{u}(t))$, and the endpoint function $m(\cdot)$ are similar to the continuous counterpart: $\mathbf{f}(\mathbf{x}(t), q(t), \mathbf{u}(t))$, $\boldsymbol{\delta}(q(t_j^-), q(t_j^+), \mathbf{x}(t_j^-))$, $l(\mathbf{x}(t), q(t), \mathbf{u}(t))$, and $m(\cdot)$ are uniformly continuous w.r.t. all arguments, $\partial \mathbf{f}/\partial \mathbf{x}$, $\partial \boldsymbol{\delta}/\partial \mathbf{x}$, $\partial l/\partial \mathbf{x}$, and $\partial m/\partial \mathbf{x}$ are bounded for all locations $q(t) \in \hat{\mathcal{Q}}$, and the set of admissible controls $\hat{\mathcal{U}}(q(t), t)$ is compact. This fact makes the application of the DP to problems with discrete state not really difficult. Actually, DP is often employed to optimize *multistage decision processes*, where a sequence of discrete decisions needs to be optimized to maximize an outcome.

To adapt the algorithm from Sect. 6.2 for hybrid systems, let us pause here to remember the main distinction we made in Chap. 3 between switched and hybrid systems. On the one hand, we have hybrid systems which can choose their locations freely at any time as shown in the left subfigure from Fig. 3.2. We denoted this special class of hybrid systems as a switched system. On the other hand, we have hybrid systems which are indeed restricted on their choice of locations as shown in the right subfigure from Fig. 3.2. In other words, for general hybrid systems we have to choose the discrete controls from an admissible control set that depends on the current discrete state. This requires a much more complex control strategy than for switched systems.

Therefore, let us express the evolution of the discrete state (6.17) at the time grid $\mathcal{G}_t$ as

$$q_{k+1} = q_k + \varpi_k,$$

where q_k is the current discrete state and ϖ_k is the current discrete control that allows for switching between the locations. The admissible set for ϖ_k is a state-dependent set denoted by $\hat{\mathsf{B}}_q = \{g(q^\circ) - g(q) \mid (q, q^\circ) \in \mathcal{B}_q\}$, where the function $g(\cdot)$ enumerates the tuples (q, q°) such that the sets contain signed values. If at a time instant t_k the discrete state q_k is active, then ϖ_k may take only values, such that q_{k+1} is still in $\hat{\mathcal{Q}}$ but only on the allowed locations.

It is then a natural choice to introduce generalized state and control vectors as

$$\mathbf{z}_k = \begin{bmatrix} \mathbf{x}_k \\ q_k \end{bmatrix}, \qquad \mathbf{w}_k = \begin{bmatrix} \mathbf{u}_k \\ \varpi_k \end{bmatrix}.$$

The function $\mathbf{g}(\cdot)$ also needs to be generalized, as it now has to reflect the difference in the generalized state $\mathbf{z}_k$ instead of only $\mathbf{x}_k$. Additionally, it has to take into account the state discontinuities given by the jump function $\boldsymbol{\delta}(\cdot)$ for $\varpi_k \neq 0$:

$$\begin{aligned} \mathbf{z}_{k+1} &= \mathbf{g}(\mathbf{z}_k, \mathbf{w}_k) \\ \mathbf{g}(\mathbf{z}_k, \mathbf{w}_k) &= \begin{bmatrix} \mathbf{x}_k + h \cdot \boldsymbol{\Gamma}_f\left(\mathbf{x}_k, q_k, \mathbf{u}_k, t_k, h\right) + \boldsymbol{\delta}(q_k, q_k + \varpi_k, \mathbf{x}_k) \\ q_k + \varpi_k \end{bmatrix}. \end{aligned}$$

The discrete dynamic programming algorithm can then be implemented by simply adapting the Algorithm 6.1 to the grid of generalized states $\mathcal{G}_z = \mathcal{G}_x^{N_x} \times \hat{\mathcal{Q}}$ instead of $\mathcal{G}_x^{N_x}$ and the enhanced control vector $\mathcal{G}_w(q) = \mathcal{G}_u^{N_u} \times \hat{\mathsf{B}}_q$ instead of $\mathcal{G}_u^{N_u}$. It should be noted that the control grid depends on the current state q and hence a separate grid needs to be created for any $\hat{\mathsf{B}}_q$. Figure 6.3 illustrates the discretization grid $\mathcal{G}_z \times \mathcal{G}_t$ for one continuous-valued state and a two-valued discrete state.

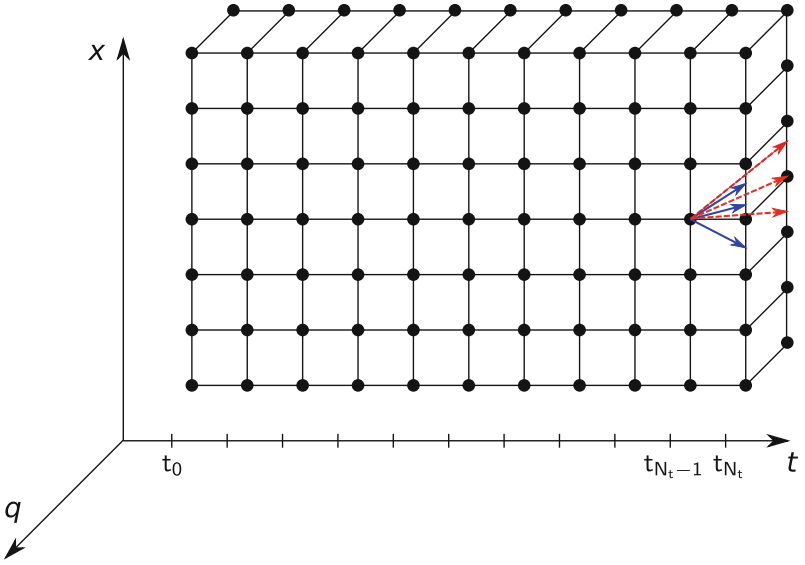


Fig. 6.3 Grid $\mathcal{G}_z \times \mathcal{G}_t$ for a continuous-valued state x_k and two discrete states $q_k \in \{q_1, q_2\}$

Algorithm 6.3 Dynamic Programming for HOCPs

Define $\mathcal{G}_t$, $\mathcal{G}_z$, and $\mathcal{G}_w$
for $k \leftarrow N_t$ **to** 0 **do**
  for all $\mathbf{z}_j \in \mathcal{G}_z$ **do**
    if $k = N_t$ **then**
      $\mathbf{V}(\mathbf{z}_j, t_k) \leftarrow m(\mathbf{z}_j)$
    else
      for all $\mathbf{w}_i \in \mathcal{G}_w(q_j)$ **do**
        $\mathbf{z}_j^+ \leftarrow \mathbf{g}(\mathbf{z}_j, \mathbf{w}_i)$
        $C_i \leftarrow h \cdot \Gamma_l\left(\mathbf{z}_j, \mathbf{w}_i, t_k, h\right) + V(\mathbf{z}_j^+, t_{k+1}) + P(\mathbf{z}_j^+, \mathbf{w}_i)$
      end for
      $i^* \leftarrow \arg\min_i C_i$
      $\mathbf{V}(\mathbf{z}_j, t_k) \leftarrow C_{i^*}$
      $\mathbf{W}(\mathbf{z}_j, t_k) \leftarrow \mathbf{w}_{i^*}$
    end if
  end for
end for

The procedure for determining an optimal trajectory from $(\overline{\mathbf{x}}^*_{[k_{init}]}, t_{k_{init}})$ is also quite similar to the procedure for the purely continuous system. However, an additional step for determining the optimal initial value for the discrete state $\overline{\mathbf{q}}^*_{[k_{init}]}$ has to be added, since this is in general not defined in the problem formulation (6.15)–(6.19). When interpolating from $\mathbf{W}(\overline{\mathbf{z}}^*_{[k]}, t_k)$, it may occur that ϖ_k takes on values that lie in between the grid points, i.e., $\varpi_k \notin \hat{\mathsf{B}}_q$, due to the interpolation between two values that are in $\hat{\mathsf{B}}_q$. To avoid this, a nearest neighbor interpolation should be used for determining ϖ_k.

Algorithm 6.4 Optimal Trajectory for HOCPs

Define k_{init}, $\overline{\mathbf{x}}^*_{[k_{init}]}$
$\overline{\mathbf{q}}^*_{[k_{init}]} \leftarrow \arg\min_q \left\{ V\left([\overline{\mathbf{x}}^*_{[k_{init}]}, q]^T, t_{k_{init}} \right) \right\}$
$\overline{\mathbf{z}}^*_{[k_{init}]} := \left[\overline{\mathbf{x}}^*_{[k_{init}]} \; \overline{\mathbf{q}}^*_{[k_{init}]} \right]^T$
for $k \leftarrow k_{init}$ **to** $N_t - 1$ **do**
  $\overline{\mathbf{w}}^*_{[k]} \leftarrow \mathbf{W}(\overline{\mathbf{z}}^*_{[k]}, t_k)$
  $\overline{\mathbf{z}}^*_{[k+1]} \leftarrow \mathbf{g}(\overline{\mathbf{z}}^*_{[k]}, \overline{\mathbf{w}}^*_{[k]})$
end for

$\overline{\mathbf{z}} := \left[\mathbf{z}_0, \mathbf{z}_1, \ldots, \mathbf{z}_{N_t}\right] \in \mathbb{R}^{(N_x+1)\cdot(N_t+1)}$ and $\overline{\mathbf{w}} := \left[\mathbf{w}_0, \mathbf{w}_1, \ldots, \mathbf{w}_{N_t-1}\right] \in \mathbb{R}^{(N_u+1)\cdot N_t}$ are the vectors of the discretization process.

Finally, it should be remarked that by considering a switched system instead of a general hybrid system the admissible discrete set becomes the set $\hat{\mathsf{B}} := \hat{\mathcal{Q}} \times \hat{\mathcal{Q}} \setminus \left\{ (q, q) \mid q \in \hat{\mathcal{Q}} \right\}$. This simplifies the Algorithm 6.3 with $\mathcal{G}_z = \mathcal{G}_x^{N_x}, \mathcal{G}_w = \mathcal{G}_u^{N_u} \times \hat{\mathcal{Q}}$, and the state and control vectors

$$\mathbf{z}_k = \mathbf{x}_k, \qquad \mathbf{w}_k = \begin{bmatrix} \mathbf{u}_k \\ q_k \end{bmatrix}.$$

The implementation effort for a switched system is almost the same but the execution time is much less.

6.4 Discussion

When implementing a DP algorithm, one issue worth noting is the evaluation of the cost function for infeasible states and controls. Infeasible states and controls are of course infinitely expensive and should therefore have infinite cost. If an infinite cost is used to represent infeasible states, an interpolation between an infinite cost-to-go $V(\mathbf{z}_j^+, t_{k+1})$ and a finite cost-to-go $V(\mathbf{z}_{j+1}^+, t_{k+1})$ leads to an infinite cost-to-go $V(\mathbf{z}_j^+, t_k)$. Consequently, the infeasible region will be artificially increased. This causes substantial errors, if not handled correctly, as pointed out by Guzzella and Sciarretta [11] and Sundström et al. [22].

The use of a big, but finite real value in the code line of Algorithm 6.3 to penalize infeasible states and controls can tackle this problem. But if the value is chosen too large for the penalty term $P(\mathbf{z}_j^+, \mathbf{w}_i)$, numerical errors can occur in the evaluation of C_i. Thus, this value should be chosen as small as possible, but larger than any value of the feasible cost-to-go that could occur. Another strategy is the calculation of the set of reachable states from an initial state, i.e., $\mathcal{R}^t(\mathbf{x}_0)$. The value function then needs to be determined only for those states that are in the reachable set. Boundary line methods explore the reachable set to find the boundary between feasible and infeasible regions (Sundström et al. [22]). A further method that approximates the reachable set is the *Level set* method (Kurzhanskiy and Varaiya [14]).

Care should be taken by choosing the weights $\mathbf{K}_f$ of the penalty term. Too small weights lead to not satisfactorily fulfilled constraints whereas too large weights prevent to find the minimum.

The maximal system dimension that can be treated with DP depends on a couple of factors. All DP algorithms have in common a complexity that scales linearly with the number of time discretization steps N_t along the time range and exponentially with the number of continuous-valued states N_x and continuous-valued controls N_u (cf. Guzzella and Sciarretta [11] and Sontag [20]). The extended dynamic programming algorithm for hybrid optimal control problems has a complexity of

$$\mathcal{O}\left(N_t \cdot \left(\#\mathcal{G}_x^{N_x} \cdot N_q\right) \cdot \left(\#\mathcal{G}_u^{N_u} \cdot N_q\right)\right). \tag{6.23}$$

As can be observed from (6.23), the complexity is less sensitive to the discrete state and control compared with continuous-valued states and continuous-valued controls, where the growth is only linear.

Clearly, a large number of quantizations $\#\mathcal{G}_x$ and $\#\mathcal{G}_u$ in each coordinate will increase the probability of an accurate solution, but will in turn require long

computation times due to the exponential character in N_x and N_u and may be prohibitive ("curse of dimensionality"). Therefore, for a low number of states, DP is a robust method for obtaining optimal trajectories or even an optimal control law (also called *policy*). For systems with $N_x \geq 4$, DP can usually not be used but the computation times can also be unfeasibly high for lower system orders. There are several ways for improving the performance of the algorithm. Frequent interpolation over small datasets or function evaluations of $\mathbf{g}(\cdot)$ and $l(\cdot)$ should be avoided due to computing overhead. The three nested loops in Algorithms 6.1 and 6.3 can bring along a large computing demand. Inspection of the code fragment advise that the inner two loops should be parallelized as they do not exchange information between each other. Parallelization is a performance strategy for accelerating loop execution, if a computer with multiple cores or even a computing cluster is available.

All these methods are suitable for reducing the computing time, in some cases significantly. Yet, they are in general not able to allow for real-time implementation and to compensate for the curse of dimensionality, meaning that for many systems, the option of using DP still has to be discarded.

One advantage of DP over any other optimization technique for optimal control (e.g., indirect and direct methods) is that it provides automatically a feedback control strategy. In actual control implementations, this feedback control strategy is stored in the matrix $\mathbf{U}(\cdot)$, rather than a single optimal control sequence. Another advantage of dynamic programming is that it can be easily modified to account for state and control constraints.

Control affine readers may wonder that we attributed closed-loop solutions as a natural result of DP without saying anything about disturbance. In fact, this is further disadvantage of dynamic programming. Within this chapter we assumed that the disturbance is to be known in advance. This limits the applicability of dynamic programming for real-time applications. More precisely, only in those cases where the disturbances are known a priori dynamic programming can be used in real-time control applications.

Finally, in Bardi and Capuzzo-Dolcetta [1], it is shown for the discrete Bellman equation (6.9)–(6.10), that by performing the limiting process, i.e., $N_t \to \infty$ and $h \to 0$, $V(\mathbf{x}_k, t_k)$ is a viscosity subsolution of the HJB.

The costates $\tilde{\boldsymbol{\lambda}}_k$ for OCPs can be theoretically recovered from the value function using the relationship

$$\tilde{\boldsymbol{\lambda}}_k = \frac{\partial V}{\partial \overline{\mathbf{x}}_{[k]}}(\overline{\mathbf{x}}_{[k]}, t_k), \quad k = 0, \ldots, N_t$$

where the transversality condition is obtained from (6.7) as

$$\tilde{\boldsymbol{\lambda}}_{N_t} = \frac{\partial V}{\partial \overline{\mathbf{x}}_{[N_t]}}(\overline{\mathbf{x}}_{[N_t]}, t_{N_t}) = \frac{\partial m}{\partial \overline{\mathbf{x}}_{[N_t]}}(\overline{\mathbf{x}}_{[N_t]}).$$

The costates of HOCPs are recovered similarly to the continuous counterpart using

$$\tilde{\boldsymbol{\lambda}}_k = \frac{\partial V_{\overline{q}_{[k]}}}{\partial \overline{\mathbf{x}}_{[k]}}(\overline{\mathbf{x}}_{[N_t]}, t_{N_t}), \quad k = 0, \ldots, N_t \tag{6.24}$$

and the transversality condition for the final time is obtained from (6.21) and (6.22) as

$$\tilde{\boldsymbol{\lambda}}_{N_t} = \frac{\partial V_{\overline{q}_{[N_t]}}}{\partial \overline{\mathbf{x}}_{[N_t]}}(\overline{\mathbf{x}}_{[N_t]}, t_{N_t}) = \frac{\partial m}{\partial \overline{\mathbf{x}}_{[N_t]}}(\overline{\mathbf{x}}_{[N_t]}).$$

By backward stepping of (6.24) the following conditions are obtained at state jumps

$$\begin{aligned}
\tilde{\boldsymbol{\lambda}}_{k+1} &= \frac{\partial V_{\overline{q}_{[k+1]}}}{\partial \overline{\mathbf{x}}_{[k+1]}}(\overline{\mathbf{x}}_{[k+1]}, t_{k+1}) = -\boldsymbol{\pi}_j \\
\tilde{\boldsymbol{\lambda}}_k &= \frac{\partial V_{\overline{q}_{[k]}}}{\partial \overline{\mathbf{x}}_{[k]}}(\overline{\mathbf{x}}_{[k]}, t_k) = -\left(\mathbf{I} + \frac{\partial \boldsymbol{\delta}}{\partial \overline{\mathbf{x}}_{[k]}}(\overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{x}}_{[k]})\right)^T \cdot \boldsymbol{\pi}_j
\end{aligned}$$

which yields the known switching condition from Sect. 4.4

$$\tilde{\boldsymbol{\lambda}}_k = \tilde{\boldsymbol{\lambda}}_{k+1} + \left(\frac{\partial \boldsymbol{\delta}}{\partial \overline{\mathbf{x}}_{[k]}}\right)^T (\overline{\mathbf{q}}_{[k]}, \overline{\mathbf{q}}_{[k+1]}, \overline{\mathbf{x}}_{[k]}) \cdot \tilde{\boldsymbol{\lambda}}_{k+1}.$$

6.5 Bibliography

Richard Bellman coined the term "dynamic programming" in the 50s as an umbrella for dealing with multistage decision process problems. Right around the time when the Pontryagin's minimum principle was being developed in the Soviet Union, Bellman and his coworkers concluded that classical calculus of variations are not able to solve modern control problems. One of his great findings is the formulation of the "principle of optimality". A good summary of Richard Bellman's autobiography is presented by Dreyfus [8] which gives some interesting backgrounds about the philosophy of dynamic programming.

There are excellent textbooks on dynamic programming that cover the underlying theory in detail, for instance [3], Bertsekas [4, 5] and of course Bellman [2]. These works cover the algorithm as well as investigations of existence and uniqueness of a solution. The basic algorithm is also very well explained in Kirk [13]. A MATLAB® version is presented by Sundström and Guzzella [21]. The use of dynamic programming for the solution of optimal control problems was encouraged in Branicky and Mitter [7]. In more detail, the incorporation of switching cost was described in Gerdts [10]. The use of dynamic programming for the much more complicated case of optimizing controls of hybrid systems with autonomous switching is the subject

in Rungger [19]. In Hedlund and Rantzer [12] and Xu and Antsaklis [23], the theory of DP is used to derive algorithmic concepts for hybrid optimal control problems.

Dynamic programming can be easily applied to small-scale optimal control problems but becomes quickly infeasible if the state number is larger than $N_x = 4$. Also, optimal control problems with free final time t_f are hard to solve by DP because the length of the problem is not known a priori. To overcome some of these limitations, several adaptations have been proposed. Among them: *iterative dynamic programming* (Luus [16]), *approximate dynamic programming* (Powell [18]), *adaptive dynamic programming* (Murray et al. [17]), and *neuro-dynamic programming* (Bertsekas and Tsitsiklis [6]). The solution obtained with these methods are applicable to a specific class of problems only. In Elbert et al. [9], a DP algorithm is proposed which avoids numerical errors that are due to the interpolation between backward-reachable and non-backward-reachable grid points.

Numerous comparative studies between DP and indirect shooting methods using Pontryagin's minimum principle have been performed by different authors (e.g., Yuan et al. [24]), where DP serves as a benchmark solution.

References

1. Bardi M, Capuzzo-Dolcetta I (1997) Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations. Springer Science and Business Media
2. Bellman RE (2003) Dynamic Programming, republication of the edition published by princeton university press (1957) edn. Dover Publications
3. Bertsekas DP (1976) Dynamic programming and stochastic control. Academic Press, New York
4. Bertsekas DP (1995) Dynamic programming and optimal control, vol 1. Athena Scientific Belmont, MA
5. Bertsekas DP (1995) Dynamic programming and optimal control, vol 2. Athena Scientific Belmont, MA
6. Bertsekas DP, Tsitsiklis JN (1995) Neuro-dynamic programming: an overview. In: Proceedings of the 34th IEEE conference on decision and control, vol 1. IEEE, pp 560–564
7. Branicky MS, Mitter SK (1995) Algorithms for optimal hybrid control. In: Proceedings of the 34th conference on decision and control, pp 2661–2666
8. Dreyfus S (2002) Richard Bellman on the birth of dynamic programming. Oper Res 50(1): 48–51
9. Elbert P, Ebbesen S, Guzzella L (2013) Implementation of dynamic programming for-dimensional optimal control problems with final state constraints. IEEE Trans Control Syst Technol 21(3):924–931
10. Gerdts M (2012) Optimal control of ordinary differential equations and differential-algebraic equations. de Gruyter, Berlin
11. Guzzella L, Sciarretta A (2005) Vehicle propulsion systems. Introduction to modeling and optimization. Springer, Berlin
12. Hedlund S, Rantzer A (1999) Optimal control of hybrid systems. In: Proceedings of the 38th IEEE conference on decision and control, vol 4. IEEE, pp 3972–3977
13. Kirk D (1970) Optimal control theory: an introduction. Englewood Cliffs, N.J., Prentice-Hall
14. Kurzhanskiy A, Varaiya P (2011) Computation of reach sets for dynamical systems. In: Levine WS (ed) The control systems handbook. CRC Press

15. Liberzon D (2012) Calculus of variations and optimal control theory: a concise introduction. Princeton University Press
16. Luus R (2000) Iterative dynamic programming. CRC Press
17. Murray JJ, Cox CJ, Lendaris GG, Saeks R (2002) Adaptive dynamic programming. IEEE Trans Sys Man Cybern Part C: Appl Rev 32(2):140–153
18. Powell WB (2007) Approximate dynamic programming: solving the curses of dimensionality, vol 703. Wiley
19. Rungger M (2011) On the numerical solution of nonlinear and hybrid optimal control problems. PhD thesis, Universität Kassel
20. Sontag ED (1998) Mathematical control theory: deterministic finite dimensional systems, 2nd edn. Springer Science and Business Media
21. Sundström O, Guzzella L (2009) A generic dynamic programming Matlab function. In: Control applications, (CCA) and intelligent control,(ISIC), 2009 IEEE. IEEE, pp 1625–1630
22. Sundström O, Ambühl D, Guzzella L (2010) On implementation of dynamic programming for optimal control problems with final state constraints. Oil and Gas Science and Technology-Revue de lInstitut Français du Pétrole 65(1):91–102
23. Xu X, Antsaklis PJ (2000) A dynamic programming approach for optimal control of switched systems. In: Proceedings of the 39th IEEE conference on decision and control, 2000, vol 2. IEEE, pp 1822–1827
24. Yuan Z, Teng L, Fengchun S, Peng H (2013) Comparative study of dynamic programming and Pontryagins minimum principle on energy management for a parallel hybrid electric vehicle. Energies 6(4):2305–2318

Chapter 7
Indirect Methods for Optimal Control

7.1 Introduction

Indirect methods (IM) are based on first-order necessary conditions of optimality obtained from Pontryagin's minimum principle and try to figure out strong optimal control and state trajectories. Applying these conditions to any given *optimal control problem* (OCP) usually results in a *two-point boundary value problem* (TPBVP) or *multi-point boundary value problem* (MPBVP), which can be solved by an appropriate boundary value solver.

Indirect methods are known to provide highly accurate solutions even for control problems with a large number of continuous-valued controls and continuous states, provided that the method for solving the *boundary value problems* (BVP) converges. A major drawback of indirect methods, which limits their practical applicability, is the requirement to derive analytically—if possible at all—for every problem instance the first-order necessary conditions. This is often cumbersome for high-dimensional systems and requires from the user to have at least some knowledge of optimal control theory to deduce properly the first-order necessary conditions, even if symbolic algebra packages like Maple are used. Indirect methods also suffer from some numerical difficulties, which make them unfeasible in many practical scenarios. For instance, a good initial guess for the approximated costates is needed in order to achieve convergence. The construction of a good initial guess is complicated, since this requires, for example, an estimate of the switching structure of the linear controls. Problems with state constraints might also be intractable for indirect methods. This leads to a large class of problems which are even impossible to be solved by indirect methods and requires alternatives: *direct methods for optimal control*. These solution methods are described in Chap. 8.

Despite these difficulties, IMs can outperform any other solution method in terms of accuracy. Additionally, the theory gives more insight into the structure of the solution, which allows for the calculation of valuable parameter sets for the calibration of controllers. This is demonstrated in Chap. 11 for the calibration of energy

T.J. Böhme and B. Frank, *Hybrid Systems, Optimal Control and Hybrid Vehicles*,
Advances in Industrial Control, DOI 10.1007/978-3-319-51317-1_7

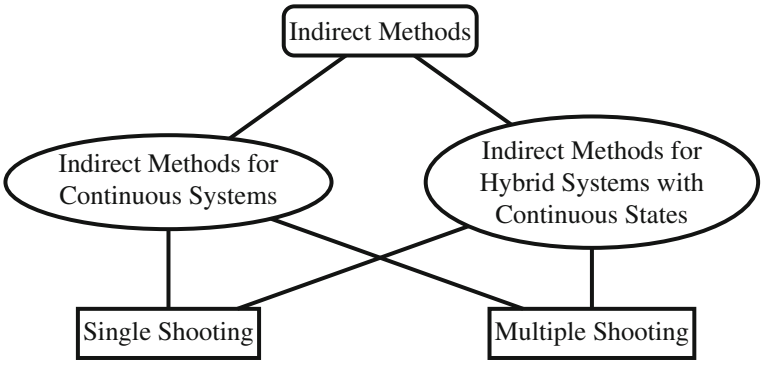


Fig. 7.1 Indirect methods for optimal control. *Elliptical nodes* indicate optimization classes; *rectangular nodes* indicate optimization methods

management systems. It is therefore worthwhile to look at the algorithmic concepts of indirect methods.

For the remaining chapter, we introduce the classification of IMs for switched systems as depicted in Fig. 7.1.

The reader may notice from Fig. 7.1 that indirect methods are just considered to solve optimal control problems for systems without state discontinuities. The reason for limiting the application of indirect methods to these classes becomes clear by inspection of the necessary conditions of hybrid systems in Sect. 4.4. For dealing with hybrid systems with state jumps it is mandatory to decompose the trajectory of the hybrid system into a fixed number of phases or arcs with constant discrete state—the so-called hybrid execution given in Definition 3.7. However, for practical problems it is rather unlikely to know the exact switching structure a priori. One possibility is to employ direct methods to obtain a good guess of the switching structure, which can then be further refined by indirect methods. However, without any knowledge of the switching structure the applicability of indirect methods is limited to hybrid systems with continuous states only.

7.2 Optimal Control for Continuous Systems

7.2.1 Indirect Shooting Method

The first-order necessary conditions for optimal control that are described in the Chap. 4 will now be used to design iterative algorithms for the solutions of optimal control problems. Without loss of generality, we start by considering a continuous OCP of Bolza-type

$$\min_{\mathbf{u}(\cdot)\in\mathcal{U}} \phi\left(\mathbf{u}(\cdot)\right) = m(\mathbf{x}^*(t_f)) + \int_{t_0}^{t_f} l(\mathbf{x}^*(t), \mathbf{u}^*(t))\,\mathrm{d}t \tag{7.1}$$

subject to

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \tag{7.2}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{7.3}$$

$$x_i(t_f) = x_{i,f}, \quad \forall i \in \mathcal{I}_f \tag{7.4}$$

where $\mathbf{x}_0$ and $x_{i,f}$ are the initial and (partially specified) final state values, respectively. The set $\mathcal{I}_f$ specifies which state is fixed at the endpoint t_f. We know from Chap. 4 that one part of the necessary conditions is the minimization of the Hamiltonian.

Therefore, let us define the Hamiltonian function corresponding to (7.1)–(7.4) for all $t \in [t_0, t_f]$ with

$$\mathcal{H}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)) := l(\mathbf{x}(t), \mathbf{u}(t)) + \boldsymbol{\lambda}^T(t)\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \tag{7.5}$$

where $\mathbf{x}(\cdot)$ and $\boldsymbol{\lambda}(\cdot)$ are the continuous states and costates, respectively. For the sake of simplicity, we introduce an enhanced state vector $\mathbf{y}(t) \in \hat{\mathcal{X}} \times \mathbb{R}^{N_x}$, which concatenates the states and costates together as

$$\mathbf{y}(t) = \begin{bmatrix} \mathbf{x}(t) \\ \boldsymbol{\lambda}(t) \end{bmatrix}. \tag{7.6}$$

The time derivative of the enhanced state vector is then given by the respective canonical equations (4.41) and (4.42) in Theorem 4.4. This results in a new system $\mathbf{G} : \hat{\mathcal{X}} \times \mathbb{R}^{N_x} \times \hat{\mathcal{U}} \to \mathbb{R}^{2N_x}$ and is represented by

$$\begin{aligned} \dot{\mathbf{y}}(t) = \mathbf{G}(\mathbf{y}(t), \mathbf{u}(t)) &= \begin{bmatrix} \left(\frac{\partial \mathcal{H}}{\partial \boldsymbol{\lambda}}\right)^T (\mathbf{x}(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)) \\ -\left(\frac{\partial \mathcal{H}}{\partial \mathbf{x}}\right)^T (\mathbf{x}(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)) \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \\ -\left(\frac{\partial l}{\partial \mathbf{x}}\right)^T (\mathbf{x}(t), \mathbf{u}(t)) - \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)^T (\mathbf{x}(t), \mathbf{u}(t)) \cdot \boldsymbol{\lambda}(t) \end{bmatrix}. \end{aligned} \tag{7.7}$$

The trajectory $\mathbf{y}(\cdot)$ of (7.7) is uniquely defined by the initial and final conditions for the continuous states

$$\begin{aligned} \mathbf{x}(t_0) &= \mathbf{x}_0 \\ \mathbf{x}_{[\mathcal{I}_f]}(t_f) &= \mathbf{x}_f \end{aligned}$$

and the transversality condition in Theorem 4.4 for the costates

$$\boldsymbol{\lambda}_{[\mathcal{I}_f^c]}(t_f) = \frac{\partial m}{\partial \mathbf{x}_{[\mathcal{I}_f^c]}(t_f)}(\mathbf{x}(t_f)). \tag{7.8}$$

To calculate a trajectory of $\mathbf{y}(\cdot)$ over a time interval $[t_0, t_f]$, the control trajectory $\mathbf{u}(\cdot)$ is also needed. From the first-order necessary conditions, we know that for almost every $t \in [t_0, t_f]$, an optimal control $\mathbf{u}(\cdot)$ must minimize the Hamiltonian function (4.43). This functional relationship can be reversed by the definition of a new control $\mathbf{u}^*(\cdot)$ as a result of an minimization process of the Hamiltonian function

$$\mathbf{u}^*(t) = \arg \min_{\mathbf{u}(t) \in \hat{\mathcal{U}}(t)} \mathcal{H}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)).$$

Substitution of $\mathbf{u}^*(\cdot)$ in (7.7) yields a BVP with $\tilde{\mathbf{G}} : \hat{\mathcal{X}} \times \mathbb{R}^{N_x} \to \mathbb{R}^{2N_x}$ and is defined as

$$\dot{\mathbf{y}}(t) = \tilde{\mathbf{G}}(\mathbf{y}(t)) = \begin{bmatrix} \mathbf{f}(\mathbf{x}(t), \mathbf{u}^*(t)) \\ -\left(\frac{\partial l}{\partial \mathbf{x}}\right)^T (\mathbf{x}(t), \mathbf{u}^*(t)) - \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)^T (\mathbf{x}(t), \mathbf{u}^*(t)) \cdot \boldsymbol{\lambda}(t) \end{bmatrix} \tag{7.9}$$

with endpoint values specified at the start t_0 and the end t_f of the trajectories. This type of boundary value problem is called a TPBVP. If the endpoint values of the continuous states are partially or completely unspecified, then the transversality condition (7.8) applies for the unspecified final state values and one has a TPBVP with

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{7.10}$$

$$x_i(t_f) = x_{i,f}, \qquad \forall i \in \mathcal{I}_f \tag{7.11}$$

$$\lambda_j(t_f) = \frac{\partial m}{\partial x_j(t_f)}(\mathbf{x}(t_f)), \qquad \forall j \in \mathcal{I}_f^c \tag{7.12}$$

where the complementary set is defined as $\mathcal{I}_f^c = \{1, \ldots, N_x\} \setminus \mathcal{I}_f$.

A special case arises if all endpoint values of the continuous states $\mathbf{x}(t_f)$ are completely prescribed. Then, the TPBVP has Dirichlet condition

$$\mathbf{x}(t_0) = \mathbf{x}_0$$
$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f.$$

A popular methodology to solve a BVP is the shooting method. The idea is to treat the BVP as an *initial value problem* (IVP) and begin the integration at t_0 of the BVP and "shoot" to the other end at t_f using an initial value solver until the boundary condition at t_f converges to its correct value. The reason for the usage of an initial value solver becomes clear by inspection of the condition (7.3) in the problem formulation. The initial states $\mathbf{x}(t_0)$ are assumed to be given, but the initial costates are completely

unknown. Indeed, we have no condition that gives us information about the initial costate values. Thus, we can only make a guess of the initial costates, which will be denoted as $\hat{\boldsymbol{\lambda}}$.

From now on, keeping in mind that we can only guess the costate's values, the initial value of the enhanced state vector is given by

$$\mathbf{y}(t_0) = \begin{bmatrix} \mathbf{x}_0 \\ \hat{\boldsymbol{\lambda}} \end{bmatrix}. \tag{7.13}$$

The boundary conditions for the enhanced state vector can be combined in a vector function $\boldsymbol{\Upsilon}(\cdot)$ using (7.10)–(7.12) as follows:

$$\boldsymbol{\Upsilon}\left(\mathbf{y}(t_f)\right) = \begin{bmatrix} \mathbf{x}_{[\mathcal{I}_f]}(t_f) - \mathbf{x}_f \\ \boldsymbol{\lambda}_{[\mathcal{I}_f^c]}(t_f) - \dfrac{\partial m}{\partial \mathbf{x}_{[\mathcal{I}_f^c]}(t_f)}(\mathbf{x}(t_f)) \end{bmatrix}. \tag{7.14}$$

In order to fulfill the boundary conditions the function (7.14) must approach zero, which in turn requires the solution of the IVP (7.9) and (7.13) for each function evaluation with guessed initial costates $\hat{\boldsymbol{\lambda}}$.

Consequently, the main problem consists in finding iteratively an estimated $\hat{\boldsymbol{\lambda}}$ that solves the nonlinear equation $\boldsymbol{\Upsilon}(\cdot) = \mathbf{0}$ up to a desired exactness. This can be done by numerical procedures for solving nonlinear equations, such as Newton-type methods.

For the numerical procedure let us introduce an equidistant time grid from Sect. 5.1

$$0 = t_0 < t_1 < t_2 < \cdots < t_{N_t} = t_f \qquad \mathcal{G}_t = \left\{t_0, t_1, \ldots, t_{N_t}\right\} \tag{7.15}$$

where

$$h = t_k - t_{k-1}, \qquad k = 1, \ldots, N_t$$

is the corresponding constant step-length for the underlying grid $\mathcal{G}_t$. A constant time grid is not mandatory, since in general, boundary value solvers work with variable step-lengths. However, for the sake of convenience we use a constant step-length.

Discretizing the extended states $\mathbf{y}(\cdot)$ and continuous-valued controls $\mathbf{u}(\cdot)$ on the grid $\mathcal{G}_t$ using any explicit Runge–Kutta scheme from Chap. 5 yields the one-step quadrature formula

$$\overline{\mathbf{y}}_{[k+1]} = \overline{\mathbf{y}}_{[k]} + h \cdot \boldsymbol{\Gamma}_{\tilde{G}}\left(\overline{\mathbf{y}}_{[k]}, \overline{\mathbf{u}}^*_{[k]}, t_k, h\right) \tag{7.16}$$

where $\boldsymbol{\Gamma}_{\tilde{G}}(\cdot)$ is the increment function of $\tilde{\mathbf{G}}(\cdot)$. The vectors $\overline{\mathbf{y}} := \left[\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{N_t}\right] \in \mathbb{R}^{2N_x \cdot (N_t+1)}$ and $\overline{\mathbf{u}} := \left[\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N_t-1}\right] \in \mathbb{R}^{N_u \cdot N_t}$ are assembled with the discretization of $\mathbf{y}(\cdot)$ and $\mathbf{u}(\cdot)$. Then, before each evaluation of the function $\boldsymbol{\Upsilon}(\cdot)$, an IVP with (7.16) and

$$\overline{\mathbf{y}}_{[0]} = \begin{bmatrix} \mathbf{x}_0 \\ \hat{\boldsymbol{\lambda}} \end{bmatrix}$$

has to be solved, where $\mathbf{x}_0$ and $\hat{\boldsymbol{\lambda}}$ are the initial values.

Before taking an integration step, the continuous-valued controls have to be determined by point-wise minimization of the discrete Hamiltonian,

$$\overline{\mathbf{u}}^*_{[k]} = \arg \min_{\overline{\mathbf{u}}_{[k]} \in \tilde{\mathcal{U}}(kh)} \mathcal{H}(\overline{\mathbf{y}}_{[k]}, \overline{\mathbf{u}}_{[k]}) = l\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right) + \overline{\boldsymbol{\lambda}}^T_{[k]} \cdot \boldsymbol{\Gamma}_f\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{u}}_{[k]}, t_k, h\right)$$

where $\boldsymbol{\Gamma}_f(\cdot)$ is the increment function of the right-hand side function $\mathbf{f}(\cdot)$ of the ODE. In order to solve this minimization problem, let us define a constrained nonlinear programming problem

$$\begin{aligned} \min_{\overline{\mathbf{u}} \in \mathbb{R}^{N_u \cdot N_t}} \quad & \mathcal{H}(\overline{\mathbf{y}}, \overline{\mathbf{u}}) \\ \text{subject to} \quad & \mathbf{c}_{\overline{u}}(\overline{\mathbf{u}}) \leq \mathbf{0}. \end{aligned} \tag{7.17}$$

The constrained optimization problem (7.17) can efficiently be solved using sequential quadratic programming (SQP) (see Chap. 2). The Hamiltonian might have several local minima. As a consequence, the Hamiltonian should first be resolved on a rough grid to determine an appropriate starting point for the SQP optimization. Algorithmically, the main function of the indirect single shooting method can be summarized in the following algorithm:

Algorithm 7.1 Main Function for Indirect Single Shooting for Continuous OCPs

Require: $\hat{\boldsymbol{\lambda}}$

1: $\overline{\mathbf{y}}_{[0]} \leftarrow \left[\mathbf{x}_0, \hat{\boldsymbol{\lambda}}\right]^T$
2: **for** $k \leftarrow 0$ **to** $N_t - 1$ **do**
3: $\quad \overline{\mathbf{u}}^*_{[k]} \leftarrow \arg \min_{\overline{\mathbf{u}}_{[k]} \in \tilde{\mathcal{U}}(kh)} \mathcal{H}\left(\overline{\mathbf{x}}_{[k]}, \overline{\boldsymbol{\lambda}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right)$
4: $\quad \overline{\mathbf{y}}_{[k+1]} \leftarrow \overline{\mathbf{y}}_{[k]} + h \cdot \boldsymbol{\Gamma}_{\tilde{G}}\left(\overline{\mathbf{y}}_{[k]}, \overline{\mathbf{u}}^*_{[k]}, t_k, h\right)$
5: **end for**
6: **return** $\overline{\mathbf{y}}$ and $\begin{bmatrix} \overline{\mathbf{x}}_i^{[N_t]} - x_{i,f} \\ \overline{\boldsymbol{\lambda}}_j^{[N_t]} - \dfrac{\partial m}{\partial \overline{\mathbf{x}}_j^{[N_t]}}(\overline{\mathbf{x}}_{[N_t]}) \end{bmatrix}$

Figure 7.2 illustrates some iterations of an indirect shooting algorithm for a simple system with only one continuous state.

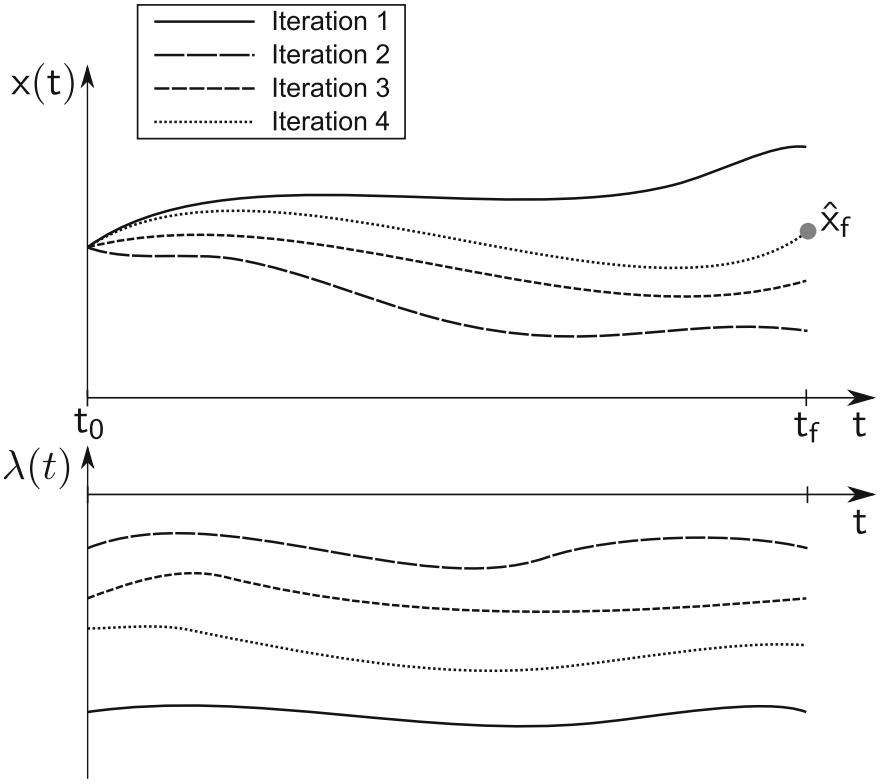


Fig. 7.2 Sketch of the iterations of the indirect shooting approach for a system with dimension $N_x = 1$

7.2.2 Indirect Multiple Shooting Method

While a simple shooting method is appealing due to its simplicity, the numerical solution of the canonical equations may be inaccurate due to the strong dependence on the initial guess of $\hat{\boldsymbol{\lambda}}$. If this guess is far from a solution that satisfies the boundary conditions (7.14), the trajectories $\boldsymbol{\lambda}(\cdot)$ and $\mathbf{x}(\cdot)$ can reach extreme values that are not desired and that imply considerable numerical difficulties in the solution of the nonlinear equation. This property of indirect shooting can be explained by the Hamiltonian's divergence (cf. Rao [16]). Taking $\mathcal{H} = \boldsymbol{\lambda}(t)\mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))$ and calculating the Gateaux derivatives w.r.t. $\mathbf{x}(\cdot)$ and $\boldsymbol{\lambda}(\cdot)$ yields

$$\frac{\partial \mathcal{H}}{\partial x_i}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)) = \boldsymbol{\lambda}(t)\frac{\partial \mathbf{f}}{\partial x_i}(\mathbf{x}(t), \mathbf{u}(t)) \tag{7.18}$$

$$\frac{\partial \mathcal{H}}{\partial \lambda_i}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)) = f_i(\mathbf{x}(t), \mathbf{u}(t)). \tag{7.19}$$

Then, apply the divergence operator to the vector entries (7.18) and (7.19) yields

$$\begin{aligned}
&\sum_{i=1}^{N_x} \frac{\partial}{\partial \lambda_i}\left(\frac{\partial \mathcal{H}}{\partial x_i}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \mathbf{u}(t))\right) + \frac{\partial}{\partial x_i}\left(\frac{\partial \mathcal{H}}{\partial \lambda_i}(\mathbf{x}(t), \boldsymbol{\lambda}(t), \mathbf{u}(t))\right) \\
&= \sum_{i=1}^{N_x} \frac{\partial f_i}{\partial x_i}(\mathbf{x}(t), \mathbf{u}(t)) - \frac{\partial f_i}{\partial x_i}(\mathbf{x}(t), \mathbf{u}(t)) \qquad (7.20) \\
&= 0.
\end{aligned}$$

The divergence of the Hamiltonian in (7.20) is constant, meaning that the vectors in a neighborhood of the optimal solution diverge equally with the same rate. This property implies that errors made in the unknown initial costates $\hat{\boldsymbol{\lambda}}$ will be amplified as the dynamics are integrated and causes that the solution is highly sensitive with respect to $\hat{\boldsymbol{\lambda}}$. That means, the longer the time interval of integration, the lower is the accuracy of the solution, if solvable at all.

A method that aims at reducing this sensitivity of the trajectories on the guess of the initial costates is the *indirect multiple shooting method* (Bulirsch [7], Keller [11] and Osborne [15]). The basic idea of this method is to extend the indirect single shooting method by breaking the numerical integration into several smaller intervals and connecting these by additional conditions such that the complete solution trajectory $\mathbf{y}(\cdot)$ of the multiple shooting procedure is continuous again. The partitioning into smaller integration intervals improves the accuracy considerably. The basic idea is sketched in Fig. 7.3.

Instead of solving the IVP over the entire time interval $t \in [t_0, t_f]$, it is solved over several partial intervals of smaller size,

$$\overline{\mathbf{y}}_{[K_j]} = \mathbf{Y}_j \qquad (7.21)$$

$$\overline{\mathbf{y}}_{[k+1]} = \overline{\mathbf{y}}_{[k]} + h \cdot \boldsymbol{\Gamma}_G\left(\overline{\mathbf{y}}_{[k]}, \overline{\mathbf{u}}^*_{[k]}, t_k, h\right), \quad k = K_j, \ldots, K_{j+1} - 1 \qquad (7.22)$$

each of them having its own initial values $\mathbf{Y}_j$. The intervals are accessed by shooting nodes $K_1, \ldots, K_{N_d}$ on the time grid that are more or less uniformly spread over the entire time interval. $K_1, \ldots, K_{N_d}$ are called *multiple shooting nodes* and are assumed for simplicity to be integer multiples of the time discretization $t_k \in \mathcal{G}_t$.

The shooting grid can then be defined based on the underlying time discretization $\mathcal{G}_t$ as

$$t_0 = t_{K_1} < t_{K_2} < \cdots < t_{K_{N_d}} = t_f, \quad \mathcal{G}_{sh} = \left\{t_{K_1}, t_{K_2}, \ldots, t_{K_{N_d}}\right\}, \quad \mathcal{G}_{sh} \subset \mathcal{G}_t \qquad (7.23)$$

where

$$\begin{aligned}
&K_1 < K_2 < \cdots < K_{N_d} \\
&K_j \in \{1, \ldots, N_t - 1\}, \quad j = 2, \ldots, N_d - 1.
\end{aligned}$$

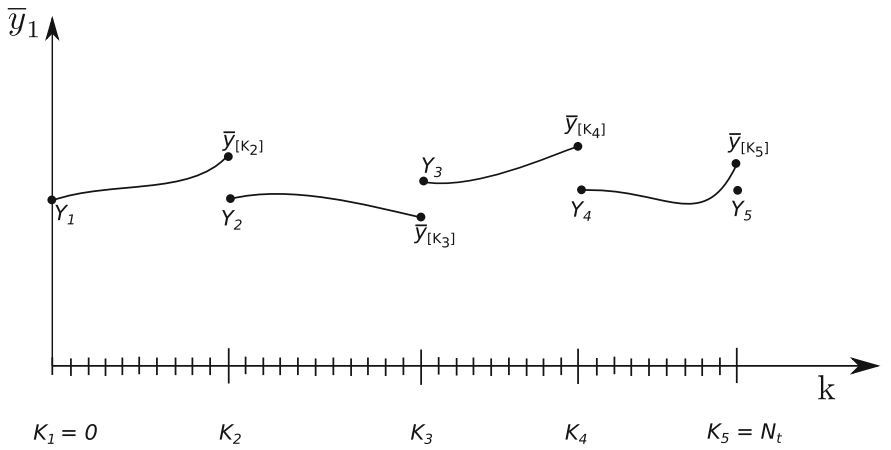


Fig. 7.3 Sketch of the indirect multiple shooting approach by depicting only the first entry of the enhanced state vector $\overline{\mathbf{y}} = \left[\overline{\mathbf{y}}_1,\ \overline{\mathbf{y}}_2\right]^T = \left[\overline{\mathbf{x}},\ \overline{\boldsymbol{\lambda}}\right]^T$. The fine grid is $\mathcal{G}_t$, the nodes $K_1, K_2, \ldots, K_5$ are integer multiples of the fine grid $\mathcal{G}_t$ and form the multiple shooting grid $\mathcal{G}_{sh}$

The first node K_1 and the last node K_{N_d} of the shooting time grid $\mathcal{G}_{sh}$ are fixed and cannot be chosen. They are set to $K_1 = 0$ and $K_{N_d} = N_t$.

For each shooting interval, a corresponding initial value $\mathbf{Y}_j$ is defined, such that the IVPs (7.21) and (7.22) can be solved in the time interval $[t_{K_j}, t_{K_{j+1}}]$ between two consecutive shooting nodes K_j, K_{j+1} independently from the other intervals. To obtain a continuous trajectory $\overline{\mathbf{y}}$ *matching conditions* are additionally required. These conditions demand that the end of any partial trajectory $\overline{\mathbf{y}}_{[K_j]}$ coincides with the initial values $\mathbf{Y}_j$ of the following trajectory.

$\mathbf{Y}_1$ and $\mathbf{Y}_{N_d}$ are boundaries and have to fulfill different conditions. $\mathbf{Y}_1$ contains the initial states $\overline{\mathbf{x}}_{[0]} = \mathbf{x}(t_0)$ and the guessed initial costates $\hat{\boldsymbol{\lambda}}_1 = \boldsymbol{\lambda}(t_0)$, whereas $\mathbf{Y}_{N_d}$ has to fulfill the boundary and transversality conditions for the final state $\overline{\mathbf{y}}_{[N_t]}$. This results in

$$\mathbf{Y}_1 = \begin{bmatrix} \overline{\mathbf{x}}_{[0]} \\ \hat{\boldsymbol{\lambda}}_1 \end{bmatrix}$$

and

$$\mathbf{Y}_{N_d} = \boldsymbol{\Upsilon}(\overline{\mathbf{y}}_{[N_t]}).$$

The matching conditions at the shooting nodes $K_2, \ldots, K_{N_d-1}$ must fulfill

$$\overline{\mathbf{y}}_{[K_j]} - \mathbf{Y}_j = \mathbf{0}, \quad j = 2, \ldots, N_d - 1$$

where $\mathbf{Y}_j$ is given as

$$\mathbf{Y}_j = \begin{bmatrix} \hat{\mathbf{x}}_j \\ \hat{\boldsymbol{\lambda}}_j \end{bmatrix}$$

and $\hat{\mathbf{x}}_j$ and $\hat{\boldsymbol{\lambda}}_j$ are the initial states and initial costates of the shooting intervals, respectively, that have to be guessed. The matching and the boundary conditions are concatenated into one vector $\boldsymbol{\Lambda}$ as

$$\boldsymbol{\Lambda} = \begin{bmatrix} \boldsymbol{\theta}_1 \\ \boldsymbol{\theta}_2 \\ \vdots \\ \boldsymbol{\theta}_{N_d-2} \\ \boldsymbol{\Upsilon}(\overline{\mathbf{y}}_{[N_t]}) \end{bmatrix}$$

with the sub-vectors

$$\begin{aligned} \boldsymbol{\theta}_1 &= \overline{\mathbf{y}}_{[K_2]} - \mathbf{Y}_2 \\ \boldsymbol{\theta}_2 &= \overline{\mathbf{y}}_{[K_3]} - \mathbf{Y}_3 \\ &\vdots \\ \boldsymbol{\theta}_{N_d-2} &= \overline{\mathbf{y}}_{[K_{N_d-1}]} - \mathbf{Y}_{N_d-1}. \end{aligned}$$

A solution implies finding iteratively estimated initial conditions

$$\left[\mathbf{Y}_1^{[N_x+1:2N_x]}, \ldots, \mathbf{Y}_{N_d-1} \right] \tag{7.24}$$

to solve $\boldsymbol{\Lambda} = \mathbf{0}$ up to a desired exactness. The main function for evaluating the vector $\boldsymbol{\Lambda}$ is based on an extended version of Algorithm 7.1:

Algorithm 7.2 Main Function for Indirect Multiple Shooting for Continuous OCPs

Require: $\mathbf{Y}$
1: **for** $j \leftarrow 1$ **to** $N_d - 1$ **do**
2: $\quad \overline{\mathbf{y}}_{[K_j]} \leftarrow \mathbf{Y}_j$
3: $\quad$ **for** $k \leftarrow K_j$ **to** $K_{j+1} - 1$ **do**
4: $\quad\quad \overline{\mathbf{u}}^*_{[k]} \leftarrow \arg \min_{\overline{\mathbf{u}}_{[k]} \in \mathcal{U}(kh)} \mathcal{H}\left(\overline{\mathbf{x}}_{[k]}, \overline{\boldsymbol{\lambda}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right)$
5: $\quad\quad \overline{\mathbf{y}}_{[k+1]} \leftarrow \overline{\mathbf{y}}_{[k]} + h \cdot \boldsymbol{\Gamma}_G\left(\overline{\mathbf{y}}_{[k]}, \overline{\mathbf{u}}^*_{[k]}, t_k, h\right)$
6: $\quad$ **end for**
7: $\quad \boldsymbol{\theta}_j = \overline{\mathbf{y}}_{[K_{j+1}]} - \mathbf{Y}_{j+1}$
8: **end for**
9: evaluate $\boldsymbol{\Upsilon}(\overline{\mathbf{y}}_{[N_t]})$ using (7.14)
10: assemble $\boldsymbol{\Lambda}$
11: **return** $\overline{\mathbf{y}}$, $\overline{\mathbf{u}}$, and $\boldsymbol{\Lambda}$

The indirect multiple shooting method is much more robust than the single shooting method but the problem of finding initial values $\mathbf{Y}$ is still difficult, even though the sensitivity towards the initial guess has been reduced. In the cases, where the algorithm is still not applicable, direct methods, as will be described in Chap. 8, can often obtain a solution in a much more robust manner.

7.3 Optimal Control for Hybrid Systems

Let us consider a *switched optimal control problem* (SOCP) of the type

$$\min_{\mathbf{u}(\cdot)\in\mathcal{U}(q(\cdot)),\ q(\cdot)\in\mathcal{Q}} \phi\left(\mathbf{u}(\cdot), q(\cdot)\right) = m(\mathbf{x}^*(t_f)) + \int_{t_0}^{t_f} l_{q^*(t)}(\mathbf{x}^*(t), \mathbf{u}^*(t))\,\mathrm{d}t \tag{7.25}$$

subject to

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)) \tag{7.26}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{7.27}$$

$$x_i(t_f) = x_{i,f}, \quad \forall i \in \mathcal{I}_f. \tag{7.28}$$

Again, we define the Hamiltonian function corresponding to the SOCP (7.25)–(7.28) as

$$\mathcal{H}(\mathbf{x}(t), q(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)) := l_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)) + \boldsymbol{\lambda}^T(t)\mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)). \tag{7.29}$$

Adapting the augmented system (7.7) to the canonical equations of the SOCP with the vector field $\mathbf{f}_{q(t)}(\cdot)$ and the Hamiltonian function $\mathcal{H}(\cdot)$ yields the system $\mathbf{K} : \mathsf{X} \times \mathbb{R}^{N_x} \times \hat{\mathcal{Q}} \times \mathsf{U} \to \mathbb{R}^{2N_x}$ represented by

$$\begin{aligned}\dot{\mathbf{y}}(t) = \mathbf{K}(\mathbf{y}(t), q(t), \mathbf{u}(t)) &= \begin{bmatrix} \left(\dfrac{\partial \mathcal{H}}{\partial \boldsymbol{\lambda}}\right)^T(\mathbf{x}(t), q(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)) \\ -\left(\dfrac{\partial \mathcal{H}}{\partial \mathbf{x}}\right)^T(\mathbf{x}(t), q(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)) \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)) \\ -\left(\dfrac{\partial l_{q(t)}}{\partial \mathbf{x}}\right)(\mathbf{x}(t), \mathbf{u}(t)) - \left(\dfrac{\partial \mathbf{f}_{q(t)}}{\partial \mathbf{x}}\right)^T(\mathbf{x}(t), \mathbf{u}(t)) \cdot \boldsymbol{\lambda}(t) \end{bmatrix}. \end{aligned} \tag{7.30}$$

The system (7.30) can now be easily transformed to a TPBVP using the Hamiltonian minimum condition

$$(q^*(t), \mathbf{u}^*(t)) = \arg \min_{q(t)\in\hat{\mathcal{Q}},\ \mathbf{u}(t)\in\hat{\mathcal{U}}(q(t),t)} \mathcal{H}(\mathbf{x}^*(t), q(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)) \tag{7.31}$$

from Theorem 4.8. Substituting the optimal control trajectories $\mathbf{u}^*(\cdot)$ and $q^*(\cdot)$ into (7.30) yields a TPBVP with $\tilde{\mathbf{K}} : \mathsf{X} \times \mathbb{R}^{N_x} \to \mathbb{R}^{2N_x}$

$$\dot{\mathbf{y}}(t) = \tilde{\mathbf{K}}(\mathbf{y}(t)) = \begin{bmatrix} \mathbf{f}_{q^*(t)}(\mathbf{x}(t), \mathbf{u}^*(t)) \\ -\left(\frac{\partial l_{q^*(t)}}{\partial \mathbf{x}}\right)(\mathbf{x}(t), \mathbf{u}^*(t)) - \left(\frac{\partial \mathbf{f}_{q^*(t)}}{\partial \mathbf{x}}\right)^T (\mathbf{x}(t), \mathbf{u}^*(t)) \cdot \boldsymbol{\lambda}(t) \end{bmatrix}$$

which can be solved with the boundary conditions (7.10)–(7.12) by an extended version of the indirect shooting algorithm for purely continuous systems. The minimum condition (7.31) can be used to get an indicator which discrete state is active at a given time step.

Again, we use the time grid (7.15) for the discretization process. The remaining challenge is to find a vector of initial costates $\overline{\boldsymbol{\lambda}}_{[1]} = \boldsymbol{\lambda}(t_0)$ that fulfill the final state conditions $\boldsymbol{\Upsilon}(\overline{\mathbf{y}}_{[N_t]}) = \mathbf{0}$. The main modification consists in the determination of the discrete state $\overline{\mathbf{q}}_{[k]}$ to be applied at each time step along with the determination of the continuous-valued controls $\overline{\mathbf{u}}_{[k]}$ at the same time step. This is done in two stages:

1. For each $q \in \hat{\mathcal{Q}}$, an optimal control value $\mathbf{u}_q^*$ is determined that minimizes the discretized Hamiltonian (7.29),

$$\mathbf{u}_q^* = \arg \min_{\mathbf{u}_q \in \hat{\mathcal{U}}(q,kh)} \mathcal{H}\left(\overline{\mathbf{x}}_{[k]}, q, \overline{\boldsymbol{\lambda}}_{[k]}, \mathbf{u}_q\right),$$

   for the given discrete state;

2. The optimal discrete state q^* is chosen that minimizes the Hamiltonian

$$\mathcal{H}\left(\overline{\mathbf{x}}_{[k]}, q, \overline{\boldsymbol{\lambda}}_{[k]}, \mathbf{u}_q^*\right)$$

   by comparing the function values $\mathcal{H}(\cdot, \mathbf{u}_q^*)$ for all $q \in \hat{\mathcal{Q}}$ using the respective control values $\mathbf{u}_q^*$. Applying the discrete state $\overline{\mathbf{q}}_{[k]} = q^*$ and the continuous-valued controls $\overline{\mathbf{u}}_{[k]} = \mathbf{u}_{q^*}^*$, the next quadrature step based on the time grid $\mathcal{G}_t$ can be computed by

$$\overline{\mathbf{y}}_{[k+1]} = \overline{\mathbf{y}}_{[k]} + h \cdot \boldsymbol{\Gamma}_{\tilde{K}}\left(\overline{\mathbf{y}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{u}}_{[k]}, t_k, h\right)$$

   for solving the initial value problem.

The vectors $\overline{\mathbf{y}} := \left[\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{N_t}\right] \in \mathbb{R}^{2N_x \cdot (N_t+1)}$, $\overline{\mathbf{u}} := \left[\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N_t-1}\right] \in \mathbb{R}^{N_u \cdot N_t}$, and $\overline{\mathbf{q}} := \left[q_0, q_1, \ldots, q_{N_t-1}\right] \in \mathbb{R}^{N_t}$ are assembled from the discretization of the extended states $\mathbf{y}(\cdot)$, the continuous-valued controls $\mathbf{u}(\cdot)$, and the discrete state $q(\cdot)$.

The two-stage procedure implies that the main function must contain an additional "for-loop" code fragment that iterates over all possible discrete states and calculates the optimal control values $\mathbf{u}_q^*$, which are used to find the optimal discrete state value q^* that minimizes the Hamiltonian. Algorithm 7.3 summarizes the main function:

Algorithm 7.3 Main Function for Indirect Single Shooting for SOCPs without State Jumps

Require: $\hat{\boldsymbol{\lambda}}$
1: $\overline{\mathbf{y}}_{[0]} \leftarrow [\overline{\mathbf{x}}_{[0]}, \hat{\boldsymbol{\lambda}}]^T$
2: **for** $k \leftarrow 0$ **to** $N_t - 1$ **do**
3: **for** $q \leftarrow 1$ **to** N_q **do**
4: $\mathbf{u}_q^* \leftarrow \arg \min_{\mathbf{u}_q \in \hat{\mathcal{U}}(q,kh)} \mathcal{H}(\overline{\mathbf{x}}_{[k]}, q, \overline{\boldsymbol{\lambda}}_{[k]}, \mathbf{u}_q)$
5: **end for**
6: $q^* \leftarrow \arg \min_{q \in \hat{\mathcal{Q}}} \mathcal{H}(\overline{\mathbf{x}}_{[k]}, q, \overline{\boldsymbol{\lambda}}_{[k]}, \mathbf{u}_q^*)$
7: $\overline{\mathbf{u}}_{[k]} \leftarrow \mathbf{u}_{q^*}^*$
8: $\overline{\mathbf{q}}_{[k]} \leftarrow q^*$
9: $\overline{\mathbf{y}}_{[k+1]} \leftarrow \overline{\mathbf{y}}_{[k]} + h \cdot \boldsymbol{\Gamma}_{\tilde{K}} \left(\overline{\mathbf{y}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{u}}_{[k]}, t_k, h\right)$
10: **end for**
11: **return** $\overline{\mathbf{y}}$, $\overline{\mathbf{u}}$, $\overline{\mathbf{q}}$, and $\boldsymbol{\Upsilon}(\overline{\mathbf{y}}_{[N_t]})$

Algorithm 7.3 can be straightforwardly extended for multiple shooting by breaking the shooting interval into smaller subintervals based upon the shooting grid specification (7.23). This makes the algorithm more complex but enhances its robustness. The multiple shooting version requires to find in an iterative manner estimated initial conditions (7.24) to solve $\boldsymbol{\Lambda} = \mathbf{0}$ up to a desired exactness. Algorithm 7.4 summarizes the main function for an indirect multiple shooting method for SOCPs without state jumps:

Algorithm 7.4 Main Function for Indirect Multiple Shooting for SOCPs without State Jumps

Require: $\mathbf{Y}$
1: **for** $j \leftarrow 1$ **to** $N_d - 1$ **do**
2: $\overline{\mathbf{y}}_{[K_j]} \leftarrow \mathbf{Y}_j$
3: **for** $k \leftarrow K_j$ **to** $K_{j+1} - 1$ **do**
4: **for** $q \leftarrow 1$ **to** N_q **do**
5: $\mathbf{u}_q^* \leftarrow \arg \min_{\mathbf{u}_q \in \hat{\mathcal{U}}(q,kh)} \mathcal{H}(\overline{\mathbf{x}}_{[k]}, q, \overline{\boldsymbol{\lambda}}_{[k]}, \mathbf{u}_q)$
6: **end for**
7: $q^* \leftarrow \arg \min_{q \in \hat{\mathcal{Q}}} \mathcal{H}(\overline{\mathbf{x}}_{[k]}, q, \overline{\boldsymbol{\lambda}}_{[k]}, \mathbf{u}_q^*)$
8: $\overline{\mathbf{u}}_{[k]} \leftarrow \mathbf{u}_{q^*}^*$
9: $\overline{\mathbf{q}}_{[k]} \leftarrow q^*$
10: $\overline{\mathbf{y}}_{[k+1]} \leftarrow \overline{\mathbf{y}}_{[k]} + h \cdot \boldsymbol{\Gamma}_{\tilde{K}} \left(\overline{\mathbf{y}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\mathbf{u}}_{[k]}, t_k, h\right)$
11: **end for**
12: $\boldsymbol{\theta}_j = \overline{\mathbf{y}}_{[K_{j+1}]} - \mathbf{Y}_{j+1}$
13: **end for**
14: evaluate $\boldsymbol{\Upsilon}(\overline{\mathbf{y}}_{[N_t]})$ using (7.14)
15: assemble $\boldsymbol{\Lambda}$
16: **return** $\overline{\mathbf{y}}$, $\overline{\mathbf{u}}$, $\overline{\mathbf{q}}$, and $\boldsymbol{\Lambda}$

By restricting the discrete decision to all admissible discrete control $\varpi \in \hat{\mathsf{B}}_q$ Algorithms 7.3 and 7.4 can be easily modified to deal with hybrid systems with continuous states.

As mentioned in the introductory section of this chapter these algorithms only apply to hybrid systems with continuous states. The most important difference to hybrid systems with state jumps is that beneath the continuous-valued controls $\mathbf{u}(\cdot)$ the discrete state $q(\cdot)$ can be used to satisfy the Hamiltonian minimum condition (4.123) without any a priori knowledge of the number of switchings. This simplification can be attributed to the continuity of the costates on a change of the discrete state. To account for state jumps, the hybrid system must be decomposed into a fixed number of phases with constant discrete state as already employed for the stacking procedure in Sect. 4.4, which in turn means that the switching structure $q = (q_0, q_1, q_2, \ldots)$ and the number of switchings N_{swt} has to be known a priori.

The proposed indirect methods for solving hybrid optimal control problems with continuous states employ the first-order necessary conditions and obtain local minimum solutions for some problems discovered in this book. But since we did not make any convergence considerations, the algorithms may not perform well for other problem instances.

7.4 Discussion

Indirect methods attempt to solve TPBVPs or MPBVPs to find trajectories that satisfy a set of first-order necessary conditions. If a solution can be found, the solution is usually highly accurate. However, there are major difficulties in practice that prevent the application of indirect methods. Let us summarize these problems again:

1. first-order necessary conditions must be derived for every new problem instance. This requires a user with a solid knowledge in optimal control theory;
2. even for users confident in using optimal control theory, it may be very difficult or even impossible to construct these expressions for complicated black-box applications (Betts [4]);
3. optimal control problems may contain state constraints $\mathbf{c}_x(\cdot)$. It is very difficult to incorporate state constraints directly into the solution method, which requires to have an a priori estimation of the constrained/unconstrained arcs. Furthermore, the sequence of constrained/unconstrained arcs introduce the additional difficulty of imposing the correct jump conditions (4.64) and (4.65) for the Hamiltonian and costates at the entry points; and
4. indirect methods deliver only open-loop solutions.

Generally, the latter point is an obstacle in real-time implementations. As already known from Chap. 6, algorithms based on the dynamic programming principle generate automatically closed-loop solutions.

For OCPs with quadratic functionals and linear dynamic systems and constraints (LQ-OCP), the first-order necessary conditions can be explicitly resolved to obtain an optimal closed-form control law. However, many practical problems cannot be casted into LQ-OCPs. Therefore, numerical procedures such as single or multiple shooting algorithms have to be applied for the determination of optimal trajectories. Optimal trajectories, however, do not serve as an analytical control law, since they were calculated for specific boundary conditions and are only optimal, if these boundary conditions apply. As noticed in Geering [9], a simple brute-force procedure to adapt the optimal open-loop trajectories to a closed-form control law is to resolve the remaining optimal control problem over the time span $[t, t_f]$ with initial states $\mathbf{x}_0 = \mathbf{x}(t)$ given at time t. This yields an optimal control law $\mathbf{u}(\mathbf{x}(t), t)$ as function of the initial states $\mathbf{x}(t)$ and time t. A modified version of this closed-form control law is discussed in Chap. 12. This methodology can also be applied to direct methods discussed in the next chapter.

Numerical Considerations

The indirect shooting method consists of three numerical steps:

- an integration method for solving the IVP for a given $\hat{\boldsymbol{\lambda}}$;
- a method for minimizing the Hamiltonian function at each step of the IVP solution to determine the continuous-valued controls and the discrete state at each time instant; and
- a solver for the nonlinear system of equations $\boldsymbol{\Upsilon}(\overline{\mathbf{y}}_{[N_t]}) = \mathbf{0}$ or $\boldsymbol{\Lambda} = \mathbf{0}$.

The solution of an IVP includes many integration steps—each of them approximated by a quadrature formula. The selection of an appropriate integration scheme is therefore crucial for the success of the solution of the OCP and should be done with some care because rounding errors can be a tremendous subject in the complete procedure. In principle, any explicit Runge–Kutta scheme can be used that satisfies the additional Hager [10] conditions.

The continuous-valued controls at each time instant are usually not given analytically but determined by a numerical minimization of the augmented Hamiltonian. The minimization of the augmented Hamiltonian is performed point-wise at each step in the solution process of the IVP and should therefore be very efficient. In general, an efficient optimization can be performed by SQP algorithms. In some cases, the Hamiltonian can exhibit multiple local minima. Evaluating the Hamiltonian on a grid first and then choosing the lowest value as start value for further refinement using SQP can improve the numerical convergence.

The minimization of $\boldsymbol{\Upsilon}(\overline{\mathbf{y}}_{[N_t]}) = \mathbf{0}$ or $\boldsymbol{\Lambda} = \mathbf{0}$ should find a solution with high precision. However, initial values for $\hat{\boldsymbol{\lambda}}$ can be particularly hard to find. In most cases they offer no physical interpretation and even the order of magnitude might not be known in advance. If no acceptable guess is made, the numerical procedure for the solution of $\boldsymbol{\Upsilon}(\overline{\mathbf{y}}_{[N_t]}) = \mathbf{0}$ or $\boldsymbol{\Lambda} = \mathbf{0}$ can fail to converge. For the special case discovered in this book that the boundary conditions reduce to a single equation $\Upsilon(\cdot) = 0$ and the costate is assumed to be constant. For this case, *regula falsi* methods

are a good choice for solving $\Upsilon(\cdot) = 0$. Two simple regula falsi methods are the *bisection* or the *secant method*. A more involved algorithm with good convergence behavior in many cases is the Pegasus method (Dowell and Jarrat [8]).

One can finally say, the applicability of the indirect shooting method strongly depends on the specific problem to be solved. For some problems, indirect methods can outperform any other method and yield highly accurate results, whereas for other problems, indirect methods cannot be robustly implemented.

7.5 Bibliography

Indirect methods for optimal control are less attractive to practitioners due to the cumbersome procedure of deriving of the necessary conditions (Ascher et al. [2]) but can outperform direct methods in terms of accuracy. Common methods to solve MPBVPs for purely continuous optimal control problems are gradient-based (Bryson and Ho [6], Kirk [12], and Stengel [20]) and implement multiple or single shooting approaches (Betts [3], Bock and Plitt [5]). The indirect multiple shooting method can be traced back to Bulirsch [7], Keller [11] and Osborne [15]. von Stryk and Bulirsch [21] emphasizes the usefulness of combining indirect methods and direct methods called *hybrid approach*.

Most algorithms in the literature for solving SOCPs are based on two-stage approaches that use the necessary conditions to improve an initial guess of the switching sequence. An indirect approach for the solution of switched optimal control problems is described in Shaikh [19]. The algorithm varies switching times and the states at these switching times based on the differences in the costates and the Hamiltonian. In Riedinger and Kratz [17] necessary conditions for hybrid systems are derived from the Pontryagin's minimum principle and the Bellman principle. These necessary conditions are used in a mixed dynamic programming and Hamiltonian approach.

A multiple shooting algorithm for hybrid optimal control problems with controlled and autonomous switching is proposed by Riedinger et al. [18], where the trajectory of the hybrid system is decomposed into a fixed number of arcs with constant discrete state. In the work of Alamir and Attia [1], an initial guess of the continuous-valued controls and discrete state sequence is made and the corresponding state trajectory and costate trajectory are calculated. In a next step, optimized control inputs and the discrete state are computed, such that the Hamiltonian function is minimized for each time instant.

For continuous optimal control problems, where the numerical solution is too costly to obtain, Lukes [13] proposed a method to obtain an approximatively optimal control law in closed form. This method works for nonlinear dynamic systems and functionals, where the right-hand side function $\mathbf{f}(\cdot)$ of the differential equation and the Lagrange term $\mathcal{L}(\cdot)$ of the cost functional can be expressed by polynomial approximations.

Teo and Jennings [22] proposed to transform state constraints into equivalent equality constraints. This method overcomes the problem of knowing the sequence of constrained/unconstrained arcs but produces always a suboptimal solution.

Oberle and Grimm [14] developed the multiple shooting algorithm BNDSCO that was successfully applied mainly to the area of flight path optimization.

References

1. Alamir M, Attia S (2004) On solving optimal control problems for switched hybrid nonlinear systems by strong variations algorithms. In: 6th IFAC symposium on nonlinear control systems (NOLCOS), Stuttgart, Germany, pp 558–563
2. Ascher UM, Mattheij RM, Russell RD (1994) Numerical solution of boundary value problems for ordinary differential equations, vol 13. SIAM
3. Betts JT (1998) Survey of numerical methods for trajectory optimization. J Guidance Control Dyn 21(2):193–207
4. Betts JT (2010) Practical methods for optimal control and estimation using nonlinear programming, 2nd edn. Society for Industrial and Applied Mathematics. doi:10.1137/1.9780898718577
5. Bock H, Plitt K (1984) A multiple shooting algorithm for direct solution of optimal control problems. In: Proceedings of the 9th world congress of the international federation of automatic control, vol 9
6. Bryson A, Ho YC (1975) Applied Optimal Control—Optimization, Estimation and Control. Taylor and Francis Inc., New York
7. Bulirsch R (1971) Die Mehrzielmethode zur numerischen Lösung von nichtlinearen Randwertproblemen und Aufgaben der optimalen Steuerung. Report der Carl-Cranz-Gesellschaft
8. Dowell M, Jarrat P (1972) The Pegasus method for computing the root of an equation. BIT Numer Math 12:503–508
9. Geering HP (2007) Optimal control with engineering applications. Springer, New York
10. Hager WW (2000) Runge-Kutta methods in optimal control and the transformed adjoint system. Numer Math 87(2):247–282
11. Keller HB (1968) Numerical methods for two-point boundary-value problems. Blaisdell, London
12. Kirk D (1970) Optimal control theory: an introduction. Englewood Cliffs, Prentice-Hall
13. Lukes DL (1969) Optimal regulation of nonlinear dynamical systems. SIAM J Control 7(1):75–100
14. Oberle H, Grimm W (1989) BNDSCO—a program for the numerical solution of optimal control problems. In: Report no. 515 der DFVLR, Reihe B, Bericht 36
15. Osborne MR (1969) On shooting methods for boundary value problems. J Math Anal Appl 27(2):417–433
16. Rao AV (2009) A survey of numerical methods for optimal control. Adv Astronaut Sci 135(1):497–528
17. Riedinger P, Kratz F (2003) An optimal control approach for hybrid systems. Eur J Control 9:449–458
18. Riedinger P, Daafouz J, Iung C (2005) About solving hybrid optimal control problems. IMACS05
19. Shaikh MS (2004) Optimal control of hybrid systems: theory and algorithms. PhD thesis, Department of electrical and computer engineering, McGill University, Montreal
20. Stengel RF (1994) Optimal control and estimation. Dover Publications
21. von Stryk O, Bulirsch R (1992) Direct and indirect methods for trajectory optimization. Ann Oper Res 37:357–373
22. Teo K, Jennings L (1989) Nonlinear optimal control problems with continuous state inequality constraints. J Optim Theory Appl 63(1):1–22

Chapter 8
Direct Methods for Optimal Control

8.1 Introduction

The complexity of optimal control of hybrid systems, as already mentioned in the introductory text of Sect. 1.3.3, makes it unlikely to develop a general solution procedure that can be applied to any subclass of hybrid systems. It is therefore more promising to use the structural information of the class of interesting problems and to develop algorithms which are tailored to these problems. Direct methods offer from all three solution classes the highest potential for tailoring algorithms. The methods, which will be discussed and used in this book, and their relations are shown in Fig. 8.1.

Dynamic programming can easily be extended for many subclasses of hybrid systems but suffers from the curse of dimensionality, which makes it only applicable to small problem sizes to achieve an acceptable accuracy. Indirect methods can deal with large hybrid systems and offer highly accurate solutions but converge only in a small domain. In contrast to these methods, we discuss in this chapter the class of direct methods. These algorithms can deal with large systems and are more flexible and robust but less accurate compared with the indirect methods.

Let us start with a survey of different direct methods for solving *switched optimal control problems* (SOCP) without state jumps

$$\min_{\mathbf{u}(\cdot)\in L^\infty([t_0,t_f],\mathsf{U}),\ q(\cdot)\in\mathcal{Q}} \phi\left(\mathbf{u}(\cdot), q(\cdot)\right) = m(\mathbf{x}^*(t_f)) + \int_{t_0}^{t_f} l_{q^*(t)}(\mathbf{x}^*(t), \mathbf{u}^*(t))\,\mathrm{d}t \tag{8.1}$$

subject to

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)), \qquad \text{for a.e. } t \in [t_0, t_f] \tag{8.2}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{8.3}$$

$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f \tag{8.4}$$

$$\mathbf{c}_{u,q}(\mathbf{u}(t)) \leq \mathbf{0}_{N_{c_{u,q}}\times 1}, \qquad \forall t \in [t_0, t_f] \tag{8.5}$$

$$\mathbf{c}_{x,q}(\mathbf{x}(t)) \leq \mathbf{0}_{N_{c_{x,q}}\times 1}, \qquad \forall t \in [t_0, t_f] \tag{8.6}$$

T.J. Böhme and B. Frank, *Hybrid Systems, Optimal Control and Hybrid Vehicles*,
Advances in Industrial Control, DOI 10.1007/978-3-319-51317-1_8

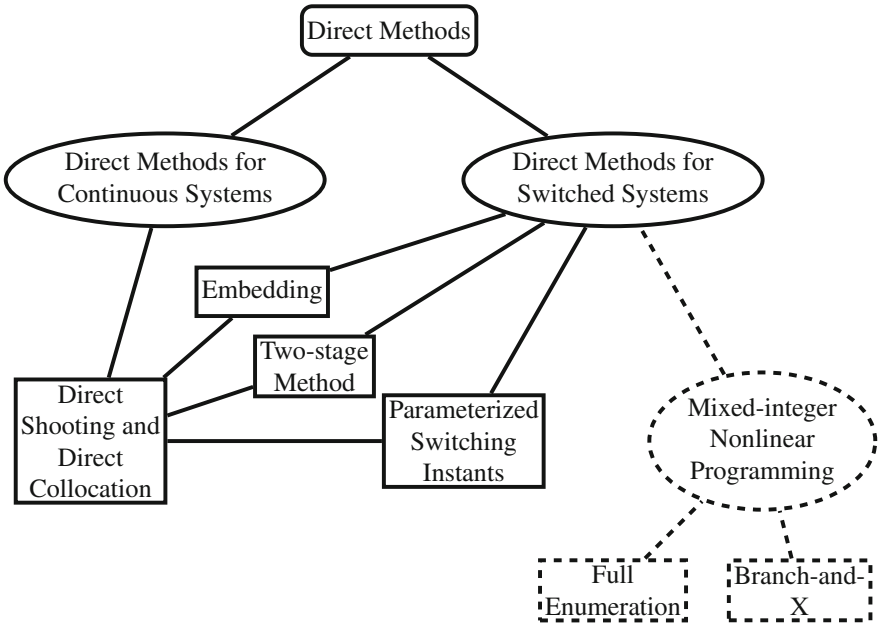


Fig. 8.1 Direct methods for solving SOCPs. *Elliptical* nodes indicate optimization classes; *rectangular* nodes indicate optimization methods; and *dashed* nodes indicate optimization classes not covered in this book

where $\mathbf{x}_0$ and $\mathbf{x}_f$ are the initial and (partially specified) final state values, respectively. The set $\mathcal{I}_f$ specifies which states are fixed at the endpoint t_f.

A common method to solve (8.1)–(8.6) is to cast this problem to a *mixed-integer nonlinear programming* (MINLP) problem. In so doing, the continuous parts of the problem formulation are transformed using a direct transcription into a finite-dimensional problem with a finite number of variables, assembled in the optimization vector $\overline{\mathbf{y}} = [y_0, y_1, \ldots, y_{N_y}]^T$. The overlined vector indicates the process of discretization. Direct transcription describes the process of transforming the infinite-dimensional continuous part of the *optimal control problem* (OCP) into a finite-dimensional NLP and can be classified in direct single shooting, direct multiple shooting, or direct collocation (Betts and Huffman [9, 11] and von Stryk and Bulirsch [66]). The discrete state is discretized simply with a piecewise constant scheme to an integer-valued vector $\overline{\mathbf{q}}$. Then, a MINLP problem is a finite-dimensional optimization problem that involves discretized continuous-valued as well as integer-valued variables, which can be regarded as a nontrivial combination of a *nonlinear programming* problem (NLP) and an integer programming problem.

Definition 8.1 (*Mixed-Integer Nonlinear Programming Problem*) The MINLP problem is defined as follows

$$\begin{aligned} \min_{\overline{\mathbf{y}} \in \mathbb{R}^{N_{\overline{y}}},\, \overline{\mathbf{q}} \in \overline{\mathcal{Q}}} \quad & f(\overline{\mathbf{y}}, \overline{\mathbf{q}}) \\ \text{subject to} \quad & \mathbf{g}(\overline{\mathbf{y}}, \overline{\mathbf{q}}) \leq \mathbf{0} \\ & \mathbf{h}(\overline{\mathbf{y}}, \overline{\mathbf{q}}) = \mathbf{0} \end{aligned} \tag{8.7}$$

where $f : \mathbb{R}^{N_{\overline{y}}} \times \overline{\mathcal{Q}} \to \mathbb{R}$, $\mathbf{g} : \mathbb{R}^{N_{\overline{y}}} \times \overline{\mathcal{Q}} \to \mathbb{R}^{N_g}$, and $\mathbf{h} : \mathbb{R}^{N_{\overline{y}}} \times \overline{\mathcal{Q}} \to \mathbb{R}^{N_h}$ are assumed to be all twice continuously differentiable and real-valued. The admissible discrete set $\overline{\mathcal{Q}}$ is defined by a polyhedral set of integers, i.e., $\overline{\mathcal{Q}} := \left\{ \overline{\mathbf{q}} \in \mathbb{Z}^{N_{\overline{q}}} \;\middle|\; \mathbf{A}\overline{\mathbf{q}} \leq \mathbf{a} \right\}$ for a matrix $\mathbf{A} \in \mathbb{R}^{N_a \times N_{\overline{q}}}$ and a vector $\mathbf{a} \in \mathbb{R}^{N_a}$. △

A naive approach for solving (8.7) is to fix the discrete state sequence $\overline{\mathbf{q}}$ and to perform a *full enumeration* (Kirches [36]). In a full enumeration, solutions of the OCP for all possible combinations of the discrete state sequence $\overline{\mathbf{q}}$ are calculated, whereby for each OCP the discrete control is then treated as fixed control. It is clear, such an exponentially increasing procedure becomes very quickly prohibitive due to the high computational effort, if $N_{\overline{q}}$ is high. A further major drawback of this naive solution process is the probability to obtain an infeasible NLP problem, which is strongly influenced by the fixed discrete state sequence.

A generalization of the enumeration technique are *Branch-and-X* (BX) methods. A good overview is given by Grossmann [34]. The most prominent member of BX is the *branch-and-bound* (BB) method for solving integer and combinatorial problems and thus gives a framework in which (8.7) can be solved. The fundamental idea behind BB is to perform a tree search in the space of the integer variables. The root consists of the original problem with all integer variables relaxed to non-integer variables. We call this NLP problem relaxation.

Definition 8.2 (*Nonlinear Programming Problem Relaxation*) Suppose the Definition 8.1 holds. Then, a *nonlinear programming subproblem relaxation* is given by

$$\begin{aligned} z_{lo} = \min_{\overline{\mathbf{y}} \in \mathbb{R}^{N_{\overline{y}}}, \overline{\mathbf{q}} \in \overline{\mathcal{Q}}_r} \quad & f(\overline{\mathbf{y}}, \overline{\mathbf{q}}) \\ \text{subject to} \quad & \mathbf{g}(\overline{\mathbf{y}}, \overline{\mathbf{q}}) \leq \mathbf{0} \\ & \mathbf{h}(\overline{\mathbf{y}}, \overline{\mathbf{q}}) = \mathbf{0} \end{aligned} \tag{8.8}$$

where $f : \mathbb{R}^{N_{\overline{y}}} \times \overline{\mathcal{Q}}_r \to \mathbb{R}$, $\mathbf{g} : \mathbb{R}^{N_{\overline{y}}} \times \overline{\mathcal{Q}}_r \to \mathbb{R}^{N_g}$, and $\mathbf{h} : \mathbb{R}^{N_{\overline{y}}} \times \overline{\mathcal{Q}}_r \to \mathbb{R}^{N_h}$ are assumed to be all twice continuously differentiable and real-valued. $\overline{\mathcal{Q}}_r$ is a relaxation of $\overline{\mathcal{Q}}$.

△

The first step in the BB approach is to solve the relaxed NLP problem (8.8), which provides a lower bound z_{lo} of the problem. This bound is an important indication, since no better cost value can be found by fixing the discrete state to feasible integers.

The next step is branching. An obvious way to divide the feasible region of the root node is to branch on a fractional variable (non-integer value), say $\overline{\mathbf{q}}_i^{[j]}$, and to enforce additional simple constraints $\boldsymbol{\alpha}_{i+}^{[j]} \leq \overline{\mathbf{q}}_{i+}^{[j]}$ and $\overline{\mathbf{q}}_{i-}^{[j]} \leq \boldsymbol{\beta}_{i-}^{[j]}$ to (8.8) due to the branching. This gives two new subproblems (nodes):

$$
\begin{aligned}
z_{lo}^{i+} = \min_{\overline{\mathbf{y}}_{i+}\in\mathbb{R}^{N_{\overline{y}}},\overline{\mathbf{q}}_{i+}\in\overline{\mathcal{Q}}_r} \quad & f(\overline{\mathbf{y}}_{i+},\overline{\mathbf{q}}_{i+}) \\
\text{subject to} \quad & \mathbf{g}(\overline{\mathbf{y}}_{i+},\overline{\mathbf{q}}_{i+}) \leq \mathbf{0} \\
& \mathbf{h}(\overline{\mathbf{y}}_{i+},\overline{\mathbf{q}}_{i+}) = \mathbf{0} \\
& \boldsymbol{\alpha}_{i+}^{[j]} \leq \overline{\mathbf{q}}_{i+}^{[j]} \leq \mathbf{q}^{max}
\end{aligned}
\quad \text{and} \quad
\begin{aligned}
z_{lo}^{i-} = \min_{\overline{\mathbf{y}}_{i-}\in\mathbb{R}^{N_{\overline{y}}},\overline{\mathbf{q}}_{i-}\in\overline{\mathcal{Q}}_r} \quad & f(\overline{\mathbf{y}}_{i-},\overline{\mathbf{q}}_{i-}) \\
\text{subject to} \quad & \mathbf{g}(\overline{\mathbf{y}}_{i-},\overline{\mathbf{q}}_{i-}) \leq \mathbf{0} \\
& \mathbf{h}(\overline{\mathbf{y}}_{i-},\overline{\mathbf{q}}_{i-}) = \mathbf{0} \\
& \mathbf{q}^{min} \leq \overline{\mathbf{q}}_{i-}^{[j]} \leq \boldsymbol{\beta}_{i-}^{[j]}.
\end{aligned}
$$

We call this branching a *variable dichotomy*. The bounds are obtained from the fractional value of the parent node i by $\boldsymbol{\alpha}_{i+}^{[j]} = \left\lceil \overline{\mathbf{q}}_i^{[j]} \right\rceil$, $\boldsymbol{\beta}_{i-}^{[j]} = \left\lfloor \overline{\mathbf{q}}_i^{[j]} \right\rfloor$. The symbols $\lceil\cdot\rceil$ and $\lfloor\cdot\rfloor$ and are ceil and floor functions, respectively. The two newly generated subproblems must be solved. However, the whole subtrees of the subproblems can be excluded from further exploration by fathoming their respective parent node. This strategy avoids a complete tree evaluation as by full enumeration and can be attributed to the success of branch-and-bound. According to Leyffer [40], a whole subtree can be neglected, if one of the following criteria applies to the parent node:

- **infeasibility**: the problem is infeasible, because any subproblem in its subtree is then also infeasible (diamond, light gray);
- **integrality**: the problem produces an integer feasible solution, because in this case the solution is optimal for the entire subtree (rectangular, gray). This solution is a new upper bound z_{up} of the problem, if its cost value is lower than the current upper bound;
- **dominance**: the lower bound z_{lo} of the problem is greater or equal than the current upper bound z_{up}, because in this case there can be no better integer solution in this subtree (diamond, gray).

These rules are demonstrated in Fig. 8.2.

This process is repeated and terminates if all branches are evaluated according to the criteria above. The basic procedure of the BB approach is summarized in Algorithm 8.1.

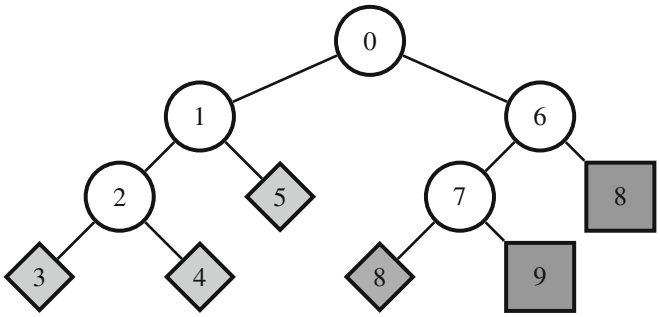


Fig. 8.2 Branch-and-Bound concept with depth-first search. Nodes in the search tree are fathomed when they are infeasible (*light gray diamonds*), dominated by upper bound (*gray diamonds*), or yield an integer solution (*gray rectangles*). Otherwise they are split up in smaller subproblems (*white circles*). The *numbers* are the execution order of the depth-search algorithm

Algorithm 8.1 Branch-and-bound Skeleton (Leyffer [40])

set $z_{up} = \infty$ as upper bound
add the NLP relaxation to the set $\mathcal{I}_{heap}$
while $\mathcal{I}_{heap} \neq \emptyset$ **do**
    remove a subproblem from the set $\mathcal{I}_{heap}$
    find a solution $(\overline{\mathbf{y}}_i, \overline{\mathbf{q}}_i)$ to the subproblem
    if subproblem is infeasible **then**
        prune subproblem by infeasibility
    else if $f(\overline{\mathbf{y}}_i, \overline{\mathbf{q}}_i) \geq z_{up}$ **then**
        prune subproblem by dominance
    else if $\overline{\mathbf{q}}_i$ is integral **then**
        update: $z_{up} = f(\overline{\mathbf{y}}_i, \overline{\mathbf{q}}_i)$ and $\overline{\mathbf{q}}^* = \overline{\mathbf{q}}_i$
        remove all subproblems from $\mathcal{I}_{heap}$ with lower bounds $z_{lo} \geq z_{up}$
    else
        branch on fractional variable $\overline{\mathbf{q}}_i^{[j]}$ and add the two new subproblems to the set $\mathcal{I}_{heap}$
    end if
end while

The core problem of Algorithm 8.1 is to obtain good heuristics to choose the integer-valued constraints $\boldsymbol{\alpha}_{i+}^{[j]}$ and $\boldsymbol{\beta}_{i-}^{[j]}$, which are crucial to obtain feasible solutions quickly. This performance depends critically on several aspects (cf. Sager [54]):

- good heuristics are important, as more subtrees can be explored right away based on their good bounds on the optimal value;
- a decision has to be made, which fractional variable is chosen to branch on. This depends on the choice of the user, which "branching" strategy, e.g., *most–violation–branching* or *strong branching*, is selected;
- the order in which the subproblems will be proceeded, with the extreme options *depth–first* search and *breadth–first* search. The first option, as the name suggests, explores the newly created subproblems in a depth manner first whereas the second option proceeds one of the subproblems on the highest level in the tree first.

It is not unusual that the heuristics are tailored to specific problems. We will not going in more depth about this topic, but interested readers may consult the survey paper of branching rules presented by Linderoth and Savelsbergh [43].

The complexity of this type of problem is non-polynomial in time, i.e., $\mathcal{NP}$-hard. Thus, Grossmann and Kravanja [31] pointed out that BB methods are generally only attractive if the NLP subproblems are relatively inexpensive to solve. This is the case if the dimensionality of the discrete variable is low, which is definitely not the case if the discrete state sequence $\overline{\mathbf{q}}$ contains many discretization points. This obstacle prevents the application to large-scale problems, but the idea can be adopted, e.g., rounding strategies for obtaining binary feasible solutions.

Keeping in mind, that we could not use a standard nonlinear programming method straightforwardly to the SOCP because of the problem's disjoint behavior. The branch-and bound method, however, made usage of a very important ingredient that can help us to overcome this problem: *relaxation*. Relaxation is basically a reformulation technique that provides us a new problem formulation with more desirable

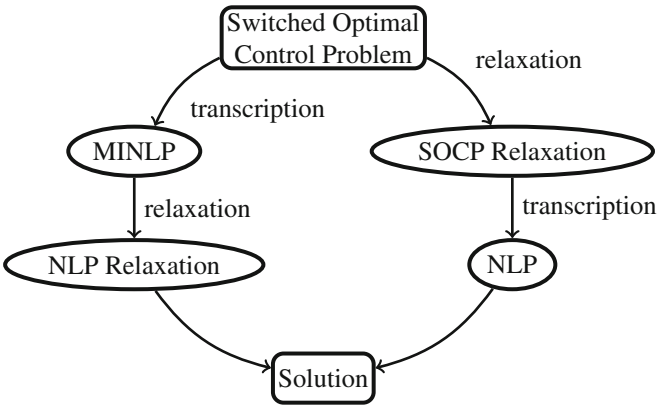


Fig. 8.3 Comparison of branch-and-bound and SOCP relaxation. Relaxation is used at different stages

properties with respect to the numerical solution. Then, it is natural to relax the discrete state of the SOCP before a direct transcription method is applied. This little trick enables us to apply standard nonlinear programming methods without numerical rank-deficiencies. Figure 8.3 shows this little difference in both approaches, which has a major impact on solving large-scale problems.

The embedding approach proposed by Bengea and DeCarlo [5] and Sager [54] share the same idea of relaxing the Boolean variables $\boldsymbol{\sigma}(\cdot)$ of a *binary switched optimal control problem* (BSOCP). It is not difficult to transform the original problem (8.1)–(8.6) to a BSOCP. The binary variables $\boldsymbol{\sigma}(t) \in \{0, 1\}^{N_q}$ after relaxation may take values

$$\hat{\boldsymbol{\sigma}}(t) \in [0, 1]^{N_q},$$

where $\hat{\boldsymbol{\sigma}}(\cdot)$ denotes the relaxed binary variables. The control set $\hat{\mathcal{P}} = \hat{\mathcal{U}}(q(t), t) \times [0, 1]^{N_q}$ is now a convex set if $\hat{\mathcal{U}}(q(t), t)$ is convex. Then, the relaxed binary system $\dot{\mathbf{x}}(t) = \mathbf{F}(\mathbf{x}(t), \boldsymbol{\rho}(t)),\ \boldsymbol{\rho}(t) \in \hat{\mathcal{P}}$ is continuous-valued and can be solved as part of a continuous OCP by direct transcription methods. In many cases, the solution to this problem will yield a control trajectory, that is of bang–bang type with respect to the discrete controls and therefore satisfies $\hat{\boldsymbol{\sigma}}(t) \in \{0, 1\}^{N_q}$. It is important to note, that the embedding approach is just able to solve switched optimal control problems, whereas the BB methods can include rules to deal with hybrid systems.

The constrained problem (8.1)–(8.5) can also be solved by *two-stage algorithms* (Xu and Antsaklis [76]). Two-stage approaches use additional information, e.g., the gradient of the Hamiltonian with respect to the discrete state, to alter the discrete state trajectory. In the first stage, a fixed discrete state sequence is used to obtain a continuous optimal control problem. Then, the OCP can be solved w.r.t. continuous-valued controls by standard nonlinear programming methods. In the second stage,

the discrete state sequence is varied to obtain a different number of switchings or to change the order of the active subsystems. Two-stage algorithms are computationally very demanding due to the nested optimization loops. A simplified version for some practical problems only finds the optimal continuous-valued controls and the optimal switching times. The mode sequence of active subsystems is assumed to be a priori known.

8.2 Optimal Control for Continuous Systems

This section introduces basic transcription approaches which are well recognized to solve continuous optimal control problems,

$$\min_{\mathbf{u}(\cdot)\in\mathcal{U}} \phi\left(\mathbf{u}(\cdot)\right) = m(\mathbf{x}^*(t_f)) + \int_{t_0}^{t_f} l(\mathbf{x}^*(t), \mathbf{u}^*(t))\,\mathrm{d}t \tag{8.9}$$

subject to

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \qquad \text{for a.e. } t \in [t_0, t_f] \tag{8.10}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{8.11}$$

$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f \tag{8.12}$$

$$\mathbf{c}_u(\mathbf{u}(t)) \leq \mathbf{0}_{N_{c,u}\times 1}, \qquad \forall t \in [t_0, t_f] \tag{8.13}$$

$$\mathbf{c}_x(\mathbf{u}(t)) \leq \mathbf{0}_{N_{c,x}\times 1}, \qquad \forall t \in [t_0, t_f] \tag{8.14}$$

which are more robust than indirect methods regarding the initial estimation for the optimization process. They are based on formulations of nonlinear programming problems and can be applied to SOCPs as well if the discrete state sequence $q(\cdot)$ is assumed to be a priori known and remains constant over the entire optimization task. For the latter case, the SOCP can be treated as a continuous OCP.

For continuous OCPs different transcriptions classes exist, among them:

- control parametrization; and
- control and state parametrization.

These classes can be divided into shooting and collocation transcriptions as depicted in Fig. 8.4.

The underlying discretization scheme is the key for the success of the whole optimization task. For the transcription it is assumed that the continuous OCP can be exactly approximated by the discretization scheme, if an arbitrarily fine discretization grid is chosen. This assumption does not hold always. We provide some remarks in Sect. 8.5. The integration scheme plays also an important role for obtaining high performance and high consistence-order. In direct collocation, the integration of the states can be approximated by using polynomials evaluated at fixed collocation points. This method is commonly known as *pseudospectral* and is used to increase the accuracy. The handling of state constraints (8.14) is usually a demanding task. An

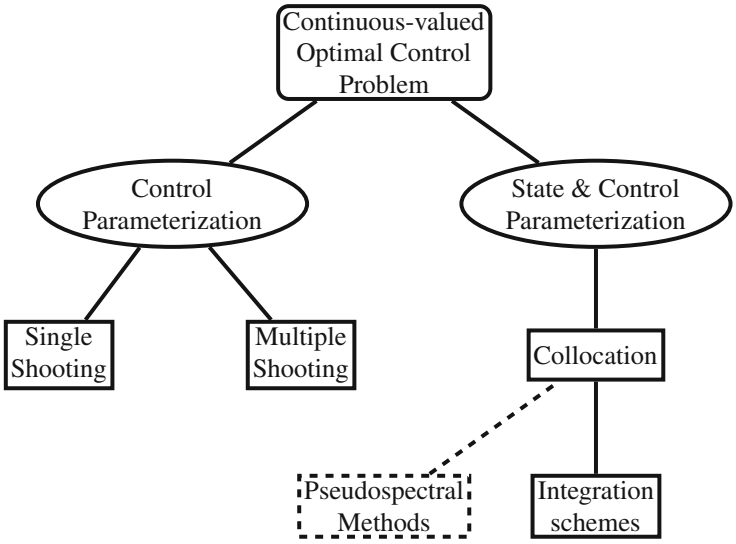


Fig. 8.4 Types of direct transcriptions for continuous optimal control problems. *Elliptical* nodes indicate optimization classes; *rectangular* nodes indicate optimization methods; and *dashed* nodes indicate optimization classes not covered in this book

advantage of direct collocation methods is the efficient treatment of these constraints. The direct shooting methods generate smaller problem sizes which can be efficiently solved for small-and medium-sized problems.

8.2.1 Direct Shooting

For the case that only the continuous-valued controls in (8.9)–(8.12) are discretized, one obtains a transcription method called *direct single shooting*. Using a piecewise constant or piecewise linear discretization scheme for the continuous-valued controls with

$$\mathbf{u}(t) = \begin{cases} \Xi_k^u \left(\mathbf{u}_k, \mathbf{u}_{k+1}, t\right), & \forall t \in \left[t_k, t_{k+1}\right), \quad k = 0, \ldots, N_t - 2 \\ \Xi_{N_t-1}^u \left(\mathbf{u}_{N_t-1}, \mathbf{u}_{N_t}, t\right), & \forall t \in \left[t_{N_t-1}, t_{N_t}\right] \end{cases} \tag{8.15}$$

gives the discretization vector

$$\overline{\mathbf{y}} = \overline{\mathbf{u}} = \left[\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N_t}\right]^T \in \mathbb{R}^{N_{\overline{\mathbf{y}}}} \tag{8.16}$$

defined on the time grid $\mathcal{G}_t$ with $N_{\overline{\mathbf{y}}} = N_u \cdot (N_t + 1)$. Applying (8.15) and (8.16) to (8.9)–(8.12) results in a finite-dimensional optimal control problem

$$\min_{\overline{\mathbf{y}} \in \mathbb{R}^{N_{\overline{y}}}} \; \phi(\overline{\mathbf{y}}) = m(\mathbf{x}^*(t_f)) + \int_{t_0}^{t_f} l\left(\mathbf{x}^*(t), \mathbf{u}^*(t)\right) \, \mathrm{d}t \tag{8.17}$$

subject to

$$\dot{\mathbf{x}}(t) = \mathbf{f}\left(\mathbf{x}(t), \mathbf{u}(t)\right), \qquad \text{for } t = t_{k+1/2}, \quad k = 0, \ldots, N_t - 1 \tag{8.18}$$

$$\mathbf{x}\left(t_0\right) = \mathbf{x}_0 \tag{8.19}$$

$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f \tag{8.20}$$

$$\mathbf{c}_u\left(\mathbf{u}(t)\right) \leq \mathbf{0}_{N_{c,u} \times 1}, \qquad \text{for } t = t_k, \quad k = 0, \ldots, N_t \tag{8.21}$$

where $t_{k+1/2} = (t_{k+1} + t_k)/2$. The switching sequence $\overline{\mathbf{q}}$ and the time grid $\mathcal{G}_t$ are treated as fixed boundary conditions for the transcriptions.

For a given set of discretized continuous-valued controls $\overline{\mathbf{u}}$, the solution of the *ordinary differential equation* (ODE) for a given initial state (8.19) can be obtained by applying any *Runge–Kutta* (RK) scheme from Chap. 5. Explicit RK methods are preferred here for the sake of simplicity, which yields

$$\begin{aligned} \overline{\mathbf{x}}_{[k+1]} &= \overline{\mathbf{x}}_{[k]} + h \cdot \boldsymbol{\Gamma}_f\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h\right), \quad k = 0, \ldots, N_t - 1 \\ \overline{\mathbf{x}}_{[0]} &= \mathbf{x}_0. \end{aligned}$$

Thus, the final state $\overline{\mathbf{x}}_{[N_t]}$ can be evaluated from the numerical solution of the ODE at time instant $k = N_t$. It is clear, that a piecewise linear control discretization makes only sense with a RK method with an order higher than one.

Let us transform the Bolza problem (8.17) to a Mayer problem using the rules from Sect. 3.3.7.1, which yields

$$\phi(\overline{\mathbf{y}}) = m(\mathbf{x}_{[N_t]}) + \overline{\tilde{\mathbf{x}}}_{[N_t]}.$$

The additional term $\overline{\tilde{\mathbf{x}}}_{[N_t]}$ is the final value of the integrated Lagrangian term $l(\cdot)$ using an explicit RK method

$$\begin{aligned} \overline{\tilde{\mathbf{x}}}_{[k+1]} &= \overline{\tilde{\mathbf{x}}}_{[k]} + h \cdot \Gamma_l\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h\right), \quad k = 0, \ldots, N_t - 1 \\ \overline{\tilde{\mathbf{x}}}_{[0]} &= 0. \end{aligned}$$

Using the discretized continuous-valued controls and states we can reformulate the OCP (8.17)–(8.21) as NLP

$$\min_{\overline{\mathbf{y}} \in \mathbb{R}^{N_{\overline{y}}}} \; \phi(\overline{\mathbf{y}}) = m\left(\overline{\mathbf{x}}^*_{[N_t]}\right) + \overline{\tilde{\mathbf{x}}}^*_{[N_t]} \tag{8.22}$$

subject to

$$\overline{\mathbf{x}}_{[0]} = \mathbf{x}_0 \tag{8.23}$$

$$\left(\overline{\mathbf{x}}_{[N_t]}\right)_{[\mathcal{I}_f]} - \mathbf{x}_f = \mathbf{0}_{\#\mathcal{I}_f \times 1} \tag{8.24}$$

$$\mathbf{c}_{\overline{u}}\left(\overline{\mathbf{u}}_{[k]}\right) \leq \mathbf{0}_{N_{c_u} \times 1}, \quad k = 0, \ldots, N_t \tag{8.25}$$

where $N_{\overline{y}}$ is the number of NLP variables. Collecting and assembling the equality and inequality constraints to

$$\mathbf{h}(\overline{\mathbf{y}}) = \left(\overline{\mathbf{x}}_{[N_t]}\right)_{[\mathcal{I}_f]} - \mathbf{x}_f$$

and

$$\mathbf{g}(\overline{\mathbf{y}}) = \left[\mathbf{c}_{\overline{u}}(\overline{\mathbf{u}}_{[0]}), \mathbf{c}_{\overline{u}}(\overline{\mathbf{u}}_{[1]}), \ldots, \mathbf{c}_{\overline{u}}(\overline{\mathbf{u}}_{[N_t]})\right]^T \in \mathbb{R}^{N_{c_u}\cdot(N_t+1)}.$$

yields the standard NLP form.

The control restraint $\mathbf{c}_{\overline{u}}(\cdot)$ is chosen such that $\mathbf{c}_{\overline{u}}(\overline{\mathbf{u}}_{[k]}) \leq \mathbf{0}$ when $\overline{\mathbf{u}}_{[k]} \in \hat{\mathcal{U}}(kh),\ k = 0, \ldots, N_t$ and is twice continuously differentiable w.r.t. $\overline{\mathbf{u}}_{[k]}$.

Problem description (8.22)–(8.24) is referred to as *direct single shooting method*. The algorithmic procedure is summarized in Algorithm 8.2.

Algorithm 8.2 Direct Single Shooting Transcription

Require: $\overline{\mathbf{y}}$, $\mathbf{x}_0$
1: $\overline{\mathbf{x}}_{[0]} = \mathbf{x}_0$
2: **for** $k \leftarrow 0$ **to** $N_t - 1$ **do**
3: $\quad \overline{\mathbf{x}}_{[k+1]} \leftarrow \overline{\mathbf{x}}_{[k]} + h \cdot \boldsymbol{\Gamma}_f\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h\right)$
4: $\quad \overline{\tilde{\mathbf{x}}}_{[k+1]} = \overline{\tilde{\mathbf{x}}}_{[k]} + h \cdot \Gamma_l\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h\right)$
5: **end for**
6: evaluate $\mathbf{h}(\overline{\mathbf{y}})$
7: evaluate $\mathbf{g}(\overline{\mathbf{y}})$
8: evaluate $m(\overline{\mathbf{x}}_{[N_t]}) + \overline{\tilde{\mathbf{x}}}_{[N_t]}$
9: calculate the gradients $\nabla_{\overline{y}}\mathbf{h}(\overline{\mathbf{y}})$
10: calculate the gradients $\nabla_{\overline{y}}\mathbf{g}(\overline{\mathbf{y}})$
11: **return** $m(\overline{\mathbf{x}}_{[N_t]}) + \overline{\tilde{\mathbf{x}}}_{[N_t]}$, $\mathbf{h}(\overline{\mathbf{y}})$, $\mathbf{g}(\overline{\mathbf{y}})$, $\nabla_{\overline{y}}\mathbf{h}(\overline{\mathbf{y}})$, and $\nabla_{\overline{y}}\mathbf{g}(\overline{\mathbf{y}})$

The choice of the constraints $\mathbf{c}_{\overline{u}}(\cdot)$ as mentioned above does not guarantee that the control is feasible in the entire interval $[t_k, t_{k+1})$ but only on the boundary of the interval.

A disadvantage of the direct single shooting method is the dependence of the endpoint function $m(\overline{\mathbf{x}}_{[N_t]}) + \overline{\tilde{\mathbf{x}}}_{[N_t]}$ on all decision variables. Betts [10] pointed out that this fact causes limited stability since changes of the optimization variables at the beginning of the trajectory propagate over the differential equation to the end of the trajectory. This causes considerably nonlinear effects at the constraints with respect to the decision variables. Consequently, the optimization problem becomes hard to solve.

Analogously to indirect multiple shooting methods, one can simply break the problem into shorter steps to reduce the sensitivity of single shooting. This technique leads to a *direct multiple shooting method*. A framework for a multiple shooting algorithm was already proposed by Bock and Plitt [12] in 1984.

In order to enforce continuity for $\overline{\mathbf{x}}$, the following *matching conditions* are enforced at the end points of each phase

$$\overline{\mathbf{x}}_{[K_j]} - \mathbf{X}_j = \mathbf{0}, \quad j = 2, \ldots, N_d - 1$$

where $\mathbf{X}_j$ are the initial values of the states at phase j and $\overline{\mathbf{x}}_{[K_j]}$ are the final values of the states at phase $j-1$. The phases are usually defined on a smaller time grid $\mathcal{G}_{sh}$ (7.23) (cf. Chap. 7),

$$t_0 = t_{K_1} < t_{K_2} < \cdots < t_{K_{N_d}} = t_f, \quad \mathcal{G}_{sh} = \left\{t_{K_1}, t_{K_2}, \ldots, t_{K_{N_d}}\right\}, \quad \mathcal{G}_{sh} \subset \mathcal{G}_t$$

with $N_d < N_t$, where

$$\begin{aligned} &K_1 < K_2 < \cdots < K_{N_d} \\ &K_j \in \{1, \ldots, N_t - 1\}, \quad j = 2, \ldots, N_d - 1. \end{aligned}$$

The first node K_1 and the last node K_{N_d} of the shooting time grid $\mathcal{G}_{sh}$ are fixed and can not be chosen. They are set to $K_1 = 0$ and $K_{N_d} = N_t$.

Hence, the optimization vector

$$\overline{\mathbf{y}} = \left[\mathbf{u}_0, \ldots, \mathbf{u}_{N_t}, \mathbf{X}_2, \ldots, \mathbf{X}_{N_d-1}\right]^T = \left[\overline{\mathbf{u}}_{[0:N_t]}, \mathbf{X}_2, \ldots, \mathbf{X}_{N_d-1}\right]^T \in \mathbb{R}^{N_{\overline{y}}}$$

increases to the size of $N_{\overline{y}} = N_u \cdot (N_t + 1) + N_x \cdot (N_d - 2)$ NLP variables. The equality constraints are augmented with the continuity conditions from the multiple shooting transcription

$$\mathbf{h}(\overline{\mathbf{y}}) = \begin{bmatrix} \boldsymbol{\theta}_1 \\ \boldsymbol{\theta}_2 \\ \vdots \\ \boldsymbol{\theta}_{N_d-2} \\ \left(\overline{\mathbf{x}}_{[N_t]}\right)_{[\mathcal{I}_f]} - \mathbf{x}_f \end{bmatrix} = \mathbf{0}_{\left(N_x \cdot (N_d-2) + \#\mathcal{I}_f\right) \times 1} \tag{8.26}$$

where the vector $\boldsymbol{\theta}$ is defined as

$$\begin{aligned} \boldsymbol{\theta}_1 &= \overline{\mathbf{x}}_{[K_2]} - \mathbf{X}_2 \\ \boldsymbol{\theta}_2 &= \overline{\mathbf{x}}_{[K_3]} - \mathbf{X}_3 \\ &\vdots \\ \boldsymbol{\theta}_{N_d-2} &= \overline{\mathbf{x}}_{[K_{N_d-1}]} - \mathbf{X}_{N_d-1}. \end{aligned}$$

The inequality constraints are not altered,

$$\mathbf{g}(\overline{\mathbf{y}}) = \left[\mathbf{c}_{\overline{u}}(\overline{\mathbf{u}}_{[0]}), \mathbf{c}_{\overline{u}}(\overline{\mathbf{u}}_{[1]}), \ldots, \mathbf{c}_{\overline{u}}(\overline{\mathbf{u}}_{[N_t]})\right]^T \in \mathbb{R}^{N_{c_u} \cdot (N_t+1)}. \tag{8.27}$$

As noted before, the conditions $\overline{\mathbf{x}}_{[0]} = \mathbf{X}_1$ and $\overline{\mathbf{x}}_{[N_t]} = \mathbf{X}_{N_d}$ are boundary conditions and are therefore not considered as optimization variables, whereas the initial values of each shooting interval $\mathbf{X}_2, \ldots, \mathbf{X}_{N_d-1}$ can be varied freely.

Obviously, the multiple shooting transcription increases the problem size by $N_x \cdot (N_d - 2)$ NLP variables. Fortunately, the Jacobian matrix of the constraints is sparse, which is needed to compute an efficient sparse Quasi-Newton update. Despite of the increased problem size the multiple shooting technique can help to reduce the sensitivity of shooting due to significantly smaller integration intervals. The stabilization effect results from the additional optimization variables $\mathbf{X}_j$ at the beginning of each integration phase

$$\begin{aligned} \overline{\mathbf{x}}_{[K_j]} &= \mathbf{X}_j \\ \overline{\mathbf{x}}_{[k+1]} &= \overline{\mathbf{x}}_{[k]} + h \cdot \boldsymbol{\Gamma}_f \left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h \right), \quad k = K_j, \ldots, K_{j+1} - 1. \end{aligned}$$

Algorithmically, the procedure for evaluating the multiple shooting transcription is summarized in the Algorithm 8.3.

Algorithm 8.3 Direct Multiple Shooting Transcription

Require: $\overline{\mathbf{y}}$, $\mathbf{X}_1$
for $j \leftarrow 1$ **to** $N_d - 1$ **do**
  $\overline{\mathbf{x}}_{[K_j]} \leftarrow \mathbf{X}_j$
  for $k \leftarrow K_j$ **to** $K_{j+1} - 1$ **do**
    $\overline{\mathbf{x}}_{[k+1]} \leftarrow \overline{\mathbf{x}}_{[k]} + h \cdot \boldsymbol{\Gamma}_f \left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h \right)$
  end for
  $\boldsymbol{\theta}_j = \overline{\mathbf{x}}_{[K_{j+1}]} - \mathbf{X}_{j+1}$
end for
for $k \leftarrow 1$ **to** $N_t - 1$ **do**
  $\tilde{\overline{\mathbf{x}}}_{[k+1]} = \tilde{\overline{\mathbf{x}}}_{[k]} + h \cdot \Gamma_l \left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h \right)$
end for
assemble $\mathbf{h}(\overline{\mathbf{y}})$ using (8.26)
evaluate $\mathbf{g}(\overline{\mathbf{y}})$ using (8.27)
calculate the gradients $\nabla_{\overline{\mathbf{y}}} \mathbf{h}(\overline{\mathbf{y}})$
calculate the gradients $\nabla_{\overline{\mathbf{y}}} \mathbf{g}(\overline{\mathbf{y}})$
evaluate $m(\overline{\mathbf{x}}_{[N_t]}) + \tilde{\overline{\mathbf{x}}}_{[N_t]}$
return $m(\overline{\mathbf{x}}_{[N_t]}) + \tilde{\overline{\mathbf{x}}}_{[N_t]}$, $\mathbf{h}(\overline{\mathbf{y}})$, $\mathbf{g}(\overline{\mathbf{y}})$, $\nabla_{\overline{\mathbf{y}}} \mathbf{h}(\overline{\mathbf{y}})$, and $\nabla_{\overline{\mathbf{y}}} \mathbf{g}(\overline{\mathbf{y}})$

Remark 8.1 A transformation into a Mayer problem reduces the for-loop code fragment for the integration of the Lagrangian term.

A further disadvantage for both indirect shooting methods occurs, if state constraints have to be considered in the problem formulation (8.9)–(8.12). In this case, the Hessian of the Lagrangian becomes dense even if the Jacobians are partly sparse. Using direct shooting methods, this leads to an optimization problem which is very inefficient to solve. Therefore, for such problems, it is recommended to use the direct collocation transcription instead, which can be derived for the special case $N_d = N_t$ from the multiple indirect shooting method.

8.2.2 Direct Collocation

Historically, two different branches of *direct collocation* methods evolved, which is shown in Fig. 8.4. On the one hand, low-order direct collocation which has been first introduced by Tsang et al. [71] and originated from the forward simulation of ODEs. On the other hand, pseudospectral methods originally evolved in the context of partial differential equations within fluid dynamics, which are not described in this book. In the sequel, we use the terminology "direct collocation" for the used low-order direct collocation.

In contrast to the direct shooting method, where only the continuous-valued controls are discretized, direct collocation methods discretize the continuous-valued states too. Let us apply (8.15) again for the discretization of the continuous-valued controls and any Runge–Kutta scheme from Chap. 5 for the discretization of the continuous-valued states,

$$\begin{aligned}\overline{\mathbf{x}}_{[k+1]} &= \overline{\mathbf{x}}_{[k]} + h \cdot \boldsymbol{\Gamma}_f \left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h\right), \quad k = 0, \ldots, N_t - 1 \\ \overline{\mathbf{x}}_{[0]} &= \mathbf{x}_0.\end{aligned}$$

Then, the optimization vector $\overline{\mathbf{y}}$ includes the discretized continuous-valued states $\overline{\mathbf{x}}_{[k]}$ as well. Consequently, the optimization vector is defined as

$$\overline{\mathbf{y}} = \left[\mathbf{u}_0, \ldots, \mathbf{u}_{N_t}, \mathbf{x}_1, \ldots, \left(\mathbf{x}_{N_t}\right)_{[\mathcal{I}_f^c]}\right]^T = \left[\overline{\mathbf{u}}_{[0:N_t]}, \overline{\mathbf{x}}_{[1:N_t]}\right]^T \in \mathbb{R}^{N_{\overline{y}}} \tag{8.28}$$

with the size $N_{\overline{y}} = N_u \cdot (N_t + 1) + N_x \cdot (N_t - 1) + \#\mathcal{I}_f^c$, where $\mathcal{I}_f^c$ is the complementary set, which is defined by $\mathcal{I}_f^c = \{1, \ldots, N_x\} \setminus \mathcal{I}_f$.

The fully discretized optimal control problem can then be stated as NLP formulation

$$\min_{\overline{\mathbf{y}} \in \mathbb{R}^{N_{\overline{y}}}} \; \phi(\overline{\mathbf{y}}) = m\left(\overline{\mathbf{x}}^*_{[N_t]}\right) + \tilde{\overline{\mathbf{x}}}^*_{[N_t]} \tag{8.29}$$

subject to

$$\overline{\mathbf{x}}_{[k+1]} - \overline{\mathbf{x}}_{[k]} - h \cdot \boldsymbol{\Gamma}_f(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, k, h) = \mathbf{0}_{N_x \times 1}, \quad k = 0, \ldots, N_t - 1 \tag{8.30}$$

$$\overline{\mathbf{x}}_{[0]} = \mathbf{x}_0 \tag{8.31}$$

$$\left(\overline{\mathbf{x}}_{[N_t]}\right)_{[\mathcal{I}_f]} - \mathbf{x}_f = \mathbf{0}_{\#\mathcal{I}_f \times 1} \tag{8.32}$$

$$\mathbf{c}_{\overline{u}}(\overline{\mathbf{u}}_{[k]}) \leq \mathbf{0}_{N_{c_u} \times 1}, \quad k = 0, \ldots, N_t \tag{8.33}$$

$$\mathbf{c}_{\overline{x}}(\overline{\mathbf{x}}_{[k]}) \leq \mathbf{0}_{N_{c_x} \times 1}, \quad k = 1, \ldots, N_t. \tag{8.34}$$

Remark 8.2 The discretization scheme for continuous-valued controls (8.15) can be exchanged with higher order ones (see for instance, von Stryk [65] and Büskens [18]).

The incremental step of the RK scheme (8.30) enforces the fulfillment of the ODE. The boundary conditions (8.31) and (8.32) imply that $\overline{\mathbf{x}}_{[0]}$ and $\left(\overline{\mathbf{x}}_{[N_t]}\right)_{[\mathcal{I}_f]}$ can not be varied and are therefore not part of the optimization vector. The discretized state vector

$$\overline{\mathbf{x}} = \left[\mathbf{x}_0, \ldots, \mathbf{x}_{N_t}\right]^T$$

is then assembled using the boundary conditions and the part of the optimization vector (8.28) dedicated to the continuous-valued states $\mathbf{x}$.

We summarize the equality and inequality constraints constraints to the convenient vector notations

$$\begin{aligned}\mathbf{h}(\overline{\mathbf{u}}, \overline{\mathbf{x}}) = &\begin{bmatrix} \overline{\mathbf{x}}_{[1]} - \overline{\mathbf{x}}_{[0]} - h \cdot \boldsymbol{\Gamma}_f(\overline{\mathbf{x}}_{[0]}, \overline{\mathbf{x}}_{[1]}, \overline{\mathbf{u}}_{[0]}, \overline{\mathbf{u}}_{[1]}, 0, h) \\ \vdots \\ \overline{\mathbf{x}}_{[N_t-1]} - \overline{\mathbf{x}}_{[N_t-2]} - h \cdot \boldsymbol{\Gamma}_f(\overline{\mathbf{x}}_{[N_t-2]}, \overline{\mathbf{x}}_{[N_t-1]}, \overline{\mathbf{u}}_{[N_t-2]}, \overline{\mathbf{u}}_{[N_t-1]}, N_t-2, h) \\ \overline{\mathbf{x}}_{[N_t]} - \overline{\mathbf{x}}_{[N_t-1]} - h \cdot \boldsymbol{\Gamma}_f(\overline{\mathbf{x}}_{[N_t-1]}, \overline{\mathbf{x}}_{[N_t]}, \overline{\mathbf{u}}_{[N_t-1]}, \overline{\mathbf{u}}_{[N_t]}, N_t-1, h) \\ \left(\overline{\mathbf{x}}_{[N_t]}\right)_{[\mathcal{I}_f]} - \mathbf{x}_f \end{bmatrix} \\ &= \mathbf{0}_{(N_x \cdot N_t + \#\mathcal{I}_f)\times 1}\end{aligned} \tag{8.35}$$

and

$$\mathbf{g}(\overline{\mathbf{u}}, \overline{\mathbf{x}}) = \begin{bmatrix} \mathbf{c}_{\overline{u}}(\overline{\mathbf{u}}_{[0]}), \mathbf{c}_{\overline{u}}(\overline{\mathbf{u}}_{[1]}), \ldots, \mathbf{c}_{\overline{u}}(\overline{\mathbf{u}}_{[N_t]}) \\ \mathbf{c}_{\overline{x}}(\overline{\mathbf{x}}_{[1]}), \mathbf{c}_{\overline{x}}(\overline{\mathbf{x}}_{[2]}), \ldots, \mathbf{c}_{\overline{x}}(\overline{\mathbf{x}}_{[N_t]}) \end{bmatrix}^T \leq \mathbf{0}_{(N_{c_u} \cdot (N_t+1) + N_{c_x} \cdot N_t)\times 1}. \tag{8.36}$$

For the direct collocation transcription any RK scheme can be applied. However, higher order integration schemes increases the required computing power and storage space. The sparsity property of the Jacobian matrix will be discussed in Chap. 9.

Algorithm 8.4 Direct Collocation Transcription

Require: $\overline{\mathbf{y}}$

1: extract discretized continuous-valued controls and assign it to the discretized control vector, i.e., $\overline{\mathbf{u}} \leftarrow \overline{\mathbf{y}}_{[0:N_t]}$
2: set the boundary conditions to the discretized state vector, i.e., $\overline{\mathbf{x}}_{[0]} = \mathbf{x}_0$ and $\left(\overline{\mathbf{x}}_{[N_t]}\right)_{[\mathcal{I}_f]} = \mathbf{x}_f$
3: extract discretized continuous-valued states and assign it to the discretized state vector, i.e., $\overline{\mathbf{x}}_{[1:N_t]} \leftarrow \overline{\mathbf{y}}_{[(N_t+2):2N_t+1]}$
4: evaluate $\mathbf{h}(\overline{\mathbf{u}}, \overline{\mathbf{x}})$ using (8.35)
5: evaluate $\mathbf{g}(\overline{\mathbf{u}}, \overline{\mathbf{x}})$ using (8.36)
6: evaluate $m(\overline{\mathbf{x}}_{[N_t]}) + \bar{\bar{\mathbf{x}}}_{[N_t]}$
7: calculate the gradients $\nabla_{\overline{y}}\mathbf{h}(\overline{\mathbf{x}}, \overline{\mathbf{u}})$
8: calculate the gradients $\nabla_{\overline{y}}\mathbf{g}(\overline{\mathbf{x}}, \overline{\mathbf{u}})$
9: **return** $m(\overline{\mathbf{x}}_{[N_t]}) + \bar{\bar{\mathbf{x}}}_{[N_t]}$, $\mathbf{h}(\overline{\mathbf{u}}, \overline{\mathbf{x}})$, $\mathbf{g}(\overline{\mathbf{u}}, \overline{\mathbf{x}})$, $\nabla_{\overline{y}}\mathbf{h}(\overline{\mathbf{x}}, \overline{\mathbf{u}})$, and $\nabla_{\overline{y}}\mathbf{g}(\overline{\mathbf{x}}, \overline{\mathbf{u}})$

8.2.3 Comparison of Direct Shooting and Direct Collocation

Direct shooting methods lead to smaller problem sizes, which is beneficial if NLP solvers are used with an algebra kernel for dense matrices. Since sparse matrix kernels were not widely applicable for a long time, these methods were very popular. Direct collocation approaches were also successfully applied only to small-and medium-sized problems ($N_{\overline{y}} \approx 10^3$ (ct. Betts [11])), because of the lack of sparse matrix kernels. However, the additional discrete state representation increases the number of elements by $N_x \cdot (N_t - 1) + \#\mathcal{I}_f^c$. Looking at the numerical solution procedure, this will require far more function evaluations for the gradient estimation and significantly more memory for storing the Hessian matrix. This caused the fact that direct collocation methods were not competitive with direct shooting methods for a long time. The transfer to large-scale engineering problems (thousands of variables and constraints) were prevented by the high number of NLP variables. This obstacle has been solved in the last years by exploiting sparse matrix structures of the problem that occur naturally due to the fact that many variables in the optimization vector are independent of each other. On the one hand, this can reduce dramatically the number of required function evaluations for the gradient calculation, on the other hand, sparse matrix algebra can also be applied for solving the quadratic subproblem in the *sequential quadratic programming* (SQP) procedure. These advanced topics are discussed in Chap. 9. The numerical treatment of sparse large-scale problems and the benefit of dealing easily with state constraints made the direct collocation methods superior to direct shooting methods.

8.2.4 Recovering the Costates from a Direct Shooting and Direct Collocation

One of the major disadvantages of direct transcription methods for the solution of continuous OCPs is the fact that the costates $\boldsymbol{\lambda}(\cdot)$ are not obtained from the solution in a direct manner. However, knowledge of the costates can be very helpful as it allows for the evaluation of the fulfillment of first-order necessary conditions and in many cases, the costates can provide helpful insight into the structure of the solution. An elegant way to recover the costates from the solution of the discretized optimal control problem is the *post-optimal* calculation. Methods for post-optimal recovering of the costates were proposed by Enright and Conway [24], von Stryk [65] for the direct collocation method and by Büskens [18] for the direct shooting method.

The method described by Büskens [18] takes the transversality condition at the final time

$$\boldsymbol{\lambda}(t_f) = \frac{\partial m}{\partial \mathbf{x}(t_f)}(\mathbf{x}^*(t_f)) + \hat{\boldsymbol{\mu}}_f + \left(\frac{\partial \mathbf{c}_x}{\partial \mathbf{x}}\right)^T_{t=t_f}(\mathbf{x}^*(t_f)) \cdot \boldsymbol{\alpha}_f \tag{8.37}$$

and applies a backward integration for the differential equation of the costates

$$\dot{\boldsymbol{\lambda}}(t) = -\frac{\partial \mathcal{H}_a}{\partial \mathbf{x}}(\mathbf{x}^*(t), \boldsymbol{\lambda}(t), \lambda_0, \boldsymbol{\gamma}(t), \boldsymbol{\rho}(t), \mathbf{u}^*(t)),$$

where $\boldsymbol{\alpha}_f \in \mathbb{R}^{N_{cx}}$ is a vector of Lagrange multipliers and $\hat{\boldsymbol{\mu}}_f$ is defined as

$$\hat{\boldsymbol{\mu}}_f := \begin{cases} \hat{\boldsymbol{\mu}}_f^{[\mathcal{I}_f]} = \boldsymbol{\mu}_f \\ \hat{\boldsymbol{\mu}}_f^{[\mathcal{I}_f^c]} = \mathbf{0}. \end{cases}$$

The derivation of these conditions can be found in Sect. 4.2.2. Please note, that the transversality condition holds for all costates, even if it is only required for the costates indicated by the set $\mathcal{I}_f^c$ in the necessary conditions.

The required Lagrange parameters of the endpoint transversality condition are returned from the NLP solver, if a direct shooting or collocation method is used for the transcription of the OCP. We exploit here the fact that the Lagrange parameters, i.e., $\boldsymbol{\mu}_f \in \mathbb{R}^{\#\mathcal{I}_f}$ and $\overline{\boldsymbol{\rho}}_{[k]} \in \mathbb{R}^{N_{cx}}$ for $k = 1, \ldots, N_t$ for the state constraints $\mathbf{c}_{\overline{x}}(\overline{\mathbf{x}}_{[k]}) \leq \mathbf{0}$, are always included in the Lagrangians of the discretized optimization problems. The endpoint transversality condition for the discretized OCP in Mayer form is

$$\tilde{\boldsymbol{\lambda}}_{N_t} = \frac{\partial m}{\partial \left(\overline{\mathbf{x}}_{[N_t]}\right)_{[\mathcal{I}_f^c]}}(\overline{\mathbf{x}}_{[N_t]}) + \hat{\boldsymbol{\mu}}_f + \left(\frac{\partial \mathbf{c}_{\overline{x}}}{\partial \left(\overline{\mathbf{x}}_{[N_t]}\right)_{[\mathcal{I}_f^c]}}\right)^T (\overline{\mathbf{x}}_{[N_t]}) \cdot \overline{\boldsymbol{\rho}}_{[N_t]},$$

where $\overline{\boldsymbol{\rho}}_{[N_t]}$ corresponds to $\boldsymbol{\alpha}_f$. It is recommended to solve the costate ODEs by a RK method, as described in Sect. 5.3, on the same time grid $\mathcal{G}_t$, that was used by the transcription method

$$\tilde{\boldsymbol{\lambda}}_k = \left[\mathbf{I} + h \cdot \frac{\partial \boldsymbol{\Gamma}_f}{\partial \overline{\mathbf{x}}_{[k]}}\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h\right)\right]^T \cdot \tilde{\boldsymbol{\lambda}}_{k+1} + \left(\frac{\partial \mathbf{c}_{\overline{x}}}{\partial \overline{\mathbf{x}}_{[k]}}\right)^T (\overline{\mathbf{x}}_{[k]}) \cdot \overline{\boldsymbol{\rho}}_{[k]}, \quad k = N_t - 1, \ldots, 0$$

where $\mathbf{I}$ is the unity matrix of dimension $N_x \times N_x$. It is also advisable to use a method with the same consistency order to ensure that the error is of the same order as the solution of the state ODEs. The required partial derivatives $\partial \boldsymbol{\Gamma}_f / \partial \overline{\mathbf{x}}_{[k]}$ and $\partial \mathbf{c}_{\overline{x}} / \partial \overline{\mathbf{x}}_{[k]}$ can be computed analytically or by finite differences.

Obviously, the described method can also be implemented as a forward integration scheme applied on the initial transversality condition

$$\boldsymbol{\lambda}(t_0) = -\frac{\partial m}{\partial \mathbf{x}(t_0)}(\mathbf{x}^*(t_0)) - \boldsymbol{\mu}_0 - \left(\frac{\partial \mathbf{c}_x}{\partial \mathbf{x}}\right)_{t=t_0}^T (\mathbf{x}^*(t_0)) \cdot \boldsymbol{\alpha}_0,$$

where $\boldsymbol{\alpha}_0 \in \mathbb{R}^{N_{cx}}$ is a vector of Lagrange multipliers. A drawback of this variant is, that the Lagrange parameters for the initial transversality condition are not always included in the Lagrangian of the discretized OCP, but they can easily be added to the implementation. The initial transversality condition for the discretized OCP in Mayer form is

$$\tilde{\boldsymbol{\lambda}}_0 = -\frac{\partial m}{\partial \left(\overline{\mathbf{x}}_{[0]}\right)}(\overline{\mathbf{x}}_{[0]}) - \boldsymbol{\mu}_0 - \left(\frac{\partial \mathbf{c}_{\overline{x}}}{\partial \left(\overline{\mathbf{x}}_{[0]}\right)}\right)^T (\overline{\mathbf{x}}_{[0]}) \cdot \overline{\boldsymbol{\rho}}_{[0]}$$

where $\overline{\boldsymbol{\rho}}_{[0]}$ corresponds to $\boldsymbol{\alpha}_0$. The corresponding forward integration scheme can be stated as

$$\tilde{\boldsymbol{\lambda}}_{k+1} = \left[\mathbf{I} - h \cdot \frac{\partial \boldsymbol{\Gamma}_f}{\partial \overline{\mathbf{x}}_{[k]}}\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{x}}_{[k+1]}, \overline{\mathbf{u}}_{[k]}, \overline{\mathbf{u}}_{[k+1]}, t_k, h\right)\right]^T \cdot \tilde{\boldsymbol{\lambda}}_k - \left(\frac{\partial \mathbf{c}_{\overline{x}}}{\partial \overline{\mathbf{x}}_{[k]}}\right)^T (\overline{\mathbf{x}}_{[k]}) \cdot \overline{\boldsymbol{\rho}}_{[k]}, \quad k = 1, \ldots, N_t.$$

The jumps of the costates within state constrained arcs are automatically generated by the forward or backward integration, because they are included in the Lagrange parameters $\overline{\boldsymbol{\rho}}_{[k]}$.

For direct transcriptions the recovering of the costates can be computed by one of these two methods. Furthermore, for the direct collocation transcription an additional procedure is possible. The Lagrange multipliers for the RK difference equation (8.30) already approximate the costates at the midpoints of the discretization grid, as it is shown in von Stryk [65]. Consequently, the approximated costates at the original grid points can be obtained from the solution of the nonlinear programming method by a simple interpolation routine, whereby the costates at the initial time t_0 and the final time t_f can be obtained by an extrapolation or the derivation of the transversality conditions.

8.3 Optimal Control for Switched Systems

As mentioned in the introductory section of this chapter a SOCP can be solved by fixing the switching sequence and solving the remaining continuous optimal control problem using direct transcription methods. This has the potential drawback that the number of switchings is typically not known in advance. More advanced branch-and-X methods can perform only satisfactorily on limited and small discretization grids because of the exponentially growing complexity of the problem (Till et al. [70]).

We propose in this section two algorithms for the solution of SOCPs with continuous states $\mathbf{x}(\cdot)$:

- embedded optimal control problem; and
- two-stage method.

The advantage of these methods is that no a priori assumptions on the number of switchings, the switching time instances, and the switching mode sequence are necessary. For the case that the solution trajectories do not meet the required accuracy one can refine the trajectories using a switching time optimization.

8.3.1 Embedded Optimal Control Problem

The control vector of the binary switched system

$$\dot{\mathbf{x}}(t) = \mathbf{F}(\mathbf{x}(t), \boldsymbol{\rho}(t))$$

concatenates the continuous-valued and discrete controls to

$$\boldsymbol{\rho}(t) = [\mathbf{u}(t), \boldsymbol{\sigma}(t)].$$

The admissible set $\hat{\mathcal{U}}(q(t), t) \times \{0, 1\}^{N_q}$ can be split into N_q-subsets which may not be connected to each other and consequently the admissible set must not be convex. However, convexity of optimization problems plays an important role and has strong implications for the numerical solution procedures. For instant, SQP (see Chap. 2) may fail to generate optimal and reliable solutions if applied to non-convex problems. Thus, let us assume, for the sake of simplicity, that the continuous-valued controls may be chosen from a common convex set. In practical scenarios, this will often be the case.

The main idea of the embedding method is the relaxation of the binary controls to obtain a continuous-valued approximation of the BSOCP. The relaxation of the Boolean vector $\boldsymbol{\sigma}(t) \in \{0, 1\}^{N_q}$ yields to

$$\hat{\boldsymbol{\sigma}}(t) \in [0, 1]^{N_q} \tag{8.38}$$

whose elements are taken from a compact set. For the sake of better transparency we denote $\hat{\boldsymbol{\sigma}}(\cdot)$ as the relaxed binary controls. We keep the notation $\boldsymbol{\sigma}(\cdot)$ for binary feasible controls.

The control vector concatenates the continuous-valued and relaxed binary controls to

$$\hat{\boldsymbol{\rho}}(t) = [\mathbf{u}(t), \hat{\boldsymbol{\sigma}}(t)]$$

and may take on values from the convex set $\hat{\mathcal{P}} = \hat{\mathcal{U}}(q(t), t) \times [0, 1]^{N_q}$. The dynamical system can now be treated as a conventional system $\mathbf{F} : \mathsf{X} \times \hat{\mathcal{P}} \to \mathsf{X}$ without discontinuity phenomena

$$\dot{\hat{\mathbf{x}}}(t) = \mathbf{F}(\hat{\mathbf{x}}(t), \hat{\boldsymbol{\rho}}(t)) = \sum_{q=1}^{N_q} \hat{\boldsymbol{\sigma}}_q(t) \cdot \mathbf{f}_{q(t)}(\hat{\mathbf{x}}(t), \mathbf{u}(t)), \quad \forall t \in [t_0, t_f] \tag{8.39}$$

where the states $\hat{\mathbf{x}}(\cdot)$ are marked with a hat to illustrate that the state trajectory is obtained from the embedded system description. The convexified problem formulation can be solved with a NLP solver, if one of the direct transcription methods as discussed in Sects. 8.2.1 and 8.2.2, is applied, whereby the binary controls $\hat{\boldsymbol{\sigma}}(\cdot)$ are best approximated by piecewise constant functions

$$\hat{\boldsymbol{\sigma}}(t) = \begin{cases} \Xi_k^{\hat{\sigma}}\left(\hat{\boldsymbol{\sigma}}_k, \hat{\boldsymbol{\sigma}}_{k+1}, t\right) = \dfrac{\hat{\boldsymbol{\sigma}}_k + \hat{\boldsymbol{\sigma}}_{k+1}}{2}, & \forall t \in \left[t_k, t_{k+1}\right], \quad k = 0, \ldots, N_t - 2 \\ \Xi_{N_t-1}^{\hat{\sigma}}\left(\hat{\boldsymbol{\sigma}}_{N_t-1}, \hat{\boldsymbol{\sigma}}_{N_t}, t\right) = \dfrac{\hat{\boldsymbol{\sigma}}_{N_t-1} + \hat{\boldsymbol{\sigma}}_{N_t}}{2}, & \forall t \in \left[t_{N_t-1}, t_{N_t}\right]. \end{cases}.$$

A direct shooting method for the convexified problem yields

$$\min_{\overline{\hat{\boldsymbol{\rho}}}} \phi\left(\overline{\hat{\boldsymbol{\rho}}}\right) = m\left(\overline{\hat{\mathbf{x}}}^*_{[N_t]}\right) \tag{8.40}$$

subject to

$$\mathbf{g}\left(\overline{\hat{\boldsymbol{\rho}}}\right) = \begin{bmatrix} \mathbf{c}_{\overline{u}}\left(\overline{\mathbf{u}}_{[k]}\right), & k = 0, \ldots, N_t \\ -\overline{\hat{\boldsymbol{\sigma}}}_{[k]}, & k = 0, \ldots, N_t \\ \overline{\hat{\boldsymbol{\sigma}}}_{[k]} - \mathbb{1}_{N_q \times 1}, & k = 0, \ldots, N_t \end{bmatrix} \leq \mathbf{0}_{(N_u + 2N_q)\cdot(N_t+1)} \tag{8.41}$$

$$\mathbf{h}\left(\overline{\hat{\boldsymbol{\rho}}}\right) = \begin{bmatrix} \left(\overline{\hat{\mathbf{x}}}_{[N_t]}\right)_{[\mathcal{I}_f]} - \mathbf{x}_f \\ \sum_{q=1}^{N_q} \overline{\hat{\boldsymbol{\sigma}}}_q^{[k]} - 1, \quad k = 0, \ldots, N_t \end{bmatrix} = \mathbf{0}_{\#\mathcal{I}_f + N_q\cdot(N_t+1)}. \tag{8.42}$$

A direct collocation transcription for the convexified problem yields

$$\min_{\overline{\hat{\boldsymbol{\rho}}}} \phi\left(\overline{\hat{\boldsymbol{\rho}}}\right) = m\left(\overline{\hat{\mathbf{x}}}^*_{[N_t]}\right) \tag{8.43}$$

subject to

$$\mathbf{g}\left(\overline{\hat{\boldsymbol{\rho}}}\right) = \begin{bmatrix} \mathbf{c}_{\overline{u}}\left(\overline{\mathbf{u}}_{[k]}\right), & k = 0, \ldots, N_t \\ \mathbf{c}_{\overline{\hat{x}}}\left(\overline{\hat{\mathbf{x}}}_{[k]}\right), & k = 1, \ldots, N_t \\ -\overline{\hat{\boldsymbol{\sigma}}}_{[k]}, & k = 0, \ldots, N_t \\ \overline{\hat{\boldsymbol{\sigma}}}_{[k]} - \quad_{N_q \times 1}, & k = 0, \ldots, N_t \end{bmatrix} \leq \mathbf{0}_{(N_u + 2N_q)\cdot(N_t+1) + N_x \cdot N_t} \tag{8.44}$$

$$\mathbf{h}\left(\overline{\hat{\boldsymbol{\rho}}}\right) = \begin{bmatrix} \overline{\hat{\mathbf{x}}}_{[k+1]} - \overline{\hat{\mathbf{x}}}_{[k]} - h \cdot \boldsymbol{\Gamma}_F\left(\overline{\hat{\mathbf{x}}}_{[k]}, \overline{\hat{\mathbf{x}}}_{[k+1]}, \overline{\hat{\boldsymbol{\rho}}}_{[k]}, \overline{\hat{\boldsymbol{\rho}}}_{[k+1]}, t_k, h\right), & k = 0, \ldots, N_t - 1 \\ \left(\overline{\hat{\mathbf{x}}}_{[N_t]}\right)_{[\mathcal{I}_f]} - \mathbf{x}_f & \\ \sum_{q=1}^{N_q} \overline{\hat{\boldsymbol{\sigma}}}_q^{[k]} - 1, & k = 0, \ldots, N_t \end{bmatrix}$$
$$= \mathbf{0}_{N_x \cdot N_t + \#\mathcal{I}_f + N_q\cdot(N_t+1)}. \tag{8.45}$$

Problem formulations (8.40)–(8.42) and (8.43)–(8.45) are called *embedded optimal control problems* (EOCP). Please note that $\boldsymbol{\Gamma}_F(\cdot)$ is the increment function from the RK scheme for the convexified dynamical system (8.39). To assess the first-order necessary conditions we define the Hamiltonian by

$$\mathcal{H}\left(\overline{\hat{\mathbf{x}}}_{[k]}, \overline{\boldsymbol{\lambda}}_{[k]}, \overline{\hat{\boldsymbol{\rho}}}_{[k]}\right) = \overline{\boldsymbol{\lambda}}_{[k]}^T \cdot \sum_{q=1}^{N_q} \overline{\hat{\boldsymbol{\sigma}}}_q^{[k]} \cdot \mathbf{f}_q\left(\overline{\hat{\mathbf{x}}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right). \tag{8.46}$$

In the Hamiltonian (8.46) the control $\overline{\hat{\boldsymbol{\sigma}}}_{[k]}$ appears linearly, which allows us to state the switching function

$$\frac{\partial \mathcal{H}}{\partial \overline{\hat{\boldsymbol{\sigma}}}_q^{[k]}}\left(\overline{\hat{\mathbf{x}}}_{[k]}, \overline{\boldsymbol{\lambda}}_{[k]}, \overline{\hat{\boldsymbol{\rho}}}_{[k]}\right) = S_q\left(\overline{\hat{\mathbf{x}}}_{[k]}, \overline{\boldsymbol{\lambda}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right) = \overline{\boldsymbol{\lambda}}_{[k]}^T \mathbf{f}_q\left(\overline{\hat{\mathbf{x}}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right) = 0$$

for all $q = 1, \ldots, N_q$. From Theorem 4.6 we know that the relaxed binary control may take on values from the boundaries, if the switching function has a sign, i.e.,

$$\overline{\hat{\boldsymbol{\sigma}}}_q^{[k]} = \begin{cases} 1 & S_q\left(\overline{\hat{\mathbf{x}}}_{[k]}, \overline{\boldsymbol{\lambda}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right) < 0 \\ 0 & S_q\left(\overline{\hat{\mathbf{x}}}_{[k]}, \overline{\boldsymbol{\lambda}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right) > 0 \end{cases} \tag{8.47}$$

for each $q = 1, \ldots, N_q$. For the case $S_q(\overline{\hat{\mathbf{x}}}_{[k]}, \overline{\boldsymbol{\lambda}}_{[k]}, \overline{\mathbf{u}}_{[k]}) = 0,\ k = 0, \ldots, N_t$, the relaxed binary control values satisfy the condition $\overline{\hat{\boldsymbol{\sigma}}}_q^{[k]} \in (0, 1)$ and the corresponding arcs are called *singular arcs* (cf. Sect. 4.2.3). Consequently, a relaxed binary control trajectory $\overline{\hat{\boldsymbol{\sigma}}}$ is called binary admissible, if $\overline{\hat{\boldsymbol{\sigma}}}_{[k]} \in \{0, 1\}^{N_q},\ k = 0, \ldots, N_t$ is satisfied. If the relaxed binary control trajectory $\overline{\hat{\boldsymbol{\sigma}}}$ is binary admissible and the trajectory $(\overline{\hat{\mathbf{x}}}, \overline{\hat{\boldsymbol{\sigma}}}, \overline{\mathbf{u}})$ is also feasible, then the solution is valid to the original BSOCP. For this case, the BSOCP can be considered as a conventional optimal control problem with continuous-valued controls. Otherwise, if the relaxed binary control trajectory $\overline{\hat{\boldsymbol{\sigma}}}$ is not completely binary admissible, the trajectory of the EOCP must be post-processed such that a feasible trajectory for the BSOCP is obtained. In this case, the BSOCP can not be solved in a classical manner and is indeed an hybrid optimal control problem. Homotopy or post-optimal approximation strategies for obtaining binary feasible trajectories are discussed in Sect. 8.4.

The embedding procedure is summarized in Algorithm 8.5.

Algorithm 8.5 Embedding Method

1: embedding of the original BSOCP into a larger family of continuous problems using (8.38)
2: applying an appropriate direct transcription: (8.40)–(8.42) or (8.43)–(8.45)
3: solving the NLP
4: applying a rounding strategy if singular arcs are presented (see Sect. 8.4)

8.3.2 Two-Stage Algorithm

Indirect shooting methods produce highly accurate solution trajectories for SOCPs but their applications can be impaired by the difficulty to make a good initial guess, i.e., of the costates. This obstacle can lead to a small domain of convergence for many practical problems. In contrast, direct transcription methods work efficiently and robustly on problems with a predefined discrete state sequence. No costates are required for the solution procedure but the costates can be recovered from the solution of the direct algorithm after the termination. It is therefore obvious to combine some characteristics of both principles to gain a more robust optimization strategy for SOCPs. This leads to the class of two-stage algorithms. Two-stage algorithms have a high computational cost but improve the accuracy and enlarge the domain of convergence.

Let us consider, the SOCP (8.1)–(8.6) formulated as Mayer problem

$$\min_{\mathbf{u}(\cdot)\in\mathcal{U}(q(\cdot)),\ q(\cdot)\in\mathcal{Q}} \phi\left(\mathbf{u}(\cdot), q(\cdot)\right) = m(\mathbf{x}^*(t_f)) \tag{8.48}$$

subject to

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)), \qquad \text{for a.e. } t \in [t_0, t_f] \tag{8.49}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{8.50}$$

$$\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f \tag{8.51}$$

where the Hamiltonian is defined by

$$\mathcal{H}(\mathbf{x}(t), q(t), \boldsymbol{\lambda}(t), \mathbf{u}(t)) = \boldsymbol{\lambda}^T(t) \cdot \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)).$$

Our intention is to insert an additional mode $q^\star$ into the switching sequence $\Theta = ((t_0, q_0), (t_1, q_1), (t_2, q_2))$ for a time interval Δt centered at time $t^\star$ with $t_1 < t^\star - 0.5\Delta t$ and $t^\star + 0.5\Delta t < t_2$, such that the new switching schedule is obtained as $\Theta = ((t_0, q_0), (t_1, q_1), (t^\star - 0.5\Delta t, q^\star), (t^\star + 0.5\Delta t, q_1), (t_2, q_2))$. The principle is illustrated in Fig. 8.5.

A necessary condition of the *hybrid minimum principle* (HMP) for SOCPs without state jumps is that the Hamiltonian before and after a change in the switching sequence must be identical, i.e.,

$$\mathcal{H}(\mathbf{x}(t_j^-), q(t_j^-), \boldsymbol{\lambda}(t_j^-), \mathbf{u}(t_j^-)) = \mathcal{H}(\mathbf{x}(t_j^+), q(t_j^+), \boldsymbol{\lambda}(t_j^+), \mathbf{u}(t_j^+)). \tag{8.52}$$

This motivates to reformulate (8.52) as a descent condition for the new mode $q(t^\star)$

$$\Delta\mathcal{H} = \mathcal{H}(\mathbf{x}(t^\star), q(t^\star), \boldsymbol{\lambda}(t^\star), \mathbf{u}(t^\star)) - \mathcal{H}(\mathbf{x}(t^\star), q(t_{j+1}), \boldsymbol{\lambda}(t^\star), \mathbf{u}(t^\star)), \tag{8.53}$$

for $[t^\star - 0.5\Delta t, t^\star + 0.5\Delta t] \subset [t_j, t_{j+1}],\ j = 0, \ldots, N_{swt}$. Egerstedt et al. [23] and Axelsson et al. [3] proved that the descent condition is a well-defined partial derivative of the cost function with respect to the switching time $t^\star$:

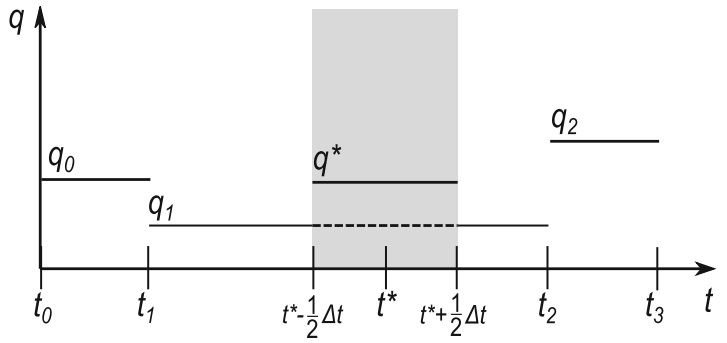


Fig. 8.5 Insertion of the new mode $q^\star$ in the interval $\left[t^\star - \frac{1}{2}\Delta t, t^\star + \frac{1}{2}\Delta t\right]$ (Schori [60])

$$\frac{\partial \phi}{\partial t^\star} = \Delta \mathcal{H}.$$

This partial derivative was also derived in Xu and Antsaklis [74] and Kamgarpour and Tomlin [35].

The insertion of a mode $q^\star$ may yield a decrease in the cost function value, when the descent condition (8.53) is negative with respect to the insertion. It is therefore desirable to alter the switching sequence, where the descent condition (8.53) has the lowest negative value

$$q(t^\star) = \arg\min \Delta \mathcal{H}. \tag{8.54}$$

Let us now discretize (8.48)–(8.51) using a direct shooting transcription which yields

$$\begin{aligned}
\min_{\overline{\mathbf{y}} \in \mathbb{R}^{N_{\overline{y}}},\, \overline{\mathbf{q}} \in \mathbb{N}_{>0}^{N_{\overline{q}}}} \quad & \phi(\overline{\mathbf{x}}^*_{[N_t]}) = m(\overline{\mathbf{x}}_{[N_t]}) & & \\
\text{subject to} \quad & \overline{\mathbf{x}}_{[0]} & = \mathbf{x}_0 & \\
& \left(\overline{\mathbf{x}}_{[N_t]}\right)_{[\mathcal{I}_f]} - \mathbf{x}_f & = \mathbf{0}_{\#\mathcal{I}_f \times 1} & \\
& \mathbf{c}_{\overline{u},\overline{q}}(\overline{\mathbf{u}}_{[k]}) & \leq \mathbf{0}_{N_{c_u} \times 1}, & k = 0, \ldots, N_t \\
& \mathbf{c}_{\overline{q}}(\overline{\mathbf{q}}_{[k]}) & \leq \mathbf{0}_{N_{c_q} \times 1}, & k = 0, \ldots, N_t
\end{aligned}$$

where $\mathbf{c}_{\overline{q}}(\cdot)$ are the discrete state constraints.

Based on this observation, the following two-stage algorithm is introduced:

1. **in the first stage**: a direct shooting transcription is applied for finding the optimal continuous-valued controls for an initial guess of the discrete state sequence $\overline{\mathbf{q}}$. Once the NLP-solver terminates, discretized continuous-valued states $\overline{\mathbf{x}}$, and continuous-valued controls $\overline{\mathbf{u}}$ as well as a set of Lagrange multipliers $\boldsymbol{\mu}_f$ are obtained for the assumed switching sequence $\overline{\mathbf{q}}$. With an approximation of the

costates $\boldsymbol{\lambda}(\cdot)$ using the recovery procedure presented in Sect. 8.2.4, the Hamiltonian can be evaluated

$$\mathcal{H}\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{q}}_{[k]}, \overline{\boldsymbol{\lambda}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right) = \overline{\tilde{\boldsymbol{\lambda}}}_{[k]}^{T} \cdot \mathbf{f}_{\overline{q}_{[k]}}\left(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right);$$

and

2. **in the second stage**: the switching schedule $\overline{\mathbf{q}}$ is altered based on the evaluations of (8.54). The schedule is altered at the time instant k, where the largest descent in the Hamiltonian function can be achieved by altering $\overline{\mathbf{q}}_{[k]}$ at this time instant.

The algorithm alternates between these two stages, until a termination criterion is fulfilled. As such criterion may serve

$$\Delta\mathcal{H}_{min} < \epsilon \tag{8.55}$$

where $\Delta\mathcal{H}_{min}$ is the largest difference between the Hamiltonian calculated for any q at one of the time instants k and the current Hamiltonian calculated at the same time instant. Due to the fact that the costates $\overline{\tilde{\boldsymbol{\lambda}}}$ are only an approximation, numerical errors may prevent the fulfillment of the termination criterion (8.55). Thus, other termination criteria might be necessary.

The overall algorithm uses here a direct shooting transcription but can also be implemented with a direct collocation transcription. The main function TwoStageMethod accepts an initial guess of $\overline{\mathbf{q}}$ and uses an ordinary, non-sparse NLP-solver to find optimized continuous-valued controls $\overline{\mathbf{u}}$ for this switching sequence. The solver iteratively calls the function IVP to calculate the state trajectory $\overline{\mathbf{x}}$ such that the cost function can be evaluated. After the completion of the nonlinear optimization, the state vector $\overline{\mathbf{x}}$ is computed for the corresponding continuous-valued controls and the given switching sequence.

The Lagrange multipliers $\boldsymbol{\mu}_f$ for the final states are returned by the NLP-solver, which are necessary for the costate recovery. The function CoStates calculates an approximation of the costates $\boldsymbol{\lambda}(\cdot)$ by a backward integration method as described in Sect. 8.2.4. The calculation by a forward integration is of course also possible.

Once $\overline{\tilde{\boldsymbol{\lambda}}}$ is returned to the main function TwoStageMethod, the minimum Hamiltonian function values $\mathcal{H}_q^k$ can be computed for every time instant k and for every discrete state q and are compared against the current Hamiltonian function values $\mathcal{H}_{cur}^k$. The combination of time instants k and the discrete state q in code line 17 from Algorithm 8.6 yields the largest decrease in the Hamiltonian function at a time instant k. The switching schedule is then altered respectively. These steps are repeated until the termination criterion is fulfilled.

The overall algorithm is rather slow, since the switching schedule is altered on each iteration at one time instant k only. More instants can be modified during each iteration, but is generally not recommended.

Algorithm 8.6 Two-Stage Algorithm (Schori [60])

function TWOSTAGEMETHOD
Define ϵ, $\bar{\mathbf{q}}_{init}$, $\bar{\mathbf{u}}_{init}$, $\bar{\mathbf{x}}_{[0]}$, $\bar{\mathbf{t}} = \mathcal{G}_t$
$\bar{\mathbf{q}} \leftarrow \bar{\mathbf{q}}_{init}$
$\bar{\mathbf{u}} \leftarrow \bar{\mathbf{u}}_{init}$
$\Delta\mathcal{H}_{min} \leftarrow -\infty$
while $\Delta\mathcal{H}_{min} < \epsilon$ **do**
  $(\bar{\mathbf{u}}, \boldsymbol{\mu}_f) \leftarrow \text{NLP}(\bar{\mathbf{q}}, \bar{\mathbf{u}}, \bar{\mathbf{t}})$
  $\bar{\mathbf{x}} \leftarrow \text{IVP}(\bar{\mathbf{q}}, \bar{\mathbf{u}}, \bar{\mathbf{t}}, \bar{\mathbf{x}}_{[0]})$
  $\bar{\tilde{\boldsymbol{\lambda}}} \leftarrow \text{COSTATES}(\bar{\mathbf{x}}, \bar{\mathbf{q}}, \bar{\mathbf{u}}, \bar{\mathbf{t}}, \boldsymbol{\mu}_f)$
  $\mathcal{H}^k_{cur} \leftarrow \bar{\tilde{\boldsymbol{\lambda}}}^T_{[k]} \cdot \mathbf{f}\left(\bar{\mathbf{x}}_{[k]}, \bar{\mathbf{q}}_{[k]}, \bar{\mathbf{u}}_{[k]}\right), \quad k = 0, 1, \ldots, N_t$
  for $k \leftarrow 0$ **to** N_t **do**
    for all $q \in \hat{\mathcal{Q}}$ **do**
      $\mathcal{H}^k_q \leftarrow \min_{\mathbf{u} \in \hat{\mathcal{U}}(kh,q)} \mathcal{H}\left(\bar{\mathbf{x}}_{[k]}, q, \bar{\tilde{\boldsymbol{\lambda}}}_{[k]}, \mathbf{u}\right)$
      $\mathbf{u}^k_q \leftarrow \arg\min_{\mathbf{u} \in \hat{\mathcal{U}}(kh,q)} \mathcal{H}\left(\bar{\mathbf{x}}_{[k]}, q, \bar{\tilde{\boldsymbol{\lambda}}}_{[k]}, \mathbf{u}\right)$
    end for
  end for
  $(i, j) \leftarrow \min_{q,k}\left(\mathcal{H}^k_q - \mathcal{H}^k_{cur}\right)$
  $\bar{\mathbf{q}}_{[j]} \leftarrow i$
  $\bar{\mathbf{u}}_{[j]} \leftarrow \mathbf{u}^j_i$
  $\Delta\mathcal{H}_{min} \leftarrow \mathcal{H}^j_i - \mathcal{H}^j_{cur}$
  $\mathcal{H}^j_{cur} \leftarrow \mathcal{H}^j_i$
end while
return $\bar{\mathbf{x}}, \bar{\mathbf{u}}, \bar{\boldsymbol{\lambda}}, \bar{\mathbf{q}}$
end function

function IVP($\bar{\mathbf{q}}, \bar{\mathbf{u}}, \bar{\mathbf{t}}, \bar{\mathbf{x}}_{[0]}$)
for $k \leftarrow 0$ **to** $N_t - 1$ **do**
  $h_k = t_{k+1} - t_k$
  $\bar{\mathbf{x}}_{[k+1]} \leftarrow \bar{\mathbf{x}}_{[k]} + h_k \cdot \boldsymbol{\Gamma}_f\left(\bar{\mathbf{x}}_{[k]}, \bar{\mathbf{q}}_{[k]}, \bar{\mathbf{q}}_{[k+1]}, \bar{\mathbf{u}}_{[k]}, \bar{\mathbf{u}}_{[k+1]}, t_k, h_k\right)$
end for
return $\bar{\mathbf{x}}$
end function

function COSTATES($\bar{\mathbf{x}}, \bar{\mathbf{q}}, \bar{\mathbf{u}}, \bar{\mathbf{t}}, \boldsymbol{\mu}_f$)
$\bar{\tilde{\boldsymbol{\lambda}}}_{[N_t]} = \dfrac{\partial m}{\partial\left(\bar{\mathbf{x}}_{[N_t]}\right)_{[\mathcal{I}^c_f]}}(\bar{\mathbf{x}}_{[N_t]}) + \hat{\boldsymbol{\mu}}_f$
for $k \leftarrow N_t - 1$ **to** 0 **do**
  $h_k = t_{k+1} - t_k$
  $\bar{\tilde{\boldsymbol{\lambda}}}_{[k]} = \left[\mathbf{I} + h_k \cdot \dfrac{\partial \boldsymbol{\Gamma}_f}{\partial \bar{\mathbf{x}}_{[k]}}\left(\bar{\mathbf{x}}_{[k]}, \bar{\mathbf{q}}_{[k]}, \bar{\mathbf{q}}_{[k+1]}, \bar{\mathbf{u}}_{[k]}, \bar{\mathbf{u}}_{[k+1]}, t_k, h_k\right)\right]^T \cdot \bar{\tilde{\boldsymbol{\lambda}}}_{[k+1]}$
end for
return $\bar{\tilde{\boldsymbol{\lambda}}}$
end function

8.3.3 Switching Time Optimization with Parameterized Switching Intervals

The solution trajectories obtained from Sect. 8.3.1 can be further refined with respect to the switching times. The corresponding optimization is called *switching time optimization* (STO) and can obtain, depending on the initial estimation of the switching times, a remarkable improvement of accuracy of the switching time instants t_j and thus of the cost value.

The main step of the switching time optimization is a variable time transformation which transcribes the SOCP into an equivalent continuous optimal control problem without discrete control variables but parameterized by the switching arcs. This requires that the feasible initial solution to the SOCP is decomposed into an hybrid execution sequence. This new formulation is continuously differentiable in all optimization variables and allows the numerical calculation of derivatives with respect to the switching arcs. Figure 8.6 illustrates the idea of representing a binary solution using switching arcs.

In Fig. 8.6, the number of stages N_ς and the mode sequence $\mathcal{D} = (\sigma_0, \sigma_1, \ldots, \sigma_6)$ have to be known, but the switching arcs $\boldsymbol{\varsigma} = [\varsigma_0, \varsigma_1, \ldots, \varsigma_6]$ can be varied.

For the time interval $[t_0, t_f]$ a variable time transformation is used according to Gerdts [28]. Therefore, a main time grid $\mathcal{G}_{t_2}$ and a minor time grid $\mathcal{G}_{t_3}$ will be introduced additionally to the standard time grid $\mathcal{G}_t$. The main time grid $\mathcal{G}_{t_2}$ contains fixed grid points $T_i,\ i = 1, \ldots, N_{t_2}$ which cannot be varied by the optimization and is defined by

$$t_0 = T_1 \leq T_2 \leq \cdots \leq T_{N_{t_2}} = t_f, \qquad \mathcal{G}_{t_2} := \left\{T_1, T_2, \ldots, T_{N_{t_2}}\right\} \tag{8.56}$$

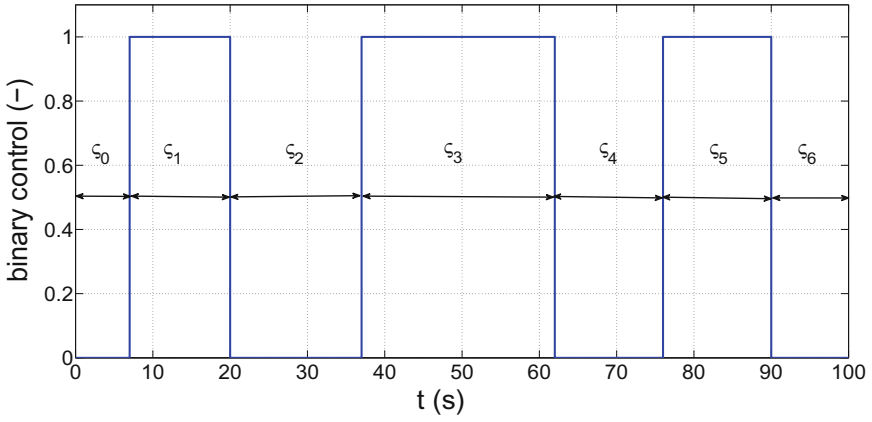


Fig. 8.6 Illustration of the switching arc representation for seven subintervals for a system with $N_q = 1$ over the time interval $\forall t \in [0, 100]$

with $N_{t_2} \in \mathbb{N}_{>0}$ intervals. The minor time grid $\mathcal{G}_{t_3}$ is chosen depending on the solution of the discrete controls obtained by the prior optimization approach. The sequence of the discrete controls $\{\boldsymbol{\sigma}_j\}_{j=0,\ldots,N_{t_3}-1}$ with piecewise constant functions $\boldsymbol{\sigma}(t) = \boldsymbol{\sigma}_j$ for $t \in [t_j, t_{j+1}]$ is used to define the minor time grid by

$$t_0 \leq t_1 \leq t_2 \leq \cdots \leq t_{N_{t_3}} = t_f, \qquad \mathcal{G}_{t_3} := \left\{t_0, t_1, \ldots, t_{N_{t_3}}\right\} \tag{8.57}$$

where $\mathcal{G}_{t_2} \subseteq \mathcal{G}_{t_3} \subseteq \mathcal{G}_t$ holds. It should be noted that the minor time grid is non-equidistant, because the grid depends on the switching arcs of $\boldsymbol{\sigma}(\cdot)$.

Now, the idea is to define an appropriate variable time transformation $t = \tilde{t}(\tau)$ in order to be able to control the length of the transformed minor time grid intervals $[t_j, t_{j+1}]$ by an additional function $\varsigma(\cdot)$. The general form of the time transformation is given by

$$\tilde{t}(\tau) := t_0 + \int_{t_0}^{\tau} \varsigma(s)\, \mathrm{d}s, \quad \tau \in [t_0, t_f] \tag{8.58}$$

with

$$\int_{t_0}^{t_f} \varsigma(s)\, \mathrm{d}s = t_f - t_0$$

where τ is a new time variable. We impose that the function $\varsigma(\cdot)$ is from the class of piecewise constant functions, i.e., $\varsigma(t) = \varsigma_j$ for $t \in [t_j, t_{j+1}]$. This allows us to define ς_j as the length of the minor time grid intervals by

$$\varsigma_j := t_{j+1} - t_j, \qquad j = 0, \ldots, N_{t_3} - 1. \tag{8.59}$$

Defining a function $\alpha : \mathbb{N} \to \mathbb{N}$ that assigns each index of the main time grid $\mathcal{G}_{t_2}$ to the corresponding index of the minor time grid $\mathcal{G}_{t_3}$, i.e., $j = \alpha(i)$ for $T_i = t_j$, allows us to obtain a relationship between the main time grid and the minor time grid with

$$\sum_{j=\alpha(i)}^{\alpha(i+1)-1} \varsigma_j = T_{i+1} - T_i, i = 1, \ldots, N_{t_2} - 1.$$

The values of the binary controls $\boldsymbol{\sigma}_j$ within each main grid interval $[T_i, T_{i+1}]$ are deleted if the corresponding minor time grid intervals ς_j approaches zero. By applying (8.59) we obtain a special case of the variable time transformation (8.58) which converts each minor grid interval onto the unity interval $\tau = [0, 1]$ with

$$\begin{aligned} \tilde{t}_j(\tau) &:= t_j + \int_0^{\tau} \varsigma_j\, \mathrm{d}s, \\ &= t_j + \tau \varsigma_j, \end{aligned}$$

for $j = 0, \ldots, N_{t_3} - 1$. This transformation can be found in [46] and has the positive effect that the continuous-valued states and controls are independent from the length of the minor time grid intervals ς_j, $j = 0, \ldots, N_{t_3} - 1$ which allows that the continuous-valued states and controls can be discretized by the methods discussed in Chap. 5.

The continuous-valued states $\mathbf{x}_j(\cdot)$, the continuous-valued controls $\mathbf{u}_j(\cdot)$, and binary controls $\boldsymbol{\sigma}_j(\cdot)$ are now functions depending on the new time variable $\tau \in [0, 1]$. The functions $\tilde{\mathbf{x}}_j : [0, 1] \to \mathsf{X}$, $\tilde{\mathbf{u}}_j : [0, 1] \to \mathsf{U}$, and $\tilde{\boldsymbol{\sigma}}_j : [0, 1] \to \Omega$ are then defined on each minor time grid interval by

$$\begin{aligned} \tilde{\mathbf{x}}_j(\tau) &:= \mathbf{x}(t_j + \tau\varsigma_j), \\ \tilde{\mathbf{u}}_j(\tau) &:= \mathbf{u}(t_j + \tau\varsigma_j), \\ \tilde{\boldsymbol{\sigma}}_j(\tau) &:= \boldsymbol{\sigma}(t_j + \tau\varsigma_j), \end{aligned} \tag{8.60}$$

for $\tau \in [0, 1]$ and $j = 0, \ldots, N_{t_3} - 1$. The evolution of the continuous-valued states (8.60) is a solution to the ODE system

$$\dot{\tilde{\mathbf{x}}}_j(\tau) = \varsigma_j \cdot \mathbf{f}_{\tilde{q}_j(\tau)}(\tilde{\mathbf{x}}_j(\tau), \tilde{\mathbf{u}}_j(\tau)), \quad j = 0, \ldots, N_{t_3} - 1$$

which is concatenated by the continuity conditions

$$\tilde{\mathbf{x}}_j(0) = \tilde{\mathbf{x}}_{j-1}(1), \quad j = 1, \ldots, N_{t_3} - 1.$$

The variable time transformation has the following consequences:

- the interval $[t_j, t_{j+1}]$ shrinks to the single point $\{t_j\}$ if $\varsigma_j = 0$; and
- the derivation of the parametrized time $\tilde{t}_j(\tau)$ with respect to the new time variable τ yields

$$\frac{\mathrm{d}\tilde{t}_j}{\mathrm{d}\tau}(\tau) = \varsigma_j, \quad j = 0, \ldots, N_{t_3} - 1.$$

Then, the continuous optimal control problem for the switching time formulation can be stated as:

Definition 8.3 (*Switching Time Optimization with Main and Minor Time Grids*) Let the binary controls, continuous-valued controls, continuous-valued states, and constraints be defined on the normalized horizon by $\tilde{\boldsymbol{\sigma}}_j : [0, 1] \to \{0, 1\}^{N_q}$ with $N_q \in \mathbb{N}_{>0}$, $\tilde{\mathbf{u}}_j : [0, 1] \to \mathsf{U}$, $\tilde{\mathbf{x}}_j : [0, 1] \to \mathsf{X}$, $\mathbf{f}_{\tilde{q}} : \mathsf{X} \times \mathsf{U} \to \mathsf{X}$, $\mathbf{c}_{\tilde{u}} : \mathsf{U} \to \mathbb{R}^{N_{cu}}$, and $\mathbf{c}_{\tilde{x}} : \mathsf{X} \to \mathbb{R}^{N_{cx}}$, respectively. Then, a switching time optimization with main time grid (8.56) and minor time grid (8.57) can be formulated as Mayer problem:

$$\min_{\tilde{\mathbf{x}}_j(\tau),\, \tilde{\mathbf{u}}_j(\tau),\, \varsigma \in \mathbb{R}^{N_{t_3}}} m\left(\tilde{\mathbf{x}}_{N_{t_3}-1}(1)\right) \tag{8.61}$$

subject to

$$\frac{\mathrm{d}\tilde{\mathbf{x}}_j}{\mathrm{d}\tau}(\tau) = \varsigma_j \cdot \sum_{q=1}^{N_q} \tilde{\boldsymbol{\sigma}}_j^{[q]}(\tau) \cdot \mathbf{f}_{\tilde{q}(\tau)}(\tilde{\mathbf{x}}_j(\tau), \tilde{\mathbf{u}}_j(\tau)), \quad j = 0, \ldots, N_{t_3} - 1, \quad \forall \tau \in [0,1] \tag{8.62}$$

$$\tilde{\mathbf{x}}_0(0) = \mathbf{x}_0 \tag{8.63}$$

$$(\tilde{\mathbf{x}}_{N_{t_3}-1}(1))_{[\mathcal{I}_f]} = \mathbf{x}_f \tag{8.64}$$

$$\sum_{j=\alpha(i)}^{\alpha(i+1)-1} \varsigma_j - T_{i+1} + T_i = 0, \quad i = 1, \ldots, N_{t_2} - 1 \tag{8.65}$$

$$\tilde{\mathbf{x}}_j(0) - \tilde{\mathbf{x}}_{j-1}(1) = \mathbf{0}_{N_x \times 1}, \quad j = 1, \ldots, N_{t_3} - 1 \tag{8.66}$$

$$-\varsigma_j \leq 0, \quad j = 0, \ldots, N_{t_3} - 1 \tag{8.67}$$

$$\mathbf{c}_{\tilde{u}}(\tilde{\mathbf{u}}_j(\tau)) \leq \mathbf{0}_{N_{c_u} \times 1}, \quad j = 0, \ldots, N_{t_3} - 1, \quad \forall \tau \in [0,1] \tag{8.68}$$

$$\mathbf{c}_{\tilde{x}}(\tilde{\mathbf{x}}_j(\tau)) \leq \mathbf{0}_{N_{c_x} \times 1}, \quad j = 0, \ldots, N_{t_3} - 1, \quad \forall \tau \in [0,1]. \tag{8.69}$$

△

One can observe that the constraint (8.66) couples the endpoint states $\tilde{\mathbf{x}}_j(0)$ of time intervals $j \in \{1, \ldots, N_{t_3} - 1\}$ with the endpoint states $\tilde{\mathbf{x}}_{j-1}(1)$ of the previous time intervals. These sequential constraints are known as *linkage conditions* or *coupling conditions*.

The STO (8.61)–(8.69) can be solved by Runge–Kutta discretization on individual time grids $\mathcal{G}_t^j$ for each switching arc ς_j, $j = 0, \ldots, N_{t_3} - 1$, which yields the discretized state equation

$$\bar{\tilde{\mathbf{x}}}_j^{[k+1]} = \bar{\tilde{\mathbf{x}}}_j^{[k]} + h \cdot \Gamma_f \left(\bar{\tilde{\mathbf{x}}}_j^{[k]}, \bar{\tilde{\mathbf{x}}}_j^{[k+1]}, \bar{\tilde{\mathbf{u}}}_j^{[k]}, \bar{\tilde{\mathbf{u}}}_j^{[k+1]}, \bar{\tilde{\boldsymbol{\sigma}}}_j^{[k]}, \bar{\tilde{\boldsymbol{\sigma}}}_j^{[k+1]}, t_k, h \right), \quad t_k \in \mathcal{G}_t^j.$$

The time grids $\mathcal{G}_t^j$ must be chosen such that the endpoints of the switching arcs are met. Applying a direct collocation method yields the NLP formulation:

$$\min_{\bar{\tilde{\mathbf{x}}},\, \bar{\tilde{\mathbf{u}}},\, \varsigma \in \mathbb{R}^{N_{t_3}}} \phi\left(\bar{\tilde{\mathbf{x}}}_{N_{t_3}-1}^{[N_t]}\right) = m\left(\left(\bar{\tilde{\mathbf{x}}}_{N_{t_3}-1}^{*}\right)_{[N_t]}\right) \tag{8.70}$$

subject to

$$\mathbf{g}\left(\bar{\tilde{\mathbf{x}}}, \bar{\tilde{\mathbf{u}}}\right) = \begin{bmatrix} \mathbf{c}_{\bar{\tilde{u}}}\left(\bar{\tilde{\mathbf{u}}}_j^{[k]}\right), & j = 0, \ldots, N_{t_3} - 1,\ k = 0, \ldots, N_t \\ \mathbf{c}_{\bar{\tilde{x}}}\left(\bar{\tilde{\mathbf{x}}}_0^{[k]}\right), & j = 0,\ k = 1, \ldots, N_t \\ \mathbf{c}_{\bar{\tilde{x}}}\left(\bar{\tilde{\mathbf{x}}}_j^{[k]}\right), & j = 1, \ldots, N_{t_3} - 1,\ k = 0, \ldots, N_t \\ -\varsigma_j, & j = 0, \ldots, N_{t_3} - 1 \end{bmatrix} \leq \mathbf{0} \tag{8.71}$$

$$\mathbf{h}\left(\bar{\tilde{\mathbf{x}}}, \bar{\tilde{\mathbf{u}}}\right) = \begin{bmatrix} \bar{\tilde{\mathbf{x}}}_j^{[k+1]} - \bar{\tilde{\mathbf{x}}}_j^{[k]} - h \cdot \boldsymbol{\Gamma}_f\left(\bar{\tilde{\mathbf{x}}}_j^{[k]}, \bar{\tilde{\mathbf{x}}}_j^{[k+1]}, \bar{\tilde{\mathbf{u}}}_j^{[k]}, \bar{\tilde{\mathbf{u}}}_j^{[k+1]}, \bar{\tilde{\boldsymbol{\sigma}}}_j^{[k]}, \bar{\tilde{\boldsymbol{\sigma}}}_j^{[k+1]}, t_k, h\right), \\ j = 0, \ldots, N_{t_3} - 1,\ k = 0, \ldots, N_t - 1 \\ \left(\bar{\tilde{\mathbf{x}}}_{N_{t_3}-1}^{[N_t]}\right)_{[\mathcal{I}_f]} - \mathbf{x}_f \\ \sum_{j=\bar{\boldsymbol{\alpha}}_{[i]}}^{\bar{\boldsymbol{\alpha}}_{[i+1]}-1} \varsigma_j - T_{i+1} + T_i,\ i = 1, \ldots, N_{t_2} - 1 \\ \bar{\tilde{\mathbf{x}}}_j^{[0]} - \bar{\tilde{\mathbf{x}}}_{j-1}^{[N_t]},\ j = 1, \ldots, N_{t_3} - 1 \end{bmatrix} = \mathbf{0}. \tag{8.72}$$

According to Sager [54] the reformulation of a SOCP to a STO can result in additional non-convexities in the optimization space and thus requires a good initial guess for the arc lengths $\varsigma_j,\ j = 0, \ldots, N_{t_3} - 1$ and initial values $\tilde{\mathbf{x}}_j(0) = \mathbf{x}(t_j),\ j = 0, \ldots, N_{t_3} - 1$ at the switching time instances. A recommended measure is to use the trajectories of an EOCP with rounding strategy. Embedded optimal control problems provide good guesses for the number of switching arcs as well as initial values for the trajectories with moderate computing power.

If an arc length is reduced to zero $\varsigma_j = 0$ by the NLP solver, the corresponding switching time is undetermined and leads to a non-regular situation. This situation requires special care to be taken. A pragmatic solution is the deletion of this switching arc if the control and state constraints will not be violated.

8.4 Numerical Methods for Obtaining Binary Feasible Control Functions

Solution trajectories from embedding approaches may have singular arcs. A naive procedure to obtain binary feasible solutions would be an isolated round-up and down by ceil $\lceil\cdot\rceil$ and floor $\lfloor\cdot\rfloor$ operators to the nearest binary value for each time instant $t_k \in \mathcal{G}_t$. In general, this is not a good idea since rounded solutions are often poor solutions or even infeasible. Indeed, it is not difficult to construct examples with this naive rounding procedure that will not work. For instant, continuous-valued states obtained from an ODE with rounded binary feasible controls $\overline{\boldsymbol{\sigma}}$ may deviate considerably from the continuous-valued states of the relaxed counterpart.

Bengea and DeCarlo [5] suggested, that a binary feasible trajectory can be calculated, such that the state trajectory is arbitrarily close to the infeasible state trajectory. This rises the question for numerical algorithms how to obtain binary admissible solutions and how to measure the performance of any relaxed solution to a specific binary solution obtained by a rounding procedure.

A strategy for dealing with singular solutions can be the attachment of complementary constraints to the NLP problem. These formulations are known as *mathematical program with complementary constraints* (MPCC). For a system with two binary controls, the respective constraint for any time instant can be written as $\sigma_1(\cdot) \perp \sigma_2(\cdot)$, which means that either $\sigma_1(\cdot)$ or $\sigma_2(\cdot)$ is zero at time t. This constraint can be formulated by

$$\begin{aligned} \sigma_1(t) \cdot (1 - \sigma_1(t)) + \sigma_2(t) \cdot (1 - \sigma_2(t)) &= 0 \\ \sigma_1(t) + \sigma_2(t) &= 1. \end{aligned}$$

The first equation is usually applied to the nonlinear programing problem as soft constraint by penalizing the non-fulfillment of the complementary condition. For the general case, we obtain the constraints

$$\begin{aligned} \sum_{q=1}^{N_q} \sigma_q(t) \cdot \big(1 - \sigma_q(t)\big) = 0, \quad \sigma_q \in [0, 1] \\ \sum_{q=1}^{N_q} \sigma_q(t) = 1 \end{aligned} \tag{8.73}$$

for all binary controls. However, MPCCs have the potential drawback that the control set is not convex and even not compact. Therefore, the constraint qualifications MFCQ and LICQ (cf. 2.14 and 2.15) do not hold. This produces the observable result that NLPs can sometimes not consistently be solved. This raised the issue in the mathematical community whether MPCCs can be regarded as numerically unsafe. Indeed, MPCCs have to be treated with some caution but can perform very well as reported by Leyffer [42].

The complementary constraints (8.73) can be ameliorated to a certain extent by introducing the formulation

$$\sum_{q=1}^{N_q} \sigma_q(t) \cdot \big(1 - \sigma_q(t)\big) \leq \gamma, \quad \sigma_q \in [0, 1],$$

where the factor γ tends to zero by repeating the optimization process. The process $\gamma \to 0$ is known as homotopy. Consequently, the method is called *homotopy method* (Kirches [36]).

A practical way to realize a homotopy approach from above is to append the penalty terms (8.73) to the cost function which yields

$$\hat{\phi} := \phi + \alpha \sum_{q=1}^{N_q} \int_{t_0}^{t_f} \sigma_q(t) \cdot \left(1 - \sigma_q(t)\right) \, \mathrm{d}t$$

where α is a weighting factor. In general, α increases and gives more weight to the penalty terms (8.73) until they hopefully vanish. The success of this approach depends strongly upon the choice of α and the problem formulation itself. We employ the homotopy approach based on the proposal by Schäfer [59] to avoid singular solutions in Chap. 13.

A numerical investigation of this approach has also been performed by Sager [54]. A methodology to minimize a bad choice of the α factor is the successive increase of the penalty term using

$$\alpha_i = \alpha_0 \alpha_i^{inc}, \qquad i = 0, 1, 2, \ldots.$$

A more recent strategy to deal with the lack of regularity is to introduce *nonlinear complementarity problem* (NCP) functions. An overview of different NCP functions that can be used for implementing the complementary constraints is given by Leyffer [42]. Unfortunately, these functions lead to other numerical difficulties, as they are usually non-convex and in many cases locally not continuously differentiable.

Sager [54] recommended the application of a combination of rounding strategies for the binary variable $\boldsymbol{\sigma}(\cdot)$ based on switching time optimizations and a penalty term homotopy. He analyzed different rounding strategies to obtain a suboptimal solution. A treatment of error bounds for a rounding strategy can be found in the article of Sager et al. [58]. Several rounding strategies are presented in Sager [54, 55]. Among them is:

Definition 8.4 (*Sum-Up Rounding*) Let $\hat{\boldsymbol{\sigma}} : [t_0, t_f] \to [0, 1]^{N_q}$ be a piecewise constant function defined by

$$\begin{aligned} \hat{\sigma}_j(t) &:= b_{j,i}, \qquad t \in [t_i, t_{i+1}), \quad i \in \{0, \ldots, N_{t4} - 2\} \\ \hat{\sigma}_j(t) &:= b_{j,N_{t4}-1}, t \in [t_{N_{t4}-1}, t_{N_{t4}}] \end{aligned}$$

on a fixed mesh grid $\mathcal{G}_{t4}$ with $t_0 = t_0 < \cdots < t_{N_{t4}} = t_f$. Let the binary feasible control function $\boldsymbol{\sigma} : [t_0, t_f] \to \{0, 1\}^{N_q}$, which satisfies constraint (3.20), be defined by

$$\begin{aligned} \sigma_j(t) &:= a_{j,i}, \qquad t \in \left[t_i, t_{i+1}\right), \quad i \in \{0, \ldots, N_{t4} - 2\} \\ \sigma_j(t) &:= a_{j,N_{t4}-1}, \qquad t \in \left[t_{N_{t4}-1}, t_{N_{t4}}\right]. \end{aligned}$$

Then, the binary values $a_{j,i}$ are approximated from the relaxed function values $b_{j,i}$ by:

$$\tilde{a}_{j,i} := \sum_{k=1}^{i} b_{j,k}\Delta t_k - \sum_{k=1}^{i-1} a_{j,k}\Delta t_k$$

$$a_{j,i} := \begin{cases} 1 & \text{if } \tilde{a}_{j,i} > \tilde{a}_{k,i} \quad \forall k \neq j \\ 1 & \text{elseif } \tilde{a}_{j,i} = \tilde{a}_{k,i} \text{ and } j < k \\ 0 & \text{else} \end{cases}$$

where $\Delta t_k = t_{k+1} - t_k$. Consequently, $\boldsymbol{\sigma}(\cdot)$ is the result of the sum-up rounding procedure applied to $\hat{\boldsymbol{\sigma}}(\cdot)$.

△

Figure 8.7 illustrates the sum-up rounding. The introduced mesh grid in Definition 8.4 can be different to the one used in the direct optimization method. For the case that the chosen mesh grid $\mathcal{G}_{t4}$ equals $\mathcal{G}_t$, the constants $b_{j,i}$ are exactly the values from the discretized control vector $\overline{\hat{\boldsymbol{\sigma}}}$.

For an estimation of the sum-up rounding error let us consider affine differential equations of the relaxed controls $\hat{\boldsymbol{\sigma}}(\cdot)$ and the binary controls $\boldsymbol{\sigma}(\cdot)$, respectively, and their connections. For this reason, let us assume that an IVP of the following form is given

$$\dot{\mathbf{y}}(t) = \mathbf{A}(\mathbf{y}(t))\hat{\boldsymbol{\sigma}}(t), \qquad \mathbf{y}(t_0) = \mathbf{y}_0, \tag{8.74}$$

where the entries of the matrix $\mathbf{A}(\mathbf{y}) \in \mathbb{R}^{N_y} \times \mathbb{R}^{N_q}$ depend on $\mathbf{y}(\cdot)$. The following theorem states how the difference of the solution trajectories to this IVP (8.74) depends on the integrated difference between control functions.

Theorem 8.1 (Sum-Up Rounding Error [56]) *Suppose Definition 8.4 holds. Then, let* $\mathbf{y}(\cdot)$ *and* $\mathbf{z}(\cdot)$ *be solutions of the IVPs*

$$\begin{aligned} \dot{\mathbf{y}}(t) &= \mathbf{A}(\mathbf{y}(t))\hat{\boldsymbol{\sigma}}(t), & \mathbf{y}(t_0) &= \mathbf{y}_0 \\ \dot{\mathbf{z}}(t) &= \mathbf{A}(\mathbf{z}(t))\boldsymbol{\sigma}(t), & \mathbf{z}(t_0) &= \mathbf{z}_0 = \mathbf{y}_0 \end{aligned}$$

with $t \in [t_0, t_f]$. *If positive numbers* $L_A,\ M \in \mathbb{R}_{>0}$ *exist such that for all* $t \in [t_0, t_f]$

$$\begin{aligned} \left\|\hat{\boldsymbol{\sigma}}(t)\right\| &\leq 1 \\ \|\boldsymbol{\sigma}(t)\| &\leq 1 \\ \|\mathbf{A}(\mathbf{z}(t))\| &\leq M & \forall \mathbf{z}(t) &\in \mathbb{R}^{N_y} \\ \|\mathbf{A}(\mathbf{y}(t)) - \mathbf{A}(\mathbf{z}(t))\| &\leq L_A \cdot \|\mathbf{y}(t) - \mathbf{z}(t)\| & \forall \mathbf{y}(t), \mathbf{z}(t) &\in \mathbb{R}^{N_y} \end{aligned}$$

applies, then it holds that

$$\left\| \int_{t_0}^{t_f} \hat{\boldsymbol{\sigma}}(t) - \boldsymbol{\sigma}(t)\, \mathrm{d}t \right\| \leq (N_q - 1)\Delta t$$

then it also holds

$$\|\mathbf{y}(t) - \mathbf{z}(t)\| \leq \left(2M \cdot (N_q - 1) \cdot \max_{i=1,\ldots,N_t-1} \{t_{i+1} - t_i\}\right) e^{L_A \cdot t}$$

for all $t \in [t_0, t_f]$.

Proof The proof is given in Sager et al. [56]. □

Theorem 8.1 allows us to relate the difference between the states $\mathbf{y}(\cdot)$ and $\mathbf{z}(\cdot)$ where $\mathbf{y}(\cdot)$ is obtained from any relaxed solution and $\mathbf{z}(\cdot)$ is a specific binary solution obtained by sum-up rounding to the mesh grid size. In other words, the maximum allowed error can be chosen by selecting the mesh grid fineness from the direct method appropriately. Hence, the error can be made arbitrarily small. In Sager et al. [55, 56], it is shown that similar results can be obtained for different sum-up rounding strategies. For nonautonomous systems $\mathbf{A}(\cdot, t)$, Sager et al. [57, 58] presented an extended error bound.

Another class of rounding procedures that find binary feasible solutions can be realized as branch-and-bound algorithms. We follow the definition in Turau [72] and Schori [60] and present the following greedy-based rounding strategy:

The tuples $\mathcal{J}$ and $\mathcal{A}$ are introduced that define constraints to be respected, when solving the NLP. The tuple $\mathcal{J}$ contains index-pairs $(q, k),\ k = 0, \ldots, N_t,\ q \in \hat{\mathcal{Q}}$ and the tuple $\mathcal{A}$ contains values a from the set $\{0, 1\}$. For each pair of elements from $\mathcal{J}$ and $\mathcal{A}$, the constraint

$$\overline{\hat{\boldsymbol{\sigma}}}_q^{[k]} = a$$

is additionally imposed.

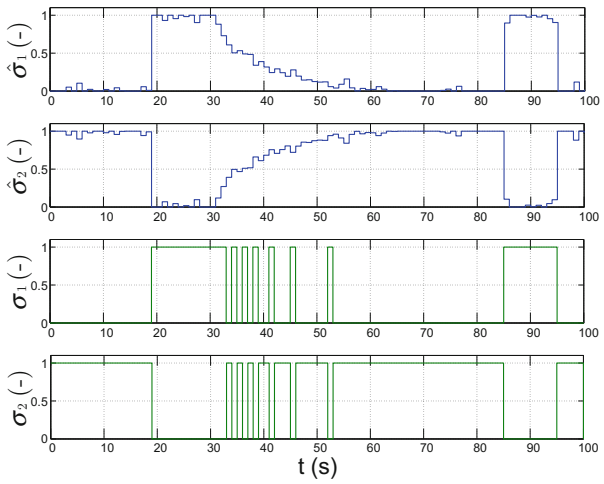


Fig. 8.7 Illustration of the sum-up rounding for a system with $N_q = 2$ over the time interval $t \in [0, 100]$

Initially, both tuples are empty and the NLP is solved. The tuples are then extended in two steps:

- **first step**: is a rounding step that rounds all $\overline{\hat{\boldsymbol{\sigma}}}_q^{[k]} > 1 - \varepsilon$ for $0 < \varepsilon \ll 1$ to 1 and all $\overline{\hat{\boldsymbol{\sigma}}}_q^{[k]} < \varepsilon$ to 0. The rounded values are then imposed as additional constraints; and
- **the second step**: the index pair $(q, k) \notin \mathcal{J}$ is found, for which the distance $|0.5 - \overline{\hat{\boldsymbol{\sigma}}}_q^{[k]}|$ is the largest. For this index pair, $\overline{\hat{\boldsymbol{\sigma}}}_q^{[k]}$ is also rounded to the nearest value: zero or one.

The second step can be repeated N_r times. Choosing N_r higher will speed up the algorithm but may yield inferior results. The same is valid for the value of ε. With the newly defined constraints, the NLP is then solved again. The algorithm stops, if all $\overline{\hat{\boldsymbol{\sigma}}}_q^{[k]}$ are constrained. Then, $\overline{\hat{\boldsymbol{\sigma}}}$ is a binary feasible solution $\overline{\boldsymbol{\sigma}}$.

8.5 Discussion

Indirect methods usually find highly accurate solutions by attempting to solve the first-order necessary conditions for hybrid optimal control problems. Thus, for an indirect method, it is necessary to derive the canonical equations, the Hamiltonian minimum condition, and all of the transversality conditions. This can be a difficult task even for experienced users with fundamental backgrounds on optimal control theory. This is even more difficult if these expressions must be obtained from complicated black-box applications as they appear often in (automotive) practice.

Many of the issues of indirect methods are solved with direct methods. Direct methods do not require the derivation the first-order necessary conditions, which is often used as argumentation that direct methods are more appealing to users. Nevertheless, the sparse NLP solvers introduce many additional degree-of-freedoms, e.g., the choice of the transcription method, the integration scheme, and the sparse NLP configuration, which need to be selected appropriately to the formulated problem by the user.

One can state the following advantages of direct methods:

- larger domain of convergence;
- easily incorporation of state constraints $\mathbf{c}_x(\cdot)$ into the NLP formulations with the direct collocation transcription; and
- direct methods combined with numeric or automatic differential schemes have the benefit to be flexible adapted to new problem instances with less effort

and disadvantages:

- solutions not as accurate as obtained from indirect methods;
- large optimal control problems need NLP solvers with sparse matrix exploration;
- costate trajectories can not be obtained directly; and

- incorporation of state jumps $\boldsymbol{\delta}_{(q(t_j^-),q(t_j^+))}(\cdot)$ lead in both direct transcriptions to a nonsmooth optimization problem even if the state jumps are assumed to occur only at multiples of the mesh grid points $\mathcal{G}_t$.

To overcome some disadvantages of the direct and indirect methods, one can combine both methodologies. A common fusion approach is *homotopy*. Homotopy in this context is the idea of embedding a complex problem into simpler subproblems. Such methods are also referred to in the literature as continuation, deformation, or embedding methods. The idea can be applied to indirect methods for providing improved initial guesses for the costates (Bulirsch et al. [16]). A direct collocation method is first applied to a simplified OCP where all state constraints are neglected as long as the resulting problem is well-defined. The costates can then be recovered as described in Sect. 8.2.4. This methodology preserves the high accuracy of indirect methods while disadvantages caused by ill-conditioning are considerably reduced.

An important question is, whether the discrete approximation of a SOCP converges to the continuous formulation for $h_k \to 0,\ k = 0, \ldots, N_t$. The necessary additional continuity requirement made by Mordukhovich [48] and Gerdts [29] in their convergence analysis can be problematic, when the discrete controls are relaxed to continuous-valued controls as in the embedding approach. In this case a bang–bang solution is expected and therefore the controls will not be continuous but continuous-valued. A convergence rate for the approximation of a discrete initial value problem for hybrid systems with state jumps is derived in the works of Tavernini [68, 69]. However, this convergence rate is only valid, if a variable time grid is used as proposed by the author.

8.6 Bibliography

The application of direct transcription methods for the solution of continuous optimal control problems is well described in Betts [11], Bock and Plitt [12], and von Stryk [65]. In the work of von Stryk [65] different discretization schemes are introduced: the piecewise constant control with piecewise linear state discretization and the piecewise linear control with piecewise cubic state discretization. In recent years pseudospectral methods have increased in popularity. In pseudospectral methods the state is approximated using fixed collocation points and global polynomials which can be increased in the order-of-degree to achieve a high accuracy. This method has been developed originally to solve partial differential equations within computational fluid dynamics problems. In their purest form, these methods do not subdivide the time horizon into time intervals but represent states as single high-order polynomials. A detailed convergence analysis for the discrete approximation of continuous optimal control problems has been addressed by Mordukhovich [47, 48], and Gerdts [29].

A good overview paper of MINLP methods like branch-and-bound, outer-approximation, generalized Benders decomposition, and cutting planes is given by Grossmann [30]; for non-convex MINLP a survey is given by Burer and Letchford [17]. A good historical view about BB methods is presented by Cook [21]. The BB algorithm was developed in the sixties for integer programming (Land and Doig

[37]) but the name was originally coined by Little et al. [44]. Later on, it was also applied to global optimization. The outer-approximation method has been first developed by Duran and Grossmann [22] and then continued by many authors. Among them, Fletcher and Leyffer [25]. Fletcher and Leyffer [25] proposed a quadratic outer approximation. The generalized Benders decomposition, proposed by Geoffrion [26], is similar to the outer-approximation method. Quesada and Grossmann [50] proposed an algorithmic scheme that combines the use of *linear programming* (LP) and NLP in an original branch-and-cut scheme. Bonami et al. [14] proposed a hybrid algorithm that uses LP/NLP branch-and-bound and outer approximation. An algorithm for MINLPs with a constraint on the maximum number of switchings is proposed in Sager et al. [57]. An efficient decomposition algorithm for BB is proposed by von Stryk and Glocker [67]. Leyffer [41] showed that branching is allowed after each iteration of the NLP solver which speeds up the optimization. Gerdts and Sager [27] generalized necessary conditions and bounds from mixed-integer optimal control problems based on ODEs to differential algebraic equations.

Xu and Antsaklis [75] described a general *two-stage* procedure. In the first stage, the continuous-valued controls $\mathbf{u}(\cdot)$ are optimized. The second step involves optimizing the switching times t_j and varying the switching-order and the total number of switchings. No specific procedure for the variation of the switching sequence is mentioned. In a different work of the same authors (Xu and Antsaklis [74]), a derivative of the value function with respect to the switching time is derived and a gradient descent approach is employed to alter the switching times. The HMP[MCS]-algorithm proposed by Shaikh [63] is based on an initial guess of the switching instances t_j and the continuous-valued states at the switching times $\mathbf{x}(t_j)$. For each segment $[t_j, t_{j+1})$, a continuous OCP is solved and the trajectories $\mathbf{x}(\cdot)$, $\boldsymbol{\lambda}(\cdot)$ are assembled for the entire time span $[t_0, t_f]$. Based on the differences in the costates and the Hamiltonian of the assembled trajectories at the switching time instant, i.e., $\boldsymbol{\lambda}(t_j^+) - \boldsymbol{\lambda}(t_j^-)$ and $\mathcal{H}(t_j^+) - \mathcal{H}(t_j^-)$, the states $\mathbf{x}(t_j)$ and the respective times are varied until the absolute values of the differences fulfill a lower bound condition. In the work of Alamir and Attia [1], a first initial guess of the discretized controls $\overline{\mathbf{u}}$ and discrete states $\overline{\mathbf{q}}$ is made and the corresponding discretized state trajectory $\overline{\mathbf{x}}$ and costate trajectory $\overline{\boldsymbol{\lambda}}$ are calculated, such that the Hamiltonian function value can be computed for any time instant. In the next step, optimized control inputs $\overline{\mathbf{u}}^*$ and $\overline{\mathbf{q}}^*$ are computed, such that the Hamiltonian function is minimized for each time instant. A penalty factor added to the Hamiltonian function stabilizes the algorithm by reducing successive variations of the system inputs over the iterations. The approach proposed by Nüesch et al. [49] alternates between solving a NLP problem for finding the continuous-valued controls and solving a dynamic programming problem for obtaining a switching sequence. In the dynamic programming problem, the cost function is composed of the sum of the Hamiltonians at all time instants and a term for the switching cost. The costates $\boldsymbol{\lambda}(\cdot)$ are assumed to be constant for the specific case regarded. In Xu and Antsaklis [77], Egerstedt et al. [23], Kamgarpour and Tomlin [35], and Axelsson et al. [2, 3] methods are described for finding the optimal switching times of a switched system with continuous states and with predefined mode sequence by tran-

scribing the switched system into a NLP using the partial derivatives of the cost function and the first-order necessary conditions for optimality with respect to the switching times. Egerstedt et al. [23] and Axelsson et al. [3] start with an assumed switching sequence and then optimized the length of each interval $[t_j, t_{j+1})$ without modifying the mode sequence. In the next step, the switching mode sequence is modified by inserting a mode, where a specifically derived derivative of the cost function with respect to the insertion of a mode does not vanish. The results were extended to switched systems with discontinuities on a switching in Schori et al. [62]. For similar problems, a numerically fast algorithm based on the differential transformation is demonstrated in Hwang et al. [33]. The two-point boundary value problem is converted to a problem of solving a system of algebraic equations. Another two-stage approach is to cast the SOCP into a MINLP and to solve it with BB methods. When SOCPs are converted to MINLPs, the number of discrete variables is in general very high and hence, this type of approach will be unfeasible for most cases. Many efforts have been made on a reduction of the search tree's size. Sager [54] gives a wide overview of methods for solving MINLPs and their application to HOCPs.

Bengea and DeCarlo [5–7] applied the convex embedding idea to switched optimal control problems and showed that the binary feasible solutions can be approximated arbitrarily close. Singular arcs in the context of embedded optimal control has been considered by Riedinger et al. [52]. In Boehme et al. [13] has been shown that a suboptimal solution can be found with a heuristic rounding strategy. For hybrid systems with state jumps, the dynamic programming algorithm can be a valid approach, if the number of continuous-valued states $\mathbf{x}(\cdot)$ is rather low. For systems with a high number of continuous-valued states, a direct method with embedding can yield a suboptimal solution that is usually close to a fully optimal solution as shown in Schori et al. [62]. In the context of autonomous systems, HOCPs with state jumps have been treated in Xu [73] and Attia [53]. Related cases are the HOCPs where no jumps in the state occur but additional costs are added to the objective function for every switching. Hybrid dynamic programming is widely applied to this type of problem as in Hedlund and Rantzer [32]. A method for problems that have an upper bound on the number of switchings is proposed in the work of Sager et al. [57]. An embedded optimal control problem is solved using a NLP solver. To find a switching schedule that satisfies the upper bound on the number of switchings, a mixed-integer linear programing problem is solved that minimizes a distance to the relaxed switching schedule obtained from the solution of the embedded problem but fulfills the integer conditions as well as the condition imposed on the number of switchings.

Nondifferentiable optimization, also known as nonsmooth optimization, has received considerable attention in the past decades (e.g., Bertsekas [8], Clarke [20]). Several textbooks are available on this subject, among them Balinski et al. [4], Bonnans et al. [15], and Shor [64]. Nondifferentiable optimization problems benefits from adapted methods such as generalized Newton-type methods for nonsmooth equations presented in Butikofer [19]. Even though no proof of convergence exists, it has been shown in many cases that quasi-Newton methods perform well on nonsmooth problems (Lewis and Overton [39] and Lukšan and Vlček [45]). Automatic differentiation (Rall [51]) in combination with a black-box method, that heuris-

tically defines the gradient from the set of sub-gradients where a function is not continuously differentiable, can improve convergence (Lemaréchal [38]). For the case of state jumps, Schori et al. [61] showed that using a convexified jump function $\boldsymbol{\delta}_{(q(t_j^-),q(t_j^+))}(\cdot)$ and sub-gradients can make discontinuous optimal control problems amenable to standard nonlinear programming solvers. With the convexified jump function the embedding approach was directly applied and a discontinuous embedded optimal control problem has been obtained.

Post-optimal recovering procedures for the costates were proposed by Enright and Conway [24], von Stryk [65] among others. Büskens [18] investigated the jumps in the costates and proposed algorithms to compute these separately.

References

1. Alamir M, Attia S (2004) On solving optimal control problems for switched hybrid nonlinear systems by strong variations algorithms. In: 6th IFAC symposium on nonlinear control systems (NOLCOS), Stuttgart, Germany, pp 558–563
2. Axelsson H, Wardi Y, Egerstedt M (2005) Transition-time optimization for switched systems. In: Proceedings of IFAC world congress
3. Axelsson H, Wardi Y, Egerstedt M, Verriest E (2008) Gradient descent approach to optimal mode scheduling in hybrid dynamical systems. J Optim Theory Appl 136. doi:10.1007/s10957-007-9305-y
4. Balinski ML, Wolfe P, Bertsekas DP, Camerini P, Cullum J (1975) Nondifferentiable optimization. North-Holland, Amsterdam
5. Bengea S, DeCarlo R (2003) Optimal and suboptimal control of switching systems. In: Proceedings of the 42nd IEEE conference on decision and control, pp 5295–5300
6. Bengea S, Uthaichana K, Žefran M, DeCarlo RA (2011) The control system handbook optimal control of switching systems via embedding into continuous optimal control problems, 2nd edn. CRC Press
7. Bengea SC, DeCarlo RA (2005) Optimal control of switching systems. Automatica 41(1):11–27
8. Bertsekas DP (1975) Nondifferentiable optimization via approximation. In: Nondifferentiable optimization. Springer, pp 1–25
9. Betts J, Huffman W (1999) Exploiting sparsity in the direct transcription method for optimal control. Comput Optim Appl 14(2):179–201
10. Betts JT (1998) Survey of numerical methods for trajectory optimization. J Guid Control Dyn 21(2):193–207
11. Betts JT (2010) Practical methods for optimal control and estimation using nonlinear programming. Society for industrial and applied mathematics, 2nd edn. doi:10.1137/1.9780898718577
12. Bock H, Plitt K (1984) A multiple shooting algorithm for direct solution of optimal control problems. In: Proceedings of the 9th world congress of the international federation of automatic control, vol 9
13. Boehme TJ, Frank B, Schultalbers M, Schori M, Lampe B (2013) Solutions of hybrid energy-optimal control for model-based calibrations of HEV powertrains. In: SAE world congress, technical Paper 2013-01-1747. doi:10.4271/2013-01-1747
14. Bonami P, Biegler LT, Conn AR, Cornuéjols G, Grossmann IE, Laird CD, Lee J, Lodi A, Margot F, Sawaya N et al (2008) An algorithmic framework for convex mixed integer nonlinear programs. Discret Optim 5(2):186–204
15. Bonnans JF, Gilbert JC, Lemaréchal C, Sagastizábal CA (2006) Numerical optimization: theoretical and practical aspects. Springer Science & Business Media

16. Bulirsch R, Nerz E, Pesch H, von Stryk O (1993) Combining direct and indirect methods in optimal control: Range maximization of a hang glider. In: Bulirsch R, Miele A, Stoer J, Well K (eds) Optimal control, ISNM international series of numerical mathematics, vol 111. Birkhäuser, Basel, pp 273–288. doi:10.1007/978-3-0348-7539-4_20
17. Burer S, Letchford AN (2012) Non-convex mixed-integer nonlinear programming: a survey. Surv Oper Res Manag Sci 17(2):97–106
18. Büskens C (1998) Optimierungsmethoden und Sensitivitätsanalyse für optimale Steuerprozesse mit Steuer- und Zustandsbeschränkungen. PhD thesis, Universität Münster
19. Butikofer S (2008) Generalized Newton-type methods for nonsmooth equations in optimization and complementarity problems. PhD thesis, ETH Zürich
20. Clarke FH (1978) Nonsmooth analysis and optimization. In: Proceedings of the international congress of mathematicians (Helsinki), pp 847–853
21. Cook W (2012) Markowitz and Manne + Eastman + Land and Doig = branch and bound. Optim Storit 227–238
22. Duran MA, Grossmann IE (1986) An outer-approximation algorithm for a class of mixed-integer nonlinear programs. Math Program 36(3):307–339
23. Egerstedt M, Wardi Y, Axelsson H (2006) Transition-time optimization for switched-mode dynamical systems. IEEE Trans Autom Control 51. doi:10.1109/TAC.2005.861711
24. Enright P, Conway B (1992) Discrete approximations to optimal trajectories using direct transcription and nonlinear programming. J Guid Control Dyn 15(4):994–1002
25. Fletcher R, Leyffer S (1994) Solving mixed integer nonlinear programs by outer approximation. Math Program 66(1–3):327–349
26. Geoffrion AM (1972) Generalized Benders decomposition. J Optim Theory Appl 10(4):237–260
27. Gerdts S (2012) Mixed-integer DAE optimal control problems: necessary conditions and bounds. In: Campbell S, Mehrmann V, Biegler L (eds) Control and optimization with differential-algebraic constraints. SIAM, pp 189–212
28. Gerdts M (2006) A variable time transformation method for mixed-integer optimal control problems. Optim Control Appl Methods 27(3):169–182
29. Gerdts M (2012) Optimal control of ordinary differential equations and differential-algebraic equations. de Gruyter, Berlin
30. Grossmann IE (2002) Review of nonlinear mixed-integer and disjunctive programming techniques. Optim Eng 3(3):227–252
31. Grossmann IE, Kravanja Z (1995) Mixed-integer nonlinear programming techniques for process systems engineering. Comput Chem Eng 19:189–204
32. Hedlund S, Rantzer A (2002) Convex dynamic programming for hybrid systems. IEEE Trans Autom Control 47:1536–1540
33. Hwang I, Li J, Du D (2008) A numerical algorithm for optimal control of a class of hybrid systems: differential transformation based approach. Int J Control 81:277–293
34. Grossmann IE, ZK, (1993) Mixed-integer nonlinear programming: a survey of algorithms and applications. IMA Vol Math Appl 93:73–100
35. Kamgarpour M, Tomlin C (2012) On optimal control of non-autonomous switched systems with a fixed mode sequence. Automatica 48(6):1177–1181
36. Kirches C (2011) Fast numerical methods for mixed-integer nonlinear model-predictive control. Springer
37. Land AH, Doig AG (1960) An automatic method of solving discrete programming problems. Econ: J Econ Soc 497–520
38. Lemaréchal C (1989) Nondifferentiable optimization. In: Handbooks in operations research and management science. Elsevier Science Publishers B.V., North-Holland
39. Lewis A, Overton M (2012) Nonsmooth optimization via Quasi-Newton methods. Math Program 141:135–163
40. Leyffer S (1993) Deterministic methods for mixed integer nonlinear programming. PhD thesis, University of Dundee

41. Leyffer S (2001) Integrating SQP and branch-and-bound for mixed integer nonlinear programming. Comput Optim Appl 18(3):295–309
42. Leyffer S (2006) Complementarity constraints as nonlinear equations: theory and numerical experience. In: Optimization with multivalued mappings. Springer, pp 169–208
43. Linderoth JT, Savelsbergh MW (1999) A computational study of search strategies for mixed integer programming. INFORMS J Comput 11(2):173–187
44. Little JD, Murty KG, Sweeney DW, Karel C (1963) An algorithm for the traveling salesman problem. Oper Res 11(6):972–989
45. Lukšan L, Vlček J (1999) Globally convergent variable metric method for convex and nonsmooth unconstrained minimization. J Optim Theory Appl 102:593–613
46. Maurer H, Büskens C, Kim JH, Kaya C (2005) Optimization methods for the verification of second order sufficient conditions for bang-bang controls. Optim Control Appl Methods 26(3):129–156
47. Mordukhovich B (1978) On difference approximations of optimal control systems. J Appl Math Mech 42(3):452–461
48. Mordukhovich B (2006) Variational analysis and generalized differentiation II, applications. Grundlehren der mathematischen Wissenschaften. Springer, Berlin
49. Nüesch T, Elbert P, Flankl M, Onder C, Guzzella L (2014) Convex optimization for the energy management of hybrid electric vehicles considering engine start and gearshift costs. Energies 7:834–856
50. Quesada I, Grossmann IE (1992) An LP/NLP based branched and bound algorithm for convex MINLP optimization problems. Comput Chem Eng 16:937–947
51. Rall L (1981) Automatic differentiation: techniques and applications, vol 120. Lecture notes in computer science. Springer, Berlin
52. Riedinger P, Daafouz J, Iung C (2003) Suboptimal switched controls in context of singular arcs. In: Proceedings of the 42nd IEEE conference on decision and control, vol 6. IEEE, pp 6254–6259
53. SA Attia JR V Azhmyakov (2007) State jump optimization for a class of hybrid autonomous systems. In: Proceedings of the 16th IEEE international conference on control applications, pp 1408–1413
54. Sager S (2005) Numerical methods for mixed-integer optimal control problems. PhD thesis, Universität Heidelberg
55. Sager S (2009) Reformulations and algorithms for the optimization of switching decisions in nonlinear optimal control. J Process Control 19(8):1238–1247
56. Sager S, Bock HG, Diehl M (2007) Solving mixed–integer control problems by sum up rounding with guaranteed integer gap. Preprint, IWR, University of Heidelberg. http://www.ub.uni-heidelberg.de/archiv/8384
57. Sager S, Jung M, Kirches C (2011) Combinatorial integral approximation. Math Methods Oper Res 73(3):363–380
58. Sager S, Bock H, Diehl M (2012) The integer approximation error in mixed-integer optimal control. Math Program 133(1–2):1–23
59. Schäfer R (2014) Gemischt-ganzzahlige Optimalsteuerung, Sensitivitätsanalyse und Echtzeitoptimierung von Parallel-Hybridfahrzeugen. Master's thesis, Universität Bremen
60. Schori M (2015) Solution of optimal control problems for switched systems. Algorithms and applications for hybrid vehicles. PhD thesis, Universität Rostock
61. Schori M, Boehme TJ, Frank B, Schultalbers M (2014) Control optimization of discontinuous hybrid systems using embedding. Discret Event Syst 12:326–331
62. Schori M, Boehme TJ, Jeinsch T, Lampe B (2015) Switching time optimization for discontinuous switched systems. Proceedings of the 2015 European control conference (ECC), July 15–17. Linz. IEEE, pp 1742–1747
63. Shaikh MS (2004) Optimal control of hybrid systems: theory and algorithms. PhD thesis, Department of Electrical and Computer Engineering, McGill University, Montreal
64. Shor NZ (1985) Minimization methods for non-differentiable functions. Springer Ser Comput Math 3. doi:10.1007/978-3-642-82118-9

65. von Stryk O (1995) Numerische Lösung optimaler Steuerungsprobleme: Diskretisierung, Parameteroptimierung und Berechnung der adjungierten Variablen. Fortschritt-Berichte VDI-Verlag 8
66. von Stryk O, Bulirsch R (1992) Direct and indirect methods for trajectory optimization. Ann Oper Res 37:357–373
67. von Stryk O, Glocker M (2000) Decomposition of mixed-integer optimal control problems using branch and bound and sparse direct collocation. In: The 4th international conference on automation of mixed processes: hybrid dynamic systems, pp 99–104
68. Tavernini L (1987) Differential automata and their discrete simulators. Nonlinear Anal Theory Methods Appl 11:583–665
69. Tavernini L (2009) Generic asymptotic error estimates for the numerical simulation of hybrid systems. Nonlinear Anal: Hybrid Syst 3(2):108–123
70. Till J, Engell S, Panek S, Stursberg O (2004) Applied hybrid system optimization: an empirical investigation of complexity. Control Eng Pract 12(10):1291–1303
71. Tsang T, Himmelblau D, Edgar T (1975) Optimal control via collocation and non-linear programming. Int J Control 21(5):763–768
72. Turau V (2009) Algorithmische Graphentheorie. Oldenbourg Verlag
73. Xu X, Antsaklis PJ (2003) Optimal control of hybrid autonomous systems with state jumps. In: Proceedings of the American control conference. IEEE, pp 5191–5196
74. Xu X, Antsaklis PJ (2000) A dynamic programming approach for optimal control of switched systems. In: Proceedings of the 39th IEEE conference on decision and control, vol 2. IEEE, pp 1822–1827
75. Xu X, Antsaklis PJ (2000) Optimal control of switched systems: new results and open problems. In: Proceedings of the American control conference, vol 4. IEEE, pp 2683–2687
76. Xu X, Antsaklis PJ (2001) Switched systems optimal control formulation and a two stage optimization methodology. In: Proceedings of the 9th Mediterranean conference on control and automation
77. Xu X, Antsaklis PJ (2004) Optimal control of switched systems based on parameterization of the switching instants. IEEE Trans Autom Control 49. doi:10.1109/TAC.2003.821417

Part III
Numerical Implementations

Chapter 9
Practical Implementation Aspects of Large-Scale Optimal Control Solvers

9.1 Sparse Linear Algebra

The key feature of modern nonlinear programming solvers for direct minimization of optimal control problems, is a sparse linear algebra kernel. The full discretization of optimal control problems generates many optimization variables and constraints, such that the linear systems, arising in QP-subproblems of *sequential quadratic programming* (SQP) methods are very high dimensional and sparse.

Therefore, we give a short introduction into sparse matrix representations and the special features of sparse linear algebra.

9.1.1 Sparse Matrix Formats

As described in many textbooks about sparse matrix computations such as Tewarson [39], Duff et al. [8], Saad [36], Davis [7], and Pissanetzky [33] there are many ways to efficiently store sparse matrices.

In the following, we use some fundamental graph theoretical notations and concepts, which are described in the Appendix A. If the reader is not familiar with these prerequisites, we recommend to read Appendix A prior to this chapter.

Instead of storing a two dimensional array of size $N_n \times N_m$, which is a common matrix representation for dense matrices, a sparse unsymmetric matrix (represented as a biadjacency matrix) can be stored by three vectors of length N_{nz}: two vectors for the row and column indices of the matrix entries and one vector for the values of the matrix at the corresponding position defined by the row and column vector. Here, N_{nz} denotes the number of nonzero entries of the matrix. Additionally, the row- and column dimension must be stored as two integer values, if the matrix has empty rows or columns.

T.J. Böhme and B. Frank, *Hybrid Systems, Optimal Control and Hybrid Vehicles*,
Advances in Industrial Control, DOI 10.1007/978-3-319-51317-1_9

A sparse symmetric matrix (represented as an adjacency matrix with additional diagonal elements) can also be stored by three vectors and two integer values. The advantage of the interpretation as an adjacency matrix is that only half of the off-diagonal entries must be stored.

The factor $d = N_{nz}/N_m$ is called *density* of the matrix. Obviously, the lower d, the more efficient is the memory storage as a sparse matrix.

Despite the low memory consumption of this sparse storage format it is a great advantage, that many matrix and matrix-vector operations can be executed very efficiently. For example, the transpose of a sparse matrix can be calculated by simply interchanging the row and column index vectors and for a matrix-vector multiplication only N_{nz} multiplications and additions must be calculated.

Even more efficient is the storage of a compressed row index vector of length N_n instead of the full row index vector of length N_{nz}. This can be achieved if in the row column index vector only the number of nonzero entries for each row is stored. This compression is known as *row compressed storage* (RCS) format. Compared with the uncompressed storage format it has the little drawback that direct matrix operations, e.g., the calculation of the transpose of the matrix, are more complex, but matrix-vector operations, e.g., the multiplication of a transpose matrix with a vector, can be executed equally efficient.

Analogous to the RCS-Format the *column compressed storage* format can be defined, for which the number of nonzero matrix entries in each column is stored in the column index vector.

9.1.2 Numerical Solution of Large-Scale Linear Systems

Since all system matrices, i.e., the KKT matrices in the QP-subproblems (cf. Section 2.3.3.1), appearing in the SQP framework for optimal control are large-scale and symmetric, we give a brief introduction to the sparse LDL^T-decomposition, which is supposedly the most efficient method to solve this type of problem. This method is described in detail in Duff et al. [8] and Hogg et al. [21].

The LDL^T-decomposition applied to a sparse symmetric matrix $\mathbf{A}$ generates a lower triangular matrix $\mathbf{L}$ and a block diagonal matrix $\mathbf{D}$, such that $\mathbf{A} = \mathbf{L}\mathbf{D}\mathbf{L}^T$ results. In case of a positive definite matrix $\mathbf{A}$ the matrix $\mathbf{D}$ is diagonal with only positive entries. Otherwise, for an indefinite matrix $\mathbf{A}$ the matrix $\mathbf{D}$ is block diagonal and contains 1×1- and 2×2-blocks. Due to pivoting strategies, which define an elimination ordering $\boldsymbol{\pi}$ of the columns and rows during the decomposition, a decomposition of the form $\mathbf{P}\mathbf{A}\mathbf{P}^T = \mathbf{L}\mathbf{D}\mathbf{L}^T$ is generally performed. Here, $\boldsymbol{\pi}$ is a permutation of the enumeration $1, ..., N_n$ and $\mathbf{P}$ is an $N_n \times N_n$ matrix with entries $\mathbf{P}_{[i],[j]} = 1$, if $\boldsymbol{\pi}_{[i]} = j$. The basic procedure of such a decomposition, which defines $\boldsymbol{\pi}$ at runtime, is shown in Algorithm 9.1.

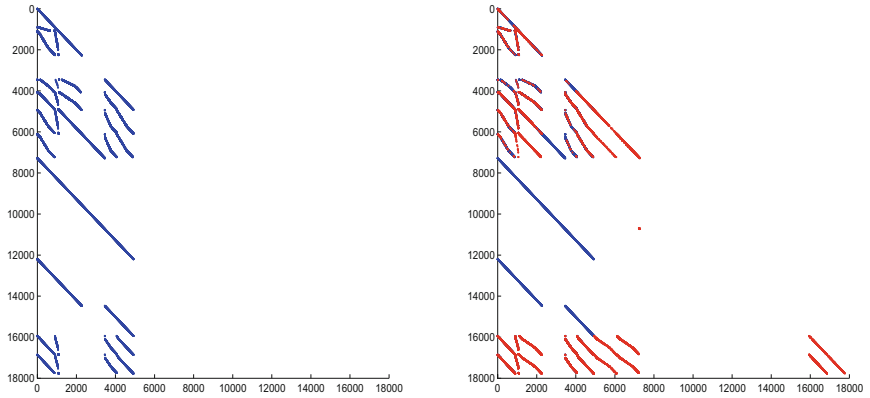


Fig. 9.1 Example of fill-in: The *blue dots* in the *left figure* mark the off-diagonal elements of the lower triangular part of a KKT matrix. The additional elements introduced by a LDL^T-decomposition are marked as *red dots* in the *right figure*. The total fill-in is in this case 19903

Algorithm 9.1 LDL^T-decomposition

Set $j \leftarrow 1$ and choose a small $\epsilon > 0$
while $j \leq N_n$ **do**
  $\boldsymbol{\pi}_{[j]} \leftarrow j$
  if $|\mathbf{A}_{[j],[j]}| > \epsilon$ **then**
    Choose the 1×1 pivot $\mathbf{D}_{[j],[j]} \leftarrow \mathbf{A}_{[j],[j]}$.
    Set $s \leftarrow 1$.
  else
    Find k such that $|\mathbf{A}_{[k],[j]}| > \epsilon$.
    Symmetrically permute row/column k to position $j+1$ and set $\boldsymbol{\pi}_{[j+1]} \leftarrow k$.
    Choose the 2×2 pivot $\mathbf{D}_{[j:j+1],[j:j+1]} \leftarrow \mathbf{A}_{[j:j+1],[j:j+1]}$.
    Set $s \leftarrow 2$.
  end if
  $\mathbf{L}_{[j:N_n],[j:j+s-1]} \leftarrow \mathbf{A}_{[j:N_n],[j:j+s-1]}\mathbf{D}^{-1}_{[j:j+s-1],[j:j+s-1]}$
  $\mathbf{A}_{[j:N_n],[j:N_n]} \leftarrow \mathbf{A}_{[j:N_n],[j:N_n]} - \mathbf{L}_{[j:N_n],[j:j+s-1]}\mathbf{D}_{[j:j+s-1],[j:j+s-1]}\mathbf{L}^T_{[j:j+s-1],[j:N_n]}$.
  $j \leftarrow j + s$.
end while

Very important for a decomposition of a large sparse symmetric matrix is a manageable memory consumption, because the sparsity pattern of the lower triangular part of matrix $\mathbf{A}$ will in most cases not be the same as the sparsity pattern of the L-factor (Fig. 9.1). In general the number of elements in the matrix L is at least the number of elements in the lower triangular part of the matrix A and in the worst case the matrix L can become dense.

Thus, the *fill-in* $F(\mathbf{A}, \boldsymbol{\pi})$ of the factors, which is the number of additional elements introduced by the LDL^T-decomposition, has to be minimized, such that the memory consumption is minimized too. This number strongly depends on the edges of the

graph defined by the adjacency matrix $\mathbf{A}$ and on the elimination ordering $\boldsymbol{\pi}$. The j-th step of the decomposition first introduces new edges, such that v_j together with its neighbors becomes a clique, and afterwards eliminates the vertex v_j from the graph. Each added edge represents one additional element in the factors. Algorithm 9.2 implements the determination of the additional elements in terms of a new set of edges $\mathcal{B}_{fill}$ for a given elimination ordering $\boldsymbol{\pi}$ and a given adjacency matrix $\mathbf{A}$ with the set of edges $\mathcal{B}$ as described by Tarjan and Yannakakis [38].

Algorithm 9.2 Fill-in Calculation (cf. Tarjan and Yannakakis [38])

```
Define the inverse elimination ordering π^{-1} as integer array.
Define the temporary integer arrays α and β.
Define the temporary integer variables v, w, and x.
B_fill ← B
for i ← 1 to N_n do
    α_[i] ← 0
    β_[i] ← 0
    π^{-1}_[π_[i]] ← i
end for
for i ← 1 to N_n do
    w ← π_[i]
    α_[w] ← w
    β_[w] ← i
    for all (v, w) ∈ B do
        if π^{-1}_[v] < i then
            x ← v
            while β_[x] < i do
                β_[x] ← i
                if x ≠ α_[x] and (α_[x], w) ∉ B then
                    x ← α_[x]
                    Add (x, w) to B_fill
                end if
            end while
            if x = α_[x] then
                α_[x] ← w
            end if
        end if
    end for
end for
```

The fill-in can afterwards be calculated by $F = |\mathcal{B}_{fill}| - |\mathcal{B}|$.

Algorithm 9.2 conveys the idea of a fill-in reducing LDL^T-decomposition. One can first try to find a fill-in reducing elimination ordering $\boldsymbol{\pi}_f$ and a permutation matrix $\mathbf{P}_f$. Then, Algorithm 9.1 can be applied to the matrix $\mathbf{P}_f\mathbf{A}\mathbf{P}_f^T$ and the elimination ordering is corrected only if numerical problems occur. These alterations of the elimination order can in turn lead to a higher fill-in than expected. Therefore, the treatment of small pivot elements is one of the main problems in the design of a direct linear equations solver.

Obviously, the best possible elimination ordering is one that does not introduce any fill-in, i.e., $F(\mathbf{A}, \boldsymbol{\pi}_f) = 0$. Such an ordering is called *perfect elimination ordering* (PEO). Unfortunately, a PEO does not exist for any symmetric sparse matrix, but it can be shown that a PEO exists, if and only if the graph of an adjacency matrix is chordal. An algorithm which calculates a PEO without any fill-in for an adjacency matrix with a chordal graph $G(\mathcal{V}, \mathcal{B})$ is the *maximum cardinality search*.

Algorithm 9.3 Maximum Cardinality Search (cf. Tarjan and Yannakakis [38])

Define the elimination ordering $\boldsymbol{\pi}$ and the inverse elimination ordering $\boldsymbol{\pi}^{-1}$ as integer arrays.
Define the temporary set array $\boldsymbol{\Lambda}$.
Define the temporary integer array $\boldsymbol{\alpha}$.
Define the temporary integer variables i, j, v and w.
for $i \leftarrow 1$ **to** N_n **do**
    $\boldsymbol{\Lambda}_{[i]} \leftarrow \emptyset$
    $\boldsymbol{\alpha}_{[i]} \leftarrow 0$
    $\boldsymbol{\pi}_{[i]} \leftarrow 0$
    $\boldsymbol{\pi}^{-1}_{[i]} \leftarrow 0$
end for
$\boldsymbol{\Lambda}_{[1]} = \{1, ..., N_n\}$
$i \leftarrow n$
$j \leftarrow 0$
while $i > 0$ **do**
    Let v be the first element of $\boldsymbol{\Lambda}_{[j+1]}$.
    Remove v from $\boldsymbol{\Lambda}_{[j+1]}$.
    $\boldsymbol{\pi}_{[v]} \leftarrow i$
    $\boldsymbol{\pi}^{-1}_{[i]} \leftarrow v$
    $\boldsymbol{\alpha}_{[v]} \leftarrow -1$
    for all $(v, w) \in \mathcal{B}$ **do**
        if $\boldsymbol{\alpha}_{[w]} \geq 0$ **then**
            Delete w from $\boldsymbol{\Lambda}_{[\boldsymbol{\alpha}_{[w]}+1]}$.
            $\boldsymbol{\alpha}_{[w]} \leftarrow \boldsymbol{\alpha}_{[w]} + 1$
            Add w to $\boldsymbol{\Lambda}_{[\boldsymbol{\alpha}_{[w]}+1]}$.
        end if
    end for
    $i \leftarrow i - 1$
    $j \leftarrow j + 1$
    while $j \geq 0$ **and** $\boldsymbol{\Lambda}_{[j+1]} = \emptyset$ **do**
        $j \leftarrow j - 1$
    end while
end while

In Yannakakis [43] it is shown that the determination of an elimination ordering with minimum fill-in for a non-chordal graph and the determination of a chordal extension of an undirected graph with the minimum number of additional edges are equivalent. Unfortunately, it is also shown by Yannakakis [43] that both problems are NP-complete, which means that a solution cannot be computed with a polynomial time effort. Even though, there are some fill-in reducing strategies, which can be computed fast and perform very well in practice. A good algorithm is the *min-*

imum degree algorithm, which was first introduced by Markowitz [28] and further developed and studied among others in Tinney and Walker [40], Rose [35], Liu [27], George and Liu [13], and Heggernes et al. [20]. Another very good algorithm is the *nested dissection algorithm* from George [11], which was further developed and studied among others in Lipton et al. [26], Gilbert and Tarjan [15], Gilbert [14], and Karypis and Kumar [24].

9.1.3 Checking the Positive Definiteness of Large-Scale Matrices

For a fast convergence of the SQP-algorithm and the application of sensitivity differentials, it is essential that the reduced Hessian is positive definite, i.e., that it has only positive eigenvalues. If we deal with very high-dimensional problems and sparse matrices, the reduced Hessian and its eigenvalues cannot be directly calculated in an efficient manner. Therefore, we introduce another approach that is based on the inertia of a matrix.

The inertia of a quadratic matrix $\mathbf{H}$ is a triple of nonnegative integers

$$In(\mathbf{H}) = (l_+(\mathbf{H}), l_-(\mathbf{H}), l_0(\mathbf{H}))$$

where l_+, l_-, and l_0 is the number of positive, negative, and zero eigenvalues of $\mathbf{H}$, respectively.

Theorem 9.1 (Theorem of Sylvester) *Let* $\mathbf{S}$ *and* $\mathbf{H}$ *be real-valued quadratic matrices. If* $\mathbf{H}$ *is symmetric and* $\mathbf{S}$ *is invertible, then* $In(\mathbf{S}^T\mathbf{HS}) = In(\mathbf{H})$.

△

Proof A proof for a more general variant of the Theorem 9.1 for complex matrices can be found in Cain [3]. □

Since the decomposition $\mathbf{H} = \mathbf{LDL}^T$ for a symmetric matrix $\mathbf{H}$ generates an invertible matrix $\mathbf{L}$ it follows from Sylvester's theorem that $In(\mathbf{H}) = In(\mathbf{L}^{-1}\mathbf{HL}^{-T}) = In(\mathbf{D})$. Therefore, to prove positive definiteness of a matrix $\mathbf{H}$ we can compute the LDL^T-decomposition and check if the matrix $\mathbf{D}$ is diagonal with only positive entries.

To prove the positive definiteness of the reduced Hessian matrix the inertia of the KKT matrix

$$\mathbf{K} = \begin{pmatrix} \mathbf{G} & \mathbf{A}^T \\ \mathbf{A} & \mathbf{0}_{N_h \times N_h} \end{pmatrix}$$

can be calculated. Here, $\mathbf{G}$ is the Hessian matrix and $\mathbf{A}$ is the Jacobian of the active constraints. According to Chabrillac and Crouzeix [4], the reduced Hessian is positive definite if and only if the KKT matrix $\mathbf{K}$ has exactly N_y positive eigenvalues. If in

addition the Jacobian of the active constraints has full rank, the inertia of $\mathbf{K}$ must be $(N_y, N_s, 0)$. To calculate the inertia of the KKT matrix $\mathbf{K}$ the LDL^T-decomposition can be calculated and the matrix $\mathbf{D}$ be analyzed. Each positive 1×1-block corresponds to one positive eigenvalue, each negative 1×1-block to one negative eigenvalue, each zero 1×1-block to one zero eigenvalue, and each 2×2-block to a conjugated pair of eigenvalues and therefore to one positive and one negative eigenvalue.

9.2 Calculating Derivatives

For the application of SQP methods to large-scale optimization problems the Jacobian matrices of the constraints must be calculated and the structure of the Hessian of the Lagrangian must be known to be able to apply sparse Quasi-Newton updates. In order to perform a sensitivity analysis or to use exact second-order derivatives, instead of Quasi-Newton updates, the Hessian matrix of the Lagrangian function must be calculated, too. Therefore, potent algorithms are needed to detect the structure of sparse Jacobian and Hessian matrices and to calculate the first- and second-order derivatives. Because of sparse matrices, graph representations of the (bi)adjacency matrices can be exploited to reduce the number of function evaluations.

In the following paragraphs common algorithms are described to achieve these two tasks.

9.2.1 Computational Graphs

For the determination of sparsity patterns we first introduce *computational graphs*, which visualize the *dependence relations* of the sequential numerical operations performed during the evaluation of the constraints or the Lagrangian function. In so doing, we follow the definitions made in Griewank and Walther [18].

For a given function $\mathbf{f} : \mathbb{R}^{N_y} \to \mathbb{R}^{N_m}$ and a vertex set $\mathcal{V} = \{v_{1-N_y}, ..., v_{N_l}\}$ we assign the N_y-input-variables of the function $\mathbf{f}(\cdot)$ to the vertices $v_{1-N_y}, ..., v_0$ and the intermediate results of all N_l elementary operations of the function evaluation to the vertices $v_1, ..., v_{N_l}$. Then, the N_m-output-variables correspond to the vertices $v_{N_l-N_m+1}, ..., .v_{N_l}$. Thus, every vertex v_i with $i > 0$ is a result of an elementary operation $\phi_i(\cdot)$ with one argument $v_i = \phi_i(v_j)_{j \prec i} = \phi_i(v_j)$ or two arguments $v_i = \phi_i(v_j)_{j \prec i} = \phi_i(v_j, v_{\hat{j}})$. Here, the *dependence or precedence relation* $j \prec i$ means that v_i depends directly on v_j.

The vertex set $\mathcal{V}$ and the set of edges $\mathcal{B} = \{v_j v_i \mid j \prec i, i = 1, ..., N_l\}$ form a directed acyclic graph, which is called *computational graph*. The computational graph of a simple exemplary function is depicted in Fig. 9.2.

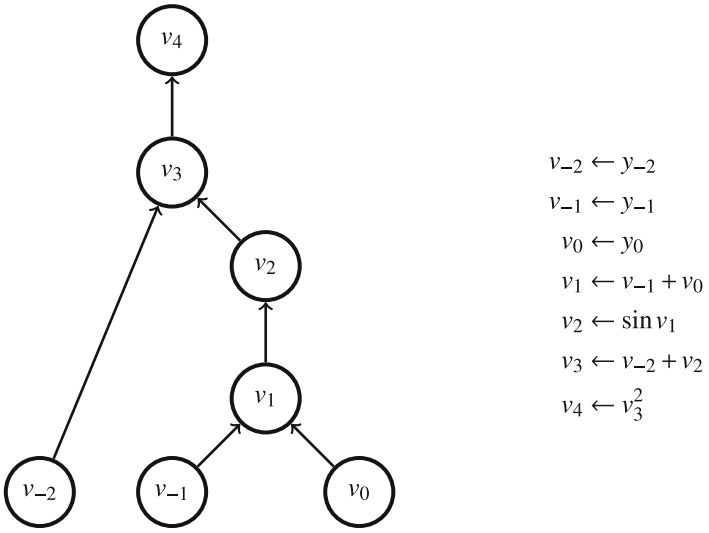


Fig. 9.2 A computational graph of $f(y) = (y_{-2} + sin(y_{-1} + y_0))^2$

9.2.2 Sparsity Pattern Determination

9.2.2.1 Sparsity Pattern of Jacobian Matrices

To determine the sparsity pattern of a Jacobian matrix of a differentiable function $\mathbf{f} : \mathbb{R}^{N_y} \rightarrow \mathbb{R}^{N_m}$ a method called *sparsity pattern propagation* can be used. The idea behind this method is to propagate the index sets of the nonzero values of the Jacobian matrix for each numerical calculation of the constraints according to the computational graph of the function. The method is described in Griewank and Walther [18].

Algorithm 9.4 Jacobian Sparsity Pattern Calculation (cf. Griewank and Walther [18])

1: **for** $i \leftarrow 1$ **to** N_y **do**
2: $\quad \mathcal{M}_{[i-N_y]} \leftarrow \{i\}$
3: **end for**
4: **for** $i \leftarrow 1$ **to** N_l **do**
5: $\quad$ **if** $\phi_i(v_j)$, i.e., $\phi_i(\cdot)$ depends only on one argument **then**
6: $\quad\quad \mathcal{M}_{[i]} \leftarrow \mathcal{M}_{[j]}$
7: $\quad$ **else if** $\phi_i(v_j, v_{\hat{j}})$, i.e., $\phi_i(\cdot)$ depends on two arguments **then**
8: $\quad\quad \mathcal{M}_{[i]} \leftarrow \mathcal{M}_{[j]} \cup \mathcal{M}_{[\hat{j}]}$
9: $\quad$ **end if**
10: **end for**

The set $\mathcal{M}$ contains the indices of all nonzero elements. Consequently, the index domains $\mathcal{M}_{[N_l-N_m+1]}$, ..., $\mathcal{M}_{[N_l]}$ contain the indices of the nonzero elements for each row of the Jacobian matrix because the inclusion

$$\left\{ j \leq N_n : \frac{\partial v_k}{\partial x_j} \not\equiv 0 \right\} \subseteq \mathcal{M}_{[k]}$$

holds. This sparsity pattern can be an overestimate of the real Jacobian structure because the algorithm does not account for numerical cancelation or degeneracy.

A practical implementation of the algorithm can be simply done by operator overloading, because the computational graph is traversed from the leaves to the root by the normal program execution. Every elementary operator used in the function evaluation must be overloaded with an implementation of lines 5–9 of Algorithm 9.4.

The result of the application of this algorithm to the constraints of a transcribed exemplary problem formulation from Chap. 10 by the direct collocation method with a step-size of 1 s for the MVEG test cycle (see Sect. 10.6) and the Labatto IIIA(4) discretization scheme can be seen in Fig. 9.3.

The rows above the dashed line are the equality constraints from the discretization of the *ordinary differential equations* (ODE) and the rows below the dashed line are

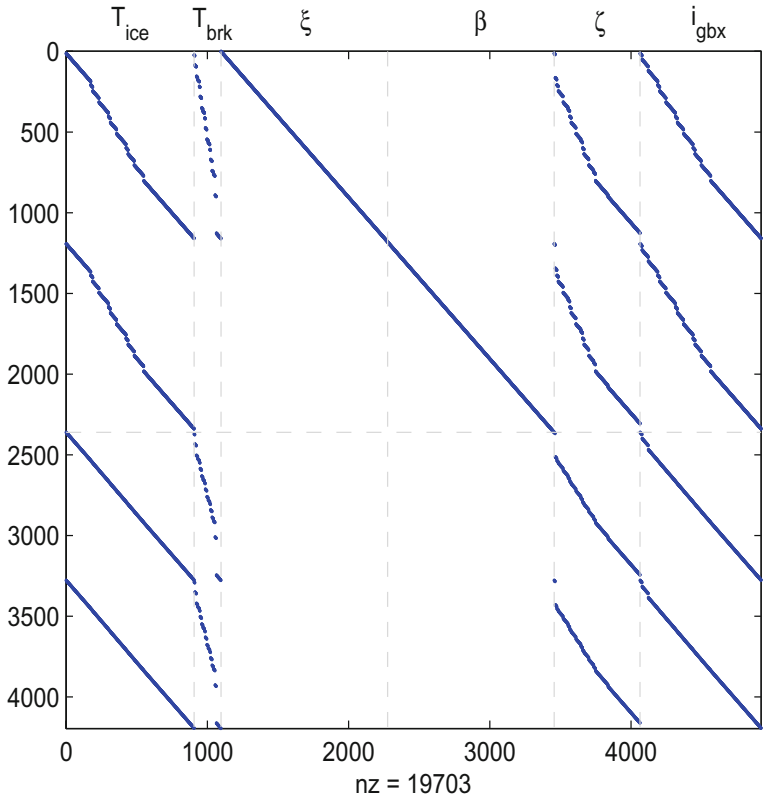


Fig. 9.3 Jacobian sparsity pattern of the constraints for an exemplary problem as Mayer-type. The rows are the equality and inequality constraints of the optimization variables T_{ice}, T_{brk}, ξ, β, ζ, and i_{gbx}. nz is the number of nonzero entries

the inequality constraints. The first two columns correspond to the continuous-valued controls (engine torque and brake torque), the third and fourth columns correspond to the discretized states of the collocation method (state of charge and fuel mass), and the last two columns correspond to the embedded discrete controls (hybrid drive mode and gear ratio).

9.2.2.2 Sparsity Pattern of Hessian Matrices

To determine the sparsity pattern of the Hessian of an at least twice differentiable function $f : \mathbb{R}^{N_y} \to \mathbb{R}$, as supposed for the Lagrangian function, we need to introduce additional index sets $\mathcal{N}_i$ called *nonlinear interaction domains* such that

$$\left\{ j \leq N_y : \frac{\partial^2 f}{\partial x_i \partial x_j} \not\equiv 0 \right\} \subseteq \mathcal{N}_{[i]}.$$

The nonlinear interaction domains can be calculated as an extension of the calculation of the index domains.

Algorithm 9.5 Hessian Sparsity Pattern Calculation (cf. Walther [42])

for $i \leftarrow 1$ **to** N_y **do**
  if v_i appears nonlinearly in $f(\cdot)$ **then**
    $\mathcal{M}_{[i-N_y]} \leftarrow \{i\}$
  else
    $\mathcal{M}_{[i-N_y]} \leftarrow \emptyset$
  end if
  $\mathcal{N}_{[i]} \leftarrow \emptyset$
end for
for $i = 1$ **to** N_l **do**
  if $\phi_i(v_j)$, i.e., $\phi_i(\cdot)$ depends only on one argument **then**
    $\mathcal{M}_{[i]} \leftarrow \mathcal{M}_{[j]}$
    if $\phi_i(v_j)$ is nonliner **then**
      $\mathcal{N}_{[k]} \leftarrow \mathcal{N}_{[k]} \cup \mathcal{M}_{[i]}, \quad \forall k \in \mathcal{M}_{[i]}$
    end if
  else if $\phi_i(v_j, v_{\hat{j}})$, i.e., $\phi_i(\cdot)$ depends on two arguments **then**
    $\mathcal{M}_{[i]} \leftarrow \mathcal{M}_{[j]} \cup \mathcal{M}_{[\hat{j}]}$
    if $\phi_i(v_j, v_{\hat{j}})$ is linear in v_j **then**
      $\mathcal{N}_{[k]} \leftarrow \mathcal{N}_{[k]} \cup \mathcal{M}_{[\hat{j}]}, \quad \forall k \in \mathcal{M}_{[j]}$
    else if $\phi_i(v_j, v_{\hat{j}})$ is nonlinear in v_j **then**
      $\mathcal{N}_{[k]} \leftarrow \mathcal{N}_{[k]} \cup \mathcal{M}_{[i]}, \quad \forall k \in \mathcal{M}_{[j]}$
    end if
    if $\phi_i(v_j, v_{\hat{j}})$ is linear in $v_{\hat{j}}$ **then**
      $\mathcal{N}_{[k]} \leftarrow \mathcal{N}_{[k]} \cup \mathcal{M}_{[j]}, \quad \forall k \in \mathcal{M}_{[\hat{j}]}$
    else if $\phi_i(v_j, v_{\hat{j}})$ is nonlinear in $v_{\hat{j}}$ **then**
      $\mathcal{N}_{[k]} \leftarrow \mathcal{N}_{[k]} \cup \mathcal{M}_{[i]}, \quad \forall k \in \mathcal{M}_{[\hat{j}]}$
    end if
  end if
end for

The result of Algorithm 9.5 is a set of nonlinear interaction domains $\mathcal{N} = \{\mathcal{N}_1, ..., \mathcal{N}_{N_y}\}$, which contains the indices of the nonzero elements for each row and column of the Hessian matrix. Similar to the index domains of a Jacobian matrix, the nonlinear interaction domains can be an overestimate for the Hessian sparsity pattern, because they do not account for numerical cancelation or degeneracy.

The Hessian for the transcribed exemplary problem formulation from Chap. 10 by the direct collocation method with a step-size of 1 s for the MVEG test cycle and the Labatto IIIA(4) discretization scheme can be seen in Fig. 9.4.

The first two rows and columns correspond to the continuous-valued controls (engine torque and brake torque), the third and fourth rows and columns correspond to the discretized states of the collocation method (state of charge and fuel mass), and the last two rows and columns correspond to the embedded discrete controls (hybrid drive mode and gear ratio).

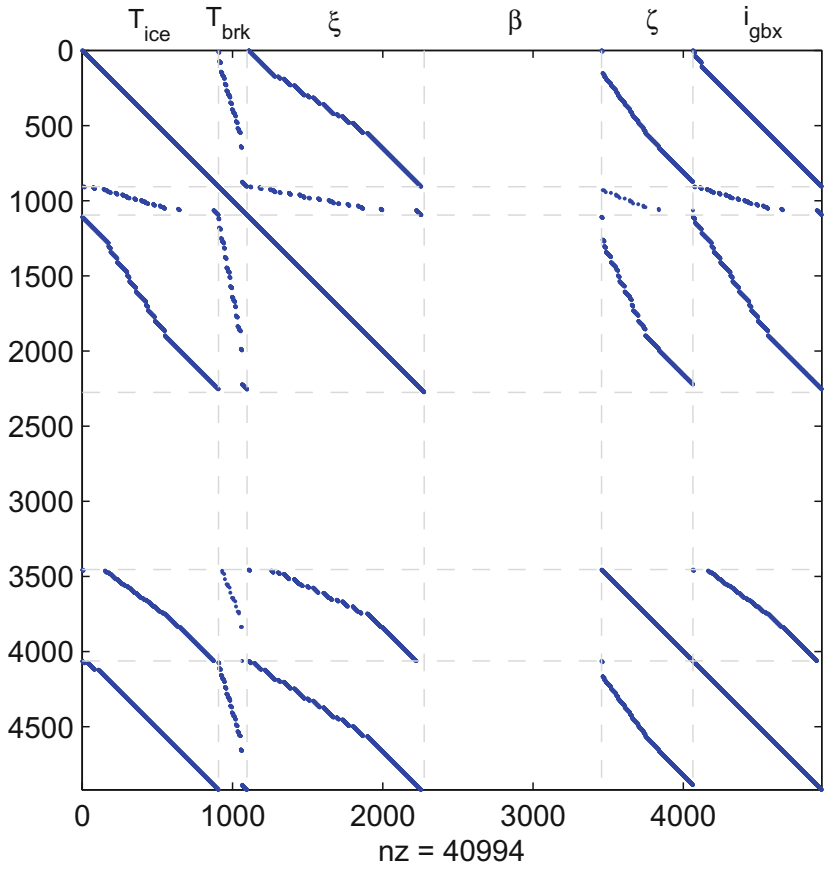


Fig. 9.4 Hessian sparsity pattern of the Lagrangian for an exemplary problem formulation as Mayer-type. T_{ice}, T_{brk}, ξ, β, ζ, and i_{gbx} are the optimization variables. nz is the number of nonzero entries

9.2.3 Compressed Derivative Calculation

For a given sparse matrix $\mathbf{H} \in \mathbb{R}^{N_n \times N_m}$ a column compressed matrix $\mathbf{B} = \mathbf{HS} \in \mathbb{R}^{N_n \times N_s}$ is defined by the binary *seed matrix* $\mathbf{S} \in \mathbb{R}^{N_m \times N_s}$, which describes the columns, that can be calculated simultaneously by numerical differentiation. A great advantage of the representation as a compressed matrix $\mathbf{B}$ is that its calculation requires usually much fewer function evaluations than the calculation of an uncompressed matrix. The seed matrix is closely related to coloring problems of the corresponding (bi)adjacency graphs. A great survey for this topic is given by Gebremedhin et al. [10].

These vertex colorings $\mathbf{c}$ of the (bi)adjacency graphs will be reviewed in the next two Sects. 9.2.3.1 and 9.2.3.2. If the coloring $\mathbf{c}$ is known, the seed matrix $\mathbf{S}$ for the compression of the Jacobian and the Hessian, respectively, can be calculated by

$$\mathbf{S}_{[i],[j]} = \begin{cases} 1, & \text{if } j = \mathbf{c}_{[i]}, \\ 0, & \text{otherwise.} \end{cases} \tag{9.1}$$

After the calculation of the compressed matrix $\mathbf{B}$, e.g., by finite differences, as described in Sect. 9.2.4, the uncompression to the full-size matrix $\mathbf{H}$ can be performed for a given coloring $\mathbf{c}$ by

$$\mathbf{H}_{[i],[j]} = \mathbf{B}_{[i],[\mathbf{c}_{[j]}]}. \tag{9.2}$$

9.2.3.1 Compressed Sparse Jacobians

The idea to reduce the number of function evaluations by a compressed calculation of sparse Jacobians can be traced back to Curtis et al. [6]. The authors described an algorithm for the partitioning of a sparse Jacobian into structurally orthogonal groups of columns, which will be represented in the following as submatrices for the sake of simplicity.

Definition 9.1 (*Structurally Orthogonal Groups of Rows and Columns*) Two groups of columns $\mathbf{C}_1 \subset \mathbf{H}$, $\mathbf{C}_1 \in \mathbb{R}^{N_n \times N_{m_1}}$, and $\mathbf{C}_2 \subset \mathbf{H}$, $\mathbf{C}_2 \in \mathbb{R}^{N_n \times N_{m_2}}$, of a matrix $\mathbf{H} \in \mathbb{R}^{N_n \times N_m}$ are said to be *structurally orthogonal*, if for any column $\mathbf{C}_1^{[:],[j_1]}$ and for any column $\mathbf{C}_2^{[:],[j_2]}$ there does not exist a row index i for which both $\mathbf{C}_1^{[i],[j_1]} \neq 0$ and $\mathbf{C}_2^{[i],[j_2]} \neq 0$ applies. In the same manner *structurally orthogonal columns* are defined, which is the case where $m_1 = m_2 = 1$ holds.

This definitions hold analogously for two rows or two groups of rows of the matrix.

△

Definition 9.2 (*Structurally Orthogonal Partition of Rows and Columns*) A column partition $\mathbf{C} = \{\mathbf{C}_1, ..., \mathbf{C}_{N_s}\}$ of a matrix $\mathbf{H} \in \mathbb{R}^{N_n \times N_m}$ into N_s groups of columns, whereby $C_j \in \mathbb{R}^{N_n \times N_{m_j}}$ and $\sum_{j=1}^{N_s} m_j = N_m$ holds, is called structurally orthogonal, if any two groups of columns are structurally orthogonal.

This definition holds analogously for a row partition of the matrix.

△

For the calculation of a column compressed Jacobian matrix a structurally orthogonal partition of columns must be determined, such that the number of groups N_s is minimized. Then, the binary seed matrix $\mathbf{S} \in \{0, 1\}^{N_n \times N_s}$ for the matrix compression can be defined by

$$\mathbf{S}_{[i],[j]} = \begin{cases} 1, & \text{if } \mathbf{C}_j^{[i],[k]} \neq 0, \text{ for any } k \in \{1, ..., m_j\} \\ 0, & \text{otherwise.} \end{cases} \tag{9.3}$$

As described by Coleman and Moré [5], the problem of finding a minimal column partition is equivalent to a distance-2 coloring problem for the bipartite graph representation $G = (\mathcal{V}_1, \mathcal{V}_2, \mathcal{B})$ of the matrix $\mathbf{H}$, which can be calculated by the following algorithm.

Algorithm 9.6 Distance-2 Coloring for a Bipartite Graph (cf. Gebremedhin et al. [10])

```
1: Let v_1, ..., v_{|V_2|} be a given ordering of V_2.
2: Initialize Γ with some value a ∉ V_2.
3: Let c be an array of length |V_2| initialized with zeros.
4: for i ← 1 to |V_2| do
5:     Determine N_1(v_i) = {w | w v_i ∈ B}.
6:     for all w ∈ N_1(v_i) do
7:         for all x ∈ N_1(w) with x ≠ a do
8:             Γ_{c_[x]} ← v_i
9:         end for
10:    end for
11:    c_[v_i] ← min{b > 0 | Γ_[b] ≠ v_i}
12: end for
```

The seed matrix, which is the same as (9.3), is then defined by (9.1) with the coloring $\mathbf{c}$.

9.2.3.2 Compressed Sparse Hessians

The idea of the compressed calculation of Jacobian matrices was extended to the compressed calculation of sparse Hessians by Powell and Toint [34]. For this purpose the authors defined symmetrically orthogonal partitions of rows and columns.

Definition 9.3 (*Symmetrically Orthogonal Partition of Rows and Columns*) A column partition $\mathbf{C} = \{\mathbf{C}_1, ..., \mathbf{C}_{N_s}\}$ of a symmetric matrix $\mathbf{H} \in \mathbb{R}^{N_n \times N_n}$ into N_s groups of columns, whereby $C_j \in \mathbb{R}^{N_n \times N_{n_j}}$ and $\sum_{j=1}^{N_s} n_j = N_n$ holds, is called

symmetrically orthogonal, if for every nonzero element $\mathbf{H}_{[i],[j]}$, either the group containing j-th column has no other column with a nonzero in the i-th row, or the group containing the i-th column has no other column with a nonzero in the j-th row.

This definition holds analogously for a partition of the rows of a symmetric matrix.

△

Then, similar to the compressed calculation of sparse Jacobians, we need to determine a symmetrically orthogonal partition of columns, such that the number of groups N_s is minimized.

According to McCormick [31], this problem is equivalent to the calculation of a *star coloring* of the adjacency graph representation of the Hessian, which exploits the symmetry of the Hessian. A *star coloring* satisfies the two conditions:

1. every pair of adjacent vertices receives a different color (distance-1 coloring); and
2. every path on four vertices uses at least three different colors.

The following algorithm for the computation of a star coloring for an adjacency graph $G = (\mathcal{V}, \mathcal{B})$ of a symmetric matrix can be found in Gebremedhin et al. [10].

Algorithm 9.7 Star Coloring for an Adjacency Graph (cf. Gebremedhin et al. [10])

Let $v_1, ..., v_{|\mathcal{V}|}$ be a given ordering of $\mathcal{V}$.
Initialize $\boldsymbol{\Gamma}$ with some value $a \notin \mathcal{V}$.
Let $\mathbf{c}$ be an array of length $|\mathcal{V}|$ initialized with zeros.
for $i \leftarrow 1$ **to** $|\mathcal{V}|$ **do**
    Determine $N_1(v_i) = \{w \mid wv_i \in \mathcal{B}\}$.
    for all $w \in N_1(v_i)$ **do**
        if $w \neq a$ **then**
            $\boldsymbol{\Gamma}_{[\mathbf{c}_{[w]}]} \leftarrow v_i$
        end if
        for all $x \in N_1(w)$ with $x \neq a$ **do**
            if $w = a$ **then**
                $\boldsymbol{\Gamma}_{[\mathbf{c}_{[x]}]} \leftarrow v_i$
            else
                if $\mathbf{c}_{[x]} < \mathbf{c}_{[w]}$ **then**
                $\boldsymbol{\Gamma}_{[\mathbf{c}_{[x]}]} \leftarrow v_i$
                end if
            end if
        end for
    end for
    $\mathbf{c}_{[v_i]} \leftarrow \min\{b > 0 : \boldsymbol{\Gamma}_{[b]} \neq v_i\}$
end for

The seed matrix is then defined by (9.1) with the coloring $\mathbf{c}$.

9.2.4 Finite Differences

A classical approach to the numerical calculation of derivatives for a function $f : \mathbb{R} \to \mathbb{R}$ are finite differences, which rely on the Taylor series expansion of the function

$$f(y+\epsilon) = f(y) + \epsilon \frac{\partial f}{\partial y}(y) + \frac{\epsilon^2}{2} \cdot \frac{\partial f}{\partial y}(\xi), \quad \xi \in [y, y+\epsilon], \quad \epsilon \in \mathbb{R}. \tag{9.4}$$

From Equation (9.4) the approximation of the first derivative by a *forward difference* can be stated as

$$\frac{\partial f}{\partial y}(y) = \frac{f(y+\epsilon) - f(y)}{\epsilon} + \mathcal{O}(\epsilon).$$

The *central difference* approximation for the first derivative is calculated by the difference of the two Taylor series

$$f(y+\epsilon) = f(y) + \epsilon \frac{\partial f}{\partial y}(y) + \frac{\epsilon^2}{2} \cdot \frac{\partial^2 f}{\partial y^2}(y) + \frac{\epsilon^3}{6} \cdot \frac{\partial^3 f}{\partial y^3}(\xi^+), \quad \xi^+ \in [y, y+\epsilon]$$

and

$$f(y-\epsilon) = f(y) - \epsilon \frac{\partial f}{\partial y}(y) + \frac{\epsilon^2}{2} \cdot \frac{\partial^2 f}{\partial y^2}(y) - \frac{\epsilon^3}{6} \cdot \frac{\partial^3 f}{\partial y^3}(\xi^-), \quad \xi^- \in [y-\epsilon, y],$$

which results in the approximation

$$\frac{\partial f}{\partial y}(y) = \frac{f(y+\epsilon) - f(y-\epsilon)}{2\epsilon} + \mathcal{O}(\epsilon^2).$$

Obviously, both approximations rely heavily on the *perturbation step-size* ϵ, but the error term of the central difference quotient approximation is quadratic in ϵ whereas the error term of the forward difference approximation is only linear in ϵ.

9.2.4.1 Gradient Calculation

The forward differences for the gradient of a function $f : \mathbb{R}^{N_y} \to \mathbb{R}$ can be calculated elementwise by

$$\frac{\partial f}{\partial y_i}(\mathbf{y}) = \frac{f(\mathbf{y} + \epsilon\, \mathbf{e}_i) - f(\mathbf{y})}{\epsilon} + \mathcal{O}(\epsilon), \text{ for } i = 1, ..., N_y,$$

where $\mathbf{e}_i$ is the i-th unit vector. The calculation of the gradient needs $N_y + 1$ function evaluations.

The gradient by central differences can be calculated elementwise by

$$\frac{\partial f}{\partial y_i}(\mathbf{y}) = \frac{f(\mathbf{y} + \epsilon\, \mathbf{e}_i) - f(\mathbf{y} - \epsilon\, \mathbf{e}_i)}{2\epsilon} + \mathcal{O}(\epsilon^2).$$

The central difference calculation has a higher precision than the forward difference calculation, but it also needs approximately double of the computation time because $2N_y$ function evaluations must be performed.

For a discretized OCP by the direct collocation method in Mayer form, the gradient of the objective function consists of only one known element and therefore the finite differences are not needed.

9.2.4.2 Jacobian Calculation

The Jacobian $\mathbf{J} \in \mathbb{R}^{N_r \times N_y}$ of a function $\mathbf{r} : \mathbb{R}^{N_y} \to \mathbb{R}^{N_r}$, e.g., a constraint function of a nonlinear programming problem, can be calculated columnwise by the forward or central finite difference schemes

$$\mathbf{J}_{[:],[i]}(\mathbf{y}) \approx \frac{\mathbf{r}(\mathbf{y} + \epsilon \mathbf{e}_i) - \mathbf{r}(\mathbf{y})}{\epsilon}, \text{ for } i = 1, ..., N_y$$

or

$$\mathbf{J}_{[:],[i]}(\mathbf{y}) \approx \frac{\mathbf{r}(\mathbf{y} + \epsilon \mathbf{e}_i) - \mathbf{r}(\mathbf{y} - \epsilon \mathbf{e}_i)}{2\epsilon}, \text{ for } i = 1, ..., N_y.$$

This leads to $N_y + 1$ evaluations of the function $\mathbf{r}(\cdot)$ for the forward difference scheme and $2N_y$ evaluations for the central difference scheme.

Using the compression technique by seed matrices, the sparsity pattern of the Jacobian can be exploited, because the differentiation of structurally orthogonal columns can be executed simultaneously. The compressed Jacobian $\mathbf{J}_c \in \mathbb{R}^{N_r \times N_s}$ can be calculated by

$$\mathbf{J}_c^{[:],[i]}(\mathbf{y}) \approx \frac{\mathbf{r}(\mathbf{y} + \epsilon\, \mathbf{p}_i) - \mathbf{r}(\mathbf{y})}{\epsilon}, \text{ for } i = 1, ..., N_s$$

where the perturbation vector $\mathbf{p}_i = \mathbf{S}_{[:],[i]}$ is the i-th column and N_s is the column dimension of the seed matrix, which is equal to the number of colors of the corresponding distance-2 coloring problem. The compressed Jacobian can then be uncompressed using (9.2).

So, instead of $N_y + 1$ constraint evaluations for forward differences or $2N_y$ evaluations for central differences, only $N_s + 1$ or $2N_s$ evaluations of the function $\mathbf{r}(\cdot)$ are necessary. For example, the forward difference calculation of the Jacobian matrix of Fig. 9.3 needs only 11 instead of 4921 function evaluations, if the described compression technique is used.

9.2.4.3 Hessian Calculation

The Hessian $\mathbf{B} \in \mathbb{R}^{N_y \times N_y}$ of a function $f : \mathbb{R}^{N_y} \to \mathbb{R}$ can be calculated by second-order forward difference quotients

$$\frac{\partial^2 f}{\partial y_i \partial y_j}(\mathbf{y}) = \frac{f(\mathbf{y} + \epsilon \mathbf{e}_i + \epsilon \mathbf{e}_j) - f(\mathbf{y} + \epsilon \mathbf{e}_i) - f(\mathbf{y} + \epsilon \mathbf{e}_j) + f(\mathbf{y})}{\epsilon^2} + \mathcal{O}(\epsilon),$$

or second-order central difference quotients

$$\begin{aligned}\frac{\partial^2 f}{\partial y_i \partial y_j}(\mathbf{y}) = &\frac{f(\mathbf{y} + \epsilon \mathbf{e}_i + \epsilon \mathbf{e}_j) - f(\mathbf{y} - \epsilon \mathbf{e}_i + \epsilon \mathbf{e}_j)}{4\epsilon^2} \\ &- \frac{f(\mathbf{y} + \epsilon \mathbf{e}_i - \epsilon \mathbf{e}_j) + f(\mathbf{y} - \epsilon \mathbf{e}_i - \epsilon \mathbf{e}_j)}{4\epsilon^2} + \mathcal{O}(\epsilon^2).\end{aligned}$$

Each entry of the Hessian must be computed separately and therefore, if we exploit the symmetry of the matrix, $\frac{1}{2}N_y^2 + \frac{3}{2}N_y + 1$ function evaluations for forward differences or $2(N_y^2 + N_y)$ for central differences must be performed if the Hessian is a full matrix. An a priori known sparsity pattern can be directly exploited and results in $\frac{1}{2}(N_{nz} + |\text{diag}(\mathbf{B})|) + N_y + 1$ function evaluations for forward differences and $2(N_{nz} + |\text{diag}(\mathbf{B})|)$ function evaluations for central differences, whereby N_{nz} is the number of nonzero elements of the Hessian matrix. But nonetheless, the effort for the computation of a large Hessian by numerical differentiation is very high.

If we restrict the Hessian calculation to the Lagrangian of a discretized OCP by the direct collocation method

$$\nabla_y^2 \mathcal{L}(\mathbf{y}, \boldsymbol{\lambda}) = \nabla_y^2 f(\mathbf{y}) + \sum_{i=1}^{N_g} \lambda_i \nabla_y^2 g_i(\mathbf{y})$$

whereby g_i is the i-th constraint with the corresponding Lagrange multiplier λ_i, we can further exploit the structure of the problem, if the OCP is stated in Mayer form. In this case, the Hessian $\nabla_y^2 f(\mathbf{y})$ has only one single entry of value 1, for which the position in the matrix is known. If we now use seed matrices for the Jacobian and the Hessian in combination, as it is shown in Algorithm 9.8, the number of the

remaining constraint evaluations can be drastically reduced. Please note, that we do not distinguish between equality and inequality constraints here for the sake of simplicity.

Algorithm 9.8 Compressed Calculation of Hessian of the Lagrangian by Forward Difference Quotients

1: Calculate $\nabla_y^2 f(\mathbf{y})$ by finite differences, if it is not known.
2: Calculate a distance-2 coloring $\mathbf{c}_J$ for the bipartite graph representation of the sparsity structure of the Jacobian of the constraints by Algorithm 9.6.
3: Let $N_J \leftarrow \max(\mathbf{c}_J)$ denote the number of colors used.
4: Compute the seed matrix $\mathbf{S}_J$ for the coloring $\mathbf{c}_J$ as defined by (9.1).
5: Calculate a star coloring $\mathbf{c}_B$ for the adjacency graph representation of the sparsity structure of the Hessian of the Lagrangian by Algorithm 9.7.
6: Let $N_B \leftarrow \max(\mathbf{c}_B)$ denote the number of colors used.
7: Compute the seed matrix $\mathbf{S}_B$ for the coloring $\mathbf{c}_B$ as defined by (9.1).
8: $\mathbf{p}_1 \leftarrow \mathbf{g}(\mathbf{y})$
9: **for** $i \leftarrow 1$ **to** N_B **do**
10: $\quad \mathbf{y}_B \leftarrow \mathbf{y} + \epsilon \mathbf{S}_B^{[:],[i]}$
11: $\quad \mathbf{p}_2 \leftarrow \mathbf{g}(\mathbf{y}_B)$
12: $\quad$ **for** $j \leftarrow 1$ **to** N_J **do**
13: $\quad\quad \mathbf{p}_3 \leftarrow \mathbf{g}\left(\mathbf{y} + \epsilon \mathbf{S}_J^{[:],[j]}\right)$
14: $\quad\quad \mathbf{p}_4 \leftarrow \mathbf{g}\left(\mathbf{y}_B + \epsilon \mathbf{S}_J^{[:],[j]}\right)$
15: $\quad\quad \mathbf{B}_{gcc}^{[j]} \leftarrow \dfrac{\mathbf{p}_1 - \mathbf{p}_2 - \mathbf{p}_3 + \mathbf{p}_4}{\epsilon^2}$
16: $\quad$ **end for**
17: $\quad$ Uncompress the double compressed matrix $\mathbf{B}_{gcc}$ with the coloring $\mathbf{c}_J$ by (9.2) to obtain $\mathbf{B}_{gunc}$.
18: $\quad \mathbf{B}_{gc}^{[:],[i]} \leftarrow \mathbf{B}_{gunc}^T \boldsymbol{\lambda}$.
19: **end for**
20: Uncompress the compressed matrix $\mathbf{B}_{gc}$ with the coloring $\mathbf{c}_B$ by Algorithm 9.2 to obtain $\mathbf{B}_g$.
21: Calculate the Hessian of the Lagrangian $\nabla_y^2 \mathcal{L}(\mathbf{y}, \boldsymbol{\lambda}) \leftarrow \nabla_y^2 f(\mathbf{y}) + \mathbf{B}_g$

Remark 9.1 The forward difference scheme in Algorithm 9.8 can be replaced by the central difference scheme without any problems.

In case of a discretized OCP in Mayer form by a direct collocation method, the calculation of the Hessian of the Lagrangian by Algorithm 9.8 needs only $2s_B s_J + s_B + 1$ constraint function evaluations and no objective function evaluations, because $\nabla_y^2 f(\mathbf{y})$ is explicitly known. For example, the compressed forward differences calculation of the Hessian from Fig. 9.4 needs only 316 constraint evaluations and no objective function evaluations, whereas the calculation without the exploitation of the sparsity structure would require 27303 constraint evaluations.

9.2.4.4 Perturbation Step-Size

The choice of an optimal perturbation step-size ϵ for finite differences is a difficult task, because it depends on roundoff errors, truncation errors, numerical errors, on the type and order of the finite difference scheme, and on the order of the derivation, which shall be calculated.

In Mathur [30], a good overview on the existing methods for the perturbation step-size calculation and estimations of nearly optimal perturbation step-sizes are given. If the errors are ignored and for double precision operations a maximal overall error of $u = 10^{-16}$ is assumed, a nearly optimal perturbation step-size for the forward difference quotients for the gradient and Jacobian calculation is $\epsilon = 10^{-8}$ and for the central difference quotients $\epsilon = 10^{-6}$. Under the same assumptions, a nearly optimal perturbation step-size for the calculation of the Hessian by forward difference quotients is $\epsilon = 10^{-5}$ and by central difference quotients $\epsilon = 10^{-4}$.

9.3 Sparse Quasi-Newton Updates

Quasi-Newton update formulas, as described in Sect. 2.2.4, lead to dense matrices in general, even if the exact Hessian has a sparse structure. Therefore, these Quasi-Newton update formulas cannot be directly applied to large-scale optimization methods, which arise from the discretization of optimal control problems with many discretization points. The computation time and the memory consumption of dense Quasi-Newton updates are not acceptable for most applications.

To achieve a better large-scale performance, sparse Quasi-Newton update algorithms should be used, which exploit the structure of discretized optimal control problems. These updates generate sparse Quasi-Newton matrices with the same or nearly the same sparsity structure as the Hessian structure of the Lagrangian. Nevertheless, they preserve the positive definiteness of the Hessian.

9.3.1 Quasi-Newton Update for Partially Separable Function

The first sparse Quasi-Newton update algorithm which exploits the structure of low-order Runge–Kutta discretized optimal control problems was introduced by Toint and Griewank [41]. This update strategy performs many small dense blockwise updates instead of one big update, if the Lagrangian of the optimization problem is *partially separable*. A function is called *partially separable*, if it can be stated as a sum of simpler functions

$$f(\mathbf{y}) = \sum_{i=1}^{N_n} f_i(\mathbf{y}),$$

which only depend on some of the optimization variables. Then, the gradient

$$\nabla f(\mathbf{y}) = \sum_{i=1}^{N_n} \nabla f_i(\mathbf{y})$$

and the Hessian

$$\nabla^2 f(\mathbf{y}) = \sum_{i=1}^{N_n} \nabla^2 f_i(\mathbf{y})$$

can also be separated and can therefore be calculated separately only for the variables appearing in each summand.

From a graph theoretical view, the adjacency matrix for the Hessian of a partially separable function is disconnected and consists of as many subgraphs as the number of summands of the partial separable function. Obviously, the updated Quasi-Newton matrix $\mathbf{B}_{k+1}$ is positive definite, if all the updated summands of $\mathbf{B}_{k+1}$ are positive definite. Then, the secant Condition (2.19) is satisfied too.

Partially separable Lagrangians are generated by the discretization of optimal control problems, if a Radau IIA(1) or a Lobatto IIIA(2) Runge–Kutta formula is used to discretize the dynamic system.

9.3.2 *Simple Quasi-Newton Update for Chordal Sparsity Structures*

If the Lagrangian of a discretized optimal control problem does not consist of separate blocks only, the update strategy for partially separable functions cannot be applied. This is generally the case, if a Radau IIA(3) or Lobatto IIIA(4) implicit Runge–Kutta formula is used to discretize the dynamic system. If one of these discretization schemes is used, the Hessian of the Lagrangian consists of overlapping blocks.

One simple approach to apply a block Quasi-Newton update anyway, is to delete those entries, which connect the subgraphs. For the calculation of the subgraphs by the deletion of a preferably small number of edges, the graph can be completed to a chordal graph by applying Algorithm 9.2 and then the vertices, which are elements of two different cliques, can be removed from one of these cliques. Therefore, in a first step the N_c-cliques $\mathcal{C}_{cl} = \{\mathcal{C}_{cl,1}, \mathcal{C}_{cl,2}, \ldots, \mathcal{C}_{cl,N_c}\}$ of the adjacency graph must be determined. This can be done by Algorithm 9.9.

Algorithm 9.9 Clique Determination for the Graph $G(\mathcal{V}, \mathcal{B})$ and the Elimination Ordering $\boldsymbol{\pi}$

$\mathcal{A} \leftarrow \emptyset$
for $i \leftarrow 1$ **to** N_c **do**
  $\mathcal{S}_{[i]} \leftarrow 1$
end for
$j \leftarrow 1$
$v \leftarrow \boldsymbol{\pi}^{-1}_{[1]}$
$\mathcal{S}_{[v]} \leftarrow 0$
for all $(v, w) \in \mathcal{B}$ **do**
  $\mathcal{A} \leftarrow \mathcal{A} \cup w$
end for
$\mathcal{C}^{[1]}_{cl} \leftarrow \mathcal{A}$
for $i \leftarrow 2$ **to** N_c **do**
  $n_1 \leftarrow \sum_{k=1}^{|\mathcal{A}|} \mathcal{S}_{[\mathcal{A}_{[k]}]}$
  $v \leftarrow \boldsymbol{\pi}^{-1}_{[i]}$
  $\mathcal{S}_{[v]} \leftarrow 0$
  $\mathcal{A} \leftarrow \emptyset$
  for all $(v, w) \in \mathcal{B}$ **do**
    $\mathcal{A} \leftarrow \mathcal{A} \cup w$
  end for
  $n_2 \leftarrow \sum_{k=1}^{|\mathcal{A}|} \mathcal{S}_{[\mathcal{A}_{[k]}]}$
  if $n_1 <= n_2$ **then**
    $j \leftarrow j + 1$
    $\mathcal{C}^{[j]}_{cl} \leftarrow v$
    for $k \leftarrow 1$ **to** $|\mathcal{A}|$ **do**
      if $\mathcal{S}_{[\mathcal{A}_{[k]}]} = 1$ **then**
        $\mathcal{C}^{[j]}_{cl} \leftarrow \mathcal{C}^{[j]}_{cl} \cup \mathcal{A}_{[k]}$
      end if
    end for
  end if
end for

For the N_c-cliques the simplicial vertex sets $\mathcal{S} = \{\mathcal{S}_1, ..., \mathcal{S}_{N_c}\}$, which are processed by the elimination ordering $\boldsymbol{\pi}$, are defined by

$$\mathcal{S}_{[i]} = \mathcal{C}^{[i]}_{cl} \setminus \left(\mathcal{C}^{[i+1]}_{cl} \cup \ldots \cup \mathcal{C}^{[N_c]}_{cl}\right), \ \forall i = 1, \ldots, N_c.$$

The remaining vertices for each clique, after the removal of the simplicial vertices, are then defined by the set $\mathcal{R} = \{\mathcal{R}_1, ..., \mathcal{R}_{N_c}\}$ with

$$\mathcal{R}_{[i]} = \mathcal{C}^{[i]}_{cl} \cap \left(\mathcal{C}^{[i+1]}_{cl} \cup \ldots \cup \mathcal{C}^{[N_c]}_{cl}\right), \ \forall i = 1, \ldots, N_c.$$

Therefore, for each clique $\mathcal{C}^{[i]}_{cl} = \mathcal{S}_{[i]} \cup \mathcal{R}_{[i]}$ holds.

To efficiently disconnect the chordal graph we remove those edges, that connect the vertices of the sets $\mathcal{S}_{[i]}$ with the vertices of the sets $\mathcal{R}_{[i]}$ for each $i = 1, ..., N_c$. In other words, an adjacency graph, which consists only of the cliques $\mathcal{S}_{[i]}$ for all $i = 1, ..., N_c$, is used for the approximation of the Hessian structure. Then, the update for partially separable functions can be applied.

Algorithm 9.10 implements the simple update strategy for Hessians with overlapping blocks.

Algorithm 9.10 Simple Block Quasi-Newton Update for Chordal Sparsity Structures

1: Calculate an elimination ordering $\boldsymbol{\pi}$ for the adjacency graph $G(\mathcal{V}, \mathcal{B})$ of the Hessian sparsity structure, e.g., by Algorithm 9.3.
2: Compute a chordal extension $\mathcal{B}_{fill}$ of the sparsity structure by Algorithm 9.2.
3: Determine the cliques $\mathcal{C}_{cl,1}, \ldots, \mathcal{C}_{cl,N_c}$ of the filled graph $G(\mathcal{V}, \mathcal{B}_{fill})$ by Algorithm 9.9 with the elimination ordering $\boldsymbol{\pi}$.
4: Determine the simplicial vertices $\mathcal{S}_{[i]} = \mathcal{C}_{cl}^{[i]} \setminus \left(\mathcal{C}_{cl}^{[i+1]} \cup \ldots \cup \mathcal{C}_{cl}^{[N_c]}\right),\ \forall i = 1, \ldots, N_c$.
5: Then, repeat the following steps in each NLP iteration for the Quasi-Newton update:
6: **for** $i \leftarrow 1$ **to** N_c **do**
7: Update the sub-matrix $\mathbf{B}_{[\mathcal{S}_{[i]}],[\mathcal{S}_{[i]}]}$ (induced by the sets $\mathcal{S}_{[i]}$) by an update formula from Sect. 2.2.4.
8: **end for**

Steps 1–4 must be calculated once at the beginning of the NLP solution procedure and steps 6–8 describe the actual update routine.

Algorithm 9.10 ensures for every iteration step, that the secant equation holds and that the positive definiteness is preserved, if all blocks can be updated by a dense Quasi-Newton update. A major advantage of the algorithm is the "local nature" of the algorithm due to the partial updates. So, even if small steps are taken in some directions, most of the blocks can be updated anyway. The only drawback of the algorithm is, that the structure of the approximated Hessian differs from the exact Hessian and therefore the approximated Hessian cannot converge to the exact Hessian, which could guarantee a superlinear convergence rate.

9.3.3 Quasi-Newton Update for Chordal Sparsity Structures

Fortunately, the simple blockwise Quasi-Newton update can be modified slightly to facilitate the implementation of the same idea for the sparsity structure of the exact Hessian, if this structure is chordal. If it is not chordal, the sparsity structure must be extended by the calculation of the fill-in to a chordal structure beforehand.

We need the following theorem for the derivation of the algorithm.

Theorem 9.2 *Let* $\mathbf{B} \in \mathbb{R}^{N_y \times N_y}$ *be a sparse matrix with a chordal adjacency graph and* $\mathcal{C}_{cl} = \left\{\mathcal{C}_{cl,1}, \ldots, \mathcal{C}_{cl,N_c}\right\}$ *the cliques of the graph. We denote by* $\mathbf{B}_{\mathcal{C}_{cl}^{[k]}} \in \mathbb{R}^{N_y \times N_y}$ *for* $k = 1, ..., N_c$ *the matrices with entries at the indices defined by the sets* $\mathcal{C}_{cl}^{[k]} \times \mathcal{C}_{cl}^{[k]}$ *only and by* $\bar{\mathbf{B}}_{\mathcal{C}_{cl}^{[k]}} \in \mathbb{R}^{|\mathcal{C}_{cl}^{[k]}| \times |\mathcal{C}_{cl}^{[k]}|}$ *the dense partial matrices of* $\mathbf{B}_{\mathcal{C}_{cl}^{[k]}}$.

Furthermore, let the matrix $\mathbf{B}$ *be defined by*

$$\mathbf{B} = \sum_{k=1}^{N_c} \mathbf{B}_{\mathcal{C}_{cl}^{[k]}}. \tag{9.5}$$

Then, $\mathbf{B}$ *is positive definite, if every* $\bar{\mathbf{B}}_{\mathcal{C}_{cl}^{[k]}}$ *is positive definite and every index* $j \in \{1, ..., N_y\}$ *is contained in at least one clique* $\mathcal{C}_{cl}^{[k]}$.

△

Proof Since it is assumed that each $\bar{\mathbf{B}}_{\mathcal{C}_{cl}^{[k]}}$ is positive definite, the matrices $\mathbf{B}_{\mathcal{C}_{cl}^{[k]}}$ are all positive semidefinite. Therefore, the relation $\mathbf{y}^T\mathbf{B}\mathbf{y} = \sum_{k=1}^{N_c} \mathbf{y}^T\mathbf{B}_{\mathcal{C}_{cl}^{[k]}}\mathbf{y} \geq 0$ holds, for every positive $\mathbf{y} \in \mathbb{R}^{N_y}$. The stricter relation $\mathbf{y}^T\mathbf{B}\mathbf{y} > 0$ holds, if at least one of the summands $\mathbf{y}^T\mathbf{B}_{\mathcal{C}_{cl}^{[k]}}\mathbf{y}$ is greater than zero. Since $\mathbf{y}$ has at least one entry that is not zero, we assume $y_j \neq 0$, $j \in \{1, ..., N_y\}$ without loss of generality. Let $\tilde{k}$ be a specific choice of index and let $\mathcal{C}_{cl}^{[\tilde{k}]}$ be a clique which contains j. Note, that such a clique exists per assumption. Then, $\mathbf{y}_{\mathcal{C}_{cl}^{[\tilde{k}]}}^T \bar{\mathbf{B}}_{\mathcal{C}_{cl}^{[\tilde{k}]}} \mathbf{y}_{\mathcal{C}_{cl}^{[\tilde{k}]}} > 0$ and therefore $\mathbf{y}^T\mathbf{B}_{\mathcal{C}_{cl}^{[\tilde{k}]}}\mathbf{y} > 0$, which proves the theorem. □

So, if the partial matrices of a positive definite matrix, which are defined by the cliques $\mathcal{C}_{cl,1}, \ldots, \mathcal{C}_{cl,N_c}$, are updated by a dense Quasi-Newton formula, the matrix defined by (9.5) is also positive definite. For this matrix

$$\tilde{\boldsymbol{\gamma}} := \mathbf{B}\boldsymbol{\delta} = \sum_{k=1}^{N_c} \mathbf{B}_{\mathcal{C}_{cl}^{[k]}}\boldsymbol{\delta} = \sum_{k=1}^{N_c} \boldsymbol{\gamma}_k$$

holds.

To satisfy the secant Condition (2.19),

$$\tilde{\boldsymbol{\gamma}} = \boldsymbol{\gamma} \tag{9.6}$$

must hold. Equation (9.6) is satisfied for all cliques that contributes to each entry $\tilde{\boldsymbol{\gamma}}_{[k]}$. Therefore, the number of cliques must be identified, which contains k. Let this number be denoted by N_γ^k and by the vector of clique counts $\mathbf{N}_\gamma = \left[N_{\gamma,1}, ..., N_{\gamma,N_y}\right]^T$. If we now use the vector

$$\bar{\boldsymbol{\gamma}} = \left[\frac{\boldsymbol{\gamma}_{[1]}}{\mathbf{N}_\gamma^{[1]}}, \cdots, \frac{\boldsymbol{\gamma}_{[N_y]}}{\mathbf{N}_\gamma^{[N_y]}}\right]^T$$

substitutional for $\boldsymbol{\gamma}$ in the Quasi-Newton update formula of each partial matrix, Condition (9.6) holds.

The updated matrix is therefore positive definite, satisfies the secant condition, and preserves the sparsity structure of the exact Hessian. Algorithm 9.11 summarizes the entire algorithm.

Algorithm 9.11 Block Quasi-Newton Update for Chordal Sparsity Structures

1: Calculate an elimination ordering $\boldsymbol{\pi}$ for the adjacency graph $G(\mathcal{V}, \mathcal{B})$ of the exact Hessian sparsity structure, e.g., by Algorithm 9.3.
2: Compute a chordal extension $\mathcal{B}_{fill}$ of the sparsity structure by Algorithm 9.2.
3: Determine the cliques $\mathcal{C}_{cl} = \{\mathcal{C}_{cl,1}, \ldots, \mathcal{C}_{cl,N_c}\}$ of the filled graph $G(\mathcal{V}, \mathcal{B}_{fill})$ by Algorithm 9.9 with the elimination ordering $\boldsymbol{\pi}$.
4: **for** $i \leftarrow 1$ **to** N_y **do**
5: Initialize $\mathbf{N}_\gamma^{[i]} \leftarrow 0$.
6: **end for**
7: **for** $i \leftarrow 1$ **to** N_c **do**
8: Initialize the partial matrix $\bar{\mathbf{B}}_{\mathcal{C}_{cl,i}} = \mathbf{I}_{|\mathcal{C}_{cl,i}|}$.
9: **for** $j \leftarrow 1$ **to** $|\mathcal{C}_{cl,i}|$ **do**
10: $\mathbf{N}_\gamma^{\left[\mathcal{C}_{cl,i}^{[j]}\right]} \leftarrow \mathbf{N}_\gamma^{\left[\mathcal{C}_{cl,i}^{[j]}\right]} + 1$.
11: **end for**
12: **end for**
13: Then, repeat the following steps in each NLP iteration for the Quasi-Newton update:
14: Calculate $\bar{\boldsymbol{\gamma}} = \left[\frac{\boldsymbol{\gamma}_{[1]}}{\mathbf{N}_\gamma^{[1]}}, \ldots, \frac{\boldsymbol{\gamma}_{[N_y]}}{\mathbf{N}_\gamma^{[N_y]}}\right]^T$.
15: **for** $i \leftarrow 1$ **to** N_c **do**
16: Perform a dense Quasi-Newton update for the partial matrix $\bar{\mathbf{B}}_{\mathcal{C}_{cl}^{[i]}}$ induced by the clique $\mathcal{C}_{cl}^{[i]}$ with $\boldsymbol{\delta}_{\left[\mathcal{C}_{cl}^{[i]}\right]}$ and $\bar{\boldsymbol{\gamma}}_{\left[\mathcal{C}_{cl}^{[i]}\right]}$ as defined in Sect. 2.2.4.
17: **end for**
18: Calculate the updated matrix $\mathbf{B} = \sum_{i=1}^{N_c} \mathbf{B}_{\mathcal{C}_{cl}^{[i]}}$.

Steps 1–12 must be performed once at the beginning of the NLP solution procedure and steps 14–18 describe the actual update routine. $\boldsymbol{\gamma}$ and $\boldsymbol{\delta}$ are defined as in Sect. 2.2.4. The same procedure can be analogously used with a damped BFGS update, if $\boldsymbol{\gamma}$ is replaced by $\boldsymbol{\eta}$.

9.3.4 *Modifications of the Quasi-Newton Update*

In Quasi-Newton updates the curvature condition $\boldsymbol{\delta}_k^T \boldsymbol{\gamma}_k$ can become very small, which leads to very badly scaled matrices. In this case the Quasi-Newton update can be modified to get a numerically more stable update.

As described in Gill et al. [16] a new point $\tilde{\mathbf{x}}_k$ can be defined, for which

$$\begin{aligned}\tilde{\boldsymbol{\delta}}_k &= \mathbf{x}_{k+1} - \tilde{\mathbf{x}}_k \\ \tilde{\boldsymbol{\gamma}}_k &= \nabla_y \mathcal{L}(\mathbf{x}_{k+1}, \boldsymbol{\lambda}_{k+1}, \boldsymbol{\mu}_{k+1}) - \nabla_y \mathcal{L}(\tilde{\mathbf{x}}_k, \boldsymbol{\lambda}_{k+1}, \boldsymbol{\mu}_{k+1})\end{aligned}$$

can be calculated for the usage in the Quasi-Newton update.

The new point $\tilde{\mathbf{x}}_k$ is chosen as a feasible point of the QP subproblem. In Gill et al. [16] the first feasible iterate of the QP-solver is taken. Another possible choice is $\tilde{\mathbf{x}}_k = \mathbf{x}_k + \alpha_k \tilde{\mathbf{d}}_k$, where α_k is the step-size of the current SQP iterate and $\tilde{\mathbf{d}}_k$ is the solution of the following linear subproblem.

Definition 9.4 (*Linear Subproblem*)

$$\begin{aligned} \min_{\tilde{\mathbf{d}}_k \in \mathbb{R}^{N_y}} \quad & \nabla f^T(\mathbf{y}_k)\tilde{\mathbf{d}}_k \\ \text{subject to} \quad & \mathbf{g}(\mathbf{y}_k) + \nabla \mathbf{g}^T(\mathbf{y}_k)\tilde{\mathbf{d}}_k \leq \mathbf{0}, \\ & \mathbf{h}(\mathbf{y}_k) + \nabla \mathbf{h}^T(\mathbf{y}_k)\tilde{\mathbf{d}}_k = \mathbf{0}. \end{aligned}$$

△

The same modification can also be applied to the modified BFGS update, for which $\boldsymbol{\gamma}_k$ is replaced with $\boldsymbol{\eta}_k$.

9.3.5 Quasi-Newton Updates for Discretized Optimal Control Problems

In the statement of Algorithm 9.11 the question about the choice of a suitable dense Quasi-Newton update for the partial matrices was left open.

Since the SR1-Update (2.2.4) provides the best convergence results, it should be used as often as possible. Unfortunately, the SR1-Update can only be applied if the curvature Condition (2.20) holds and the numerical stability is preserved, which requires the stricter condition $\boldsymbol{\delta}_k^T \boldsymbol{\gamma}_k > \epsilon_1$ with a small $\epsilon_1 > 0$. If the SR1-Update cannot be applied, the damped Quasi-Newton (2.54) with the substitution of $\boldsymbol{\gamma}_k$ by $\boldsymbol{\eta}_k$ should be used, which is defined by (2.55). For the application of the damped Quasi-Newton update the condition $\boldsymbol{\delta}_k^T \boldsymbol{\eta}_k > \epsilon_2$ with a small $\epsilon_2 > 0$ must hold. If the damped Quasi-Newton can also not be applied, one can try the modification described in Sect. 9.3.4 and perform a damped Quasi-Newton update with the newly computed $\tilde{\boldsymbol{\delta}}_k$ and $\tilde{\boldsymbol{\eta}}_k$, if the condition $\tilde{\boldsymbol{\delta}}_k^T \tilde{\boldsymbol{\eta}}_k > \epsilon_3$ with a small $\epsilon_3 > 0$ holds. If this update can also not be applied, one has to skip the update for the partial matrix. Motivated by the damped Quasi-Newton update we choose

$$\begin{aligned} \epsilon_1 &= 10^{-8} + \boldsymbol{\delta}_k^T \mathbf{B}_k \boldsymbol{\delta}_k \\ \epsilon_2 &= \beta \boldsymbol{\delta}_k^T \mathbf{B}_k \boldsymbol{\delta}_k \\ \epsilon_3 &= \beta \tilde{\boldsymbol{\delta}}_k^T \mathbf{B}_k \tilde{\boldsymbol{\delta}}_k \end{aligned}$$

with a small $\beta > 0$. A common choice is $\beta \in (0.2, 0.3)$.

The entire Quasi-Newton update algorithm for discretized optimal control problems is stated in the following algorithm.

Algorithm 9.12 Block Quasi-Newton Update for Discretized Optimal Control Problems

Choose $\beta \in [0.2, 0.3]$.
Define $\epsilon_1 = 10^{-8}$.
Calculate an elimination ordering $\boldsymbol{\pi}$ for the adjacency graph $G(\mathcal{V}, \mathcal{B})$ of the exact Hessian sparsity structure, e.g., by Algorithm 9.3.
Compute a chordal extension $\mathcal{B}_{fill}$ of the sparsity structure by Algorithm 9.2.
Determine the cliques $\mathcal{C}_{cl} = \{\mathcal{C}_{cl,1}, \mathcal{C}_{cl,2}, \ldots, \mathcal{C}_{cl,N_c}\}$ of the filled graph $G(\mathcal{V}, \mathcal{B}_{fill})$ by Algorithm 9.9 with the elimination ordering $\boldsymbol{\pi}$.
for $i \leftarrow 1$ **to** N_y **do**
    Initialize $\mathbf{N}_\gamma^{[i]} \leftarrow 0$.
end for
for $i \leftarrow 1$ **to** N_c **do**
    Initialize the partial matrix $\bar{\mathbf{B}}_{\mathcal{C}_{cl,i}} = \mathbf{I}_{|\mathcal{C}_{cl,i}|}$.
    for $j \leftarrow 1$ **to** $|\mathcal{C}_{cl,i}|$ **do**
        $\mathbf{N}_\gamma^{\left[\mathcal{C}_{cl,i}^{[j]}\right]} \leftarrow \mathbf{N}_\gamma^{\left[\mathcal{C}_{cl,i}^{[j]}\right]} + 1$.
    end for
end for
Then, repeat the following steps in each NLP iteration for the Quasi-Newton update:
Calculate $\bar{\boldsymbol{\gamma}} = \left(\frac{\boldsymbol{\gamma}_{[1]}}{\mathbf{N}_\gamma^{[1]}}, \ldots, \frac{\boldsymbol{\gamma}_{[N_y]}}{\mathbf{N}_\gamma^{[N_y]}}\right)^T$, $\bar{\boldsymbol{\eta}} = \left(\frac{\boldsymbol{\eta}_{[1]}}{\mathbf{N}_\gamma^{[1]}}, \ldots, \frac{\boldsymbol{\eta}_{[N_y]}}{\mathbf{N}_\gamma^{[N_y]}}\right)^T$, and $\bar{\tilde{\boldsymbol{\eta}}} = \left(\frac{\tilde{\boldsymbol{\eta}}_{[1]}}{\mathbf{N}_\gamma^{[1]}}, \ldots, \frac{\tilde{\boldsymbol{\eta}}_{[N_y]}}{\mathbf{N}_\gamma^{[N_y]}}\right)^T$
for $i \leftarrow 1$ **to** N_c **do**
    if $\boldsymbol{\delta}_{\left[\mathcal{C}_{cl}^{[i]}\right]}^T \bar{\boldsymbol{\gamma}}_{\left[\mathcal{C}_{cl}^{[i]}\right]} > \epsilon_1 + \boldsymbol{\delta}_{\left[\mathcal{C}_{cl}^{[i]}\right]}^T \bar{\mathbf{B}}_{\left[\mathcal{C}_{cl}^{[i]}\right]} \boldsymbol{\delta}_{\left[\mathcal{C}_{cl}^{[i]}\right]}$ **then**
        Perform a dense SR1 update for the partial matrix $\bar{\mathbf{B}}_{\left[\mathcal{C}_{cl}^{[i]}\right]}$ induced by the clique $\mathcal{C}_{cl}^{[i]}$ with $\boldsymbol{\delta}_{\left[\mathcal{C}_{cl}^{[i]}\right]}$ and $\bar{\boldsymbol{\gamma}}_{\left[\mathcal{C}_{cl}^{[i]}\right]}$, as defined in Sect. 2.2.4 .
    else
        Calculate $\epsilon_2 = \beta \boldsymbol{\delta}_{\left[\mathcal{C}_{cl}^{[i]}\right]}^T \bar{\mathbf{B}}_{\left[\mathcal{C}_{cl}^{[i]}\right]} \boldsymbol{\delta}_{\left[\mathcal{C}_{cl}^{[i]}\right]}$.
        if $\boldsymbol{\delta}_{\left[\mathcal{C}_{cl}^{[i]}\right]}^T \bar{\boldsymbol{\eta}}_{\left[\mathcal{C}_{cl}^{[i]}\right]} > \epsilon_2$ **then**
            Perform a dense BFGS update for the partial matrix $\bar{\mathbf{B}}_{\left[\mathcal{C}_{cl}^{[i]}\right]}$ induced by the clique $\mathcal{C}_{\left[\mathcal{C}_{cl}^{[i]}\right]}$ with $\boldsymbol{\delta}_{\left[\mathcal{C}_{cl}^{[i]}\right]}$ and $\bar{\boldsymbol{\eta}}_{\left[\mathcal{C}_{cl}^{[i]}\right]}$, as defined in Sect. 2.2.4.
        else
            Calculate $\epsilon_3 = \beta \tilde{\boldsymbol{\delta}}_{\left[\mathcal{C}_{cl}^{[i]}\right]}^T \bar{\mathbf{B}}_{\left[\mathcal{C}_{cl}^{[i]}\right]} \tilde{\boldsymbol{\delta}}_{\left[\mathcal{C}_{cl}^{[i]}\right]}$.
            if $\tilde{\boldsymbol{\delta}}_k^T \bar{\tilde{\boldsymbol{\eta}}}_k > \epsilon_2$ **then**
                Perform a dense BFGS update for the partial matrix $\bar{\mathbf{B}}_{\left[\mathcal{C}_{cl}^{[i]}\right]}$ induced by the clique $\mathcal{C}_{cl}^{[i]}$ with $\tilde{\boldsymbol{\delta}}_{\left[\mathcal{C}_{cl}^{[i]}\right]}$ and $\bar{\tilde{\boldsymbol{\eta}}}_{\left[\mathcal{C}_{cl}^{[i]}\right]}$, as defined in Sect. 2.2.4.
            end if
        end if
    end if
end for
Calculate the updated matrix $\mathbf{B} = \sum_{i=1}^{N_c} \mathbf{B}_{\mathcal{C}_{cl}^{[i]}}$.

Steps 1–14 must be performed once at the beginning of the NLP solution procedure and steps 16–32 describe the actual update routine.

9.4 Bibliographical Notes

We only introduced the fundamentals of sparse linear equation solvers, which we needed for the description of the algorithms in this chapter. By so doing, we skipped many other key features of these solvers, such as supernodes, delayed elimination, and multifrontal strategies. We refer the reader to the textbooks and works of Davis [7], Duff et al. [8], George and Liu [12], Hogg et al. [21], and Pissanetzky [33].

We presented the fundamentals of compressed Jacobian and Hessian calculation via seed matrices, but we skipped the more advanced techniques such as *bidirectional partitioning*, where the compressed Hessian $\mathbf{B} = \mathbf{W}^T\mathbf{HS}$ is calculated by two seed matrices $\mathbf{S}$ and $\mathbf{W}$. In many cases the number of function evaluations can be further reduced by bidirectional partitioning, but the algorithms become much more complex. A description of bidirectional partitioning can be found in Gebremedhin et al. [10] and the references in this paper.

We also skipped the *substitution methods*, which can also further reduce the number of necessary function evaluations by dropping the condition of structurally orthogonal columns and of symmetrically orthogonal columns. A drawback of these methods is that the evaluation of the uncompressed Jacobian or Hessian needs the solution of triangular systems of equations. Therefore, a trade-off between the reduction of function evaluations and the more complex uncompression algorithm must be found. A description of this technique can also be found in Gebremedhin et al. [10] and the references therein.

Furthermore, we only focused on the derivative calculation by finite differences and skipped the commonly used automatic differentiation and the more unusual calculation by complex steps, which are both very efficient, too.

The idea to use complex calculations for the numerical calculation of derivatives of real-valued functions goes back to J N Lyness [22], who used Cauchy's integral theorem for this purpose. An alternative algorithm, which relies on Fast Fourier Transformation, was described by Fornberg [9]. Unfortunately, both algorithms need many function evaluations for an accurate approximation of the numerical derivatives.

An efficient algorithm for the calculation of first-order derivatives, the complex step algorithm, was later described by Squire and Trapp [37] and Martins et al. [29]. The advantage of this method compared to finite differences is the avoidance of cancelation errors, which makes this method more accurate than finite differences. A drawback is, that the implementation of all elementary operations used in the function evaluation must be implemented for complex numbers, too. An extension for the calculation of higher-order derivatives was described by Lantoine et al. [25], which in turn needs the implementation of all elementary operations for multicomplex numbers.

Another efficient way to calculate derivatives is *automatic differentiation* (AD). The idea of AD goes back to Nolan [32] and Kahrimanian [23]. A FORTRAN implementation, called ADIFOR, is described in Bischof et al. [1, 2] and a C implementation, called ADOL-C, is described in Griewank et al. [19]. A detailed description of automatic differentiation is presented in the textbook of Griewank and Walther [18].

AD has the great advantage that the calculation of the derivatives is exact and not an approximation as for finite differences, and that the exploitation of sparsity can be automatically applied. To use AD all operations performed during the evaluation must be overloaded. AD can be implemented in two modes, the forward-mode and the reverse mode. In forward-mode the calculation tree is traversed from the leaves to the root (in order of the function evaluation) and the chain-rule for the internal derivatives is applied. In reverse-mode the calculation tree is stored in the memory and the derivative calculation is performed from the root to the leaves. The reverse-mode calculation requires in general less function evaluations, but needs more memory for the storage of the calculation tree, the so called tape.

Another advantage of AD schemes is that the propagation of index domains and nonlinear interaction domains can be simply added as a further operator overloading feature. A very fast reverse mode method to determine the sparsity pattern of a Hessian matrix named *edge pushing* was recently introduced by Gower and Mello [17].

References

1. Bischof CH, Carle A, Corliss GF, Griewank A, Hovland PD (1992) ADIFOR—generating derivative codes from Fortran programs. Sci Program 1(1):11–29
2. Bischof CH, Carle A, Khademi P, Mauer A (1996) ADIFOR 2.0: automatic differentiation of Fortran 77 programs. IEEE Comput Sci Eng 3(3):18–32
3. Cain BE (1980) Inertia theory. Linear Algebr Appl 30:211–240
4. Chabrillac Y, Crouzeix JP (1984) Definiteness and semidefiniteness of quadratic forms revisited. Linear Algebr Appl 63:283–292
5. Coleman TF, Moré JJ (1983) Estimation of sparse Jacobian matrices and graph coloring problems. SIAM J Numer Anal 20(1):187–209
6. Curtis AR, Powell MJD, Reid JK (1974) On the estimation of sparse Jacobian matrices. IMA J Appl Math 13(1):117–119
7. Davis T (2006) Direct methods for sparse linear systems. Fundamentals of algorithms, Society for Industrial and Applied Mathematics
8. Duff I, Erisman A, Reid J (1986) Direct methods for sparse matrices. Monographs on numerical analysis. Clarendon Press
9. Fornberg B (1981) Numerical differentiation of analytic functions. ACM Trans Math Soft 7(4):512–526. doi:10.1145/355972.355979
10. Gebremedhin AH, Manne F, Pothen A (2005) What color is your Jacobian? graph coloring for computing derivatives. SIAM REV 47:629–705
11. George A (1973) Nested dissection of a regular finite element mesh. SIAM J Numer Anal 10(2):345–363
12. George A, Liu JW (1981) Computer solution of large sparse positive definite. Prentice Hall Professional Technical Reference

13. George A, Liu JW (1989) The evolution of the minimum degree ordering algorithm. Siam Rev 31(1):1–19
14. Gilbert JR (1988) Some nested dissection order is nearly optimal. Inf Process Lett 26(6):325–328
15. Gilbert JR, Tarjan RE (1986) The analysis of a nested dissection algorithm. Numerische Mathematik 50(4):377–404
16. Gill PE, Murray W, Saunders MA (2002) SNOPT: an SQP algorithm for large-scale constrained optimization. SIAM J OPTIM 12(4):979–1006
17. Gower RM, Mello MP (2014) Computing the sparsity pattern of Hessians using automatic differentiation. ACM Trans Math Softw 40(2):10:1–10:15. doi:10.1145/2490254
18. Griewank A, Walther A (2008) Evaluating derivatives: principles and techniques of algorithmic differentiation, 2nd edn, Society for Industrial and Applied Mathematics. SIAM e-books
19. Griewank A, Juedes D, Mitev H, Utke J, Vogel O, Walther A (1999) ADOL-C: a package for the automatic differentiation of algorithms written in C/C++. Technical Report, Institute of Scientific Computing, Technical University Dresden, updated version of the paper published in. ACM Trans Math Softw 22(1996):131–167
20. Heggernes P, Eisestat S, Kumfert G, Pothen A (2001) The computational complexity of the minimum degree algorithm. Technical Report, DTIC Document
21. Hogg J, Hall J, Grothey A, Gondzio J (2010) High performance cholesky and symmetric indefinite factorizations with applications. University of Edinburgh
22. Lyness JN, Moler CB (1967) Numerical differentiation of analytic functions. SIAM J Numer Anal 4(2):202–210. http://www.jstor.org/stable/2949389
23. Kahrimanian HG (1953) Analytical differentiation by a digital computer. PhD thesis, Temple University
24. Karypis G, Kumar V (1998) A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM J Sci Comput 20(1):359–392
25. Lantoine G, Russell RP, Dargent T (2012) Using multicomplex variables for automatic computation of high-order derivatives. ACM Trans Math Softw 38(3):1–21. doi:10.1145/2168773.2168774
26. Lipton RJ, Rose DJ, Tarjan RE (1979) Generalized nested dissection. SIAM J Numer Anal 16(2):346–358
27. Liu JW (1985) Modification of the minimum-degree algorithm by multiple elimination. ACM Trans Math Softw (TOMS) 11(2):141–153
28. Markowitz HM (1957) The elimination form of the inverse and its application to linear programming. Manag Sci 3(3):255–269
29. Martins JRRA, Sturdza P, Alonso JJ (2003) The complex-step derivative approximation. ACM Trans Math Softw 29(3):245–262. doi:10.1145/838250.838251
30. Mathur R (2012) An analytical approach to computing step sizes for finite-difference derivatives. PhD thesis, University of Texas at Austin
31. McCormick ST (1983) Optimal approximation of sparse Hessians and its equivalence to a graph coloring problem. Math Program 26(2):153–171
32. Nolan JF (1953) Analytical differentiation on a digital computer. PhD thesis, Massachusetts Institute of Technology
33. Pissanetzky S (2014) Sparse Matrix Technology. Elsevier Science
34. Powell M, Toint PL (1979) On the estimation of sparse Hessian matrices. SIAM J Numer Anal 16(6):1060–1074
35. Rose DJ (1972) A graph-theoretic study of the numerical solution of sparse positive definite systems of linear equations. Graph theory and computing 183:217
36. Saad Y (2003) Iterative methods for sparse linear systems, 2nd edn. Society for Industrial and Applied Mathematics
37. Squire W, Trapp G (1998) Using complex variables to estimate derivatives of real functions. SIAM Rev 40(1):110–112. doi:10.1137/S003614459631241X
38. Tarjan RE, Yannakakis M (1984) Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. SIAM J Comput 13(3):566–579

39. Tewarson P, (1973) Sparse matrices, Mathematics in Science and Engineering. Elsevier Science
40. Tinney WF, Walker JW (1967) Direct solutions of sparse network equations by optimally ordered triangular factorization. Proc IEEE 55(11):1801–1809
41. Toint PL, Griewank A (1982) On the unconstrained optimization of partially separable objective functions. Nonlinear Optimization. Academic Press, London
42. Walther A (2008) Computing sparse hessians with automatic differentiation. ACM Trans Math Softw 34(1):1–15
43. Yannakakis M (1981) Computing the minimum fill-in is NP-complete. SIAM J Algebr Discret Methods 2(1):77–79

Part IV
Modeling of Hybrid Vehicles for Control

Chapter 10
Modeling Hybrid Vehicles as Switched Systems

10.1 Introduction

Increasing prices for crude oil, growing environmental concerns as well as stronger legislative requirements have caused a strong need for the development of new powertrain architectures for reducing the emissions of carbon-dioxide and other combustion products that are harmful to the environment and/or to human health. Hybrid vehicles constitute one of those developments that has attained considerable progress over the last years. In this type of powertrain architecture, an additional energy source and an additional converter are added to the conventional powertrain with fuel tank and internal combustion engine. In most cases, the secondary energy source will be a high-voltage battery and the additional converter will be at least one electrical motor/generator. In order to distinguish between other proposed hybrid configurations, e.g., mechanical, pneumatic (Colin et al. [12], Lee et al. [33]), hydraulic (Du et al. [14]), and fuel cell hybrids (Ehsani et al. [16]), hybrids with electrical configurations are more accurately called *hybrid electric vehicles* (HEV). If the vehicle can be recharged externally, using the local power grid, the vehicle can be referred to as a plug-in hybrid vehicle.

One major motivation to develop HEVs is the possibility of combining the advantages of pure electric vehicles and conventional combustion-based vehicles to enhance fuel economy. They profit from various possibilities including the following:

1. downsizing of the internal combustion engine;
2. recovering of some energy during deceleration phases instead of dissipating it as heat on the mechanical friction brake. Energy recovery during braking is termed *recuperation*;
3. optimization of the combustion engine's efficiency by using controlled energy distributions between the internal combustion engine and the motor/generator;
4. avoiding engine idling by turning off the combustion engine when no power is required; and

T.J. Böhme and B. Frank, *Hybrid Systems, Optimal Control and Hybrid Vehicles*,
Advances in Industrial Control, DOI 10.1007/978-3-319-51317-1_10

5. eliminating operation points at low engine efficiencies by using the motor/generator alternatively.

Certainly, not all of the possibilities above can be used at the same time. For instance, the level of downsizing is economically limited by the fact that extremely downsized engines with high efficiencies prevent costly hybridizations. A good compromise is therefore needed.

For the design and control of hybrid vehicles appropriate models which capture the main characteristics of the physics are important, usually modeled by a set of coupled differential equations, algebraic equations, static (nonlinear) characteristics, and the *degrees of freedom* (DOF) in the form of mode transitions. The DOF of a hybrid vehicle is essential for efficient control and give the control strategy several possibilities to reach the goal. This requires the incorporation of discrete decisions into the vehicle model in form of discrete equations or events. Combining these models results naturally in an abstract hybrid system model of hybrid vehicles and is the subject of this chapter. A broader view can be made by additionally including the network communication, discontinuities, and embedded software artifacts. This class of system is often classified as cyber-physical system (Lee and Seshia [35], Lee [34]) and results in more complex descriptions and thus in higher level of abstraction to manage the complexity.

The abstraction of hybrid vehicles as hybrid systems is twofold. First, it represents the main characteristics of interest and allows us to calibrate these models with information only obtained from a component's data-sheets and reproducible vehicle experiments. Second, these models have low complexity and are readily implementable in real-time solutions for online control (see Chaps. 11 and 12).

We start the modeling task by describing the longitudinal vehicle dynamics (Sect. 10.2) and the main mechatronic system components including the internal combustion engine, motor/generator, gearbox, and battery.

Then, we change from the component-level view to the system-level view. Several layouts exist that differ in the method of energy coupling from the thermal path and the electrical path. In Sect. 10.4, the most important hybrid configurations are discussed. The additional complexity of the powertrain brings along new DOF, which can be used to improve the overall efficiency of the powertrain and hence can be described as additional control inputs for the system.

Many automotive systems are in nature hybrid systems and a wealth of well-known examples are reported in the literature. For example, a vehicle with a manual gearbox equipped with four gears is modeled as a switched system in the book of Van Der Schaft et al. [61]. The vehicle movement requires externally forced switchings between the gears and thus discrete decisions are generated along the evolution of the two-dimensional continuous dynamics. A major component for *spark-ignition* (SI) engines is the electronic throttle which is considered as a hybrid system in Morari et al. [44]. An apparent example is the control of mode transitions in multimode combustion engines (Roelle et al. [52]). Automotive software usually contains many hysteresis elements to avoid toggling between discrete decisions. Such a discontinuity can be formulated as an hybrid automata with two locations and transition events

generated by internally forced switching as presented in Branicky [10] and Van Der Schaft et al. [61].

In Sect. 10.5, all hybrid vehicle models used for the following chapters are described as switched systems. Throughout this book we only consider well-behaved switched systems. That means, non-Zeno sequences which switch at most a finite number of times in the time range of interest $[t_0, t_f]$.

In Sect. 10.6, some important drive cycles for the design and control of (hybrid) vehicles are introduced.

10.2 Vehicle Dynamics

To model the longitudinal dynamic behavior of a vehicle behaving as a rigid body that moves along a sloped road we apply *Newton's second law of motion* in x-direction and two static equilibrium equations

$$\sum F_y(t) = 0 \tag{10.1}$$

$$\sum T(t) = 0. \tag{10.2}$$

The force (10.1) and torque (10.2) constraints imply that all vertically acting forces and torques are balanced.

The applied forces and torques in Fig. 10.1 with the constraints (10.1)–(10.2) results in three *ordinary differential equations* (ODE) and one algebraic equation

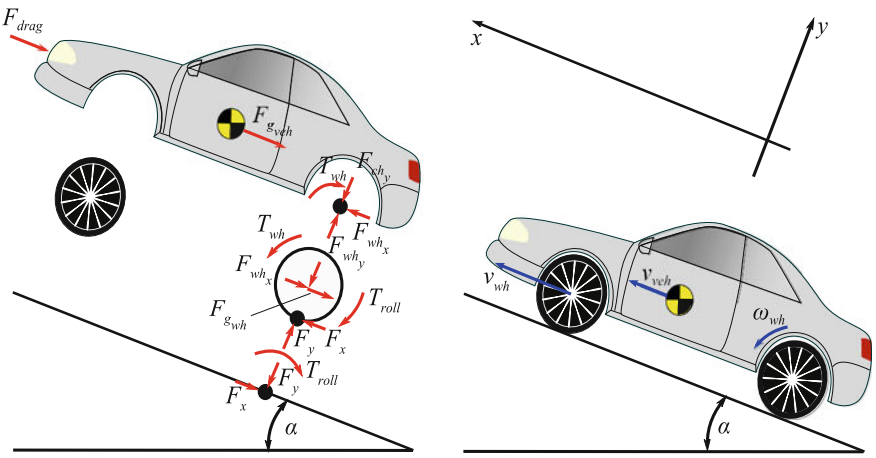


Fig. 10.1 Free body diagram of the applied forces and torques on the vehicle

$$m_{ch}\dot{v}_{veh}(t) = F_{wh_x}(t) - F_{g_{veh}}(t) - F_{drag}(t) \tag{10.3}$$
$$m_{wh}\dot{v}_{wh}(t) = F_x(t) - F_{wh_x}(t) - F_{g_{wh}}(t) \tag{10.4}$$
$$0 = F_y(t) - F_{ch_y}(t) - F_{wh_y}(t) \tag{10.5}$$
$$I_{wh}\dot{\omega}_{wh}(t) = T_{wh}(t) - T_{roll}(t) - r_{wh}F_x(t) \tag{10.6}$$

where r_{wh} is the wheel radius, $\omega_{wh}(\cdot)$ is the angular wheel speed, $F_{wh_x}(\cdot)$ is the interaction force between wheel and vehicle in longitudinal direction, $F_{g_{veh}}(\cdot)$ is the force due to gravity for the chassis, $F_{g_{wh}}(\cdot)$ is the force due to gravity for the wheel, $F_{drag}(\cdot)$ is the air-drag resistance force, $F_x(\cdot)$ is the traction force, $F_y(\cdot)$ is the normal force, I_{wh} is the wheel inertia, m_{ch} is the chassis mass, m_{wh} is the wheel mass, $T_{roll}(\cdot)$ is the roll friction torque acting on the point of wheel contact with the road, and $T_{wh}(\cdot)$ is the desired wheel torque.

The Forces in x-Direction

When the vehicle is turning the speed differs between the wheels. For simplicity, we assume that the speeds of the wheels are equal. Furthermore, we additionally assume that no slip occurs, i.e., wheel speed and the vehicle speed are identical

$$v(t) = v_{wh}(t) = v_{veh}(t). \tag{10.7}$$

Using (10.7), we can add (10.3)–(10.4) together and obtain the acceleration force

$$m\dot{v}(t) = F_x(t) - F_g(t) - F_{drag}(t) \tag{10.8}$$

with the total vehicle mass

$$m = m_{ch} + m_{wh}.$$

The desired wheel torque $T_{wh}(\cdot)$ and roll resistance torque $T_{roll}(\cdot)$ can be expressed as forces using

$$F_{wh}(t) = \frac{T_{wh}(t)}{r_{wh}} \tag{10.9}$$
$$F_{roll}(t) = \frac{T_{roll}(t)}{r_{wh}}. \tag{10.10}$$

Insertion of (10.9) and (10.10) into (10.6) yields the ODE

$$I_{wh}\dot{\omega}_{wh}(t) = r_{wh} \cdot (F_{wh}(t) - F_{roll}(t) - F_x(t)). \tag{10.11}$$

With the assumption that the wheels are not deformed while driving and have no slip, i.e.,

$$v(t) = r_{wh}\omega_{wh}(t)$$

we stiffly couple (10.8) and (10.11) and obtain one ODE for the longitudinal vehicle movement

$$\left(m + \frac{I_{wh}}{r_{wh}^2}\right)\dot{v}(t) = F_{wh}(t) - F_{drag}(t) - F_g(t) - F_{roll}(t) \tag{10.12}$$

which can be equivalently expressed as

$$\tilde{m}\dot{v}(t) = F_{wh}(t) - F_w(t) \tag{10.13}$$

where

$$F_w(t) = F_{drag}(t) + F_g(t) + F_{roll}(t)$$

is the total friction force and

$$\tilde{m} = m + \frac{I_{wh}}{r_{wh}^2}$$

is the effective translateral inertia. The terms of (10.12) are:

- $F_{drag}(\cdot)$, the air-drag, is approximated by

$$F_{drag}(t) = \frac{1}{2}\rho_{air} c_w A_{sec} v^2(t)$$

  where c_w is the drag coefficient, A_{sec} is the maximum vehicle cross-sectional area, and ρ_{air} is the specific density of air. The drag coefficient is dependent of the form of the vehicle and therefore on the outer surround-flow and the flow through the vehicle for engine cooling and air-conditioning purpose;
- $F_{roll}(\cdot)$, the rolling resistance, is approximated by

$$F_{roll}(t) = \frac{T_{roll}(t)}{r_{wh}} = F_y(t) c_r(v(t))$$

  where $c_r(\cdot)$ is the rolling coefficient which depends on tires and tire pressure and increases with the vehicle speed;
- $F_g(\cdot)$, the gravitational force, is given by

$$\begin{aligned} F_g(t) &= F_{g_{ch}}(t) + F_{g_{wh}}(t) \\ &= (m_{ch} + m_{wh})\, g \sin\alpha(t) \\ &= mg \sin\alpha(t) \end{aligned}$$

  where $\alpha(\cdot)$ is the slope of the road; and
- $F_a(\cdot)$, the acceleration resistance, is given by

$$F_a(t) = \left(m + \frac{I_{wh}}{r_{wh}^2}\right)\dot{v}(t).$$

For the sake of notational simplicity, the acceleration resistance can be rewritten as

$$F_a(t) = \gamma_m m \dot{v}(t)$$

where the ratio of effective translateral inertia to vehicle mass

$$\gamma_m = \frac{\tilde{m}}{m} = 1 + \frac{I_{wh}}{m r_{wh}^2}$$

is called the *mass factor*.

Remark 10.1 In practice, the air drag and rolling resistance forces are difficult to determine because of bearings friction, elastic tires, etc. It is therefore common to describe these terms by a polynomial of second degree of the vehicle speed as

$$F_{drag}(t) + F_{roll}(t) \approx a_2 v^2(t) + a_1 v(t) + a_0$$

where the coefficients a_0, a_1, and a_2 are usually determined by a *coast-down experiment* on a chassis dynometer. In the coast-down test, the fuel supply to the engine is turned off and the vehicle is allowed to slow under the effects of aerodynamic drag and rolling resistance.

The Forces in y-Direction

The normal force is given by

$$F_y(t) = m_{ch} g \cos\alpha(t) + m_{wh} g \cos\alpha(t) = mg \cos\alpha(t).$$

An important factor is the *x-direction friction coefficient* or *tire-road friction coefficient* described by the ratio

$$\mu_x = \frac{F_x}{F_y}$$

which describes how much traction force can be applied to the vehicle. This factor depends on the tire slip (Reza [50]).

10.3 Mechatronic Systems

In this section, we derive the main mechatronic systems employed in many different hybrid powertrains.

There are two basic approaches to modeling mechatronic systems. A theoretical model is based on principles like energy, mass, and impulse balance equations, constitutive equations, phenomenological and entropy balance equations of irreversible

processes, and coupling equations of process elements (Isermann [26]). A theoretical model contains the functional description between the physical data and its parameters and is therefore known as a *whitebox modeling* technique. In the opposite, *blackbox modeling* assumes that the process under consideration is totally or partial unknown and experimentally obtained data is available to describe the process with numerically driven parameters whose functional relationship with the physical data remains unknown. Between whitebox and blackbox there are uncountable many *graybox modeling* techniques.

Systems with dependence on space and time are usually governed by partial differential equations. For the case in which the space dependence can be neglected, the system reduces to ordinary differential equations with *lumped parameters*.

A typical approach to the design of computational models is to focus on the dynamics that have a significant influence on the system behavior. Depending on the modeling depth one obtains then static, quasi-static, and low-frequency dynamic models and high-frequency dynamic models. This results in widely varying model complexities. For instance, there is a wealth of combustion engine models that can be grouped into crank angle resolved, or crank angle averaged, or even quasi-static (de Jager et al. [27]). The same analogy applies to the electric machines, batteries, and so forth.

The proposed models of combustion engines, electric machines, gearboxes, and batteries in this section can be classified as *quasi-static graybox models with lumped parameters*. The models are derived with the focus on providing a simple representation which can be easily calibrated but remains physically interpretable as much as affordable. Please note that "affordable" needs to be read in the context of the goal of the model. We do not need all physical effects to be modeled by analytical or differential equations. This would yield models which are more complex than needed and hardly calibrateable. Some nonlinear effects can be easily mapped by polynomials or spline functions. Standard linear look-up tables are not considered for system modeling since they introduce nondifferentiability at the grid points due to the piecewise linear interpolation character which prevents computationally efficient optimization procedures, like sequential quadratic programming.

A dynamic combustion engine model for the purpose of emission control is proposed in Sect. 10.5.2. This model is air-charge and ignition angle resolved and requires several continuous states from other subsystems including discrete decisions. The combined model forms a complete (hybrid) vehicle and is therefore discussed later in this chapter.

10.3.1 Internal Combustion Engine

Hybrid vehicles may employ different types of *internal combustion engines* (ICEs) (or shorter engines) to exploit their fuel benefits but the piston engine is the most used engine type. Pistons are basically defined via the piston movement generated by the gas pressure and can be differentiated by the process management type. *Diesel*

or *Gasoline* engines are classified as open process whereas the *Stirling* engine is classified as closed process. Despite the high technical maturity of piston engine development, only a small part of the fuel energy is converted into mechanically effective work. The main sources of engine losses can be summarized as:

- exhaust gas (more than 50%);
- not ideal combustion;
- leakiness;
- heat loss;
- gas exchange; and
- engine friction, support drives, and side aggregates.

These losses vary with the engine type: gasoline or diesel. Especially under partial load, diesel engines show significantly lower losses than gasoline engines. The reader may refer to Kiencke and Nielsen [30] for more details.

We concentrate in this book only on hybrid vehicles equipped with gasoline engines. So it seems obvious to use the common terminology "internal combustion engines" with the shorthand ICE for gasoline engines.

10.3.1.1 Quasi-static Modeling of Engines

An analytical engine model is very hard to obtain. It is therefore common to describe the fuel consumption of the IC engines using maps. One important map is the *brake specific fuel consumption* (BSFC) map. This map can be determined by empirical procedures on an engine test-rig or can be computed by some software packages. For both procedures the maps obtained are only valid for warm engine operation.

The gasoline engine acts as a fuel converter to produce mechanical output power where its thermodynamic efficiency is defined by the ratio of mechanical output power to petrochemical power

$$\eta_{ice}(t) = \frac{\omega_{ice}(t) T_{ice}(t)}{\dot{Q}_{fuel}(t)} \tag{10.14}$$

where $\omega_{ice}(\cdot)$ is the engine angular speed, $T_{ice}(\cdot)$ is the engine output torque (effective torque), and $\dot{Q}_{fuel}(\cdot)$ is the enthalpy flow associated with the mass flow

$$\dot{m}_{fuel}(t) = \frac{\dot{Q}_{fuel}(t)}{H_l}$$

where H_l is the fuel's lower heating value with the unit MJ/kg, which describes the reaction enthalpy with respect to the matter used. The total fuel energy supplied is the time integral of the enthalpy flow

$$Q_{fuel}(t) = H_l \int_{t_0}^{t} \dot{m}_{fuel}(\tau)\, \mathrm{d}\tau.$$

The fuel consumption in liters of the ICE can be calculated by the differential equation

$$\dot{\beta}(t) = \gamma_f \cdot \zeta(t) \cdot \mathrm{bsfc}(T_{ice}(t), \omega_{ice}(t)) \cdot T_{ice}(t) \cdot \omega_{ice}(t) \tag{10.15}$$

$$\beta(t_0) = 0, \tag{10.16}$$

using a smooth brake specific fuel consumption map and a product of natural constants γ_f. $\zeta(\cdot)$ is a switch to model the fuel injection on/off command. The brake specific fuel consumption map bsfc : $\mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is represented by a smooth function, e.g., B-splines or tensor-product splines, of the arguments torque and speed and is usually temperature independent and only valid for warm engines, i.e., coolant water of $\vartheta_{cw}(t) = 90\,(^\circ\mathrm{C})$. If necessary, a warm-up correction factor can be introduced to account for warm-up losses (see therefore Sect. 10.3.1.2).

The engine efficiency is often depicted in the form of *equipotential* curves as shown in Fig. 10.2.

The consumption is mostly given as an amount of fuel per unit work g/kWh. This can be rewritten in SI-units as

$$1\left(\frac{\mathrm{g}}{\mathrm{kWh}}\right) = 1\left(\frac{\mathrm{g}}{3.6\mathrm{MJ}}\right) = 0.278\left(\frac{\mathrm{kg}}{\mathrm{GJ}}\right).$$

With the reaction enthalpy H_l, the fuel consumption can be directly converted into an engine efficiency $\eta_{ice}(\cdot)$. For example, for a gasoline engine the lower heating value of premium petrol is $H_l = 41.8$ (MJ/kg). The fuel consumption of the best operating point in Fig. 10.2 complies to an efficiency of

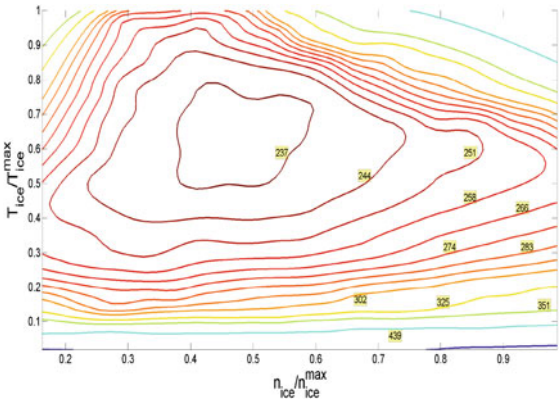


Fig. 10.2 Brake specific fuel consumption map $\left(\frac{\mathrm{g}}{\mathrm{kWh}}\right)$ for a direct-injected spark ignition engine

$$\eta_{ice} = \frac{1}{237\left(\frac{\text{g}}{\text{kWh}}\right) \cdot 41.8\left(\frac{\text{MJ}}{\text{kg}}\right)} = \frac{1}{66.72\left(\frac{\text{kg}}{\text{GJ}}\right) \cdot 41.8\left(\frac{\text{MJ}}{\text{kg}}\right)} = 0.363.$$

Remark 10.2 An IC engine is not able to turn on by itself. It needs a helper in the form of a belt-driven start/generator in mild hybrids or electric traction motors in full hybrids. In both cases, the engine is pulled to a certain engine speed (typically 500 rpm) where the fuel injection starts.

However, each turn on of the IC engine causes additional losses in terms of a higher amount of injected fuel to guarantee a stable engine burn-process for the first revolutions, larger applied electric torques to overcome static friction and compression work. The latter one can cause a large breakaway torque if the piston of one of the cylinders has to be completely moved to the upper cylinder position before the injection starts. These effects requires a deep understanding of the internal combustion process using first principles in mechanics and thermodynamics which is beyond the scope of this book. However, such effects can be simply modeled as an instantaneous engine start loss by accumulation of all loss contributions as will be presented in Sect. 10.5.1.

Operating Regime:

The feasible operating range of the engine is limited. Thus, it is the task of the control strategy that the following constraints for the speed

$$\omega_{ice}^{min} \leq \omega_{ice}(t) \leq \omega_{ice}^{max}$$

and the torque

$$T_{ice}^{min} \leq T_{ice}(t) \leq T_{ice}^{max}(\omega_{ice}(t))$$

are fulfilled.

The function of the upper engine torque limit $T_{ice}^{max}(\cdot)$ can be approximated by a polynomial of second-order (Reza [50]) or splines. Because of the instable combustion process of the ICE during low engine speeds a vehicle start-up from engine standstill ($\omega_{ice}(t) = 0$) is prohibited, which can taken into account with the following constraint

$$\zeta(t) = 0, \qquad \forall t \in [t_0, t_f] \text{ with } \omega_{ice}(t) < \omega_{ice}^{min}. \tag{10.17}$$

Willans Line Method:

An alternative approach to model the ICE efficiency is to use the *Willans line method* (see Wei [69], Guzzella and Sciarretta [20], and Zuurendonk [72]). This method

suggests a relationship between chemical input energy and mechanical output energy by the approximation

$$T_{ice}(t)\omega_{ice}(t) = e_{ice}(\omega_{ice}(t))\dot{Q}_{fuel}(t) - P_{ice,loss}(t) \tag{10.18}$$

where $e_{ice}(\cdot)$ is the efficiency of the thermodynamic energy conversion. However, the relationship of input and output power is not a straight line due to the strong nonlinear behavior of $e_{ice}(\cdot)$ and $P_{ice,loss}(\cdot)$. By dividing (10.18) on both sides by $\omega_{ice}(\cdot)$ and introducing the following relationships

$$\begin{aligned} p_{me}(t) &= \frac{4\pi}{V_d} \cdot T_{ice}(t) \\ p_{mf}(t) &= \frac{4\pi}{V_d} \cdot \frac{\dot{Q}_{fuel}(t)}{c_m(t)} \\ c_m(t) &= \frac{S}{\pi} \cdot \omega_{ice}(t) \end{aligned}$$

one yields the Willans relationship in the so-called *normalized variables*

$$p_{me}(t) = e_{ice}(c_m(t))p_{mf}(t) - p_{me0}(c_m(t))$$

where $p_{me}(\cdot)$ is the *brake mean effective pressure*, $p_{mf}(\cdot)$ is the *fuel mean effective pressure*, S is the piston stroke, and V_d is the total displacement. $p_{mf}(\cdot)$ is the pressure that an engine with a thermodynamic efficiency of 100% achieves by burning a mass $m_{fuel}(\cdot)$ with a lower heating value H_l and $p_{me0}(\cdot)$ summarizes all mechanical friction and gas exchange in the engine. If $p_{me0}(c_m(t)) = 0$ is assumed, one can redefine the thermodynamic efficiency (10.14) as

$$\eta_{ice}(t) = \frac{p_{me}(t)}{p_{mf}(t)}.$$

10.3.1.2 Extended Quasi-static Modeling of Engines with Coolant-Water and Three-Way Catalytic Converter Temperature

A simple model to account for the engine and catalytic converter warm-up is shown in Boehme et al. [8] which is based on the similarity principle. In order to account for the engine warm-up, the fuel differential equation (10.15) has been extended with a warm-up correction factor $CF_{fc}(\cdot)$ dependent on the coolant water. Then, the temperature-dependent fuel rate can easily be described by

$$\begin{aligned} \dot{\beta}(t) &= \gamma_f \cdot \zeta(t) \cdot CF_{fc}(\vartheta_{cw}(t)) \cdot \text{bsfc}(T_{ice}(t), \omega_{ice}(t)) \cdot T_{ice}(t) \cdot \omega_{ice}(t) \\ \beta(t_0) &= 0 \end{aligned} \tag{10.19}$$

where $\vartheta_{cw}(\cdot)$ is temperature of the coolant water.

By assuming that the relative cylinder charge $m_{cyl}(\cdot)$, the average temperature in the cylinder $\vartheta_{cyl}(\cdot)$, and the raw exhaust temperature $\vartheta_{exh}(\cdot)$ can be represented by maps, $m_{cyl}(T_{ice}(t), \omega_{ice}(t))$, $\vartheta_{cyl}(T_{ice}(t), \omega_{ice}(t), \zeta(t))$, and $\vartheta_{exh}(T_{ice}(t), \omega_{ice}(t), \zeta(t))$, respectively, and that no explicit modeling of the temperature losses to cylinder wall, exhaust valves, exhaust manifold, and turbine is necessary, the current temperature of the coolant water and the temperature of the *three-way catalytic converter* (TWC) $\vartheta_{twc}(\cdot)$ can be approximated by two nonlinear first-order ODEs

$$\begin{aligned}\dot{\vartheta}_{cw}(t) &= c_1 \cdot CF_{d\vartheta}(\vartheta_{cw}(t)) \cdot \dot{m}_{cyl}(T_{ice}(t), \omega_{ice}(t)) \cdot \left[\vartheta_{cyl}(T_{ice}(t), \omega_{ice}(t), \zeta(t)) - \vartheta_{cw}(t)\right] \\ &\quad -c_2 \cdot [\vartheta_{cw}(t) - \vartheta_{amb}(t)] \\ \dot{\vartheta}_{twc}(t) &= c_1 \cdot \dot{m}_{cyl}(T_{ice}(t), \omega_{ice}(t)) \cdot [\vartheta_{exh}(T_{ice}(t), \omega_{ice}(t), \zeta(t)) - \vartheta_{twc}(t)] \\ &\quad -c_2 \cdot [\vartheta_{twc}(t) - \vartheta_{amb}(t)]\end{aligned} \tag{10.20}$$

where $CF_{fc}(\cdot)$, $CF_{d\vartheta}(\cdot)$, and c_1 and c_2 are coefficients determined from measurements collected from ICE-test rigs.

It is absolutely necessary to heat-up the TWC to the light-off temperature at the beginning of the drive cycle. This requires that the ICE is switched on to produce a high amount of hot exhaust gases. The ECU is in charge of computing the light-off point with complex on-board function evaluations. In Sect. 11.2, an analytical model is presented to account for this on-board functions which in turn requires a high amount of computing resources. From a practical point of view, it is also possible to estimate a minimum duration for the engine to be switched on to achieve a safe heat-up of the TWC for the desired engine type and cooling system. If so, the additional constraint ensures

$$\zeta(t) = 1, \qquad t \in [t_{twc,0}, t_{twc,f}]$$

that the engine is switched on, where $t_{twc,0}$ and $t_{twc,f}$ are the start and the end time of the TWC heating, respectively.

10.3.2 Electric Machine

Depending on the vehicle concept, different electric machines (also known as *motor/generator* (MG)) may be used. Todays electric traction motors in hybrid or electric vehicles are mainly *permanent magnet synchronous machines* (PMSM) and *asynchronous machines* (ASM). The first one benefits from a high efficiency and good controllability, which make them highly attractive candidates for applications in (plug-in) HEVs or *battery electric vehicles* (BEV) (Chi [11]), whereas the latter will often be used if cost and robust operation is in focus in the vehicle application. A major drawback of ASM compared with PMSM is the higher required installation space in the vehicle. Switched reluctance motors are tested only in prototype cars and direct current machines are used only for special applications.

In this book, we consider only PMSMs because of their wide spread use in (plug-in) HEV and BEV applications. The functional layout consists of a stationary part, the stator, and a rotating part, the rotor. In general, electric power is supplied or outputted to and from the stator and mechanical power to or from the rotor. The operating regime T/n must be differentiated between *rated* and *nominal values*. Rated values, such as maximum torque T_{mg}^{max} and maximum power P_{mg}^{max}, can be permanently adjusted. Whereas, nominal values, such as nominal torque T_{mg}^{n} and nominal power P_{mg}^{n} are only adjusted for a short time. Consequently, it is important to distinguish between permanent nominal operating limits and transient rated operating limits for a proper design. It is therefore common for automotive applications to create several operating limits between both extremes to support low and high dynamic control loops.

Two basic operating regimes can be distinguished (see Fig. 10.3). First, the basic rotational speed range. This range is characterized by the fact, that for each speed starting from zero, the rated torque T_{mg}^{max} can be adjusted. If for constant T_{mg}^{max}, the rotational speed is increased, the mechanical power increases linearly until it reaches the rated power. The speed at this point is known as base speed

$$\omega_{mg}^{base} = \frac{P_{mg}^{max}}{T_{mg}^{max}}$$

and plays an important role in designing the MG properly for the required longitudinal vehicle dynamics. Second, in continuous operation, the rated power may not be exceeded. In order to achieve higher rotational speeds, the rated torque must be lowered

$$T_{mg}^{max}(\omega_{mg}(t)) = \frac{P_{mg}^{max}}{\omega_{mg}(t)}, \quad \omega_{mg}(t) > \omega_{mg}^{base}.$$

This area is referred to as the range of constant power and is achieved by field weakening. The field weakening range is mainly determined by the capability of the power inverter. When the speed of the MG increases, then the voltage applied to the motor must increase accordingly to counteract the speed proportional-induced *back electromagnetic force* in the stator windings. When the speed reaches the rated voltage, the voltage applied to the MG cannot be further increased to maintain the stator current for torque production. This characteristic can be influenced by varying the ratio of base speed over maximum speed, i.e.,

$$b_{mg} = \frac{\omega_{mg}^{base}}{\omega_{mg}^{max}},$$

where the ratio for PMSMs lies in an empirically determined region of

$$0.2 \leq b_{mg} \leq 0.6 \tag{10.21}$$

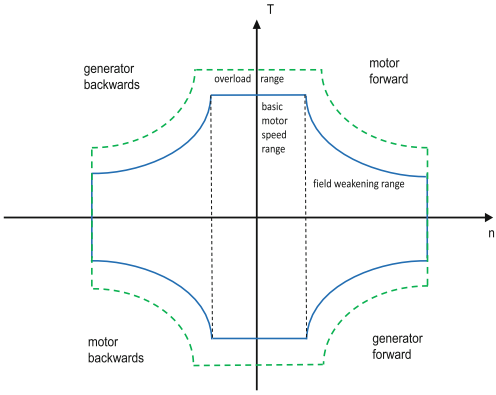


Fig. 10.3 Operating chart of an electric machine. Figure shows the four-quadrant operation of a MG, which means that the machine can accelerate and decelerate for each of the two rotational directions (forward and backward)

which is admissible (Boehme et al. [6]). This assumption has been experimentally validated by an analysis of a series of PMSMs.

Limiting parameters are temperature, mechanical stability, and service lifetime. If a machine is stressed beyond permitted values, thermal overload occurs owing to too high currents. For example, the winding insulation melts at approx. 180 °C.

Overload capabilities usually allow operation with a multiple factor of 1 up to 4 according to design and dimensioning. That means, the a MG can handle short-term loads four times its nominal load.

We found that P_{mg}^{max} and b_{mg} are design parameter to adjust the rated torque $T_{mg}^{max}(\cdot)$. The next two sections describe the efficiency of the MG within the rated torque limits. The first method uses a stationary efficiency map, the second method approaches this goal more physically.

10.3.2.1 Quasi-static Modeling of Motor/Generators

When a stationary efficiency map of the form $\eta_{mg}(\omega_{mg}(t), T_{mg}(t)) < 1, \ \forall \omega_{mg}, \ T_{mg}$ is available, the electrical power $P_1(\cdot)$ can be calculated as

$$P_1(t) = \frac{P_2(t)}{\eta_{mg}(\omega_{mg}(t), T_{mg}(t))} = \frac{T_{mg}(t)\omega_{mg}(t)}{\eta_{mg}(\omega_{mg}(t), T_{mg}(t))}, \qquad P_2(t) > 0 \quad (10.22)$$

$$P_1(t) = P_2(t)\eta_{mg}(\omega_{mg}(t), T_{mg}(t)), \qquad P_2(t) < 0 \quad (10.23)$$

where $P_2(\cdot)$ is the mechanical power. The efficiency map $\eta_{mg} : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is usually only valid for warm operation. The first case (10.22) is the motor operation, the second case (10.23) is the generator operation.

The efficiency map can be determined by empirical procedures on a test-rig or can be computed by some software packages, e.g., *finite element method* computing software. The latter requires an accurate physical model of the PMSM and a tight

control scheme. This in turn requires a good knowledge of the working principles and becomes only worthwhile if some deeper physical views are necessary.

Analogously to the IC engines, the Willans method can be used to describe the efficiency of a MG. For a motor/generator, this approach takes the form

$$P_2(t) = e_{mg}(\omega_{mg}(t))\, P_1(t) - P_0$$

where P_0 is the aggregated power loss after the energy conversion (e.g., friction, heat loss, etc.), and $e_{mg}(\cdot)$ is the indicated efficiency, i.e., the maximum efficiency when P_0 is zero. Rearranging yields

$$e_{mg}(\omega_{mg}(t)) = \frac{P_2(t) + P_0}{P_1(t)}$$

which is equivalent to

$$\eta_{mg}(\omega_{mg}(t), T_{mg}(t)) = e_{mg}(\omega_{mg}(t)) - \frac{P_0}{P_1(t)}.$$

10.3.2.2 Physics-Based Quasistatic Modeling of Motor/Generators

A more physically motivated model can be derived by considering the electrical dynamic equations in terms of phase variables in the stator frame (a-, b-, c-). These can be written in a convenient matrix form as

$$\mathbf{V}^s_{abc} = \begin{bmatrix} V^s_a(t) \\ V^s_b(t) \\ V^s_c(t) \end{bmatrix} = \operatorname{diag}\begin{bmatrix} R_s & R_s & R_s \end{bmatrix} \cdot \begin{bmatrix} I^s_a(t) \\ I^s_b(t) \\ I^s_c(t) \end{bmatrix} + \frac{\mathrm{d}\boldsymbol{\Psi}^s_{abc}}{\mathrm{d}t} \tag{10.24}$$

where $V^s_a(\cdot)$, $V^s_b(\cdot)$, and $V^s_c(\cdot)$ are the a,b,c terminal voltages, $I^s_a(\cdot)$, $I^s_b(\cdot)$, and $I^s_c(\cdot)$ are the a,b,c stator currents, R_s is the unique stator resistance of each phase of the stator winding. The flux linkage is expressed in terms of the stator currents as

$$\boldsymbol{\Psi}^s_{abc} = \begin{bmatrix} \Psi^s_a(t) \\ \Psi^s_b(t) \\ \Psi^s_c(t) \end{bmatrix} = \begin{pmatrix} L_{aa}(\theta(t)) & L_{ab}(\theta(t)) & L_{ac}(\theta(t)) \\ L_{ab}(\theta(t)) & L_{bb}(\theta(t)) & L_{bc}(\theta(t)) \\ L_{ac}(\theta(t)) & L_{bc}(\theta(t)) & L_{cc}(\theta(t)) \end{pmatrix} \begin{bmatrix} I^s_a(t) \\ I^s_b(t) \\ I^s_c(t) \end{bmatrix} + \begin{bmatrix} \Psi_{ma}(\theta(t)) \\ \Psi_{mb}(\theta(t)) \\ \Psi_{mc}(\theta(t)) \end{bmatrix} \tag{10.25}$$

where $\theta(\cdot)$ is the rotor angle in rad and $L_{aa}(\cdot)$, $L_{bb}(\cdot)$, and $L_{cc}(\cdot)$ are called *self inductances* of the stator, which can be written as

$$
\begin{aligned}
L_{aa}(\theta(t)) &= L_{0s} + L_{sl} + L_{2s} \cdot \cos(2\theta(t)) \\
L_{bb}(\theta(t)) &= L_{0s} + L_{sl} + L_{2s} \cdot \cos\left(2\theta(t) + \frac{2}{3}\pi \text{ (rad)}\right) \\
L_{cc}(\theta(t)) &= L_{0s} + L_{sl} + L_{2s} \cdot \cos\left(2\theta(t) - \frac{2}{3}\pi \text{ (rad)}\right)
\end{aligned}
$$

and $L_{ab}(\cdot)$, $L_{bc}(\cdot)$, and $L_{ac}(\cdot)$ are called *mutual inductances*

$$
\begin{aligned}
L_{ab}(\theta(t)) &= -\frac{1}{2} \cdot L_{0s} + L_{2s} \cdot \cos\left(2\theta(t) - \frac{2}{3}\pi \text{ (rad)}\right) \\
L_{bc}(\theta(t)) &= -\frac{1}{2} \cdot L_{0s} + L_{2s} \cdot \cos(2\theta(t)) \\
L_{ac}(\theta(t)) &= -\frac{1}{2} \cdot L_{0s} + L_{2s} \cdot \cos\left(2\theta(t) + \frac{2}{3}\pi \text{ (rad)}\right)
\end{aligned}
$$

where L_{sl} is the leakage in the stator, L_{0s} and L_{2s} are the magnetizing inductance components of the stator windings. The peak flux linkage $\Psi_{ma}(\cdot)$, $\Psi_{mb}(\cdot)$, and $\Psi_{mc}(\cdot)$ established by the rotor magnets are defined as

$$
\begin{aligned}
\Psi_{ma}(\theta(t)) &= \Psi_m \cdot \cos(\theta(t)) \\
\Psi_{mb}(\theta(t)) &= \Psi_m \cdot \cos\left(\theta(t) - \frac{2}{3}\pi(\text{rad})\right) \\
\Psi_{mc}(\theta(t)) &= \Psi_m \cdot \cos\left(\theta(t) + \frac{2}{3}\pi(\text{rad})\right).
\end{aligned}
$$

One can observe from Eq. (10.25) that the inductances are functions of the position of the rotor and (assuming that the rotor is spinning) are functions of the time. This means that the inductance parameters are constantly changing—space making the analysis of the machine very difficult in its present form. Two transformations commonly referred to as the Clark's transformation and the Park's transformation allow the stator voltages, currents, and inductances to be transferred into a reference frame where the inductances no longer vary with the position of the rotor. Applying both transformations together one obtains a (3×3)-transformation matrix

$$
\boldsymbol{T}(\theta(t))_{abc \to dq0} = \frac{2}{3} \cdot \begin{pmatrix} \cos(\theta(t)) & \cos\left(\theta(t) - \frac{2}{3}\pi(\text{rad})\right) & \cos\left(\theta(t) + \frac{2}{3}\pi(\text{rad})\right) \\ \sin(\theta(t)) & \sin\left(\theta(t) - \frac{2}{3}\pi(\text{rad})\right) & \sin\left(\theta(t) + \frac{2}{3}\pi(\text{rad})\right) \\ 1/2 & 1/2 & 1/2 \end{pmatrix}. \tag{10.26}
$$

Since the transformation is linear and $\boldsymbol{T}(\theta(t))_{abc \to dq0}$ has full rank, its inverse transformation exists and is defined as

$$\boldsymbol{T}(\theta(t))_{dq0\to abc} = \boldsymbol{T}(\theta(t))^{-1}_{abc\to dq0}$$
$$= \begin{pmatrix} \cos(\theta(t)) & \sin(\theta(t)) & 1 \\ \cos\left(\theta(t) - \frac{2}{3}\pi\,(\text{rad})\right) & \sin\left(\theta(t) - \frac{2}{3}\pi\,(\text{rad})\right) & 1 \\ \cos\left(\theta(t) + \frac{2}{3}\pi\,(\text{rad})\right) & \sin\left(\theta(t) + \frac{2}{3}\pi\,(\text{rad})\right) & 1 \end{pmatrix}.$$

Applying (10.26) to the system described by (10.24)–(10.25), we obtain a set of simple but coupled differential equations in rotor frame as

$$\frac{\mathrm{d}I_q^r}{\mathrm{d}t} = \frac{V_q^r(t) - R_s I_q^r(t) - \omega(t) L_d I_d^r(t) - \omega(t)\Psi_m}{L_q} \tag{10.27}$$
$$\frac{\mathrm{d}I_d^r}{\mathrm{d}t} = \frac{V_d^r(t) - R_s I_d^r(t) + \omega(t) L_q I_q^r(t)}{L_d} \tag{10.28}$$

where $\omega(\cdot)$ is the angular frequency of the electrical system (see Fig. 10.4). The mechanical angular speed on the shaft $\omega_{mg}(\cdot)$ is related to the electrical angular speed by the number of pole pairs p of the PMSM

$$\omega(t) = p\omega_{mg}(t).$$

Remark 10.3 Some comments on the transformation:

- the artificial currents $I_q^r(\cdot)$ and $I_d^r(\cdot)$ produce the same flux as the stator a,b,c currents;
- L_d is the longitudinal (d-axis) inductance, L_q is the lateral (q-axis) inductance;
- the flux linkage for the q-axis is $\Psi_q(t) = L_q I_q^r(t)$, the flux linkage for the d-axis is $\Psi_d(t) = L_d I_d^r(t) + \Psi_m$; and
- $-\omega(t)\Psi_q(t) = -\omega(t) L_q I_q^r(t)$ and $\omega(t)\Psi_d(t) = \omega(t) L_d I_d^r(t)$ are called speed voltages.

For stationary operation the coupled differential equations (10.27)–(10.28) reduce to

$$V_q^r(t) = R_s I_q^r(t) + \omega(t) L_d I_d^r(t) + \omega(t)\Psi_m \tag{10.29}$$
$$V_d^r(t) = R_s I_d^r(t) - \omega(t) L_q I_q^r(t). \tag{10.30}$$

Then, a balance of power at the two sides of a power inverter yields the electrical power as

$$P_1(t) = \frac{3}{2} \cdot \left(V_q^r(t) I_q^r(t) + V_d^r(t) I_d^r(t)\right) + P_{loss,c} \tag{10.31}$$

where $P_{loss,c}$ is the power loss of the inverter and the factor 3 accounts for the power terms of the a,b,c phases (factor 1/2 for the conversion of peak values to effective values). The first term in Eq. (10.31) represents the electrical power fed

to the electric machine. Ignoring the ohmic voltage drop in (10.29) and (10.30), the mechanical power at the shaft is given by

$$\frac{3}{2} \cdot \left[\left(\omega(t) L_d I_d^r(t) + \omega(t) \Psi_m \right) I_q^r(t) - \omega(t) L_q I_q^r(t) I_d^r(t) \right] = T_m(t) \omega_{mg}(t) = \frac{1}{p} T_m(t) \omega(t). \tag{10.32}$$

Rearranging (10.32) reveals a possible asymmetry of the lateral inductance to the longitudinal inductance

$$T_m(t) = \frac{3}{2} p \cdot \left[\Psi_m I_q^r(t) + \underbrace{\left(L_d - L_q \right) I_d^r(t) I_q^r(t)}_{\text{reluctance torque}} \right] \tag{10.33}$$

which generates an additional reluctance torque. Electric machines with such characteristics employ buried magnets and are known as *interior permanent magnet synchronous machine* (IPMSM). IPMSM drives are preferred in automotive HEV/BEV applications because of their high mechanical and thermal stability. For the case that the q-axis and d-axis inductance are equal (i.e., $L_q = L_d = L_s$) in (10.33) the torque generated at the rotor shaft reduces to

$$T_m(t) = \frac{3}{2} p \Psi_m I_q^r(t). \tag{10.34}$$

Electric machines with such characteristics employ surface mounted magnets and are called accordingly *surface permanent magnet synchronous machine*. Applying Newton's second law to the motor shaft yields

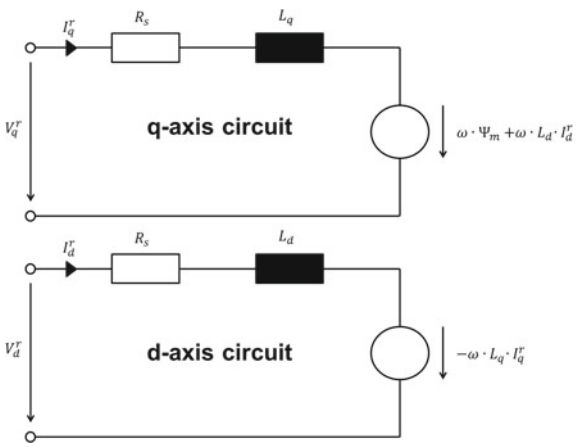


Fig. 10.4 Equivalent circuit model of the PMSM. The speed voltage term, $\omega(t) L_d I_d^r(t)$, appears in the $V_q^r(\cdot)$ equation and the speed voltage term, $\omega(t) L_q I_q^r(t)$, appears in the $V_d^r(\cdot)$ equation. Thus, the system equations (10.27) and (10.28) are linear but coupled

$$\frac{\mathrm{d}\omega_{mg}}{\mathrm{d}t} = \frac{T_m(t) - T_{mg}(t)}{I_{mg}},$$

where $T_{mg}(\cdot)$ is the load torque acting on the rotor shaft and I_{mg} is the moment of inertia of the motor. Using the steady-state equations (10.29) and (10.30) and the torque at the rotor shaft (10.34) in the quasi-stationary case, i.e., $T_m(t) = T_{mg}(t)$, the torque equation can be expressed as a function of the design parameters R_s, Ψ_m, and L_s

$$T_{mg}(t) = \frac{3}{2} p\Psi_m \cdot \frac{V_q^r(t)R_s - p\omega(t)L_s V_d^r(t) - p\omega(t)R_s\Psi_m}{R_s^2 + p^2 L_s^2 \omega^2(t)}. \tag{10.35}$$

We can use (10.22), (10.23), and (10.35) to derive a physical expression for the efficiency of the PM synchronous machine. However, this would require the knowledge of the q-axis and d-axis voltages, which in turn requires knowledge of the power inverter and the firing of the inverter switches (*gate turn-off thyristors* for high power levels or *insulated-gate bipolar transistor* for medium power levels). A special but common approach is to consider the so-called *common operating mode of the three-phase inverter* (Guzzella and Sciarretta [20]). Through this simplification the d-axis voltage $V_d^r(\cdot)$ is assumed to be zero which reduces (10.35) to

$$T_{mg}(t) = \frac{3}{2} p\Psi_m \cdot \left(\frac{V_q^r(t)R_s}{R_s^2 + L_s^2 p^2 \omega^2(t)} - \frac{p\omega(t)R_s\Psi_m}{R_s^2 + L_s^2 p^2 \omega^2(t)} \right) \tag{10.36}$$

and the efficiency to

$$\eta_{mg}(T_{mg}(t), \omega_{mg}(t)) = \left(1 + \frac{R_s}{\frac{3}{2} p^2 \Psi_m^2} \cdot \frac{T_{mg}(t)}{\omega_{mg}(t)} + \frac{L_s^2}{R_s} \cdot \frac{\omega_{mg}(t) T_{mg}(t)}{\frac{3}{2}\Psi_m^2} + \frac{P_{loss,c}}{T_{mg}(t)\omega_{mg}(t)} \right)^{-1}$$

(cf. Guzzella and Sciarretta [20]).

10.3.3 Gearbox

If an IC engine is directly coupled to the drivetrain, the engine's speed spread is neither sufficient to support the entire velocity range of a vehicle nor to provide a good dynamic responsibility. In order to achieve

- ecological fuel consumption;
- good dynamic responsibility; and

- targeted vehicle top speed,

passenger cars are equipped with manual, automatic, or power-split gearbox devices. Manual and automatic gearboxes are the dominate form of speed and torque converters in conventional vehicles and parallel hybrids.

The first two aims are contradictory since an ecological fuel consumption design leads in general to larger gear spreads compared with the high-traction design. This is mainly because the speed constraints imposed by the gear ratios might force the internal combustion engine to operate in some operating regimes which have low-engine efficiencies. Therefore, one can say that gearboxes serve as general speed and torque transformers. An important characteristic is the transmission ratio of the gearbox, which is defined by the speed ratio

$$i_{gbx}(t) = \frac{\omega_{gbx1}(t)}{\omega_{gbx2}(t)} \tag{10.37}$$

where $\omega_{gbx1}(\cdot)$ and $\omega_{gbx2}(\cdot)$ are the input and output speeds, respectively. The input power into the gearbox is then given by

$$P_{gbx1}(t) = \omega_{gbx1}(t) T_{gbx1}(t)$$

and the output power transmitted to the remaining driveline (reduction gear, differential, etc.) is given by

$$P_{gbx2}(t) = \omega_{gbx2}(t) T_{gbx2}(t).$$

If we assume that the gearbox works without losses and without inertia, that means

$$P_{gbx1}(t) = \omega_{gbx1}(t) T_{gbx1}(t) = P_{gbx2}(t) = \omega_{gbx2}(t) T_{gbx2}(t),$$

one can readily derive the torque ratio as

$$i_{gbx}(t) = \frac{T_{gbx2}(t)}{T_{gbx1}(t)}.$$

In practice, the gearbox realizes a power loss because of bearing and gear-wheel frictions. In order to account for this an efficiency representation can be found by using an analogy to the Willans description (10.18)

$$P_{gbx2}(t) = e_{gbx}(\omega_{gbx1}(t)) P_{gbx1}(t) - P_0 \tag{10.38}$$

where P_0 is a constant or speed dependent power loss term. If $P_0 = 0$ is assumed, one can redefine the efficiency (10.38) as

$$\eta_{gbx}(t) = \frac{P_{gbx2}(t)}{P_{gbx1}(t)}.$$

The usual efficiency of a gearbox with one fixed transmission ratio is about $\eta_{gbx} \approx 0.95 \ldots 0.98$. The overall efficiency of a multispeed gearbox depends on the engaged gear.

10.3.3.1 Automatic Gearboxes

There are two gearing devices between the clutch and the wheel: the gearbox and the final drive. The differential as final gearing device is ignored. The transmission characteristics of an automatic gearbox are defined by (10.37). The transmission characteristics of the final drive are defined by the ratio of gearbox output speed $\omega_{fd1}(t) = \omega_{gbx2}(t)$ to wheel speed $\omega_{fd2}(t) = \omega_{wh}(t)$ and thus

$$i_{fd} = \frac{\omega_{fd1}(t)}{\omega_{fd2}(t)}. \tag{10.39}$$

Then, the overall transmission ratio of the driveline is

$$i_t(t) = i_{gbx}(t) i_{fd} = \frac{\omega_{gbx1}(t)}{\omega_{wh}(t)} = \frac{r_{wh}\omega_{gbx1}(t)}{v(t)} \tag{10.40}$$

where $\omega_{gbx1} \in [\omega_{gbx1}^{min}, \omega_{gbx1}^{max}]$ is constrained to the allowed engine operating regime which imposes that the gearbox consists of a set of gear ratios rather than a simple gear ratio. The number of gears

$$\mathbf{i}_{gbx} := \left[i_{gbx,1}, i_{gbx,2}, \ldots, i_{gbx,N_{gbx}}\right]^T$$

depends on the fulfillment of the vehicle performance specified by an envelope curve.

The highest gear ratio is dedicated to the first gear and is determined by fulfillment of a desired creep velocity, e.g., $v_{crp}(t) < 7$ (km/h), and/or the maximum desired road inclination α_{max}. For example, using (10.40) the fulfillment of the creep velocity is given by

$$\mathbf{i}_{gbx}^{[1]} = \frac{r_{wh}\omega_{idle}}{i_{fd} v_{crp}}$$

where ω_{idle} is the desired engine idle speed. Whereas, the gear ratio $\mathbf{i}_{gbx}^{[1]}$ for fulfillment of the maximum hill-climbing ability can be derived by assuming an equilibrium ($a = 0$, i.e., no vehicle acceleration) at the maximum road inclination. We obtain then the torque balance with

$$\mathbf{i}_{gbx}^{[1]} T_{gbx1}^{max} \eta_{gbx}\left(T_{gbx1}^{max}\right) = r_{wh} \cdot \left(F_{roll} + F_g^{max}\right)$$

where the air drag is neglected. Rearranging yields the gear ratio to fulfill the maximum road inclination α^{max}

$$\mathbf{i}_{gbx}^{[1]} = \frac{r_{wh} \cdot \left(F_{roll} + F_g^{max}\right)}{T_{gbx1}^{max} \cdot \eta_{gbx}\left(T_{gbx1}^{max}\right)} = \frac{r_{wh} \cdot [mg\,(c_r + \sin\alpha^{max})]}{T_{gbx1}^{max} \cdot \eta_{gbx}\left(T_{gbx1}^{max}\right)}$$

where

$$T_{gbx1}^{max} = \max_{\forall \omega_{gbx1} \in \left[\omega_{gbx1}^{min}, \omega_{gbx1}^{max}\right]} \left[T_{ice}^{max}(\omega_{gbx1}) + T_{mg}^{max}(\omega_{gbx1})\right]$$

is the maximum gearbox input torque.

The lowest gear ratio is dedicated to the last gear and is usually determined by the fulfillment of the vehicle's top speed. However, in some cases, it might be beneficial to design the second-last gear to fulfill the vehicle's top speed, whereas the last gear can be freely chosen to support ecological driving.

The ratio of the first gear ratio to the last gear ratio

$$\varphi_s := \frac{\mathbf{i}_{gbx}^{[1]}}{\mathbf{i}_{gbx}^{[N_{gbx}]}}$$

defines the total gear spread, where N_{gbx} is the maximum number of gears.

The gear steps between two gears are defined by

$$\varphi_j := \frac{\mathbf{i}_{gbx}^{[j]} i_{fd}\omega_{wh}(t)}{\mathbf{i}_{gbx}^{[j+1]} i_{fd}\omega_{wh}(t)} = \frac{\mathbf{i}_{gbx}^{[j]}}{\mathbf{i}_{gbx}^{[j+1]}}, \quad j = 1, \ldots, (N_{gbx} - 1) \tag{10.41}$$

and can be determined by the

- geometrical design which yields gear ratios which are constant

$$\varphi_j := \frac{\mathbf{i}_{gbx}^{[j]}}{\mathbf{i}_{gbx}^{[j+1]}} = const, \quad j = 1, \ldots, (N_{gbx} - 1);$$

- progressive design which is characterized by an additional factor p multiplied to the geometrical gear step. This results in bigger steps for larger gear ratios but smaller steps for small gear ratios which is beneficial for gearbox designs with either less gears or smooth steps for small gear ratios. The progressive design is defined by

$$\mathbf{i}_{gbx}^{[j]} = \mathbf{i}_{gbx}^{[N_{gbx}]} \varphi_1^{(N_{gbx}-j)} p^{0.5(N_{gbx}-j)(N_{gbx}-j-1)}, \quad j = 1, \ldots, (N_{gbx} - 1)$$

where

$$\varphi_1 = \sqrt[(N_{gbx}-1)]{\frac{\varphi_s}{p^{0.5(N_{gbx}-1)(N_{gbx}-2)}}}$$

is the basic gear step and p is the progressiveness factor. For $p = 1$, the progressive design becomes the geometrical design; and

- free design (drive cyle based).

One can find the following value ranges for $\varphi_1 = 1.1 \ldots 1.7$ and $p = 1.05 \ldots 1.2$ in the literature. The latter design approach will be discussed in Chap. 13.

For all design procedures, the constraint known as the *gearbox stability condition* (Reza [50]) must be fulfilled. That means, a downshift at maximum engine torque should not cause the engine to operate above the maximum allowed speed. This implies a maximum gear step of

$$\varphi^{max} = \frac{\omega_{ice}^{max}}{\omega_{T_{ice}^{max}}}$$

where ω_{ice}^{max} is the maximum allowed engine speed and $\omega_{T_{ice}^{max}}$ is the engine speed at maximum engine torque.

10.3.3.2 Planetary Gearboxes

Compared with a conventional vehicle, whose transmission is a spur gearbox, the most different and important mechanical system used for a power-split HEV is a *planetary gearbox* (PG). A PG is more compact as a spur gearbox that includes normally five up to eight spur gears, while it can provide a wider range of speed ratio. Depending on the construction of PG, torques can be superimposed or split. The latter one is the key element for power-split hybrids. Figure 10.5 depicts two basic types of planetary gearboxes: a minus and a plus planetary gearbox.

Both types have three coaxial shafts and consists of four main components: a sun gear, several planetary gears (pinion gears), a planetary carrier gear, and a ring gear as shown in Fig. 10.5. Three or more planetary gears are held by the planetary

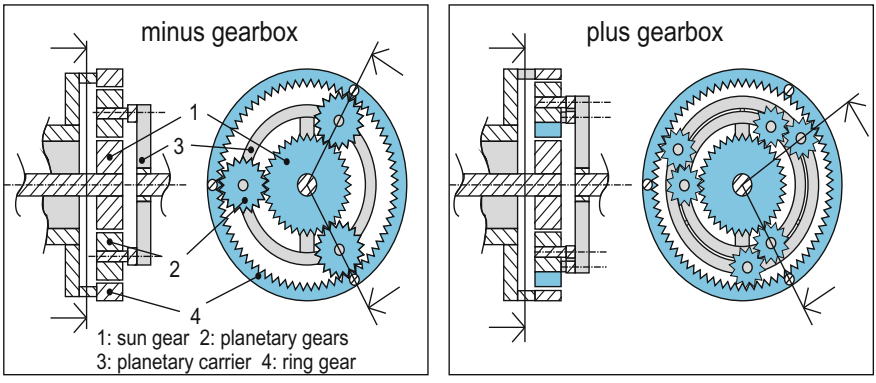


Fig. 10.5 Mechanical design of minus and plus planetary gearboxes

carrier, which rotates around the sun gear, while the shaft of the sun and ring gears are fixed. One can realize from Fig. 10.5 that a plus gearbox needs an additional set of planetary gears to reverse the speed direction and therefore requires more effort in the mechanical construction. One elementary characteristic is given, if the carrier shaft is held to zero speed by applying a brake or by a mechanical connection to the chassis. One then obtains the stationary gear ratio as

$$i_{sr} = \frac{r_r}{r_s}$$

where r_r and r_s are the radii of the ring and sun gear, respectively. In terms of a mechatronics-based characterization, it is more advantageous to express the stationary gear as the ratio of sun speed to ring speed

$$i_{sr} = \frac{\omega_s(t)}{\omega_r(t)}.$$

The stationary gear ratio is also an important parameter for calculating the distribution of rotational speed and torques. The relationship of speeds can be described by the Willis equation

$$0 = \omega_s(t) - i_{sr}\omega_r(t) - (1 - i_{sr})\omega_c(t)$$

and the ratio of torques (without consideration of inertia and losses)

$$\begin{aligned}
\frac{T_r(t)}{T_s(t)} &= -i_{sr} \\
\frac{T_c(t)}{T_s(t)} &= i_{sr} - 1 \\
\frac{T_c(t)}{T_r(t)} &= \frac{1}{i_{sr}} - 1
\end{aligned}$$

are constant, where $T_r(\cdot)$, $T_c(\cdot)$, and $T_s(\cdot)$ are the torques applying on the ring gear, carrier gear, and sun gear, respectively. The stationary gear ratio is negative for a minus planetary gearbox and positive for a plus planetary gearbox. For simplification and better clearness, the stationary gear ratios for minus and plus PGs are substituted with $i_{01} = -i_{sr}$ and $i_{01} = i_{sr}$, respectively. That means, for a minus PG

$$\begin{aligned}
0 &= \omega_s(t) + i_{01}\omega_r(t) - (1 + i_{01})\omega_c(t) \qquad (10.42) \\
\frac{T_r(t)}{T_s(t)} &= i_{01} \\
\frac{T_c(t)}{T_s(t)} &= -(1 + i_{01}) \\
\frac{T_c(t)}{T_r(t)} &= -\left(\frac{1}{i_{01}} + 1\right)
\end{aligned}$$

applies, whereas for a plus PG

$$\begin{aligned} 0 &= \omega_s(t) - i_{01}\omega_r(t) - (1 - i_{01})\omega_c(t) \\ \frac{T_r(t)}{T_s(t)} &= -i_{01} \\ \frac{T_c(t)}{T_s(t)} &= i_{01} - 1 \\ \frac{T_c(t)}{T_r(t)} &= \frac{1}{i_{01}} - 1 \end{aligned}$$

applies.

The dynamic relationships of the PG can be derived by applying the principle of angular momentum and Euler's second law to the rigid body of the PG with inertia of the carrier gear I_c, ring gear I_r, and sun gear I_s. For convenience, we multiply (10.42) by -1. Then, we obtain for a minus PG the following dynamics by

$$T_c(t) = I_c\dot{\omega}_c(t) + (1 + i_{01})\, T_i(t) \tag{10.43}$$

$$T_r(t) = I_r\dot{\omega}_r(t) - i_{01} T_i(t) \tag{10.44}$$

$$T_s(t) = I_s\dot{\omega}_s(t) - T_i(t) \tag{10.45}$$

where

$$T_i(t) = F(t)r_s \tag{10.46}$$

is the internal torque of the PG and r_s is the radius of the planetary carrier. The evolution of these first order ODEs are subject to the static Willis equation (10.42). One obtains differential-algebraic equations describing the system dynamics which can be easily represented as

$$\begin{bmatrix} T_c(t) \\ T_r(t) \\ T_s(t) \\ 0 \end{bmatrix} = \begin{pmatrix} I_c & 0 & 0 & (1 + i_{01}) \\ 0 & I_r & 0 & -i_{01} \\ 0 & 0 & I_s & -1 \\ (1 + i_{01}) & -i_{01} & -1 & 0 \end{pmatrix} \begin{bmatrix} \dot{\omega}_c(t) \\ \dot{\omega}_r(t) \\ \dot{\omega}_s(t) \\ T_i(t) \end{bmatrix}.$$

10.3.4 Clutch

Clutches are coupling elements of the powertrain. When the clutch is engaged and no slip is assumed then the input angular speed $\omega_{clth1}(\cdot)$ and output angular speed $\omega_{clth2}(\cdot)$ are identical, i.e.,

$$\omega_{clth1}(t) = \omega_{clth2}(t).$$

Real electro-mechanical clutches have a negative loss torque due to friction

$$T_{clth1}(t) = T_{clth2}(t) + T_{clth,loss}(t)$$

and consume electrical power for the operation of the actuator, where $T_{clth1}(\cdot)$ and $T_{clth2}(\cdot)$ are the input and output torque of the clutch, respectively. These effects are not easy to model and can be abstracted to an energy loss. Therefore, for the purpose of this book, the clutch operation is simplified as instantaneous switchings between opened and closed and assume an energy loss for each switching operation. Readers interested in modeling the clutch are recommended to consult Koprubasi [31] and Beck et al. [4].

This model abstraction has already been encountered by the engine start problematic and has advantages and disadvantageous. The major advantage of this method is to make the modeling task much easier. This is especially beneficial if the physical process under consideration is not really well understood or the modeling effort is extraordinary high. But the designer has to keep in mind that a physical system does not exhibit jumps in reality. This is just a simplification and introduces at the same time much more complexity in the control task. Why? The instantaneous energy loss must be accounted on some or all continuous-valued states of the system which causes discontinuities. In the previous chapters, we have already seen that discontinuities in continuous-valued states of hybrid systems causes some serious control problems. The designer of the hybrid system model must decide which problem, modeling or controlling, causes less efforts.

10.3.5 Battery

Rechargeable batteries (also known as secondary batteries) are devices which provide a possibility of transforming chemical energy into electrical energy and vice versa by chemical reaction processes. This feature makes batteries portable energy sources for many automotive applications. Traction batteries are a key element for BEV and HEV design which affects the overall cost-benefit of these vehicles and come in different technologies, sizes, and shapes and can be categorized by qualities such as specific energy, specific power, capacity, voltage, and chemistry. It is therefore important that traction batteries must have certain attributes like high specific power, high specific energy, long calendar and cycle lifetime, high reliability, and high robustness to be good candidates.

Typically, in a battery several electrochemical cells are connected in series to provide fixed voltage. Main elements of each battery cell are two electrodes, cathode and anode, and an electrolyte, which separates the two electrodes. At the electrodes chemical reactions occur for gain and loss of electrons. During the discharge, a reductant (R) donates m electrons at the anode, which are released to the connected circuit. These m electrons are then accepted by an oxidant (O) at the cathode. The first electrochemical process is called an *oxidation reaction*, the last one is called an *reduction reaction*. The general reaction schemata is

$$\begin{cases} \mathrm{R} \rightarrow \mathrm{R}^{+} + me^{-}, & \text{oxidation reaction} \\ \mathrm{O} + me^{-} \rightarrow \mathrm{O}^{-}, & \text{reduction reaction} \\ \mathrm{R} + \mathrm{O} \rightarrow \mathrm{R}^{+} + \mathrm{O}^{-} & \text{redox reaction .} \end{cases} \tag{10.47}$$

The charge separation will continue until an equilibrium condition is reached.

We may observe from (10.47) that battery modeling is a quite complex task. This has led in the literature to many different complex battery models specific to application domains. The main used model types are

- empirical models;
- equivalent circuit models;
- physics-based models (electro-chemical models); and
- high level stochastic models.

Empirical models are the simplest ones and describe an observed relationship and usually does not take any physical property of the investigated battery into account. Whereas, physics-based models provide insight into the physical process during cell operation. These models can easily grow in complexity and consist of a dozens of parameters. Stochastic battery models are based on Markov processes, e.g., for predicting battery life span (Rao et al. [49]). However, such models rely on huge datasets which should represent the characteristics of interest accurately and are, therefore, only applicable if many experiments can be performed in a reproducible manner.

A good compromise for the complexity of the problem set discussed in this book are equivalent circuit analogies consisting of ideal voltage source, resistors, and capacitors. Similar modeling assumptions are made frequently in the literature, among them Hu et al. [25] and Stockar et al. [60].

10.3.5.1 Quasi-static Modeling of Batteries

A brief definition of important battery parameters:

Battery Capacity

The battery capacity Q_{bat}, usually expressed in Ah, indicates the amount of charge that can be drawn from a fully charged battery until it gets fully discharged. This is defined by

$$I_{1h} = \frac{Q_{bat}}{1\ (\mathrm{h})}$$

which provides the amount of current drawn from the battery that completely discharges in 1 h.

C-rate

This parameter is used to show the amount of current used for charging the battery

$$c(t) = \frac{I_{bat}(t)}{I_{1h}}.$$

State of Charge

State of charge $\xi(\cdot)$ is the ratio of available charge $Q(\cdot)$ of a battery to the rated capacity of the battery

$$\xi(t) = \frac{Q(t)}{Q_{bat}}$$

where $Q(\cdot)$ is difficult to measure directly, but the variation of the charge related to the current $I_{bat}(\cdot)$ can be used

$$\dot{Q}(t) = I_{bat}(t).$$

This leads to the well-known *Coulomb counting* method

$$Q(t) = \int_{t_0}^{t} I_{bat}(\tau)\, \mathrm{d}\tau + Q(t_0)$$

which is reliable as long as the current measurement is accurate (Guzzella and Sciarretta [20]). More advanced techniques use estimation methods such as the Extended Kalman Filter to determine the state of charge.

Depth of Discharge

The depth of discharge DoD$(\cdot)$ is defined as

$$\mathrm{DoD}(t) = 1 - \xi(t).$$

This quantity is usually used by the battery manufacturer according to lifetime issues, e.g., to recommend that DoD $= 0.7$ should not be exceeded.

Equivalent Circuit Model

A fairly simple model of a battery is obtained using an ideal voltage source and an ohmic internal resistance (see Fig. 10.6). Considering the power losses caused by the battery's internal resistance R_{bat} and applying Kirchhoff's law, one obtains the battery power $P_{bat}(\cdot)$ related to the battery current $I_{bat}(\cdot)$ through the following power balance equation

$$V_{oc}(\xi(t))I_{bat}(t) + R_{bat}I_{bat}^2(t) = P_{bat}(t), \tag{10.48}$$

where $V_{oc}(\cdot)$ is the open circuit voltage. The sign of the battery power indicates battery discharges for $P_{bat}(t) < 0$ and battery charges for $P_{bat}(t) > 0$.

The battery power is the net power consisting of the electrical power $P_{mg}(\cdot)$ of the MG (which can be more than one MG) and the power $P_{aux}(\cdot)$ required to supply the electrical on-board system and is given by

$$P_{bat}(t) = -P_{mg}(T_{mg}(t), \omega_{mg}(t)) - P_{aux}(t). \tag{10.49}$$

Solving Eq. (10.48) for the battery current $I_{bat}(\cdot)$ yields

$$I_{bat}(t) = \frac{-V_{oc}(\xi(t)) + \sqrt{V_{oc}^2(\xi(t)) + 4R_{bat}P_{bat}(t)}}{2R_{bat}}. \tag{10.50}$$

The differential equation for $\xi(\cdot)$ can then be written using (10.50) as

$$\dot{\xi}(t) = \frac{1}{Q_{bat}} \cdot I_{bat}(\xi(t), u(t)) \tag{10.51}$$

$$\xi(t_0) = \xi_0. \tag{10.52}$$

Open Circuit Voltage

The *open circuit voltage* (OCV), $V_{oc}(\cdot)$, is the voltage between the battery terminals when no load is applied. This voltage is a function of the battery's state of charge and can be modeled using physical principles or empirical approaches. For the latter, a general formula for the OCV is given by Weng et al. [70]:

$$V_{oc}(\xi(t)) = K_0 + K_1\xi(t) + \frac{K_2}{\xi(t)} + K_3 \ln(\xi(t)) + K_4 \ln(1 - \xi(t)). \tag{10.53}$$

Using (10.53), two popular parameterizations are common.

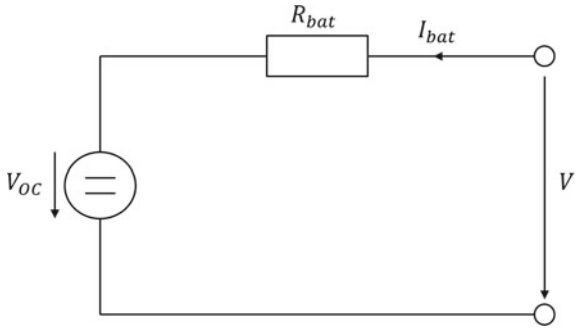


Fig. 10.6 Simple equivalent circuit model of a battery. This equivalent circuit model has been applied mainly for lead-acid batteries, nickel-cadmium, nickel-metal hydride, and modern lithium-ion batteries (Guzzella and Sciarretta [20])

Affine relationship:

$$V_{oc}(\xi(t)) = K_0 + K_1\xi(t)$$

where K_0 and K_1 depend only on the battery construction and the number of cells (Guzzella and Sciarretta [20]).

Nernst equation:

$$V_{oc}(\xi(t)) = K_0 + K_3 \ln(\xi(t)) + K_4 \ln(1 - \xi(t))$$

is derived from the *Nernst equation* (Pichler and Cifrain [47]), where the parameters can be obtained by comparison of the coefficients, i.e., $K_0 = V_0$ and $K_3 = K_4 = \frac{R \cdot T}{n_e \cdot F}$. This yields

$$V_{oc}(\xi(t)) = V_0 + \frac{R \cdot T}{n_e \cdot F} \ln\left(\frac{\xi(t)}{1 - \xi(t)}\right)$$

where R is the universal gas constant (8.315J/mol K), V_0 is the standard cell potential in volts, F is the Faraday constant ($\approx$96485 C/mol), and n_e is the number of free electrons.

Mapping:
Alternatively, $V_{oc}(\cdot)$ can be tabulated or mapped by a spline as a function of the state of charge.

Some battery types exhibit a considerable hysteresis behavior in the OCV. That means in other words, that different OCV curves apply for charge and discharge mode. These effects are particularly significant in batteries of *nickel-metal hydride* type (Verbrugge and Tate [62]). However, for modern battery types, e.g., lithium-ion, the hysteresis effect can usually be neglected, since they exhibit only small hysteresis.

Battery Resistance:

The internal ohmic battery resistance R_{bat} depends on the state of charge and the temperature and can be represented by a spline function. In many cases a constant approximation is sufficient for optimization of the vehicle's energy consumption.

Battery Efficiency

The battery efficiency is defined on the basis of a charge/discharge cycle as the ratio of discharged energy to the energy, which is necessary to recharge the battery with a current of the same intensity. Assuming a steady-state equivalent circuit model, the energy over a given time interval $[t_0 = 0, t_f]$ can be calculated as

$$E = \int_0^{t_f} P_2(t)\,\mathrm{d}t = \int_0^{t_f} \left[\tilde{V}_{oc} + R_{bat} I_{bat}(t)\right] \cdot I_{bat}(t)\,\mathrm{d}t$$
$$= t_f \cdot \left[\tilde{V}_{oc} + R_{bat} I_{bat}(t_f)\right] \cdot I_{bat}(t_f),$$

where $\tilde{V}_{oc}$ and R_{bat} are constant. For the reason of better transparency, a constant open circuit voltage and a constant battery current are marked with a tilde. Then, the discharge energy E_d with a constant discharging current $\tilde{I}_{bat} < 0$ can be calculated by

$$E_d = \int_0^{t_f} \left|\tilde{P}_d\right| \,\mathrm{d}t = \int_0^{t_f} \left[\tilde{V}_{oc} - R_{bat}\left|\tilde{I}_{bat}\right|\right] \cdot \left|\tilde{I}_{bat}\right| \,\mathrm{d}t$$
$$= t_f \cdot \left[\tilde{V}_{oc} - R_{bat}\left|\tilde{I}_{bat}\right|\right] \cdot \left|\tilde{I}_{bat}\right|.$$

The discharging current $\tilde{I}_{bat}$ has to be limited such that the term $\tilde{V}_{oc} - R_{bat}\left|\tilde{I}_{bat}\right|$ remains positive. Recharging the battery with a current of the same intensity, i.e., $\tilde{I}_{bat} \equiv \left|\tilde{I}_{bat}\right|$, yields

$$E_c = \int_0^{t_f} \tilde{P}_c \,\mathrm{d}t = \int_0^{t_f} \left[\tilde{V}_{oc} + R_{bat}\left|\tilde{I}_{bat}\right|\right] \cdot \left|\tilde{I}_{bat}\right| \,\mathrm{d}t$$
$$= t_f \cdot \left[\tilde{V}_{oc} + R_{bat}\left|\tilde{I}_{bat}\right|\right] \cdot \left|\tilde{I}_{bat}\right|.$$

The ratio of E_d over E_c is the battery efficiency

$$\eta_{bat,g} = \frac{E_d}{E_c} = \frac{\tilde{V}_{oc} - R_{bat}\left|\tilde{I}_{bat}\right|}{\tilde{V}_{oc} + R_{bat}\left|\tilde{I}_{bat}\right|}. \tag{10.54}$$

Equation (10.54) is known as *global battery efficiency*. As shown in Guzzella and Sciarretta [20], the *global battery efficiency* is based on a full charge/discharge cycle and is therefore cycle-pattern dependent, whereas the *local battery efficiency* is based on instantaneous power evaluations which yields a similar relationship

$$\eta_{bat,l}(t) = \frac{|P_d(t)|}{P_c(t)} = \frac{V_{oc}(t) - R_{bat}\,|I_{bat}(t)|}{V_{oc}(t) + R_{bat}\,|I_{bat}(t)|}. \tag{10.55}$$

The difference is that in (10.55) the open circuit voltage $V_{oc}(\cdot)$ and the battery current $I_{bat}(\cdot)$ depends on time.

10.3.5.2 Dynamic Modeling of Batteries

Dynamic models of practical use are the *Randles* and *Thevenin models* (see Figs. 10.7 and 10.8). The additional dynamical attribute modifies the quasi-static model with a nonohmic voltage drop. The added passive circuit drives two parallel branches, in which a capacitive current and a charge-transfer current flow. The capacitive current flows across a double-layer capacitor $C_{bat,1}$, which describes the effects of the charge accumulation/separation that occurs at the interface between electrodes and electrolyte. The charge-transfer current is caused by charge separation. The dynamic equations for this circuit are derived from Kirchhoff's voltage and current laws which yield

$$\begin{aligned} V(t) &= V_{oc}(\xi(t)) + R_{bat,0} I_{bat}(t) + V_{RC1}(t) \\ I_{bat}(t) &= I_{C1}(t) + I_{R1}(t) = C_{bat,1} \cdot \frac{\mathrm{d}V_{RC1}}{\mathrm{d}t} + \frac{V_{RC1}}{R_{bat,1}} \end{aligned} \tag{10.56}$$

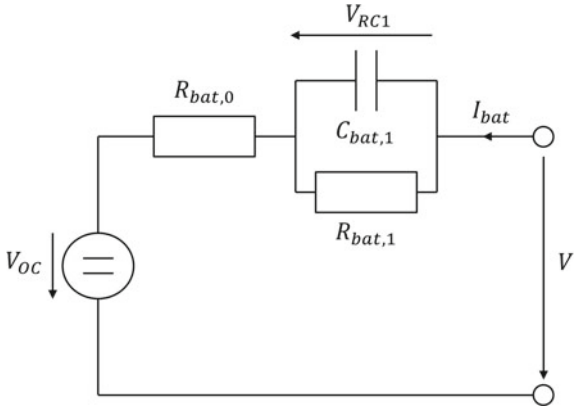


Fig. 10.7 Randles model for batteries

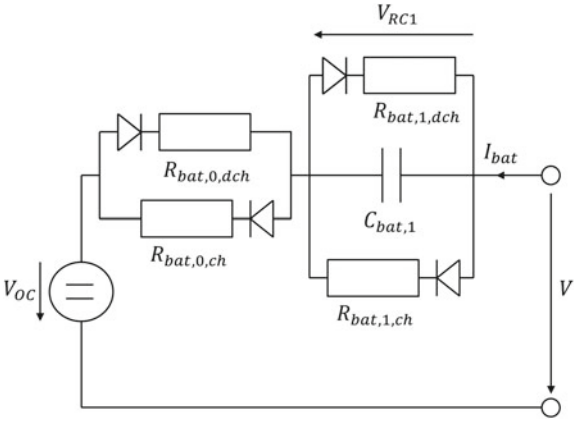


Fig. 10.8 Extended Randles model for batteries with considerable hysteresis effect. The diodes can be considered as switches for controlling the current flow through the respective resistance

where $V_{RC1}(\cdot)$ is the nonohmic over-potential. Insertion of $I_{bat}(\cdot)$ into (10.56) and rearranging yields the differential equation as

$$R_{bat,0}C_{bat,1}\frac{\mathrm{d}V_{RC1}}{\mathrm{d}t} = V(t) - V_{RC1}(t)\cdot\left(1+\frac{R_{bat,0}}{R_{bat,1}}\right) - V_{oc}(\xi(t))$$

where $R_{bat,0}C_{bat,1}$ has the unit of time and is the time constant of the RC-circuit.

At steady-state, the capacitor can be considered as a circuit-cut, i.e.,

$$V(t) = V_{oc}(\xi(t)) + \left(R_{bat,0} + R_{bat,1}\right)\cdot I_{bat}(t)$$

with $R_{bat} = R_{bat,0} + R_{bat,1}$.

Nonignorable hysteresis effects may limit a uniform modeling of the battery and require some treatment. In practice, some *ad hoc* procedures may be applied to capture these nonlinear effects. A common way is to modify the Randles model with some ideal diodes to reflect the fact that the resistance in the ohmic and nonohmic circuits are different during charging and discharging (Hu et al. [25]). This requires some additional smooth conditions for transition between these maps.

10.4 Hybrid Vehicle Configurations

The main topics of this section are the three main types of hybrid electric vehicles:

- **parallel hybrid**: both the combustion engine and the motor/generator are mechanically coupled with the wheels;
- **power-split hybrid**: a combination of parallel and serial features; and
- **serial hybrid**: the electric machine serves as prime mover and drives the vehicle and the IC engine provides the traction power.

These configurations employ different-sized prime movers, different functional layouts, and different drive modes. The right choice of one of these HEV configurations depends on several crucial factors including the following:

- type of the application;
- cost and weight; and
- expectations of the targeted customers.

Some important drive modes are:

- **start/stop**: turn on and off the IC engine;
- **load-level-shifting**: shifting of the operating points of the IC engine towards its best efficiency area;
- **recuperation**: recovery of the vehicle's kinetic energy;
- **boost**: torque assistance if the desired torque exceeds the maximum ICE torque $T_{ice}^{max}(\omega_{ice})$;

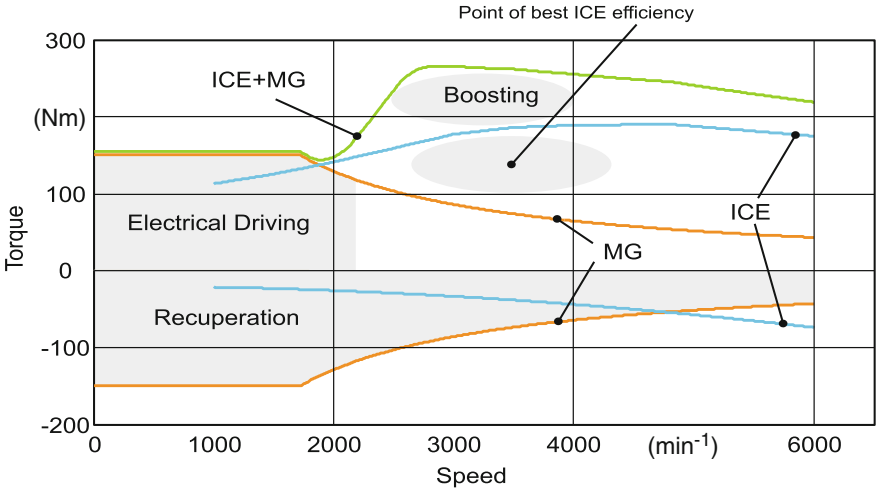


Fig. 10.9 Drive modes of hybrid vehicles

- **zero-emission driving**: propelling of the vehicle with the electric traction motor only; and
- **external charging**: recharging of the battery using a net power grid.

Figure 10.9 visualize the areas of the different drive modes in a speed-torque diagram.

Baumann et al. [3] proposed a measure of hybridization as the *degree of hybridization* (DOH) which is defined as the ratio of

$$\text{DOH} = 1 - \frac{\left|P_{mg}^{max} - P_{ice}^{max}\right|}{P_{mg}^{max} + P_{ice}^{max}}, \tag{10.57}$$

where P_{mg}^{max} is the maximum power of the motor/generator and P_{ice}^{max} is the maximum power of the IC engine. The value range of this function is $0 \leq \text{DOH} \leq 1$. That means, a completely combustion-based vehicle corresponds to $\text{DOH} = 0$ and a completely electric vehicle corresponds to $\text{DOH} = 0$ as well. A fully hybridized design is achieved according to (10.57) if the maximum MG power is equal to the maximum ICE power.

Combustion-based concepts with a low DOH are often regarded as micro or mild hybrids, even though their architecture is the same as that of full parallel hybrids. Electric-based concepts with a low DOH are often regarded as range extenders.

This simple metric can help the designer to classify early vehicle designs to the functionality group (shown in Fig. 10.10) and therefore provides indications as to which controls should be emphasized. For instant, if the HEV is ICE dominated then the main energy flow comes from the IC engine and the focus on control design should be an optimal chemical–mechanical energy flow. Whereas, the design of HEVs with

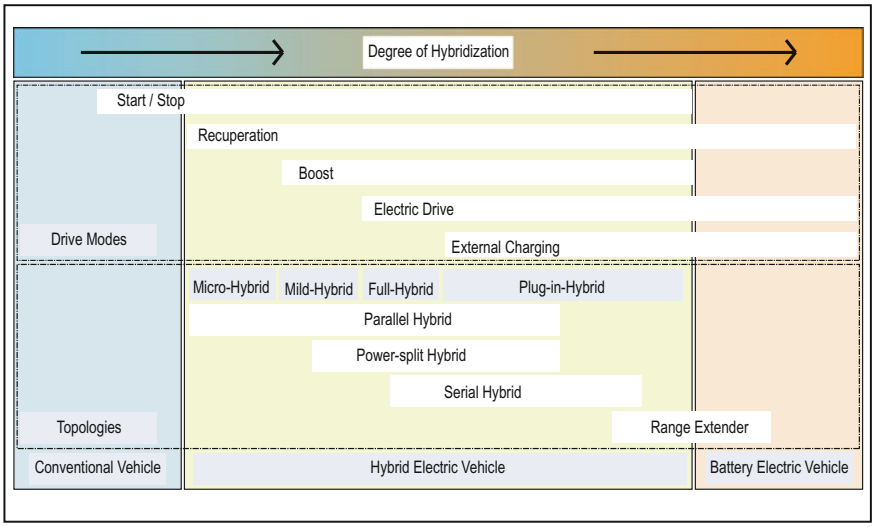


Fig. 10.10 Relationship of degree of hybridization and drive modes

dominated electric traction systems should focus on an optimal electrical–mechanical energy flow.

Figure 10.10 shows the relationship between the degree of hybridization and the functional modes.

10.4.1 Parallel Hybrids

Parallel HEV configurations are shown in Fig. 10.11 with the energy flow depicted in Fig. 10.12. They employ an additional MG, which is either coupled mechanically with the combustion engine on the same shaft or mounted on the post-transmission axle. The common architectures are distinguished depending on the location of the MG in the driveline. P1 and P2 configurations are architectures with MG and ICE coupled on the same shaft whereas the MG is coupled on the so-called secondary axle (nondriven axle) in the P4 configuration. The P4 configuration is also known as a *through-the-road hybrid*. Besides the both prime movers (combustion engine and motor/generator) and two energy storages (fuel reservoir and battery), a parallel configuration also contains several coupling and conversion elements. The torque coupling between ICE and driveline of P1 and P2 configurations is achieved by employing a spur gear and a clutch, which is installed at the input side of a gearbox. A speed coupling is also possible but rather seldom applied. It uses a planetary gear to superimpose the rotational speeds of ICE and MG. In all structures, the electrical machine is permanently connected to the drivetrain when energy flows from or to the wheels. The advantage of these architectures are their relative simplicity and

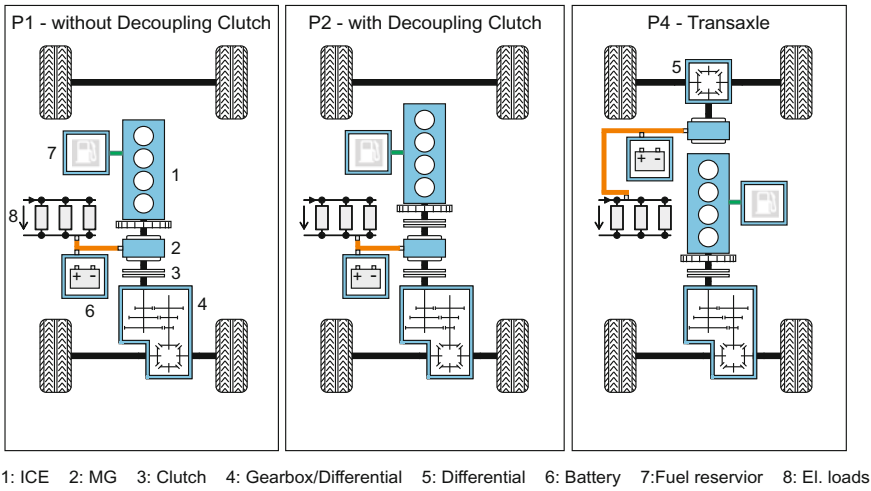


Fig. 10.11 Topologies of parallel hybrids

the ability to apply mass-produced components from conventional vehicles without expensive redesigns.

Both prime movers can be used to drive the vehicle, either individually or at the same time. In case of torque superposition, the configuration allows the ICE torque to be chosen within certain limits independently from the driver's request. Whereas, in case of rotational speed superposition, the combustion engine's speed can be chosen within certain limits independently from the vehicle speed.

Simple driveline models for the P1, P2, and P4 topologies are developed using the fundamental equations derived by using Euler's second law. For all driveline models we assume zero pinion gear inertia and no drive-shaft flexibility. A good overview about driveline dynamics can be found in Kiencke and Nielsen [30].

We start by modeling the P2 driveline dynamics first.

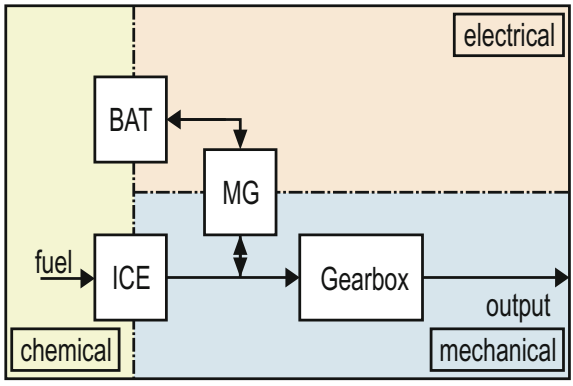


Fig. 10.12 Principle of energy flow in a parallel hybrid

Engine:

Euler's second law yields the following model

$$I_{ice}\dot{\omega}_{ice}(t) = T_{ice}(t) - T_{clth1}(t) \tag{10.58}$$

where I_{ice} is the mass moment of inertia of the engine.

Clutch:

The clutch is assumed to be stiff. When the clutch is engaged and no friction is assumed the following equations for the torque and angular velocity hold

$$T_{clth1}(t) = T_{clth2}(t) \tag{10.59}$$
$$\omega_{ice}(t) = \omega_{clth}(t) \tag{10.60}$$

where $\omega_{clth}(\cdot)$ is the clutch speed, $T_{clth1}(\cdot)$ is the clutch input torque, and $T_{clth2}(\cdot)$ is the clutch output torque.

Motor/Generator:

Applying Euler's second law to the motor/generator yields the following model

$$I_{mg}\dot{\omega}_{mg}(t) = T_{clth2}(t) + T_{mg}(t) - T_{gbx1}(t) \tag{10.61}$$
$$\omega_{clth}(t) = \omega_{mg}(t) \tag{10.62}$$

where $T_{gbx1}(\cdot)$ is the input torque of the gearbox. Conversion of (10.61) to a function of engine speed is obtained by using (10.58)–(10.60) and (10.62), which yields

$$(I_{ice} + I_{mg})\dot{\omega}_{ice}(t) = T_{ice}(t) + T_{mg}(t) - T_{gbx1}(t).$$

Gearbox:

The gearbox is modeled by only one rotating inertia I_{gbx}, a set of gear ratios $\mathbf{i}_{gbx} \in \mathbb{R}^{N_{gbx}}$, and friction. The actual gear ratio $i_{gbx}(t) \in \mathbf{i}_{gbx},\ t \in [t_0, t_f]$ can be represented as a piecewise constant function of time. Then, the model of the gearbox is given by

$$I_{gbx}\dot{\omega}_{gbx2}(t) = i_{gbx}(t)T_{gbx1}(t) - T_{gbx,f}(t) - T_{gbx2}(t) \tag{10.63}$$
$$\omega_{ice}(t) = \omega_{gbx1}(t) = i_{gbx}(t)\omega_{gbx2}(t) \tag{10.64}$$

where $T_{gbx2}(\cdot)$ and $T_{gbx,f}(\cdot)$ are the output torque and friction torque of the gearbox, respectively. The friction torque can be measured on a powertrain test bench. The torque constraint of the P2 topology obeys the condition that the gearbox input torque $T_{gbx1}(\cdot)$ is given by

$$T_{gbx1}(t) = T_{ice}(t) + T_{mg}(t) - (I_{ice} + I_{mg})\dot{\omega}_{ice}(t). \tag{10.65}$$

Applying (10.64) to (10.63) and inserting (10.65) into the gearbox model (10.63) yields an ODE of engine speed as

$$\begin{aligned}\left[I_{gbx} + i_{gbx}^2(t) \cdot \left(I_{ice} + I_{mg}\right)\right]\dot{\omega}_{ice}(t) = i_{gbx}^2(t) \cdot \left(T_{ice}(t) + T_{mg}(t)\right) - i_{gbx}(t)T_{gbx,f}(t) \\ -i_{gbx}(t)T_{gbx2}(t).\end{aligned} \tag{10.66}$$

As convention for the remaining book, $T_{gbx}(\cdot)$ without numbering means always the gearbox input torque.

Final drive:

Similar to the gearbox, the final drive is modeled by one rotating inertia I_{fd}. The model of the final drive is given by

$$I_{fd}\dot{\omega}_{fd2}(t) = i_{fd}T_{gbx2}(t) - T_{fd,f}(t) - T_{wh}(t) \tag{10.67}$$

$$\begin{aligned}\omega_{fd2}(t) &= \omega_{wh}(t) \\ \omega_{fd1}(t) &= i_{fd}\omega_{wh}(t) \\ \omega_{ice}(t) &= i_{gbx}(t)i_{fd}\omega_{wh}(t)\end{aligned} \tag{10.68}$$

where $\omega_{fd1}(\cdot)$ and $\omega_{fd2}(\cdot)$ is the input and output speed of the final drive, respectively, and $T_{fd,f}(\cdot)$ is the friction torque of the final drive. Rearranging (10.66) and inserting into the final drive model (10.67) and applying (10.68) yields an ODE of engine speed as

$$\begin{aligned}\left\{I_{fd} + i_{fd}^2 \cdot \left[I_{gbx} + i_{gbx}^2(t) \cdot \left(I_{ice} + I_{mg}\right)\right]\right\}\dot{\omega}_{ice}(t) = i_{gbx}^2(t)i_{fd}^2 \cdot \left(T_{ice}(t) + T_{mg}(t)\right) \\ - i_{gbx}(t)i_{fd}^2 T_{gbx,f}(t) \\ - i_{gbx}(t)i_{fd}T_{fd,f}(t) \\ - i_{gbx}(t)i_{fd}T_{wh}(t).\end{aligned} \tag{10.69}$$

For the sake of simplicity let us aggregate the friction torques as

$$T_{loss}(t) = i_{fd}T_{gbx,f}(t) + T_{fd,f}(t).$$

Wheel:

Using (10.9) and applying Euler's second law to (10.13) yields

$$\begin{aligned}T_{road}(t) &= r_{wh}F_w(t) \\ I_{veh}\dot{\omega}_{wh}(t) &= T_{wh}(t) - T_{road}(t) - T_{brk}(t)\end{aligned} \tag{10.70}$$

where

$$I_{veh} = \tilde{m} r_{wh}^2$$

is the effective rotating inertia, $T_{road}(\cdot)$ is the road load, and $T_{brk}(\cdot)$ is an additional mechanical torque contribution for the friction brake.

P2 driveline:

Simplifying (10.69) using the compact notation of the substitute inertia

$$\tilde{I}_{p2}(t) = \frac{I_{fd} + i_{fd}^2 \cdot \left[I_{gbx} + i_{gbx}^2(t) \cdot \left(I_{ice} + I_{mg} \right) \right]}{i_t(t)},$$

and the rotational speed

$$\omega_{ice}(t) = \omega_{mg}(t) = i_{gbx}(t) i_{fd} \omega_{wh}(t) = i_t(t) \omega_{wh}(t), \tag{10.71}$$

yields

$$\tilde{I}_{p2}(t) \dot{\omega}_{ice}(t) = i_t(t) \cdot \left(T_{ice}(t) + T_{mg}(t) \right) - T_{loss}(t) - T_{wh}(t). \tag{10.72}$$

The step that remains to obtain a driveline model for the P2 hybrid is to couple (10.70) with (10.72). Thus, calculating the time derivative of (10.71) using the chain-rule yields

$$\dot{\omega}_{ice}(t) = \frac{\mathrm{d}i_t}{\mathrm{d}t} \omega_{wh}(t) + i_t(t) \dot{\omega}_{wh}(t). \tag{10.73}$$

The time derivative of the piecewise constant function $i_t(\cdot)$ is zero, except for the time instances t_j of gear shifting. At these time instances the derivatives $\frac{\mathrm{d}i_t}{\mathrm{d}t}\big|_{t=t_j}$ are not defined. This let us argue that (10.73) reduces to

$$\dot{\omega}_{ice}(t) = i_t(t) \dot{\omega}_{wh}(t). \tag{10.74}$$

Inserting (10.70) into (10.74) and equating with (10.72) yields the governing equation of the P2 driveline as

$$I_{p2h}(t) \dot{\omega}_{ice}(t) = i_t(t) \cdot \left(T_{ice}(t) + T_{mg}(t) \right) - T_{loss}(t) - T_{road}(t) - T_{brk}(t) \tag{10.75}$$

where the substitution inertia for hybrid driving is given as

$$I_{p2h}(t) = \frac{m r_{wh}^2 + I_{wh} + I_{fd} + i_{fd}^2 \cdot \left[I_{gbx} + i_{gbx}^2(t) \cdot \left(I_{ice} + I_{mg} \right) \right]}{i_t(t)}.$$

The governing equation (10.75) is derived with the assumption that the clutch is engaged. When the clutch is disengaged, the governing equation reduces to

$$I_{p2e}(t)\dot{\omega}_{mg}(t) = i_t(t)T_{mg}(t) - T_{loss}(t) - T_{road}(t) - T_{brk}(t)$$

where the substitution inertia for electrical driving is given as

$$I_{p2e}(t) = \frac{mr_{wh}^2 + I_{wh} + I_{fd} + i_{fd}^2 \cdot \left(I_{gbx} + i_{gbx}^2(t)I_{mg}\right)}{i_t(t)}.$$

The acceleration resistance $F_a(\cdot)$ and the acceleration torque $T_a(\cdot)$ for the P2 topology can then be refined to

$$\begin{aligned} F_a(t) &= m \cdot \left(1 + \frac{I_{wh} + I_{fd} + i_{fd}^2 \cdot \left[I_{gbx} + i_{gbx}^2(t) \cdot \left(I_{ice} + I_{mg}\right)\right]}{mr_{wh}^2}\right) \dot{v}(t) \\ &= m\tilde{\gamma}_m(i_{gbx}(t))\dot{v}(t) \end{aligned}$$

and

$$T_a(t) = mr_{wh}^2\tilde{\gamma}_m(i_{gbx}(t))\dot{\omega}_{wh}(t),$$

respectively. $\tilde{\gamma}_m(\cdot)$ is the enhanced mass factor, which can be determined by vehicle tests. One can realize that the mass factor has the highest value for the highest gear ratio.

P1 driveline:

The difference between the P1 and the P2 topology is the missing decoupling clutch between ICE and MG. Thus, the governing equation of the P1 driveline is identical with the P2 driveline when the clutch is engaged.

P4 driveline:

In the P4 topology, a MG is installed on the secondary axle. Using the models derived before it is easy to obtain a driveline model for the P4 topology. Let us assume that the power transfer from the secondary axle over the road to the primary axle is without any losses and that the MG is connected to a fixed gear ratio i_{fd2} with inertia I_{fd2}. Since, we have two driving axles half of the wheel torque applies to each axle.

Then, applying the principles derived for the final drive model (10.67)–(10.69) to the secondary axle yields

$$\left(I_{fd2} + i_{fd2}^2 I_{mg}\right)\dot{\omega}_{mg}(t) = i_{fd2}^2 T_{mg}(t) - T_{fd2,f}(t) - \frac{i_{fd2}}{2} \cdot T_{wh}(t) \tag{10.76}$$

$$\left[I_{fd} + i_{fd}^2 \cdot \left(I_{gbx} + i_{gbx}^2(t) I_{ice}\right)\right]\dot{\omega}_{ice}(t) = i_t^2(t) T_{ice}(t) - i_t(t) i_{fd} T_{gbx,f}(t) - i_t(t) T_{fd,f}(t) - \frac{i_t(t)}{2} \cdot T_{wh}(t). \tag{10.77}$$

Converting (10.76) to an ODE of engine speed using

$$\omega_{mg}(t) = \frac{i_{fd2}}{i_t(t)} \cdot \omega_{ice}(t)$$

results in

$$\left(I_{fd2} + i_{fd2}^2 I_{mg}\right)\dot{\omega}_{ice}(t) = i_t(t) i_{fd2} T_{mg}(t) - \frac{i_t(t)}{i_{fd2}} \cdot T_{fd2,f}(t) - \frac{i_t(t)}{2} \cdot T_{wh}(t). \tag{10.78}$$

Superposition of (10.77) and (10.78) yields

$$\left[I_{fd} + I_{fd2} + i_{fd}^2 \cdot \left(I_{gbx} + i_{gbx}^2(t) I_{ice}\right) + i_{fd2}^2 I_{mg}\right]\dot{\omega}_{ice}(t) = i_t(t) \cdot \left(i_t(t) T_{ice}(t) + i_{fd2} T_{mg}(t) - i_{fd} T_{gbx,f}(t) - T_{fd,f}(t) - \frac{1}{i_{fd2}} T_{fd2,f}(t) - T_{wh}(t)\right). \tag{10.79}$$

Simplifying (10.79) using the substitution inertia

$$\tilde{I}_{p4} = \frac{I_{fd} + I_{fd2} + i_{fd}^2 \cdot \left(I_{gbx} + i_{gbx}^2(t) I_{ice}\right) + i_{fd2}^2 I_{mg}}{i_t(t)}$$

yields

$$\tilde{I}_{p4}\dot{\omega}_{ice}(t) = i_t(t) T_{ice}(t) + i_{fd2} T_{mg}(t) - T_{loss}(t) - \frac{1}{i_{fd2}} T_{fd2,f}(t) - T_{wh}(t). \tag{10.80}$$

It is now straightforward to apply the wheel model to (10.80) to obtain the governing equation of the P4 driveline.

10.4.2 Power-Split Hybrids

A common combination between parallel and serial principles is the so-called power-split hybrid. Figure 10.13 shows two popular power-split topologies. In this hybrid configuration the input power is split into two parts as shown in Fig. 10.14. A part of the mechanical output power of the ICE is directly transmitted through one or more planetary gear(s) to the drive wheels. The remainder is transferred through the so-called electrical variator to the wheels. The electrical variator is a composition of the two motor/generators. One works as a generator while the other operates as a motor. That allows the variance in ICE speed independently from the vehicle speed and is therefore known as *electrical continuously variable transmission* (ECVT). The lower efficiency of the variator is caused by its twice electrical energy conversion. This is certainly not as good as the efficiency of the mechanical part.

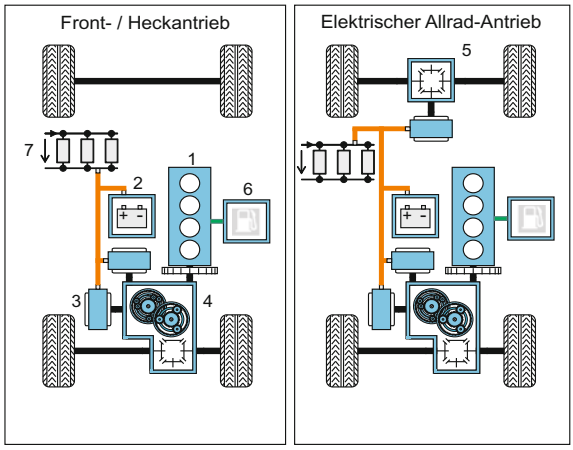


Fig. 10.13 Topologies of powersplit hybrids

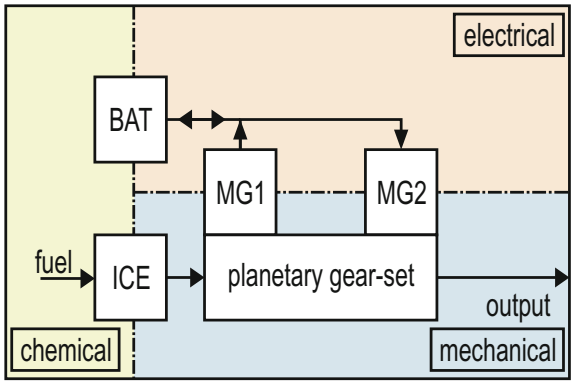


Fig. 10.14 Topologies of powersplit hybrids

Many different power-split configurations have therefore been developed over the last four decades to minimize the power throughput over the electrical variator and the recirculation power losses over various speed ratios, among them Schmidt [54, 55], and Holmes and Schmidt [23]. The gear ratios of the planetary gears must therefore be chosen, such that the power-splitting factor

$$\epsilon(t) := \frac{P_{el}(t)}{P_{mech}(t)}, \tag{10.81}$$

which resembles the ratio of the power on the electrical branch $P_{el}(\cdot)$ and the power on the mechanical branch $P_{mech}(\cdot)$, is small for a defined vehicle speed range.

There are mainly three different configurations for realizing the power-split transmission:

- input power-split;
- output power-split; and
- compound power-split.

All have in common that a part of the input power from the ICE is transmitted through the mechanical path while the other part is sent through the electrical variator.

Power-split configurations are often designated as more complex hybrid drive variants compared with parallel and serial configurations. The mechanical structure is much simpler in many design variants but the increased complexity arises from the tighter control requirements. Overviews can be found in Guzzella and Sciarretta [20].

10.4.2.1 Characteristics of Power Splits

Planetary gearboxes can be connected together to obtain different configurations. If the power-split configuration results in two input/output shafts for the electrical variator, the ICE and the remaining drivetrain, then the static speed relationships can be calculated by the linear system of equations

$$\begin{bmatrix} \omega_{mg1}(t) \\ \omega_{mg2}(t) \end{bmatrix} = \mathbf{A} \cdot \begin{bmatrix} \omega_{ice}(t) \\ \omega_{wh}(t) \end{bmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{bmatrix} \omega_{ice}(t) \\ \omega_{wh}(t) \end{bmatrix} \tag{10.82}$$

where $\omega_{mg1}(\cdot)$ and $\omega_{mg2}(\cdot)$ are the angular speeds of MG1 and MG2, respectively. With the assumption that the system is without losses and the electrical powers of both MGs compensate each other, i.e., battery power is zero and the battery is neither charged nor discharged, the linear system (10.82) can be reformulated to the static torque relationships

$$\begin{bmatrix} T_{mg1}(t) \\ T_{mg2}(t) \end{bmatrix} = (-\mathbf{A}^T)^{-1} \cdot \begin{bmatrix} T_{ice}(t) \\ -T_{road}(t) - T_{brk}(t) \end{bmatrix}$$
$$= \frac{1}{a_{11}a_{22} - a_{12}a_{21}} \begin{pmatrix} -a_{22} & a_{21} \\ a_{12} & -a_{11} \end{pmatrix} \begin{bmatrix} T_{ice}(t) \\ -T_{wh}(t) \end{bmatrix} \tag{10.83}$$

where $T_{mg1}(\cdot)$ and $T_{mg2}(\cdot)$ are the torques of MG1 and MG2, respectively.

Defining the most desirable direction of electrical power flow cannot be performed for all power-split configurations in the same manner. The preferred power flow depends on the layout and thus on the operation mode of MG1 and MG2. Consequently, we define a general power-split ratio (10.81) by using the following definition

$$\epsilon(t) := \frac{P_{mg1}(t)}{P_{ice}(t)} = \frac{T_{mg1}(t)\omega_{mg1}(t)}{T_{ice}(t)\omega_{ice}(t)}. \tag{10.84}$$

We will see that some power-split configurations need a slightly modified definition of (10.84) including an additional minus sign.

The power-split ratio obtained (10.84) is still imprecise in terms of the properties of the configuration used. Therefore, using $T_{mg1}(\cdot)$ from (10.83) and $\omega_{mg1}(\cdot)$ from (10.82) we are able to refine $\epsilon(\cdot)$ more configuration specifically to

$$\epsilon(t) := -\frac{a_{11}a_{22} + a_{12}a_{21}}{a_{11}a_{22} - a_{12}a_{21}} + \frac{-a_{11}a_{21}i_{ecvt}(t) - \dfrac{a_{12}a_{22}}{i_{ecvt}(t)}}{a_{11}a_{22} - a_{12}a_{21}}$$
$$= m_0 + m_1 i_{ecvt}(t) + \frac{m_2}{i_{ecvt}(t)} \tag{10.85}$$

where $i_{evct}(\cdot)$ is the total transmission ratio, which is defined by

$$i_{evct}(t) := \frac{\omega_{ice}(t)}{\omega_{wh}(t)}.$$

Obviously, the power-splitting ratio consists of three terms. The first term is constant, the second term is proportional to the ECVT transmission ratio $i_{ecvt}(\cdot)$ (if $a_{11},\ a_{21} \neq 0$), and the third term is proportional to the inverse of the $i_{ecvt}(\cdot)$ (if $a_{12},\ a_{22} \neq 0$). In anticipation of the next sections, inspection of (10.85) reveals that beside input, output, and compound power-split no additional basic types of power-split exist. Tables 10.1 and 10.2 summarizes the power-split ratios for the elementary power-split configurations.

For single node power splits, one can observe from (10.85) that $\epsilon(\cdot)$ will be zero if

$$i_{ecvt,1} = -\frac{a_{22}}{a_{21}} \quad \text{for } a_{11} = 0 \vee a_{12} = 0$$

or

$$i_{ecvt,1} = -\frac{a_{12}}{a_{11}} \quad \text{for } a_{21} = 0 \vee a_{22} = 0$$

applies. For dual node power splits we obtain

$$i_{ecvt,1} = -\frac{a_{22}}{a_{21}}$$
$$i_{ecvt,2} = -\frac{a_{12}}{a_{11}}.$$

These transmission ratios generate a purely mechanical transmission characteristics of the ECVT, because $\omega_{mg1}(\cdot)$ or $\omega_{mg2}(\cdot)$ is zero. These speeds are called *mechanical nodes*.

The total efficiency of the power-split system can be characterized by

$$\eta_{tot}(t) = \frac{P_{wh}(t)}{P_{ice}(t)} = \left(1 - \frac{|\epsilon(t)|}{\eta_{mg1}(t)}\right)\eta_{mech} + |\epsilon(t)|\,\eta_{mg2}(t) \tag{10.86}$$

where η_{mech} is the efficiency of the mechanical driveline including the efficiencies of the planetary gears.

10.4.2.2 Input Power-Split

The first commercial power-split HEV offered to the market was presented by Toyota (Toyota Prius I) employing an input power-split architecture, named *Toyota Hybrid System* (THS). The system consists of one minus planetary gear and two motors/generators. In 2003, Toyota released a THS system, abbreviated THS II, improved by taking advantage of higher voltages to improve the electrical efficiency. In 2005, Lexus and Toyota improved this system configuration with a second planetary gear to reduce the power of MG2, which is known as the *second generation of the input power-split* (IPS2). This architecture has been employed in Toyota Prius III and Lexus RX 450 h and is shown in Fig. 10.15.

The *input power-split* (IPS) configuration must satisfies the rotational speed constraints of one planetary gear (10.42)

$$0 = (1 + i_{01})\omega_{ice}(t) - \omega_{mg1}(t) - i_{01} i_{fd} \omega_{wh}(t) \tag{10.87}$$

where i_{01} is the stationary gear ratio of planetary gear 1. The carrier of the PG2 is fixed as shown in Fig. 10.15. Then, PG2 serves as a reduction gear with the ratio i_{rd}. The wheel speed is then simply given by

$$\omega_{wh}(t) = \frac{1}{i_{rd} i_{fd}} \omega_{mg2}(t). \tag{10.88}$$

The reduction gear has the purpose to lower the torque at the MG2.

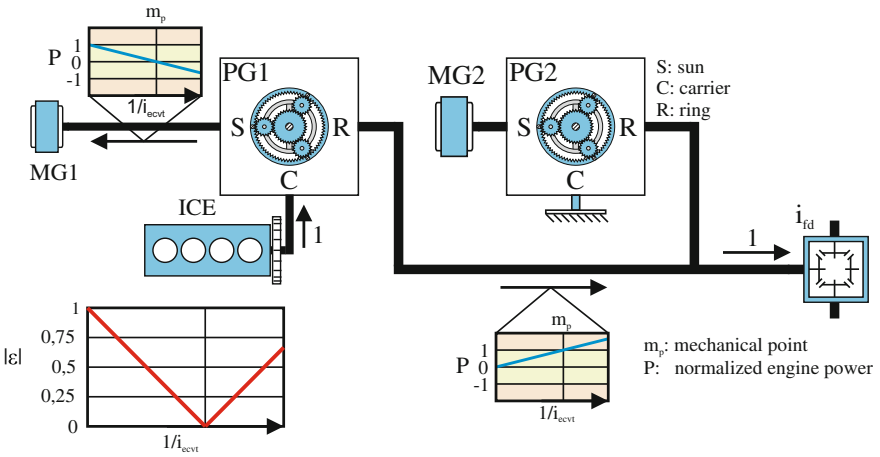


Fig. 10.15 Schematic of the input power-split configuration. Reactive power results for the inverse transmission ratios higher than the mechanical node $-a_{11}/a_{12}$

It is now easy to derive the kinematic constraints using the speed equations (10.87) and (10.88). The kinematic rotational speed constraints are

$$\begin{bmatrix} \omega_{mg1}(t) \\ \omega_{mg2}(t) \end{bmatrix} = \mathbf{A}_{ips} \cdot \begin{bmatrix} \omega_{ice}(t) \\ \omega_{wh}(t) \end{bmatrix} = \begin{pmatrix} 1 + i_{01} & -i_{01} i_{fd} \\ 0 & i_{rd} i_{fd} \end{pmatrix} \begin{bmatrix} \omega_{ice}(t) \\ \omega_{wh}(t) \end{bmatrix}.$$

The kinematic torque constraints can be calculated by transposing and inverting matrix $\mathbf{A}_{ips}$ which yields

$$\begin{bmatrix} T_{mg1}(t) \\ T_{mg2}(t) \end{bmatrix} = (-\mathbf{A}_{ips}^T)^{-1} \cdot \begin{bmatrix} T_{ice}(t) \\ -T_{wh}(t) \end{bmatrix} = \begin{pmatrix} \dfrac{-1}{1 + i_{01}} & 0 \\ \dfrac{-i_{01}}{(1 + i_{01}) i_{rd}} & \dfrac{-1}{i_{rd} i_{fd}} \end{pmatrix} \begin{bmatrix} T_{ice}(t) \\ -T_{wh}(t) \end{bmatrix}. \tag{10.89}$$

It should be noted that these equations only hold with the assumption of zero battery power and no mechanical power losses.

The design of the input-split system requires that MG1 works as a generator in order to stabilize the ICE and to obtain no power recirculation. Thus, the signed power-split ratio is defined by

$$\epsilon(t) := -\frac{P_{mg1}(t)}{P_{ice}(t)} \tag{10.90}$$

which includes an additional minus sign compared with the general definition in (10.84). As can be seen from the transfer matrix $\mathbf{A}_{ips}$, the value a_{21} is zero. This

leads to a zero entry in the first equation of the torque constraint (see (10.89)). The power-split ratio simplifies to

$$\epsilon(t) = \frac{1}{1+i_{01}} \cdot \left(1 + i_{01} - \frac{i_{01} i_{fd}}{i_{ecvt}(t)}\right) = 1 - \frac{i_{01} i_{fd}}{1+i_{01}} \cdot \frac{1}{i_{ecvt}(t)}. \tag{10.91}$$

Comparing (10.91) with (10.85) shows that

$$\epsilon(t) = -1 + \frac{m_2}{i_{ecvt}(t)} = -1 - \frac{a_{12}}{a_{11}} \cdot \frac{1}{i_{ecvt}(t)}. \tag{10.92}$$

Since the sign of (10.90) is different to definition (10.84), (10.92) has to be multiplied by -1, which yields

$$\epsilon(t) = 1 + \frac{a_{12}}{a_{11}} \cdot \frac{1}{i_{ecvt}(t)} = 1 - \frac{i_{01} i_{fd}}{1+i_{01}} \cdot \frac{1}{i_{ecvt}(t)}.$$

The power-split ratio over the inverse transmission ratio using (10.91) is depicted in Fig. 10.16. Comparing with (10.85) yields

$$-m_0 = 1, \quad -m_2 = \frac{a_{12}}{a_{11}}.$$

The mechanical node ($\epsilon(t) = 0$) of the inverse transmission ratio for the design values $i_{01} = 2.6$ and $i_{fd} = 3.08$ is given by:

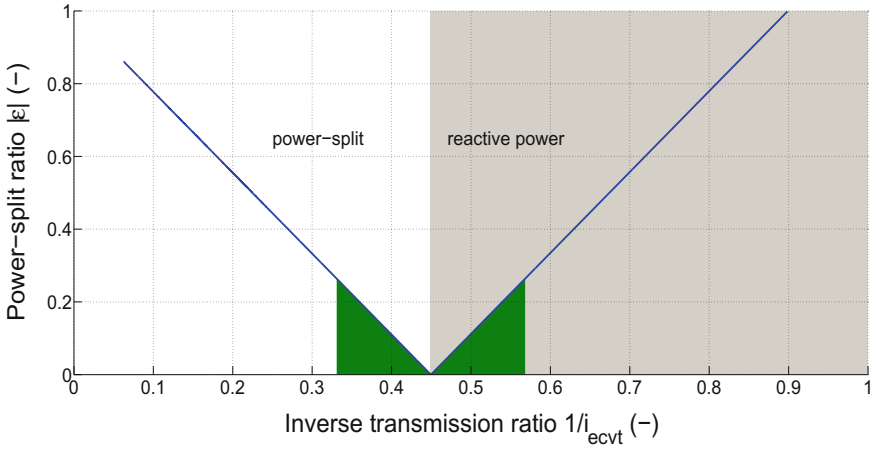


Fig. 10.16 Power-split ratio ϵ of the input-split configuration with $i_{01} = 2.6$ and $i_{fd} = 3.08$. The *light gray shaded area* indicates inverse transmission ratios with reactive power. The *green shaded areas* determined using (10.86) indicates the operating domain with total efficiency higher than 90%

$$\frac{1}{i_{ecvt,1}} = -\frac{a_{11}}{a_{12}} = \frac{1+i_{01}}{i_{01}i_{fd}} = \frac{1+2.6}{2.6\cdot 3.08} \approx 0.45. \tag{10.93}$$

One can observe from (10.93) that the modified power-split definition (10.90) has no effect on the mechanical node.

Start-up from standstill is possible without decoupling the ICE from the driveline. This might explain, why most power-split hybrids do not have a decoupling element. However, study of (10.91) reveals that at low velocities the input-split system depicted in Fig. 10.15 transfers all power from the ICE through the electrical variator to the wheels, which is quite inefficient because of the double energy conversion. Only at the mechanical node (10.93) is the ICE power completely transfered from the carrier to the ring gear which yields the highest efficiency of the system. It is therefore desirable to design the system close to this mechanical node. This is possible for nearly all vehicle speeds as shown in Boehme et al. [5], however, this compromise leads to less dynamic vehicle responses.

Consequently, a good design operates the ICE near the mechanical node in order to cut the electrical energy flow from MG1 to MG2 and uses MG2 as a traction motor for start-up or acceleration support. The agility of the vehicle is then mainly dependent of the size of MG2. The overdrive functionality with low transmission ratios, i.e., $i_{ecvt}(t) < 1$, which is usually employed for vehicles with manual or automatic gearboxes in order to reduce fuel consumption, is here directly proportional to high electrical power flow and losses due to power recirculation.

The area of mechanical power recirculation as shown on the right-side of Fig. 10.16 occurs whenever any shaft of the planetary gear is loaded with more mechanical power than the ICE provides. The power recirculation is known as reactive power. On the one hand, because of the relatively high efficiency of the planetary gear(s), the additional mechanical losses are relative small but may require the resizing of the shafts and bearings. On the other hand, because of the high losses in the electrical path, this reactive power can reduce the efficiency of the overall ECVT transmission seriously. It can generally be recommended to keep the power-split ratio below 30%.

The dynamic behavior of the driveline is derived by assuming zero pinion gear inertia and vehicle longitudinal dynamics only (Liu and Peng [38]). By applying Euler's second law for the ring gear, sun gear, and carrier gear node of PG1, respectively, and using (10.43)–(10.45), the governing equations of the input power-split configuration are obtained. They can be summarized as

$$\begin{aligned}
(I_{ice} + I_{c1})\,\dot{\omega}_{ice}(t) &= T_{ice}(t) - (1+i_{01})T_{i1}(t) \\
\left(I_{mg1} + I_{s1}\right)\dot{\omega}_{mg1}(t) &= T_{mg1}(t) + T_{i1}(t) \\
\left[I_{veh} + \left(I_{mg2}i_{rd}^2 + I_{r1}\right)i_{fd}^2\right]\dot{\omega}_{wh}(t) &= -(T_{brk}(t) + T_{road}(t)) \\
&\quad + i_{rd}i_{fd}T_{mg2}(t) + i_{01}i_{fd}T_{i1}(t)
\end{aligned}$$

where $T_{i1}(\cdot)$ represent the internal torque on the pinion gears of PG1 (see (10.46)).

$$
\begin{bmatrix} T_{ice}(t) \\ T_{mg1}(t) \\ -(T_{brk}(t) + T_{road}(t)) + i_{rd} i_{fd} T_{mg2}(t) \\ 0 \end{bmatrix}
$$

$$
= \begin{pmatrix} I_{ice} + I_{c1} & 0 & 0 & 1 + i_{01} \\ 0 & I_{mg1} + I_{s1} & 0 & -1 \\ 0 & 0 & I_{veh} + (I_{mg2} i_{rd}^2 + I_{r1}) i_{fd}^2 & -i_{01} i_{fd} \\ 1 + i_{01} & -1 & -i_{01} i_{fd} & 0 \end{pmatrix} \begin{bmatrix} \dot{\omega}_{ice}(t) \\ \dot{\omega}_{mg1}(t) \\ \dot{\omega}_{wh}(t) \\ T_{i1}(t) \end{bmatrix}. \quad (10.94)
$$

Inversion of the left-hand side matrix yields the differential-algebraic equations of the system dynamics

$$
\begin{bmatrix} \dot{\omega}_{ice}(t) \\ \dot{\omega}_{mg1}(t) \\ \dot{\omega}_{wh}(t) \\ T_{i1}(t) \end{bmatrix} = \underbrace{\begin{pmatrix} \mathbf{I}_{ips} & \mathbf{D}_{ips}^{[1:3]} \\ \left(\mathbf{D}_{ips}^{[1:3]}\right)^T & 0 \end{pmatrix}^{-1}}_{\mathbf{P}_{ips} = \begin{pmatrix} p_{11} & \cdots & p_{14} \\ \vdots & \ddots & \vdots \\ p_{41} & \cdots & p_{44} \end{pmatrix}} \begin{bmatrix} T_{ice}(t) \\ T_{mg1}(t) \\ -(T_{brk}(t) + T_{road}(t)) + i_{rd} i_{fd} T_{mg2}(t) \\ 0 \end{bmatrix} \quad (10.95)
$$

where $\mathbf{I}_{ips} \in \mathbb{R}^{3\times 3}$ from (10.95) is the diagonal inertia matrix of each gear node

$$
\mathbf{I}_{ips} = \text{diag}\left\{ I_{ice} + I_{c1},\, I_{mg1} + I_{s1},\, I_{veh} + (I_{mg2} i_{rd}^2 + I_{r1}) i_{fd}^2 \right\}
$$

and $\mathbf{D}_{ips} \in \mathbb{R}^{4\times 1}$ is the kinematic constraint matrix of the input power-split, which describes the static speed constraints from (10.87) as

$$
\mathbf{D}_{ips} = \begin{pmatrix} 1 + i_{01} \\ -1 \\ -i_{01} i_{fd} \\ 0 \end{pmatrix}.
$$

10.4.2.3 Output Power-Split

The *output power-split* (OPS) powertrain consists of one minus planetary gear and two motors/generators, similar to the input-split configuration. The engine and MG2 are coaxial on one shaft and connected to the carrier, MG1 is connected to the sun gear, and the remaining driveline consisting a final drive gear and a differential is connected to the ring gear.

According to Fig. 10.17 the output power-split configuration must satisfies the rotational speed constraint of one planetary gear

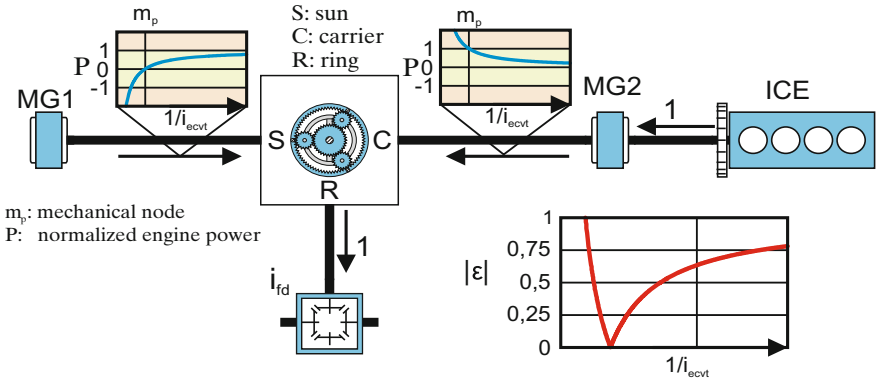


Fig. 10.17 Schematic of the output power-split configuration. Reactive power results for the inverse transmission ratios lower than the mechanical node $-a_{11}/a_{12}$

$$\omega_{mg1}(t) = (1 + i_{01})\omega_{ice}(t) - i_{01} i_{fd} \omega_{wh}(t) \tag{10.96}$$

$$\omega_{mg2}(t) = \omega_{ice}(t). \tag{10.97}$$

According to (10.96) and (10.97), the kinematic speed constraints are

$$\begin{bmatrix} \omega_{mg1}(t) \\ \omega_{mg2}(t) \end{bmatrix} = \mathbf{A}_{ops} \cdot \begin{bmatrix} \omega_{ice}(t) \\ \omega_{wh}(t) \end{bmatrix} = \begin{pmatrix} 1 + i_{01} & -i_{01} i_{fd} \\ 1 & 0 \end{pmatrix} \begin{bmatrix} \omega_{ice}(t) \\ \omega_{wh}(t) \end{bmatrix}. \tag{10.98}$$

The kinematic torque constraints can be calculated by transposing and inverting matrix $\mathbf{A}_{ops}$, which yields

$$\begin{bmatrix} T_{mg1}(t) \\ T_{mg2}(t) \end{bmatrix} = (-\mathbf{A}_{ops}^T)^{-1} \cdot \begin{bmatrix} T_{ice}(t) \\ -T_{wh}(t) \end{bmatrix} = \begin{pmatrix} 0 & \dfrac{1}{i_{01} i_{fd}} \\ -1 & \dfrac{-1 - i_{01}}{i_{01} i_{fd}} \end{pmatrix} \begin{bmatrix} T_{ice}(t) \\ -T_{wh}(t) \end{bmatrix}. \tag{10.99}$$

The signed power-split ratio of the output-split system is defined by

$$\epsilon(t) := \frac{P_{mg1}(t)}{P_{ice}(t)}. \tag{10.100}$$

Using the first equations of (10.98) and (10.99) we obtain the power-split ratio

$$\epsilon(t) = -\frac{i_{ecvt}(t)}{i_{01} i_{fd}} \cdot \left(1 + i_{01} - \frac{i_{01} i_{fd}}{i_{ecvt}(t)}\right) = 1 - \frac{1 + i_{01}}{i_{01} i_{fd}} i_{ecvt}(t). \tag{10.101}$$

In contrast to an input-split hybrid, one can observe from (10.101) that power recirculation occurs at high transmission ratios and thus at low vehicle speeds. However, the output-split system achieves efficient power splitting at high vehicle speeds.

Hence, the power-split ratio from Fig. 10.18 can be plotted by

$$\epsilon(t) = 1 + m_1 i_{ecvt}(t) = 1 + \frac{a_{11}}{a_{12}} i_{ecvt}(t) = 1 - \frac{1 + i_{01}}{i_{01} i_{fd}} i_{ecvt}(t).$$

Comparing with (10.85) yields

$$m_0 = 1, \quad m_1 = \frac{a_{11}}{a_{12}}.$$

The corresponding mechanical node over the inverse transmission ratio is obtained as

$$\frac{1}{i_{ecvt,1}} = -\frac{a_{11}}{a_{12}} = \frac{1 + i_{01}}{i_{01} i_{fd}}.$$

For the design values $i_{01} = 2.6$ and $i_{fd} = 3.08$ we obtain for the following mechanical node for the output-split system:

$$\frac{1}{i_{ecvt,1}} = \frac{1 + 2.6}{2.6 \cdot 3.08} \approx 0.45$$

which is the same as for the input-split system (10.93).

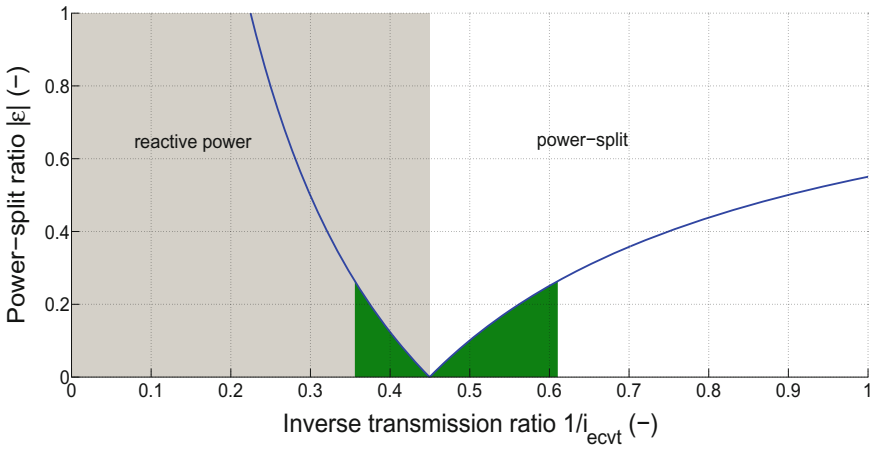


Fig. 10.18 Power-split ratio ϵ of the output-split configuration with $i_{01} = 2.6$ and $i_{fd} = 3.08$. The *light gray shaded area* indicates inverse transmission ratios with reactive power. The *green shaded areas* determined using (10.86) indicate the operating domain with total efficiency higher than 90%

Due to the huge power recirculation at high transmission ratios, the output-split system is not suitable for start-up. One can tackle this problem by propelling the vehicle at low speeds using only one motor which is connected to the final drive gear via the planetary gear. In this situation, the planetary gear works as a simple fixed gear by reducing its DOF to one, which is constraining its motion. The energy management then switches to the output-split configuration if a certain vehicle speed corresponding to a low transmission ratio is reached. This strategy is applied in GM Volt and Opel Ampera (for more details see Matthé and Eberle [41]).

Again, by applying Euler's second law for the ring gear, sun gear, and carrier gear node of PG1, respectively, and using (10.43)–(10.45), the governing equations of the output power-split system are obtained. They can be summarized as

$$\left(I_{ice} + I_{mg2} + I_{c1}\right)\dot{\omega}_{ice}(t) = T_{ice}(t) + T_{mg2}(t) - (1 + i_{01})T_{i1}(t) \tag{10.102}$$

$$\left(I_{mg1} + I_{s1}\right)\dot{\omega}_{mg1}(t) = T_{mg1}(t) + T_{i1}(t) \tag{10.103}$$

$$\left(I_{veh} + I_{r1}i_{fd}^2\right)\dot{\omega}_{wh}(t) = -(T_{brk}(t) + T_{road}(t)) + i_{01}i_{fd}T_{i1}(t) \tag{10.104}$$

$$\begin{bmatrix} T_{ice}(t) + T_{mg2}(t) \\ T_{mg1}(t) \\ -(T_{brk}(t) + T_{road}(t)) \\ 0 \end{bmatrix} = \begin{pmatrix} I_{ice} + I_{mg2} + I_{c1} & 0 & 0 & 1 + i_{01} \\ 0 & I_{mg1} + I_{s1} & 0 & -1 \\ 0 & 0 & I_{veh} + I_{r1}i_{fd}^2 & -i_{01}i_{fd} \\ 1 + i_{01} & -1 & -i_{01}i_{fd} & 0 \end{pmatrix} \begin{bmatrix} \dot{\omega}_{ice}(t) \\ \dot{\omega}_{mg1}(t) \\ \dot{\omega}_{wh}(t) \\ T_{i1}(t) \end{bmatrix}.$$

Inversion of the left-hand side matrix yields the differential-algebraic equations of the system dynamics

$$\begin{bmatrix} \dot{\omega}_{ice}(t) \\ \dot{\omega}_{mg1}(t) \\ \dot{\omega}_{wh}(t) \\ T_{i1}(t) \end{bmatrix} = \underbrace{\begin{pmatrix} \mathbf{I}_{ops} & \mathbf{D}_{ops}^{[1:3]} \\ \left(\mathbf{D}_{ops}^{[1:3]}\right)^T & 0 \end{pmatrix}^{-1}}_{\mathbf{P}_{ops} = \begin{pmatrix} p_{11} & \cdots & p_{14} \\ \vdots & \ddots & \vdots \\ p_{41} & \cdots & p_{44} \end{pmatrix} \in \mathbb{R}^{4\times 4}} \begin{bmatrix} T_{ice}(t) + T_{mg2}(t) \\ T_{mg1}(t) \\ -(T_{brk}(t) + T_{road}(t)) \\ 0 \end{bmatrix} \tag{10.105}$$

where $\mathbf{I}_{ops} \in \mathbb{R}^{3\times 3}$ from (10.105) is the diagonal inertia matrix

$$\mathbf{I}_{ops} = \operatorname{diag}\left\{I_{ice} + I_{mg2} + I_{c1},\, I_{mg1} + I_{s1},\, I_{veh} + I_{r1}i_{fd}^2\right\}$$

and $\mathbf{D}_{ops} \in \mathbb{R}^{4\times 1}$ is the kinematic constraint matrix of the output power-split system, which describes the static speed constraint from (10.96) as

$$\mathbf{D}_{ops} = \begin{pmatrix} 1 + i_{01} \\ -1 \\ -i_{01} i_{fd} \\ 0 \end{pmatrix}.$$

10.4.2.4 Compound Power-Split

The *compound power-split* (CPS) configuration is the last of the elementary power-split systems. It was invented to achieve a large area of power-split ratio below 30%. Unfortunately, the system alone can neither propel the vehicle from standstill nor provide good efficiencies at high vehicle velocities. The necessary large dimensions of the motors/generators would make this configuration totally unacceptable. It is therefore only applicable in combination with other power-split layouts.

The CPS system consists of two minus planetary gears and two motors/generators. In contrast to the input/output-split systems, none of the electrical machines is directly connected to an input or an output axle. This provides an additional degree of freedom and results in the total number of 2. The configuration is depicted in Fig. 10.19.

The compound power-split configuration must satisfy the rotational speed relationships of the two planetary gears

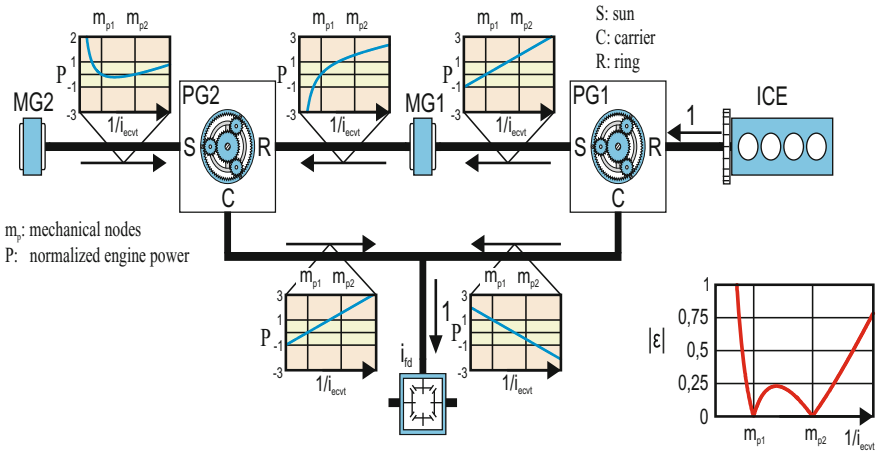


Fig. 10.19 Schematic of the compound power-split configuration. Reactive power results for the inverse transmission ratios lower than the mechanical node $-a_{11}/a_{12}$ and higher than the mechanical node $-a_{21}/a_{22}$

$$0 = -i_{01}\omega_{ice}(t) - \omega_{mg1}(t) + (1 + i_{01})i_{fd}\omega_{wh}(t) \tag{10.106}$$

$$0 = -i_{02}\omega_{mg1}(t) - \omega_{mg2}(t) + (1 + i_{02})i_{fd}\omega_{wh}(t) \tag{10.107}$$

where i_{01} and i_{02} are the stationary gear ratios of planetary gears 1 and 2, respectively.

Putting (10.106) into (10.107) yields the kinematic rotational speed constraints as

$$\begin{bmatrix}\omega_{mg1}(t)\\ \omega_{mg2}(t)\end{bmatrix} = \mathbf{A}_{cps} \cdot \begin{bmatrix}\omega_{ice}(t)\\ \omega_{wh}(t)\end{bmatrix} = \begin{pmatrix}-i_{01} & (1 + i_{01})i_{fd}\\ i_{01}i_{02} & (1 - i_{01}i_{02})i_{fd}\end{pmatrix}\begin{bmatrix}\omega_{ice}(t)\\ \omega_{wh}(t)\end{bmatrix}. \tag{10.108}$$

The kinematic torque constraints can be calculated by transposing and inverting matrix $\mathbf{A}_{cps}$

$$\begin{bmatrix}T_{mg1}(t)\\ T_{mg2}(t)\end{bmatrix} = (-\mathbf{A}_{cps}^{T})^{-1} \cdot \begin{bmatrix}T_{ice}(t)\\ -T_{wh}(t)\end{bmatrix} = \begin{pmatrix}\dfrac{1 - i_{01}i_{02}}{(1 + i_{02})i_{01}} & \dfrac{-i_{02}}{(1 + i_{02})i_{fd}}\\ \dfrac{-1 - i_{01}}{(1 + i_{02})i_{01}} & \dfrac{-1}{(1 + i_{02})i_{fd}}\end{pmatrix}\begin{bmatrix}T_{ice}(t)\\ -T_{wh}(t)\end{bmatrix}. \tag{10.109}$$

Using the definition (10.100) with the first equations of (10.108) and (10.109), we obtain the power-split ratio

$$\epsilon(t) = \left(-i_{01} + \frac{(1 + i_{01})i_{fd}}{i_{ecvt}(t)}\right)\left(\frac{1 - i_{01}i_{02}}{(1 + i_{02})i_{01}} + \frac{i_{02}}{(1 + i_{02})i_{fd}}i_{ecvt}(t)\right). \tag{10.110}$$

Expanding (10.110) and comparing with (10.85) yields

$$\begin{aligned}\epsilon(t) = {} & \frac{-1 + i_{02} + 2i_{01}i_{02}}{1 + i_{02}} - \frac{i_{01}i_{02}}{(1 + i_{02})i_{fd}}i_{ecvt}(t)\\ & + \frac{(1 + i_{01})(1 - i_{01}i_{02})i_{fd}}{(1 + i_{02})i_{01}} \cdot \frac{1}{i_{ecvt}(t)}\end{aligned}$$

with

$$m_0 = \frac{-1 + i_{02} + 2i_{01}i_{02}}{1 + i_{02}}, \quad m_1 = -\frac{i_{01}i_{02}}{(1 + i_{02})i_{fd}}, \quad m_2 = \frac{(1 + i_{01})(1 - i_{01}i_{02})i_{fd}}{(1 + i_{02})i_{01}}.$$

It is easy to observe that the compound-split system has two distinct mechanical nodes at

$$\frac{1}{i_{ecvt,1}} = -\frac{a_{11}}{a_{12}} = \frac{i_{01}}{(1 + i_{01})i_{fd}} \tag{10.111}$$

$$\frac{1}{i_{ecvt,2}} = -\frac{a_{21}}{a_{22}} = \frac{i_{01}i_{02}}{(i_{01}i_{02} - 1)i_{fd}}. \tag{10.112}$$

For the design values $i_{01} = 2.46$, $i_{02} = 1.54$, and $i_{fd} = 2.64$ we obtain the following mechanical nodes:

$$\frac{1}{i_{ecvt,1}} = \frac{2.46}{(1+2.46)\cdot 2.64} \approx 0.27$$
$$\frac{1}{i_{ecvt,2}} = \frac{2.46\cdot 1.54}{(2.46\cdot 1.54-1)\cdot 2.64} \approx 0.51.$$

The power-split ratio of the compound-split system is shown in Fig. 10.20. $\epsilon(\cdot)$ is low if the inverse transmission ratio is located between the two mechanical nodes but increases quickly outside. It is therefore preferable to keep the operation within these nodes.

By applying Euler's second law to the ring gear, sun gear, and carrier gear node of PG1, and the sun gear node of PG2, respectively, the governing equations of the power-split configuration are obtained. They can be summarized as

$$(I_{ice} + I_{r1})\,\dot{\omega}_{ice}(t) = T_{ice}(t) + i_{01}T_{i1}(t) \tag{10.113}$$
$$\left(I_{mg1} + I_{s1} + I_{r2}\right)\dot{\omega}_{mg1}(t) = T_{mg1}(t) + T_{i1}(t) + i_{02}T_{i2}(t) \tag{10.114}$$
$$\left(I_{mg2} + I_{s2}\right)\dot{\omega}_{mg2}(t) = T_{mg2}(t) + T_{i2}(t) \tag{10.115}$$
$$\left(I_{veh} + (I_{c1} + I_{c2})i_{fd}^2\right)\dot{\omega}_{wh}(t) = -T_{brk}(t) - (1+i_{01})i_{fd}T_{i1}(t) - (1+i_{02})i_{fd}T_{i2}(t) - T_{road}(t) \tag{10.116}$$

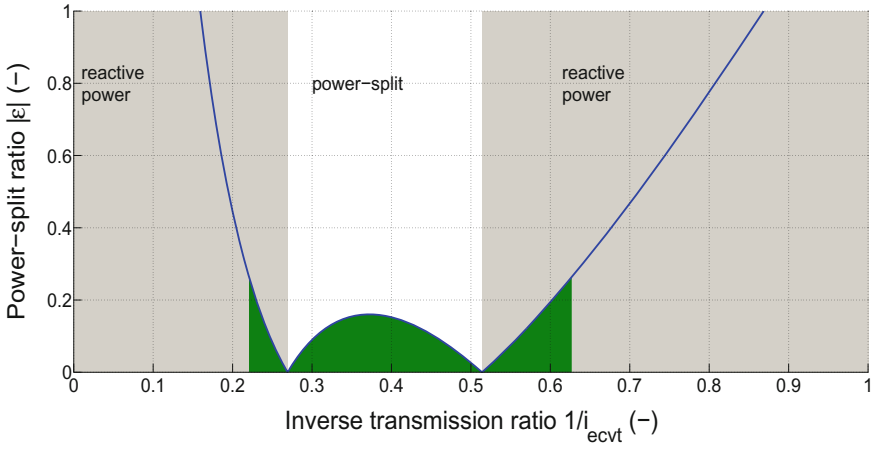


Fig. 10.20 Power ratio ϵ of the compound power-split configuration with $i_{01} = 2.46$, $i_{02} = 1.54$, and $i_{fd} = 2.64$. The *light gray shaded area* indicates inverse transmission ratios with reactive power. The *green shaded areas* determined using (10.86) indicate the operating domain with total efficiency higher than 90%

where I_{r1}, I_{r2}, I_{s1}, I_{s2}, I_{c1}, and I_{c2} are the inertias of the ring gear, sun gear, and carrier gear, respectively. $T_{i1}(\cdot)$ and $T_{i2}(\cdot)$ represent the internal torques on the pinion gears of PG1 and PG2, respectively. To simplify the equations, neither viscous nor Coulomb friction is assumed. The differential-algebraic equations (10.106)–(10.116) represent the system dynamics in the following form

$$\begin{bmatrix} \dot{\omega}_{ice}(t) \\ \dot{\omega}_{mg1}(t) \\ \dot{\omega}_{mg2}(t) \\ \dot{\omega}_{wh}(t) \\ T_{i1}(t) \\ T_{i2}(t) \end{bmatrix} = \underbrace{\begin{pmatrix} \mathbf{I}_{cps} & \mathbf{D}_{cps} \\ \mathbf{D}_{cps}^T & \mathbf{0}_{2\times 2} \end{pmatrix}^{-1}}_{\mathbf{P}_{cps} = \begin{pmatrix} p_{11} & \cdots & p_{16} \\ \vdots & \ddots & \vdots \\ p_{61} & \cdots & p_{66} \end{pmatrix} \in \mathbb{R}^{6\times 6}} \begin{bmatrix} T_{ice}(t) \\ T_{mg1}(t) \\ T_{mg2}(t) \\ -T_{road}(t) - T_{brk}(t) \\ 0 \\ 0 \end{bmatrix} \tag{10.117}$$

where $\mathbf{I}_{cps} \in \mathbb{R}^{4\times 4}$ from (10.117) is the diagonal inertia matrix

$$\mathbf{I}_{cps} = \operatorname{diag}\left\{ I_{ice} + I_{r1},\, I_{mg1} + I_{s1} + I_{r2},\, I_{mg2} + I_{s2},\, I_{veh} + (I_{c1} + I_{c2}) i_{fd}^2 \right\}$$

and $\mathbf{D}_{cps} \in \mathbb{R}^{4\times 2}$ is the kinematic constraint matrix of the compound power-split, which describes the static speed relationships from (10.106) and (10.107) as

$$\mathbf{D}_{cps} = \begin{pmatrix} -i_{01} & 0 \\ -1 & -i_{02} \\ 0 & -1 \\ (1 + i_{01}) i_{fd} & (1 + i_{02}) i_{fd} \end{pmatrix}.$$

10.4.2.5 Two-Mode Power-Split

As the name implies, a *two-mode power-split* (TMPS) configuration supports two operation modes. There are many different topologies reported in the literature. In the schematic shown in Fig. 10.21, the two-mode is a combination of an one-node input-split system and a two-node compound-split system. Two clutches are employed to switch between these systems. The linkage of both elementary systems requires three minus planetary gears which are connected such that the overall system has the same DOF as the compound power-split system.

The system has two modes. The first ECVT mode uses the input-split system. In this mode the third planetary gear is part of the IPS system and must be considered in the rotational speed and torque constraints.

This leads to the speed constraint

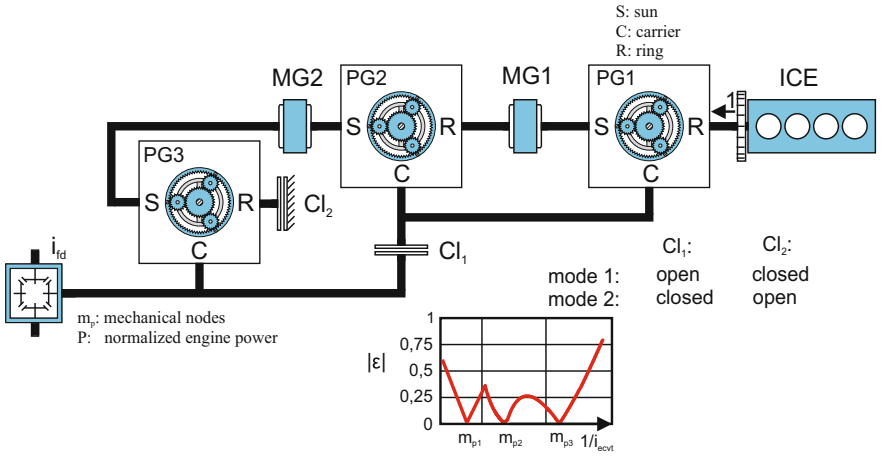


Fig. 10.21 Schematic of the two-mode power-split configuration. Reactive power results at mode 1 for the inverse transmission ratios higher than the mechanical node $-a_{11}/a_{12}$ and at mode 2 for the inverse transmission ratios lower than the mechanical node $-a_{11}/a_{12}$ and higher than the mechanical node $-a_{21}/a_{22}$

$$\begin{bmatrix}\omega_{mg1}(t)\\ \omega_{mg2}(t)\end{bmatrix} = \mathbf{A}_{tmps}\cdot\begin{bmatrix}\omega_{ice}(t)\\ \omega_{wh}(t)\end{bmatrix} = \begin{pmatrix}\dfrac{(1+i_{02})i_{01}}{i_{01}i_{02}-1} & \dfrac{(1+i_{01})(1+i_{03})i_{fd}}{1-i_{01}i_{02}}\\ 0 & (1+i_{03})i_{fd}\end{pmatrix}\begin{bmatrix}\omega_{ice}(t)\\ \omega_{wh}(t)\end{bmatrix}$$

and torque constraint

$$\begin{bmatrix}T_{mg1}(t)\\ T_{mg2}(t)\end{bmatrix} = (-\mathbf{A}_{tmps}^{T})^{-1}\cdot\begin{bmatrix}T_{ice}(t)\\ -T_{wh}(t)\end{bmatrix} = \begin{pmatrix}\dfrac{1-i_{01}i_{02}}{(1+i_{02})i_{01}} & 0\\ \dfrac{-1-i_{01}}{(1+i_{02})i_{01}} & \dfrac{-1}{(1+i_{03})i_{fd}}\end{pmatrix}\begin{bmatrix}T_{ice}(t)\\ -T_{wh}(t)\end{bmatrix}.$$

The second ECVT mode uses the compound power-split system. In this mode the third planetary gear has no mechanical contribution. Thus, the rotational speed and torque constraints from (10.108) and (10.109) can be used without modification.

Using the signed power-split ratio definition (10.90) we obtain

$$\begin{aligned}\epsilon(t) &= -\frac{1-i_{01}i_{02}}{(1+i_{02})i_{01}}\cdot\left(\frac{(1+i_{02})i_{01}}{i_{01}i_{02}-1}+\frac{(1+i_{01})(1+i_{03})i_{fd}}{1-i_{01}i_{02}}\cdot\frac{1}{i_{ecvt}(t)}\right)\\ &= 1-\frac{(1+i_{01})(1+i_{03})i_{fd}}{(1+i_{02})i_{01}}\cdot\frac{1}{i_{ecvt}(t)}.\end{aligned} \tag{10.118}$$

The mechanical node of the first ECVT mode (10.118) over the inverse transmission ratio is given by

$$\frac{1}{i_{ecvt,1}} = -\frac{a_{11}}{a_{12}} = \frac{(1+i_{02})i_{01}}{(1+i_{01})(1+i_{03})i_{fd}}.$$

The mechanical nodes of the second ECVT mode are then given by (10.111) and (10.112). For the design example with $i_{01} = 2.46$, $i_{02} = 1.54$, $i_{03} = 2.72$, and $i_{fd} = 2.64$, following mechanical nodes are obtain:

$$\frac{1}{i_{ecvt,1}} \approx 0.18$$
$$\frac{1}{i_{ecvt,2}} \approx 0.27$$
$$\frac{1}{i_{ecvt,3}} \approx 0.51.$$

The dynamic system description can be derived analogously to (10.95) and (10.117).

One can dedicate mode 1 to the vehicle's low speed range and mode 2 to the higher speed range. A smooth mode change (i.e., no shocks, no vibrations) between input-split and compound-split system is necessary for the success of this configuration. The mode change can be initiated:

- by keeping continuity of $\omega_{mg1}(\cdot)$ and $\omega_{mg2}(\cdot)$ (Villeneuve [64]);
- at a mechanical node: this means, the speed at one side of the variator is equal to zero; and
- using a pair of mutually closed/open brakes (Villeneuve [64]).

The power-split ratio of the two-mode system is shown in Fig. 10.22. It shows that ϵ is low if the inverse transmission ratio is located between the mechanical node 0.18 and 0.51, which is the largest region of all the power splits discussed. Nevertheless, it is foreseeable that even with the best design the two-mode system cannot provide overdrive functionality. General motors introduced a modified version of the two-mode system with fixed gear ratios. This topology, named *Advanced Hybrid System2*, incorporates an additional stationary clutch and an additional rotating clutch that require a complex control strategy. By engaging or disengaging the four clutches, the system realizes six different modes including two ECVT modes and four fixed gear modes.

The general definition of the power-split ratio allows a comparison of the four power-split topologies as shown in Fig. 10.23.

10.4.3 Serial Hybrids

Serial hybrids employ a series connection of two electrical machines without mechanical throughput of the ICE to the wheels as Fig. 10.24 shows. The ICE powers directly a generator (MG1), which supplies the main part of the energy to a traction motor (MG2) and the remaining part to a battery.

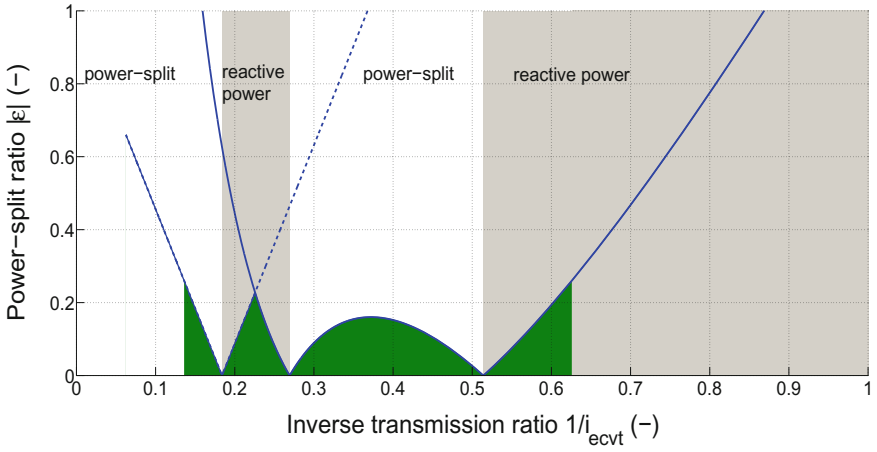


Fig. 10.22 Power ratio ϵ of the two-mode power-split configuration with $i_{01} = 2.46$, $i_{02} = 1.54$, $i_{03} = 2.72$, and $i_{fd} = 2.64$. The *light gray shaded area* indicates inverse transmission ratios with reactive power. The *green shaded areas* determined using (10.86) indicate the operating domain with total efficiency higher than 90%

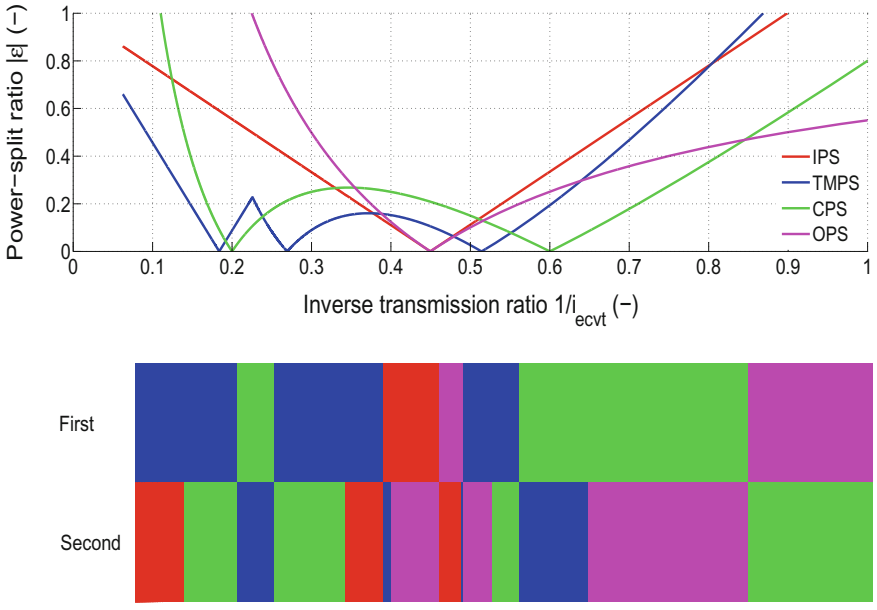


Fig. 10.23 Comparison of power ratios ϵ of different power-split configurations. The *first row* indicates the power-split topologies with the lowest power-split ratios ϵ. The *second row* indicates the power-split topologies with the second lowest power-split ratios ϵ. The configurations are IPS: $i_{01} = 2.6$, $i_{fd} = 3.08$; TMPS: $i_{01} = 2.46$, $i_{02} = 1.54$, $i_{03} = 2.72$, $i_{fd} = 2.64$; CPS: $i_{01} = 1.60$, $i_{02} = 1.36$, $i_{fd} = 3.08$; OPS: $i_{01} = 2.6$, $i_{fd} = 3.08$

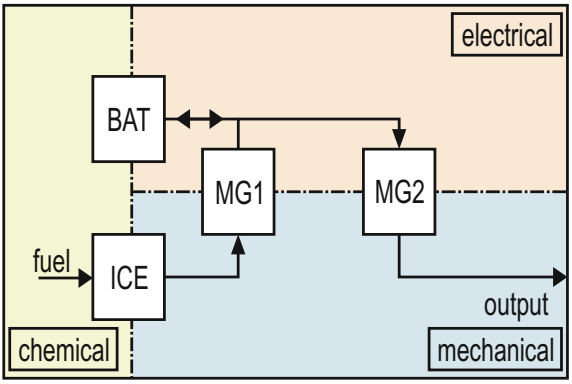


Fig. 10.24 Principle of energy flow in a serial hybrid

Table 10.1 Relationships between configurations and power-split ratios

Configuration	Serial hybrid	Input power-split
Parameter	$a_{12} = a_{21} = 0$ or $a_{11} = a_{22} = 0$	MG2 directly connected to the drivetrain, i.e., $a_{11} = 0$ or $a_{21} = 0$
Power-split ratio $\epsilon(\cdot)$	1	$\pm 1 + \frac{m_2}{i_{ecvt}(t)}$

A serial hybrid can be imagined as a power-split configuration with an extreme power-split ratio of $\epsilon(t) \equiv 1$ (see Table 10.1). Using the definitions from the power splits, the rotational speed is constrained to

$$\begin{bmatrix} \omega_{mg1}(t) \\ \omega_{mg2}(t) \end{bmatrix} = \mathbf{A}_{se} \cdot \begin{bmatrix} \omega_{ice}(t) \\ \omega_{wh}(t) \end{bmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & i_{fd} \end{pmatrix} \begin{bmatrix} \omega_{ice}(t) \\ \omega_{wh}(t) \end{bmatrix} \tag{10.119}$$

and the torque is constrained to

$$\begin{bmatrix} T_{mg1}(t) \\ T_{mg2}(t) \end{bmatrix} = (-\mathbf{A}_{se}^T)^{-1} \cdot \begin{bmatrix} T_{ice}(t) \\ -T_{wh}(t) \end{bmatrix} = \begin{pmatrix} -1 & 0 \\ 0 & -\frac{1}{i_{fd}} \end{pmatrix} \begin{bmatrix} T_{ice}(t) \\ -T_{wh}(t) \end{bmatrix}. \tag{10.120}$$

Using (10.119) and (10.120) the signed power-split ratio is then defined by

$$\epsilon(t) = -\frac{P_{mg1}(t)}{P_{ice}(t)} = \frac{T_{ice}(t)\omega_{ice}(t)}{T_{ice}(t)\omega_{ice}(t)} = 1$$

and is constant.

Because of the complete mechanical decoupling, serial hybrids provide the greatest flexibility in choosing their IC engine operation regimes independently from the driver's request. The IC engine provides the power to the generator and can there-

Table 10.2 Relationships between configurations and power-split ratios (continue)

Configuration	Output power-split	Compound power-split
Parameter	MG1 and ICE are directly connected to the drivetrain, i.e., $a_{12} = 0$ or $a_{22} = 0$	none of the MGs are directly connected to the drivetrain, i.e., $a_{11} \neq 0$, $a_{12} \neq 0$, $a_{21} \neq 0$, and $a_{22} \neq 0$
Power-split ratio $\epsilon(\cdot)$	$\pm 1 + m_1 i_{ecvt}(t)$	$m_0 + m_1 i_{ecvt}(t) + \dfrac{m_2}{i_{ecvt}(t)}$

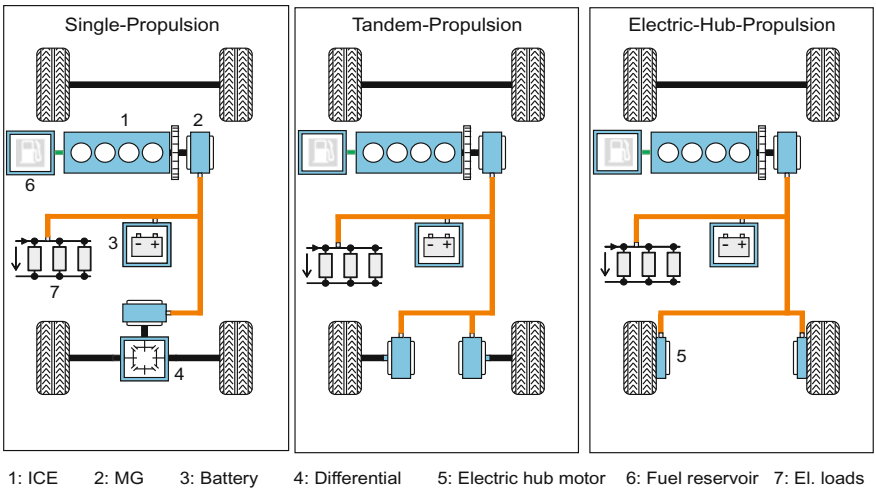


Fig. 10.25 Configurations of serial hybrids. **a** left layout: single traction motor with differential, **b** middle layout: two traction motors per axle without differential, and **c** right layout: electric wheel hubs

fore be dynamically, stationarily, or intermittently operated to meet some specific performance, economy, or emission targets.

There exist variants of serial hybrids with single traction motor and differential (Fig. 10.25a), concepts with two traction motors per axle saving a mechanical differential gear (Fig. 10.25b), and concepts with electric wheel hubs (Fig. 10.25c).

There are also variants for the left layout possible with one or more reduction gears between the MG2 and the differential.

The aim of energy management of serial hybrids is twofold: on the one hand to operate the ICE stationary as long as possible in its best operating points (i.e., with one or multiple operating points along the best engine efficiencies) as transients can cause large emission peaks; on the other to maximize the efficiency of the overall powertrain.

Operating in the best ICE points may result in larger ICE powers as required to propel the vehicle. In such driving scenarios, the battery is used to buffer the superfluous energy. Conversely, the superfluous energy must be consumed eventually.

Such a combination of battery charge and discharge represents a *duty-cycle* operation, which is typical of serial hybrid vehicles. This duty-cycle operation increases the battery losses and thus should be limited.

The dynamic system description is given by two ODEs

$$\begin{aligned}(I_{ice} + I_{mg1})\dot{\omega}_{ice}(t) &= T_{ice}(t) + T_{mg1}(t)\\ (I_{veh} + I_{mg2}i_{fd}^2)\dot{\omega}_{wh}(t) &= i_{fd}T_{mg2}(t) - T_{brk}(t) - T_{road}(t).\end{aligned}$$

In matrix notation

$$\begin{bmatrix} T_{ice}(t) + T_{mg1}(t) \\ i_{fd}T_{mg2}(t) - T_{brk}(t) - T_{road}(t) \end{bmatrix} = \underbrace{\begin{pmatrix} I_{ice} + I_{mg1} & 0 \\ 0 & I_{veh} + I_{mg2}i_{fd}^2 \end{pmatrix}}_{\mathbf{I}_{se}\in\mathbb{R}^{2\times 2}} \begin{bmatrix} \dot{\omega}_{ice}(t) \\ \dot{\omega}_{wh}(t) \end{bmatrix}.$$

Inversion of the left-hand side matrix yields the dynamic system as

$$\begin{aligned}\begin{bmatrix} \dot{\omega}_{ice}(t) \\ \dot{\omega}_{wh}(t) \end{bmatrix} &= \mathbf{I}_{se}^{-1} \cdot \begin{bmatrix} T_{ice}(t) + T_{mg1}(t) \\ i_{fd}T_{mg2}(t) - T_{brk}(t) - T_{road}(t) \end{bmatrix}\\ &= \begin{pmatrix} \dfrac{1}{I_{ice} + I_{mg1}} & 0 \\ 0 & \dfrac{1}{I_{veh} + I_{mg2}i_{fd}^2} \end{pmatrix} \begin{bmatrix} T_{ice}(t) + T_{mg1}(t) \\ i_{fd}T_{mg2}(t) - T_{brk}(t) - T_{road}(t) \end{bmatrix}.\end{aligned} \tag{10.121}$$

One can observe that (10.121) contains no algebraic constraints.

Serial hybrids are typically used in heavy-duty vehicles such as trucks and locomotives (Miller [42]), whereas serial hybrids for passenger cars have only been demonstrated as prototypes or in combination with other powertrain concepts, e.g., GM Volt (Matthé and Eberle [41]).

10.4.4 Combined Hybrids

A combined hybrid is mostly a parallel hybrid configuration but with features from a serial hybrid. The powertrain of the combined hybrid employs two electric machines. One electric machine is connected on the engine's output shaft and acts as a generator to provide the electrical traction power or to charge the battery while the other is used as a traction motor or generator for regenerative braking. For the case that MG1 works as generator and MG2 as motor, the combined hybrid operates in powersplit mode with $\epsilon(t) > 0$ and $\epsilon(t) << 1$. A schematic of the energy flow of a combined hybrid configuration is depicted in Fig. 10.26.

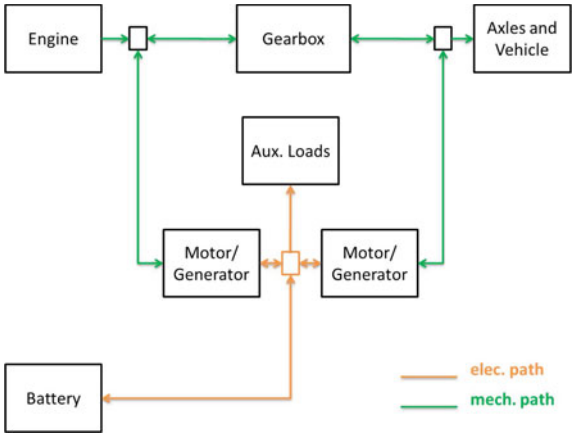


Fig. 10.26 Energy flow of a combined hybrid configuration

At urban speed range, the purely electric drive mode is activated. If the state of charge of the battery is too low, the IC engine kicks in and provides the traction power including the recharge power for the battery.

In contrast to a serial hybrid configuration, a combined hybrid offers the option of directly transferring a part of the mechanical power of the IC engine to the wheels by engaging a clutch. Such a parallel mode is especially important at operating conditions with a high power demand to reduce the ϵ and thus to improve the overall efficiency.

In some configurations, depending on the installed power of both electric machines, both motors can be powered to provide an electric 4-wheel-drive car. The electrical energy is provided either by the battery or the IC engine in a manner similar to the serial concept. An example is the Porsche 918 Spyder with a totally installed electrical power of 180 kW and a maximal combustion power of 426 kW [48].

10.4.5 Plug-In Hybrids

A *plug-in hybrid electric vehicle* (PHEV) differs from a charge-sustaining HEV in the usage of a high-capacity energy storage system with the ability to recharge independently of the vehicle utilization and driving profile through the connection to an electric power grid. While PHEVs need far less battery capacity than BEVs, they will likely need at least five times the battery capacity of todays charge-sustaining HEVs. In general, PHEVs can be realized using parallel, power-split, serial, or combined hybrid configurations but with usually larger electrical prime movers. Nevertheless, power-split PHEV configurations may suffer from the drawback of not being able to decouple the ICE from the drivetrain. Under this circumstance, a drag torque always applies.

The power grid connection introduces an additional DOF compared with conventional hybrids by controlling the major energy flow from either the battery or the IC engine. This introduces the operating modes:

- charge-depleting;
- charge-sustaining; and
- charge-blended.

The charge-depleting mode refers to a purely electric drive mode that allows the depletion of the battery's state of charge down to a certain threshold. The all-electric-range depends therefore on the size of the battery.

The charge-sustaining mode is used to maintain the state of charge of the battery during the drive cycle. This mode is already used in conventional hybrids. Therefore, conventional hybrids are sometimes called *charge-sustaining hybrids*.

As proposed by many authors, among them Zhang et al. [71], a predefined strategy can be employed by selecting the charge-depleting operation mode at vehicle start until the battery's state of charge has been depleted to a certain threshold and then transitioning to the charge-sustaining operation mode. This PHEV operating strategy is probably the most frequent implementation in practice. However, in terms of system efficiency and the smaller sizes of electric traction systems the charge-blended mode has been regarded as more efficient. In the charge-blended mode, the IC engine can be turned on depending on the system's efficiency, which requires a more complex control strategy.

10.4.6 Battery Electric Vehicles

Purely electric propulsion systems such as battery electric vehicles are characterized by an electric energy flow only. This zero emission vehicle consists of a high-density battery and an electric traction motor. The high-energy batteries are usually based on lithium technology and the energy density ranges from 30 to 80 Wh/kg. The top speed of the newly released VW UP reaches 130 km/h and the range is more than 120 km per charge [65].

Many arguments suggest that the most suitable application of BEVs is in use of urban contexts, especially within car-sharing organizations.

The powertrain layout of a BEV can be simply derived by removing the fuel path in the serial hybrid configuration (see Fig. 10.24). The rotational speed and torque constraints (10.119) and (10.120) reduce to

$$\omega_{mg}(t) = i_{fd}\omega_{wh}(t)$$
$$T_{mg}(t) = \frac{1}{i_{fd}} \cdot T_{wh}(t).$$

The dynamic system description is then given by one ODE of the form

$$\dot{\omega}_{wh}(t) = I_{bev}^{-1} \cdot \left[i_{fd} T_{mg}(t) - T_{brk}(t) - T_{road}(t) \right]$$

where $I_{bev} \in \mathbb{R}$ is the inertia of the BEV and is given as $I_{bev} = (I_{veh} + I_{mg} i_{fd}^2)$.

10.5 Hybrid Vehicle Models

During the operation of hybrid vehicles, continuous-valued controls as well as discrete decisions can be made to achieve a desired target. Continuous-valued controls may include, for instance, a large set of parameters of the internal combustion engine, such as ignition angle, throttle, crank-shaft positions and many others. Discrete decisions can imply the gear choice of an automatic transmission, different clutch-states as well as several discrete parameters in the operation of the ICE, such as the activation of the charge-motion-valve. A model, containing all these decisions and their effects on the system, would be very expensive in terms of computation time and would require a large set of information that is usually not easily available in the early stages of the automotive calibration process. It is therefore advisable, to select the controls and states needed in a model and to define the required depth of the model carefully. In most cases, the basic operation parameters of the ICE will be well defined before the calibration process of energy management for hybrid vehicles begins. Consequently, these parameters can be assumed as given and the required model dimension is significantly reduced.

In this section, the mechatronic submodels constructed from the previous sections will be blended together to describe four major models for hybrid vehicles. All models are quasi-static in terms of some dynamics:

- the first model is based on the quasi-steady fuel relationship and uses the torque split in the hybrid vehicle between ICE and MG as continuous-valued control input and a drive mode (hybrid or pure electric) as well as the gear selection as discrete control input. This model allows for the optimization of these parameters under the assumption that the vehicle is warmed up;
- the condition that the vehicle is warmed up cannot always be assumed. Especially over the rather short cycles applied for homologation purposes, the thermodynamic influence especially on the ICE cannot be entirely disregarded, since the heat-up process amounts for a significant part of the entire drive cycle. Additionally, some constraints that largely depend on thermodynamic conditions apply. Especially noxious emissions are strongly limited by the diverse legislations and cannot be disregarded in the calibration process. Therefore, the second model includes the most important thermodynamic states and has the ignition angle as an additional control variable to control the heat-up of the three-way catalytic converter;
- the third model applies the same model depth as the first model but for a power-split hybrid; and

- the last model builds upon the first model but introduces an additional state (coolant water temperature) and an additional continuous-valued control variable (mechanical brake torque).

10.5.1 Quasi-static Model for Parallel Hybrids

The following model is described for a parallel hybrid vehicle architecture with a N_{gbx}-speed automatic gearbox, as can be seen in Fig. 10.12. The proposed models have in common the quasi-static relationships between the interacting torques and a simple short-circuit analogy for modeling the electrical subsystem. The highly nonlinear efficiencies for the MG and ICE are stored with blended splines or tensor-product splines (see Sect. 10.7).

The parallel configuration allows different operating modes such as purely electric drive, electric launch, engine load shifting, engine torque assist, and regenerative braking to be selected. If the clutch is commanded to be open, electric driving, launching, and regenerative braking are possible without the drag torque of the engine. Conversely, if the clutch is engaged, the engine and the MG together provide the torque required by the driver. Therefore, one can cast the numerous operating modes to a clutch-dependent drive mode represented as

$$\zeta(t) = \begin{cases} 0, & \text{pure electric drive mode (clutch open)} \\ 1, & \text{hybrid drive mode (clutch closed).} \end{cases}$$

For vehicles employing an automatic gearbox, the gear numbers are enumerated in the set $K = \{1, 2, \ldots, N_{gbx}\}$. The active gear at time t is given by the discrete function $\kappa(t) \in K$. Consequently, the discrete decisions at time t can be identified by the function

$$q(t) = N_{gbx} \cdot \zeta(t) + \kappa(t),$$

that assigns an unique value $q(t) \in \Theta = \{1, 2, \ldots, 2 \cdot N_{gbx}\}$ to every possible combination of gear and drive mode.

Assuming that the driver follows exactly a given vehicle speed trajectory $v(t) > 0\,, \forall t$. Then, using the longitudinal vehicle dynamics from Sect. 10.2 the torque due to acceleration can be calculated by

$$T_a(t) = \tilde{\gamma}_m \cdot m \cdot r_{wh} \cdot \dot{v}(t).$$

The wheel torque

$$T_{wh}(t) = T_a(t) + T_{road}(t) \tag{10.122}$$

can then be determined directly from the vehicle speed trajectory. Using the governing equation of the P2 driveline (10.75), the torque $T_{gbx}(\cdot)$ and angular speed $\omega_{gbx}(\cdot)$ on the input side of the gearbox are obtained depending on the selected gear:

$$\begin{aligned} T_{gbx,\kappa(t)}(t) &= \frac{1}{i_t(\kappa(t))} \cdot [T_{wh}(t) + T_{loss}(\kappa(t), T_{wh}(t), \omega_{wh}(t)) + T_{brk}(t)] \\ \omega_{gbx,\kappa(t)}(t) &= i_t(\kappa(t))\omega_{wh}(t). \end{aligned}$$

Here, the torque loss is a function of $T_{loss}(\kappa(t), T_{wh}(t), \omega_{wh}(t))$. At any time t, the gearbox input torque must be provided in sum by ICE and MG. That means,

$$T_{gbx,\kappa(t)}(t) = T_{mg,q(t)}(t) + T_{ice}(t). \tag{10.123}$$

Please note, that the dynamic part of (10.65) is ignored for simplicity reasons. The torque split between MG and ICE in (10.123) provides a continuous degree of freedom that can be used as control input to the system. The definition of the control $u(t) \in \mathsf{U}$ is herein somewhat arbitrary and hence the definition

$$u(t) = T_{ice}(t) \tag{10.124}$$

is made. During pure electric drive mode, the ICE is disconnected from the powertrain by a clutch and switched off. In this case (10.124) becomes

$$u(t) = T_{ice}(t) \equiv 0.$$

The corresponding speeds are

$$\begin{aligned} \omega_{mg,\kappa(t)}(t) &= \omega_{gbx,\kappa(t)}(t) \\ \omega_{ice,q(t)}(t) &= \begin{cases} 0 & \zeta(t) = 0 \\ \omega_{gbx,\kappa(t)}(t) & \zeta(t) = 1. \end{cases} \end{aligned}$$

Thus, the set of admissible continuous-valued controls can be defined as

$$\hat{\mathcal{U}}(q(t), t) = \begin{cases} 0, & \zeta(t) = 0 \\ \left\{ u(t) \in \mathsf{U} \;\middle|\; T_{ice}^{min}(\omega_{ice,q(t)}(t)) \le u(t) \le T_{ice}^{max}(\omega_{ice,q(t)}(t)) \right\}, & \zeta(t) = 1. \end{cases}$$

The battery power $P_{bat,q(t)}(\cdot)$ (10.49) also depends on the discrete decision

$$P_{bat,q(t)}(u(t)) = -P_{mg}(T_{mg,q(t)}(t), \omega_{mg,\kappa(t)}(t)) - P_{aux}$$

where P_{aux} is assumed to be constant and $T_{mg,q(t)}(\cdot)$ can be obtained from the torque constraint (10.123). $P_{mg}(\cdot)$ is represented by a smooth map. Therefore, one finds that

$$\dot{\xi}(t) = \frac{1}{Q_{bat}} \cdot \frac{-V_{oc}(\xi(t)) + \sqrt{V_{oc}^2(\xi(t)) + 4R_{bat}\, P_{bat,q(t)}(u(t))}}{2R_{bat}}$$
$$= \frac{1}{Q_{bat}} \cdot I_{bat,q(t)}(\xi(t), u(t)). \tag{10.125}$$

The vehicle model for backward simulation can now be concatenated from the continuous-valued states of battery's state of charge and fuel consumption

$$\mathbf{x}(t) = \begin{bmatrix} \beta(t) \\ \xi(t) \end{bmatrix}, \qquad \mathbf{x}(t_0) = \mathbf{x}_0 = \begin{bmatrix} 0 \\ \xi_0 \end{bmatrix}$$

where the initial state ξ_0 is predefined. From the differential equations (10.125) and (10.15) the switching character can easily be observed so that the system notation follows definitions made in Chap. 3

$$\mathcal{M}_1 : \dot{\mathbf{x}}(t) = \begin{bmatrix} \dot{\beta}(t) \\ \dot{\xi}(t) \end{bmatrix} = \mathbf{f}_{q(t)}(\mathbf{x}(t), u(t)) \quad \forall q \in \Theta. \tag{10.126}$$

The system (10.126) consists of q subsystems, where each subsystem has one continuous-valued control input and two continuous-valued states. The vector field of the subsystems is defined by

$$f_{1,q(t)} := \gamma_f \cdot \zeta(t) \cdot \mathrm{bsfc}(u(t), \omega_{ice,q(t)}(t)) \cdot u(t) \cdot \omega_{ice,q(t)}(t), \ \forall q \in \Theta$$
$$f_{2,q(t)} := \frac{1}{Q_{bat}} \cdot I_{bat,q(t)}(x_2(t), u(t)), \ \forall q \in \Theta.$$

For optimization purposes the *backward simulation approach* where the vehicle motion is treated as quasi-steady and thus the vehicle velocity is not be assumed as a system state, but considered as a time-dependent fixed input. This helps to further reduce the dimension of the model and makes the execution faster than forward simulation models. Backward simulation means that a drive cycle is assumed in advance and the torque and speed required at the wheel are calculated based on the predefined drive cycle. Therefore, the driver behavior is not modeled. Very similar models were derived by Wei [69], Guzzella and Sciarretta [20], de Jager et al. [27], and others. Instead, the forward simulation approach mimics the causality of events that take place in a real vehicle and requires the modeling of the complex behavior of human beings. Model predictive control strategies are obvious candidates for this task.

For the consideration of a vehicle's energy management, a start of the IC engine can be assumed to be executed within one time-instant without loss of accuracy for most vehicles. An engine start will require some additional torque of ICE and/or MG whereas an engine stop is usually performed by cutting of fuel injection and therefore no additional energy is needed for the stop, nor is any kinetic energy recuperated. The additional energy for the start is modeled with the jump-function

$$\boldsymbol{\delta}_{(\zeta(t_j^-),\zeta(t_j^+))}(\mathbf{x}(t_j^-)) := \begin{cases} [\Delta\beta,\ \Delta\xi]^T, & \zeta(t_j^-) = 0 \wedge \zeta(t_j^+) = 1 \\ \mathbf{0}, & \text{otherwise,} \end{cases}$$

where $\Delta\beta$ and $\Delta\xi$ are the respective measured jumps that result from an engine start. It should be noted that the model $\mathcal{M}_1$ is appropriate only for electrical start-up of the vehicle. For simulation of start-stop systems with less or even no e-drive capability the models have to be adapted. See therefore Koprubasi [31].

10.5.2 Thermodynamic Model for Parallel Hybrids Using Spark Ignition Engines

The temperature evolution is a crucial quantity for all mechatronic components in the hybrid powertrain. The evolution depends on the thermal masses of the components and the connected cooling system. In terms of emissions, the TWC should be heated-up to the light-off temperature as fast as possible to ensure its converting capability but with the smallest air flow throughput, since a cold TWC is almost inactive. The models described in the previous sections are based on a general quasi-static engine description and are therefore valid for hybrid vehicles employing gasoline, diesel, or any other engine type. Nevertheless, the rudimentary description implies that these models are only valid for a heated-up engine with cooling water temperatures of well above 330 K. The heating-process itself cannot be further investigated using this simplified model. Therefore, in this section, a much more detailed SI model is described that also incorporates a thermodynamic model (Schori et al. [56], see Fig. 10.27). In the beginning of any drive cycle, certain attention needs to be paid to heating the TWC in the exhaust system since the TWC operates with acceptable efficiency only above a given temperature threshold. A common measure to achieve a quick heat-up is the retardation of the ignition angle, which leads to higher exhaust enthalpies at the cost of lower combustion efficiencies. To allow for very late ignition angles, a *homogeneous-split injection scheme* (HIS) is commonly used (van Basshuysen [2]), as opposed to the *standard injection scheme* (SIS). Additional DOF during TWC heating come from the states of the clutches C0 and C1. In contrast to the model in the previous section, the ICE is not necessarily switched off, when C0 is opened

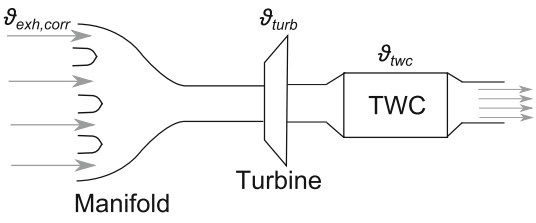


Fig. 10.27 Sketch of the elements in the exhaust system regarded in the model

Table 10.3 Discrete decisions made at any time t for a given value $q(\cdot)$

$q(\cdot)$	C0	C1	ICE on/off	Injection scheme
1	Closed	Open	On	HIS
2	Open	Closed	On	HIS
3	Closed	Closed	On	HIS
4	Open	Closed	Off	None
5	Closed	Closed	On	SIS

but the engine can be operated in idling mode. The overall discrete decisions are summarized in Table 10.3.

The modes for $q(t) \in \{1, 2, 3\}$ are modes designed specifically for TWC heating, the mode $q(t) = 4$ is the purely electric drive mode, and $q(t) = 5$ denotes the conventional hybrid drive mode. Consequently, the discrete decisions at time t are uniquely identified by $q(t) \in \Theta = \{1, 2, 3, 4, 5\}$.

To reduce the already very high model complexity, the backward simulation approach is used and the gearbox is not regarded in the system description. Instead, measured or calculated trajectories of $\omega_{gbx}(\cdot)$ and $T_{gbx}(\cdot)$ serve as time-dependent inputs.

For the engine speed

$$\omega_{ice,q(t)}(t) = \begin{cases} 0, & q(t) \in \{4\} \\ \omega_{idle}, & q(t) \in \{1, 2\} \\ \omega_{gbx}(t), & q(t) \in \{3, 5\} \end{cases}$$

applies and for the MG speed

$$\omega_{mg,q(t)}(t) = \begin{cases} \omega_{idle}, & q(t) \in \{1\} \\ \omega_{gbx}(t), & q(t) \in \{2, 3, 4, 5\} \end{cases}$$

where ω_{idle} is a constant angular speed at engine idling. Again, the gearbox input torque has to be supplied in sum by ICE and MG and hence the condition

$$T_{gbx}(t) = T_{ice}(t) + T_{mg}(t)$$

holds. The vector of continuous-valued control inputs comprises the relative cylinder charge $m_{cyl}(\cdot)$ and the ignition angle $\chi(\cdot)$:

$$\mathbf{u}(t) = \begin{bmatrix} m_{cyl}(t) \\ \chi(t) \end{bmatrix}.$$

The convex sets of admissible continuous-valued controls are defined as

$$\hat{\mathcal{U}}(q(t),t) =$$
$$\left\{ \mathbf{u}(t) \in \mathsf{U} \;\middle|\; \begin{bmatrix} m_{cyl}^{min}(\omega_{ice,q(t)}(t)) \\ \chi_{q(t)}^{min}(m_{cyl}(t),\omega_{ice,q(t)}(t)) \end{bmatrix} \leq \mathbf{u}(t) \leq \begin{bmatrix} m_{cyl}^{max}(\omega_{ice,q(t)}(t)) \\ \chi_{q(t)}^{max}(m_{cyl}(t),\omega_{ice,q(t)}(t)) \end{bmatrix} \right\}.$$

For HIS injection, the ignition angle can usually be retarded much more, such that the set of admissible controls is larger for $q(t) \in \{1,2,3\}$. For electric drive mode $\hat{\mathcal{U}}(q(t),t) = \{0\}$ applies.

Based on these control variables, the engine output torque $T_{ice}(\cdot)$ is formed as follows: An optimal ignition angle is given by the smooth functions $g_1(\cdot)$ and $g_2(\cdot)$, depending on whether HIS or standard injection is active. Throughout this section, smooth mappings will be denoted as $g_i(\cdot)$

$$\chi_{opt,q(t)}(t) = \begin{cases} g_1(m_{cyl}(t),\omega_{ice,q(t)}(t)), & q(t) \in \{1,2,3\} \\ g_2(m_{cyl}(t),\omega_{ice,q(t)}(t)), & q(t) \in \{5\} \end{cases}.$$

Applying the optimal ignition angle would yield a theoretically optimal engine torque

$$T_{\chi_{opt},q(t)}(t) = g_3(m_{cyl}(t),\omega_{ice,q(t)}(t)).$$

Deviating from the optimal ignition angle leads to a decrease in combustion efficiency

$$\eta_{\mathrm{d}\chi,q(t)}(t) = g_4(\chi_{opt,q(t)}(t) - \chi(t))$$

which in turn reduces the inner engine torque

$$T_{\chi,q(t)}(t) = T_{\chi_{opt},q(t)}(t) \cdot \eta_{\mathrm{d}\chi,q(t)}(t).$$

The engine output torque is then obtained by subtracting the temperature-dependent frictional torque $T_l(\cdot)$

$$T_{ice,q(t)}(t) = T_{\chi,q(t)}(t) - T_l(\vartheta_{cw}(t)).$$

Herein, the cooling water temperature $\vartheta_{cw}(\cdot)$ is used to express the temperature dependence. In general, the oil temperature would be a better measure to express the internal friction loss but this would require the introduction of an additional state. Assuming a constant air-fuel ratio λ_f, the fuel volume flow is proportional to the air mass flow $\dot{m}_{air}(\cdot)$ passing the cylinder, which gives

$$\dot{\beta}(t) = \frac{1}{\lambda_f} \cdot \gamma_f \cdot \dot{m}_{air}(m_{cyl}(t),\omega_{ice,q(t)}(t)).$$

Again, γ_f is a product of different natural constants. The electrical subsystem is modeled completely analogously to the preceding sections with the simple circuit model. The thermodynamics of the system are modeled using a system of three temperature states

$$\vartheta(t) = \begin{bmatrix} \vartheta_{cw}(t) \\ \vartheta_{cyl}(t) \\ \vartheta_{twc}(t) \end{bmatrix}$$

describing the temperatures of the cooling water, cylinder and manifold, and TWC. The intermediate state $\vartheta_{cyl}(\cdot)$ is a state that combines the wall temperatures of several elements in the exhaust system, for instance cylinder, outlet valve, and manifold. Modeling each pipe element in the exhaust system with a separate state as usually done in commercial engine-simulation packages (e.g., GT-Power from Gamma Technologies) would again lead to a very high system dimension, which is to be avoided. The raw exhaust temperature is given by a mapping

$$\vartheta_{exh,q(t)}(t) = g_5(m_{cyl}(t), \omega_{ice,q(t)}(t)).$$

Retarding the ignition angle $\chi(\cdot)$ leads to an increase of the exhaust temperature and the respective correction factors obtained from measurements are given by $g_6(\cdot)$ for HIS and by $g_7(\cdot)$ for standard injection:

$$\vartheta_{exh,corr,q(t)}(t) = \begin{cases} \vartheta_{exh,q(t)}(t) \cdot g_6(m_{cyl}(t), \omega_{ice,q(t)}(t)), & q(t) \in \{1,2,3\} \\ \vartheta_{exh,q(t)}(t) \cdot g_7(m_{cyl}(t), \omega_{ice,q(t)}(t)), & q(t) \in \{5\}. \end{cases}$$

Owing to temperature losses to the cylinder wall, exhaust valves, and exhaust manifold wall, the gas temperature in the manifold is reduced to

$$\vartheta_{man,q(t)}(t) = \vartheta_{exh,corr,q(t)}(t) - p_1 \cdot (\vartheta_{exh,corr,q(t)}(t) - \vartheta_{cyl}(t))$$

and the evolution of the temperature state $\vartheta_{cyl}(\cdot)$ is governed by the nonlinear differential equation

$$\dot{\vartheta}_{cyl}(t) = p_2 \cdot (\vartheta_{exh,corr,q(t)}(t) - \vartheta_{cyl}(t)) - p_3 \cdot (\vartheta_{cyl}(t) - \vartheta_{cw}(t)).$$

Here and in the following, the parameters p_i include heat capacities, heat transfer coefficients, and natural constants. Further, the gas temperature in the exhaust system is reduced in the turbine and the reduction is described by

$$\vartheta_{turb,q(t)}(t) = g_8(\dot{m}_{exh}(t), \vartheta_{man,q(t)}(t)) \cdot \vartheta_{man,q(t)}(t).$$

It is assumed that the injected fuel has only a minor effect on the exhaust mass flow and therefore $\dot{m}_{exh} = \dot{m}_{air}$ holds. Finally, the catalytic converter temperature can be modeled by the nonlinear differential equation

$$\dot{\vartheta}_{twc}(t) = p_4 \cdot (\vartheta_{turb,q(t)}(t) - \vartheta_{twc}(t)) - p_5 \cdot (\vartheta_{twc}(t) - \vartheta_{amb}(t)).$$

Exothermic reactions caused by unburned hydrocarbon also play a role in the temperature increase, when the TWC-light-off temperature is reached. As we are mostly interested in the heating behavior, before the light-off temperature is reached, exothermic reactions are not considered. The time derivative of the cooling water temperature is given by the energy flow balance

$$\dot{\vartheta}_{cw}(t) = p_6 \cdot \left[P_{fuel,q(t)}(t) - P_{ice,q(t)}(t) - \dot{\tau}_{exh,q(t)}(t) - \dot{\tau}_{amb}(t) \right], \qquad (10.127)$$

where $P_{fuel,q(t)}(\cdot)$, $\dot{\tau}_{exh,q(t)}(\cdot)$, and $\dot{\tau}_{amb}(\cdot)$ denote the energy flows due to combustion, losses to the exhaust and to the environment, respectively. $P_{ice,q(t)}(\cdot)$ is the mechanical power of the ICE. The quantities of (10.127) are given by

$$\begin{aligned} P_{fuel,q(t)}(t) &= H_l \dot{m}_{fuel,q(t)}(t) \\ \dot{\tau}_{exh,q(t)}(t) &= p_7 \dot{m}_{exh}(t) \vartheta_{man,q(t)}(t) \\ \dot{\tau}_{amb}(t) &= p_8 \cdot (\vartheta_{cw}(t) - \vartheta_{amb}(t)) \\ P_{ice,q(t)}(t) &= T_{ice}(t) \omega_{ice,q(t)}(t) \end{aligned}$$

where $\dot{m}_{fuel,q(t)}(\cdot)$ denotes the fuel mass flow.

The modeling of noxious emissions has been a growing research area for many years. On the one hand, the calculation times of most detailed emission models are still too high for optimization purposes, however. On the other hand, it is well known that quasi-steady map-based models do not provide sufficient accuracy to achieve quantitatively reliable results (Silva et al. [58]). As a consequence, artificial states Z_i are introduced that resemble the emission components at least qualitatively. For every emission component $i \in \{1, \ldots, E\}$, a state governed by the differential equation

$$\dot{Z}_i(t) = \dot{m}_{e,i}(m_{cyl}(t), \omega_{ice,q(t)}(t), q(t)) \cdot (1 - \eta_{conv}(\vartheta_{twc}(t)))$$

with the initial state $Z_i(t_0) = 0$ is added to the system description. The function $\dot{m}_{e,i}(m_{cyl}(t), \omega_{ice,q(t)}(t), q(t))$ is a map of raw emissions that may additionally depend on the injection scheme applied and η_{conv} is the temperature-dependent conversion efficiency of the TWC. The conversion efficiency can be approximated by an arcus tangent function as shown by Kum et al. [32]:

$$\eta_{conv}(\vartheta_{twc}(t)) = \frac{1}{\pi} \cdot \left(\arctan\left(\frac{\vartheta_{twc}(t) - \vartheta_{lo}}{s_1} \right) + \frac{\pi}{2} \right).$$

The parameters ϑ_{lo} and s_1 are the light-off temperature and a fitting parameter, respectively. The average conversion efficiency of the emission component i in the interval $[t_0, t]$ can be expressed by

$$\eta_{twc}(t) = 1 - \frac{Z_i(t)}{m_{e,i}(t)}.$$

The overall model is then given by the hybrid system and consists of 5 subsystems

$$\mathcal{M}_2 : \dot{\mathbf{x}}(t) = \begin{bmatrix} \dot{\beta}(t) \\ \dot{\xi}(t) \\ \dot{\vartheta}_{cw}(t) \\ \dot{\vartheta}_{cyl}(t) \\ \dot{\vartheta}_{twc}(t) \\ \dot{Z}_1(t) \\ \vdots \\ \dot{Z}_E(t) \\ \dot{m}_{e,1}(t) \\ \vdots \\ \dot{m}_{e,N_e}(t) \end{bmatrix} = \mathbf{f}_{q(t)}(\mathbf{x}(t), u(t)), \ \forall q \in \Theta. \tag{10.128}$$

The heat-up procedure can be modeled with high accuracy. At higher temperatures $\vartheta_{twc} \gg 550\,\mathrm{K}$, the exothermic reactions in the catalytic converter have a significant impact on the TWC temperature. The modeling of these chemical reactions, however, is rather difficult. Since, up to this temperature, the heat-up process itself is of the biggest interest, the model is still sufficiently reliable.

10.5.3 Quasi-static Model for Power-Split Hybrids

The power-split configurations allow operating modes similar to parallel configurations without the necessity of decoupling the ICE from the planetary gear set. This simplifies the modeling but requires the casting of the operating modes depending on the injection command as

$$\zeta(t) = \begin{cases} 0, & \text{pure electric mode (injection off)} \quad (q = 1) \\ 1, & \text{hybrid mode (injection on)} \quad (q = 2). \end{cases}$$

For this reason we have two discrete decisions and the subsystems are uniquely identified by $q(t) \in \Theta = \{1, 2\}$.

Inserting the wheel speed differential equation from the dynamical system (IPS: (10.95), OPS: (10.105), CPS: (10.117), etc.) into (10.70) yields the wheel torque, which is now dependent on ICE, MG1, and MG2, respectively.

For instance, considering the compound power-split hybrid and putting (10.117) into (10.70) yields

$$T_{wh}(t) = I_{veh} \cdot \left(p_{41} T_{ice}(t) + p_{42} T_{mg1}(t) + p_{43} T_{mg2}(t) \right) \tag{10.129}$$

where $p_{44} = 1/I_{veh}$ is obtained by a comparison of coefficients. Rearranging (10.129) for $T_{mg2}(\cdot)$ and inserting the result into the engine speed differential equation (10.117) yields the rearranged engine speed differential equation as

$$\dot{\omega}_{ice}(t) = \left(p_{11} - p_{13}\frac{p_{41}}{p_{43}}\right) T_{ice}(t) + \left(p_{12} - p_{13}\frac{p_{42}}{p_{43}}\right) T_{mg1}(t) - \\ + \frac{p_{13}}{p_{43} I_{veh}} \cdot T_{wh}(t) - p_{14} \cdot (T_{road}(t) + T_{brk}(t)). \tag{10.130}$$

The reader should note that the p_{ij} parameters are obtained from the matrix $\mathbf{P}_{cps}$.

The continuous-valued controls $\mathbf{u}(\cdot)$ are defined on the control space U. The first continuous-valued control $u_1(\cdot)$ is defined again as

$$u_1(t) = T_{ice}(t)$$

for the hybrid drive mode. During purely electric drive mode, the fuel injection of the ICE is stopped but the engine is still connected on the planetary gear shaft. In this case, the drag torque of the ICE

$$u_1(t) = T_{ice}(t) = T_{ice}^{drag}(t)$$

applies. The second control $u_2(\cdot)$ is defined as

$$u_2(t) = T_{mg1}(t).$$

For the case, that both motor/generators are not sufficient for braking, the optional third continuous-valued control $u_3(\cdot)$ can be defined as

$$u_3(t) = T_{brk}(t).$$

The input control vector can then be written as

$$\mathbf{u}(t) = \begin{bmatrix} T_{ice}(t) \\ T_{mg1}(t) \\ T_{brk}(t) \end{bmatrix}. \tag{10.131}$$

The sets of admissible continuous-valued controls are then defined as

$$\hat{\mathcal{U}}(q(t), t) = \begin{cases} \left\{ \mathbf{u}(t) \in \mathsf{U} \;\middle|\; \begin{bmatrix} T_{ice}^{drag}(\omega_{ice}(t)) \\ T_{mg1}^{min}(\omega_{ice}(t)) \\ 0 \end{bmatrix} \le \mathbf{u}(t) \le \begin{bmatrix} T_{ice}^{drag}(\omega_{ice}(t)) \\ T_{mg1}^{max}(\omega_{ice}(t)) \\ T_{brk}^{max} \end{bmatrix} \right\}, & \zeta(t) = 0 \\ \left\{ \mathbf{u}(t) \in \mathsf{U} \;\middle|\; \begin{bmatrix} T_{ice}^{min}(\omega_{ice}(t)) \\ T_{mg1}^{min}(\omega_{ice}(t)) \\ 0 \end{bmatrix} \le \mathbf{u}(t) \le \begin{bmatrix} T_{ice}^{max}(\omega_{ice}(t)) \\ T_{mg1}^{max}(\omega_{ice}(t)) \\ T_{brk}^{max} \end{bmatrix} \right\}, & \zeta(t) = 1. \end{cases}$$

The first box constraint shows that $u_1(\cdot)$ is not usable for the drive mode $\zeta(\cdot) = 0$.

The battery's state of charge now depends on two continuous-valued controls $u_1(\cdot)$ and $u_2(\cdot)$

$$\dot{\xi}(t) = \frac{1}{Q_{bat}} \cdot \frac{-V_{oc}(\xi(t)) + \sqrt{V_{oc}^2(\xi(t)) + 4R_{bat}\, P_{bat,q(t)}(\mathbf{u}(t))}}{2R_{bat}} \tag{10.132}$$

with

$$P_{bat,q(t)}(\mathbf{u}(t)) = -P_{mg1}(T_{mg1,q(t)}(t), \omega_{mg1}(t)) - P_{mg2}(T_{mg2,q(t)}(t), \omega_{mg2}(t)) - P_{aux}$$

where P_{aux} is assumed to be constant. $T_{mg2,q(t)}(\cdot)$ can be obtained from $\mathbf{u}(\cdot)$ and the torque constraint (10.129) whereas $\omega_{mg1}(\cdot)$ and $\omega_{mg2}(\cdot)$ can be obtained from the dynamic system, e.g., compound power-split (10.117).

The vehicle model for backward simulation can now be concatenated from the continuous states of engine speed, battery's state of charge, and fuel consumption

$$\mathbf{x}(t) = \begin{bmatrix} \beta(t) \\ \xi(t) \\ \omega_{ice}(t) \end{bmatrix}, \qquad \mathbf{x}(t_0) = \mathbf{x}_0 = \begin{bmatrix} 0 \\ \xi_0 \\ \omega_{ice_0} \end{bmatrix}$$

where the initial states ω_{ice_0} and ξ_0 are predefined. Using the differential equations (10.15), (10.130), and (10.132) the switched system can be formulated as

$$\mathcal{M}_3 : \dot{\mathbf{x}}(t) = \begin{bmatrix} \dot{\beta}(t) \\ \dot{\xi}(t) \\ \dot{\omega}_{ice}(t) \end{bmatrix} = \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)), \quad q(t) \in \Theta. \tag{10.133}$$

The system (10.133) consists of two subsystems, where each subsystem has at least two continuous-valued control inputs and three continuous states. The vector field is defined as

$$\begin{aligned}
f_{1,q(t)} &:= \gamma_f \cdot \zeta(t) \cdot \mathrm{bsfc}(u_1(t), x_3(t)) \cdot u_1(t) \cdot x_3(t), \quad \forall q(t) \in \Theta \\
f_{2,q(t)} &:= \frac{1}{Q_{bat}} \cdot I_{bat,q(t)}(x_2(t), u_1(t), u_2(t)), \quad \forall q(t) \in \Theta, \\
f_{3,q(t)} &:= \left(p_{11} - p_{13}\frac{p_{41}}{p_{43}}\right) \cdot u_1(t) + \left(p_{12} - p_{13}\frac{p_{42}}{p_{43}}\right) \cdot u_2(t) \\
&\quad + \frac{p_{13}}{p_{43} I_{veh}} T_{wh}(t) - p_{14}\left(T_{road}(t) + u_3(t)\right), \quad \forall q(t) \in \Theta.
\end{aligned}$$

The fixed trajectories $T_{wh}(t)$ and $T_{road}(t)$ are determined from the drive cycle.

10.5.4 Extended Quasi-static Model for Parallel Hybrids

The quasi-steady engine model (10.19)–(10.20) from Sect. 10.5.1 can be extended to include the coolant water state. Including the ODE (10.127) yields the hybrid system

$$\mathcal{M}_4 : \dot{\mathbf{x}}(t) = \begin{bmatrix} \dot{\beta}(t) \\ \dot{\xi}(t) \\ \dot{\vartheta}_{cw}(t) \end{bmatrix} = \mathbf{f}_{q(t)}(\mathbf{x}(t), \mathbf{u}(t)), \ \forall q(t) \in \Theta \tag{10.134}$$

with

$$\mathbf{x}(t) = \begin{bmatrix} \beta(t) \\ \xi(t) \\ \vartheta_{cw}(t) \end{bmatrix}, \qquad \mathbf{x}(t_0) = \mathbf{x}_0 = \begin{bmatrix} 0 \\ \xi_0 \\ \vartheta_{cw_0} \end{bmatrix}$$

where the vector of continuous-valued controls is defined by

$$\mathbf{u}(t) = \begin{bmatrix} T_{ice}(t) \\ T_{brk}(t) \end{bmatrix}.$$

Thus, the set of admissible continuous-valued controls can then be defined as

$$\hat{\mathcal{U}}(q(t), t) = \begin{cases} \left\{ \mathbf{u}(t) \in \mathsf{U} \,\middle|\, \begin{bmatrix} 0 \\ 0 \end{bmatrix} \le \mathbf{u}(t) \le \begin{bmatrix} 0 \\ T_{brk}^{max} \end{bmatrix} \right\}, & \zeta = 0 \\ \left\{ \mathbf{u}(t) \in \mathsf{U} \,\middle|\, \begin{bmatrix} T_{ice}^{min}(\omega_{ice,q(t)}(t)) \\ 0 \end{bmatrix} \le \mathbf{u}(t) \le \begin{bmatrix} T_{ice}^{max}(\omega_{ice,q(t)}(t)) \\ T_{brk}^{max} \end{bmatrix} \right\}, & \zeta = 1. \end{cases}$$

10.6 Drive Cycles

For standardized assessment of the vehicle's energy economy and pollutant emissions several test procedures with associated drive (test) cycles are available. These drive cycles consist of speed and elevation profiles, sometimes with gear selection instruction, and differ according to de Jager et al. [27] with respect to the vehicle type (light or heavy-duty), the vehicle use-case (e.g., short distances with typical urban speed or long-range distances with high contributions of highway sections), the regional homologation bodies (Europe, Japan, USA, etc.), and dynamic or static operating regimes.

Relevant standard emissions certification tests are

- *motor vehicle emission group* (MVEG) (see Fig. 10.28). Also known as *new European drive cycle* (NEDC). This drive cycle is supposed to represent a typical

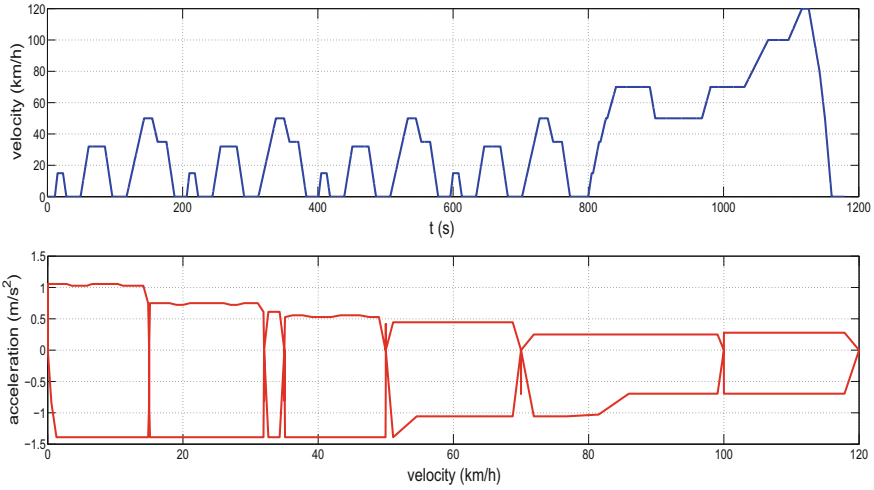


Fig. 10.28 MVEG test cycle

vehicle usage in Europe and consists of urban, suburban, and highway parts. Since 1970, this cycle is used for emission certification in the European Union;

- *assessment and reliability of transport emission models and inventory systems* (ARTEMIS) (see Fig. 10.30). This European drive cycle is based on statistical analysis of real world driving patterns (Andre [1]). Analogously to MVEG drive cycle it consists of clearly separated urban, suburban, and highway part;
- *federal test procedure* (FTP-72) or (FTP-75). The FTP-72 cycle is depicted in Fig. 10.29. The FTP drive cycles have been used for emission certification in the United States since 1975 and are based on measured real drive cycles;
- *urban dynometer driving schedule* (UDDS) is equivalent to FTP-72;
- supplemental federal test procedure (US06) was additionally developed to address the shortcomings with the FTP-75 cycle concerning aggressive and high-speed driving;
- 10–15 mode is the official drive cycle for energy and emission certification in Japan. This cycle has a relative low average velocity of 22.7 km/h and is especially designed for urban traffic flows in Japan (cf. Khajepour et al. [29]); and
- *worldwide harmonized light vehicles test procedures* (WLTP) (see Fig. 10.31). A globally harmonized drive cycle for light-duty vehicles. Final version is not yet released.

These tests are carried out in controlled environments (temperature, humidity, etc.), with strict procedures being followed to reach precisely defined thermal initial conditions for the (hybrid) vehicle.

Ideally, the drive cycles will have been constructed in such a way that they provide a realistic approximation of the actual conditions vehicles encounter on the road. However, this is not always possible because the emission's certification tests must

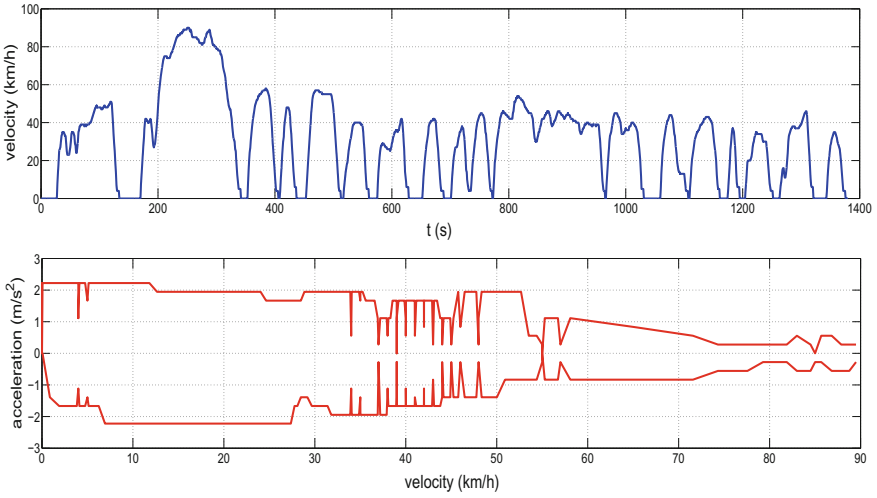


Fig. 10.29 FTP-72 or UDDS test cycle

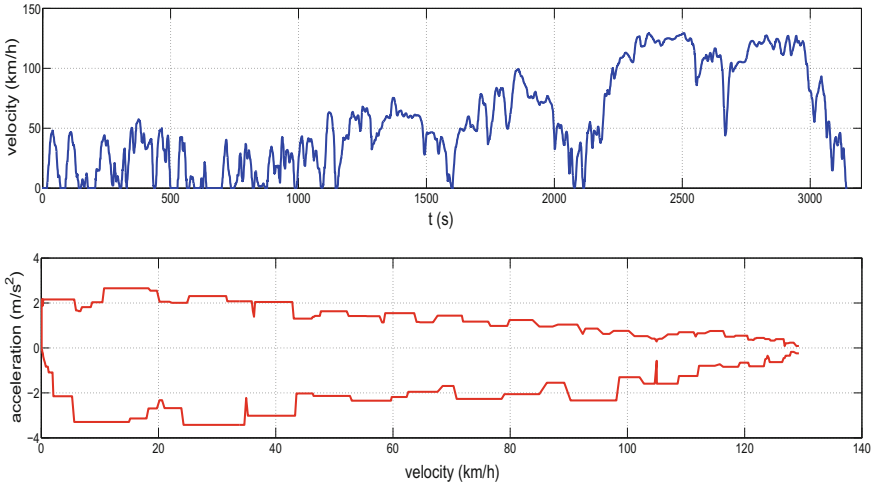


Fig. 10.30 ARTEMIS test cycle

have tight boundaries to ensure that results from different vehicles can be directly compared, and that all vehicles sold in a given market are held to the same standards. This situation has led that vehicle manufactures consider their regional characteristics (e.g., varying topologies, acceleration profiles, stop probabilities, etc.) as realistic use-cases. One such regional test cycle, specific to the area around the cities of Wolfsburg and Braunschweig (both in Germany) as depicted in Fig. 10.32, with a high contribution of urban and rural parts is shown in Fig. 10.33. The acceleration

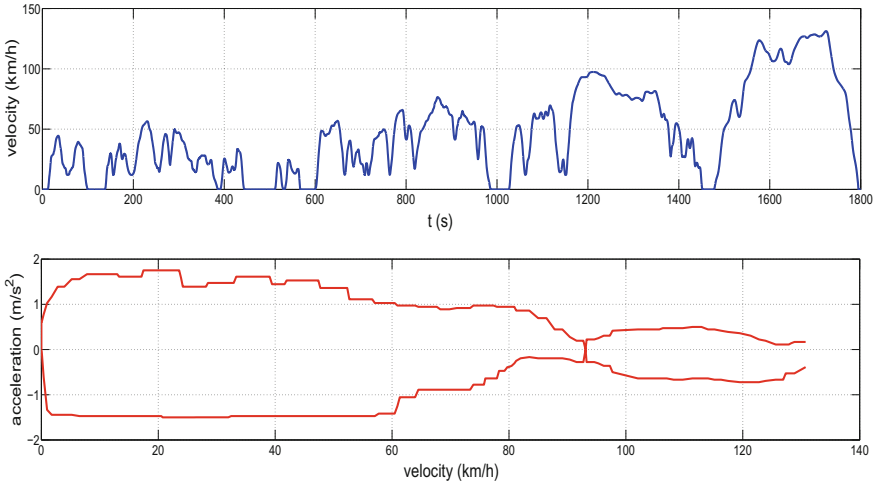


Fig. 10.31 WLTP test cycle

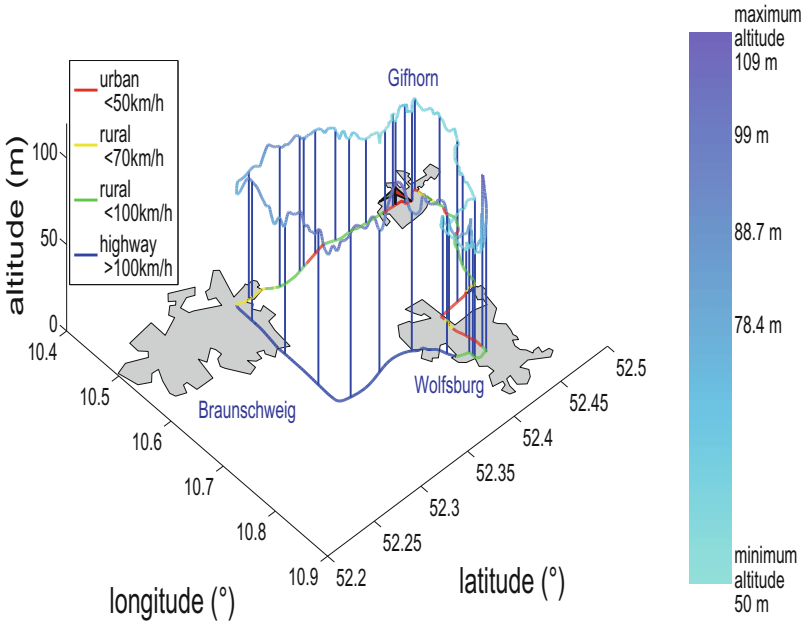


Fig. 10.32 Route map of the real-world benchmark cycle 1

spectrum is obviously larger compared with homologation specific drive cycles in order to represent better long-range driving profiles.

Some important statistical measures of time-dependent drive cycles can be obtained by a time-average analysis as proposed by Guzzella and Sciarretta [20].

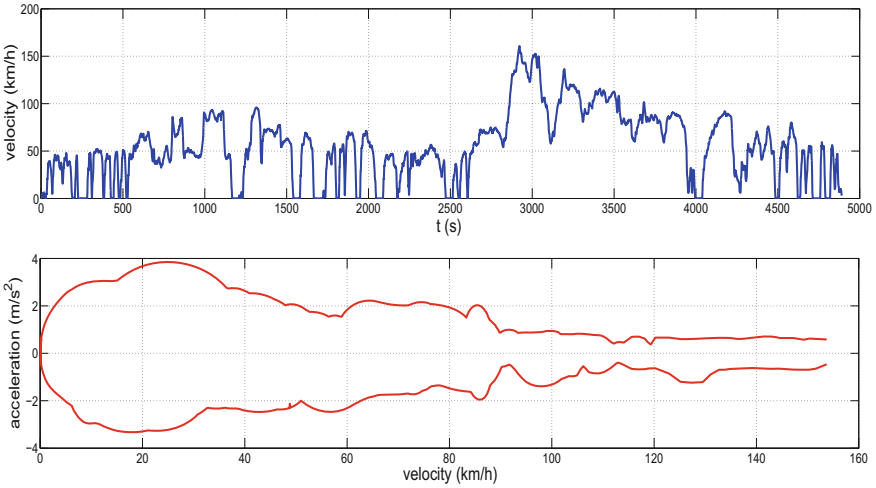


Fig. 10.33 Real-world benchmark cycle 1

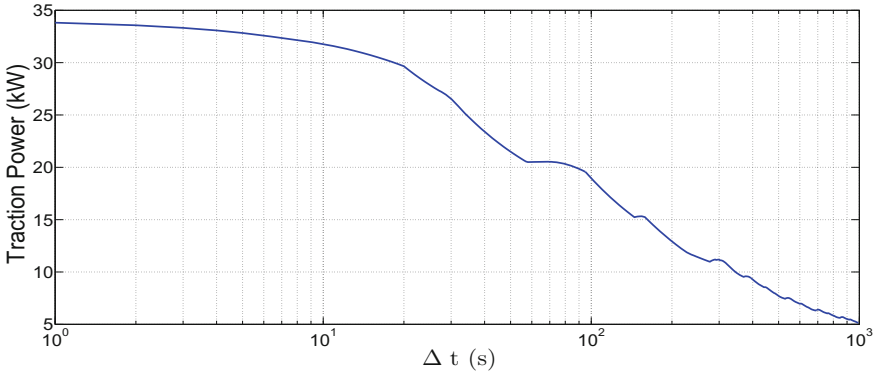


Fig. 10.34 Time-average diagram of the MVEG test cycle

A time-average analysis computes mean values for different time-window sizes Δt according to

$$\bar{P}(\Delta t) = \max_t \left\{ \frac{1}{\Delta t} \int_t^{t+\Delta t} P(\tau)\, d\tau \right\}.$$

For the two extreme cases: $\lim_{\Delta t \to 0} \bar{P}(\Delta t) = \bar{P}(0)$ and $\lim_{\Delta t \to \infty} \bar{P}(\Delta t) = \bar{P}(\infty)$ one obtains the **maximum value** and the **mean value** of the drive cycle, respectively.

Figures 10.34, 10.35, 10.36, 10.37 and 10.38 show the evolution of the time-average analysis of the drive cycles depicted in Figs. 10.28, 10.29, 10.30, 10.31, 10.32, and 10.33. These profiles can be consulted by sizing tasks of the powertrain components.

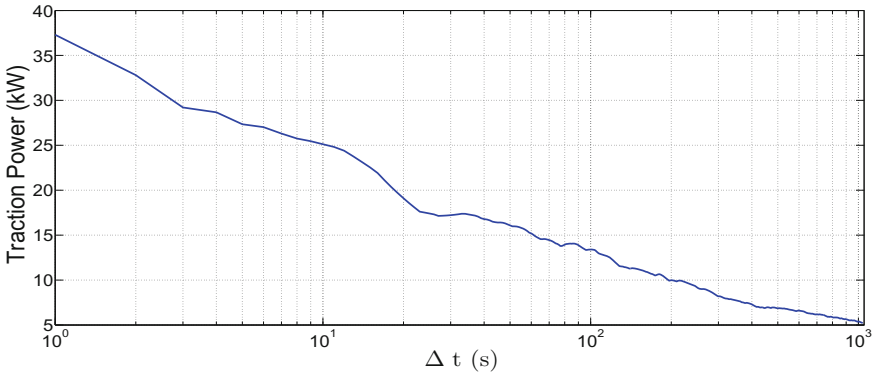


Fig. 10.35 Time-average diagram of the FTP-72 test cycle

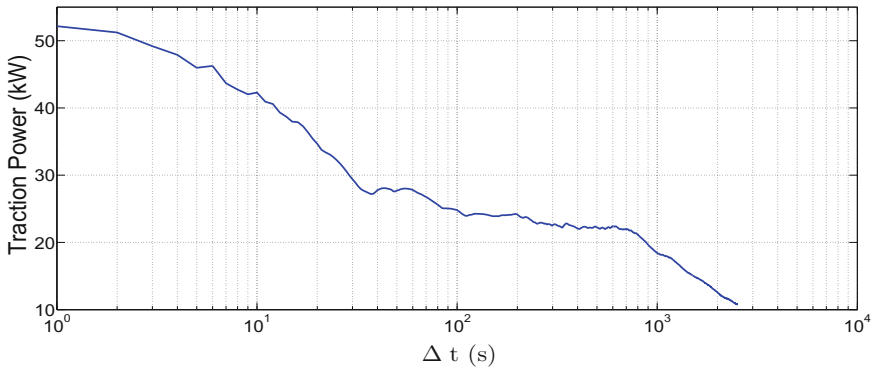


Fig. 10.36 Time-average diagram of the ARTEMIS test cycle

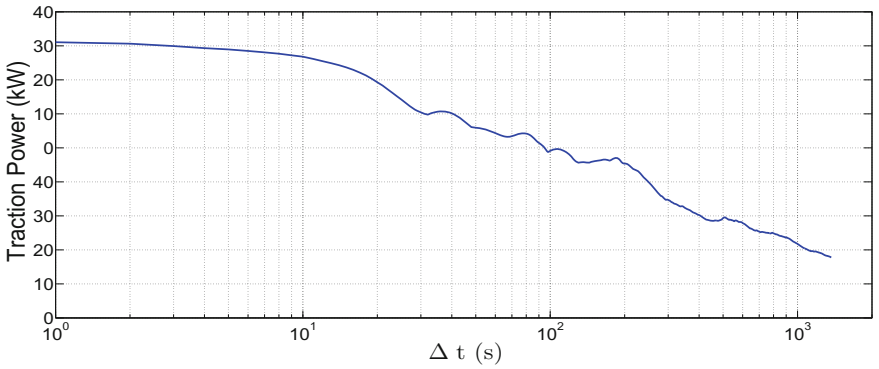


Fig. 10.37 Time-average diagram of the WLTP test cycle

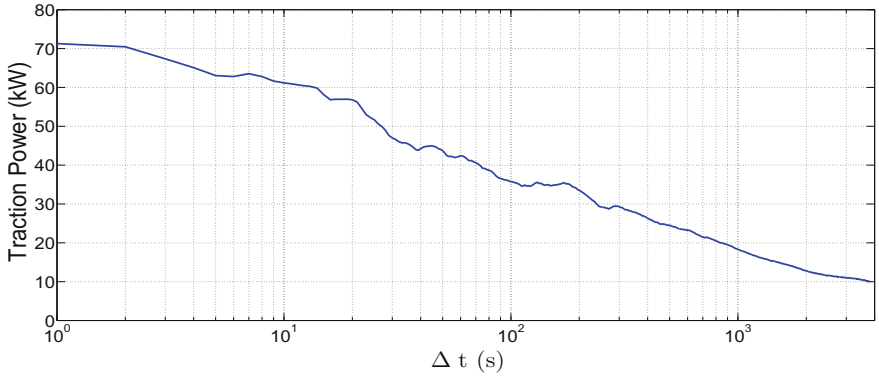


Fig. 10.38 Time-average diagram of the Wolfsburg test cycle

10.7 Static Function Representation

The efficiencies or power losses of individual powertrain elements are usually determined from measurements and need to be stored in some form for later evaluation in the optimization procedure. These representations of static nonlinear functions are commonly stored in tabular functions in ECUs. However, tabular functions are totally inappropriate for SQP because of the only piecewise differentiability. It is of major importance that a function $g(\cdot)$ is sufficiently smooth to ensure that the gradients of the objective function or the constraints in the optimal control problem formulation can be calculated in a predictable way, whereas in nonsmooth problems, kinks and jumps may occur. Therefore, a function $g(\cdot)$ should have the following properties:

- The function $g(\cdot)$ should be twice continuously differentiable in the direction for which gradients are needed;
- The optimization procedure needs a high amount of function evaluations. Consequently, the evaluation needs to be fast; and
- Measured data are usually subject to noise, which might prevent convergence of the optimization. The function should smooth the measured data up to a desired extent.

Common techniques for the representation of function $g(\cdot)$ are interpolating least squares (Boehme et al. [7]), blended splines, and tensor-product splines (Boehme et al. [9]).

10.8 Switching Costs

Optimization of switched systems may result in frequent switchings that have to be limited for some obvious reasons. For instant, switching of the electro-mechanical clutch reduce the life span of this device by contact wear and tear. It is therefore a

natural idea to penalize too many switchings by including an appropriate additional cost term to the cost function $\phi(\cdot)$ (see Stewart [59], Passenberg et al. [46]).

Three ways to account for switching costs are common:

- **introducing state jumps**: cost values related to the switching-energy can be added to some of the physical states which mimics an average energy consumption at each switching;
- **penalty of the number of switchings**: an additional cost term related to number of switchings can be added to the cost function; and
- **penalty of switching arc-lengths**: an additional cost term related to the switching arc-lengths can be added to the cost function.

The first method approach is motivated by selecting energy-based states and adding penalty terms to these states in form of instantaneous state jumps at the switchings. A higher number of switchings causes a higher energy consumption and hopefully the optimization procedure reduces the number of switchings. However, this penalty method introduces discontinuities in the state trajectories and leads thus to a nondifferentiable optimization problem. Nonlinear programming methods, such as SQP, may fail to find a solution, if the objective function or the constraints are not continuously differentiable everywhere. This failure can have several reasons (Lemaréchal [36], Sagastizábal [53]), among them are:

- the objective function can be poorly approximated by smooth methods when encountering points where the gradient does not exist and close to these points;
- stopping tests, that implement the norm of the gradient at a current iterate as stopping criterion, cannot be used since the gradient might not exist at a KKT-point; and
- calculating derivatives by finite differencing may yield poor approximations.

In the second approach, the number of switchings is penalized by adding a total variation term, which measures the changes in the discrete control signal. Minimization of the total variation results in a minimization of the switchings (see Loxton et al. [39]).

The third approach penalizes small switching arcs more directly. An additional cost term related to the switching arc-lengths is added to the cost function. The challenge of this methodology is to use a differentiable cost term which is a function of the switchings arc-lengths. Such a cost term requires a reformulation of the optimal control problem to a *switching time optimization* (STO) with parameterized switching intervals as shown in Sect. 8.3.3.

A penalization can be realized by the inclusion of an additional term of the following form

$$\tilde{\phi} = \gamma_l \sum_{j=1}^{N_{t_3}-1} \exp\left(-\frac{1}{2} \cdot \left(\frac{\varsigma_j - \frac{c}{2}}{\frac{0.9545}{2} - \frac{c}{4}}\right)^2\right) \tag{10.135}$$

to the cost function

$$\breve{\phi}\left(\mathbf{u}(\cdot)\right) = \phi\left(\mathbf{u}(\cdot)\right) + \tilde{\phi},$$

where γ_l is a constant weighting factor that increases monotonically by each iteration l. Equation (10.135) represents a modified Gaussian bell-shaped curve with its maximum at $c/2$. Arc-lengths which are smaller than c should be either forced to zero or c by the nonlinear programming solver. The parameter c is chosen, in general, larger than the smallest switching arc.

In order to save computing time, the first STO can be computed with the halved desired accuracy. Every re-optimization due to penalizing and filtering of small switching arcs should then be carried out with the desired numerical precision. Optionally, many "state-of-the-art" sequential quadratic programming solvers allow to provide the exact Hessian which can reduce drastically the number of iterations.

The filtering of small switching arcs is very important for the success of STOs. Connected switching arcs with the same values of the binary control functions $\boldsymbol{\sigma}(\cdot)$ can be filtered by simply removing a certain number of these small switching arcs. The filtering procedure can be repeated which increases the iteration counter l. However, for every filtering step the STO has to be solved again. This is necessary since a strong filtering of the solution trajectories might result in bad initial conditions for the succeeding optimization and finally to a badly conditioned STO which can be poorly solvable or is even infeasible. It is therefore crucial to remove for every filtering step only a small number of short switching arcs. The filtering approach can be seen as an opposite method to interval insertion proposed by several authors, among them Kaya and Noakes [28].

10.9 Bibliographical Notes

Several textbooks are available on the subject of hybrid vehicles, among them Miller [42], Guzzella and Sciarretta [20], Hofmann [22], and Ehsani et al. [16]. An overview paper of modeling methods for hybrid vehicles is presented in Rizzoni et al. [51]. A detailed treatment of modeling principles for mechatronic systems is given in the textbook from Isermann [26]. Topics about numerical integration and model representation forms are significant for simulation success and are discussed in Gao et al. [17]. One is the *resistive companion form*. This method originates from electrical engineering but is also suitable for modeling hybrid powertrains. Using the resistive companion modeling technique, one can obtain high-fidelity physics-based models of each mechatronic subsystem in modular format.

There is a wealth of literature available for treatment of high-complexity models for feature-rich vehicle simulations, e.g., Gopal and Rousseau [19], Halbach et al. [21], but less material for optimization suitable models with low complexity but good fidelity.

Feature-rich vehicle models for forward-oriented simulation may be developed in different languages and interact with different numerical solvers. Such models arise typically from a product development process, e.g., "V-chart," with many cooperating disciplines. Experts on IC engines may use GT-Power whereas experts of transmissions may prefer AMESim. Such models may also include additional hardware prototypes of selected components. Some simulation packages to cope with these challenges are commercially available. Notably, the *powertrain system analysis toolkit* is a MATLAB®/Simulink®-based simulation software developed by Argonne National Laboratory (Gao et al. [17]) and *advanced vehicle simulator* (ADVISOR) developed by the National Renewable Energy Laboratory (Markel et al. [40]). A further development is *Autonomie*, optimized for legacy code reuse. These simulation environments are examples of quasi-static HEV simulators.

Verdonck et al. [63] presented a methodology to transform forward-oriented dynamic models automatically to backward-oriented quasi-static counterparts.

Hybrids with fuel cell systems have been extensively considered by Ward et al. [68].

A hybrid vehicle model for drivability and stability problems has been investigated by Koprubasi [31]. The proposed longitudinal dynamics capture low-to-mid frequency dynamics to provide more insight to the relevant effects that have impact on the stability, drivability, and handling of a vehicle. Mixed-bandwidth HEV model is discussed by Waltermann [66]. Here, the design of a vehicle stability controller for a serial HEV is supported by a simplified longitudinal drivetrain model which is combined with detailed lateral dynamics.

Earlier studies on power-split mechanisms can be traced back to the 1970s referring to the work by Gelb [18]. The prehistory of power-split-device development can be found in the work of Miller and Everett [43]. A comparison of different single and two-mode ECVT operating modes can be found in Conlon [13] and Boehme et al. [5]. Liu [37] studied the automatic generation of power-split layouts based on a universal format. More complex planetary gearboxes with more than 4 coaxial shafts (e.g., Ravigneaux wheelset) is considered in Koprubasi [31].

Batteries are the Achilles heel of every electrified powertrain. Recently, many research has been undertaken to investigate the life span of batteries. Generally, battery aging benefits from a multitude of factors. In Wang et al. [67], aging tests have shown a strong correlation of capacity fade to current rates and temperature. It is therefore obvious to reduce these stress factors by applying an appropriate control scheme. A quantity which can be related to battery aging is the state of health. Some research has dealt with battery temperature variations without explicitly considering aging and vice versa. Padovani et al. [45] proposed an ECMS strategy with an additional soft constraint on the battery temperature. Ebbesen et al. [15] used a state of health model to find an optimal control law which minimizes the fuel consumption as well as the wear of the battery. Both aging effects have been considered by Sciarretta et al. [57] and Hu et al. [24].

References

1. Andre M (2004) Real-world driving cycles for measuring cars pollutant emissions—part A: the artemis european driving cycles. Rep Inrets-LTE 411:97
2. van Basshuysen R (2013) Ottomotor mit Direkteinspritzung: Verfahren, Systeme, Entwicklung, Potenzial. Atz/Mtz-Fachbuch, Springer Fachmedien Wiesbaden
3. Baumann BM, Washington G, Glenn BC, Rizzoni G (2000) Mechatronic design and control of hybrid electric vehicles. IEEE/ASME Trans Mechatron 5(1):58–72
4. Beck R, Richert F, Bollig A, Abel D, Saenger S, Neil K, Scholt T, Noreikat KE (2005) Model predictive control of a parallel hybrid vehicle drivetrain. In: 44th IEEE conference on decision and control, 2005 and 2005 European control conference. CDC-ECC'05. IEEE, pp 2670–2675
5. Boehme T, Metwally O, Becker B, Meinhardt N, Rucht M, Rabba H (2012) A simulation-based comparison of different power split configurations with respect to the system efficiency. In: SAE world congress, Technical paper 2012-01-0438. doi:10.4271/2012-01-0438
6. Boehme TJ, Becker B, Ruben-Weck M, Rothschuh M, Boldt A, Rollinger C, Butz R, Rabba H (2013) Optimal design strategies for different hybrid powertrain configurations assessed with European drive cycles. In: SAE world congress, Technical paper 2013-01-1751. doi:10.4271/2013-01-1751
7. Boehme TJ, Frank B, Schultalbers M, Schori M, Lampe B (2013) Solutions of hybrid energy-optimal control for model-based calibrations of HEV powertrains. In: SAE world congress, Technical paper 2013-01-1747. doi:10.4271/2013-01-1747
8. Boehme TJ, Schori M, Frank B, Schultalbers M, Lampe B (2013) Solution of a hybrid optimal control problem for parallel hybrid vehicles subjected to thermal constraints. In: 52nd IEEE conference on decision and control
9. Boehme TJ, Rothschuh M, Frank B, Schultalbers M, Schori M, Jeinsch T (2014) Multi-objective optimal design of parallel plug-in hybrid powertrain configurations with respect to fuel consumption and driving performance. SAE Int J Alt Power 3(2):176–192. doi:10.4271/2014-01-1158
10. Branicky MS (1995) Studies in hybrid systems: modeling, analysis, and control. PhD thesis, Massachusetts Institute of Technology
11. Chi S (2007) Position-sensorless control of permanent magnet synchronous machines over wide speed range. PhD thesis, Ohio state University
12. Colin G, Bloch G, Chamaillard Y, Ivanco A (2010) A real time neural energy management strategy for a hybrid pneumatic engine. IFAC symposium advances in automotive control. AAC, Munich, pp 75–80
13. Conlon B (2005) Comparative analysis of single and combined hybrid electrically variable transmission operating mode. In: SAE world congress, Technical paper 2005-01-1162. doi:10.4271/2005-01-1162
14. Du Z, Cheong KL, Li PY, Chase TR (2013) Fuel economy comparisons of series, parallel and HMT hydraulic hybrid architectures. In: Proceedings of the American control conference, Washington, DC, pp 5974–5979
15. Ebbesen S, Elbert P, Guzzella L (2012) Battery state-of-health perceptive energy management for hybrid electric vehicles. IEEE Trans Veh Technol 61(7):2893–2900
16. Ehsani M, Gao Y, Emadi A (2010) Modern electric, hybrid electric, and fuel cell vehicles. Fundamentals, theory, and design, 2nd edn. CRC Press
17. Gao DW, Mi C, Emadi A (2007) Modeling and simulation of electric and hybrid vehicles. Proc IEEE 95(4):729–745
18. Gelb G, Richardson N, Wang T, Berman B (1971) An electromechanical transmission for hybrid vehicle power trains—design and dynamometer testing. In: SAE world congress, Technical Paper 710235. doi:10.4271/710235
19. Gopal RV, Rousseau AP (2011) System analysis using multiple expert tools. In: SAE world congress, Technical Paper 2011-01-0754
20. Guzzella L, Sciarretta A (2005) Vehicle propulsion systems. Introduction to modeling and optimization. Springer, Berlin

21. Halbach S, Sharer P, Pagerit S, Rousseau AP, Folkerts C (2010) Model architecture, methods, and interfaces for efficient math-based design and simulation of automotive control systems. In: SAE world congress, Technical Paper 2010-01-0241
22. Hofmann P (2010) Hybridfahrzeuge. Springer, Wien, New York
23. Holmes A, Schmidt M (2002) Hybrid electric powertrain including a two-mode electrically variable transmission. US Patent 6,478,705
24. Hu X, Johannesson L, Murgovski N, Egardt B (2015) Longevity-conscious dimensioning and power management of the hybrid energy storage system in a fuel cell hybrid electric bus. Appl Energy 137:913–924
25. Hu Y, Yurkovich S, Guezennec Y, Bornatico R (2008) Model-based calibration for battery characterization in HEV applications. Proceedings of the 2008 American control conference, Seattle. IEEE, pp 318–325
26. Isermann R (2007) Mechatronic systems: fundamentals. Springer Science & Business Media
27. de Jager B, van Keulen T, Kessels J (2013) Optimal control of hybrid vehicles. Springer
28. Kaya CY, Noakes JL (2003) Computational method for time-optimal switching control. J Optim Theory Appl 117(1):69–92
29. Khajepour A, Fallah MS, Goodarzi A (2014) Electric and hybrid vehicles: technologies, modeling and control—a mechatronic approach. Wiley
30. Kiencke U, Nielsen L (2000) Automotive control systems. for engine, driveline, and vehicle. Springer, Berlin
31. Koprubasi K (2008) Modeling and control of a hybrid-electric vehicle for drivability and fuel economy improvements. PhD thesis, The Ohio State University
32. Kum D, Peng H, Bucknor N (2011) Supervisory control of parallel hybrid electric vehicles for fuel and emission reduction. ASME J Dyn Syst Meas Control 133(6):4498–4503
33. Lee CY, Zhao H, Ma T (2008) A low cost air hybrid concept. In: Les Rencontres Scientifiques de l'IFP—Advances in hybrid powertrains
34. Lee EA (2006) Cyber-physical systems—are computing foundations adequate? In: Workshop NSF on research motivation, techniques and roadmap, cyber-physical systems
35. Lee EA, Seshia SA (2011) Introduction to embedded systems—a cyber-physical systems approach. LeeSeshia.org
36. Lemaréchal C (1989) Nondifferentiable optimization. In: Handbooks in operations research and management science. Elsevier Science Publishers B.V., North-Holland
37. Liu J (2007) Modeling, configuration and control optimization of power-split hybrid vehicles. PhD thesis, The University of Michigan
38. Liu J, Peng H (2006) Control optimization for a power-split hybrid vehicle. In: Proceedings of the 2006 American control conference, Minneapolis, pp 466–471
39. Loxton R, Lin Q, Teo KL (2013) Minimizing control variation in nonlinear optimal control. Automatica 49(9):2652–2664
40. Markel T, Brooker A, Hendricks T, Johnson V, Kelly K, Kramer B, O?Keefe M, Sprik S, Wipke K, (2002) ADVISOR: a systems analysis tool for advanced vehicle modeling. J Power Sources 110(2):255–266
41. Matthé R, Eberle U (2014) The voltec system-energy storage and electric propulsion. Elsevier, Amsterdam, The Netherlands
42. Miller JM (2004) Propulsion systems for hybrid vehicles, vol 45. The Institution of Electrical Engineers
43. Miller JM, Everett M (2005) An assessment of ultracapacitors as the power cache in toyota THS-II, GM-Allision AHS-2 and ford FHS hybrid propulsion system. In: Applied power electronics conference and exposition, APEC 2005. Twentieth annual IEEE, pp 481–490
44. Morari M, Baotic M, Borrelli F (2003) Hybrid systems modeling and control. Eur J Control 9(2):177–189
45. Padovani TM, Debert M, Colin G, Chamaillard Y (2013) Optimal energy management strategy including battery health through thermal management for hybrid vehicles. In: Advances in automotive control (AAC 2013), pp 384–389

46. Passenberg B, Leibold M, Stursberg O, Buss PC (2011) The minimum principle for time-varying hybrid systems with state switching and jumps. In: Proceedings of the IEEE conference on decision and control, pp 6723–6729
47. Pichler F, Cifrain M (2014) Application-related battery modelling: from empirical to mechanistic approaches. In: Automotive battery technology, Springer, pp 53–69
48. Porsche-Spyder (2013) Hybrid-Supersportler mit 887 PS. http://www.auto-motor-und-sport.de
49. Rao V, Singhal G, Kumar A, Navet N (2005) Battery model for embedded systems. In: 18th international conference on VLSI design, 2005. IEEE, pp 105–110
50. Reza NJ (2009) Vehicle dynamics. Theory and application. Springer
51. Rizzoni G, Guzzella L, Baumann BM (1999) Unified modeling of hybrid electric vehicle drivetrains. IEEE Trans Mechatron 4(3):246–257
52. Roelle MJ, Shaver GM, Gerdes JC (2004) Tackling the transition: a multi-mode combustion model of SI and HCCI for mode transition control. In: ASME 2004 international mechanical engineering congress and exposition. American Society of Mechanical Engineers, pp 329–336
53. Sagastizábal C (1997) Nonsmooth optimization. In: Numerical optimization—theoretical and practical aspects, 2nd edn. Springer, Berlin
54. Schmidt MR (1996) Two-mode, input-split, parallel, hybrid transmission
55. Schmidt MR (1999) Two-mode, compound-split, electro-mechanical vehicular transmission
56. Schori M, Boehme T, Jeinsch T, Schultalbers M (2014) Optimal catalytic converter heating in hybrid vehicles. In: SAE World congress, Technical paper 2014-01-1351. doi:10.4271/2014-01-1351
57. Sciarretta A, di Domenico D, Pognant-Gros P, Zito G (2014) Optimal energy management of automotive battery systems including thermal dynamics and aging. In: Optimization and optimal control in automotive systems, Springer, pp 219–236
58. Silva C, Farias T, Frey H, Rouphail N (2006) Evaluation of numerical models for simulation of real-world hot-stabilized fuel consumption and emissions of gasoline light-duty vehicles. Transp Res Part D 1(5):377–385
59. Stewart DE (1992) A numerical algorithm for optimal control problems with switching costs. J Aust Math Soc Ser B Appl Math 34(02):212–228
60. Stockar S, Marano V, Canova M, Rizzoni G, Guzzella L (2011) Energy-optimal control of plug-in hybrid electric vehicles for real-world driving cycles. IEEE Trans Veh Control 60(7):2949–2962
61. Van Der Schaft AJ, Schumacher JM, van der Schaft AJ, van der Schaft AJ (2000) An introduction to hybrid dynamical systems, vol 251. Lecture notes in control and information science. Springer, London
62. Verbrugge M, Tate E (2004) Adaptive state of charge algorithm for nickel metal hydride batteries including hysteresis phenomena. J Power Sources 126(1):236–249
63. Verdonck N, Chasse A, Pognant-Gros P, Sciarretta A (2010) Automated model generation for hybrid vehicles optimization and control. Oil & Gas Science and Technology-Revue de lInstitut Français du Pétrole 65(1):115–132
64. Villeneuve A (2004) Dual mode electric infinitely variable transmission. In: SAE World congress, Technical paper 04CVT-19
65. VW-UP (2013) Volkswagen e-mobility. http://emobility.volkswagen.de
66. Waltermann P (1996) Modelling and control of the longitudinal and lateral dynamics of a series hybrid vehicle. In: Proceedings of the 1996 IEEE international conference on control applications. IEEE, pp 191–198
67. Wang J, Liu P, Hicks-Garner J, Sherman E, Soukiazian S, Verbrugge M, Tataria H, Musser J, Finamore P (2011) Cycle-life model for graphite-LiFePO 4 cells. J Power Sources 196(8):3942–3948
68. Ward J, Moawad A, Kim N, Rousseau A (2012) Light-duty vehicle fuel consumption, cost and market penetration potential by 2020. EVS26, Los Angeles, CA
69. Wei X (2004) Modeling and control of a hybrid electric drive train for optimum fuel economy, performance and drivability. PhD thesis, Ohio State University

70. Weng C, Sun J, Peng H (2014) A unified open-circuit-voltage model of lithium-ion batteries for state-of-charge estimation and state-of-health monitoring. J Power Sources 258:228–237
71. Zhang B, Mi CC, Zhang M (2011) Charge-depleting control strategies and fuel optimization of blended-mode plug-in hybrid electric vehicles. IEEE Trans Veh Technol 60. doi:10.1109/TVT.2011.2122313
72. Zuurendonk B (2005) Advanced fuel consumption and emission modeling using Willans line scaling techniques for engines. Technische Universiteit Eindhoven, Department Mechanical Engineering Dynamics and Control Technology Group, Technical report

Part V
Applications

Chapter 11
Advanced Vehicle Calibration

11.1 Introduction

Engineers aiming at efficient functional design and calibration of energy managements can benefit greatly from the numerical methods and theory of optimal control in several ways. Being able to solve an *optimal control problem* (OCP) for a specific drive cycle, the engineer can get a feeling of what the optimal solution must look like and certain parameters can be adapted, such that an existing calibration comes closer to the solution of the OCP. This procedure can be repeated for several drive cycles to avoid results based on exceptional cases in which a calibration performs well on one specific drive cycle only.

In this chapter, the problem of finding appropriate parameters and lookup tables for rule-based energy management can be done by formulating open-loop energy management as equivalent problem to the original problem. Open-loop energy management $\mathcal{K}$ is shown in Fig. 11.1.

Solving $\mathcal{K}$ for an arbitrary test cycle can retrieve calibration parameters, which dramatically facilitates the calibration process and improves the calibration quality.

11.2 Offline Solution of Switched Optimal Control Problems for Known Driving Profiles

In this section, we formulate OCPs for the optimal operation of a parallel hybrid vehicle over test cycles and evaluate the advantages and disadvantages of numerical methods described in the preceding chapters.

A Problem Without State Jumps

For the first three problems formulated, the quasi-steady model $\mathcal{M}_1$ from Chap. 10 that includes two continuous-valued states (namely, fuel consumption $\beta(\cdot)$ and state of charge $\xi(\cdot)$) is used. The control problem aims at finding the optimal torque split

T.J. Böhme and B. Frank, *Hybrid Systems, Optimal Control and Hybrid Vehicles*,
Advances in Industrial Control, DOI 10.1007/978-3-319-51317-1_11

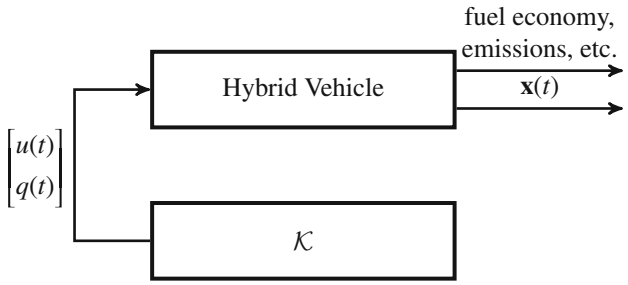


Fig. 11.1 Energy management as open-loop control strategy

between *internal combustion engine* (ICE) and *motor/generator* (MG) characterized by the ICE-torque $T_{ice}(\cdot)$ that is defined as continuous-valued control $u(\cdot)$, as well as the optimal engine on/off decision $\zeta(\cdot)$. In the first problem, the costs for an engine start that could be modeled using the state jump function $\boldsymbol{\delta}_{(\zeta(t_j^-),\zeta(t_j^+))}(\cdot)$ are not considered. The formulation for this problem can then be summarized as Mayer problem as follows:

$$\mathcal{P}_1 := \begin{cases} \min\limits_{\zeta(t)\in\{0,1\},\ u(t)\in\hat{\mathcal{U}}(q(t),t)} \beta(t_f) \\ \text{subject to } \mathcal{M}_1 \ (10.126) \\ \xi(t_0) = 0.5 \\ \xi(t_f) = 0.5 \\ \beta(t_0) = 0 \\ \boldsymbol{\delta}_{(\zeta(t_j^-),\zeta(t_j^+))}(\mathbf{x}(t_j^-)) = \mathbf{0},\ \forall \zeta(t_j^-), \zeta(t_j^+) \in \{0,1\}. \end{cases} \tag{11.1}$$

A final boundary condition is herein imposed that demands the high-voltage battery to have the same energy level ($\xi(t_f) = 0.5$) as at the beginning of the test cycle. This is commonly demanded in many countries' legislative homologation requirements.

Evaluating the first-order necessary conditions is helpful as it can provide useful insight into the problem structure and hence facilitate the choice of an appropriate algorithm for the solution of the *switched optimal control problem* (SOCP). Especially, the first-order necessary conditions yield some interesting results that can be used to simplify the problem.

Recall that the state vector of the model $\mathcal{M}_1$ is defined as

$$\mathbf{x}(t) := \begin{bmatrix} \beta(t) \\ \xi(t) \end{bmatrix}$$

then the Hamiltonian of the Mayer problem is given as

$$\begin{aligned} \mathcal{H}(\mathbf{x}(t), q(t), \boldsymbol{\lambda}(t), u(t)) &= \boldsymbol{\lambda}^T(t) \cdot \mathbf{f}_{q(t)}(\mathbf{x}(t), u(t)) \\ &= \lambda_1(t) \cdot \dot{\beta}(t) + \lambda_2(t) \cdot \dot{\xi}(t). \end{aligned}$$

Evaluating the transversality condition (4.124), the time derivative of the costates results in

$$\dot{\lambda}_1(t) = -\frac{\partial \mathcal{H}}{\partial x_1}(\mathbf{x}(t), q(t), \boldsymbol{\lambda}(t), u(t)) = 0 \tag{11.2}$$
$$\dot{\lambda}_2(t) = -\frac{\partial \mathcal{H}}{\partial x_2}(\mathbf{x}(t), q(t), \boldsymbol{\lambda}(t), u(t)) = -\frac{1}{Q_{bat}} \cdot \frac{\partial I_{bat,q(t)}}{\partial \xi}(\xi(t), u(t)).$$

Hence, the first costate (11.2) remains constant. The total variation over a given interval $[t_0, t_f]$ of the second costate will also be small, since the last term in the second costate's time derivative yields

$$\frac{\partial I_{bat,q(t)}}{\partial \xi}(\xi(t), u(t)) = \frac{\partial I_{bat,q(t)}}{\partial V_{oc}}(\xi(t), u(t)) \cdot \frac{\partial V_{oc}}{\partial \xi}(\xi(t)). \tag{11.3}$$

For modern batteries, the last term in this equation takes on very small values, caused by the minor dependence of the open-circuit voltage of the state of charge. Due to the fact that for this problem $\boldsymbol{\delta}_{(\zeta(t_j^-),\zeta(t_j^+))}(\mathbf{x}(t_j^-)) = \mathbf{0}$ is assumed, the Hamiltonian as well as the costates are continuous on a switching (cf. (4.125) and (4.126)):

$$\boldsymbol{\lambda}(t_j^+) = \boldsymbol{\lambda}(t_j^-)$$
$$\mathcal{H}(\mathbf{x}(t_j^+), q(t_j^+), \boldsymbol{\lambda}(t_j^+), u(t_j^+)) = \mathcal{H}(\mathbf{x}(t_j^-), q(t_j^-), \boldsymbol{\lambda}(t_j^-), u(t_j^-)).$$

According to the transversality condition (4.124), the final value of the first costate can then be calculated as

$$\lambda_1(t_f) = \frac{\partial \beta(t_f)}{\partial \beta(t_f)} = 1.$$

Thus, the first costate is constant in the Hamiltonian since $\lambda_1(t_0) = \lambda_1(t_f) = 1$ applies and the second costate can be remapped to

$$\lambda(t) \equiv \lambda_2(t)$$

for simplicity. Then, the Hamiltonian can be rewritten as

$$\mathcal{H}(\mathbf{x}(t), q(t), \lambda(t), u(t)) = \dot{\beta}(t) + \lambda(t) \cdot \dot{\xi}(t). \tag{11.4}$$

According to the Hamiltonian minimum condition (4.123), the Hamiltonian needs to be minimized by the continuous-valued control $u(\cdot)$ and the discrete state $q(\cdot)$ for almost every time $t \in [t_0, t_f]$.

A solution to this problem can be efficiently obtained using the indirect single shooting method (cf. Sect. 7.3) and the embedding method with either direct shooting or direct collocation (cf. Sect. 8.3.1). Both outperform *dynamic programming* (DP)

Table 11.1 Comparison of cost function value and computation time for $\mathcal{P}_1$

	$\phi(\mathbf{x}(t_f))$ (ml)	t_{comp}(s)
Indirect shooting	496.9	29.39
Embedding (direct shooting)	497.1	10.28
Dynamic programming	496.9	4386.5

by far as they perform Newton-type optimizations, whereas dynamic programming searches a discrete control and state space.

To benchmark the solution trajectories obtained from the indirect single shooting method and the embedding method with direct shooting, the solution trajectories for the *motor vehicle emission group* (MVEG) test cycle are compared to a DP solution. The computation times can be seen in Table 11.1 and the trajectories are depicted in Fig. 11.2. The DP requires an optimal control problem to be formulated as Bolza-type without final state values. The final state values $\mathbf{x}_{[\mathcal{I}_f]}(t_f) = \mathbf{x}_f$ can then be included as a soft constraint in the endpoint function term.

The indirect single shooting method and the embedding direct shooting method converge to the same solution and the difference from the DP solution is entirely negligible. For the embedding direct shooting method, the relaxed discrete controls are binary admissible, such that the solution trajectory of $\mathcal{P}_1$ is binary feasible and no further rounding scheme has to be applied (see Sect. 8.3.1). The computation time of the embedding direct shooting method strongly depends on a number of factors. For instant, gradients calculated using the compression methods from Chap. 9 reduce the computation time by more than 85%. Additionally, by transcribing the problem as an embedded direct collocation problem instead of an embedded direct shooting problem it is highly recommended to employ a SQP solver with sparse matrix algebra to exploit the sparse problem structure. This reduces the computation time significantly.

For longer test cycles or small step-lengths h, the indirect single shooting method can be an efficient option if the initial costate $\hat{\boldsymbol{\lambda}}$ can be easily guessed and the total variation of (11.3) remains small. If this simplification cannot be guaranteed, then the indirect multiple shooting method must be applied. In case of state constraints because of a limited battery capacity, the transcription of the problem $\mathcal{P}_1$ as an embedding direct collocation method is highly recommended.

Constant Binary Controls

The binary controls—σ_1 for hybrid drive mode and σ_2 for electric drive mode—are for some arcs a priori known. For instant, the engine must not be switched on if the minimal engine speed is not reached. For these time intervals $\mathcal{K}$, the binary controls must be set to $\overline{\boldsymbol{\sigma}}_1^{[\mathcal{K}]} = \mathbf{0}$ and $\overline{\boldsymbol{\sigma}}_2^{[\mathcal{K}]} = \mathbf{1}$ and remain constant for the entire optimization. Therefore, these intervals can be ignored for the optimization, which reduces the optimization dimensionality for all methods.

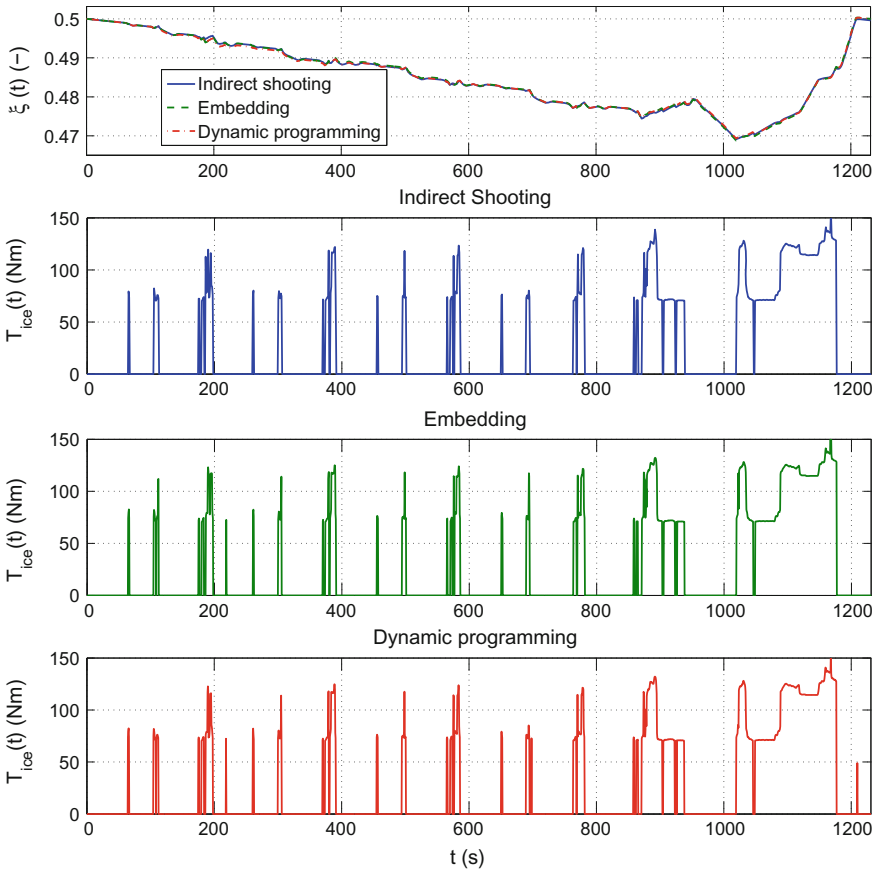


Fig. 11.2 Solution trajectories of problem $\mathcal{P}_1$ for the MVEG test cycle obtained with the indirect shooting method, the embedding method, and the dynamic programming method

Binary Feasible Controls

For some cases, binary feasible controls cannot be obtained. Then, problem $\mathcal{P}_1$ has to be modified with a penalty term, which is usually added to the cost function. This yields

$$\hat{\beta}(t_f) := \beta(t_f) + \sum_{q=1}^{2} \alpha \int_{t_0}^{t_f} \hat{\sigma}_q(t) \cdot \left(1 - \hat{\sigma}_q(t)\right) \, \mathrm{d}t,$$

where α is increased in a homotopy approach.

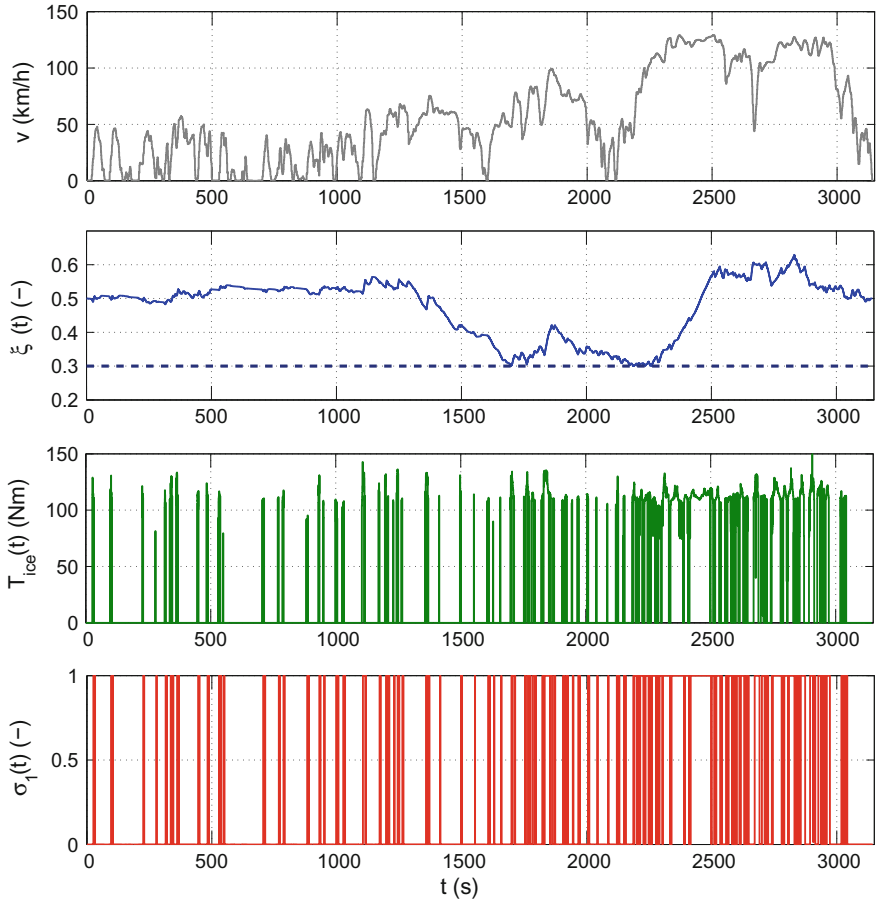


Fig. 11.3 Solution trajectories of problem $\mathcal{P}_1$ for the ARTEMIS test cycle obtained with the embedding direct collocation method. σ_1 is the discrete control for the hybrid drive mode

Constrained State of Charge

Figure 11.3 shows the solution trajectories of the ARTEMIS test cycle using the embedding direct collocation approach. The dark thick blue line indicates a constrained state of charge due to a limited battery. The battery charge is increased in the first half of the test cycle to avoid long boundary arcs, since sliding on the lower limit would prevent to switch to the electric drive mode. For real-world drive cycles it is impossible to predict the boundary arcs a priori, such that indirect shooting can cope with that. Such problems appear also in the design phase of hybrid vehicles where the proper battery capacities are only roughly known and therefore sometimes are too small designed. Thus, embedded direct collocation is recommended for such problems as it can naturally handle state constraints without guessing the constrained/unconstrained arcs.

Frequent Switchings

Note that frequent engine on/off commands occur in both solutions. In Fig. 11.3 this effect can be readily observed and is not a problem of the embedding direct collocation approach. Instead, this effect can always be realized for highly dynamic test cycles like ARTEMIS and WLTP if no cost is associated to an engine start. Therefore, these solutions are optimal with respect to the formulation of $\mathcal{P}_1$ which may appear rather unrealistic. Nevertheless, the results obtained will be helpful for the calibration of rule-based energy managements, which will be covered later in this chapter. Certainly, additional switching costs will help to overcome this problem, but that will introduce a lot more complexity in the optimization approach. We come back to this later.

More Discrete Decisions: Gear Selection

The second optimal control problem considers additional discrete decisions, namely the gear selection $\kappa(\cdot)$ of the automatic transmission. This increases the number of possible discrete decisions in the model up to $N_q = 12$ to cover all possible combinations of drive modes including $\zeta(t) \in \{0, 1\}$ and gears $\kappa(t) \in \{1, \ldots, 6\}$. The problem can then be formulated as follows:

$$\mathcal{P}_2 := \begin{cases} \min\limits_{\zeta(t)\in\{0,1\},\ \kappa(t)\in\{1,\ldots,6\},\ u(t)\in\hat{\mathcal{U}}(q(t),t)} \beta(t_f) \\ \text{subject to } \mathcal{M}_1 \text{ (10.126)} \\ \xi(t_0) = 0.5 \\ \xi(t_f) = 0.5 \\ \beta(t_0) = 0 \\ \boldsymbol{\delta}_{(\zeta(t_j^-),\xi(t_j^+))} = \mathbf{0},\ \forall \zeta(t_j^-), \zeta(t_j^+) \in \{0, 1\}. \end{cases} \tag{11.5}$$

The indirect single shooting method can be preferable when a high number of discrete decisions have to be managed because of less coding. The solution trajectories obtained within 150 s are depicted in Fig. 11.4.

Aspects like driving comfort were not considered in model $\mathcal{M}_1$. Therefore, the highest gear $\kappa = 6$ that is possible in a given driving situation is often selected because it has the lowest drag loss and the possibility to operate the ICE as well as the MG at higher loads. To get more realistic results in terms of drivability, the model needs to be enhanced and either the cost function or the constraints of the problem formulation will have to be altered.

Including State Jumps

In the following, we will consider that a start of the ICE requires some additional amount of fuel and some electrical energy and therefore leads to jumps in the

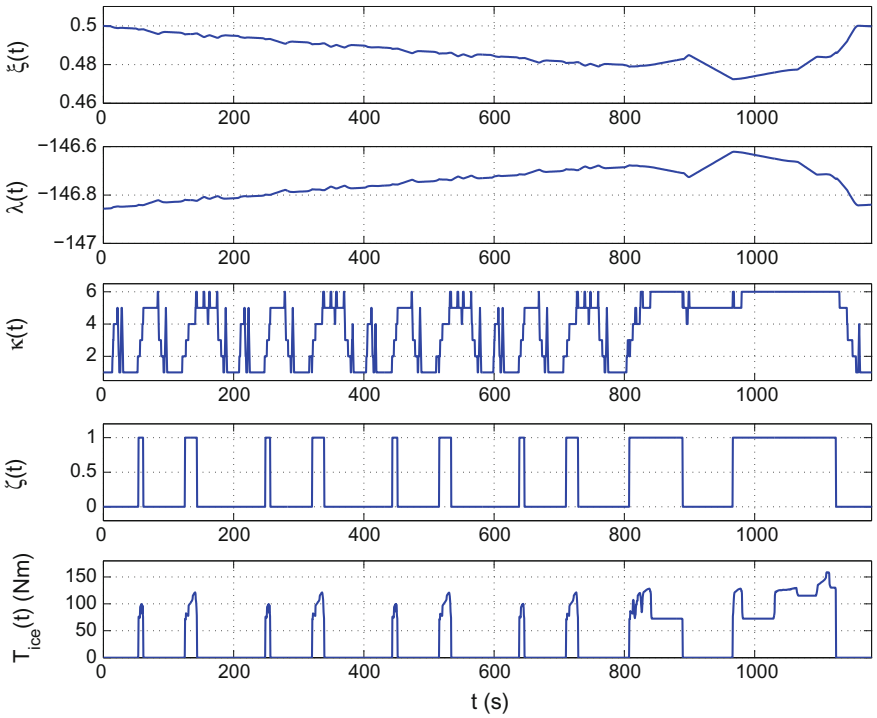


Fig. 11.4 Solution trajectories of problem $\mathcal{P}_2$ obtained with the indirect shooting method

respective state trajectories. Consequently, to account for the losses during the engine start, a jump function is defined as follows

$$\boldsymbol{\delta}_{(\zeta(t_j^-),\zeta(t_j^+))}(\mathbf{x}(t_j^-)) = \begin{cases} [\Delta\beta,\ \Delta\xi]^T, & \zeta(t_j^-) = 0 \wedge \zeta(t_j^+) = 1 \\ \mathbf{0}, & \text{otherwise}, \end{cases} \tag{11.6}$$

where $\Delta\beta$ and $\Delta\xi$ are the amounts of fuel and the percentage of battery capacity consumed for an engine start, respectively. The problem is then formulated by:

$$\mathcal{P}_3 := \begin{cases} \min\limits_{\zeta(t)\in\{0,1\},\ u(t)\in\hat{\mathcal{U}}(q(t),t)} \beta(t_f) \\ \text{subject to } \mathcal{M}_1 \text{ (10.126)} \\ \xi(t_0) = 0.5 \\ \xi(t_f) = 0.5 \\ \beta(t_0) = 0 \\ \text{and (11.6)}. \end{cases} \tag{11.7}$$

Table 11.2 Cost function value and computation time for $\mathcal{P}_3$

	$\phi(\mathbf{x}(t_f))$ (ml)	t_{comp} (s)
Dynamic programming	505.75	9922

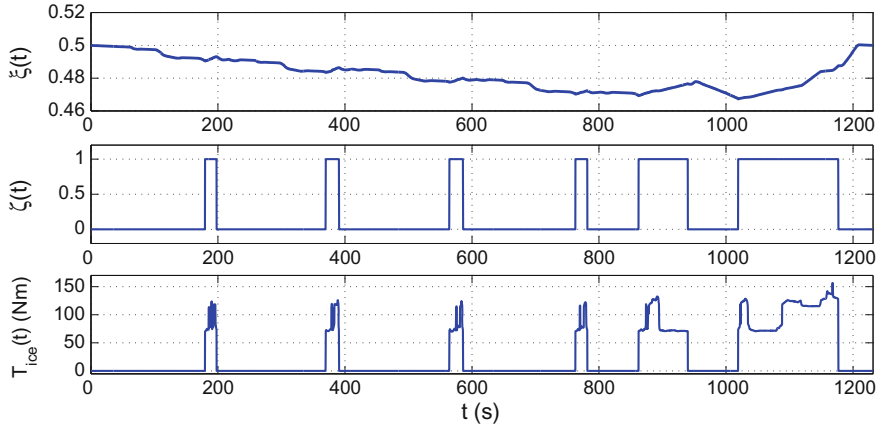


Fig. 11.5 Solutions for $\mathcal{P}_3$ obtained with dynamic programming

DP is an adequate solution method for $\mathcal{P}_3$, as the dimension of the continuous-valued states can be reduced to 1. The Lagrangian term of the Bolza formulation accounts for the fuel consumption and therefore the state $\beta(\cdot)$ is not explicitly needed and hence does not need to be gridded. Consequently, an optimum with respect to a chosen discretization of the state $\xi(\cdot)$ can be found by DP in a feasible amount of time.

The continuous but nonsmooth jump function can be implemented as

$$\boldsymbol{\delta}(\zeta(t_j^-), \zeta(t_j^+)) = \max\left(0, \left[\zeta(t_j^+) - \zeta(t_j^-)\right]\right) \cdot \begin{bmatrix} \Delta\beta \\ \Delta\xi \end{bmatrix}.$$

The height of the jump at an engine start was assumed to be $[\Delta\beta,\ \Delta\xi]^T = [0.65\,(\text{ml}),\ -0.00013]^T$. The results of DP are shown in Table 11.2 and Fig. 11.5.

Remark 11.1 The height of the state jump $\boldsymbol{\delta}_{(\zeta(t_j^-),\zeta(t_j^+))}(\cdot)$ is assumed to be small compared with the variation of the state $\mathbf{x}(\cdot)$ between two switchings.

Since the hybrid execution sequence is not known in advance, the embedding method and indirect shooting method are only valid approaches if the state trajectories are assumed to be continuous.

Optimal Catalytic Converter Heating

The following problem uses the much more complex model $\mathcal{M}_2$ that uses the ignition angle $\chi(\cdot)$ and the cylinder charge $m_{cyl}(\cdot)$ as continuous-valued controls and models

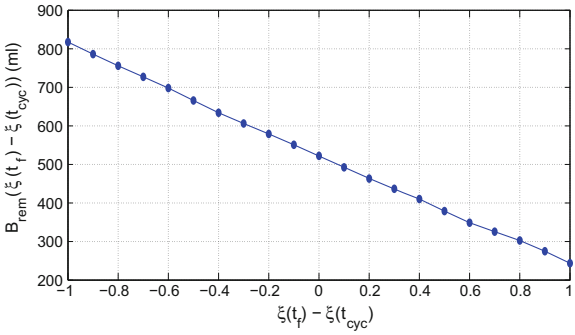


Fig. 11.6 Function $B_{rem}(\xi(t_f) - \xi(t_{cyc}))$ that provides the optimal fuel consumption for the remaining drive cycle $t \in (t_f, t_{cyc}]$ depending on the difference $\xi(t_f) - \xi(t_{cyc})$

different combinations of clutch states and injection schemes as discrete state values as introduced in Sect. 10.5.2. The problem is formulated over the first 175 s of the FTP-75 cycle, where the aim is to find a strategy for optimally heating up the ICE and the exhaust system. In this case, the final state conditions for the SOCP are formulated differently. It is imposed that the ICE is heated up to at least 353 K at final time t_f. The entire driving cycle is not regarded but $t_f = 175$ s is chosen such that $0 < t_f < t_{cyc}$. It is therefore not necessary to demand a fixed value for $\xi(t_f)$ as $\xi(\cdot)$ needs to be balanced at least at t_{cyc} but not for $t \in [0, t_f]$. However, if the state of charge $\xi(t_f)$ has a low value, more charging has to be performed over the remaining drive cycle with $t \in (t_f,\ t_{cyc}]$. The function $B_{rem}(\xi(t_f) - \xi(t_{cyc}))$ gives the fuel consumption for the remaining drive cycle, depending on the difference between $\xi(t_f)$ and the target value $\xi(t_{cyc})$. The function is shown in Fig. 11.6. Since it is demanded that the ICE is heated up sufficiently at t_f, the function can be easily set up by solving a SOCP for the time range $(t_f, t_{cyc}]$ with the simpler model $\mathcal{M}_1$. This is done on a grid of $\xi(t_f)$. Between the grid values, a cubic spline interpolation is performed. The cost function is then defined as

$$B(\mathbf{x}(t_f)) = \beta(t_f) + B_{rem}(\xi(t_f))$$

and the problem formulation becomes

$$\mathcal{P}_4 := \begin{cases} \min\limits_{q(t)\in\{1,\dots,5\},\ u(t)\in\hat{\mathcal{U}}(q(t),t)} B(\mathbf{x}(t_f)) \\ \text{subject to } \mathcal{M}_2 \text{ (10.128)} \\ \beta(t_0) = 0 \\ \xi(t_0) = 0.345 \\ \vartheta_{cw}(t_0) = \vartheta_{cyl}(t_0) = \vartheta_{twc}(t_0) = 293\,(\mathrm{K}) \\ \vartheta_{cw}(t_f) = 353\,(\mathrm{K}) \\ Z(t_f) = Z^{max}. \end{cases} \tag{11.8}$$

Only one artificial emission component is regarded, which is resembled by the state $Z(\cdot)$ and an upper bound on $Z(t_f)$ is imposed. State jumps are not considered.

Problem $\mathcal{P}_4$ is much more complex than problems $\mathcal{P}_1$–$\mathcal{P}_3$ due to the high dimension of the continuous states as well as the high number of discrete state values. Consequently, dynamic programming is prohibitive due to the high dimensionality of the system. Indirect shooting is unlikely to yield a solution because of its low convergence region. It is also a difficult task to guess initial costates that will converge to a solution satisfying the final boundary conditions. Applying embedding to the problem will lead to a large optimization vector due to the high number of discrete state values. An alternative is the two-stage method described in Sect. 8.3.2, which works well for this type of problem (see also Schori et al. [23]).

The solution gives valuable information for defining the catalytic converter heating phase. To evaluate the effect of different bounds on the artificial emission component $Z(\cdot)$, optimizations were performed with three different upper bounds $Z_1^{max} = Z_{ref}$, $Z_2^{max} = 1.1 \cdot Z_{ref}$, and $Z_3^{max} = 1.2 \cdot Z_{ref}$ to see, how a less stringent constraint on emission affects the continuous-valued controls as well as the switching mode sequence. The results can be seen in Fig. 11.8. Note that a lower upper bound leads to only slightly decreased time span used for the *three-way catalytic converter* (TWC) heating but to a significant retardation of the ignition angle, which causes a faster TWC-heating. TWC-heating is mostly performed during idling with the clutch K0 closed and the MG connected ($q(t) = 1$). As soon as a TWC-temperature with a good conversion efficiency is attained, the ignition angle tends to the lower bound to increase combustion efficiency. The base ignition angle was set as the lower limit. The late ignition angle also leads to lower values of $\xi(\cdot)$ at the end of the catalytic heating process, since a lower output torque is provided by the ICE. During TWC-heating, the exhaust mass flow is constrained, as high mass flow decreases the catalytic converter efficiency. In the solution trajectory, the relative cylinder charge tends to the respective bound.

The first subplot of Fig. 11.7 shows the convergence of the proposed algorithm. The convergence is rather slow but monotonically decreasing. It has shown to be very robust against initial guesses of $q(\cdot)$ that differ considerably from the solution found. The third subplot $\#\mathcal{H}_{min}$ illustrates, how far the optimality condition (4.123) is satisfied for the current iteration. A value of 0.1 means that at 10% of the points on the time grid, the value of the Hamiltonian function is not minimal.

The recovered costates are also depicted in Fig. 11.8. The costates offer in this case a physical interpretation. Whenever the costate for a respective state is low, increasing the corresponding state at this point is more beneficial than at other times. For instant, increasing the TWC-temperature $\vartheta_{twc}(\cdot)$ early in the drive cycle is recommended to quickly achieve good conversion efficiencies. Once the light-off temperature is reached, a further increase of the temperature has hardly any effect. The costate $\lambda_{\vartheta_{twc}}(\cdot)$ tends to zero, as soon as the light-off temperature is exceeded.

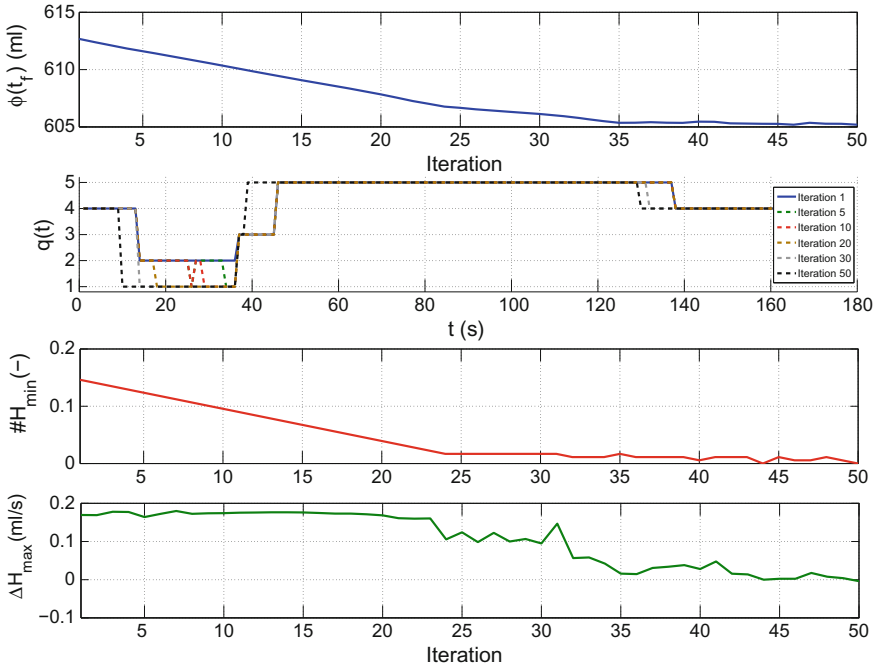


Fig. 11.7 Convergence of the two-stage algorithm for $\mathcal{P}_4$. $\#\mathcal{H}_{min}$ is a measure for optimality. $\Delta\mathcal{H}_{max}$ is the largest possible decrease in the Hamiltonian function for one time instant

11.3 Analytical Calibration for Rule-Based Energy Managements

The solution of optimal control problems is only a first step in the definition of energy management for *hybrid electric vehicles* (HEV). The results obtained from the numerical solution of OCPs only serve as a visual orientation, as the control and state trajectories are cycle-specific and do not give any further information for other unknown drive cycles or for different initial or final states. At the same time, the solution of SOCPs online during the vehicle operation is usually prevented by the very limited computational performance available in today's *electronic control units* (ECU). Especially direct methods are difficult to implement due to the high memory requirements needed for storing the Jacobian and (approximated) Hessian matrices. Most energy management systems for hybrid vehicles still rely on rule-based control strategies because of their ease of implementation and the low computational demand. Energy managements that only use the current state of the vehicle, consisting of all values that can be measured or estimated at a certain time t to determine the set values for the powertrain, are called *causal* (Guzzella and Sciarretta [6]). If estimated future information is also used to define the current set values, then energy management is called *predictive*.

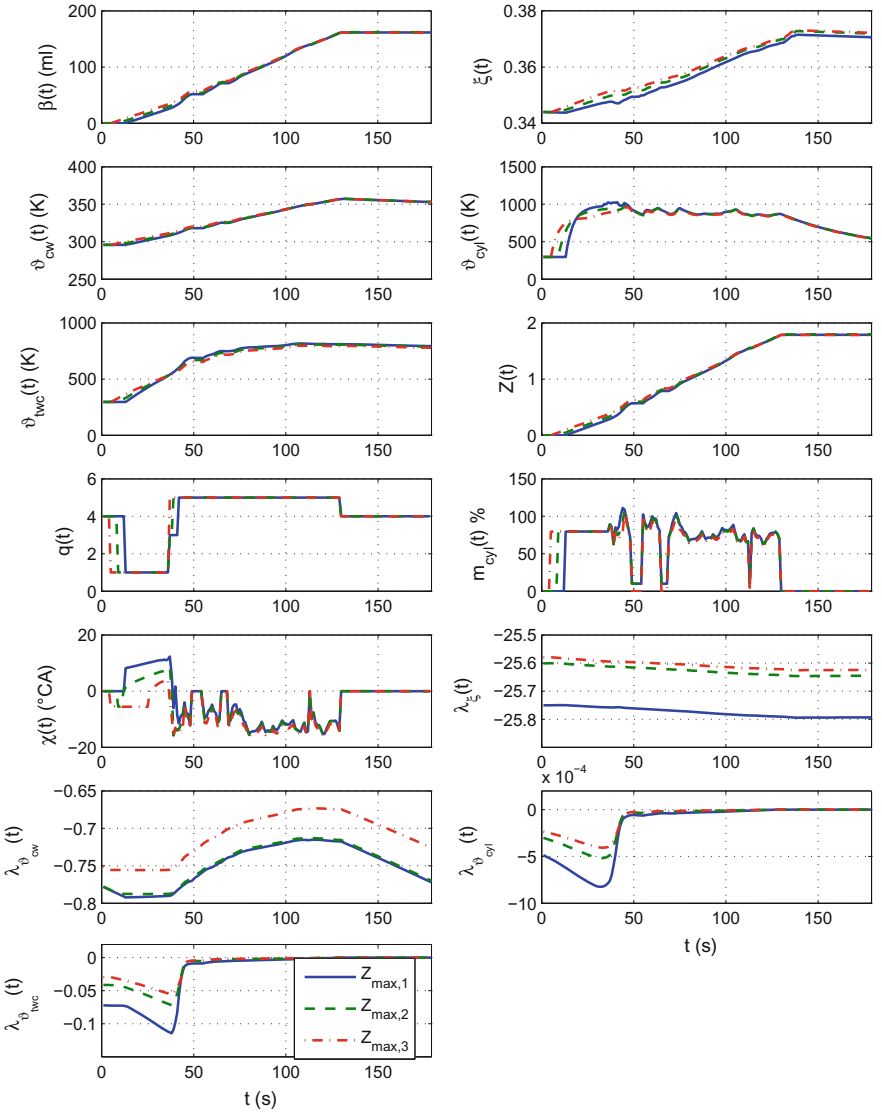


Fig. 11.8 Solution trajectories and recovered costates obtained for $\mathcal{P}_4$ with different bounds for the artificial emission state $Z(\cdot)$. The corresponding continuous state of each depicted costate is written as subscript

Yet, by exploiting optimal control theory, the results can be generalized and used for the determination of energy management parameters. Over the last decade, research in this area has focused more and more on analytical methods. The most attention has been paid to the *Pontryagin's minimum principal* (PMP) and the related *equivalent consumption minimization strategy* (ECMS) strategies. The use of these

approaches is appealing due to the fact that the high-dimensional optimization problem is reduced to solve just a nonlinear equation and hence to finding a feasible initial costate value. The problem is further simplified by assuming the costate to be constant. An evaluation of the effect of this assumption will be given later in this section. If the costate is known, the optimal continuous-valued controls that lead to a desired final state of charge for a specific drive cycle are entirely defined, if only continuous-valued controls are regarded. The problem then remains to define an initial costate value without knowledge of the drive cycle. Kim et al. [13] suggested to select the costate based on heuristically defined values that represent the drive pattern. A learning procedure that corrects the costate value, when a lower or upper bound for the state of charge is hit, is described in the work of Chen and Salman [2]. A study on the relationship between different road-type events and the costate value that leads to a charge-sustaining operation of the HEV was performed by Gong et al. [5].

11.3.1 Constant Costate Assumption

With a known costate, the optimal continuous-valued controls can be found by minimization of the Hamiltonian function at each time instant via a numerical procedure. Despite the fact that, for most powertrain architectures, the minimization problem will involve only a single variable, the numerical procedure is not ready to implement on today's ECUs. In this section, we show how *lookup tables* (LUT) storing the optimal continuous-valued controls as well as optimal discrete controls can be automatically generated. In Fig. 11.9, a sketch of an LUT-based energy management is depicted. The set point values $\hat{\kappa}(\cdot)$, $\hat{\zeta}(\cdot)$, and $\hat{T}_{mg}(\cdot)$, supplied to the lower level

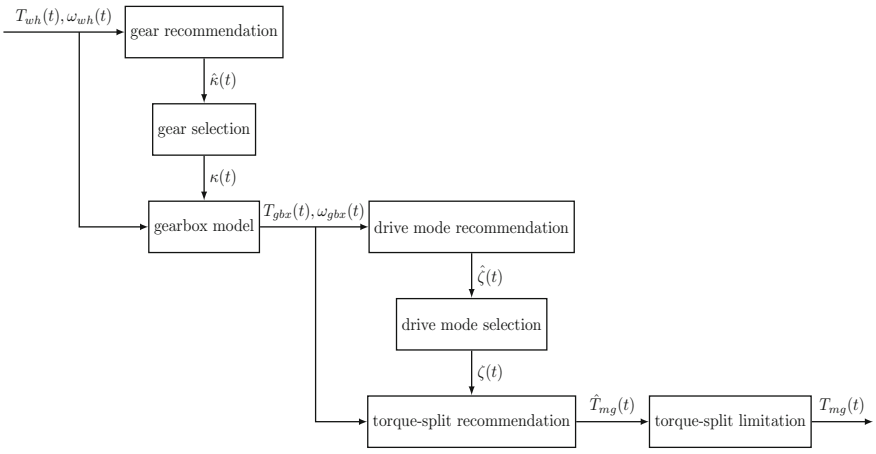


Fig. 11.9 Sketch of rule-based energy management for hybrid electric vehicles

controller structures are herein determined by LUTs. Based on the current driving condition that is represented by the torque demand at the wheel $T_{wh}(\cdot)$ and the current wheel speed $n_{wh}(\cdot)$, a gear recommendation denoted by $\hat{\kappa}(\cdot)$ is made based on a LUT. Whether this gear is actually used will depend on the gearbox control strategy and thus on many more conditions that are not covered in this book. With a simple gearbox model, the gearbox input torque $T_{gbx}(\cdot)$ and gearbox input speed $\omega_{gbx}(\cdot)$ can then be calculated. Based on these values, a recommendation for the drive mode $\hat{\zeta}(\cdot)$ is again obtained from a LUT. If hybrid drive mode is selected, the torque split between MG and ICE finally needs to be determined. The respective MG-torque $\hat{T}_{mg}(\cdot)$ is stored in a LUT.

Solving a SOCP with an indirect shooting method, a cycle-specific costate trajectory is readily obtained. This trajectory can also be recovered from the embedding solution, as shown in Sect. 8.2.4. It can be seen from the solutions that the costate remains nearly constant over the time. This is not surprising. Looking at Equation (11.3), the derivative $\mathrm{d}V_{oc}/\mathrm{d}\xi$ is usually small for modern battery types, since the dependence of the open-circuit voltage is rather small. Thus, assuming

$$\frac{\mathrm{d}V_{oc}}{\mathrm{d}\xi} \approx 0,$$

the costate remains constant over the entire time interval $t \in [t_0, t_f]$. To further investigate the effect, when the variation of the state of charge is much larger, calculations were performed for a charge-depleting scenario, where the initial state of charge was set to 0.8 and the final state of charge to 0.3 (see Fig. 11.10). These calculations were

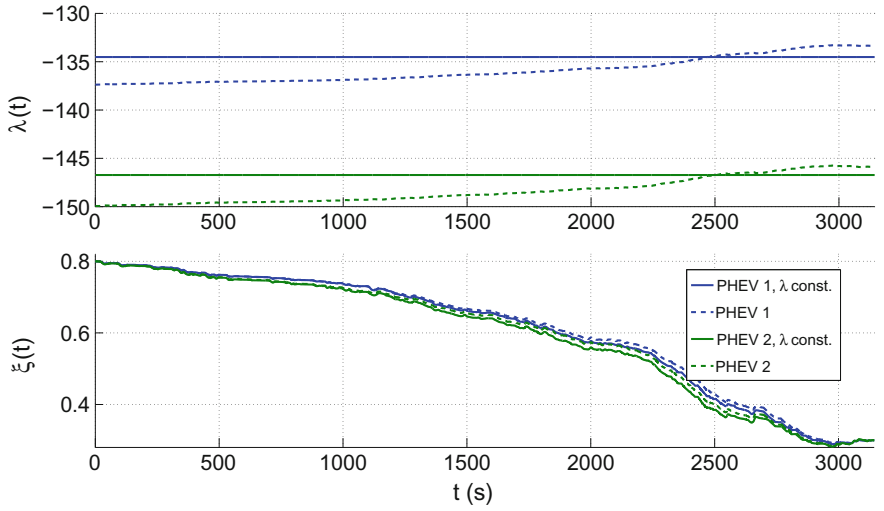


Fig. 11.10 Comparison of calculated trajectories for $\lambda(\cdot)$ and $\xi(\cdot)$ with and without constant costate assumption

Table 11.3 Effect of the constant costate assumption on the fuel consumption

	Fuel consumption (l) for varying $\lambda(\cdot)$	Fuel consumption (l) for constant $\lambda(\cdot)$
PHEV1	0.8691	0.8703
PHEV2	1.2718	1.2761

repeated for two vehicle configurations of different mass categories and for two different drive cycles. The resulting fuel consumptions can be seen in Table 11.3. One can observe that the solutions with constant costate are acceptable.

11.3.2 Influence of Switching Costs

As has been pointed out in Sect. 10.8, switching costs (including both fuel and electrical energy) can be modeled as state jumps. Compared with the constant costate assumption, the influence of whether state jumps are regarded or disregarded is more perceptible. Higher switching cost can lead to less ICE-starts. To compensate for this, the engine is usually switched on for longer time intervals and the torque is chosen slightly higher. A direct comparison of a solution obtained for a problem considering state jumps of different heights is shown in Fig. 11.11 for the highly dynamical FTP-72 test cycle.

Problem $\mathcal{P}_3$ can only naturally be solved using DP, which is in turn slow and limited to the problem size.

An alternative is the reformulation of the state jumps in form of switching costs (cf. Sect. 3.3.7.4). If this is possible and if the switching cost term can be formulated in terms of switching arc-lengths, then the original state jumps can be substituted with differentiable switching costs, such that nonlinear programming methods like SQP can cope with that (cf. Sect. 10.8). The idea here is to enable the *switching time optimization* (STO). Thus, we solve problem $\mathcal{P}_3$ without state jumps at first using the embedding method and use the hybrid execution sequence to transform the problem formulation $\mathcal{P}_3$ to a STO (cf. Sect. 8.3.3) as

$$\mathcal{P}_5 := \begin{cases} \min\limits_{\tilde{\zeta}_j(\tau)\in\{0,1\},\ \tilde{u}_j(\tau)\in\tilde{\mathcal{U}}(\tilde{q}(\tau),\tau)} & \tilde{\beta}_{N_{t_3}-1}(1) + \tilde{\phi} \\ \text{subject to } \mathcal{M}_1 \ (10.126) & \\ \tilde{\xi}_0(0) = 0.5 & \\ \tilde{\xi}_{N_{t_3}-1}(1) = 0.5 & \\ \tilde{\beta}_0(0) = 0 & \end{cases}$$

where $\tilde{\phi}$ is given by (10.135).

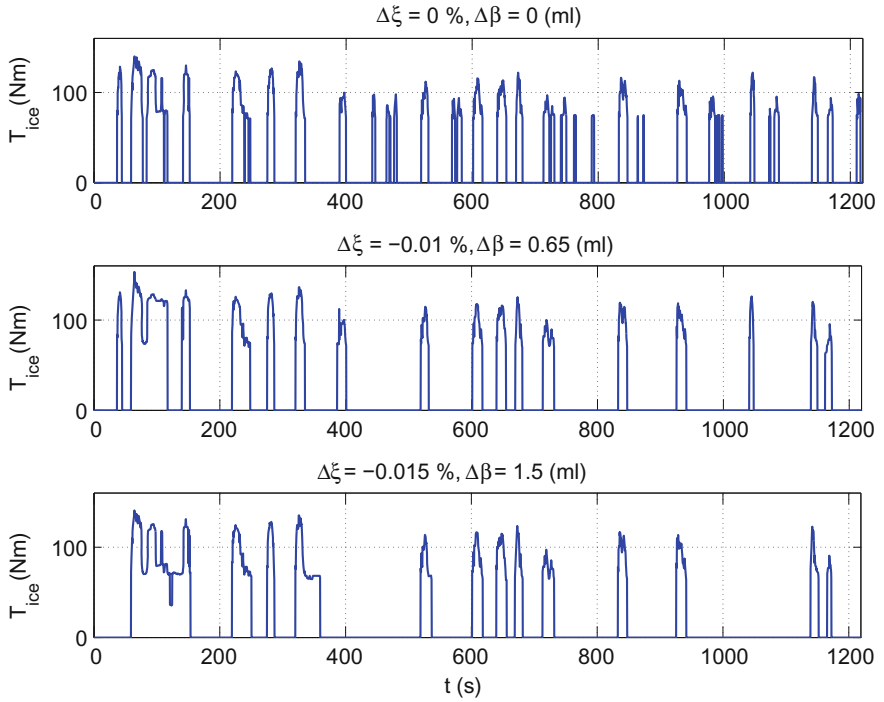


Fig. 11.11 Influence of different state jump heights on the switching sequence $\zeta(\cdot)$ of a part of the FTP-72 test cycle using dynamic programming

Figures 11.12 and 11.13 show the optimization of the switching arc-lengths for the WLTP test cycle. The optimization begins with an embedded direct collocation where the gear sequence is only roughly known. The STO is able to eliminate the unnecessary arcs and to allocate the hybrid drive mode arcs with at least 5 s duration.

In case for moderately present state jumps, the solution trajectories obtained from $\mathcal{P}_1$ will approximate the first-order necessary conditions for switched systems with state jumps sufficiently well. In the case of larger state jumps, the results obtained from $\mathcal{P}_1$ will still be valuable for the calibration of energy management. It is evident from practical evaluations that additional nonlinear elements like hysteresis and delay times can yield solution trajectories close to those obtained by DP.

11.3.3 Lookup Table Calculation

For a simplified rule-based energy management, we are interested in calculating lookup tables for the optimal choice of the gear, the drive mode, and the torque split for a given driving situation. Let us now use discretized variables because the numerical

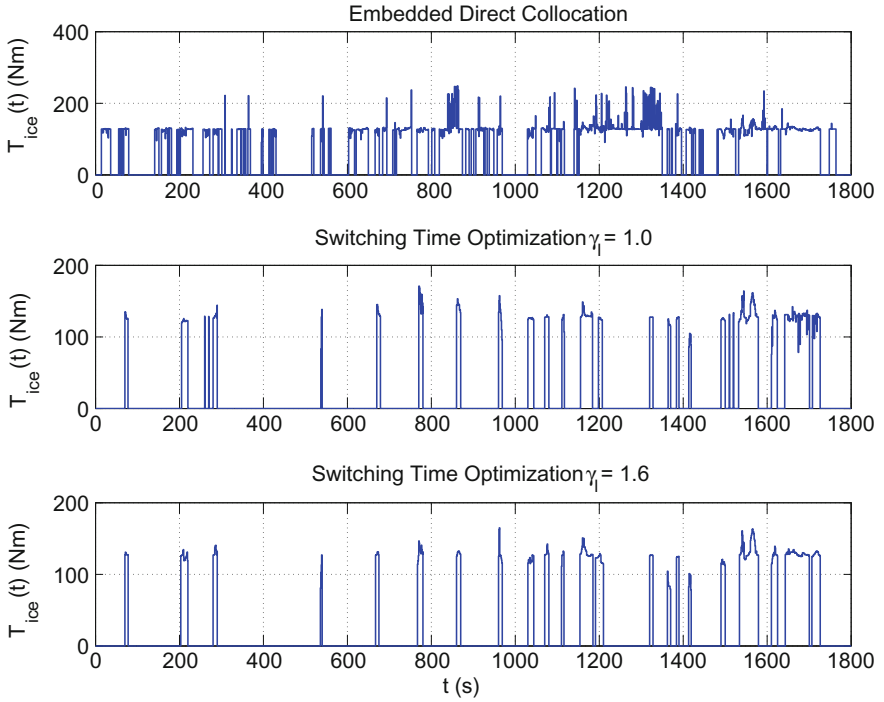


Fig. 11.12 Influence of two different penalties γ_l on the switching sequence $\zeta(\cdot)$ of the WLTP test cycle using embedded direct collocation and STO

calculation is in focus. With the Hamiltonian defined as (11.4), we recapitulate that $\overline{\mathbf{T}}_{ice}^{[k]}$ is the continuous-valued control $u(\cdot)$, $\overline{\mathbf{T}}_{mg}^{[k]}$ is obtained as the difference of the gearbox input torque $\overline{\mathbf{T}}_{gbx}^{[k]}$ and engine torque $\overline{\mathbf{T}}_{ice}^{[k]}$. Since the motor/generator, the gearbox, and the engine are connected axial in a P2 hybrid configuration, the MG speed $\overline{\mathbf{n}}_{mg}^{[k]}$ equals the gearbox input shaft speed $\overline{\mathbf{n}}_{gbx}^{[k]}$ and the engine speed $\overline{\mathbf{n}}_{ice}^{[k]}$ equals $\overline{\mathbf{n}}_{mg}^{[k]}$ for hybrid drive mode and is zero for electric drive mode and hence additionally depends on the discrete decision $\overline{\boldsymbol{\zeta}}_{[k]}$ that determines the drive mode. $\overline{\mathbf{T}}_{gbx}^{[k]}$ and $\overline{\mathbf{n}}_{gbx}^{[k]}$ in turn depend on the wheel torque demand $\overline{\mathbf{T}}_{wh}^{[k]}$, the vehicle velocity $\overline{\mathbf{v}}_{[k]}$, and the selected gear $\overline{\boldsymbol{\kappa}}_{[k]}$, as described in Sect. 10.5.1. It is assumed that the power required to supply the auxiliary devices P_{aux} is known and we assume that the state of charge is sustained to $\xi = 0.5$. We can do so, as $\overline{\boldsymbol{\xi}}_{[k]}$ has only a minor effect on $\dot{\xi}$, as explained Sect. 11.3.1. This simplifies the Hamiltonian and reduces the size of the generated LUTs by one dimension. Also, the costate value is assumed to be known for now. Determining a costate value is one of the major challenges in energy management. Different strategies for determining this value will be given in Sect. 11.4.

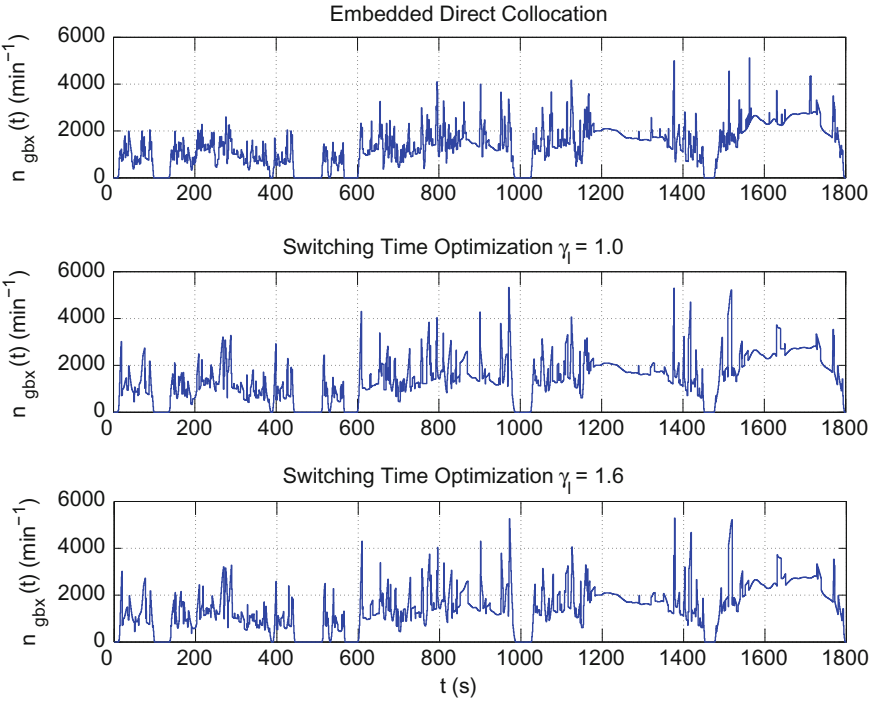


Fig. 11.13 Influence of two different penalties γ_l on the switching sequence $\kappa(\cdot)$ of the WLTP test cycle using embedded direct collocation and STO

For a given value of the costate, the Hamiltonian can be minimized for every quantization point on a grid of $\overline{\mathbf{T}}_{gbx}^{[k]} \in \mathcal{G}_{T_{gbx}}$ and $\overline{\mathbf{n}}_{gbx}^{[k]} \in \mathcal{G}_{n_{gbx}}$ with

$$T_{gbx}^1 < T_{gbx}^2 < \cdots < T_{gbx}^{N_{\mathcal{G}_T}}, \quad \mathcal{G}_{T_{gbx}} = \left\{ T_{gbx}^1, T_{gbx}^2, \ldots, T_{gbx}^{N_{\mathcal{G}_T}} \right\}$$

and

$$n_{gbx}^1 < n_{gbx}^2 < \cdots < n_{gbx}^{N_{\mathcal{G}_n}}, \quad \mathcal{G}_{n_{gbx}} = \left\{ n_{gbx}^1, n_{gbx}^2, \ldots, n_{gbx}^{N_{\mathcal{G}_n}} \right\}$$

to find the optimal torque split in hybrid drive mode for a given driving situation. The Hamiltonian minimum condition yields:

$$\hat{T}_{ice}\left(\overline{\mathbf{T}}_{gbx}^{[k]}, \overline{\mathbf{n}}_{gbx}^{[k]}\right) = \arg\min_{\overline{\mathbf{T}}_{ice}^{[k]}} \mathcal{H}\left(\overline{\mathbf{n}}_{gbx}^{[k]}, \overline{\mathbf{T}}_{gbx}^{[k]}, \lambda, \overline{\mathbf{T}}_{ice}^{[k]}\right). \tag{11.9}$$

Moreover, since the engine torque $\overline{\mathbf{T}}_{ice}^{[k]}$ is a continuous-valued control that appears nonlinear in the Hamiltonian, we can even use the more manageable Hamiltonian minimum condition with

$$\frac{\mathrm{d}\mathcal{H}}{\mathrm{d}\overline{\mathbf{T}}_{ice}^{[k]}}\left(\overline{\mathbf{n}}_{gbx}^{[k]}, \overline{\mathbf{T}}_{gbx}^{[k]}, \lambda, \overline{\mathbf{T}}_{ice}^{[k]}\right) = 0.$$

In the next step, the LUT for the optimal choice of the drive mode $\hat{\zeta}(\cdot)$ can be calculated by

$$\hat{\zeta}\left(\overline{\mathbf{n}}_{gbx}^{[k]}, \overline{\mathbf{T}}_{gbx}^{[k]}\right) = \arg\min_{\overline{\boldsymbol{\zeta}}^{[k]} \in \{0,1\}} \mathcal{H}\left(\overline{\mathbf{n}}_{gbx}^{[k]}, \overline{\mathbf{T}}_{gbx}^{[k]}, \lambda, \overline{\mathbf{T}}_{ice}^{[k]}\right), \tag{11.10}$$

on the same grid.

Since the drive mode $\overline{\boldsymbol{\zeta}}_{[k]}$ is not now regarded as fixed, $\overline{\mathbf{T}}_{ice}^{[k]}$ has the arguments $(n_{gbx}, T_{gbx}, \zeta)$. For $\overline{\boldsymbol{\zeta}}_{[k]} = 1$, $\overline{\mathbf{T}}_{ice}^{[k]}$ is interpolated from the previously calculated LUT $\hat{T}_{ice}$. For $\overline{\boldsymbol{\zeta}}_{[k]} = 0$, $\overline{\mathbf{T}}_{ice}^{[k]} = 0$ applies. For both drive modes, recommended gears $\overline{\boldsymbol{\kappa}}_{[k]}$ can be calculated over a quantization grid of $\overline{\mathbf{T}}_{wh}^{[k]} \in \mathcal{G}_{T_{wh}}$ and $\overline{\mathbf{n}}_{wh}^{[k]} \in \mathcal{G}_{n_{wh}}$ with

$$T_{wh}^1 < T_{wh}^2 < \cdots < T_{wh}^{N_{\mathcal{G}_T}}, \quad \mathcal{G}_{T_{wh}} = \left\{T_{wh}^1, T_{wh}^2, \ldots, T_{wh}^{N_{\mathcal{G}_T}}\right\}$$

and

$$n_{wh}^1 < n_{wh}^2 < \cdots < n_{wh}^{N_{\mathcal{G}_n}}, \quad \mathcal{G}_{n_{wh}} = \left\{n_{wh}^1, n_{wh}^2, \ldots, n_{wh}^{N_{\mathcal{G}_n}}\right\}$$

as follows:

$$\hat{\kappa}\left(\overline{\mathbf{T}}_{wh}^{[k]}, \overline{\mathbf{n}}_{wh}^{[k]}\right) = \arg\min_{\overline{\boldsymbol{\kappa}}_{[k]} \in K} \mathcal{H}\left(\overline{\mathbf{n}}_{gbx}^{[k]}, \overline{\mathbf{T}}_{gbx}^{[k]}, \lambda, \overline{\mathbf{T}}_{ice}^{[k]}\right). \tag{11.11}$$

It should be noted that in this equation, $\overline{\mathbf{n}}_{gbx}^{[k]}$ and $\overline{\mathbf{T}}_{gbx}^{[k]}$ are functions of T_{wh}, n_{wh}, and κ. The required ICE-torque for hybrid mode is again interpolated from LUT $\hat{T}_{ice}$.

Principally, these maps can be calculated for an arbitrary number of costate values and the costate can be seen as an additional axis of interpolation. LUTs that have the costate as additional input are depicted in Figs. 11.14 and 11.15. The first LUT shows the MG-torque $\overline{\mathbf{T}}_{mg}^{[k]}$ depending on gearbox input torque and speed and the costate as an additional axis. It is more common in automotive practice, to store the MG-torques instead of the ICE-torques. The lower LUT shows the torque limit $\hat{T}_{start}$. As soon as the gearbox input torque exceeds this torque limit for a given driving situation, an engine start is requested. $\hat{T}_{start}$ can be calculated from the LUT $\hat{\zeta}$ for a given costate.

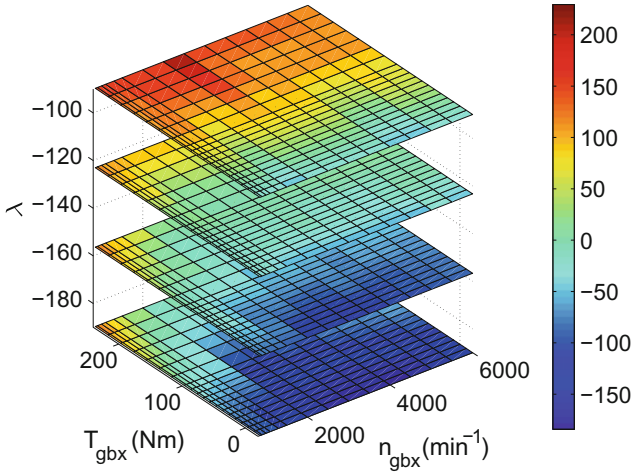
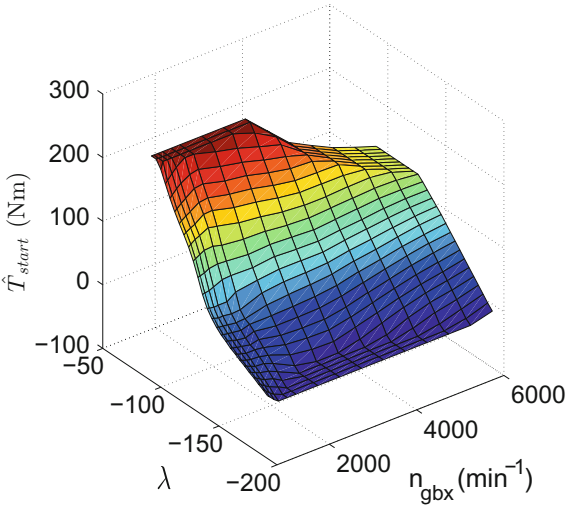


Fig. 11.14 Exemplary LUTs for MG-torque $\hat{T}_{mg}$

Fig. 11.15 Exemplary LUT for start-torque $\hat{T}_{start}$



11.4 Rule-Based Strategies for Choosing the Costate

Having stored optimal continuous-valued and discrete controls for a given costate value and for all driving situations in LUTs, the remaining problem for energy management is the determination of the costate value. A very appealing approach for determining the costate λ during vehicle operation is a predictive energy management that uses available information on the future driving route to make a good estimation of the costate. Predictive energy managements will be treated separately

in Chap. 12. The focus in this section is on strategies that are implementable without knowledge of the future driving profile.

11.4.1 Rule-Based Selection Using Costate Maps

For *charge-sustaining* (CS) of unknown drive cycles, the concept of costate maps is introduced. The costate is herein selected depending on the current road type and the current state of charge (see Fig. 11.16). Other information can also be used as shown in Boehme et al. [1]. Each color represents one costate value and therefore one set of LUTs. The LUT procedure described in the previous sections is employed to calculate $\hat{T}_{mg}$ and $\hat{T}_{start}$ maps for several costate values

$$\lambda_1 < \lambda_2 < \lambda_3 < \lambda_4 < \lambda_5,$$

where λ_2 is chosen by solving the SOCP (11.1) or (11.5) for different urban scenarios. λ_3 and λ_4 are selected to sustain $\xi(\cdot)$ on rural and highway scenarios, respectively. λ_1 and λ_5 are chosen to reliably protect the high-voltage battery from being depleted too far or from being overcharged. The current road type can either be obtained from the navigation system using an electric horizon (cf. Sect. 12.3) but can also easily be determined using the vehicle speed averaged over a short time span. Modern PHEV-vehicles can also leave the choice of an operation mode to the driver. A *charge-depleting* (CD) mode, that intends to minimize the fuel consumption used over an unknown route until the next recharge facility is available, can be implemented by using different costate maps where the next higher heuristically determined costate value is employed until a low state of charge is reached. One can observe from Fig. 11.16 that for the CD mode, higher costate values are employed for most of the ξ-range than in the CS mode.

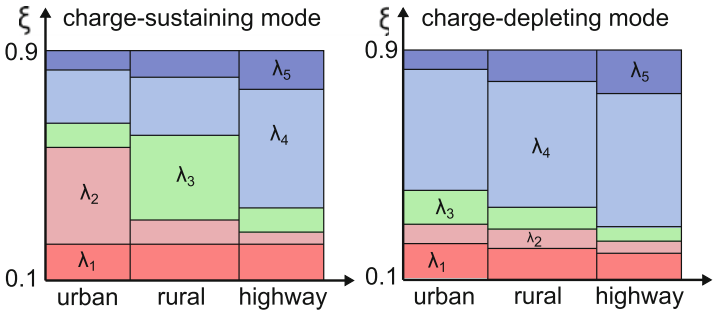


Fig. 11.16 Exemplary costate maps for CS and CD mode

11.4.2 Costate for Optimal CO_2 Emissions

If a CO_2-optimal strategy is desired (cf. Schori et al. [24]), the objective function of the SOCP can be stated as

$$\phi_{CO_2}(\mathbf{x}(t)) = r_1 \cdot \beta(t_f) + r_2 \cdot (\xi_0 - \xi(t_f)),$$

where r_1 is a constant factor, that expresses how much CO_2 is emitted for every liter of fuel used, which corresponds to the engine brake specific CO_2 (Stockar et al. [30]). The factor r_2 resembles the CO_2 cost for the electric energy consumed from the battery and includes information about the energy mix in the respective country, a charging efficiency η_{grid}, natural constants, and the battery capacity Q_{bat}. The Hamiltonian can then be defined as

$$\mathcal{H}_{CO_2}(\mathbf{x}(t), q(t), \boldsymbol{\lambda}(t), u(t)) = \lambda_{1,CO_2}(t) \cdot \dot{\beta}(t) + \lambda_{2,CO_2}(t) \cdot \dot{\xi}(t).$$

Evaluating the transversality condition (4.46) yields

$$\lambda_{1,CO_2}(t_f) = \frac{\partial \phi_{CO_2}}{\partial \beta(t_f)}(\mathbf{x}(t_f)) = r_1$$
$$\lambda_{2,CO_2}(t_f) = \frac{\partial \phi_{CO_2}}{\partial \xi(t_f)}(\mathbf{x}(t_f)) = -r_2.$$

The dynamics of the costates are given by

$$\dot{\lambda}_{1,CO_2}(t) = -\frac{\partial \mathcal{H}_{CO_2}}{\partial \beta}(\mathbf{x}(t), q(t), \boldsymbol{\lambda}(t), u(t)) = 0$$
$$\dot{\lambda}_{2,CO_2}(t) = -\frac{\partial \mathcal{H}_{CO_2}}{\partial \xi}(\mathbf{x}(t), q(t), \boldsymbol{\lambda}(t), u(t)) \approx 0.$$

The Hamiltonian for a CO_2-optimal drive mode can thus be written as

$$\mathcal{H}_{CO_2}(\mathbf{x}(t), q(t), \boldsymbol{\lambda}(t), u(t)) = r_1 \cdot \dot{\beta}(t) - r_2 \cdot \dot{\xi}(t). \tag{11.12}$$

Again, we exploit the fact that $\mathrm{d}V_{oc}/\mathrm{d}\xi$ is small for modern batteries. Minimizing the Hamiltonian (11.12) for obtaining the continuous-valued control $u(\cdot)$ yields the same result as minimizing the scaled Hamiltonian

$$\hat{\mathcal{H}}_{CO_2}(\mathbf{x}(t), q(t), \boldsymbol{\lambda}(t), u(t)) = \dot{\beta}(t) - \frac{r_2}{r_1} \cdot \dot{\xi}(t).$$

Note that for the CO_2 optimal drive mode, the costate can be determined by setting $\lambda_{CO_2}(t) = -r_2/r_1$ to a constant value. It is remarkable that the CO_2 costate now becomes an analytical expression, whereas for the fuel-optimal mode, the costate

must be determined from an optimization procedure. The alternating and country-specific factor r_2 can be transferred to the vehicle from the charging station or via a wireless network connection. The costate calculated this way will usually lead to a CD behavior. To prevent the battery from being depleted so far, the drive mode switches to the rule-based mode using λ-maps for low values of $\xi(\cdot)$.

11.5 Implementation Issues

Some implementation issues need to be considered when implementing rule-based energy management strategies:

The LUT $\hat{T}_{mg}$ can usually be implemented in a PHEV without further modification. In some cases, the map might require further smoothing to avoid jumps in the desired MG-torque. However, countless numerical experiments have shown that this manipulation hardly affects the fuel consumption.

Equations (11.10) and (11.11) can easily be evaluated for systems without switching costs only. Switching costs introduce discontinuities in the state trajectory and thus require the hybrid execution sequence to be known in advance, which is without further aid for practical problems nearly impossible. However, practical experience has shown that the LUTs derived from these equations can still be used for systems with switching costs. In combination with hysteresis and delay times, these LUTs obtain results close to those calculated offline using dynamic programming. The number of hysteresis and delay parameters is low and can hence be found by applying derivative-free optimization methods such as evolutionary algorithms (see Sect. 2.5.1).

To determine the recommended gears, additional constraints such as limitations due to driving comfort apply. These factors are hard to account for in a mathematical model. As a consequence, the gear-shifting recommendations cannot always be followed and the calculated LUTs serve rather as a basis for further analysis and driving experiments than ready-to-use vehicle applications. It has been shown to be helpful to evaluate the effect of deviating from the recommended solution. If the values from the LUTs are used, in general, Eq. (4.126) holds and the Hamiltonian is continuous during a change in the piecewise constant discrete state $q(\cdot)$. For instance, a transition from one drive mode to another or at gear changes. When deviating from the recommended transitions, a difference in the Hamiltonian jump condition

$$\Delta\mathcal{H} = \mathcal{H}(\mathbf{x}(t_j^+), q(t_j^+), \boldsymbol{\lambda}(t_j^+), u(t_j^+)) - \mathcal{H}(\mathbf{x}(t_j^-), q(t_j^-), \boldsymbol{\lambda}(t_j^-), u(t_j^-))$$

occurs. The meaning of this difference is twofold: on the one hand, it constitutes a deviation from the optimality conditions; on the other hand, with the interpretation of the Hamiltonian as the weighted sum of battery current and fuel mass flow, it indicates that a continuous-valued control $u(\cdot)$ with lower value of this weighted

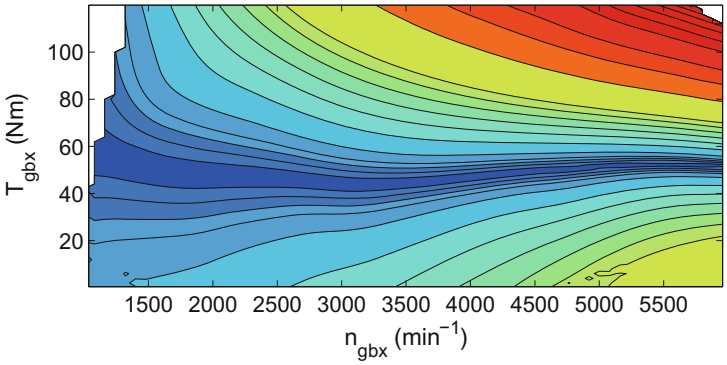


Fig. 11.17 $|\Delta\mathcal{H}|$ for hybrid and electric drive mode calculated for a given costate value

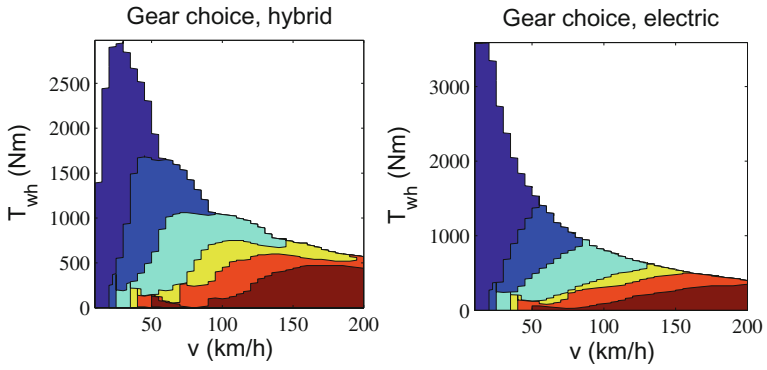


Fig. 11.18 Gear choice maps for electric and hybrid drive modes

sum exists, but cannot be used because of some constraints. Additionally, with the constant costate assumption, the value of the Hamiltonian does not explicitly depend on time, but on the current driving situation. Figure 11.17 shows the absolute value $|\Delta\mathcal{H}|$ depending on $T_{gbx}(\cdot)$ and $n_{gbx}(\cdot)$ for electric and hybrid drive mode. The recommended switching is where the difference vanishes. If this recommended switching cannot be followed, Fig. 11.17 allows for an evaluation of the effects. A deviation from the recommended switching is more acceptable, when the value of $|\Delta\mathcal{H}|$ is low.

Defining the gear map based on the costate-map might also cause the vehicle behavior to change unexpectedly for the driver and is therefore not recommended. However, defining separate gear maps for electric and hybrid drive modes can yield considerable efficiency improvements, as the recommended gear choices may differ strongly (see Fig. 11.18).

11.6 Bibliography

Many different energy management strategies have been reported in the literature. Comparative overviews are given in Salmasi [20], Hofman et al. [7], Serrao et al. [28], and Wang et al. [34].

Early publications on energy management of hybrid vehicles often had a focus on rule-based concepts with the advantage of being completely causal and hence directly applicable to given hybrid powertrain configurations. Energy management based on a fuzzy logic controller was proposed by Schouten et al. [25, 26], and Farrokhi and Mohebbi [4]. In the work Lin et al. [16], the results of a cycle-specific solution obtained by dynamic programming are used for the manual definition of rules. A very similar approach was chosen by Karbowski et al. [11]. A calibration method for LUTs of rule-based energy management strategies is proposed by Schori [21, 22].

Over the last decade, research in the area of hybrid vehicle calibration has focused more and more on analytical methods. The most attention has been paid to the PMP and the related ECMS-strategies. ECMS was originally proposed in Paganelli et al. [18] and expanded in Chen and Salman [2], Musardo et al. [17] among others. Tulpule et al. [32] applied a modified version of the ECMS strategy to PHEVs.

A fuel minimization strategy based on PMP is proposed in Kim et al. [13, 14], and Serrao and Rizzoni [27] where at any time instant the continuous-valued control is determined by minimizing the Hamiltonian. The costate necessary for setting up the Hamiltonian is determined by evaluating driving patterns from the past. A similar approach is presented in Stockar et al. [30] with a weaker focus on online implementation but more stressed on the influence of different usage conditions, environmental factors, and geographic scenarios. Additionally, the optimal parameter choice of controller parameters for minimizing cumulative CO_2-emissions strongly depends on the local energy mix, which is demonstrated in Elgowainy et al. [3].

The problem of finding an appropriate costate value is tackled in different ways. Kim et al. [13] suggested to select the costate based on heuristically defined values that represent the drive pattern. A learning procedure that corrects the costate value, when a lower or upper bound for the state of charge is hit, is described in the work of Chen and Salman [2]. A study on the relationship between different road-type events and the costate value that leads to a charge-sustaining operation of the HEV was performed by Gong et al. [5]. The idea of storing the minimum of the Hamiltonian with respect to the continuous-valued control was also proposed by Chen and Salman [2] in the context of ECMS. Also with the ECMS-formulations, it was extended to the gear choice by Sivertsson et al. [29].

A constant costate is widely assumed in the literature and its effect has been investigated in several prior studies by Guzzella and Sciarretta [6], Kim et al. [14] for charge-sustaining operation, where the state of charge varies only in a narrow window. It is shown in Sivertsson et al. [29] and Schori et al. [21] that saving the continuous-valued control on a quantized grid has only minor effects on the fuel consumption.

The embedding approach using a direct collocation method has been proposed in Uthaichana et al. [33] for the optimization of a two-mode operation of a parallel HEV.

Stochastic dynamic programming has been used by different authors to minimize the fuel consumption; among them Kim et al. [12] and Tate et al. [31].

Control strategies for minimizing fuel consumption and emissions have been regarded by many authors, among them Johnson et al. [10], Tate et al. [9], Kum et al. [15].

Rousseau et al. [19] optimized the parameter of the energy control strategy using a derivative-free algorithm called DIRECT. Similar instantaneous optimization strategies have been proposed by Johnson et al. [10], Tate et al. [9], Huang et al. [8]

References

1. Boehme TJ, Sehnke T, Schultalbers M, Jeinsch T (2014) Implementation of an optimal control like energy management for hybrid vehicles based on driving profiles. In: SAE world congress, Technical paper 2014-01-1903. doi:10.4271/2014-01-1903
2. Chen JS, Salman M (2005) Learning energy management strategy for hybrid electric vehicles. In: IEEE conference on vehicle power and propulsion, pp 68–73
3. Elgowainy A, Burnham A, Wang M, Molburg J, Rousseau A (2009) Well-to-wheels energy use and greenhouse gas emissions of plug-in hybrid electric vehicles. In: SAE world congress, Technical paper 2009-01-1309
4. Farrokhi M, Mohebbi M (2005) Optimal fuzzy control of parallel hybrid electric vehicles. ICCAS2005 June, pp 2–5
5. Gong Q, Tulpule P, Marano V, Midlam-Mohler S, Rizzoni G (2011) The role of ITS in PHEV performance improvement. In: Proceedings of the 2011 American control conference, on O'Farrell Street. IEEE, San Francisco, pp 119–124
6. Guzzella L, Sciarretta A (2005) Vehicle propulsion systems. Introduction to modeling and optimization. Springer, Berlin
7. Hofman T, Steinbuch M, Van Druten R, Serrarens A (2008) Rule-based equivalent fuel consumption minimization strategies for hybrid vehicles. In: Proceedings of the 17th IFAC world congress, pp 5652–5657
8. Huang Y, Yin C, Zhang J (2008) Modeling and development of the real-time control strategy for parallel hybrid electric urban buses. WSEAS Trans Inf Sci Appl 5(7):1113–1126
9. Huang Y, Yin C, Zhang J (2008) Optimal torque distribution control strategy for parallel hybrid electric urban buses. WSEAS Trans Syst 7(6):758–773
10. Johnson VH, Wipke KB, Rausen DJ (2000) HEV control strategy for real-time optimization of fuel economy and emissions. In: SAE world congress, Technical paper 2000-01-1543
11. Karbowski D, Rousseau A, Pagerit S, Sharer P (2006) Plug-in vehicle control strategy: from global optimization to real time application. In: 22nd electric vehicle symposium, EVS22, Yokohama, Japan
12. Kim MJ, Peng H, Lin CC, Stamos E, Tran D (2005) Testing, modeling, and control of a fuel cell hybrid vehicle. In: Proceedings of the 2005 American control conference. IEEE, pp 3859–3864
13. Kim N, Lee D, Cha SW, Peng H (2009) Optimal control of a plug-in hybrid electric vehicle (PHEV) based on driving patterns. In: International battery, hybrid and fuel cell electric vehicle symposium
14. Kim N, Cha S, Peng H (2011) Optimal control of hybrid electric vehicles based on Pontryagin's minimum principal. IEEE Trans Control Syst Technol 1279–1287

15. Kum D, Peng H, Bucknor N (2011) Supervisory control of parallel hybrid electric vehicles for fuel and emission reduction. ASME J Dyn Syst Measur Control 133(6):4498–4503
16. Lin CC, Kang JM, Grizzle JW, Peng H (2001) Energy management strategy for a parallel hybrid electric truck. In: Proceedings of the 2001 American control conference, Arlington, vol 4, pp 2878–2883
17. Musardo C, Rizzoni G, Staccia B (2005) A-ECMS: An adaptive algorithm for hybrid electric vehicle energy management. In: Proceedings of the 44th IEEE conference on decision and control, pp 1816–1823
18. Paganelli G, Guezennec Y, Rizzoni G (2002) Optimizing control strategy for hybrid fuel cell vehicle. In: SAE world congress, Technical paper 2002-01-0102. doi:10.4271/2002-01-0102
19. Rousseau A, Pagerit S, Gao DW (2008) Plug-in hybrid electric vehicle control strategy parameter optimization. J Asian Electr Veh 6(2):1125–1133
20. Salmasi FR (2007) Control strategies for hybrid electric vehicles: Evolution, classification, comparison, and future trends. IEEE Trans Veh Technol 56(5):2393–2404
21. Schori M, Boehme TJ, Frank B, Schultalbers M (2013) Calibration of parallel hybrid vehicles based on hybrid optimal control theory. In: 9th IFAC symposium on nonlinear control systems (NOLCOS), Toulouse, pp 475–480
22. Schori M, Boehme TJ, Frank B, Schultalbers M (2013) Solution of a hybrid optimal control problem for a parallel hybrid vehicle. In: 7th IFAC symposium on advances in automotive control (AAC), Tokyo, pp 109–114
23. Schori M, Boehme T, Jeinsch T, Schultalbers M (2014) Optimal catalytic converter heating in hybrid vehicles. In: SAE world congress, Technical pper 2014-01-1351. doi:10.4271/2014-01-1351
24. Schori M, Boehme TJ, Frank B, Lampe B (2014) Optimal calibration of map-based energy management for plug-in parallel hybrid configurations: a hybrid optimal control approach. IEEE Trans Veh Technol 64(9):3897–3907. doi:10.1109/TVT.2014.2363877
25. Schouten N, Salman M, Kheir N (2002) Fuzzy logic control for parallel hybrid vehicles. IEEE Trans Control Syst Technol 10(3):460–468
26. Schouten NJ, Salman MA, Kheir NA (2003) Energy management strategies for parallel hybrid vehicles using fuzzy logic. Control Eng Pract 11:171–177
27. Serrao L, Rizzoni G (2008) Optimal control of power split for a hybrid refuse vehicle. In: Proceedings of the American control conference. IEEE, pp 4498–4503
28. Serrao L, Onori S, Rizzoni G (2011) A comparative analysis of energy management strategies for hybrid electric vehicles. J Dyn Syst Measur Control 133(3):031,012. doi:10.1115/1.4003267
29. Sivertsson M, Sundstroem C, Eriksson L (2011) Adaptive control of a hybrid powertrain with map-based ECMS. In: Proceedings of the IFAC world congress, pp 357–362
30. Stockar S, Marano V, Canova M, Rizzoni G, Guzzella L (2011) Energy-optimal control of plug-in hybrid electric vehicles for real-world driving cycles. IEEE Trans Veh Control 60(7):2949–2962
31. Tate ED, Grizzle JW, Peng H (2010) SP-SDP for fuel consumption and tailpipe emissions minimization in an EVT hybrid. IEEE Trans Control Syst Technol 18(3):673–687
32. Tulpule P, Marano V, Rizzoni G (2009) Effects of different PHEV control strategies on vehicle performance. In: Proceedings of the 2009 American control conference, St. Louis, pp 3950–3955
33. Uthaichana K, Bengea S, Decarlo R, Pekarek S, Zefran M (2008) Hybrid model predictive control tracking of a sawtooth driving profile for an HEV. In: American control conference, 2008. IEEE, pp 967–974
34. Wang X, He H, Sun F, Sun X, Tang H (2013) Comparative study on different energy management strategies for plug-in hybrid electric vehicles. Energies 6

Chapter 12
Predictive Real-Time Energy Management

12.1 Introduction

The optimal control solutions from the previous chapter cannot be implemented directly on a real vehicle, since disturbances and changing drive conditions can effect heavily the optimal solution trajectories. In order to make the feedforward results robust against these disturbances we need some form of feedback of the current state. This chapter concentrates on the closed-loop realization of energy management as shown in Fig. 12.1. The feedback of the state in Fig. 12.1 can be realized as permanent feedback or as event-based feedback.

In general, the real-time control strategies $\mathcal{K}(\mathbf{x}(t))$ for energy management can be classified as

- rule-based strategy;
- equivalent consumption minimization strategy;
- model-predictive control strategy.

Rule-based (RB) strategies (see Lin et al. [29], Schouten et al. [43, 44], Farrokhi and Mohebbi [11], Karbowski et al. [21], and Schori et al. [39, 40]) as discussed in Chap. 11 rely on multidimensional control maps and have the advantage of being completely causal and hence directly applicable to given hybrid powertrain configurations. However, a wide number of parameters depend strongly on the ad hoc calibration, which is a time-consuming and cumbersome task. This limits the results of a rule-based strategy usually to one hybrid powertrain configuration.

Equivalent consumption minimization strategies (ECMS) (Paganelli et al. [45], Serrao et al. [35]) reduce an optimization problem to an instantaneous minimization problem as shown in Sect. 1.3.2. The strength of this method is certainly the low computation time, which makes it a candidate for the implementation as a real-time control strategy. However, the determination of a meaningful equivalence factor can be a hard task.

Model-predictive control (MPC) strategies, which are often cited in the literature as a promising control implementation for a more general energy management, depend on accurate sensor information and powerful *electronic control units* (ECU). This control strategy is often used for a rather short prediction horizon, as proposed by many authors (see for instance Back [1], Borhan et al. [8]).

T.J. Böhme and B. Frank, *Hybrid Systems, Optimal Control and Hybrid Vehicles*,
Advances in Industrial Control, DOI 10.1007/978-3-319-51317-1_12

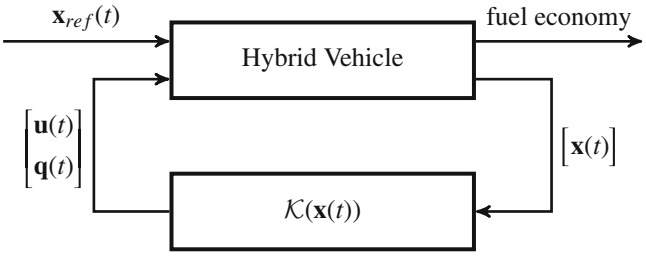


Fig. 12.1 General structure of a real-time control strategy for energy management

The MPC strategies discussed here are closely linked to optimal control techniques. However, the on-board solution of *optimal control problems* (OCP) is on the one hand usually prevented by the limited computing capacity of the ECU. Despite some online optimization strategies proposed by some authors (e.g., Ferreau et al. [12]) in the recent past, their applications upon the ECU is still challenging. On the other hand, the solution is based on a predefined drive cycle given by the trajectories of velocity $v(\cdot)$ and road slope $\alpha(\cdot)$, which are generally unknown for real-world drive cycles. It is, however, possible, to predict the trajectories based on the information provided by an *intelligent traffic system* (ITS) as it is included in many recent navigation systems [37]. The developments in this area are continuously growing such that more and more additional road information will be known to energy management. With the help of this finite set of information, an estimation of the trajectories can be made and these estimations can be used for solving the OCP. The existence of discrete phenomena makes the problems harder to solve. However, it has been shown in Chap. 11, that the indirect shooting method is highly efficient on the problems $\mathcal{P}_1$ and $\mathcal{P}_2$ if state constraints can be ignored. The *switched optimal control problem* (SOCP) is reduced to find only a single initial costate. At the same time, as shown in Sect. 11.5, the optimal continuous-valued controls for a given costate and a given driving situation can easily be stored in *lookup tables* (LUT) without significant loss of accuracy. These facts make the on-board implementation of a real-time control strategy that is based on the solution of a SOCP possible.

This chapter discusses three different energy management problems for on-board implementation:

- predictive real-time trip management for *battery electric vehicles* (BEV);
- predictive real-time energy management with medium prediction horizon and event-based feedback for *hybrid electric vehicles* (HEV); and
- predictive real-time energy management with long prediction horizon for *plug-in hybrid electric vehicles* (PHEV).

12.2 Real-World Benchmark-Cycles

For evaluation of the predictive energy management strategies, several real-world benchmark-cycles are chosen additionally to Sect. 10.6. All located in the north of Germany, which should reflect typical driving scenarios. The real-world benchmark-cycles, depicted in Figs. 12.2, 12.3, and 12.4, contain different contributions of urban, rural, highway, and mountain road traffic-situations. The second real-world benchmark-cycle is characterized by low average speeds and a long driving part through the city of Gifhorn with a high probability of stop events and therefore a high number of accelerations and decelerations. The third benchmark-cycle is characterized by a routing with urban, rural, and large highway parts with a day-time-dependent high traffic volume. The fourth benchmark-cycle is a typical mountain road which starts from Braunlage in Oberharz via Sankt Andreasberg and ends in Braunlage again. As can be seen from Fig. 12.4 the trip consists of two valleys associated with long up-hill and down-hill driving parts. Furthermore, the cycle includes urban and rural parts as well.

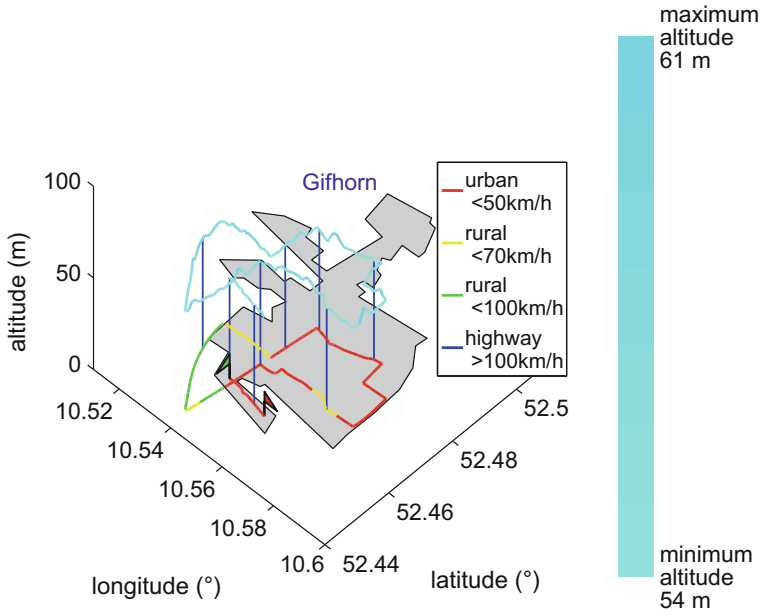


Fig. 12.2 Route map of the real-world benchmark-cycle 2

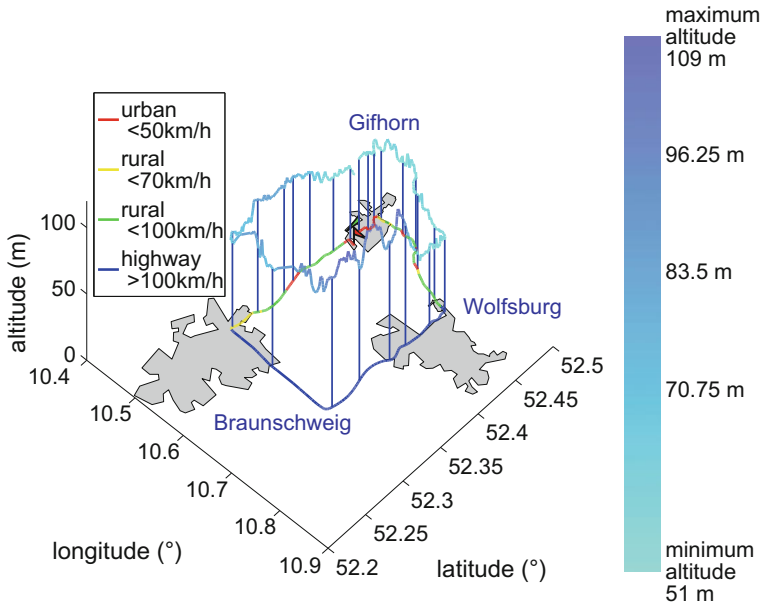


Fig. 12.3 Route map of the real-world benchmark-cycle 3

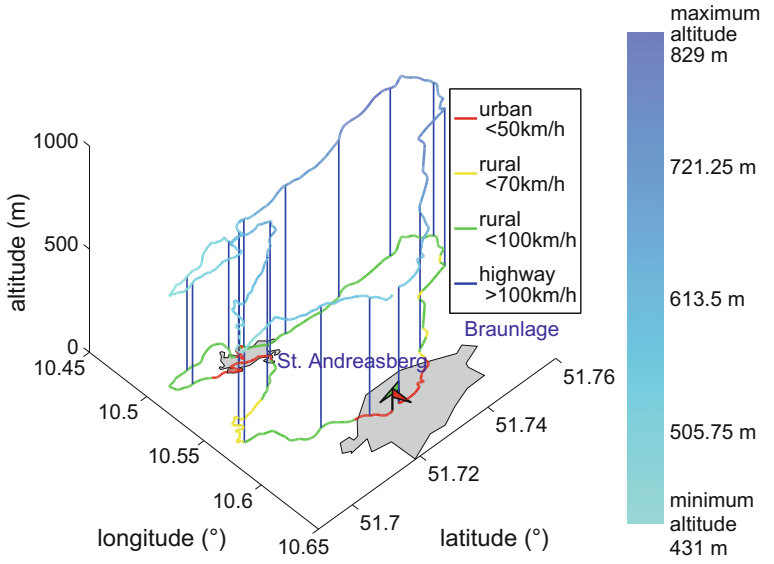


Fig. 12.4 Route map of the real-world benchmark-cycle 4

12.3 Intelligent Traffic System

In order to solve an OCP a predefined drive cycle based on the trajectories of velocity $v(\cdot)$ and road slope $\alpha(\cdot)$, which are generally unknown on real-world drivings, is necessary. A state-of-the-art approach is to employ an electronic horizon provider, which provides a finite set of routing information. This routing information is fed to a driver model to obtain a good estimation of the desired trajectories, which can be used for predictive energy management.

The electronic horizon is based on the transceiver/receiver principle. The transceiver is known as horizon provider and the receiver of an application is called the reconstructor. A horizon provider is implemented in a MATLAB®/Simulink® environment which accesses parts of the noncommercial *OpenStreetMap* database [48] and runs on a Windows PC to mimic a modern navigation system with *geographic information system* (GIS) conform database. The navigation system needs the vehicle position to determine all relevant main-paths and sub-paths in the target direction for the prediction horizon. At the initialization of the navigation system, the vehicle position is determined from a differential GPS device. During the prediction the in-vehicle increment sensor of the wheel is used to measure the vehicle speed and thus the vehicle position. The communication between horizon provider and reconstructor is performed using the standardized *advanced driver assistance systems* (ADAS) protocol [37] that provides in this implementation the following attributes:

- speed limit;
- form of way;
- stop points (intersection without right of way); and
- road inclination.

The curve radii is an attribute which is available in nowadays GIS systems, but seldom used to improve the quality of energy management.

The route is then divided into segments on the spatial grid

$$s_{seg,0} < s_{seg,1} < \cdots < s_{seg,N_r}, \quad \mathcal{G}_{seg} = \left\{ s_{seg,0}, s_{seg,1}, \ldots, s_{seg,N_r} \right\}$$

based on the changes of speed limits and the road inclination [5].

Since dynamic memory allocation is in general not allowed in today's ECUs, the first $N_r + 1$ segments of speed limits and road slopes are then stored in static allocated vectors

$$\bar{\mathbf{s}}_{seg} = \left[s_{seg,0}, s_{seg,1}, \ldots, s_{seg,N_r} \right] \tag{12.1}$$

$$\bar{\mathbf{v}}_{seg} = \left[v_{seg,0}, v_{seg,1}, \ldots, v_{seg,N_r} \right] \tag{12.2}$$

$$\bar{\boldsymbol{\alpha}}_{seg} = \left[\alpha_{seg,0}, \alpha_{seg,1}, \ldots, \alpha_{seg,N_r} \right] \tag{12.3}$$

and transferred to the driver model that generates a vehicle speed trajectory $v(\cdot)$, which will be discussed in the next sections.

The reader should note that the segments are not equidistant due to the characteristics of the topology of the trip. For example, some parts of the trip may contain more changes in the altitude than other ones or contain a higher number of curves which in both cases imposes a higher number of segments in order to describe the topology as accurately as desired.

12.3.1 Time-Based Driver Model

The aim of using the driver model is to predict 10 min ahead the vehicle speed trajectory as realistic as possible. The trajectories in the time domain are calculated over an uniform grid (5.9) of constant size $N_t + 1$. The time instances t_k, $k = 0, 1, \ldots, N_t$ are stored in the vector

$$\bar{\mathbf{t}} = \left[t_0, t_1, \ldots, t_{N_t}\right].$$

The time grid is chosen to be equidistant, then the step-length

$$h = t_k - t_{k-1} = \text{const}, \quad k = 1, \ldots, N_t$$

applies.

Many investigations has been performed in the past six decades to model the longitudinal interaction between adjacent vehicles. Those approaches can be roughly distinguished between macroscopic or microscopic traffic simulations. Macroscopic models try to predict the traffic behavior upon statistical quantities such as traffic flow and traffic density, whereas microscopic models are based on self-organized agents. A Markov chain model is often proposed as macroscopic model (Gong et al. [17]), as it is able to reflect the statistical behavior in the real profile. However, the expenditure for the determination of the respective transition matrices is quite high and the matrices would require a large amount of storage in the ECU. Alternatively, one popular agent-class is depending on the reaction of the adjacent vehicles in front and is known as car-following model, where the most popular is the Gipps model (Gipps [15]). An overview of different car-following models can be found in [36]. A further development is the continuous-time single-lane *intelligent-driver-model* (IDM) proposed by Treiber and Kesting [46]. It provides a set of differential equations to imitate the driver's behavior for different driving situations based on the available information of obstacles in front. The time domain description makes it easily parameterizable to different driver types δ and motivated us to use parts of the IDM to model acceleration and deceleration scenarios to the next speed limits which are provided by the electronic horizon.

The velocity derivation is approximated by the explicit Euler approach as follows:

$$\overline{\mathbf{v}}_p^{[k+1]} = \begin{cases} \overline{\mathbf{v}}_p^{[k]} + h \cdot \left[1 - \left(\dfrac{\overline{\mathbf{v}}_p^{[k]}}{v_{lim}} \right)^{\delta} \right] \cdot e, & d = 1 \\ \overline{\mathbf{v}}_p^{[k]} - h \cdot \left[1 - \left(\dfrac{v_{lim}}{\overline{\mathbf{v}}_p^{[k]}} \right)^{\delta} \right] \cdot e, & d = 2 \\ \overline{\mathbf{v}}_p^{[k]} + h \cdot \left(\dfrac{\overline{\mathbf{v}}_p^{[k]} \cdot \Delta v}{2 s_{rem}} \right)^2 \cdot \dfrac{1}{f}, & d = 3 \end{cases}$$

where e and f denote driver-dependent average accelerations/decelerations values, s_{rem} and Δv denote the actual spacing and the difference speed to the next speed limit v_{lim}, respectively. Since the difference speed Δv to a nonmoving traffic sign is the current vehicle speed the last term can be simplified to $(\overline{\mathbf{v}}_p^{[k]})^2 / 2 s_{rem}$. The driving situation $d = 1$ defines an acceleration on the free road to the speed limit above the current speed, $d = 2$ models the deceleration situation when approaching a speed limit below the current speed, and $d = 3$ describes a complete deceleration up to standstill. The speed limits are treated as spatial fixed obstacles in the IDM model.

Defining the vectors

$$\overline{\mathbf{v}}_p = \left[v_{p,0}, v_{p,1}, \ldots, v_{p,N_t-1} \right]^T \tag{12.4}$$

$$\overline{\mathbf{a}}_p = \left[a_{p,0}, a_{p,1}, \ldots, a_{p,N_t-1} \right]^T \tag{12.5}$$

$$\overline{\boldsymbol{\alpha}}_p = \left[\alpha_{p,0}, \alpha_{p,1}, \ldots, \alpha_{p,N_t-1} \right]^T, \tag{12.6}$$

the predicted speed, distance, and acceleration are calculated as

$$\overline{\mathbf{v}}_p^{[k]} = \max\left(0, \overline{\mathbf{v}}_p^{[k]}\right) \tag{12.7}$$

$$\overline{\mathbf{s}}_p^{[k+1]} = \overline{\mathbf{s}}_p^{[k]} + h \cdot \overline{\mathbf{v}}_p^{[k]} \tag{12.8}$$

$$\overline{\mathbf{a}}_p^{[k]} = \frac{\overline{\mathbf{v}}_p^{[k+1]} - \overline{\mathbf{v}}_p^{[k]}}{h}. \tag{12.9}$$

In each time instance, the driver model is computed with the quantities v_{lim} and s_{rem}. The electronic horizon provides static allocated arrays to retrieve the speed limits v_{lim} and the spacing s_{rem} to the next traffic signs. The adapted IDM model is enhanced by a range-of-sight

$$r_s := p_0 \cdot \left(\overline{\mathbf{v}}_p^{[k]}\right)^2 + p_1 \cdot \overline{\mathbf{v}}_p^{[k]}$$

where p_0 and p_1 again are driver-dependent values. This should imitate the human perception and starts the braking procedure when a lower speed limit or a stop position has been detected to be within the range-of-sight.

Investigations have shown that the values of the costate for the state of charge determined from the predicted vehicle speed trajectory are close to the values which are determined from experimentally measured vehicle speed trajectory. In other words, the solution of the OCP is robust against some errors in the prediction. It should be noted that the considered concept of spatial fixed speed limits can be easily extended to incorporate fixed or moving traffic flow speed limits, e.g., moving traffic jams.

The depicted Algorithm 12.1 sketches the calculation procedure for the time-based driver model.

Algorithm 12.1 Sketch of the time-based driver model

initialization: fill the static allocated vectors $\bar{\mathbf{s}}_{seg}$, $\bar{\mathbf{v}}_{seg}$, and $\bar{\boldsymbol{\alpha}}_{seg}$
set the segment counter $j \leftarrow 0$
for $k := 0$ **to** N_t **do**
    $\bar{\boldsymbol{\alpha}}_p^{[k]} \leftarrow \bar{\boldsymbol{\alpha}}_{seg}^{[j]}$
    $s_{rem} = \bar{\mathbf{s}}_{seg}^{[j]} - \bar{\mathbf{s}}_p^{[k]}$
    if $s_{rem} <$ threshold **then**
        $j = j + 1$
        $v_{lim} \leftarrow \bar{\mathbf{v}}_{seg}^{[j]}$
    end if
    compute $\bar{\mathbf{v}}_p$, $\bar{\mathbf{s}}_p$, and $\bar{\mathbf{a}}_p$ using (12.7), (12.8), and (12.9), respectively
end for

12.3.2 Spatial-Based Driver Model

The distance can be thought as an alternative independent variable to time. This makes it appealing to use spatial longitudinal coordinates instead of time variables in order to reduce the problem size of optimization strategies for long-time predictions. Therefore, it is essential to obtain the driver model from the previous section in spatial coordinate. The equations can be transferred to the spatial domain by dividing the right-hand side of the differential equation by $v(\cdot)$. This becomes clear, when considering the following transformation:

$$\frac{\mathrm{d}v}{\mathrm{d}s} = \frac{\mathrm{d}v}{\mathrm{d}t} \cdot \frac{\mathrm{d}t}{\mathrm{d}s} = \frac{\mathrm{d}v}{\mathrm{d}t} \cdot \frac{1}{v(t)}. \tag{12.10}$$

The reference trajectory is calculated over the spatial grid

$$0 = s_0 < s_1 < \cdots < s_i < \cdots < s_{N_s} = s_{dist}, \quad \mathcal{G}_s = \left\{ s_0, s_1, \ldots, s_{N_s} \right\}. \tag{12.11}$$

The grid can be chosen to be equidistant over $N_s + 1$ positions

$$\Delta s = s_i - s_{i-1} = \text{const}$$

or none equidistant

$$\Delta s_i = s_i - s_{i-1}, \quad i = 1, \ldots, N_s$$

where s_i is the cumulated distance up to i-th spatial discretization point and s_{dist} is the total length of the driving cycle. For notational clarity, the enumeration in the spatial domain is distinguished from the time domain by writing the index i instead of index k. The number of $N_s + 1$ discretizations depends certainly on the topology of the trip and the optimization algorithm used.

As soon as the real distance exceeds the covered distance on the spatial grid (12.11) or the route-planning is recomputed, the reconstructor triggers the electronic horizon for an update of the spatial vectors (12.1)–(12.3).

Let us define the vectors

$$\begin{aligned}
\overline{\mathbf{v}}_p &= \left[v_{p,0}, v_{p,1}, \ldots, v_{p,N_s-1}\right]^T \\
\overline{\mathbf{t}} &= \left[t_0, t_1, \ldots, t_{N_s-1}\right]^T \\
\overline{\mathbf{a}}_p &= \left[a_{p,0}, a_{p,1}, \ldots, a_{p,N_s-1}\right]^T \\
\overline{\boldsymbol{\alpha}}_p &= \left[\alpha_{p,0}, \alpha_{p,1}, \ldots, \alpha_{p,N_s-1}\right]^T .
\end{aligned}$$

Depending on the current driving situation d, the velocity's spatial derivative is given by

$$\frac{\mathrm{d}\overline{\mathbf{v}}_p^{[i]}}{\mathrm{d}s} = \begin{cases} \dfrac{e}{\overline{\mathbf{v}}_p^{[i]}} \cdot \left[1 - \left(\dfrac{\overline{\mathbf{v}}_p^{[i]}}{v_{lim}}\right)^{\delta}\right], & d = 1 \\ -\dfrac{e}{\overline{\mathbf{v}}_p^{[i]}} \cdot \left[1 - \left(\dfrac{v_{lim}}{\overline{\mathbf{v}}_p^{[i]}}\right)^{\delta}\right], & d = 2 \\ \dfrac{1}{f \cdot \overline{\mathbf{v}}_p^{[i]}} \cdot \left(\dfrac{\left(\overline{\mathbf{v}}_p^{[i]}\right)^2}{2 s_{rem}}\right)^2, & d = 3. \end{cases} \tag{12.12}$$

The constants e and f have the same meaning as in Sect. 12.3.1. As can be noticed from measurements, the velocity often exhibits oscillations around the speed limit due to inharmonic traffic flow. To account for these oscillations, the sum of l-cosines with different amplitudes A_r, frequencies ω_r, and phase shifts ϕ_r is added to the current speed limit with

$$\begin{aligned}
v_{lim} &= \overline{\mathbf{v}}_{seg}^{[j]} + v_d \\
&= \overline{\mathbf{v}}_{seg}^{[j]} + \sum_{r=1}^{l} A_r \left(\overline{\mathbf{v}}_{seg}^{[j]}\right) \cdot \cos(\omega_r \cdot \overline{\mathbf{t}}_{[i]} + \phi_r).
\end{aligned}$$

The amplitudes, frequencies, and phase shifts can be identified from measurements via Fourier analysis. With Eq. (12.12), an *initial value problem* (IVP) can be solved using an explicit Euler approach as follows:

$$\overline{\mathbf{v}}_p^{[i+1]} = \max\left(\epsilon, \overline{\mathbf{v}}_p^{[i]} + \Delta s \cdot \frac{\mathrm{d}\overline{\mathbf{v}}_p^{[i]}}{\mathrm{d}s}\right). \tag{12.13}$$

The constant ϵ is a lower bound for the speed. This is necessary, since (12.12) is not defined for $\overline{\mathbf{v}}_p^{[i]} = 0$. Knowing the predicted vehicle speed $\overline{\mathbf{v}}_p^{[i]}$ over the spatial domain, the corresponding values for the time and acceleration can be approximated by

$$\bar{\mathbf{t}}_{[i+1]} = \bar{\mathbf{t}}_{[i]} + \frac{\Delta s}{\overline{\mathbf{v}}_p^{[i]}} \tag{12.14}$$

$$\overline{\mathbf{a}}_p^{[i+1]} = \frac{\overline{\mathbf{v}}_p^{[i+1]} - \overline{\mathbf{v}}_p^{[i]}}{\bar{\mathbf{t}}_{[i+1]} - \bar{\mathbf{t}}_{[i]}}. \tag{12.15}$$

The depicted Algorithm 12.2 sketches the calculation procedure for the spatial-based driver model.

Algorithm 12.2 Sketch of the Spatial-based Driver Model

```
initialization: fill the static allocated vectors s̄_seg, v̄_seg, and ᾱ_seg
set the segment counter j ← 0
for i := 1 to N_s do
    ᾱ_p^[i] ← ᾱ_seg^[j]
    s_rem = s̄_seg^[j] − s̄_[i]
    if s_rem < threshold then
        j = j + 1
        v_lim ← v̄_seg^[j] + v_d
    end if
    compute v̄_p^[i], t̄_[i], ā_p^[i], and ᾱ_p^[i] using (12.13), (12.14), and (12.15), respectively
end for
```

Figure 12.5 shows the validation of the driver model where only information on speed limits $\overline{\mathbf{v}}_{seg}$, road slopes $\overline{\boldsymbol{\alpha}}_{seg}$, and holding situations were fed to the driver model. Not all traffic events, especially dense traffic situations, can be foreseen and hence deviations from the predicted vehicle speed trajectory occur. However, due to the satisfactory shape capturing of the true vehicle speed trajectory and the desired robust energy management design, this has no significant performance impact.

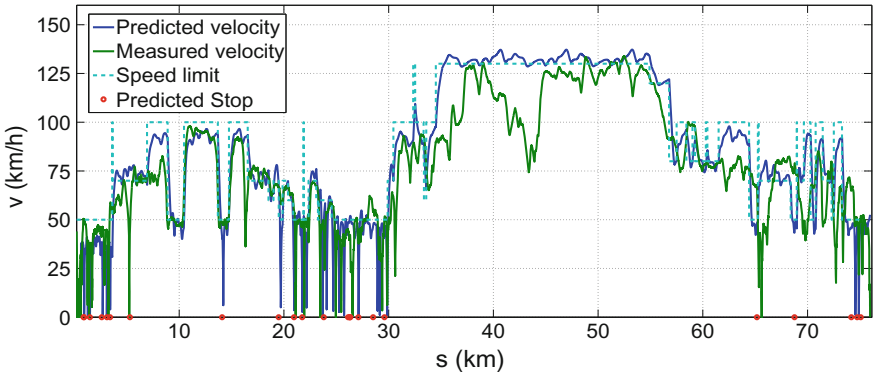


Fig. 12.5 Predicted and measured vehicle speed trajectories of the real-world benchmark-cycle 1

12.3.3 Estimation of Stop Events

It is assumed that stop events, which cannot be determined from our implementation of the GIS database, are classified as stochastic. This classification holds for the following traffic signs: right before left, grant right of way, traffic lights, roundabouts, pedestrian crossings, etc. They are referred to as stochastic stop classes. A number of independent experiments have shown that there is a correlation between the number of stochastic stop events, the quasi steady velocity, and the day-time. The Fig. 12.6 shows the expected values for some stochastic stop classes depending on the quasi steady velocity and Fig. 12.7 shows the expected values depending on the day-time. It is obvious that the number of stops increases at the rush hours.

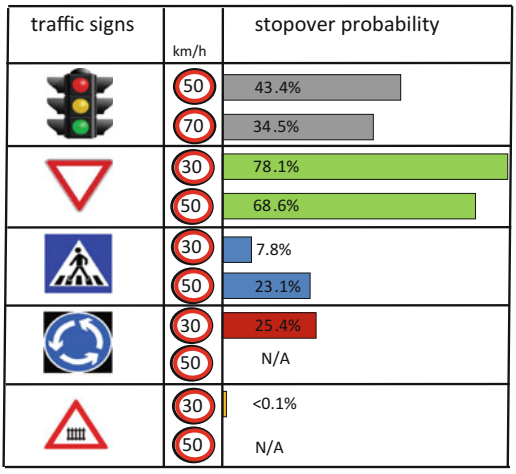


Fig. 12.6 Traffic signs with stop probabilities of the real-world benchmark-cycle 1

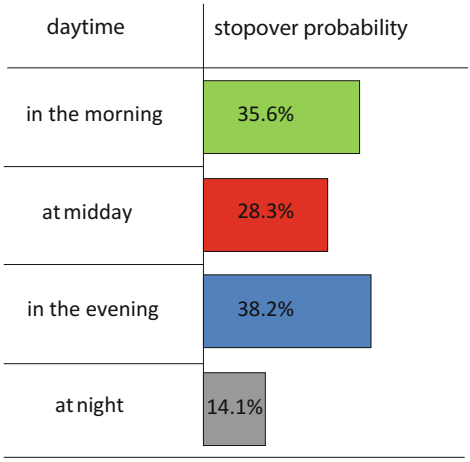


Fig. 12.7 Stop probabilities depending on day-time of the real-world benchmark-cycle 1

The probability distribution of each stop class has been estimated by a maximum likelihood estimator which reveals in all cases a normal distribution. To avoid multiple counting of the same stop event, due to inharmonic traffic densities, each stop event must be observed with an individual observation radius around the traffic sign.

The confidence in the stop estimations is related to the requirement to learn the probability distributions for unknown trips. It is therefore necessary to adapt the distributions by an online calculation of the maximum likelihood estimator.

12.4 Predictive Energy Management for Battery Electric Vehicles

The consideration of energy management in today's BEVs is constrained to auxiliary consumer management. This is mainly due to the lack of available DOFs, which the system provides. However, by introducing an "artificial" vehicle speed limit the energy consumption can be influenced by advising the driver. A *trip management* (TM) as a supervisory control for the BEV's energy management can provide recommended maximal vehicle speeds to the driver under the condition to reach the trip destination safely with the current battery status and the highest possible velocity, i.e., yielding the best time/energy consumption trade-off, if this is possible at all. Realization of such an algorithm can be classified as driver assistance.

A real-time implementable *dynamic programming* (DP) algorithm on spatial domain is used (originally proposed by Gong et al. [16] and further developed in [3]) using ITS information. Different random events which affect the $\bar{\boldsymbol{\xi}}$ trajectory, e.g., traffic lights, etc. are considered as external disturbances. The expected values for some disturbances are used to relate ξ-values as pre-control, which are released at

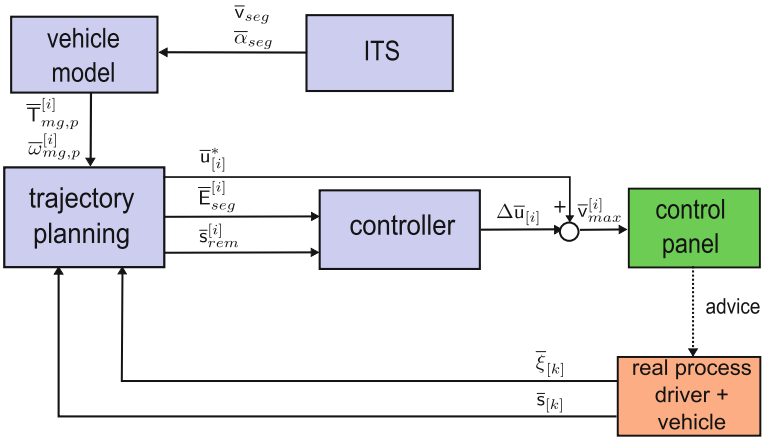


Fig. 12.8 Control structure of the predictive trip management

certain trip segments to improve the predicted $\overline{\boldsymbol{\xi}}_{ref}$ trajectory over the whole driving trip.

The predictive trip management is implemented as a 2-DOF control structure consisting of a trajectory planning unit and a feedback control law as shown in Fig. 12.8.

The control strategy consists of the following parts:

- **ITS**: determine the trip segmentation for the changing speed limits $\overline{\mathbf{v}}_{seg}$ and road slopes $\overline{\boldsymbol{\alpha}}_{seg}$;
- **vehicle-model**: calculates the trajectories $\overline{\mathbf{T}}^{[i]}_{mg,p}$, $\overline{\boldsymbol{\omega}}^{[i]}_{mg,p}$, and $\overline{\mathbf{P}}^{[i]}_{bat,p}$;
- **trajectory planning**: calculates a reference trajectory for $\overline{\boldsymbol{\xi}}_{ref}$ and an optimal recommended maximal speed limit $\overline{\mathbf{u}}^*$ using DP; and
- **instantaneous speed corrections**: a feedback control law calculates a correction $\Delta\overline{\mathbf{u}}_{[i]}$.

Based on the ITS information, the trajectory planning unit and the feedback control law determine together the recommended maximal vehicle speed trajectory $\overline{\mathbf{v}}^{[i]}_{max} = \overline{\mathbf{u}}^*_{[i]} + \Delta\overline{\mathbf{u}}_{[i]}$. The trajectory planning unit consists of a DP paradigm that calculates for each segment of the remaining trip an optimal maximal vehicle speed trajectory $\overline{\mathbf{u}}^*$ and the corresponding predicted state of charge trajectory $\overline{\boldsymbol{\xi}}^{[i]}_{ref}$. To account for random stop events the trajectory planning unit also estimates the required energy for these events on the trip, which is used to modify the initial state of charge before the DP calculation starts. The feedback controller is designed to compensate process disturbances (e.g., changing vehicle mass, energy consumption of the air conditioning, etc.) and modeling mismatch, which takes place within the rough DP griding.

12.4.1 Vehicle Model

The predictive trip management is implemented on a BEV-prototype that belongs to the subcompact vehicle class and employs a high-speed electric machine and high-energy traction battery. The high-voltage battery used is based on Lithium-Ion technology and has a capacity of 77 Ah. Some of the vehicle specifications are listed in Table 12.1.

To reduce the dimensionality for the real-time DP, the state of charge (10.51) must be converted from the time domain into the spatial domain. Using the spatial transformation rule (12.10), the state equation is transformed from time domain to spatial domain by simply dividing (10.51) by the recommended speed limit $\overline{\mathbf{u}}_{[i]}$ which yields

$$\overline{\boldsymbol{\xi}}_{ref}^{[i+1]} = \overline{\boldsymbol{\xi}}_{ref}^{[i]} + \frac{\Delta s}{Q_{bat}} \cdot \frac{\overline{\mathbf{I}}_{bat}^{[i]}}{\overline{\mathbf{u}}_{[i]}}$$

in discrete form. The recommended speed limit is also used to estimate the requested battery power

$$\overline{\mathbf{P}}_{bat,p}^{[i]} = \frac{\overline{\mathbf{F}}_{w,p}^{[i]} \cdot \overline{\mathbf{u}}_{[i]}}{\eta_{mg}\left(\overline{\mathbf{T}}_{mg,p}^{[i]}, \overline{\boldsymbol{\omega}}_{mg,p}^{[i]}\right)} + P_{aux}$$

where $\eta_{mg}(\cdot)$ is the MG efficiency and P_{aux} is an electrical auxiliary power consumption. The latter one is assumed to be constant for simplicity.

The predicted angular speed $\overline{\boldsymbol{\omega}}_{mg,p}$ and the predicted torque $\overline{\mathbf{T}}_{mg,p}$ can be calculated by a simplified dynamics model with a single gear ratio as

$$\overline{\boldsymbol{\omega}}_{mg,p}^{[i]} = i_{gbx} \cdot \frac{\overline{\mathbf{u}}_{[i]}}{r_{wh}}$$
$$\overline{\mathbf{T}}_{mg,p}^{[i]} = \frac{r_{wh}}{i_{gbx}} \cdot \overline{\mathbf{F}}_{w,p}^{[i]}$$

Table 12.1 Vehicle parameters of the BEV-prototype (subcompact class)

Symbol	Value	Unit	Description
m	1085	kg	Vehicle mass
Q_{bat}	77	Ah	Battery capacity
V_{oc}	330	V	Open-circuit voltage
i_{gbx}	9.8	–	Fixed gear ratio
$A_{sec} \cdot c_w$	0.66	m^2	Drag area
P_{mg}^{max}	60	kW	Motor's maximum power

where $\overline{\mathbf{F}}_{w,p}^{[i]}$ is the predicted total friction force

$$\begin{aligned}\overline{\mathbf{F}}_{w,p}^{[i]} &= \overline{\mathbf{F}}_{drag,p}^{[i]} + \overline{\mathbf{F}}_{roll,p}^{[i]} + mg\sin\overline{\boldsymbol{\alpha}}_{seg}^{[i]} \\ &\approx a_2 \cdot \overline{\mathbf{u}}_{[i]}^2 + a_1 \cdot \overline{\mathbf{u}}_{[i]} + a_0 + mg\sin\overline{\boldsymbol{\alpha}}_{seg}^{[i]}.\end{aligned}$$

The quadratic polynomial in the last equation is an approximation of the air-drag force and rolling resistance force as described in Sect. 10.2.

The spatial domain procedure drastically reduces the number of grid points as compared with the time domain procedure. A further positive side effect of this transformation is that the final distance s_f is fixed, whereas in the time domain the same task would result in a variable final time t_f. In the latter case, the application of DP is difficult.

12.4.2 Dynamic Programming for the Maximal Speed Limit

An essential element of solving the predictive trip management is to formulate the energy limitation procedure as an OCP. Using the spatial grid (12.11), the OCP in discrete form can be formulated as

$$\mathcal{P}_6 := \begin{cases} \min\limits_{\overline{\mathbf{u}}_{[i]} \in \hat{\mathcal{U}}_i} & m\left(\overline{\boldsymbol{\xi}}_{ref}^{[N_s]}\right) + \sum\limits_{i=1}^{N_s} l\left(\overline{\mathbf{u}}_{[i]}\right) \\ \overline{\boldsymbol{\xi}}_{ref}^{[i+1]} & = \overline{\boldsymbol{\xi}}_{ref}^{[i]} + \dfrac{\Delta s}{Q_{bat}} \cdot \dfrac{\overline{\mathbf{I}}_{bat}^{[i]}}{\overline{\mathbf{u}}_{[i]}} \\ \overline{\boldsymbol{\xi}}_{ref}^{[i]} & \in \hat{\mathcal{X}}_i \\ \overline{\boldsymbol{\xi}}_{ref}^{[0]} & = \xi_0 \\ \overline{\boldsymbol{\xi}}_{ref}^{[N_s]} & = \xi_f \end{cases} \tag{12.16}$$

where ξ_0 and ξ_f are the boundary values at the beginning and the end of the trip, respectively. The objective of $\mathcal{P}_6$ is to find the minimal energy consumption, which implies a "virtual" limitation of the electricity consumption accumulated over the trip distance up the target destination s_{N_s}. The "virtual" limitation is realized by recommending the driver an optimal maximal vehicle speed $\overline{\mathbf{u}}^*$. It is then a natural choice to define the instantaneous cost function as

$$l\left(\overline{\mathbf{u}}_{[i]}\right) = \frac{1}{\overline{\mathbf{u}}_{[i]}} \cdot \overline{\mathbf{P}}_{bat,p}^{[i]} \cdot \Delta s,$$

where $\Delta s/\overline{\mathbf{u}}_{[i]}$ is the time span for the i-th distance segment. The admissible sets in the problem formulation are spanned by simple box constraints, which yields for the admissible state set

$$\hat{\mathcal{X}}_i = \left\{ \boldsymbol{\xi}_{ref}^{[i]} \in \mathsf{X} \;\middle|\; \xi^{min} \leq \overline{\boldsymbol{\xi}}_{ref}^{[i]} \leq \xi^{max} \right\} \tag{12.17}$$

and for the admissible control set

$$\hat{\mathcal{U}}_i = \left\{ \mathbf{u}_{[i]} \in \mathsf{U} \;\middle|\; v_{tr,i}^{min} \leq \overline{\mathbf{u}}_{[i]} \leq v_{tr,i}^{max} \right\} \tag{12.18}$$

where ξ^{min} and ξ^{max} are box constraints of the continuous state and $v_{tr,i}^{min}$ and $v_{tr,i}^{max}$ are speed limits obeyed by the ITS for each i-th distance segment.

In order to apply the dynamic programming Algorithm 6.1 in spatial domain [3, 16], the final state boundary condition in $\mathcal{P}_6$ must be realized as a soft constraint and imposed on the cost function to penalize deviations from the desired target value ξ_f. The penalty term is implemented as the end-point function $m(\cdot)$ in $\mathcal{P}_6$ as

$$m\left(\overline{\boldsymbol{\xi}}_{ref}^{[N_s]}\right) = K_f \cdot \left(\overline{\boldsymbol{\xi}}_{ref}^{[N_s]} - \xi_f\right)^2, \tag{12.19}$$

where K_f is a weighting factor. Hence, the stored energy in the battery can be totally consumed up to a reserve. The admissible state set (12.17) and the admissible control set (12.18) are incorporated into the algorithm as an additional penalty function, which is defined by

$$P_i\left(\overline{\boldsymbol{\xi}}_{ref}^{[i+1]}, \overline{\mathbf{u}}_{[i]}\right) = K_{p1} \cdot \max\left(\overline{\mathbf{u}}_{[i]} - v_{tr}^{max}, 0\right) + K_{p2} \cdot \max\left(v_{tr}^{min} - \overline{\mathbf{u}}_{[i]}, 0\right)$$

where K_{p1} and K_{p2} are weighting factors. The recommended maximal vehicle speed $\overline{\mathbf{u}}_{[i]}$ depends on the power consumption P_{aux} of the electric auxiliary devices too. The higher the electric loads the higher the recommended maximal vehicle speed should be.

The DP algorithm proceeds backwards, beginning from the trip destination to the trip start. Hence, the initial condition is imposed from (12.19) at N_s by

$$V\left(\overline{\boldsymbol{\xi}}_{ref}^{[N_s]}\right) = m\left(\overline{\boldsymbol{\xi}}_{ref}^{[N_s]}\right),$$

where $V(\cdot)$ is the cost-to-go function. The next step backwards is the cost-to-go function evaluated at $(s_{N_s-1}, \overline{\boldsymbol{\xi}}_{ref}^{[N_s-1]})$ in the spatial state-space as

$$V\left(\overline{\boldsymbol{\xi}}_{ref}^{[N_s-1]}, s_{N_s-1}\right) = \min_{\overline{\mathbf{u}}_{[N_s-1]} \in \hat{\mathcal{U}}_{N_s-1}} C\left(\overline{\boldsymbol{\xi}}_{ref}^{[N_s-1]}, \overline{\mathbf{u}}_{[N_s-1]}\right)$$

where

$$C\left(\overline{\boldsymbol{\xi}}_{ref}^{[N_s-1]}, \overline{\mathbf{u}}_{[N_s-1]}\right) = l(\overline{\mathbf{u}}_{[N_s-1]}) + V\left[g\left(\overline{\boldsymbol{\xi}}_{ref}^{[N_s-1]}, \overline{\mathbf{u}}_{[N_s-1]}\right), s_{N_s}\right] + P_i\left[g\left(\overline{\boldsymbol{\xi}}_{ref}^{[N_s-1]}, \overline{\mathbf{u}}_{[N_s-1]}\right), \overline{\mathbf{u}}_{[N_s-1]}\right],$$

where $g(\cdot)$ calculates the consecutive state $\overline{\boldsymbol{\xi}}_{ref}^{[i+1]}$. This procedure is repeated up to the trip start. Thus, minimizing the cost-to-go over the set of all feasible control sequences $\overline{\mathbf{u}} = \{u_1, \ldots, u_{N_s}\}$ gives the optimal control solution.

12.4.3 Instantaneous Speed Limit Corrections

The feedback controller for obtaining $\Delta\overline{\mathbf{u}}_{[i]}$ can be designed by calculating the ratio of "remaining energy for the current segment $\overline{\mathbf{E}}_{seg}$ to be released to the powertrain" to the "remaining distance $\overline{\mathbf{s}}_{rem}$ of the current segment." The ratio is evaluated at each spatial grid point i, which yields

$$F_{seg,p} = \frac{\overline{\mathbf{E}}_{seg}^{[i]} \cdot \eta_{mg}\left(\overline{\mathbf{T}}_{mg,p}^{[i]}, \overline{\boldsymbol{\omega}}_{mg,p}^{[i]}\right) \cdot \eta_{bat}\left(\overline{\mathbf{P}}_{bat,p}^{[i]}\right)}{\overline{\mathbf{s}}_{rem}^{[i]}} \tag{12.20}$$

and includes the powertrain efficiencies. Analogously, a ratio for remaining energy to remaining distance must also be calculated for the measured vehicle signals and is denoted by F_{seg}. From a physical viewpoint, both ratios represent forces and the difference

$$\Delta F = F_{seg,p} - F_{seg}$$

indicates the force, which can additionally recommended to be applied to the vehicle. Thus, the controller output $\Delta\overline{\mathbf{u}}_{[i]}$ can be calculated by evaluating the force balance

$$F_{seg,p} + \Delta F = a_2 v_c^2 + a_1 v_c + a_0 + mg \sin\overline{\boldsymbol{\alpha}}_{seg}^{[i]}$$

where a_0, a_1, and a_2 are the vehicle's drag resistance factors and g is the acceleration due to gravity. Solving for the corrected vehicle speed v_c yields

$$\Delta\overline{\mathbf{u}}_{[i]} = \frac{-a_1 + \sqrt{4a_2 \cdot \left(F_{seg,p} + \Delta F - mg \sin\overline{\boldsymbol{\alpha}}_{seg}^{[i]} - a_0\right) + a_1^2}}{2a_2} - \overline{\mathbf{u}}_{[i]}^{*}.$$

The controller output $\Delta\overline{\mathbf{u}}_{[i]}$ can be optionally filtered using floor $\lfloor\cdot\rfloor$ and ceil $\lceil\cdot\rceil$ functions in order to reduce small incremental speed updates at the driver's display panel.

The deviations from the measured $\bar{\boldsymbol{\xi}}$-trajectory and planned $\bar{\boldsymbol{\xi}}_{ref}$-trajectory are evaluated in the trajectory planning unit, which executes a reset in case the deviations are beyond the tolerance of $\pm 5\%$ of the battery's state of charge.

12.4.4 Experimental Results

The predictive trip management is implemented in a BEV-prototype vehicle using a real-time prototyping system on PC level. The real-time workshop of Mathworks© has been used to generate real-time executable code for a small-scale computer with 2.8 GHz. The third benchmark-cycle (see Fig. 12.3) has been used for real-time validation and is discretized by 49 segments, which spans the discretization grid for the DP. The final state of charge should be at the trip destination around 15%. Thus, we set $\xi_f = 0.15$. For the benchmark-cycle, a reserved energy contingent of 1.8% was assumed for random stopovers, which will be specifically released as disturbance compensation at the moment when a certain stop velocity pattern is detected.

Figures 12.9, 12.10, and 12.11 show the experimental results. The gray shaded area in Fig. 12.9 is spanned by the speed limit and the minimal speed by traffic rules. This area is the admissible speed set within which the driver can freely maneuver. The gray shaded area in Fig. 12.10 is spanned by the recommended maximal vehicle speed and the minimal speed by traffic rules. This area is restricted by the predictive trip management such that the driver can safely reach the trip destination. If the driver exceeds the admissible velocity set for more than 60 s, a warning will be generated.

Figure 12.11 shows the $\bar{\boldsymbol{\xi}}$-trajectories for the planned and the measured state of charge. During the test drive, two re-optimizations marked with red dots occurred mainly because of the unexpected traffic flow, e.g., slow cars in front, stop-and-go, and so on. The re-optimizations are performed with modified griding in an acceptable time, since the trajectory planning unit always cover only the remaining trip distance.

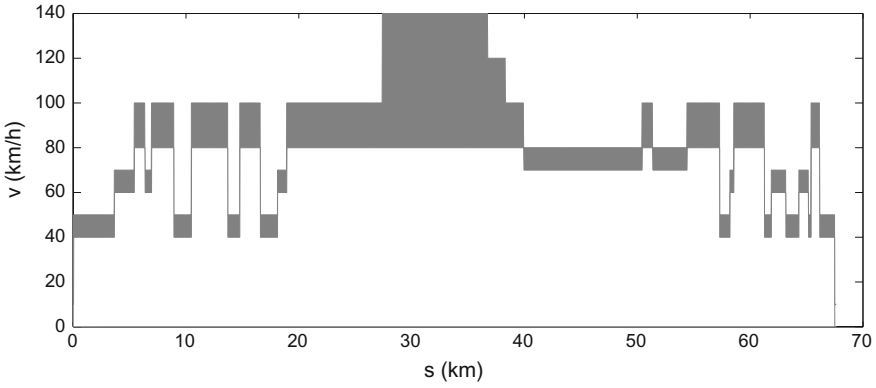


Fig. 12.9 The gray-filled area is the admissible vehicle speed $\hat{\mathcal{U}}_i$ determined by the traffic rules

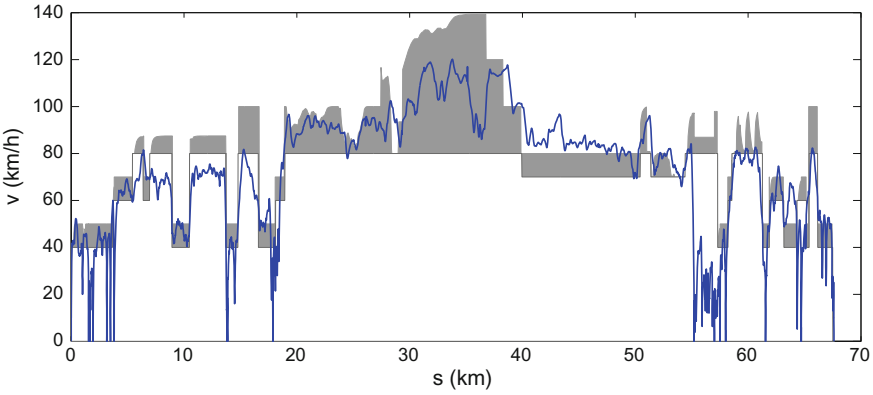


Fig. 12.10 The gray-filled area is the admissible vehicle speed $\hat{\mathcal{U}}_i$ determined by the TM. The blue trajectory is the measured vehicle speed by a test drive

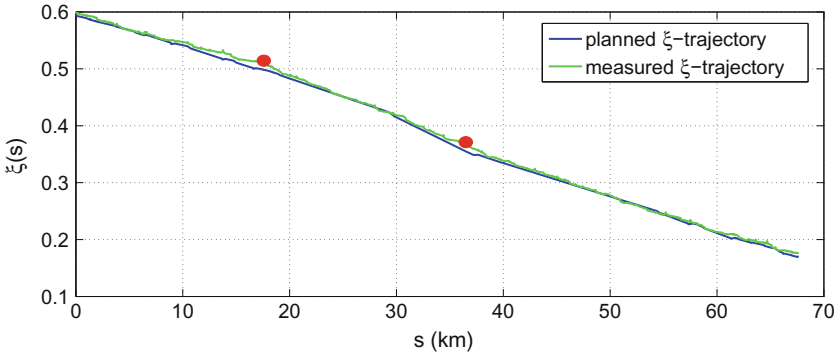


Fig. 12.11 Planned and measured $\overline{\boldsymbol{\xi}}$ trajectories from the benchmark trip. The red-filled *dots* indicate triggers points for re-optimization

It is also not usual that stressed drivers disregard the recommended speed advice. The predictive trip management must, therefore, be robust to cope with that.

12.5 Predictive Energy Management for Hybrid Vehicles

This paragraph discusses predictive control designs for energy management problems in hybrid vehicles, which is based on solving an OCP on-board. This technique has been adopted in many ways. One natural way is trying to solve the OCP directly on-board applying the indirect shooting approach. An indirect shooting approach is probably one of the best choices for implementation on ECUs due to the less computational demand compared with DP and direct approaches.

We begin with the formulation of the OCP for a simplified $\mathcal{P}_1$ to obtain fuel-optimal driving. Then, going quickly to the steps of solving the problem to catch-up the idea how to implement such a strategy on-board.

The task is to find the continuous-valued control $\overline{\mathbf{u}}^*_{[k]}$ that minimizes the fuel consumption for charge-sustaining operation

$$\min_{\overline{\mathbf{u}}_{[k]}} \overline{\boldsymbol{\beta}}_{[N_t]} \tag{12.21}$$

over the time grid (5.9) subject to system equations

$$\overline{\boldsymbol{\beta}}_{[k+1]} = \overline{\boldsymbol{\beta}}_{[k]} + h \cdot \gamma_f \cdot \text{bsfc}\left(\overline{\mathbf{u}}_{[k]}, \overline{\boldsymbol{\omega}}^{[k]}_{ice}\right) \cdot \overline{\mathbf{u}}_{[k]} \cdot \overline{\boldsymbol{\omega}}^{[k]}_{ice} \tag{12.22}$$

$$\overline{\boldsymbol{\xi}}_{[k+1]} = \overline{\boldsymbol{\xi}}_{[k]} + \frac{h}{Q_{bat}} \cdot \overline{\mathbf{I}}^{[k]}_{bat}, \tag{12.23}$$

the boundaries

$$\overline{\boldsymbol{\beta}}_{[0]} = 0 \tag{12.24}$$

$$\overline{\boldsymbol{\xi}}_{[0]} = \xi_0 \tag{12.25}$$

$$\overline{\boldsymbol{\xi}}_{[N_t]} = \xi_f, \tag{12.26}$$

and the control restraints

$$\mathbf{c}_{\overline{u}}(\overline{\mathbf{u}}_{[k]}) := \begin{bmatrix} \overline{\mathbf{u}}_{[k]} - T^{max}_{ice}\left(\overline{\boldsymbol{\omega}}^{[k]}_{ice}\right) \\ T^{min}_{ice} - \overline{\mathbf{u}}_{[k]} \\ \overline{\mathbf{T}}^{[k]}_{gbx,p} - \overline{\mathbf{u}}_{[k]} - T^{max}_{mg}\left(\overline{\boldsymbol{\omega}}^{[k]}_{mg}\right) \\ T^{min}_{mg}\left(\overline{\boldsymbol{\omega}}^{[k]}_{mg}\right) + \overline{\mathbf{u}}_{[k]} - \overline{\mathbf{T}}^{[k]}_{gbx,p} \end{bmatrix} \tag{12.27}$$

where the continuous-valued control $\overline{\mathbf{u}}_{[k]}$ is the engine torque $\mathbf{T}^{[k]}_{ice}$, the boundary values $T^{max}_{mg}(\cdot)$, $T^{min}_{mg}(\cdot)$, and $T^{max}_{ice}(\cdot)$ for *motor/generator* (MG) and *internal combustion engine* (ICE), respectively, depend on the speeds. The speed-dependent engine drag torque is approximated by a constant torque $T^{min}_{ice}(\overline{\boldsymbol{\omega}}^{[k]}_{ice}) \equiv T^{min}_{ice}$ for simplicity. Equations (12.22) and (12.23) are explicit Euler discretizations of the ODEs (10.15)–(10.16) and (10.51)–(10.52), respectively.

The OCP (12.21)–(12.27) can then be solved using Algorithm 7.1, which needs the definition of the Hamiltonian in discrete form

$$\mathcal{H}(\overline{\mathbf{x}}_{[k]}, \overline{\mathbf{u}}_{[k]}, \overline{\boldsymbol{\lambda}}_{[k]}) = \gamma_f \cdot \text{bsfc}\left(\overline{\mathbf{u}}_{[k]}, \overline{\boldsymbol{\omega}}^{[k]}_{ice}\right) \cdot \overline{\mathbf{u}}_{[k]} \cdot \overline{\boldsymbol{\omega}}^{[k]}_{ice} + \overline{\boldsymbol{\lambda}}_{[k]} \cdot \frac{\overline{\mathbf{I}}^{[k]}_{bat}}{Q_{bat}}. \tag{12.28}$$

Observing from the Hamiltonian (12.28) that the fuel consumption has a constant costate of 1, let us define the extended state as

$$\overline{\mathbf{y}}_{[k]} = \begin{bmatrix} \overline{\boldsymbol{\xi}}_{[k]} \\ \overline{\boldsymbol{\lambda}}_{[k]} \end{bmatrix}.$$

The time derivatives of these states yield

$$\mathbf{G}(\overline{\mathbf{y}}_{[k]}, \overline{\mathbf{u}}_{[k]}) = \begin{bmatrix} \dfrac{\overline{\mathbf{I}}_{bat}^{[k]}}{Q_{bat}} \\ -\dfrac{\overline{\boldsymbol{\lambda}}_{[k]}}{Q_{bat}} \cdot \dfrac{\partial \overline{\mathbf{I}}_{bat}^{[k]}}{\partial \overline{\boldsymbol{\xi}}_{[k]}} \end{bmatrix}$$

where $\overline{\mathbf{I}}_{bat}^{[k]}$ depends on $\overline{\boldsymbol{\xi}}_{[k]}$. The trajectory $\overline{\mathbf{y}}_{[k+1]}$ can be obtained by a simple explicit Euler integration

$$\overline{\mathbf{y}}_{[k+1]} = \overline{\mathbf{y}}_{[k]} + h \cdot \mathbf{G}(\overline{\mathbf{y}}_{[k]}, \overline{\mathbf{u}}_{[k]}). \tag{12.29}$$

In order to solve (12.29), we must guess an initial costate value $\overline{\boldsymbol{\lambda}}_{[0]} = \hat{\lambda}$ and solve the Hamiltonian minimization problem at each time instant k

$$\overline{\mathbf{u}}_{[k]}^{*} = \arg \min_{\overline{\mathbf{u}}_{[k]} \in \hat{\mathcal{U}}_k} \mathcal{H}\left(\overline{\mathbf{y}}_{[k]}, \overline{\mathbf{u}}_{[k]}\right),$$

where the admissible control set is defined by

$$\hat{\mathcal{U}}_k := \left\{ \overline{\mathbf{u}}_{[k]} \in \mathsf{U} \mid \mathbf{c}_{\overline{u}}(\overline{\mathbf{u}}_{[k]}) \leq \mathbf{0} \right\}.$$

This procedure is repeated until the boundary condition is satisfied

$$\Upsilon\left(\overline{\mathbf{y}}_{[N_t]}\right) = \overline{\boldsymbol{\xi}}_{[N_t]} - \xi_f. \tag{12.30}$$

Implementing and solving the OCP online on the ECU has the advantage of having low memory requirements. But the algorithm from above has one Achilles heel. The most expensive operation in this algorithm is the minimization of the Hamiltonian to find the optimal continuous-valued control. This operation is very costly and would require a significant amount of resources on the ECU. This requirement prohibits an online application in vehicles and makes it desirable to simplify the operation. Fortunately, we can exchange this costly operation with less demanding LUT operation. In Sect. 11.3.3 we have seen how controls can be stored without significant loss of accuracy in LUTs and this technique can be directly applied to solve predictive energy management by calculating offline LUTs for a range of values of the costate.

In the succeeding sections, we demonstrate two control structures to embed the operation (12.30) and LUT interpolation to perform predictive energy management.

12.5.1 Event-Triggered Predictive Energy Management

We follow Musardo et al. [34] reasoning for the development of a control structure that aperiodically solves the OCP (12.21)–(12.27) over a short-time prediction horizon (receding horizon) of the driving profile to generate feedforward control. This open-loop control strategy is transformed to closed-loop by embedding the open-loop strategy into an event-based paradigm. We obtain then an event-triggered predictive energy management, which closes the control-loop only at certain time instances. The strategy is designed, implemented, and validated for a P2 HEV with a dry seven-speed-dual-clutch transmission.

The key technology of this section is to solve (12.30) aperiodically. An event-based control-loop has the property that the sampling is not invoked by an external clock but by the system behavior exceeding certain bounds due to changes in the environmental conditions. This control paradigm can reduce the usage of the feedback link within a control-loop to time instants at which an event indicates the need for an new information exchange in order to retain the desired closed-loop performance.

In this version of predictive energy management, we make the assumption that the drive mode is modeled by a simple rule-based logic which depends on vehicle speed and the requested wheel torque. This means, the switching structure is obtained by simulation of a prediction model over the prediction horizon such that the SOCP is simplified to an ordinary OCP (cf. Boehme et al. [6]).

The OCP can be formulated as Mayer problem as follows:

$$\mathcal{P}_7 := \begin{cases} & \min\limits_{\overline{\mathbf{u}}_{[k]}} \ \overline{\boldsymbol{\beta}}_{[N_t]} \\ \overline{\boldsymbol{\beta}}_{[k+1]} & = \overline{\boldsymbol{\beta}}_{[k]} + h \cdot \gamma_f \cdot \mathrm{bsfc}\left(\overline{\mathbf{u}}_{[k]}, \overline{\boldsymbol{\omega}}_{ice,p}^{[k]}\right) \cdot \overline{\mathbf{u}}_{[k]} \cdot \overline{\boldsymbol{\omega}}_{ice,p}^{[k]} \\ \overline{\boldsymbol{\xi}}_p^{[k+1]} & = \overline{\boldsymbol{\xi}}_p^{[k]} + \dfrac{h}{Q_{bat}} \cdot \overline{\mathbf{I}}_{bat}^{[k]} \\ \overline{\boldsymbol{\beta}}_{[0]} & = 0 \\ \overline{\boldsymbol{\xi}}_p^{[0]} & = \xi_0 \\ \overline{\boldsymbol{\xi}}_p^{[N_t]} & = \xi_f . \end{cases} \tag{12.31}$$

The task of solving $\mathcal{P}_7$ is to find the continuous-valued control $\overline{\mathbf{u}}_{[k]}^*$ that minimizes the fuel consumption with the predicted trajectories $\overline{\mathbf{T}}_{gbx,p}$, $\overline{\boldsymbol{\omega}}_{ice,p}$, and $\overline{\boldsymbol{\omega}}_{mg,p}$ over a prediction horizon of 10 min. The control restraints in $\mathcal{P}_7$ is defined as

$$\mathbf{c}_{\overline{u}}(\overline{\mathbf{u}}_{[k]}) := \begin{bmatrix} \overline{\mathbf{u}}_{[k]} - T_{ice}^{max}\left(\overline{\boldsymbol{\omega}}_{ice,p}^{[k]}\right) \\ T_{ice}^{min} - \overline{\mathbf{u}}_{[k]} \\ \overline{\mathbf{T}}_{gbx,p}^{[k]} - \overline{\mathbf{u}}_{[k]} - T_{mg}^{max}\left(\overline{\boldsymbol{\omega}}_{mg,p}^{[k]}\right) \\ T_{mg}^{min}\left(\overline{\boldsymbol{\omega}}_{mg,p}^{[k]}\right) + \overline{\mathbf{u}}_{[k]} - \overline{\mathbf{T}}_{gbx,p}^{[k]} \end{bmatrix}. \tag{12.32}$$

Table 12.2 Vehicle parameters of the P2 HEV-prototype (compact class)

Symbol	Value	Unit	Description
m	1520	kg	Vehicle mass
Q_{bat}	1.1	kWh	Battery capacity
$i_{gbx}(1)$	15.5	–	First gearbox ratio
$i_{gbx}(2)$	9.3	–	Second gearbox ratio
$i_{gbx}(3)$	6	–	Third gearbox ratio
$i_{gbx}(4)$	4.1	–	Fourth gearbox ratio
$i_{gbx}(5)$	3.1	–	Fifth gearbox ratio
$i_{gbx}(6)$	2.5	–	Sixth gearbox ratio
$i_{gbx}(7)$	2.1	–	Seventh gearbox ratio
T_{ice}^{max}	250	Nm	Engine's maximum torque
T_{mg}^{max}	160	Nm	Motor's maximum torque

The vehicle configuration data is shown in Table 12.2, whose gear-numbers are included in the set $K = \{1, 2, \ldots, 7\}$. The control strategy consists of the following parts:

- **ITS**: provides information on speed limit and road slope;
- **time-based driver-model**: calculates $\overline{\mathbf{v}}_p$ over a 10 min prediction time horizon;
- **vehicle-model**: calculates the trajectories $\overline{\mathbf{T}}_{gbx,p}$, $\overline{\boldsymbol{\omega}}_{ice,p}$, and $\overline{\boldsymbol{\omega}}_{mg,p}$;
- **open-loop control generator**: an online optimization that minimizes the OCP over the prediction horizon and generates the battery's state of charge trajectory $\overline{\boldsymbol{\xi}}_p$ and control trajectory $\overline{\mathbf{u}}$; and
- **event generator**: triggers a re-calculation of the open-loop control generator to ensure closed-loop performance

and is depicted in Fig. 12.12.

The speed limit and the predicted road slope $\overline{\boldsymbol{\alpha}}_{seg}$ are provided by the ITS and fed to the reconstructor, which is implemented on the dSPACE MicroAutoBox. The time-based driver model uses the reconstructor as an interface to obtain updated and preprocessed speed limits and road slopes to generate a predicted vehicle speed trajectory $\overline{\mathbf{v}}_p$ over a 10 min time horizon. A simple vehicle model calculates the required gearbox input torque $\overline{\mathbf{T}}_{gbx,p}$ trajectory and the angular speed $\overline{\boldsymbol{\omega}}_{ice,p}$, $\overline{\boldsymbol{\omega}}_{mg,p}$ trajectories, which are then passed on to the open-loop control generator that solves the OCP and determines a feasible value for λ. Before the problem can be solved, the predicted transitions $\overline{\boldsymbol{\zeta}}_p$ between the electrical drive mode and hybrid drive mode and the predicted gear changes $\overline{\boldsymbol{\kappa}}_p$ needs to be determined over the prediction horizon, which is simulated and depends on the predicted vehicle speed $\overline{\mathbf{v}}_p^{[k]}$ and predicted wheel torque $\overline{\mathbf{T}}_{wh,p}^{[k]}$ only. Together with the measured $\overline{\mathbf{v}}_{[k]}$, $\overline{\mathbf{T}}_{wh}^{[k]}$, and λ the continuous-valued control at any time are then interpolated from a LUT. The event generator repeats this process, as soon as the predicted $\overline{\boldsymbol{\xi}}_p^{[k]}$ on the real trip deviates from the

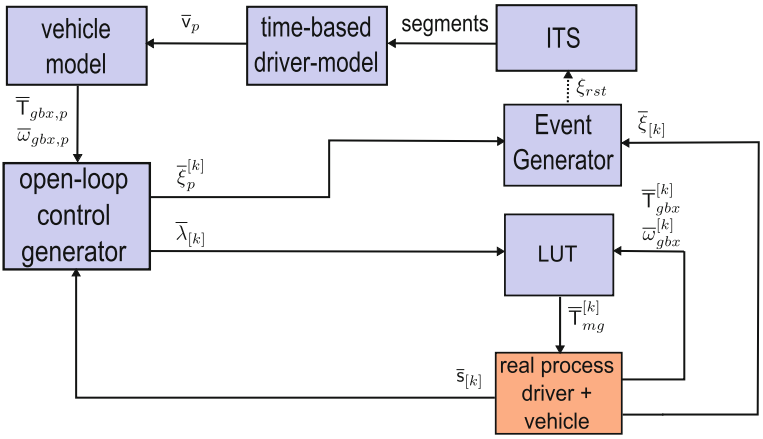


Fig. 12.12 Control structure of the event-triggered predictive energy management. The *solid lines* represent continuous signals, whereas the *dashed line* indicates that the signals are only transmitted at event time instants that are determined by the event generator

measured $\overline{\boldsymbol{\xi}}_{[k]}$ more than a given threshold or as soon as the predicted driving trip changes.

The continuous-valued control must be sent to the vehicle's on-board ECU in order to use the underlying software structure of the P2 parallel hybrid. Therefore, the communication between the ECU and the MicroAutoBox is established using a number of reprogrammed CAN signals. These CAN signals are in the ECU remapped to provide access to the torque structure. A torque structure is a software layer below the energy management and coodinates the degree-of-freedoms of the ICE and MG.

12.5.1.1 Vehicle Model

The predicted vehicle speed $\overline{\mathbf{v}}_p^{[k]}$, vehicle acceleration $\overline{\mathbf{a}}_p^{[k]}$, and road slope $\overline{\boldsymbol{\alpha}}_p^{[k]}$ trajectories are provided by the time-based driver model using the statically allocated vectors (12.4), (12.5), and (12.6).

The angular wheel speed $\overline{\boldsymbol{\omega}}_{wh,p}$ and the predicted wheel torque $\overline{\mathbf{T}}_{wh,p}$ can be calculated with a simplified longitudinal vehicle dynamics model as

$$\overline{\boldsymbol{\omega}}_{wh,p}^{[k]} = \frac{\overline{\mathbf{v}}_p^{[k]}}{r_{wh}}$$

$$\overline{\mathbf{T}}_{wh,p}^{[k]} = I_{veh} \cdot \frac{\overline{\mathbf{a}}_p^{[k]}}{r_{wh}} + r_{wh} \cdot \overline{\mathbf{F}}_{w,p}^{[k]}$$

where $\overline{\mathbf{F}}_{w,p}^{[k]}$ is the predicted total friction force

$$\overline{\mathbf{F}}_{w,p}^{[k]} = \overline{\mathbf{F}}_{drag,p}^{[k]} + \overline{\mathbf{F}}_{roll,p}^{[k]} + mg \sin \overline{\boldsymbol{\alpha}}_{p}^{[k]}.$$

The gearbox input torque $\overline{\mathbf{T}}_{gbx,p}^{[k]}$ and the angular speed $\overline{\boldsymbol{\omega}}_{gbx,p}^{[k]}$ depend on the gear ratio and are obtained as follows:

$$\begin{aligned}
\overline{\mathbf{T}}_{gbx,p}^{[k]} &= \frac{\overline{\mathbf{T}}_{wh,p}^{[k]}}{i_{gbx}\left(\overline{\boldsymbol{\kappa}}_{p}^{[k]}\right)} + T_{loss}\left(\overline{\boldsymbol{\kappa}}_{p}^{[k]}, \overline{\mathbf{T}}_{wh,p}^{[k]}, \overline{\boldsymbol{\omega}}_{wh,p}^{[k]}\right) \\
\overline{\boldsymbol{\omega}}_{gbx,p}^{[k]} &= i_{gbx}\left(\overline{\boldsymbol{\kappa}}_{p}^{[k]}\right) \cdot \overline{\boldsymbol{\omega}}_{wh,p}^{[k]},
\end{aligned}$$

where $\overline{\boldsymbol{\kappa}}_{p}^{[k]} \in K$ is the predicted active gear at time instant k.

Using $\overline{\boldsymbol{\zeta}}_{p}$, the speeds of the engine and motor/generator are

$$\begin{aligned}
\overline{\boldsymbol{\omega}}_{ice,p}^{[k]} &= \overline{\boldsymbol{\zeta}}_{p}^{[k]} \cdot \overline{\boldsymbol{\omega}}_{gbx,p}^{[k]} \\
\overline{\boldsymbol{\omega}}_{mg,p}^{[k]} &= \overline{\boldsymbol{\omega}}_{gbx,p}^{[k]}.
\end{aligned}$$

The predicted signals $\overline{\mathbf{T}}_{gbx,p}$, $\overline{\boldsymbol{\omega}}_{ice,p}$, and $\overline{\boldsymbol{\omega}}_{mg,p}$ are used as fixed trajectories for $\mathcal{P}_7$ over the time horizon of 10 min.

12.5.1.2 Solving the OCP Online

The trajectories from the vehicle model can now be used to generate a control trajectory $\overline{\mathbf{u}}$ in open-loop. This task is sometimes also referred to as *control generator*. Basically, in our case the problem $\mathcal{P}_7$ is reduced to a BVP and is solved by an indirect single shooting approach with the scalar equation

$$\overline{\boldsymbol{\xi}}_{p}^{[N_t]} - \xi_f = 0. \tag{12.33}$$

An optimal λ^* is then found by iteratively improving the initial guess of $\hat{\lambda}$ such that the boundary condition (12.33) is satisfied. This is done numerically by finding a sequence $\{\lambda_l\}_{l=1,2,\ldots}$ such that $|\Upsilon| < \epsilon$ approaches an lower error limit up to a desired exactness. This problem can be efficiently solved with the Pegasus method as shown in Fig. 12.13. In all cases a valid value for λ is determined within 5 iterations. In the average, the determination lasts 0.3 s on the dSPACE prototyping MicroAutoBox.

During the determination of an appropriate constant value of the costate λ using the Pegasus method, the minimization of the Hamiltonian function is replaced by a LUT interpolation, which reduces the computing time for solving (12.33) by more than 90%. The optimal continuous-valued control is then fed to the powertrain.

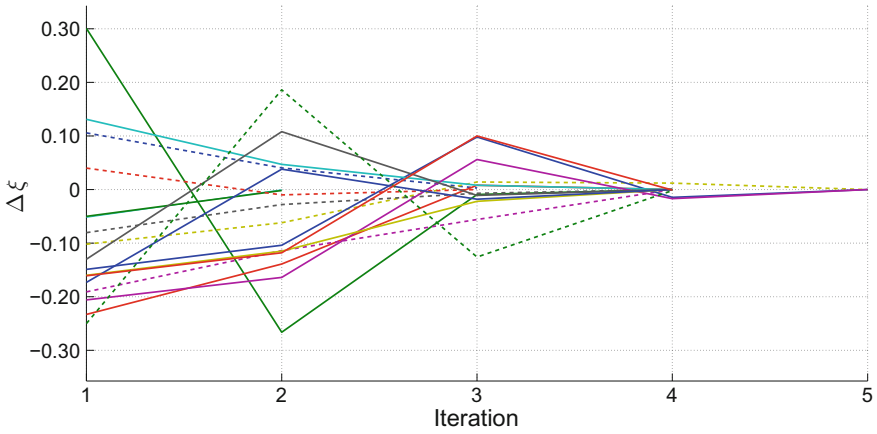


Fig. 12.13 Convergence of Pegasus method for solving the OCP (Boehme et al. [6])

12.5.1.3 Event Generator

The continuous-valued open-loop control $\overline{\mathbf{u}}_{[k]}$ is applied to the powertrain until a new event is triggered or the prediction horizon is exceeded. In both cases a re-calculation of the open-loop control generator is started and a new computation thread starts.

On the one hand, in order to achieve closed-loop stability the prediction time horizon T_p should be chosen not too large, but large enough to ensure that the OCP optimization operates efficiently. On the other hand, the T_p should not approach zero because of Zeno-behavior. The length of the prediction horizon is directly determined by the maximum number of prediction steps and indirectly by the choice of the reset threshold ξ_{rst}, which is more or less intuitive. These two constraints are the major lever for the controller design and should be chosen carefully. Also, one should keep in mind that the sequence of events is not known in advance and therefore not equally distributed.

The prediction time horizon T_p is defined as the time between two trigger-events and is limited to 10 min. For convenience, let us denote t_j as the time where an event is initiated. Then, the trigger logic for the event-triggered predictive controller can be simply implemented as

$$t_{j+1} = \begin{cases} t_j + k \cdot h, & \text{if } k = \arg\min_k \left(\left\| \overline{\boldsymbol{\xi}}_{[k]} - \overline{\boldsymbol{\xi}}_p^{[k]} \right\| > \xi_{rst} \right) \\ t_j + N_p \cdot h, & \text{if } \left\| \overline{\boldsymbol{\xi}}_{[k]} - \overline{\boldsymbol{\xi}}_p^{[k]} \right\| \leq \xi_{rst}, \ k = 0, 1, \ldots, N_p \\ t_j + k \cdot h, & \text{if } b_{ext} = 1 \end{cases}$$

where the bit b_{ext} is triggered if the route has been changed.

12.5.1.4 Experimental Results

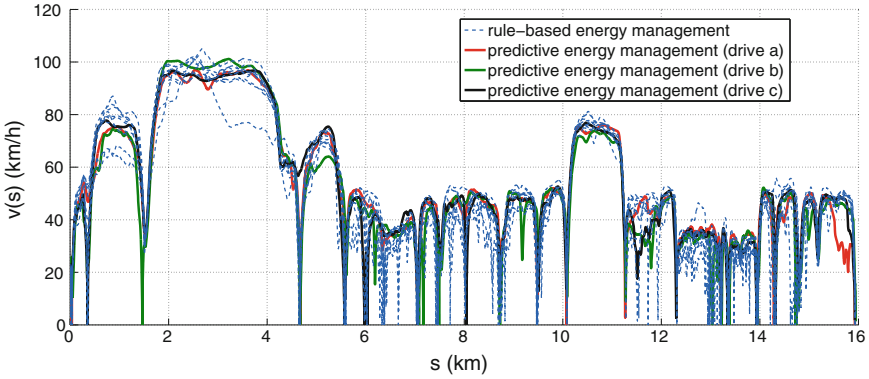
The event-triggered control strategy is implemented for a charge-sustaining HEV. For evaluation of the control strategy the real-world benchmark-cycles 2, 3, and 4 were used. The predicted velocity profile generated by the driver model does not reflect different driver types nor does it include information on traffic density. Instead, the parameters were determined to fit the measured data of an average driver. Consequently, it predicts the same vehicle speed trajectory each time. Therefore, the test-drives were conducted by different drivers and during different day-times to investigate the robustness of the control strategy.

The reset threshold for ξ_{rst} is chosen to be 0.05 and $N_p = 1000$ discretization values which corresponds to 10 min. As benchmark-strategy serves a rule-based energy management that uses optimized operation points of the ICE. On the given benchmark-cycles, fuel savings of up to 6% compared to the already very well calibrated rule-based strategy could be achieved, as shown in Table 12.3.

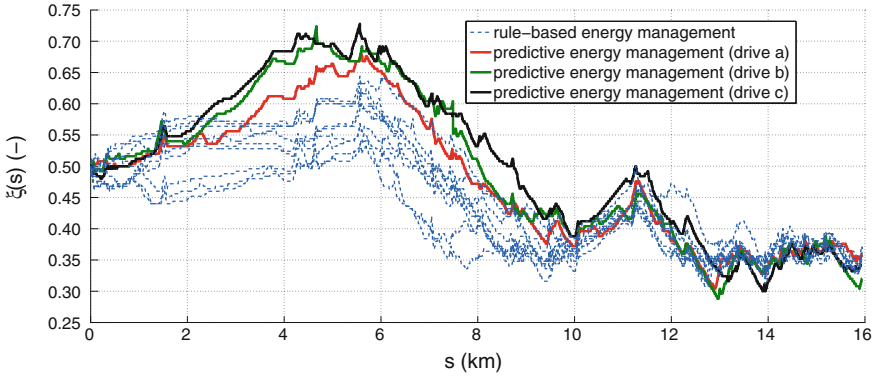
The predictive control strategy increases the state of charge $\overline{\boldsymbol{\xi}}$ for the real-world benchmark-cycle 2 before the urban route part is entered as shown in Fig. 12.14. This action increases the electrical drive capacity in urban surroundings and thus the increased electrical drive feeling, which can be a significant promotional message. The charging action is correlated by a high absolute value of the costate. The highest fuel savings can be observed for the real-world benchmark-cycle 4 in Fig. 12.16. The predictive control strategy decreases the state of charge more than rule-based energy management in order to enlarge the buffering capacity of the high-voltage battery before a long down-hill serpentine starts to overcome an altitude difference of nearly 400 m. The state of charge controlled by rule-based energy management decreases as well, since the control strategy cannot compensate the electrical energy consumption for short electrical drive modes during the up-hill driving. For the rural parts in the plane the predictive control strategy decreases $\overline{\boldsymbol{\xi}}$ notably as shown in Fig. 12.15. At the highway part in real-world benchmark-cycle 3 the MG torque is reduced, which can be observed from the low absolute value of the costate. The state of charge stays nearly constant at this driving condition, because the power is kept constant due to the higher speed of the MG.

Table 12.3 Comparison of fuel economy of event-triggered predictive energy management and rule-based energy management

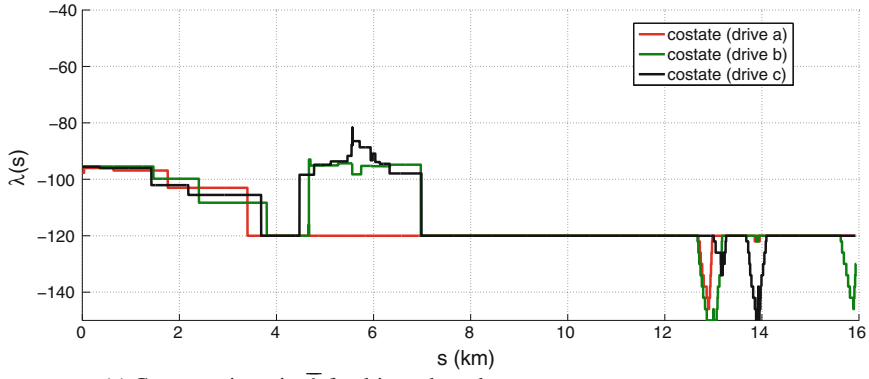
Benchmark-cycle	Mean fuel economy (predictive strategy) (l/100 km)	Mean fuel economy (rule-based strategy) (l/100 km)	Mean fuel savings (%)
Second	5.58	5.63	0.8
Third	5.58	5.68	1.8
Fourth	5.38	5.74	6.2



(a) The dashed blue curves are measured $\overline{\mathrm{v}}$-trajectories for drives with the rule-based energy management, the solid curves are measured $\overline{\mathrm{v}}$-trajectories for drives with the predictive energy management

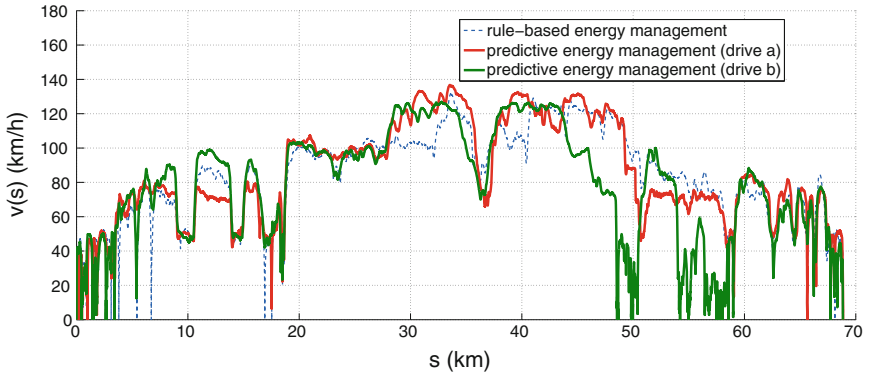


(b) The dashed blue curves are measured $\overline{\xi}$-trajectories for drives with the rule-based energy management. The solid curves are measured $\overline{\xi}$-trajectories for the predictive energy management for drive a, b, and c

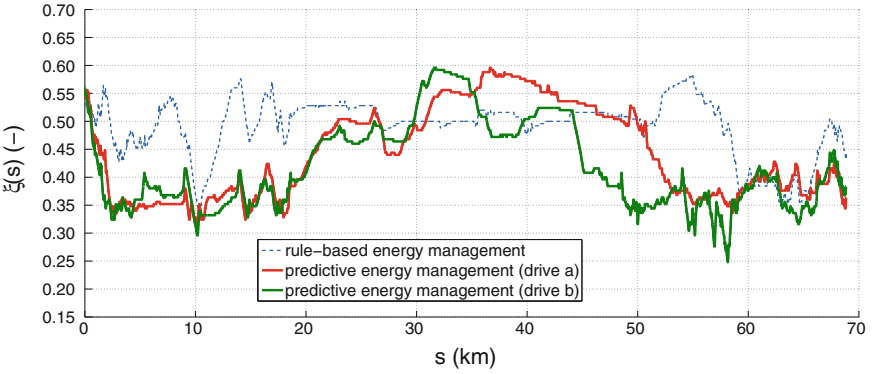


(c) Costate trajectories $\overline{\lambda}$ for drive a, b, and c

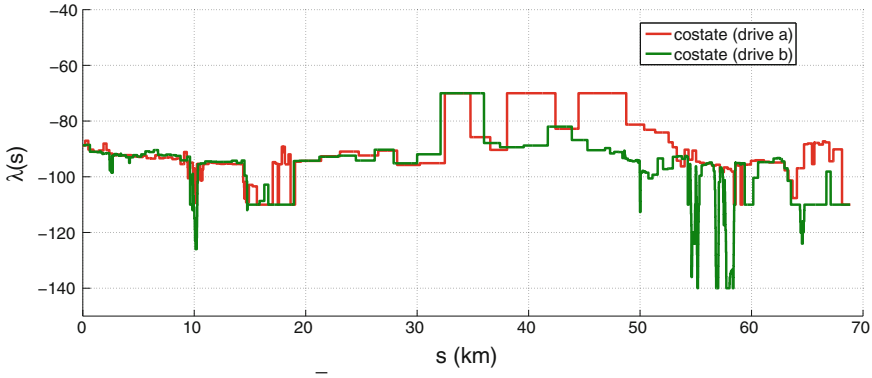
Fig. 12.14 Trajectories of the real-world benchmark-cycle 2



(a) The dashed blue curve is the measured $\bar{\mathbf{v}}$-trajectory for the drive with the rule-based energy management, the solid curves are the measured $\bar{\mathbf{v}}$-trajectories for drives with the predictive energy management

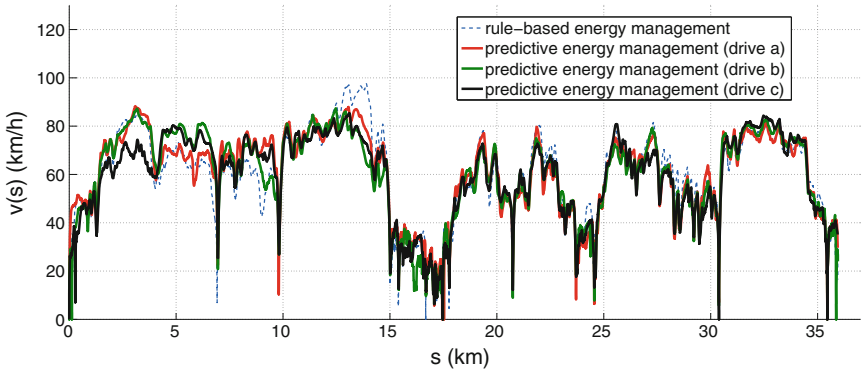


(b) The dashed blue curve is the measured $\bar{\boldsymbol{\xi}}$-trajectory for the drive with the rule-based energy management. The solid curves are the measured $\bar{\boldsymbol{\xi}}$-trajectories for predictive energy management for drive a and b

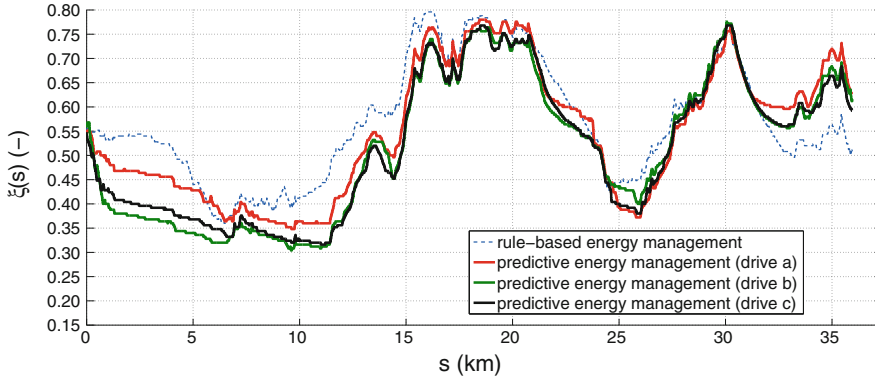


(c) Costate trajectories $\bar{\boldsymbol{\lambda}}$ for drive a and b

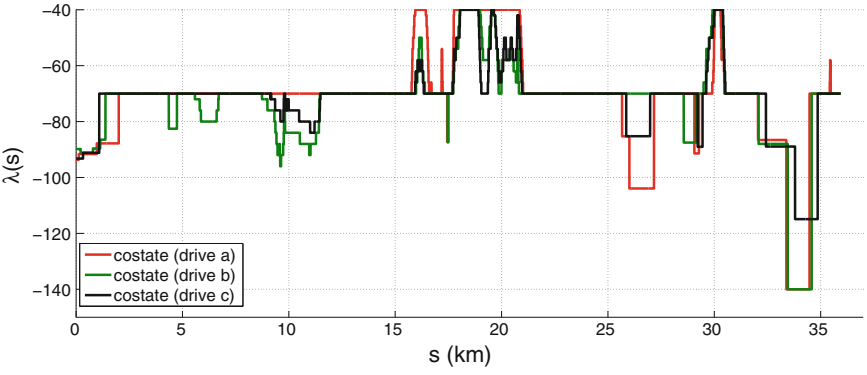
Fig. 12.15 Trajectories of the real-world benchmark-cycle 3



(a) The dashed blue curve is the measured $\bar{\mathbf{v}}$-trajectory for the drive with the rule-based energy management, the solid curves are the measured $\bar{\mathbf{v}}$-trajectories for drives with the predictive energy management



(b) The dashed blue curve is the measured $\bar{\boldsymbol{\xi}}$-trajectory for the drive with the rule-based energy management. The solid curves are the measured $\bar{\boldsymbol{\xi}}$-trajectories for predictive energy management for drive a, b, and c



(c) Costate trajectories $\bar{\boldsymbol{\lambda}}$ for drive a, b, and c

Fig. 12.16 Trajectories of the real-world benchmark-cycle 4

12.5.1.5 Performance Issues

The question about stability of the proposed event-triggered nonlinear model-predictive control for energy management cannot be answered easily and only conservative estimates can usually be obtained. This topic is beyond the scope of this book and interested readers find valuable results in the following literature Chen and Allgöwer [10], Findeisen et al. [13], and Fontes [14].

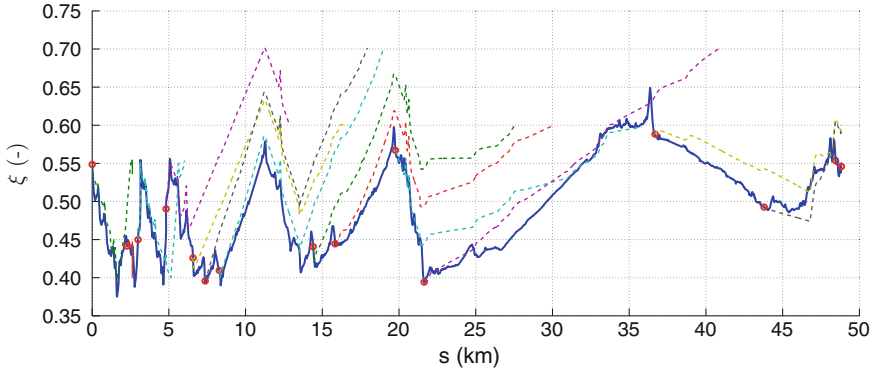
Therefore, we restrict the concerns about stability to the following comments:

- since the process is steered in open-loop until the threshold ξ_{thd} is exceeded, we cannot expect asymptotic stability of the closed-loop, but ultimate boundedness around the origin; and
- the open-loop model is stable and of first-order type. But the closed-loop system is of second-order due to the measured state. Then, there might be an event-triggering sequence such that the closed-loop system becomes instable or a limit cycle.

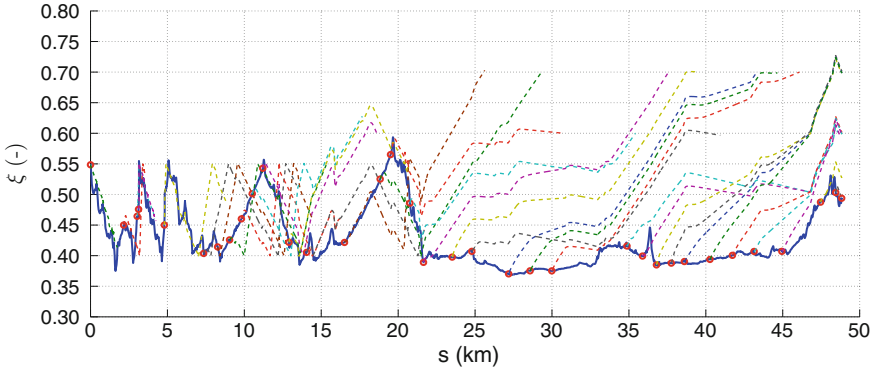
Let us assume for the remaining chapter that our closed-loop system is stable with respect to ultimate boundedness (Khalil and Grizzle [26]). Loosely speaking, a solution trajectory is said to be uniformly ultimately bounded, if there exists a constant b such that the solution trajectory is bounded $|\xi(t)| < b$ for any time. The key point is that the solution trajectory is not asymptotically stable, which implies that $\lim_{t\to\infty} \xi(t) = 0$ can not be realized.

One practical way to assess the robustness of event-triggered predictive energy management is to perturb several parameters of the nominal process model by introducing modeling errors and then to analyze the effects on the target value and the number of iterations necessary to converge to the local minimum. Therefore, the values of the traffic limitation and the drag parameters a_0, a_1, and a_2 of the driver model are disturbed with random variables subject to a normal distribution and a standard deviation of 20% of the original value. Simulations over the same benchmark-cycle were repeatedly performed with these uncertainties, but not used in the prediction. In both cases, with undisturbed and disturbed model parameters, if the predicted and measured $\overline{\boldsymbol{\xi}}$ trajectories deviate more than $\xi_{rst} = 0.05$, than a re-optimization of the OCP over a prediction horizon of 10 min is triggered. One can readily observe from Fig. 12.17 that for the disturbed model parameters the deviations between predicted and measured $\overline{\boldsymbol{\xi}}$ trajectories exceed far more frequent the 5% threshold and hence requires more than twice as much re-calculations of the costate. However, the fuel consumptions of the disturbed system deviate only in the range of 0.48–0.75% from the fuel consumptions of the undisturbed system, which demonstrates the robustness of predictive energy management in achieving the target value under model uncertainties.

A high number of re-computations can certainly derogate the attractiveness of the control strategy because modern ECUs are designed with less additional computing capabilities. It is, therefore, desirable to reduce the computational base load of this control strategy as much as possible and thus the number of triggered events without deterioration of the fuel savings. The improvement of the accuracy of the predicted vehicle speed trajectory is certainly one key element and can be accomplished by



(a) The dashed curves are the predicted $\bar{\xi}_p$ trajectories for the undisturbed case. The solid blue curve is the simulated $\bar{\xi}$ trajectory from the vehicle model. The red filled circles indicate re-optimization events



(b) The dashed curves are the predicted $\bar{\xi}_p$ trajectories for the disturbed case. The solid blue curve is the simulated $\bar{\xi}$ trajectory from the vehicle model. The red filled circles indicate re-optimization events

Fig. 12.17 Predictions and number of optimizations for undisturbed and disturbed model parameters

using more attributes of the route (e.g., curve radii) or traffic flow information (e.g., traffic density) in the driver model.

12.5.2 Predictive Energy Management with Long Prediction Horizon

Energy management for PHEVs with large battery packs need long prediction horizons to find the optimal controls. In case of the previously discussed event-triggered

predictive control strategy a long prediction time horizon is prone to be interrupted by disturbances. This lack of robustness of the event-triggered predictive control strategy for long predictions can result in many re-optimization steps depending on the driving profile, the quality of the vehicle speed prediction, and the accuracy of the nominal vehicle model. In this section, we present a more robust predictive control strategy for long prediction horizons that uses again an indirect shooting method to solve a SOCP, involving the torque-split as continuous-valued control $\overline{\mathbf{u}}_{[k]}$ and the drive mode as discrete decisions $\overline{\boldsymbol{\zeta}}_{[k]}$. The gear-selection strategy is not regarded in the SOCP but is defined as rule-based gear-selection strategy similar to a real vehicle. The necessary informations for solving the SOCP are derived again from a driver model and a vehicle model. Instead of re-calculation of the SOCP in case of deviations from the nominal model, the constant costate value is continuously adapted to the new situation.

The predictive energy management is implemented for *charge-depleting* (CD) and *charge-sustaining* (CS) drive modes but can be generalized to any situation, where a certain target value of $\overline{\boldsymbol{\xi}}$ has to be attained over a predictable cycle. For example, before driving through a zero-emission-zone it requires that the battery's state of charge is conditioned for correctly passing through this area without starting the ICE. In other words, the predictive energy management has to decide when and how intensive should the battery during the remaining drive cycle be charged before reaching the zero-emission-zone to ensure that the vehicle can be propelled just with the MG.

For the CD-strategy, whose target is to minimize the fuel consumption $\overline{\boldsymbol{\beta}}$ under the condition that the battery's state of charge $\overline{\boldsymbol{\xi}}$ be nearly minimal, when reaching the target destination, the SOCP can be formulated in discretized form as follows:

$$\mathcal{P}_8 := \begin{cases} \min\limits_{\overline{\boldsymbol{\zeta}}_{[i]},\, \overline{\mathbf{u}}_{[i]}} \overline{\boldsymbol{\beta}}_{[N_s]} & \\ \overline{\boldsymbol{\beta}}_{[i+1]} & = \overline{\boldsymbol{\beta}}_{[i]} + \Delta s \cdot \gamma_f \cdot \overline{\boldsymbol{\zeta}}_{[i]} \cdot \mathrm{bsfc}\left(\overline{\mathbf{u}}_{[i]}, \overline{\boldsymbol{\omega}}_{ice,p}^{[i]}\right) \cdot \dfrac{\overline{\mathbf{u}}_{[i]} \cdot \overline{\boldsymbol{\omega}}_{ice,p}^{[i]}}{\overline{\mathbf{v}}_p^{[i]}} \\ \overline{\boldsymbol{\xi}}_{ref}^{[i+1]} & = \overline{\boldsymbol{\xi}}_{ref}^{[i]} + \dfrac{\Delta s}{Q_{bat}} \cdot \dfrac{\overline{\mathbf{I}}_{bat,\xi}^{[i]}}{\overline{\mathbf{v}}_p^{[i]}} \\ \overline{\boldsymbol{\beta}}_{[0]} & = 0 \\ \overline{\boldsymbol{\xi}}_{ref}^{[0]} & = \xi_0 \\ \overline{\boldsymbol{\xi}}_{ref}^{[N_s]} & = \xi_f \end{cases} \tag{12.34}$$

where ξ_f is the desired target value for the state of charge at the end of the trip and is defined to be nearby ξ^{min}.

The control constraints are defined as

$$\mathbf{c}_{\overline{u}}\left(\overline{\mathbf{u}}_{[i]},\overline{\boldsymbol{\zeta}}_{[i]}\right) := \begin{bmatrix} \overline{\boldsymbol{\zeta}}_{[i]} - 1 \\ 0 - \overline{\boldsymbol{\zeta}}_{[i]} \\ \overline{\mathbf{u}}_{[i]} - T_{ice}^{max}\left(\overline{\boldsymbol{\omega}}_{ice,p}^{[i]}\right) \\ T_{ice}^{min} - \overline{\mathbf{u}}_{[i]} \\ \overline{\mathbf{T}}_{gbx,p}^{[i]} - \overline{\mathbf{u}}_{[i]} - T_{mg}^{max}\left(\overline{\boldsymbol{\omega}}_{mg,p}^{[i]}\right) \\ T_{mg}^{min}\left(\overline{\boldsymbol{\omega}}_{mg,p}^{[i]}\right) + \overline{\mathbf{u}}_{[i]} - \overline{\mathbf{T}}_{gbx,p}^{[i]} \end{bmatrix} \leq \mathbf{0} \tag{12.35}$$

where the predicted trajectories $\overline{\mathbf{v}}_p$, $\overline{\mathbf{T}}_{gbx,p}$, $\overline{\boldsymbol{\omega}}_{ice,p}$, and $\overline{\boldsymbol{\omega}}_{mg,p}$ serve as fixed values, N_s corresponds to the total length of the route, and the boundary values $T_{mg}^{max}(\cdot)$, $T_{mg}^{min}(\cdot)$, and $T_{ice}^{max}(\cdot)$ for MG and ICE, respectively, depend on the speeds.

The task of solving $\mathcal{P}_8$ is to find the controls $\overline{\boldsymbol{\zeta}}_{[i]}^{*}$ and $\overline{\mathbf{u}}_{[i]}^{*}$ that minimize the fuel consumption over the spatial grid (12.11). This leads to a minimum of local CO_2 emissions produced by the ICE. A CS-strategy is simply be obtained by exchanging the final state boundary condition in $\mathcal{P}_8$ with

$$\overline{\boldsymbol{\xi}}_{ref}^{[N_s]} = \xi_0.$$

As depicted in Fig. 12.18, the robust predictive control strategy consists of the following elements:

- **ITS**: provides information on speed limit and road slope;
- **spatial-based driver-model**: calculates the trajectories $\overline{\mathbf{v}}_p$ and $\overline{\mathbf{T}}_{wh,p}$ over the spatial grid;
- **vehicle-model**: calculates the trajectories $\overline{\mathbf{T}}_{gbx,p}$, $\overline{\boldsymbol{\omega}}_{ice,p}$, and $\overline{\boldsymbol{\omega}}_{mg,p}$ over the spatial grid;
- **trajectory planning**: calculates a reference trajectory for $\overline{\boldsymbol{\xi}}_{ref}$ and a constant costate λ_{ref};
- **instantaneous costate updates**: calculates an offset $\Delta\overline{\boldsymbol{\lambda}}_{[k]}$ to the constant costate to assure that the reference trajectory is being followed; and
- **LUTs**: calculation of the instantaneous controls $\overline{\mathbf{u}}_{[k]}$ and $\overline{\boldsymbol{\zeta}}_{[k]}$.

We use again an ITS, which provides a prediction for the speed limit and the road slope to the reconstructor on the application side. The quantities $\overline{\mathbf{v}}_p^{[i]}$, $\overline{\boldsymbol{\omega}}_{ice,p}^{[i]}$, and $\overline{\boldsymbol{\omega}}_{mg,p}^{[i]}$ are determined for the complete drive cycle from the spatial-based driver model and the vehicle model and then passed to the trajectory planning unit. This unit calculates a reference trajectory for the battery's state of charge $\overline{\boldsymbol{\xi}}_{ref}^{[i]}$ and a constant costate λ_{ref} by solving $\mathcal{P}_8$ over the spatial grid. This calculation is performed once and only repeated if the route has been changed. The constant costate is adapted to the current driving situation at each time instant k. An offset $\Delta\lambda_k$, which depends on the deviation of the measured $\overline{\boldsymbol{\xi}}$ and the predicted $\overline{\boldsymbol{\xi}}_{ref}$ trajectories and which is calculated by a PI controller, is added to the costate. The instantaneous controls for the torque-split $\overline{\mathbf{u}}_{[k]}$

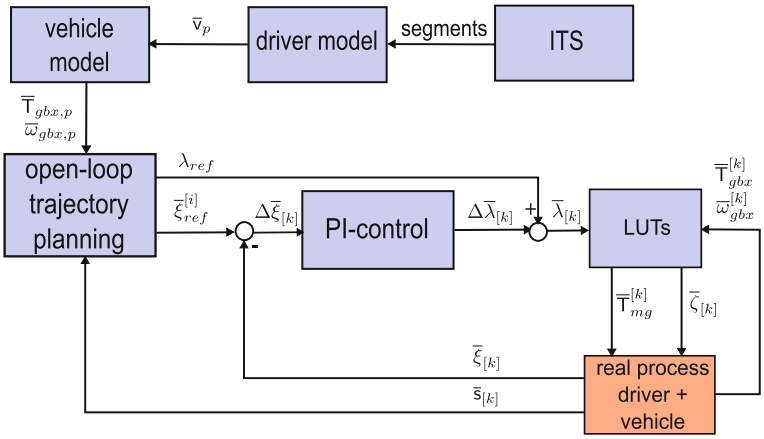


Fig. 12.18 Control structure of robust predictive energy management

and the drive mode $\overline{\boldsymbol{\zeta}}_{[k]}$ are then determined by LUTs using the measured vehicle speed $\overline{\mathbf{v}}_{[k]}$, the measured wheel torque $\overline{\mathbf{T}}_{wh}^{[k]}$, and the adapted costate $\overline{\boldsymbol{\lambda}}_{[k]}$.

12.5.2.1 Solving the SOCP Online

For the online application of the indirect shooting approach we need to store the controls for the MG and for the drive mode in LUTs $\hat{T}_{mg}(\overline{\boldsymbol{\omega}}_{gbx}^{[k]}, \overline{\mathbf{T}}_{gbx}^{[k]}, \lambda)$ and $\hat{T}_{start}(\overline{\boldsymbol{\omega}}_{gbx}^{[k]}, \lambda)$ according to Sect. 11.3.3. The predicted mode-sequence $\overline{\boldsymbol{\zeta}}_p$ is obtained from the predicted gearbox input speed $\overline{\boldsymbol{\omega}}_{gbx,p}$ as follows:

$$\overline{\boldsymbol{\zeta}}_p^{[i]} = \begin{cases} 1, & \overline{\boldsymbol{\omega}}_{gbx,p}^{[i]} \geq \hat{T}_{start}(\overline{\boldsymbol{\omega}}_{gbx,p}^{[i]}, \lambda_{ref}) \\ 0, & \overline{\boldsymbol{\omega}}_{gbx,p}^{[i]} < \hat{T}_{start}(\overline{\boldsymbol{\omega}}_{gbx,p}^{[i]}, \lambda_{ref}). \end{cases}$$

The quantities $\overline{\boldsymbol{\omega}}_{gbx,p}^{[i]}$ and $\overline{\mathbf{T}}_{gbx,p}^{[i]}$ are obtained from the vehicle model (Sect. 12.5.1.1) in spatial domain.

The costate λ_{ref} is assumed again to be constant and obtained from solving the SOCP over the entire drive cycle. Once the IVP has been solved, the boundary value $\overline{\boldsymbol{\xi}}_{ref}^{[N_s]}$ can be evaluated and the initial guess of $\lambda_{ref,0}$ can be improved. The SOCP is therefore reduced to solve the scalar equation

$$\overline{\boldsymbol{\xi}}_{ref}^{[N_s]} - \xi_f = 0. \tag{12.36}$$

The Pegasus method is used again and performs efficiently and robustly even on a rapid-prototyping platform.

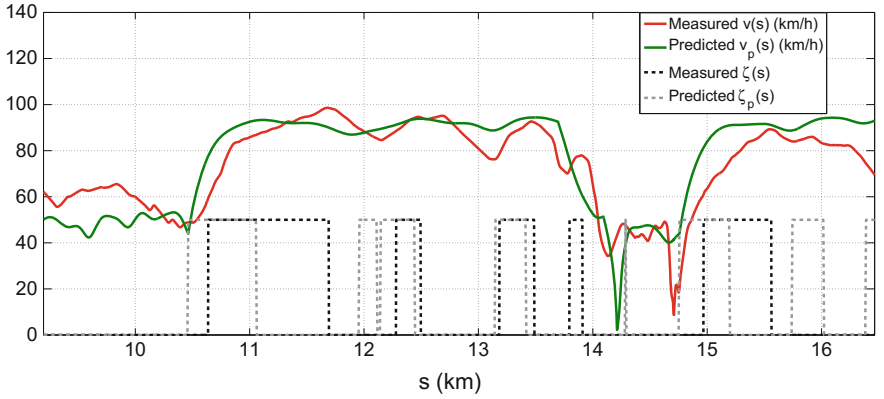


Fig. 12.19 Predicted vehicle speed trajectory $\overline{\mathbf{v}}_p$ and predicted optimal engine start/stop sequence $\overline{\boldsymbol{\zeta}}_p$

Figure 12.19 shows a comparison of the predicted and measured vehicle speed trajectories, $\overline{\mathbf{v}}_p$ and $\overline{\mathbf{v}}$, respectively, and the predicted and measured engine start/stop trajectories, $\overline{\boldsymbol{\zeta}}_p$ and $\overline{\boldsymbol{\zeta}}$, respectively. Deviations can be noticed, especially, the predicted engine start/stop sequence deviates notably at some arcs. The unpredictable behavior of the driver that demands sometimes more or less wheel torque as estimated is one of the major sources for this behavior. Additionally, the torque structure of the ECU can prevent mode changes dependent on further information, which is not used for the prediction. However, these deviations are still acceptable.

When the trajectory $\overline{\boldsymbol{\xi}}_{ref}$ on the spatial grid contains a boundary arc for which

$$\overline{\boldsymbol{\xi}}_{ref}^{[i]} < \xi^{min}$$

applies, an interior-boundary condition is added to the SOCP. The spatial instant $s_{i_{min}}$ that has the lowest value $\overline{\boldsymbol{\xi}}_{ref}^{[i]}$ is identified and the interior-boundary condition

$$\overline{\boldsymbol{\xi}}_{ref}^{[i_{min}]} - \xi^{min} = 0 \tag{12.37}$$

is added to the SOCP formulation, as proposed in de Jager et al. [18]. The SOCP is then resolved, first over the interval $[s_0, s_{i_{min}})$ with (12.37) as final state and then over the interval $[s_{i_{min}}, s_{N_s}]$ with ξ^{min} as initial value and ξ_f as final value. The procedure is depicted in Fig. 12.20. The $\overline{\boldsymbol{\xi}}_{ref}$-trajectory 1 contains an arc that falls below a lower boundary $\xi^{min} = 0.2$. The spatial instant $s_{i_{min}}$ is identified with 47.7 km which has the lowest value $\overline{\boldsymbol{\xi}}_{ref}^{[i_{min}]}$. An interior-boundary condition is then inserted that requires this point to be on the lower state bound ξ^{min}. The $\overline{\boldsymbol{\xi}}_{ref}$-trajectory 2 is the solution with interior-point condition as initial condition.

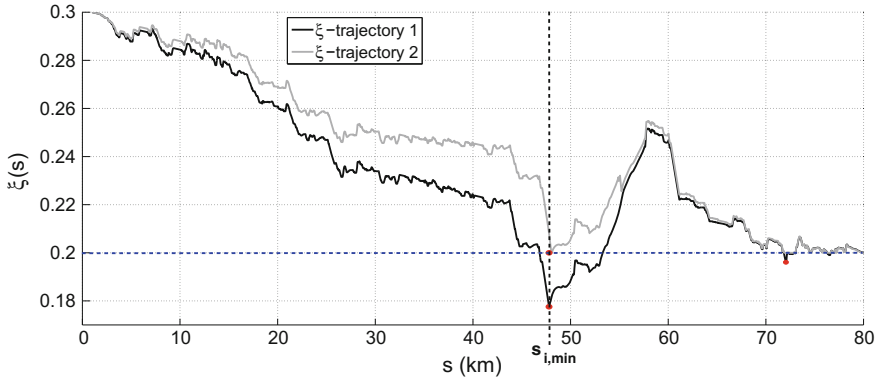


Fig. 12.20 Procedure for obtaining a suboptimal trajectory fulfilling the state constraint. The *blue dashed line* indicates the state boundary ξ^{min}

12.5.2.2 Instantaneous Costate Updates

Several uncertainties in the trajectory planning make a controller inalienable, among them model error and error due to discretization. As already said, the most influential uncertainty, however, is the driver behavior, which can be only roughly predicted. The costate λ can well be used as control variable, as it has a direct influence on the torque-split $\hat{T}_{mg}(\overline{\boldsymbol{\omega}}_{gbx}^{[k]}, \overline{\mathbf{T}}_{gbx}^{[k]}, \lambda)$ as well as on the start torque $\hat{T}_{start}(\overline{\boldsymbol{\omega}}_{gbx}^{[k]}, \lambda)$. A lower value of the costate will lead to earlier engine starts and higher load torques for the ICE. A simple PI controller can fulfill the task of disturbance rejection. Small deviations from the planned $\overline{\boldsymbol{\xi}}_{ref}$-trajectory are acceptable. Thus, the gains of proportional path and integral path are kept rather low, which has shown to be advantageous for the fuel consumption. The instantaneous controls $\overline{\mathbf{T}}_{ice}^{[k]}$ and $\overline{\boldsymbol{\zeta}}^{[k]}$ can be determined using the LUTs $\hat{T}_{mg}(\overline{\boldsymbol{\omega}}_{gbx}^{[k]}, \overline{\mathbf{T}}_{gbx}^{[k]}, \overline{\boldsymbol{\lambda}}_{[k]})$ and $\hat{T}_{start}(\overline{\boldsymbol{\omega}}_{gbx}^{[k]}, \overline{\boldsymbol{\lambda}}_{[k]})$ defined in (11.9) and (11.10) with the corrected costate $\overline{\boldsymbol{\lambda}}_{[k]}$

$$\overline{\boldsymbol{\lambda}}_{[k]} = \lambda_{ref} + \Delta\overline{\boldsymbol{\lambda}}_{[k]}$$

and the current driving condition $\overline{\boldsymbol{\omega}}_{gbx}^{[k]}$ and $\overline{\mathbf{T}}_{gbx}^{[k]}$. Since the measured value $\overline{\boldsymbol{\xi}}_{[k]}$ is only estimated in the real vehicle and the estimation is corrected from time to time, jumps may occur. Due to the low controller gains, this will not cause instabilities.

12.5.2.3 Experimental Results

The predictive strategy is implemented in a PHEV with a battery capacity of 7.5 kWh. As prototyping unit a dSPACE MicroAutobox is used. The route is discretized with $N_s = 10000$ discretization values. The calculation of the reference trajectory

$\bar{\boldsymbol{\xi}}_{ref}^{[i]}$ and the constant value λ_{ref} on the MicroAutobox takes between 1 and 5 seconds, depending on the number of iterations needed to solve (12.36). To avoid frequent engine starts, a hysteresis around $\hat{T}_{start}$ is defined and turn-on/turn-off-delays, whose parameters were defined offline using a genetic algorithm, are implemented.

As drive cycle, the real-world benchmark-cycle 1 is used. The test-drives were again conducted by different drivers and during different day-times to investigate the robustness toward deviations from the predicted drive profile.

Charge-Blending Operation

The CD-strategy is employed when the target destination provides a charging facility and the total driving distance exceeds the electrical range for the current state of charge. In this case the entire electrical energy can be depleted but the ICE has to be started several times, to prevent the battery from falling below its minimum value before the target destination is reached. It is the task of predictive energy management,

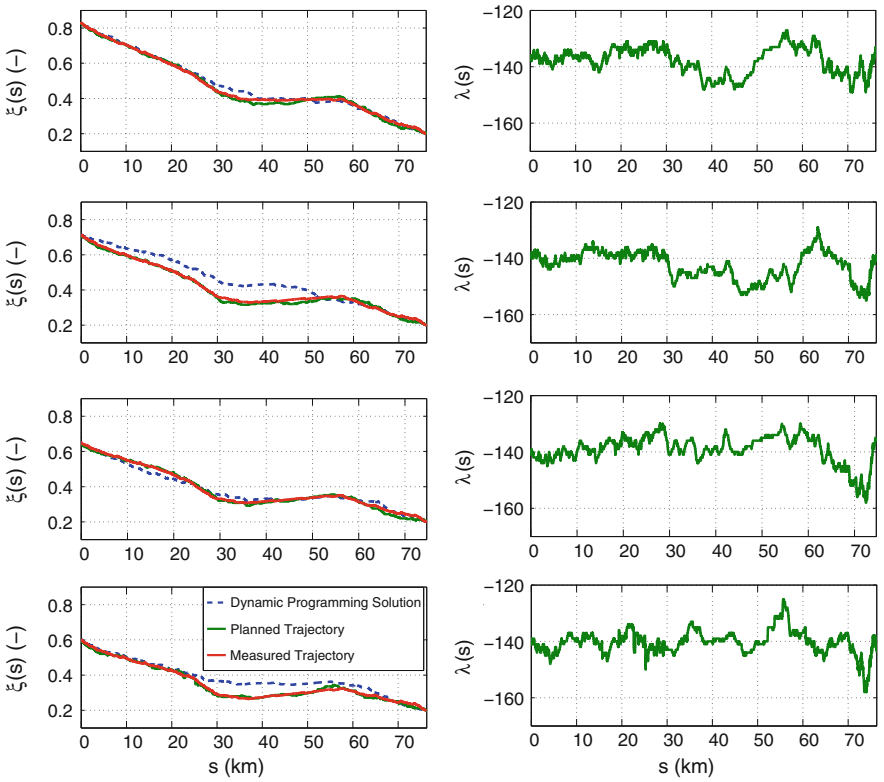


Fig. 12.21 Planned and measured trajectories for $\bar{\boldsymbol{\xi}}$ and the corrected costate $\bar{\boldsymbol{\lambda}}$ for four measurements over the benchmark-cycle 1 with different initial charging states for the CB mode. A DP solution serves as reference $\bar{\boldsymbol{\xi}}$-trajectory

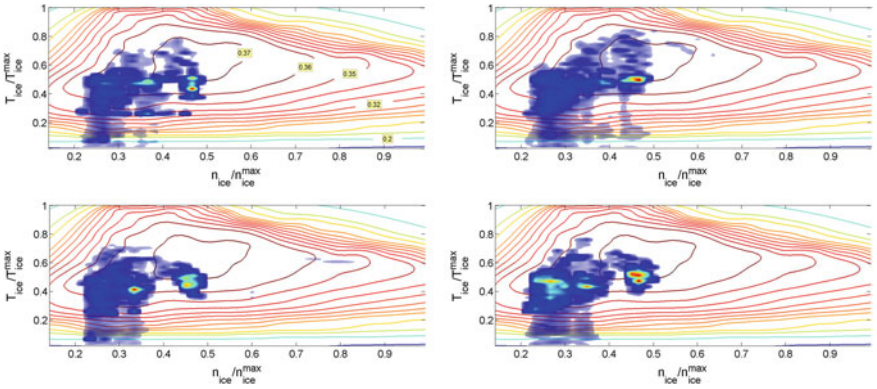


Fig. 12.22 ICE efficiencies obtained with predictive energy management for the CB mode

to determine when the ICE is started and to determine the torque-split. The target value for the state of charge ξ_f is set to 0.2.

Figure 12.21 depicts the reference trajectories $\overline{\boldsymbol{\xi}}_{ref}$ and the measured trajectories $\overline{\boldsymbol{\xi}}$ for the state of charge of four test drives. The reference trajectories can be followed and the desired target value $\xi_f = 0.2$ is reached with a narrow tolerance $\Delta\xi = \overline{\boldsymbol{\xi}}_{[N_s]} - 0.2$ of $\pm 1.2\%$ in all test cases. Even though the predicted vehicle speed trajectory is close to the measured profile, the controller has to correct the costate visibly. This is mostly due to the prediction error in the engine start/stop sequence $\overline{\boldsymbol{\zeta}}_p$ as can be seen for the four test drives in Figs. 12.24, 12.25, 12.26, and 12.27. Figure 12.22 depicts the operation points of the ICE of the four test drives in the efficiency map. Especially during the highway-drive, where higher engine speeds occur, the ICE operates with nearly optimal efficiency. This is also the case for lower speeds, where the spread is slightly higher due to the less constant driving conditions in urban or rural driving situations. The best points obtained by predictive energy management are tightly controlled compared with the operating points obtained from rule-based energy management for two test drives. Certainly, rule-based energy management is

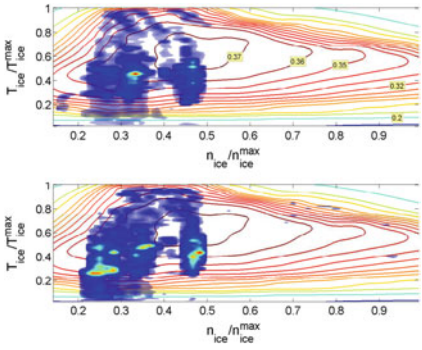


Fig. 12.23 ICE efficiencies obtained with a rule-based energy management for the CB mode

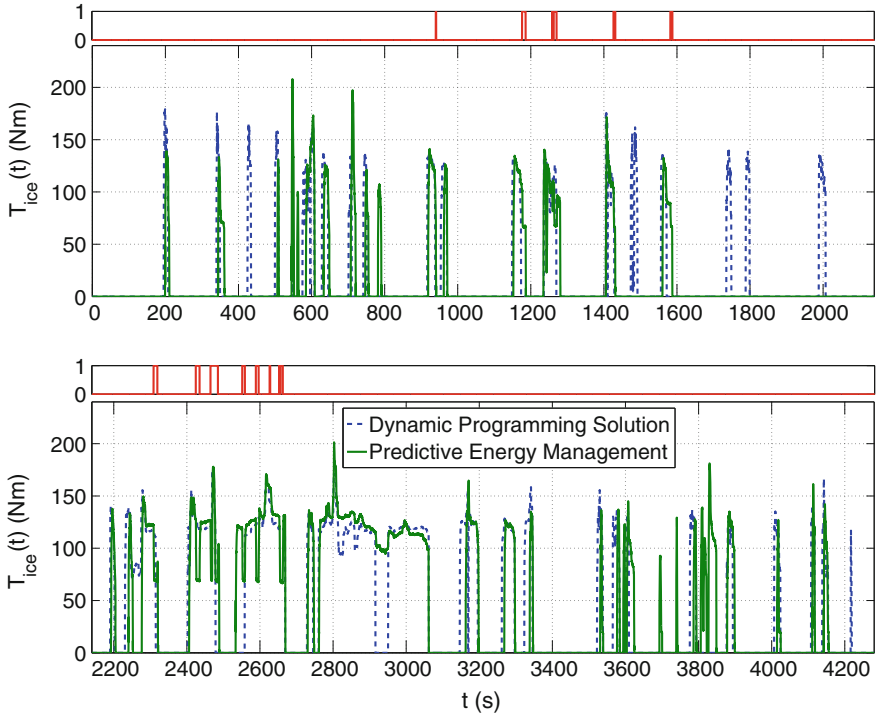


Fig. 12.24 First comparison of $\overline{\mathbf{T}}_{ice}$-trajectories from the measured solution of predictive energy management and the DP solution for the benchmark-cycle 1 for the CB mode. The *upper plot* shows the trajectories over $t \in [0, 2140]$, the *lower plot* shows the trajectories over $t \in [2140, 4280]$

able to yield comparable results as shown in the upper plot of Fig. 12.23. However, such strategies perform badly, if the conditions change significantly from the nominal design expectation.

Figures 12.24, 12.25, 12.26, and 12.27 show a comparison of the ICE torques selected by predictive energy management and the optimal solution obtained with DP. Some measurements were provided to the DP procedure such that the same boundary conditions apply and the vehicle speed profile and the gear sequence are assumed to be exactly known. Hence, the figures show, how predictive energy management should have performed, if the benchmark-cycles were perfectly known. As the prediction still deviates from the real cycle, deviations from the optimal solution cannot be avoided. However, the results are very close in terms of the engine start/stop sequence and the ICE torques chosen. The differences in fuel consumption between the optimal solutions by DP and the measured predictive energy management solutions were in all cases below 2.5%.

The red signal indicates an intervention of a diagnosis function, which changes the ICE torque selected by predictive energy management.

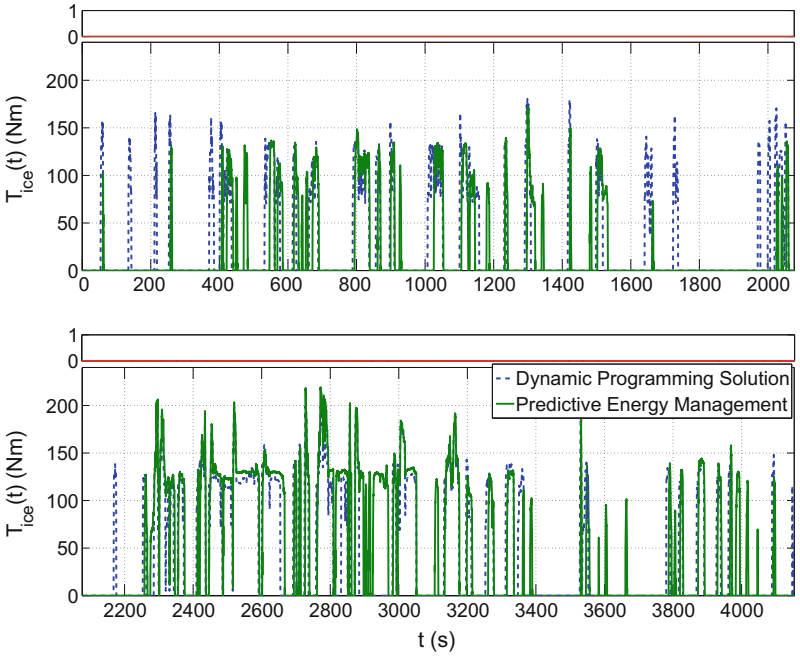


Fig. 12.25 Second comparison of $\overline{\mathbf{T}}_{ice}$-trajectories from the measured solution of predictive energy management and the DP solution for the benchmark-cycle 1 for the CB mode. The *upper plot* shows the trajectories over $t \in [0, 2076]$, the *lower plot* shows the trajectories over $t \in [2076, 4152]$

Table 12.4 shows a quantitative comparison of the rule-based and predictive energy management for the benchmark-cycle 1 of a total distance of 75 km.

The table entries are calculated with the following assumptions: charging efficiency $\eta_{grid} = 0.7$, fuel cost 1.49 €/l, and electrical energy cost 26 ct/kWh.

Charge-Sustaining Operation:

The CS-strategy is an alternative DOF of the PHEV that can also be chosen manually. In this case the battery's state of charge is maintained. It is the task of predictive energy management, to determine when the ICE is started and to determine the torque-split.

Figure 12.28 depicts the reference trajectories $\overline{\boldsymbol{\xi}}_{ref}$ and the measured trajectories $\overline{\boldsymbol{\xi}}$ for the state of charge of four test drives. Compared with the CB mode the costate is adjusted more frequently by the PI controller to maintain the state of charge. The best points obtained by predictive energy management as shown in Fig. 12.29 are again more tightly controlled compared with the operating points obtained from rule-based energy management as shown in Fig. 12.30 for four test drives. Notably, the clusters for sub-urban parts at low engine torques and low engine speed are shifted by predictive energy management to clusters at higher loads and therefore to better efficiency.

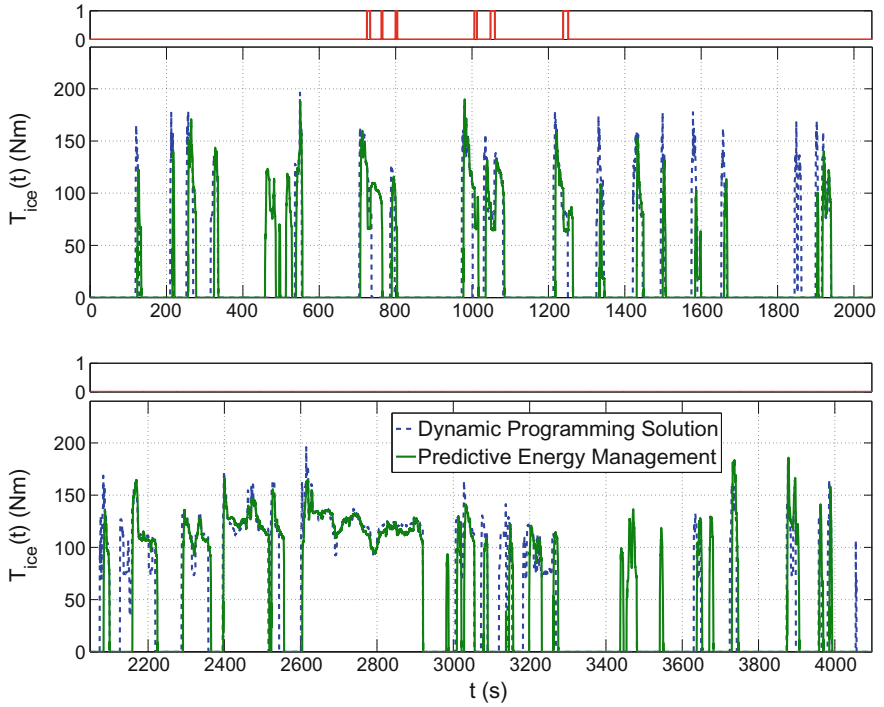


Fig. 12.26 Third comparison of $\overline{\mathbf{T}}_{ice}$-trajectories from the measured solution of predictive energy management and the DP solution for the benchmark-cycle 1 for the CB mode. The *upper plot* shows the trajectories over $t \in [0, 2048]$, the *lower plot* shows the trajectories over $t \in [2048, 4096]$

Figures 12.31, 12.32, 12.33 and 12.34 show a comparison of the ICE torques selected by predictive energy management and the optimal solution obtained with DP. The differences in fuel consumption between the optimal solutions by DP and the measured predictive energy management solutions were in all cases below 1.1%.

12.5.2.4 Robustness Aspects

Analogous to Sect. 12.5.1.5 several parameters of the nominal model are perturbed to verify the robustness of the predictive strategy against modeling errors. The drag parameters a_0, a_1, and a_2 were disturbed with an equally distributed factor $w \in [0.9, 1.1]$. The factor is applied to the simulated vehicle, but is not used in the prediction. Despite the model error, $\overline{\boldsymbol{\xi}}_{[N_s]} - \xi_f = 0$ is still fulfilled with a narrow tolerance, as can be seen from the left side of Fig. 12.35. From the scatter plot, it can be noted, that the distribution of the fuel consumption, which is caused by the disturbance of the drag coefficients, can be directly linked to the gearbox input energy

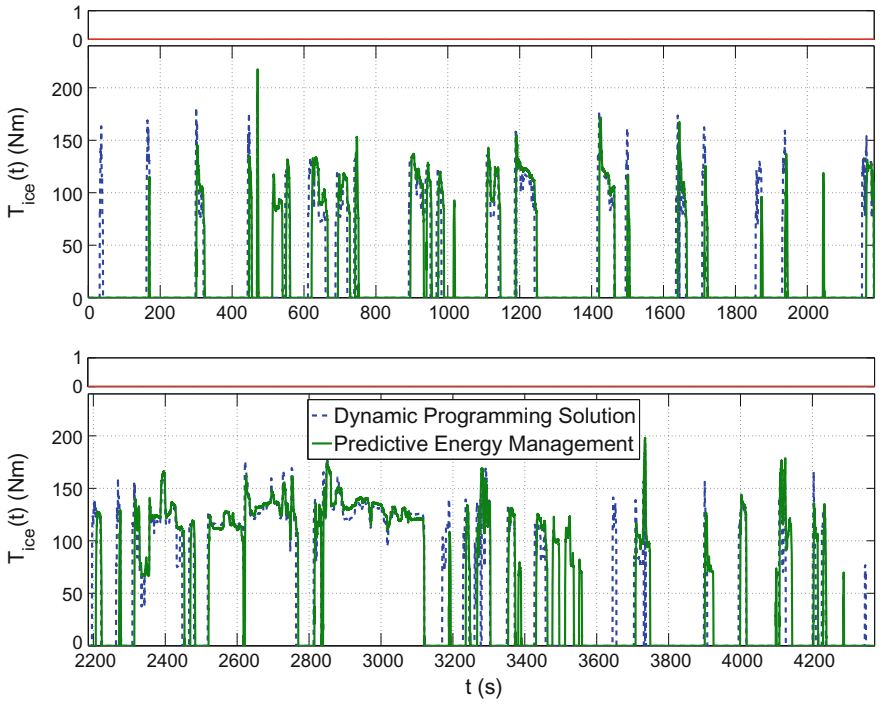


Fig. 12.27 Fourth comparison of $\overline{\mathbf{T}}_{ice}$-trajectories from the measured solution of predictive energy management and the DP solution for the benchmark-cycle 1 for the CB mode. The *upper plot* shows the trajectories over $t \in [0, 2186]$, the *lower plot* shows the trajectories over $t \in [2186, 4372]$

Table 12.4 Exemplary comparison of the fuel economy of predictive energy management and rule-based energy management

	Rule-based strategy	Predictive strategy
$\overline{\boldsymbol{\xi}}_{[0]}$ (-)	0.83	0.83
$\overline{\boldsymbol{\xi}}_{[N_s]}$ (-)	0.31	0.20
elec. energy (kWh)	4.44	5.43
$\overline{\boldsymbol{\beta}}_{[N_s]}$ (l)	2.98	2.46
elec. energy cost (€)	1.63	2.00
fuel cost (€)	4.44	3.66
total cost (€/100 km)	8.09	7.54

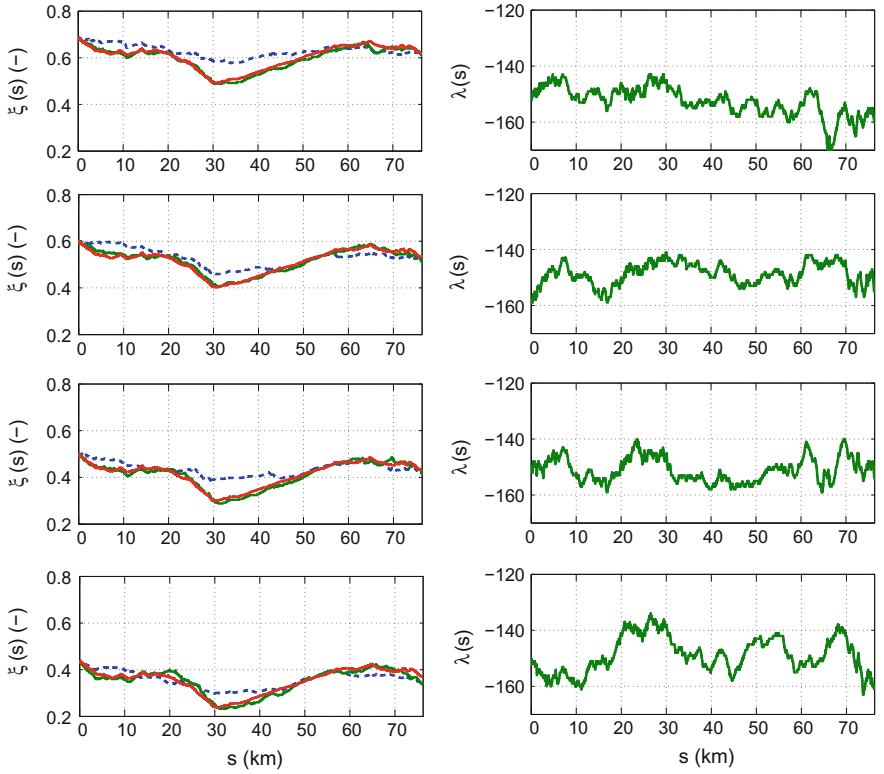


Fig. 12.28 Planned and measured trajectories for $\bar{\boldsymbol{\xi}}$ and the corrected costate $\bar{\boldsymbol{\lambda}}$ for four measurements over the benchmark-cycle 1 with different initial charging states for the CS mode. A DP solution serves as reference $\tilde{\boldsymbol{\xi}}$-trajectory

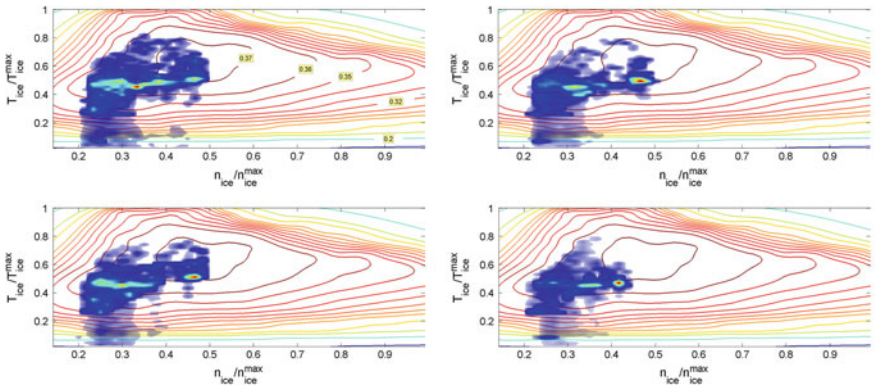


Fig. 12.29 ICE efficiencies obtained with predictive energy management for the CS mode

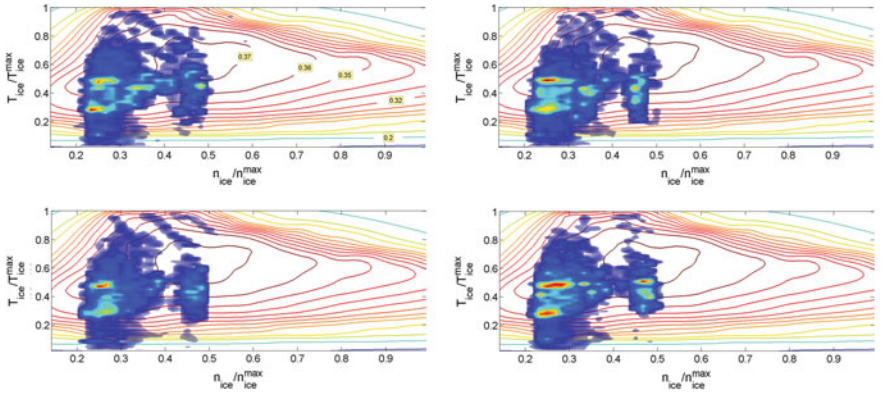


Fig. 12.30 ICE efficiencies obtained with rule-based energy management for the CS mode

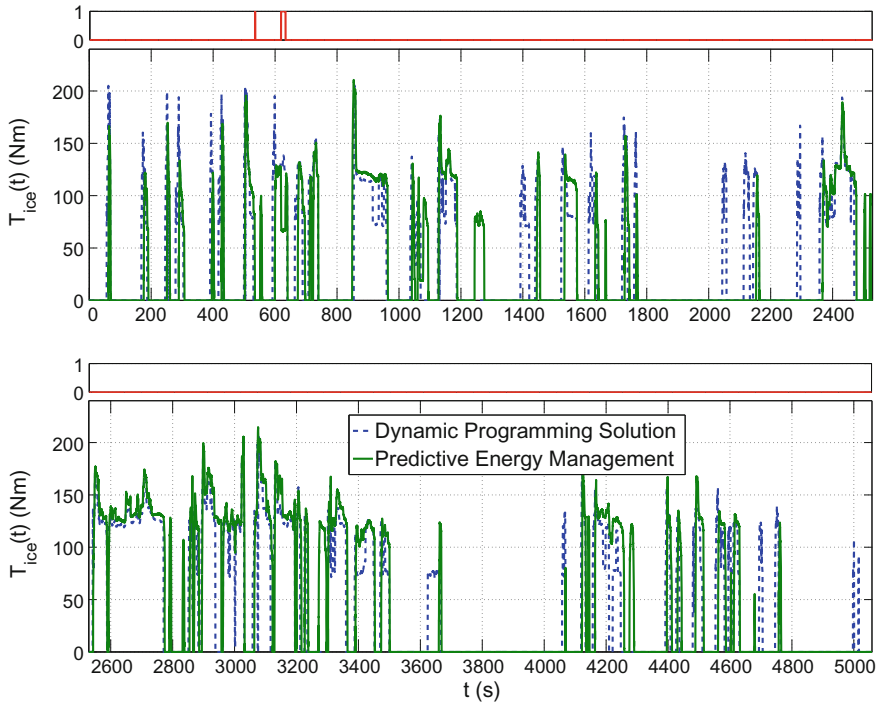


Fig. 12.31 First comparison of $\overline{\mathbf{T}}_{ice}$-trajectories from the measured solution of predictive energy management and the DP solution for the benchmark-cycle 1 for the CS mode. The *upper plot* shows the trajectories over $t \in [0, 2529]$, the *lower plot* shows the trajectories over $t \in [2529, 5058]$

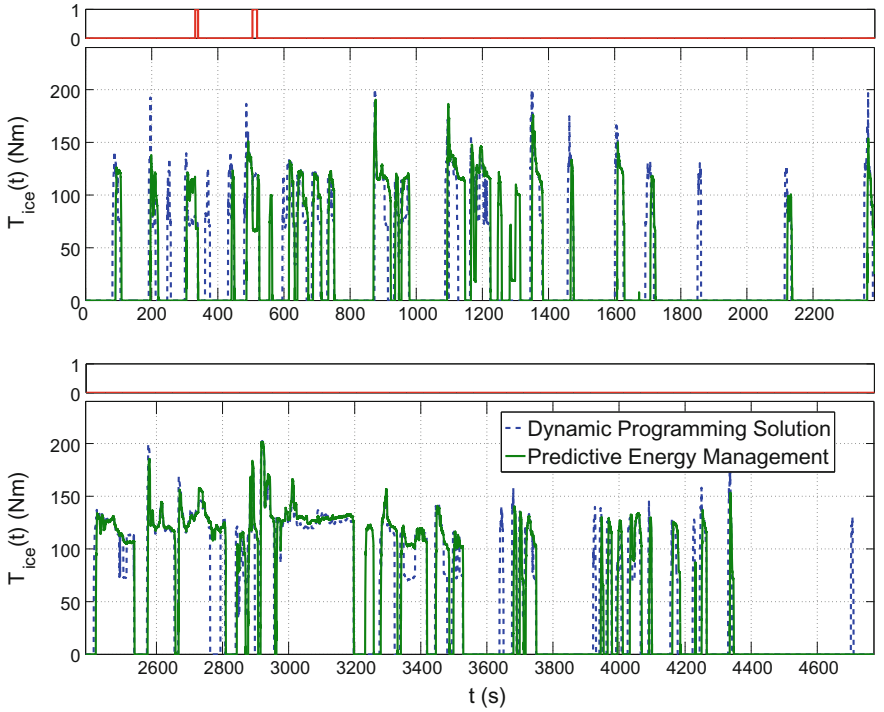


Fig. 12.32 Second comparison of $\overline{\mathbf{T}}_{ice}$-trajectories from the measured solution of predictive energy management and the DP solution for the benchmark-cycle 1 for the CS mode. The *upper plot* shows the trajectories over $t \in [0, 2385]$, the *lower plot* shows the trajectories over $t \in [2385, 4770]$

$$E_{gbx} = \frac{1}{3600} \int_{t_0}^{t_f} T_{gbx}(t) \cdot \omega_{gbx}(t) \, \mathrm{d}t$$

required to propel the vehicle. To survey the robustness against different drivers and traffic situations, simulations were performed over a diversity of 22 recorded velocity profiles. The predicted vehicle speed trajectory generated by the ITS and the driver model, as described in Sects. 12.3 and 12.3.2, respectively, does not reflect driver types nor does it include information on traffic density. Consequently, it predicts the same velocity profile each time. The effect on $\Delta\xi$ is in this case stronger, but still within acceptable bounds, as can be seen on the right side in Fig. 12.35.

12.6 Bibliographical Notes

In the literature on hybrid vehicle control, model-predictive control is often used for a rather short prediction horizon, as in the works of [1, 8]. However, strategies based on PMP or ECMS grow in importance as well. Many approaches based on finding

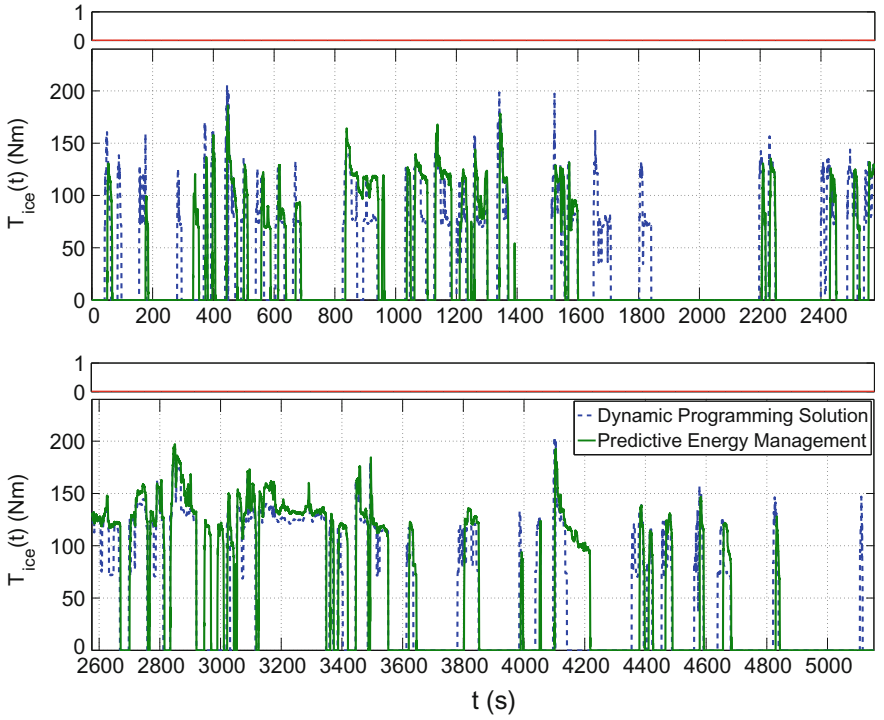


Fig. 12.33 Third comparison of $\overline{\mathbf{T}}_{ice}$-trajectories from the measured solution of predictive energy management and the DP solution for the benchmark-cycle 1 for the CS mode. The *upper plot* shows the trajectories over $t \in [0, 2576]$, the *lower plot* shows the trajectories over $t \in [2576, 5152]$

the costate for a purely continuous OCP are explained in the literature. A general framework for the adaptive control using ECMS by periodically updating the equivalence factor, based on a prediction of the driving profile, is described in Musardo et al. [34]. In Lee et al. [27] the costate is chosen from a table based on the predicted vehicle speed profile and on the average requested wheel-power. The prediction is then used to estimate an initial value of the costate. In the work of Kermani et al. [25], a model-predictive controller is implemented to determine the costate over a predicted driving profile. Predictive energy management that also incorporates discrete decisions is proposed by Johannesson et al. [20]. Herein, the optimization of the clutch-states and the optimization of the continuous-valued controls are performed in two stages. Dynamic programming and approximate dynamic programming are employed for the optimizations. In Katsargyri et al. [24], the route is decomposed into a series connection of route segments with partially known properties (e.g., the road class). Then, dynamic programming is applied to determine the sequence of the set-points for the battery's state of charge for each route segment. The authors proposed for on-board implementation a receding horizon control strategy [23]. An alternative to the sophisticated predictive control strategies are non-predictive strategies, which

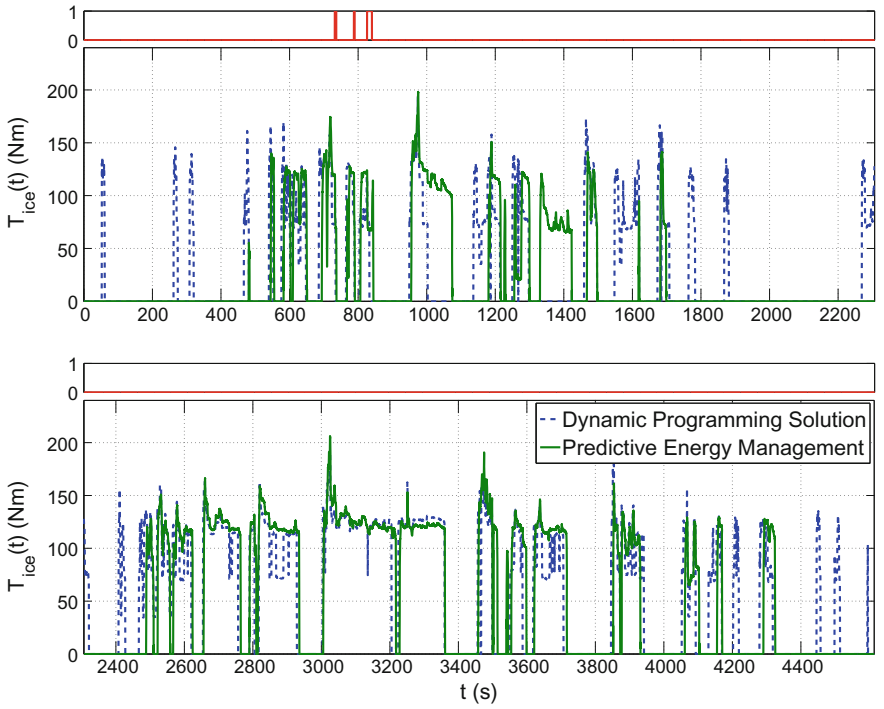


Fig. 12.34 Fourth comparison of $\overline{\mathbf{T}}_{ice}$-trajectories from the measured solution of predictive energy management and the DP solution for the benchmark-cycle 1 for the CS mode. The *upper plot* shows the trajectories over $t \in [0, 2306]$, the *lower plot* shows the trajectories over $t \in [2306, 4614]$

use information about the current driving situation provided by the navigation system and on-board sensors to identify the current road type or other helpful properties. Based on this instantaneous identification, a RB control strategy, which has been optimized for these conditions, can be selected as shown by Lin et al. [30]. A similar control strategy has been implemented by Boehme et al. [7] for a charge-sustaining HEV.

A predictive control strategy for EVs using spatial domain dynamic programming has been proposed by Boehme et al. [4, 5].

We showed in this chapter the usage of the indirect shooting method to implement MPC control strategies. Indirect shooting algorithms can be easily implemented on ECUs Schori et al. [41, 42], but have some tremendous drawbacks as discussed in Chap. 7. In case of state constraints, de Jager et al. [18] presented an algorithm that rewrites the state-constrained OCP as a sequence of OCPs without inequality state constraints, which can be solved with the indirect shooting methods from Chap. 7.

Several suggestions have been made to increase the level of route information mainly based on GPS, GIS, and historical data Lin and Peng [28] proposed a drive pattern recognition with DP solutions. A deep investigation of the potential of an

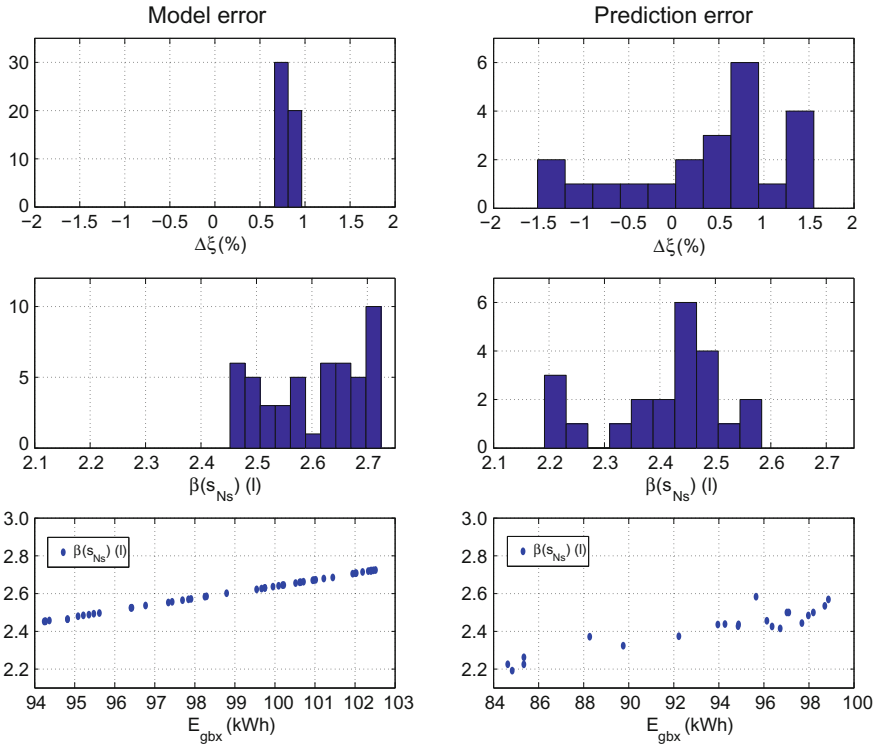


Fig. 12.35 Robustness of final state attainment and fuel consumption against modeling errors and unpredictable behavior of drivers

ITS and the established statistical and heuristic relationships of road events, like road profile, vehicle speed profile, trip distance, and weather conditions to the performance of energy management can be found in Gong et al. [17], Tulpule et al. [47]. A convenient way to get route information including speed limits, historic data on traffic pattern speeds, the road slopes, and the positions of stop signs and traffic lights is the ADAS research platform (Karbowski et al. [22]).

Many of the statistical relationships are based on Markov chain models, especially when a number of drive cycles can be found statistically representative of the vehicle utilization. Then, stochastic dynamic programming appears particularly appealing. Recent research, among them Moura et al. [33], Liu et al. [32], showed improvements in energy consumptions due to the ability to optimize the hybrid powertrains for a probabilistic distribution of many drive cycles rather than one single drive cycle. However, the stochastic dynamic programming approach is well-recognized to require a significant amount of data for validation and large computation time (Bertsekas [2], Johannesson et al. [19], Lin et al. [31]). The latter characteristic makes it difficult to implement this technique in a real-time algorithm.

The advantage of personalized driver models calibrated by machine learning tools is discussed in Carvalho et al. [9]. A driver model in time domain has been successfully applied to predictive energy management for a power-split hybrid Boehme et al. [6].

A nonlinear optimal regulation problem has been proposed by Sampathnarayanana et al. [38] for energy management using a Control-Lyapunov function.

References

1. Back M (2005) Prädiktive Antriebsregelung zum energieoptimalen Betrieb von Hybridfahrzeugen. PhD thesis, Universität Stuttgart
2. Bertsekas DP (1976) Dynamic programming and stochastic control. Academic Press, New York
3. Bin Y, Li Y, Gong Q, Peng ZR (2009) Multi-information integrated trip specific optimal power management for plug-in hybrid electric vehicles. In: Proceedings of the 2009 Amercian control conference. Hyatt Regency riverfront. IEEE, St. Louis, pp 4607–4612
4. Boehme TJ, Held F, Rollinger C, Rabba H, Schultalbers M, Lampe B (2013) Application of an optimal control problem to a trip-based energy management for electric vehicles. SAE Int J Alt Power 6(1):115–126. doi:10.4271/2013-01-1465
5. Boehme TJ, Held F, Schultalbers M, Lampe B (2013) Trip-based energy management for electric vehicles: an optimal control approach. In: Proceedings of the 2013 American control conference (ACC 2013). IEEE, Washington, pp 5998–6003
6. Boehme TJ, Schori M, Frank B, Schultalbers M, Drewelow W (2013) A predictive energy management for hybrid vehicles based on optimal control theory. In: Proceedings of the 2013 American control conference (ACC 2013). IEEE, Washington, pp 6004–6009
7. Boehme TJ, Sehnke T, Schultalbers M, Jeinsch T (2014) Implementation of an optimal control like energy management for hybrid vehicles based on driving profiles. In: SAE world congress, Technical paper 2014-01-1903. doi:10.4271/2014-01-1903
8. Borhan HA, Vahidi A, Phillips AM, Kuang ML (2009) Predictive energy management of a power-split hybrid electric vehicle. In: Proceedings of the 2009 American control conference. IEEE, St. Louis, pp 3970–3976
9. Carvalho A, Lefèvre S, Schildbach G, Kong J, Borrelli F (2015) Automated driving: the role of forecasts and uncertainty—a control perspective. Eur J Control 24:14–32
10. Chen H, Allgöwer F (1998) A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. Automatica 34(10):1205–1217
11. Farrokhi M, Mohebbi M (2005) Optimal fuzzy control of parallel hybrid electric vehicles. In: ICCAS2005, pp 2–5
12. Ferreau HJ, Bock HG, Diehl M (2008) An online active set strategy to overcome the limitations of explicit MPC. Int J Robust Nonlinear Control 18(8):816–830
13. Findeisen R, Imsland L, Allgöwer F, Foss B (2003) Towards a sampled-data theory for nonlinear model predictive control. In: New trends in nonlinear dynamics and control and their applications. Springer, pp 295–311
14. Fontes FA (2001) A general framework to design stabilizing nonlinear model predictive controllers. Syst Control Lett 42(2):127–143
15. Gipps PG (1981) A behavioural car-following model for computer simulation. Transp Res Part B Methodol 15:105–111
16. Gong Q, Li Y, Peng ZR (2008) Computationally efficient optimal power management for plug-in hybrid electric vehicles based on spatial-domain two-scale dynamic programming. In: IEEE international conference on vehicular electronics and safety, ICVES 2008, pp 90–95

17. Gong Q, Tulpule P, Marano V, Midlam-Mohler S, Rizzoni G (2011) The role of ITS in PHEV performance improvement. In: Proceedings of the 2011 American control conference on O'Farrell Street. IEEE, San Francisco, pp 119–124
18. de Jager B, van Keulen T, Kessels J (2013) Optimal control of hybrid vehicles. Springer
19. Johannesson L, Åsbogård M, Egardt B (2007) Assessing the potential of predictive control for hybrid vehicle powertrains using stochastic dynamic programming. IEEE Trans Intell Transp Syst 8(1):71–83
20. Johannesson L, Petterson S, Egardt B (2009) Predictive energy management of a 4QT series-parallel hybrid electric bus. Control Eng Pract 17:1440–1453
21. Karbowski D, Rousseau A, Pagerit S, Sharer P (2006) Plug-in vehicle control strategy: from global optimization to real time application. In: 22nd electric vehicle symposium, EVS22, Yokohama, Japan
22. Karbowski D, Pagerit S, Calkins A (2012) Energy consumption prediction of a vehicle along a user-specified real-world trip. In: Proceedings from the electric vehicle symposium (EVS26)
23. Katsargyri GE, Kolmanovsky I, Michelini J, Kuang M, Phillips A, Rinehart M, Dahleh M (2009) Path dependent receding horizon control policies for hybrid electric vehicles. In: 2009 IEEE control applications, (CCA) & intelligent control, (ISIC). IEEE, pp 607–612
24. Katsargyri GE, Kolmanovsky IV, Michelini J, Kuang ML, Phillips AM, Rinehart M, Dahleh MA (2009) Optimally controlling hybrid electric vehicles using path forecasting. In: Proceedings of the 2009 American control conference. IEEE, pp 4613–4617
25. Kermani S, Delprat S, Guerra T, Trigui R, Jeanneret B (2012) Predictive energy management for hybrid vehicle. Control Eng Pract 20:408–420
26. Khalil HK, Grizzle J (1996) Nonlinear systems, vol 3. Prentice hall, New Jersey
27. Lee D, Cha S, Rousseau A, Kim N, Karbowski D (2012) Optimal control strategy for PHEVs using prediction of future driving schedule. In: Proceedings of the 26th electric vehicle symposium
28. Lin CC, Peng H (2002) Control of a hybrid electric truck based on driving pattern recognition. In: Proceedings of the 2002 advanced vehicle control conference, Hiroshima, Japan
29. Lin CC, Kang JM, Grizzle JW, Peng H (2001) Energy management strategy for a parallel hybrid electric truck. In: Proceedings of the 2001 American control conference, Arlington, vol 4, pp 2878–2883
30. Lin CC, Jeon S, Peng H, Moo Lee J (2004) Driving pattern recognition for control of hybrid electric trucks. Veh Syst Dyn 42(1–2):41–58
31. Lin CC, Peng H, Grizzle J (2004) A stochastic control strategy for hybrid electric vehicles. In: Proceedings of the 2004 American control conference, vol 5. IEEE, pp 4710–4715
32. Liu J, Peng H (2008) Modeling and control of a power-split hybrid vehicle. IEEE Trans Control Syst Technol 16:1242–1251
33. Moura SJ, Fathy HK, Callaway DS, Stein JL (2011) A stochastic optimal control approach for power management in plug-in hybrid electric vehicles. IEEE Trans Control Syst Technol 19. doi:10.1109/TCST.2010.2043736
34. Musardo C, Rizzoni G, Staccia B (2005) A-ECMS: An adaptive algorithm for hybrid electric vehicle energy management. In: Proceedings of the 44th IEEE conference on decision and control, pp 1816–1823
35. Paganelli G, Ercole G, Brahma A, Guezennec Y, Rizzoni G (2001) General supervisory control policy for the energy optimization of charge-sustaining hybrid electric vehicles. JSAE Rev 22(4):511–518
36. Ranjitkar P, Nakatsuji T, Kawamua A (2005) Car-following models: an experiment based benchmarking. J Eastern Asia Soc Transp Stud 6:1582–1596
37. Ress C, Balzer D, Bracht A, Durekovic S, Loewenau J (2013) Adasis protocol for advanced in-vehicle applications. http://www.ertico.com/assets/pdf/ADASISv2-ITS-NY-Paper-Finalv4.pdf
38. Sampathnarayanana B, Onori S, Yurkovich S (2014) An optimal regulation strategy with disturbance rejection for energy management of hybrid electric vehicles. Automatica 50:128–140

39. Schori M, Boehme TJ, Frank B, Schultalbers M (2013) Calibration of parallel hybrid vehicles based on hybrid optimal control theory. In: 9th IFAC symposium on nonlinear control systems (NOLCOS), Toulouse, pp 475–480
40. Schori M, Boehme TJ, Frank B, Schultalbers M (2013) Solution of a hybrid optimal control problem for a parallel hybrid vehicle. In: 7th IFAC symposium on advances in automotive control (AAC), Tokyo, pp 109–114
41. Schori M, Boehme TJ, Frank B, Lampe B (2014) Optimal calibration of map-based energy management for plug-in parallel hybrid configurations: a hybrid optimal control approach. IEEE Trans Veh Technol 64(9):3897–3907. doi:10.1109/TVT.2014.2363877
42. Schori M, Boehme TJ, Jeinsch T, Schultalbers M (2015) A robust predictive energy management for plug-in hybrid vehicles based on hybrid optimal control theory. In: Proceedings of the 2015 American control conference (ACC). IEEE, Palmer House Hilton, Chicago, pp 2278–2283. doi:10.1109/ACC.2015.7171072
43. Schouten N, Salman M, Kheir N (2002) Fuzzy logic control for parallel hybrid vehicles. IEEE Trans Control Syst Technol 10(3):460–468
44. Schouten NJ, Salman MA, Kheir NA (2003) Energy management strategies for parallel hybrid vehicles using fuzzy logic. Control Eng Pract 11:171–177
45. Serrao L, Onori S, Rizzoni G (2011) A comparative analysis of energy management strategies for hybrid electric vehicles. J Dyn Syst Measur Control 133(3):031012. doi:10.1115/1.4003267
46. Treiber M, Kesting A (2010) Verkehrsdynamik und -simulation: Daten, Modelle und Anwendungen der Verkehrsflussdynamik. Springer, Berlin, Heidelberg
47. Tulpule P, Marano V, Rizzoni G, Mcgee R, Yu H (2011) A statistical approach to assess the impact of road events on PHEV performance using real world data. In: SAE world congress, Technical Paper 2011-01-0875. doi:10.4271/2011-01-0875
48. Wiki O (2013) Srtm2Osm—OpenStreetMap Wiki. http://wiki.openstreetmap.org/w/index.php?title=Srtm2Osm&oldid=936315

Chapter 13
Optimal Design of Hybrid Powertrain Configurations

13.1 Introduction

Traditionally, the structure and parameters of the powertrain of *hybrid electric vehicles* (HEV) and its plug-in derivatives have been optimized first. Then, a control strategy for energy coordination is designed (cf. Guzzella and Sciarretta [10]) based on the hybrid vehicle configuration. In this design philosophy both tasks, structure and parameter optimization and control design, are performed consecutively. In modern vehicle design approaches (cf. Cook et al. [7]), the strong interdependence of the hybrid vehicle system and its powertrain components with the control strategy $\mathcal{K}$ are respected to increase the vehicle design success, which leads to a simultaneous design approach.

The aspect of the strong interdependence can be illustrated by the problem statement as shown in Fig. 13.1. We denote the parameters, which span the *configuration space* of the vehicle designs, as *design parameters*. These design parameters influence the objectives, such as fuel economy, drivability, etc., in a contradictory way. The control strategy $\mathcal{K}$ from Fig. 13.1 depends on the configuration space because different design parameters stretch, compress, or even disjoin the reachable set and makes a design feasible or infeasible.

It is therefore inalienable for a good design process to match the structure and parameter optimization with the design of energy management such that the entire powertrain ensemble achieves its optimal efficiency. This aspect makes the complete design procedure a complex subject. We can only give some hints how to tackle such vast subjects with the optimization methods proposed in the previous chapters. For the sake of simplicity, we make the following assumptions:

- the powertrain structure is fixed to a P2 hybrid powertrain;
- the *internal combustion engine* (ICE) and the battery are preselected; and
- the number of gears are fixed.

Moreover, in order to keep the computational effort manageable the component models must be simple. This is achieved by considering only models with concentrated

T.J. Böhme and B. Frank, *Hybrid Systems, Optimal Control and Hybrid Vehicles*,
Advances in Industrial Control, DOI 10.1007/978-3-319-51317-1_13

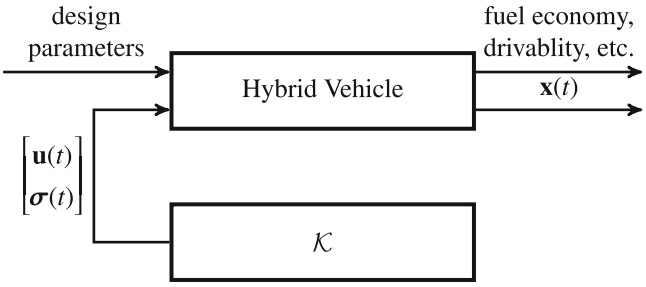


Fig. 13.1 Interdependence of design parameters and energy management

parameters such that FEM computations are avoided. The essential component characteristics are gathered from measurements and mapped to tensor product surfaces (see, e.g., Boehme et al. [6]).

In this chapter, we follow the argumentation of simultaneous optimization of both powertrain configuration and energy management of the P2 HEV system through use of a multi-objective optimization strategy. The studies made in Boehme et al. [5, 6] are extended by solving batched *optimal control problems* (OCP), which are embedded into a *multi-objective genetic algorithm* (MOGA).

13.2 Process Description

The process model for the hybrid powertrain design in Fig. 13.1 consists of design parameters, continuous-valued controls $\mathbf{u}(\cdot)$, binary controls $\boldsymbol{\sigma}(\cdot)$, continuous states $\mathbf{x}(\cdot)$, and performance outputs, which are described in the next sections.

13.2.1 Drivability Performance Index

One important performance output besides fuel consumption is the drivability performance. The drivability performance of a vehicle can be assessed using a performance index. There are different performance indices thinkable, e.g., the calculation of the acceleration time from 0 to 100 km/h or the top speed or a combination of both. For assessment of a vehicle equipped with a multispeed gearbox the criterion of weighting an achievable acceleration of each gear has shown to give reliable solutions. The weighting factors are normally obtained from drive performance tests assessed by many drivers. We denote this performance index as *subjective drivability performance index* (SDPI) to highlight the subjective character of these factors.

Formally, the SDPI is determined by computing an achievable acceleration at each gear. These selected accelerations are then weighted. One can define different rules to select a value from each acceleration profile. For instant, the maximum achievable

acceleration for each engaged gear. The following rules for determination of an acceleration point at each gear work well in practice. The first selected acceleration is calculated by

$$a_1^{sel} = \max \left(\frac{F_{wh,j}(v) - F_w(v)}{m\tilde{\gamma}_m \left(\mathbf{i}_{gbx}^{[1]} \right)} \right)$$

where $F_w(\cdot)$ is the total friction force, $\tilde{\gamma}_m(\cdot)$ is the mass factor dependent on the engaged gear, v is the vehicle speed, and

$$F_{wh,j}(v) = \left[T_{mg}^{max} \left(\frac{v \cdot i_{t,j}}{r_{wh}} \right) + T_{ice}^{max} \left(\frac{v \cdot i_{t,j}}{r_{wh}} \right) - T_{loss} \left(\frac{v \cdot i_{t,j}}{r_{wh}} \right) \right] \cdot \frac{i_{t,j}}{r_{wh}}$$

is the achievable wheel torque using the installed maximum engine and motor power at the transmission ratio $i_{t,j}$ of the engaged j-th gear. The next selected accelerations are then calculated by

$$\tilde{v}_j = \arg\min_{\forall v} \left(F_{trac}^{max}(v) - F_{wh,j}(v) \right)$$
$$a_j^{sel} = \frac{F_{wh,j}(\tilde{v}_{j-1}) - F_w(\tilde{v}_{j-1})}{m\tilde{\gamma}_m \left(\mathbf{i}_{gbx}^{[j]} \right)}$$

for all $j = 2, \ldots, N_{gbx}$. The maximal traction force $F_{trac}^{max}(\cdot)$ is simply calculated by

$$F_{trac}^{max}(v) = \frac{P_{ice}^{max} + P_{mg}^{max}}{v}.$$

The SDPI is then given by a weighted sum of all selected accelerations

$$\text{SDPI} := \sum_{j=1}^{N_{gbx}} a_j^{sel} \mathbf{c}_{sdpi}^{[j]}$$

where the constant coefficients $\mathbf{c}_{sdpi}$ are determined from a questionnaire of many drivers and therefore different subjective impressions. A different version, where the maximum achievable acceleration in each gear is used, has been presented by Rechs et al. [25].

13.2.2 Design Parameters

A good fuel economy and a high drive performance depend strongly upon a proper choice of the gear ratios. Consequently, the gear steps of the automatic transmission

(10.41) are optimized within certain limits, whereas the highest transmission ratio $i_{t,1}$ is prescribed to fulfill the creep velocity and the maximum road inclination constraints and is thus not considered for the optimization. Then, one obtains the optimization vector for the gear steps as

$$\mathbf{y}_t = \left[\varphi_1,\ \varphi_2,\ \varphi_3,\ \ldots,\ \varphi_{N_{gbx}-1}\right]^T \tag{13.1}$$

where the change of each gear step is constrained to

$$\varphi_j^{min} \leq \varphi_j \leq \varphi_j^{max}$$

where φ_j^{min} and φ_j^{max} is the minimum and maximum step of the j-th gear, respectively. Using the gear steps from (13.1), the transmission ratios can be calculated by

$$\mathbf{i}_t = \left[i_{t,1},\ \frac{i_{t,1}}{\varphi_1},\ \frac{i_{t,2}}{\varphi_2},\ \frac{i_{t,3}}{\varphi_3},\ \frac{i_{t,4}}{\varphi_4},\ \ldots,\ \frac{i_{t,N_{gbx}-1}}{\varphi_{N_{gbx}-1}}\right]^T. \tag{13.2}$$

A proper scaled *motor/generator* (MG) is also very important to obtain a good efficiency/weight compromise. Therefore, the maximum power P_{mg}^{max} and the ratio of base speed over maximum speed b_{mg} are used as design variables for the MG, which gives the optimization vector

$$\mathbf{y}_{mg} = \begin{bmatrix} P_{mg}^{max} \\ b_{mg} \end{bmatrix}.$$

Since only *permanent magnet synchronous machines* (PMSM) are used, it is assumed that the ratio b_{mg} lies in the admissible region (10.21).

13.2.3 Powertrain Dynamics

The powertrain dynamics of the P2 hybrid is based on the model $\mathcal{M}_4$. The state vector consists of

$$\overline{\mathbf{x}}_{[k]} = \begin{bmatrix} \overline{\xi}_{[k]} \\ \overline{\beta}_{[k]} \\ \overline{\vartheta}_{cw}^{[k]} \end{bmatrix}$$

where $\overline{\xi}_{[k]}$, $\overline{\beta}_{[k]}$, and $\overline{\vartheta}_{cw}^{[k]}$ is the state of charge, the fuel consumption, and coolant water temperature, respectively. The model has two continuous-valued controls

$$\overline{\mathbf{u}}_{[k]} = \begin{bmatrix} \overline{\mathbf{T}}_{ice}^{[k]} \\ \overline{\mathbf{T}}_{brk}^{[k]} \end{bmatrix} \tag{13.3}$$

where $\overline{\mathbf{T}}_{brk}^{[k]}$ provides the remaining braking torque if the battery's charging limit or $T_{mg}^{max}(\overline{\boldsymbol{\omega}}_{mg}^{[k]})$ of the MG are reached.

The discrete decisions are the drive modes $\overline{\boldsymbol{\zeta}}_{[k]} \in \{0, 1\}$ and the selected gears $\overline{\boldsymbol{\kappa}}_{[k]} \in \{1, 2, \ldots, 7\}$, which can be aggregated to the discrete state as

$$\overline{\mathbf{q}}_{[k]} = 7 \cdot \overline{\boldsymbol{\zeta}}_{[k]} + \overline{\boldsymbol{\kappa}}_{[k]}.$$

For reasons of the numerical optimization approach, the drive mode state $\overline{\boldsymbol{\zeta}}_{[k]}$ is expressed as two Boolean variables: $\overline{\boldsymbol{\sigma}}_1^{[k]}$ for the hybrid drive mode and $\overline{\boldsymbol{\sigma}}_2^{[k]}$ for the electric drive mode.

For later use in parametric sensitivity studies, we introduce the following parameters:

p_1	battery capacity Q_{bat}	p_9	scaled velocity
p_2	battery resistance R_{bat}	p_{10}	scaled MG power P_{mg}^{max}
p_3	auxiliary power P_{aux}	p_{11}	initial state of charge ξ_0
p_4	vehicle mass m	p_{12}	final state of charge ξ_f
p_5	wheel radius r_{wh}	$p_{13,\ldots,19}$	gear ratios
$p_{6,7,8}$	drag coefficients a_0, a_1, a_2		$i_{t,1}, \ldots, i_{t,7}$.

These parameters build the parameter space $\hat{\mathcal{P}}$ and appear in the system dynamics of model $\mathcal{M}_4$ as well as in the boundary conditions.

The state equations from model $\mathcal{M}_4$ can be discretized by any RK method from Chap. 5. Here, we use an implicit Euler scheme to discretize the state equations. Then, the state of charge is calculated by

$$\begin{aligned} \overline{\boldsymbol{\xi}}_{[k+1]} &= \overline{\boldsymbol{\xi}}_{[k]} + h \cdot \Gamma_{\xi}\left(\overline{\boldsymbol{\xi}}_{[k+1]}, \overline{\mathbf{T}}_{ice}^{[k+1]}, \overline{\boldsymbol{\sigma}}_{[k+1]}\right) \\ &= \overline{\boldsymbol{\xi}}_{[k]} + \sum_{q=1}^{2} \overline{\boldsymbol{\sigma}}_q^{[k+1]} \cdot \frac{h}{p_1} \cdot I_{bat,q}\left(\overline{\boldsymbol{\xi}}_{[k+1]}, \overline{\mathbf{T}}_{ice}^{[k+1]}\right), \end{aligned} \tag{13.4}$$

the fuel rate is calculated by

$$\begin{aligned} \overline{\boldsymbol{\beta}}_{[k+1]} &= \overline{\boldsymbol{\beta}}_{[k]} + h \cdot \Gamma_{\beta}\left(\overline{\boldsymbol{\vartheta}}_{cw}^{[k+1]}, \overline{\mathbf{T}}_{ice}^{[k+1]}, \overline{\boldsymbol{\sigma}}_1^{[k+1]}\right) \\ &= \overline{\boldsymbol{\beta}}_{[k]} + h \cdot \overline{\boldsymbol{\sigma}}_1^{[k+1]} \cdot \gamma_f \cdot CF_{fc}\left(\overline{\boldsymbol{\vartheta}}_{cw}^{[k+1]}\right) \cdot \mathrm{bsfc}\left(\overline{\mathbf{T}}_{ice}^{[k+1]}, \overline{\boldsymbol{\omega}}_{ice}^{[k+1]}\right) \cdot \overline{\mathbf{T}}_{ice}^{[k+1]} \cdot \overline{\boldsymbol{\omega}}_{ice}^{[k+1]}, \end{aligned} \tag{13.5}$$

and the temperature of the coolant water is calculated by

$$\begin{aligned}\overline{\vartheta}_{cw}^{[k+1]} &= \overline{\vartheta}_{cw}^{[k]} + h \cdot \Gamma_{\vartheta_{cw}}\left(\overline{\vartheta}_{cw}^{[k+1]}, \overline{\mathbf{T}}_{ice}^{[k+1]}, \overline{\boldsymbol{\sigma}}_{1}^{[k+1]}\right) \\ &= \overline{\vartheta}_{cw}^{[k]} + h \cdot c_1 \cdot \left[\gamma_{cw} \cdot H_l \cdot \Gamma_{\beta}\left(\overline{\vartheta}_{cw}^{[k+1]}, \overline{\mathbf{T}}_{ice}^{[k+1]}, \overline{\boldsymbol{\sigma}}_{1}^{[k+1]}\right) - \overline{\mathbf{T}}_{ice}^{[k+1]} \cdot \overline{\boldsymbol{\omega}}_{ice}^{[k+1]}\right] \\ &- c_2 \cdot \left[\overline{\vartheta}_{cw}^{[k+1]} - \overline{\vartheta}_{amb}^{[k+1]}\right] \end{aligned} \tag{13.6}$$

where $\Gamma_{\xi}(\cdot)$, $\Gamma_{\beta}(\cdot)$, and $\Gamma_{\vartheta_{cw}}(\cdot)$ are the increment functions for the implicit Euler integration scheme for state of charge, fuel consumption, and coolant water temperature, respectively. H_l is the lower heating value of the fuel and $CF_{fc}(\cdot)$ is the fuel correction for engine warm-up. The parameters c_1, c_2, γ_f, and γ_{cw} include heat capacity and natural constants.

The differential equations (13.4), (13.5), and (13.6) use the following quantities:

$$\overline{\boldsymbol{\omega}}_{gbx}^{[k]} = \frac{p_9 \overline{\mathbf{v}}_{[k]}}{2\pi p_5} \cdot \bar{\mathbf{i}}_t^{[k]} \tag{13.7}$$

$$\overline{\boldsymbol{\omega}}_{mg}^{[k]} = \overline{\boldsymbol{\omega}}_{gbx}^{[k]} \tag{13.8}$$

$$\overline{\boldsymbol{\omega}}_{ice}^{[k]} = \overline{\boldsymbol{\zeta}}_{[k]} \cdot \overline{\boldsymbol{\omega}}_{gbx}^{[k]} \tag{13.9}$$

$$\overline{\mathbf{T}}_{gbx}^{[k]} = \frac{p_5}{\bar{\mathbf{i}}_t^{[k]}} \cdot \left(p_4 \overline{\mathbf{a}}_{[k]} + p_6 + p_7 p_9 \overline{\mathbf{v}}_{[k]} + p_8 \left(p_9 \overline{\mathbf{v}}_{[k]}\right)^2 + \overline{\mathbf{T}}_{loss}^{[k]} + \overline{\mathbf{T}}_{brk}^{[k]}\right) \tag{13.10}$$

$$I_{bat,q}\left(\overline{\boldsymbol{\xi}}_{[k]}, \overline{\mathbf{T}}_{ice}^{[k]}\right) = \frac{-V_{oc}\left(\overline{\boldsymbol{\xi}}_{[k]}\right) + \sqrt{V_{oc}^2\left(\overline{\boldsymbol{\xi}}_{[k]}\right) + 4 p_2 P_{bat,q}\left(\overline{\mathbf{T}}_{ice}^{[k]}\right)}}{2 p_2}$$

$$P_{bat,1}\left(\overline{\mathbf{T}}_{ice}^{[k]}\right) = -P_{mg}\left(\frac{\overline{\mathbf{T}}_{gbx}^{[k]} - \overline{\mathbf{T}}_{ice}^{[k]}}{p_{10}}, \overline{\boldsymbol{\omega}}_{mg}^{[k]}\right) - p_3$$

$$P_{bat,2} = -P_{mg}\left(\frac{\overline{\mathbf{T}}_{gbx}^{[k]}}{p_{10}}, \overline{\boldsymbol{\omega}}_{mg}^{[k]}\right) - p_3$$

$$\bar{\mathbf{i}}_t^{[k]} \in \left[p_{13}, p_{14}, p_{15}, p_{16}, p_{17}, p_{18}, p_{19}\right]^T$$

where $\overline{\mathbf{a}}$ and $\overline{\mathbf{v}}$ are the discretized vehicle acceleration and speed, respectively.

13.3 Multi-objective Powertrain Design

The design targets for a charge-sustaining HEV, which are commonly employed, can essentially look like this (Ehsani et al. [9]):

1. satisfying the performance requirements (gradability, high traction force, maximum cruising speed, and top speed);
2. recovering brake energy as much as possible;

3. achieving high fuel economy;
4. achieving low emissions;
5. maintaining the state of charge at reasonable levels; and
6. operating range of each components should not be exceeded.

Classifying these design targets reveals the fact that design target 1 depends on the design parameters of the vehicle, such as MG rated power and gear steps, whereas targets 2–5 depend on the design parameters and the energy control strategy. These not surprising facts will help us to structure the optimization algorithm in such a way that design target 1 can be mastered independently from the energy control design. For the other 5 design targets, this simplification does not apply. Furthermore, the design targets 2 and 3 can be summarized to one goal, namely, saving energy. The design goal 5 represents equality constraints and the design target 6 represents inequality constraints. The design goal 4 is very demanding and requires additional complex dynamics as shown in Sect. 10.5.2. We skip this goal in the presentation of this chapter, but make some comments in Sect. 13.6 to deal with such problems.

Powertrain designs with respect to fuel economy only result in gear sizings with small gear ratios at high gears, which are beneficial for low fuel consumption but lead to poor drive performance. This suboptimal drive performance can be prevented by describing the traction force requirement from design target 1 as SDPI. The optimization problem has now two objectives, minimizing the fuel consumption and maximizing the subjective drivability performance index, which are contradictory criteria and therefore no single optimum can be found. This inevitably leads to a multi-objective optimization problem and the best achievable performance is characterized by a Pareto front. The maximum cruise speed and top speed goals can be fulfilled by fixing the first and the last gear to their required values (cf. Sect. 10.3.3.1).

A MOGA is often applied in numerical optimization practice to find simultaneously the best parameters for components and energy management (Cook et al. [7]). This is mainly motivated by the fact that evolutionary algorithms can cope naturally with multi-objective, discontinuous, and non-differentiable problems (cf. Sect. 2.5.1). In this chapter, a combination of MOGA and optimal control algorithms for switched systems is proposed for the simultaneous optimization of design parameters and energy management controls. The structure of this approach consists of a master problem and associated *switched optimal control subproblems* (SOCP). In the master problem, the design parameters are varied and sorted by the NSGA-II algorithm (Deb et al. [8]). The energy control problem is then casted into a batch of SOCPs to generate more and more realistic optimal control and state trajectories. We denote this optimization approach as *multi-objective powertrain design* (MOPD). Since new design parameters imply a correction of the vehicle weight and efficiency maps, the optimization strategy includes the steps of sizing and mass correction of the powertrain components.

The proposed optimization procedure assumes that the test cycle is perfectly known. This assumption is not realistic as has been shown in Chap. 12 and implies that we find for each vehicle configuration always the best possible energy management strategy $\mathcal{K}$. It is certainly possible to generate automatically predictive energy

management strategies with the methods demonstrated in Chaps. 11 and 12. But this approach is not pursued.

13.3.1 Master Problem

The master problem coordinates besides searching for optimal design parameters different subtasks: component scaling, vehicle mass correction, and energy management design.

The design parameters for a P2 hybrid powertrain are the rated power and the speed ratio of the MG and the gear steps of the transmission. The ICE and battery are preselected and kept fixed. Then, the constrained master problem can be stated as follows: find the optimal design parameters

$$\mathbf{y} = \begin{bmatrix} \mathbf{y}_{mg} \\ \mathbf{y}_t \end{bmatrix}, \tag{13.11}$$

such that the static multi-objective function is minimized

$$\mathbf{f}(\mathbf{y}) = \begin{bmatrix} \overline{\boldsymbol{\beta}}_{[N_t]} \\ -\text{SDPI} \end{bmatrix}$$

subject to the design inequality constraints $\mathbf{g}(\mathbf{y}) \leq \mathbf{0}$, which restrict the parameters to the ranges listed in Table 13.1.

13.3.2 Map Scaling for Powertrain Components

The electrical and mechanical component characteristics are scaled by linear or polynomial relationships.

In case of sizing of the MG, the electrical input power is obtained by linear scaling of the torque axis. The basis for the scaling procedure is a template map

Table 13.1 Powertrain design constraints

Constraints	Range
Rated power P_{mg}^{max} of MG (kW)	8–40
Speed ratio b_{mg} of MG (–)	0.2–0.6
Gear step φ_1 (–)	1.60–2.10
Gear step φ_2 (–)	1.55–1.85
Gear step φ_3 (–)	1.40–1.60
Gear step φ_4 (–)	1.20–1.50
Gear step φ_5 (–)	1.12–1.50
Gear step φ_6 (–)	1.12–1.45

of the master efficiency, which is typically generated from measurements collected from a test bench. The grid axis for speed and torque of the master template are defined by

$$\mathcal{G}_{x,mg} := \left\{x^1, x^2, \ldots, x^{\#\mathcal{G}_{x,mg}}\right\},$$

and

$$\mathcal{G}_{y,mg} := \left\{y^1, y^2, \ldots, y^{\#\mathcal{G}_{y,mg}}\right\},$$

respectively. Then, the values of the electrical input power at the grid points $\mathcal{G}_{x,mg}$ and $\mathcal{G}_{y,mg}$ are represented by

$$\mathcal{G}_{z,mg} = \left\{z^{1,1}, \ldots, z^{2,1}, \ldots, z^{\#\mathcal{G}_{z,mg}}\right\}.$$

Firstly, the maximum torque $T_{mg}^{max}(\cdot)$ and the minimum torque $T_{mg}^{min}(\cdot)$, which envelopes the electrical input power, are calculated by

$$T_{mg}^{max}(\omega_{mg}) = \frac{P_{mg}^{max}}{\omega_{mg}}$$

and

$$T_{mg}^{min}(\omega_{mg}) = -\frac{P_{mg}^{max}}{\omega_{mg}},$$

respectively, for all $\omega_{mg} \in \mathcal{G}_{x,mg}$. The torques must be restricted to the maximal and minimal achievable torques by

$$T_{mg}^{max}\left(\omega_{mg}\right) = \min\left(T_{mg}^{max}(\omega_{mg}), \frac{P_{mg}^{max}}{\omega_{mg}^{base}}\right)$$

and

$$T_{mg}^{min}\left(\omega_{mg}\right) = \max\left(T_{mg}^{min}(\omega_{mg}), -\frac{P_{mg}^{max}}{\omega_{mg}^{base}}\right),$$

respectively, for all $\omega_{mg} \in \mathcal{G}_{x,mg}$, where

$$\omega_{mg}^{base} = b_{mg} \cdot \omega_{mg}^{max}$$

is the base speed of the MG and b_{mg} is the speed ratio. The reader should note, that the maximum power P_{mg}^{max} and the speed ratio b_{mg} are design variables from Sect. 13.2.2, which are passed from the master problem to the scaling procedure.

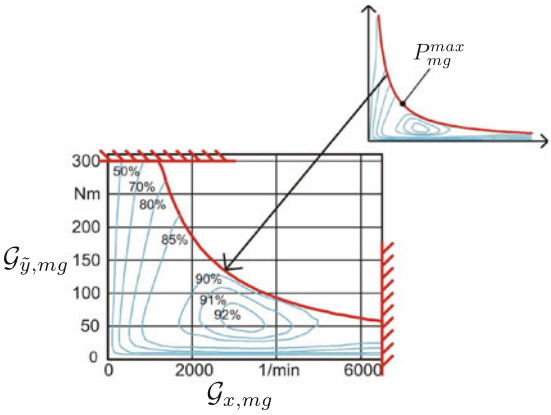


Fig. 13.2 Principle of the map scaling. The *upper right picture* shows the unscaled efficiency for the new P_{mg}^{max}, where the master template is used as basis for the scaling. The *lower left picture* shows the torque proportional scaled efficiencies with $\boldsymbol{\eta}_{mg}^{[i],[j]} = \gamma_p \cdot x^i \cdot \tilde{y}^j / \tilde{z}^{i,j}$ on the grids $\mathcal{G}_{x,mg}$ and $\mathcal{G}_{\tilde{y},mg}$

Secondly, the torque grid axis are scaled linearly by

$$\tilde{y}^j = \frac{\max\limits_{\forall \omega_{mg}} \left\{ T_{mg}^{max}(\omega_{mg}) \right\}}{T_{orig}^{max}} \cdot y^j, \quad \forall y^j \in \mathcal{G}_{y,mg}$$

where T_{orig}^{max} is the maximum torque of the master MG and $\tilde{y}^j \in \mathcal{G}_{\tilde{y},mg}$ is the new scaled axis entry.

Calculating the stationary efficiency of the MG yields

$$\boldsymbol{\eta}_{mg}^{[i],[j]} = \frac{\gamma_p \cdot x^i \cdot y^j}{z^{i,j}}, \quad \forall x^i \in \mathcal{G}_{x,mg}, \ \forall y^j \in \mathcal{G}_{y,mg}, \ \text{and}, \forall z^{i,j} \in \mathcal{G}_{z,mg}$$

where γ_p is a constant to obtain the same unit of $\mathcal{G}_{z,mg}$. Then, the new electrical input power is computed on the scaled torque grid as

$$\tilde{z}^{i,j} = \frac{\gamma_p \cdot x^i \cdot \tilde{y}^j}{\boldsymbol{\eta}_{mg}^{[i],[j]}}, \quad \forall x^i \in \mathcal{G}_{x,mg} \text{ and } \forall \tilde{y}^j \in \mathcal{G}_{\tilde{y},mg}. \tag{13.12}$$

The principle of the map scaling is illustrated in Fig. 13.2.

This simple map scaling procedure can lead to some distortion at the field weakening region. If these distortions cannot be ignored, then the scaling procedure must split the speed axis $\mathcal{G}_{x,mg}$ into the regions of constant torque and field weakening. The electrical input power in these regions must then be scaled differently.

In other cases, where some characteristics of the new electrical input power are not preserved, one can test if the power loss due to friction, heat loss, etc., do not scale well with the power. If this is true, one can refine (13.12) by employing the affine Willans method with

$$\gamma_p \cdot \tilde{x}^i \cdot \tilde{y}^j = \boldsymbol{\eta}_{mg}^{[i],[j]} \cdot \tilde{z}^{i,j} - P_0, \quad \forall \tilde{x}^i \in \mathcal{G}_{\tilde{x},mg},\ \forall \tilde{y}^j \in \mathcal{G}_{\tilde{y},mg},\ \text{and, } \tilde{z}^{i,j} \in \mathcal{G}_{\tilde{z},mg}$$

where P_0 is the aggregated power loss after the energy conversion and $\mathcal{G}_{\tilde{x},mg}$ is the new scaled speed axis.

The mass of the MG can be scaled polynomially (cf. Boehme et al. [4]) with an empirically found relationship

$$m_{mg} = a \cdot \sqrt{P_{mg}^{max} \cdot T_{mg}^{max}} + b$$

where a and b are constant coefficients determined from an analysis of a series of PMSMs.

The transmission size used in this work is fixed to a seven-speed dual-clutch gearbox. However, if transmission resizing is desired, the efficiency maps for each gear can be scaled by the same procedure.

Since gradients in x and y directions are needed, tensor product surfaces are used to approximate the efficiency maps.

13.3.3 Batched Optimal Control Subproblems

The energy management problem for finding the optimal fuel consumption in powertrain configuration studies is defined as follows:

$$\mathcal{P}_9 := \begin{cases} \min\limits_{\overline{\mathbf{q}},\,\overline{\mathbf{u}}} & \overline{\boldsymbol{\beta}}_{[N_t]} & & \\ \text{subject to} & \text{(13.3), (13.4), (13.5), and (13.6)} & & \\ \overline{\boldsymbol{\xi}}_{[0]} & = p_{11} & & \\ \overline{\boldsymbol{\xi}}_{[N_t]} & = p_{12} & & \\ \overline{\boldsymbol{\beta}}_{[0]} & = 0 & & \\ \overline{\boldsymbol{\vartheta}}_{cw}^{[0]} & = 293\ (\mathrm{K}) & & \\ \mathbf{c}_{\overline{u}}\left(\overline{\mathbf{u}}_{[k]}\right) & \leq \mathbf{0}, & k = 0, \ldots, N_t & \\ \mathbf{c}_{\overline{x}}\left(\overline{\boldsymbol{\xi}}_{[k]}\right) & \leq \mathbf{0}, & k = 1, \ldots, N_t - 1 & \\ \overline{\boldsymbol{\zeta}}_{[k]} & \in \{0, 1\}, & k = 0, \ldots, N_t & \\ \overline{\boldsymbol{\kappa}}_{[k]} & \in \{1, \ldots, 7\}, & k = 0, \ldots, N_t. & \end{cases} \tag{13.13}$$

If test cycles with many discretization points are used as fixed inputs for the SOCP, the problem becomes prohibitive for the *branch-and-bound* (BB), Two-Stage method, and *dynamic programming* (DP). Indirect shooting must also be excluded because of active state constraints, which is likely due to the scaling of the rated power of the MG. A promising optimization method for this problem type is direct collocation since it can easily deal with state constraints. Problems with many discretization points require a sparse SQP solver implementation from Chap. 9.

In order to obtain reliable solutions, some constraints are applied stepwise. This leads to a methodology, which solves a sequence of individual optimal control problems (called batched optimal control problems).

First Reformulation

We start by reformulating the original problem $\mathcal{P}_9$ as *binary switched optimal control problem* (BSOCP). However, expressing the complete discrete state ($N_q = 14$) as Boolean variables needs a lot of coding. A trick to avoid a BSOCP with a high number of binary decisions is to approximate the discrete gear ratio as a continuous-valued gear ratio, i.e.,

$$\tilde{\mathbf{i}}_t^{[k]} \in \left[i_{t,7}, i_{t,1}\right].$$

This relaxation mimics a *continuously variable transmission* (CVT) behavior and allows to control the gear ratio as a continuous-valued control that enlarges the control vector to

$$\overline{\mathbf{u}}_{[k]} = \begin{bmatrix} \overline{\mathbf{T}}_{ice}^{[k]} \\ \overline{\mathbf{T}}_{brk}^{[k]} \\ \tilde{\mathbf{i}}_t^{[k]} \end{bmatrix}. \tag{13.14}$$

The discrete state reduces to $\overline{\mathbf{q}}_{[k]} \in \{1, 2\}$, which are then only two Boolean variables. These Boolean variables $\overline{\boldsymbol{\sigma}}_{[k]} \in \{0, 1\}^2$ are relaxed such that $\overline{\hat{\boldsymbol{\sigma}}}_{[k]}$ may take values from the compact set $[0, 1]^2$. The enhanced control vector is then defined as

$$\overline{\boldsymbol{\rho}}_{[k]} = \begin{bmatrix} \overline{\mathbf{u}}_{[k]} \\ \overline{\hat{\boldsymbol{\sigma}}}_{[k]} \end{bmatrix}. \tag{13.15}$$

The *embedded optimal control problem* (EOCP) can then be formulated based on the time grid $\mathcal{G}_t$ as:

$$\mathcal{P}_{10} := \begin{cases} \phi\left(\overline{\boldsymbol{\rho}}^*\right) = \min\limits_{\overline{\boldsymbol{\rho}}} \ \overline{\hat{\boldsymbol{\beta}}}_{[N_t]} + \ \gamma_\sigma \sum\limits_{k=0}^{N_t} \prod\limits_{q=1}^{2} \overline{\hat{\sigma}}_q^{[k]} & \\ \text{subject to} & \text{(13.4), (13.5), (13.6), (13.14), and (13.15)} \\ \overline{\hat{\boldsymbol{\xi}}}_{[0]} & = p_{11} \\ \overline{\hat{\boldsymbol{\xi}}}_{[N_t]} & = p_{12} \\ \overline{\hat{\boldsymbol{\beta}}}_{[0]} & = 0 \\ \overline{\hat{\vartheta}}_{cw}^{[0]} & = 293\ (\mathrm{K}) \\ \mathbf{c}_{\overline{\rho}}\left(\overline{\boldsymbol{\rho}}_{[k]}\right) & \leq \mathbf{0}, \qquad k = 0, \ldots, N_t \\ \mathbf{c}_{\overline{\tilde{x}}}\left(\overline{\tilde{\boldsymbol{\xi}}}_{[k]}\right) & \leq \mathbf{0}, \qquad k = 1, \ldots, N_t - 1 \\ \sum\limits_{q=1}^{2} \overline{\hat{\boldsymbol{\sigma}}}_q^{[k]} & = 1, \qquad k = 0, \ldots, N_t \end{cases} \tag{13.16}$$

where the state and control constraints are defined as

$$\mathbf{c}_{\overline{\tilde{x}}}\left(\overline{\hat{\boldsymbol{\xi}}}_{[k]}\right) := \begin{bmatrix} \overline{\hat{\boldsymbol{\xi}}}_{[k]} - \xi^{max} \\ \xi^{min} - \overline{\hat{\boldsymbol{\xi}}}_{[k]} \end{bmatrix} \tag{13.17}$$

and

$$\mathbf{c}_{\overline{p}}\left(\overline{\boldsymbol{\rho}}_{[k]}\right) := \left[\begin{array}{c} \overline{\hat{\boldsymbol{\sigma}}}_1^{[k]} \cdot \overline{\mathbf{T}}_{ice}^{[k]} - T_{ice}^{max}\left(\overline{\boldsymbol{\omega}}_{ice}^{[k]}\right) \\ \overline{\mathbf{T}}_{gbx}^{[k]} - \overline{\hat{\boldsymbol{\sigma}}}_1^{[k]} \cdot \overline{\mathbf{T}}_{ice}^{[k]} - p_{10} \cdot T_{mg}^{max}\left(\overline{\boldsymbol{\omega}}_{mg}^{[k]}\right) \\ p_{10} \cdot T_{mg}^{min}\left(\overline{\boldsymbol{\omega}}_{mg}^{[k]}\right) - \overline{\mathbf{T}}_{gbx}^{[k]} + \overline{\hat{\boldsymbol{\sigma}}}_1^{[k]} \cdot \overline{\mathbf{T}}_{ice}^{[k]} \\ \overline{\boldsymbol{\omega}}_{gbx}^{[k]} - \omega_{gbx}^{max} \\ \omega_{gbx}^{min} - \overline{\boldsymbol{\omega}}_{gbx}^{[k]} \\ \hline T_{ice}^{min} - \overline{\mathbf{T}}_{ice}^{[k]} \\ \overline{\mathbf{T}}_{brk}^{[k]} - T_{brk}^{max} \\ T_{brk}^{min} - \overline{\mathbf{T}}_{brk}^{[k]} \\ \bar{\tilde{\mathbf{i}}}_t^{[k]} - i_{t,1} \\ i_{t,7} - \bar{\tilde{\mathbf{i}}}_t^{[k]} \\ -\overline{\hat{\boldsymbol{\sigma}}}_{[k]} \\ \overline{\hat{\boldsymbol{\sigma}}}_{[k]} - \mathbb{1}_{2\times 1} \end{array}\right], \tag{13.18}$$

respectively. The fixed trajectories $\overline{\boldsymbol{\omega}}_{gbx}$, $\overline{\boldsymbol{\omega}}_{mg}$, $\overline{\boldsymbol{\omega}}_{ice}$, and $\overline{\mathbf{T}}_{gbx}$ are obtained from (13.7), (13.8), (13.9), and (13.10), respectively, and depend on the selected test cycle. The speed dependent engine drag torque is approximated by a constant torque $T_{ice}^{min}(\overline{\boldsymbol{\omega}}_{ice}^{[k]}) \equiv T_{ice}^{min}$. For numerical reasons T_{ice}^{min} is set to a small positive value to ensure that a turned off engine is uniquely determined by $\overline{\hat{\boldsymbol{\sigma}}}_1^{[k]}$ and not by $\overline{\mathbf{T}}_{ice}^{[k]} = 0$. This is important for the coherent calculation of the dynamic system and the constraints. The minimum values T_{ice}^{min}, ξ^{min}, ω_{gbx}^{min}, and T_{brk}^{min} and the maximum values ξ^{max}, ω_{gbx}^{max}, and T_{brk}^{max} are fixed, whereas the minimum value $T_{mg}^{min}(\cdot)$ and the maximum values $T_{mg}^{max}(\cdot)$ and $T_{ice}^{max}(\cdot)$ depend on the speeds at time instant k and therefore depend on the gear ratio $\bar{\tilde{\mathbf{i}}}_t$. The control constraints $\mathbf{c}_{\overline{u}}(\cdot)$ remain autonomous. The constraints below the horizontal line are implemented as box constraints. The lower speed limit ω_{gbx}^{min} corresponds to the idle speed limit.

The EOCP is transcribed by a direct collocation method into a NLP and solved using the sparse SQP method. The time grid $\mathcal{G}_t$ is chosen to be equidistant. The step length h must be chosen to tackle the compromise between accuracy and computing time.

In order to obtain a bang-bang solution for the relaxed binary controls an iterative penalization procedure for the relaxed binary controls has been applied. The term

$$\gamma_\sigma \sum_{k=0}^{N_t} \prod_{q=1}^{2} \overline{\hat{\boldsymbol{\sigma}}}_q^{[k]}$$

penalizes non-integer values and is added to the Mayer cost function. Problem $\mathcal{P}_{10}$ is solved several times with a steadily increasing weighting factor $\gamma_\sigma = \gamma_{\sigma,0}\gamma_{\sigma,i}^{inc}$, where $\gamma_{\sigma,0}$ is a constant weighting factor. For the first NLP solution the factor $\gamma_{\sigma,1}^{inc}$ is set to zero, such that only the weaker condition

$$\sum_{q=1}^{2} \overline{\hat{\boldsymbol{\sigma}}}_q^{[k]} = 1$$

for $k = 0, \ldots, N_t$ applies. If no binary feasible solution can be found within a maximum number of iterations, then a sum-up rounding strategy is applied.

Second Reformulation

The second reformulation uses the integer-valued gear sequence $\overline{\boldsymbol{\kappa}}$ obtained by rounding of the continuous-valued trajectory to the nearest gear ratio of (13.2) found by NSGA-II as fixed control values for the EOCP. This has to be done with caution because brute-force rounding can result in some serious violations of the constraints and the procedure of the consecutive OCPs might fail. A common problem is the nonachievement of the minimum engine speed. A remedy for this problem is a simple gear down shifting if the lowest gear is not already applied. Otherwise, the engine must be switched off, i.e., $\overline{\boldsymbol{\sigma}}_1^{[k]} = 0$ and $\overline{\boldsymbol{\sigma}}_2^{[k]} = 1$.

By the usage of the integer-valued gear sequence as a fixed control the continuous-valued control vector is reduced to (13.3).

The time grid $\mathcal{G}_t$ can be chosen identically to problem formulation (13.16)–(13.18). Then, the second reformulation yields the EOCP based on the time grid $\mathcal{G}_t$ by

$$\mathcal{P}_{11} := \begin{cases} \phi\left(\overline{\boldsymbol{\rho}}^*\right) = \min\limits_{\overline{\boldsymbol{\rho}}} \ \overline{\hat{\boldsymbol{\beta}}}_{[N_t]} + \ \gamma_\sigma \sum\limits_{k=0}^{N_t} \prod\limits_{q=1}^{2} \overline{\hat{\boldsymbol{\sigma}}}_q^{[k]} & \\ \text{subject to} & \text{(13.3), (13.4), (13.5), (13.6), and (13.15)} \\ \overline{\hat{\boldsymbol{\xi}}}_{[0]} & = p_{11} \\ \overline{\hat{\boldsymbol{\xi}}}_{[N_t]} & = p_{12} \\ \overline{\hat{\boldsymbol{\beta}}}_{[0]} & = 0 \\ \overline{\hat{\boldsymbol{\vartheta}}}_{cw}^{[0]} & = 293\ (\mathrm{K}) \\ \mathbf{c}_{\overline{p}}\left(\overline{\boldsymbol{\rho}}_{[k]}\right) & \leq \mathbf{0}, \qquad k = 0, \ldots, N_t \\ \mathbf{c}_{\overline{\hat{x}}}\left(\overline{\hat{\boldsymbol{\xi}}}_{[k]}\right) & \leq \mathbf{0}, \qquad k = 1, \ldots, N_t - 1 \\ \sum\limits_{q=1}^{2} \overline{\hat{\boldsymbol{\sigma}}}_q^{[k]} & = 1, \qquad k = 0, \ldots, N_t \end{cases} \tag{13.19}$$

with the state constraints (13.17) and the control constraints

$$\mathbf{c}_{\overline{p}}\left(\overline{\boldsymbol{p}}_{[k]}\right) := \begin{bmatrix} \overline{\mathbf{T}}_{ice}^{[k]} - T_{ice}^{max}\left(\overline{\boldsymbol{\omega}}_{ice}^{[k]}\right) \\ \overline{\mathbf{T}}_{gbx}^{[k]} - \overline{\hat{\boldsymbol{\sigma}}}_{1}^{[k]} \cdot \overline{\mathbf{T}}_{ice}^{[k]} - p_{10} \cdot T_{mg}^{max}\left(\overline{\boldsymbol{\omega}}_{mg}^{[k]}\right) \\ \underline{p_{10} \cdot T_{mg}^{min}\left(\overline{\boldsymbol{\omega}}_{mg}^{[k]}\right) - \overline{\mathbf{T}}_{gbx}^{[k]} + \overline{\hat{\boldsymbol{\sigma}}}_{1}^{[k]} \cdot \overline{\mathbf{T}}_{ice}^{[k]}} \\ T_{ice}^{min} - \overline{\mathbf{T}}_{ice}^{[k]} \\ \overline{\mathbf{T}}_{brk}^{[k]} - T_{brk}^{max} \\ T_{brk}^{min} - \overline{\mathbf{T}}_{brk}^{[k]} \\ -\overline{\hat{\boldsymbol{\sigma}}}_{[k]} \\ \overline{\hat{\boldsymbol{\sigma}}}_{[k]} - \mathbb{1}_{2\times 1} \end{bmatrix}. \tag{13.20}$$

Again, the continuous optimal control problem is transcribed by a direct collocation method into a NLP and solved using the SQP method.

Third Reformulation

In order to reduce the number of switching arcs for the drive mode control command and the gear shifting command, problem $\mathcal{P}_{11}$ is reformulated to a *switching time optimization* (STO) with main and minor time grids.

According to Sect. 8.3.3, the main time grid

$$\mathcal{G}_{t_2} := \left\{T_1, T_2, \ldots, T_{N_{t_2}}\right\}$$

contains fixed intervals with the length of 10 s and time instances which must not be moved. Such time phases, e.g., the catalytic converter heating, must be known in advance to fix these intervals. The minor time grid

$$\mathcal{G}_{t_3} := \left\{t_0, t_1, \ldots, t_{N_{t_3}}\right\}$$

contains movable time instances and is based on the switching arcs of the drive mode binary commands $\overline{\hat{\boldsymbol{\sigma}}}$ and the gear shifting commands $\overline{\hat{\boldsymbol{\kappa}}}$.

One obtains state and control functions, which are defined on the time grid $\mathcal{G}_{t_3}$

$$\begin{aligned} \tilde{\xi}_j(\tau) &:= \xi(t_j + \tau\,\varsigma_j), \\ \tilde{\beta}_j(\tau) &:= \beta(t_j + \tau\,\varsigma_j), \\ \tilde{\vartheta}_{cw,j}(\tau) &:= \vartheta_{cw}(t_j + \tau\,\varsigma_j), \\ \tilde{\mathbf{u}}_j(\tau) &:= \mathbf{u}(t_j + \tau\,\varsigma_j), \\ \tilde{\boldsymbol{\sigma}}_j(\tau) &:= \boldsymbol{\sigma}(t_j + \tau\,\varsigma_j), \end{aligned}$$

for $j = 0, \ldots, N_{t_3} - 1$ and $\tau \in [0, 1]$. Then, these functions are discretized on the time grid $\mathcal{G}_t$ individually for each switching arc ς_j, which yield

$$\begin{aligned}\overline{\tilde{\boldsymbol{\xi}}}_j^{[k+1]} &= \overline{\tilde{\boldsymbol{\xi}}}_j^{[k]} + h \cdot \varsigma_j \cdot \Gamma_{\tilde{\xi}} \left(\overline{\tilde{\boldsymbol{\xi}}}_j^{[k+1]}, \overline{\tilde{\mathbf{T}}}_{ice,j}^{[k+1]}, \overline{\tilde{\boldsymbol{\sigma}}}_j^{[k+1]} \right) \\ &= \overline{\tilde{\boldsymbol{\xi}}}_j^{[k]} + h \cdot \varsigma_j \cdot \left[\sum_{q=1}^{2} \overline{\tilde{\boldsymbol{\sigma}}}_{q,j}^{[k+1]} \cdot \frac{1}{p_1} \cdot I_{bat,q} \left(\overline{\tilde{\boldsymbol{\xi}}}_j^{[k+1]}, \overline{\tilde{\mathbf{T}}}_{ice,j}^{[k+1]} \right) \right], \end{aligned} \tag{13.21}$$

$$\begin{aligned}\overline{\tilde{\boldsymbol{\beta}}}_j^{[k+1]} &= \overline{\tilde{\boldsymbol{\beta}}}_j^{[k]} + h \cdot \varsigma_j \cdot \Gamma_{\tilde{\beta}} \left(\overline{\tilde{\boldsymbol{\vartheta}}}_{cw,j}^{[k+1]}, \overline{\tilde{\mathbf{T}}}_{ice,j}^{[k+1]}, \overline{\tilde{\boldsymbol{\sigma}}}_{1,j}^{[k+1]} \right) \\ &= \overline{\tilde{\boldsymbol{\beta}}}_j^{[k]} + h \cdot \varsigma_j \cdot \\ &\quad \left[\overline{\tilde{\boldsymbol{\sigma}}}_{1,j}^{[k+1]} \cdot \gamma_f \cdot CF_{fc} \left(\overline{\tilde{\boldsymbol{\vartheta}}}_{cw,j}^{[k+1]} \right) \cdot \mathrm{bsfc} \left(\overline{\tilde{\mathbf{T}}}_{ice,j}^{[k+1]}, \overline{\tilde{\boldsymbol{\omega}}}_{ice,j}^{[k+1]} \right) \cdot \overline{\tilde{\mathbf{T}}}_{ice,j}^{[k+1]} \cdot \overline{\tilde{\boldsymbol{\omega}}}_{ice,j}^{[k+1]} \right], \end{aligned} \tag{13.22}$$

and

$$\begin{aligned}\overline{\tilde{\boldsymbol{\vartheta}}}_{cw,j}^{[k+1]} &= \overline{\tilde{\boldsymbol{\vartheta}}}_{cw,j}^{[k]} + h \cdot \varsigma_j \cdot \Gamma_{\tilde{\vartheta}_{cw}} \left(\overline{\tilde{\boldsymbol{\vartheta}}}_{cw,j}^{[k+1]}, \overline{\tilde{\mathbf{T}}}_{ice,j}^{[k+1]}, \overline{\tilde{\boldsymbol{\sigma}}}_{1,j}^{[k+1]} \right) \\ &= \overline{\tilde{\boldsymbol{\vartheta}}}_{cw,j}^{[k]} + h \cdot \varsigma_j \cdot \left\{ c_1 \cdot \left[\gamma_{cw} \cdot H_l \cdot \Gamma_{\tilde{\beta}} \left(\overline{\tilde{\boldsymbol{\vartheta}}}_{cw,j}^{[k+1]}, \overline{\tilde{\mathbf{T}}}_{ice,j}^{[k+1]}, \overline{\tilde{\boldsymbol{\sigma}}}_{1,j}^{[k+1]} \right) - \overline{\tilde{\mathbf{T}}}_{ice,j}^{[k+1]} \cdot \overline{\tilde{\boldsymbol{\omega}}}_{ice,j}^{[k+1]} \right] \right. \\ &\quad \left. - c_2 \cdot \left[\overline{\tilde{\boldsymbol{\vartheta}}}_{cw,j}^{[k+1]} - \overline{\tilde{\boldsymbol{\vartheta}}}_{amb,j}^{[k+1]} \right] \right\}, \end{aligned} \tag{13.23}$$

respectively. $\Gamma_{\tilde{\xi}}(\cdot)$, $\Gamma_{\tilde{\beta}}(\cdot)$, and $\Gamma_{\tilde{\vartheta}_{cw}}(\cdot)$ are the increment functions of the state of charge, fuel consumption, and coolant water temperature, respectively. We use again for the discretization of the dynamics the implicit Euler method to be coherent to problem $\mathcal{P}_{11}$, but this is not mandatory. Without loss of generality, any RK scheme from Chap. 5 can be employed.

With

$$\overline{\tilde{\mathbf{u}}}_j^{[k]} = \begin{bmatrix} \overline{\tilde{\mathbf{T}}}_{ice,j}^{[k]} \\ \overline{\tilde{\mathbf{T}}}_{brk,j}^{[k]} \end{bmatrix}, \tag{13.24}$$

problem $\mathcal{P}_{11}$ is reformulated as STO:

$$
\mathcal{P}_{12} := \begin{cases}
\phi\left(\overline{\tilde{\mathbf{u}}}^{*}\right) = \min\limits_{\overline{\tilde{\mathbf{u}}},\, \boldsymbol{\varsigma} \,\in\, \mathbb{R}^{N_{t_3}}} \overline{\tilde{\boldsymbol{\beta}}}^{[N_t]}_{N_{t_3}-1} + \tilde{\phi} & \\
\text{subject to} & \text{(13.21), (13.22), (13.23), and (13.24)} \\
\overline{\tilde{\boldsymbol{\xi}}}^{[0]}_{0} & = p_{11} \\
\overline{\tilde{\boldsymbol{\xi}}}^{[N_t]}_{N_{t_3}-1} & = p_{12} \\
\overline{\tilde{\boldsymbol{\beta}}}^{[0]}_{0} & = 0 \\
\overline{\tilde{\vartheta}}^{[0]}_{cw,0} & = 293\ (\mathrm{K}) \\
\mathbf{c}_{\overline{\tilde{u}}}\left(\overline{\tilde{\mathbf{u}}}^{[k]}_{j}\right) & \leq \mathbf{0}, \quad j = 1, \ldots, N_{t_3}-1, \quad k = 0, \ldots, N_t \\
\mathbf{c}_{\overline{\tilde{x}}}\left(\overline{\tilde{\boldsymbol{\xi}}}^{[k]}_{j}\right) & \leq \mathbf{0}, \quad j = 1, \ldots, N_{t_3}-1, \quad k = 0, \ldots, N_t \\
\overline{\boldsymbol{\alpha}}_{[i+1]-1} & \\
\quad \sum\limits_{j=\overline{\boldsymbol{\alpha}}_{[i]}} \varsigma_j - T_{i+1} + T_i & = 0, \quad i = 1, \ldots, N_{t_2}-1 \\
-\varsigma_j & \leq 0, \quad j = 1, \ldots, N_{t_3}-1 \\
\overline{\tilde{\boldsymbol{\xi}}}^{[0]}_{j} - \overline{\tilde{\boldsymbol{\xi}}}^{[N_t]}_{j-1} & = 0, \quad j = 1, \ldots, N_{t_3}-1
\end{cases}
\tag{13.25}
$$

where T_i are fixed grid points of the main time grid $\mathcal{G}_{t_2}$ and $\overline{\boldsymbol{\alpha}}$ assigns indices of the main time grid to the indices of the minor time grid $\mathcal{G}_{t_3}$.

The state and control constraints are defined as

$$
\mathbf{c}_{\overline{\tilde{x}}}\left(\overline{\tilde{\boldsymbol{\xi}}}^{[k]}_{j}\right) := \begin{bmatrix} \overline{\tilde{\boldsymbol{\xi}}}^{[k]}_{j} - \xi^{max} \\ \xi^{min} - \overline{\tilde{\boldsymbol{\xi}}}^{[k]}_{j} \end{bmatrix}
$$

and

$$
\mathbf{c}_{\overline{\tilde{u}}}\left(\overline{\tilde{\mathbf{u}}}^{[k]}_{j}\right) := \begin{bmatrix}
\overline{\tilde{\mathbf{T}}}^{[k]}_{ice,j} - T^{max}_{ice}\left(\overline{\tilde{\boldsymbol{\omega}}}^{[k]}_{ice,j}\right) \\
\overline{\tilde{\mathbf{T}}}^{[k]}_{gbx,j} - \overline{\tilde{\boldsymbol{\sigma}}}^{[k]}_{1,j} \cdot \overline{\tilde{\mathbf{T}}}^{[k]}_{ice,j} - p_{10} \cdot T^{max}_{mg}\left(\overline{\tilde{\boldsymbol{\omega}}}^{[k]}_{mg,j}\right) \\
p_{10} \cdot T^{min}_{mg}\left(\overline{\tilde{\boldsymbol{\omega}}}^{[k]}_{mg,j}\right) - \overline{\tilde{\mathbf{T}}}^{[k]}_{gbx,j} + \overline{\tilde{\boldsymbol{\sigma}}}^{[k]}_{1,j} \cdot \overline{\tilde{\mathbf{T}}}^{[k]}_{ice,j} \\
\hline
T^{min}_{ice} - \overline{\tilde{\mathbf{T}}}^{[k]}_{ice,j} \\
\overline{\tilde{\mathbf{T}}}^{[k]}_{brk,j} - T^{max}_{brk} \\
T^{min}_{brk} - \overline{\tilde{\mathbf{T}}}^{[k]}_{brk,j}
\end{bmatrix}.
$$

The fixed trajectories $\overline{\tilde{\boldsymbol{\omega}}}_{mg,j}$, $\overline{\tilde{\boldsymbol{\omega}}}_{ice,j}$, and $\overline{\tilde{\mathbf{T}}}_{gbx,j}$ are obtained from transforming of the trajectories (13.8), (13.9), and (13.10).

To give the STO as many *degrees of freedom* as possible zero arcs are introduced by switchings, which skip over intermediate gears. For instance, the gear sequence $4 \rightarrow 6$ will be transformed to the gear sequence $4 \rightarrow 5 \rightarrow 6$, where the arc duration of the engaged fifth gear is zero. If this transformation does not affect the optimality we can expect that these artificially introduced arcs are squeezed out by the numerical optimization.

Problem $\mathcal{P}_{12}$ is solved several times with a steadily increasing weighting factor γ_l of the term $\tilde{\phi}$ (10.135) for penalizing small switching arcs. After each iteration the connected switching arcs with the same discrete values are filtered by simply removing a certain number of small switching arcs. The filtering and penalizing of short switching arcs is very important for a successful solution of $\mathcal{P}_{12}$. However, the filtering procedure has to be performed with caution. Deletion of small but significant switching arcs can result in badly conditioned or even infeasible optimization problems. It is therefore advisable to remove only a small number of small arcs during each filtering step. We propose to filter and penalize the binary controls for the drive mode $\overline{\tilde{\boldsymbol{\sigma}}}_j$ at first. After that the gear shifting $\overline{\tilde{\boldsymbol{\kappa}}}_j$ should be filtered.

13.4 P2-Hybrid Design Study

For the powertrain design study a P2 hybrid with a dry seven-speed dual-clutch transmission is exemplarily investigated. We assume that the drive cycle is a priori known. The ARTEMIS test cycle (see Fig. 10.30) is chosen because of its realistic speed profile and its acceptable length. In general, the choice of a proper test cycle is very important and can strongly influence the design results. The parameters p_{11} and p_{12} are set to 0.5.

A good practice for the design study is to perform optimizations on some of the variables on fixed grids. We choose therefore a simplified model without coolant water state and fix the maximum power of the motor/generator to the values $P_{mg}^{max} = 15\,\text{kW}$, $P_{mg}^{max} = 20\,\text{kW}$, $P_{mg}^{max} = 25\,\text{kW}$, $P_{mg}^{max} = 30\,\text{kW}$, $P_{mg}^{max} = 35\,\text{kW}$, and $P_{mg}^{max} = 40\,\text{kW}$. The Pareto front needs not to be very accurate, so limiting the number of generations to 20 should give us already good approximations of the Pareto fronts to infer which power size of the MG is a good choice to carry on further investigations and if the power size has an impact at all.

We obtain a set of non-dominated solutions for each fixed maximal MG power. The Pareto fronts can be observed from Fig. 13.3. They illustrate the quantitative increase of fuel consumption due to higher drivability performance. Inspection of these plots gives us the hint that a MG with 40 kW outperforms vehicle configurations with smaller P_{mg}^{max} in terms of the fuel consumption and the drivability performance. Figures 13.4, 13.5, and 13.6 depict the corresponding speed ratio and gear ratios along the Pareto fronts.

With this prior knowledge, we can expect that the MOPD algorithm using the MG power as an additional variable and the coolant water $\overline{\vartheta}_{cw}$ as an additional state converge to a similar Pareto front, which gives us some confidence in the solution. Applying now the algorithm to the enlarged problem, we obtain the Pareto front as shown in Fig. 13.7a. The sub-figure shows the composition of the Pareto front dependent on the gear spreads for all individuals of the population for 40 generations. The increased fuel consumption in Fig. 13.7b is due to the fuel correction $CF_{fc}(\cdot)$ for engine warm-up.

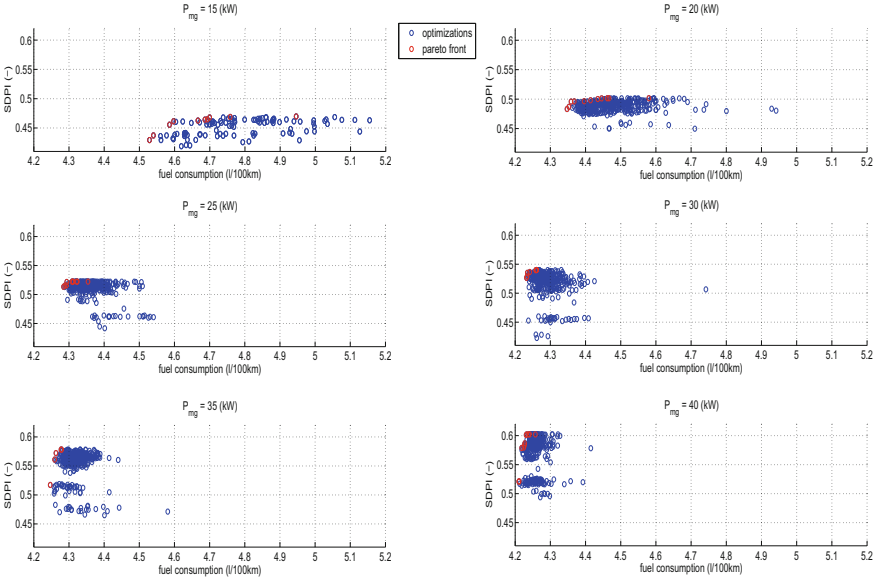


Fig. 13.3 Matrix of six panels of Pareto fronts

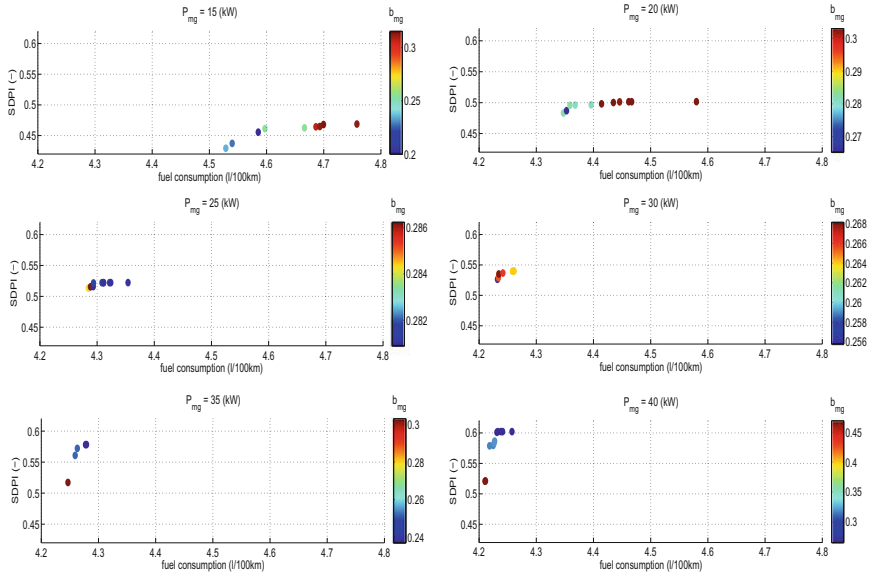


Fig. 13.4 Matrix of six panels of speed ratio b_{mg} over fuel consumption and SDPI

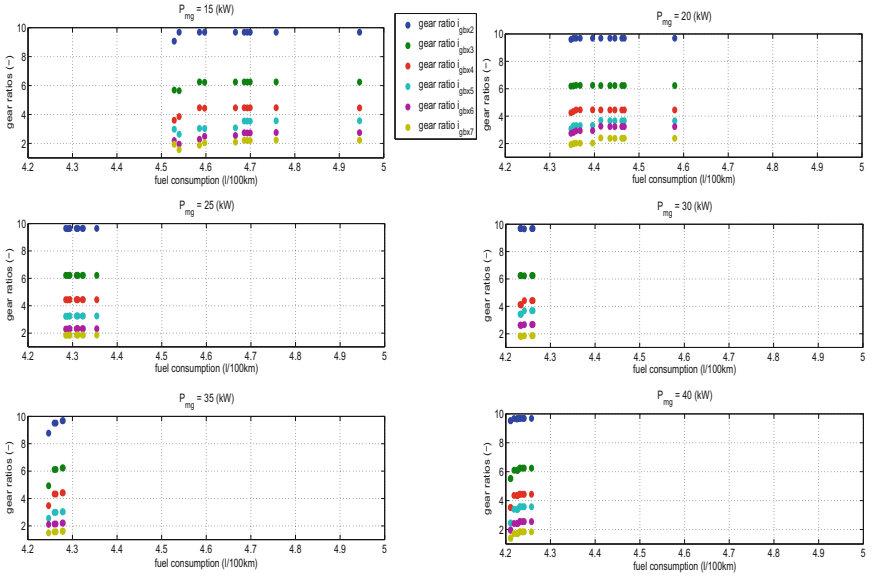


Fig. 13.5 Matrix of six panels of gear ratios over fuel consumption

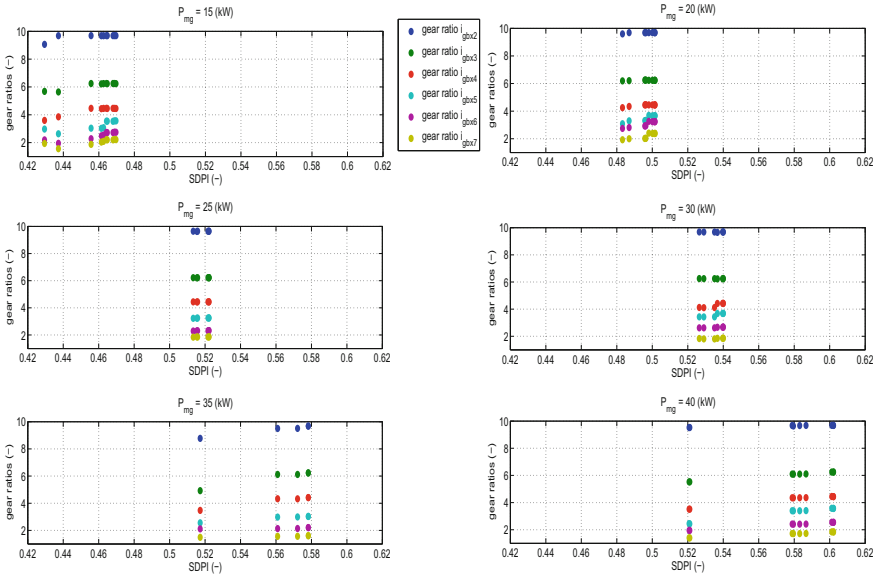
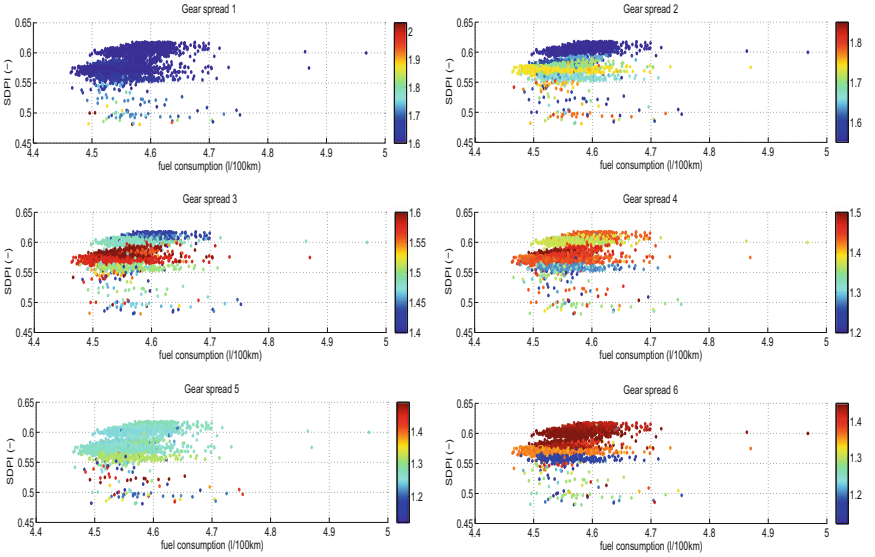
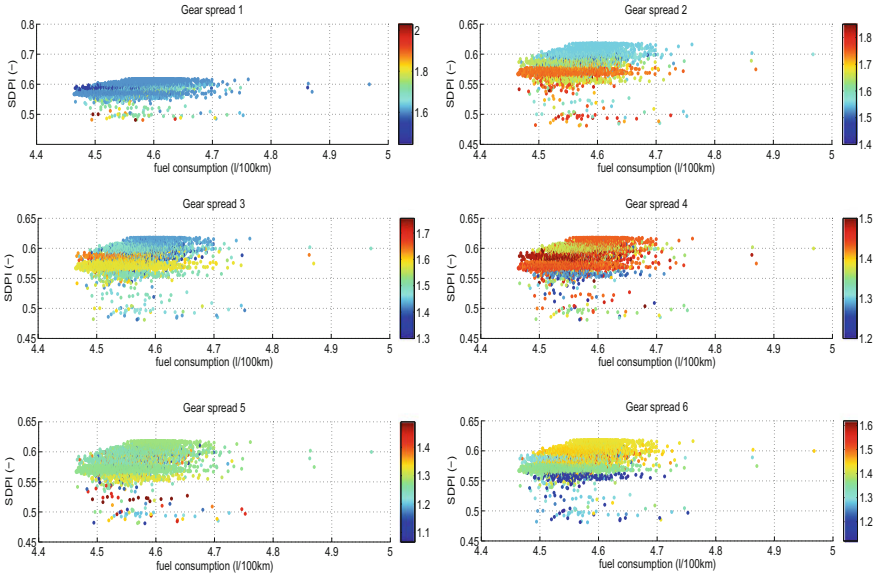


Fig. 13.6 Matrix of six panels of gear ratios over the SDPI



(a) Gear spreads constrained according to Table 13.1



(b) Gear spreads constrained according to Table 13.2

Fig. 13.7 Gear spreads over fuel consumption and SDPI

Table 13.2 Adapted design constraints for gear steps

Constraints	Range
Gear step φ_1 (–)	1.30–2.10
Gear step φ_2 (–)	1.40–1.85
Gear step φ_3 (–)	1.30–1.80
Gear step φ_4 (–)	1.20–1.50
Gear step φ_5 (–)	1.05–1.70
Gear step φ_6 (–)	1.05–1.80

It is clearly observable from Fig. 13.7a that the Pareto front is not completely connected. Especially, in the range of 0.57–0.60 of the subjective drivability performance index there may be vehicle configurations, which cannot be reached. Inspection of the gear spreads at this region suggests that especially gear spread 3 from Table 13.1 is too tightly bounded. As a remedy we assume that the mechanical construction of the gearbox allows us to relax some of the gear spread constraints according to Table 13.2. This assumption might not apply in general, since changing the ratios of the spur gears corresponds to different radii of the gearwheels and thus to a different gearbox volume.

Adapting the design constraints leads to the Pareto front as shown in Fig. 13.8. The figure shows all 11,250 solutions for 150 vehicle individuals per generation. The upper right picture zooms the Pareto front. The solutions below a subjective drivability performance index of 0.56 are irrelevant for the analysis, since Pareto solutions (red circles) dominate those. The gaps are now better covered, but could not be completely avoided. Again, inspection of the gear spreads shown in Fig. 13.7b reveals that gear spreads 1 and 4 are still too restrictive to obtain a smooth front.

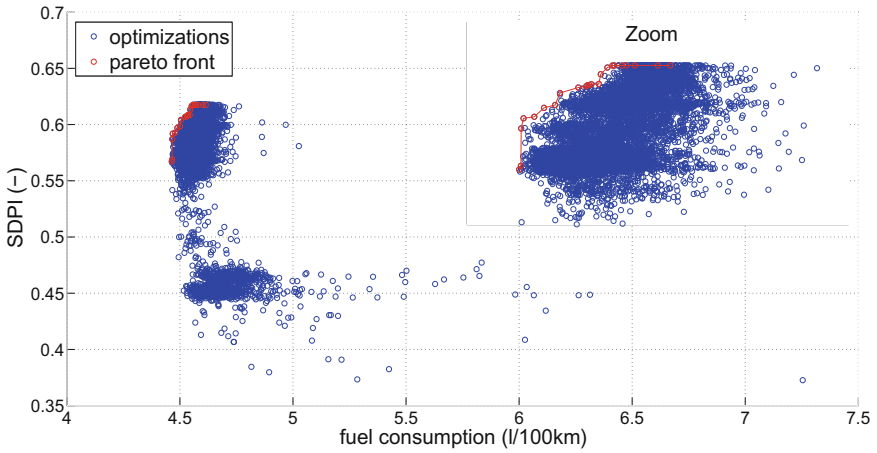


Fig. 13.8 All computed vehicle configuration solutions within 75 generations and with a population size of 150 individuals

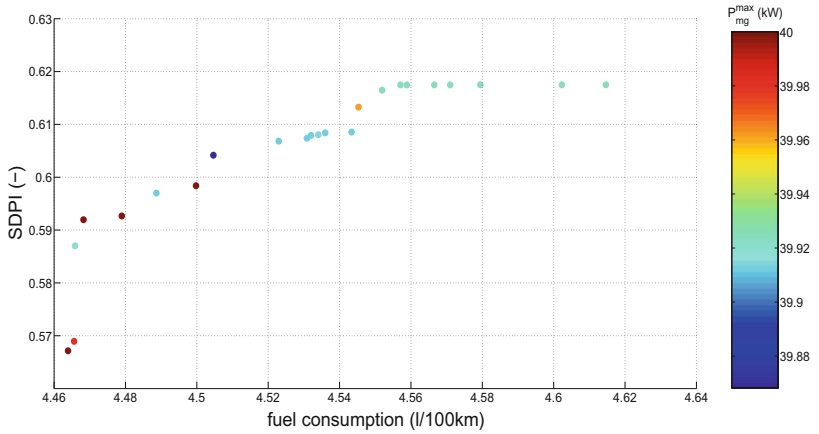


Fig. 13.9 Maximum power of the MG over fuel consumption and subjective drivability performance index

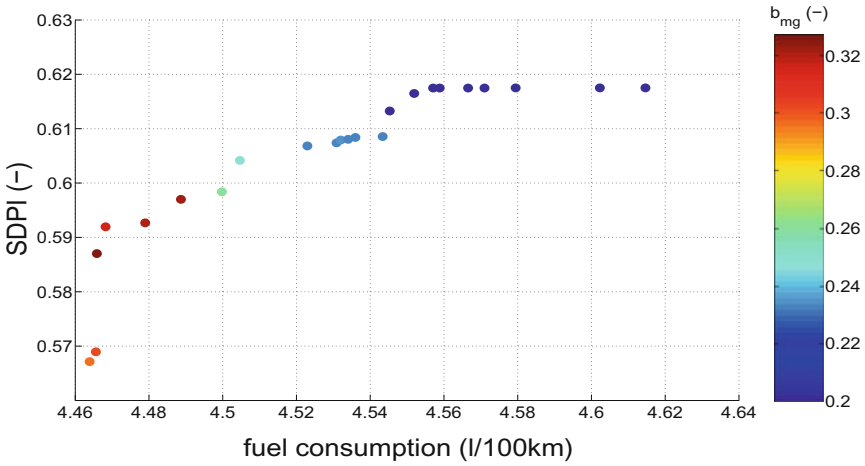


Fig. 13.10 Speed ratio b_{mg} over fuel consumption and subjective drivability performance index

Allowing these constraints to be relaxed even more should make it easier for the algorithm to explore these gaps, if desired at all.

As expected, the output power of the MG in Fig. 13.9 is for all vehicle configurations along the Pareto front constantly the maximum.

Considering Figs. 13.11 and 13.12, one can observe that the gear ratios approximate a progressive design, which means bigger steps for the lower gears and smaller steps for the higher gears. One can clearly see from the Fig. 13.12 that small gear ratios at higher gears lead to the best fuel economy. In Fig. 13.11, an increase of the vehicle agility leads to higher gear ratios at higher gears. Smaller gear steps are

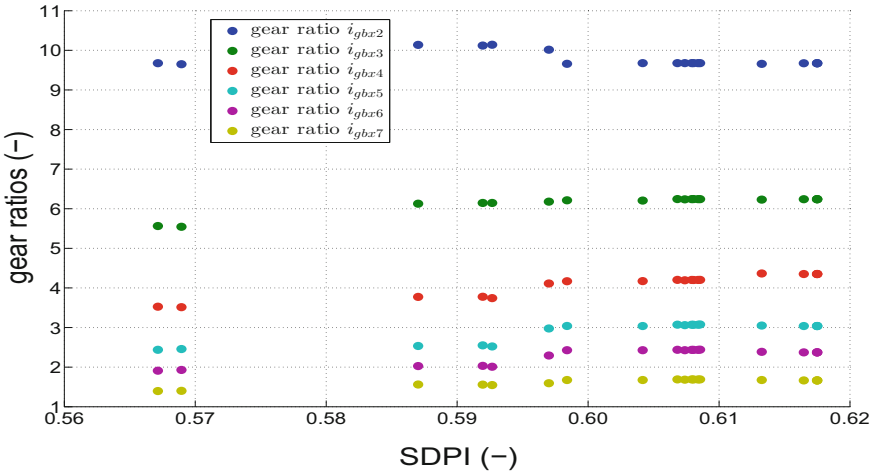


Fig. 13.11 Gear ratios over subjective drivability performance index

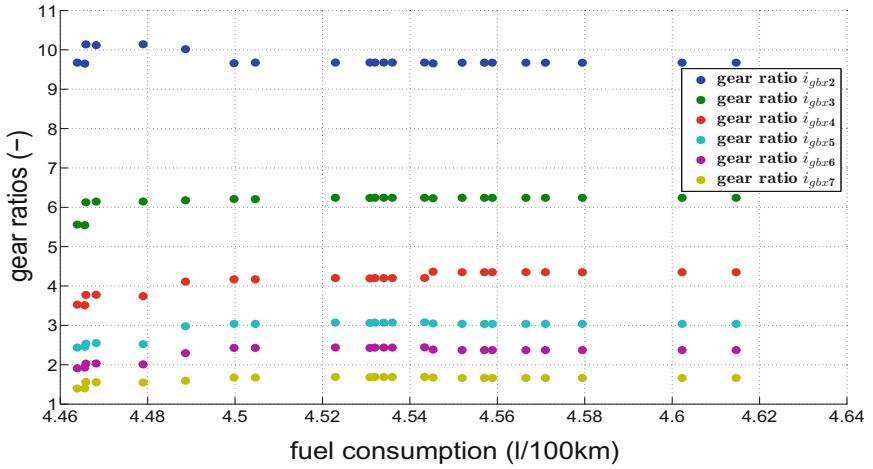


Fig. 13.12 Gear ratios over fuel consumption

selected to support the higher demanded longitudinal vehicle dynamics up to SDPI ≈ 0.6.

A further increase in the maximum acceleration can only be achieved by changing a property of the MG. A proper action for a higher vehicle agility can be explained by taking a view on the torque characteristics of the installed ICE. The maximum torque of 250 Nm of the employed ICE is reached at a revolution of 1500 min^{-1} (in mechanical engineering textbooks often abbreviated with 250 @ 1500 min^{-1}), which implies that the MG must provide its highest torques at least to the revolution of

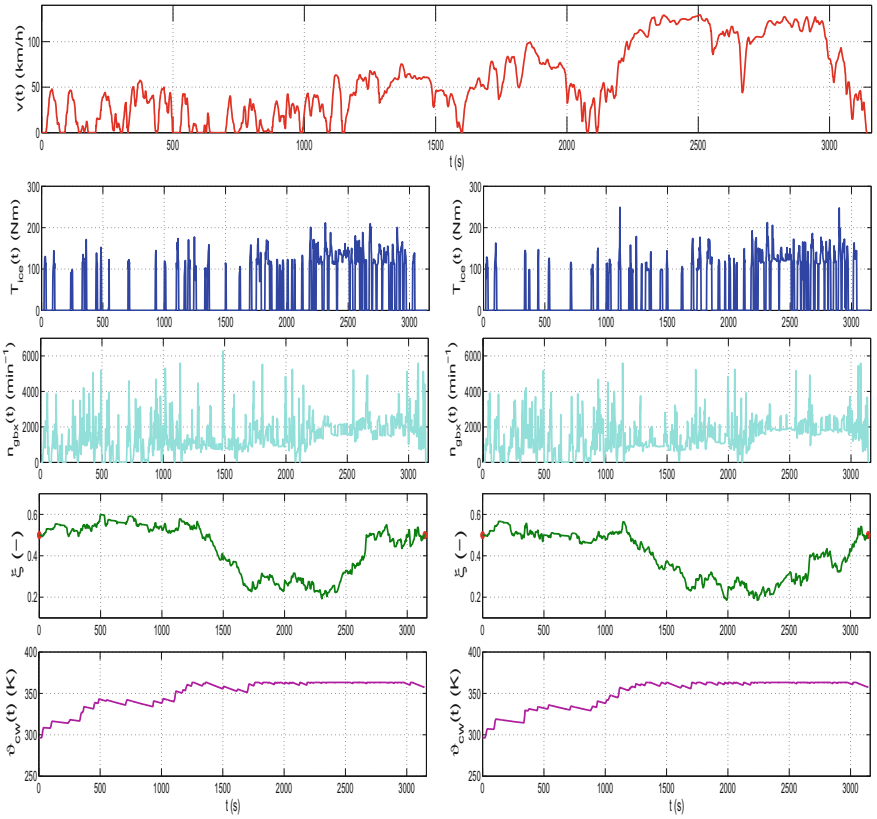


Fig. 13.13 *Left column* shows optimal trajectories for the vehicle configuration with the best fuel economy (SDPI $= 0.5671$ and $\beta(t_f) = 4.4639$ (l)); *Right column* shows optimal trajectories for the vehicle configuration with the highest agility (SDPI $= 0.6175$ and $\beta(t_f) = 4.5571$ (l))

$1500\,\mathrm{min}^{-1}$ before the field weakening region becomes active. This can be achieved by decreasing the speed ratio b_{mg} as shown in Fig. 13.10.

Shifting the speed ratio to low values can be interpreted as an action to shift the MG design to a "torque machine," which means higher currents and therefore thicker wire cross sections. This increases the mass of the MG and consequently the mass of the vehicle. Shortening the gear steps has the positive side effect to apply higher recuperation torques, which is beneficial for highly dynamical drive cycles.

Let us now investigate how the control strategies perform. Thus, the trajectories of the extremes of the Pareto solutions: a vehicle configuration with the best fuel economy and a vehicle configuration with the highest agility are depicted in Fig. 13.13.

The controls of both energy control strategies do not differ much. It is then not surprising that the state trajectories for the state of charge $\overline{\overline{\xi}}$ and the coolant water

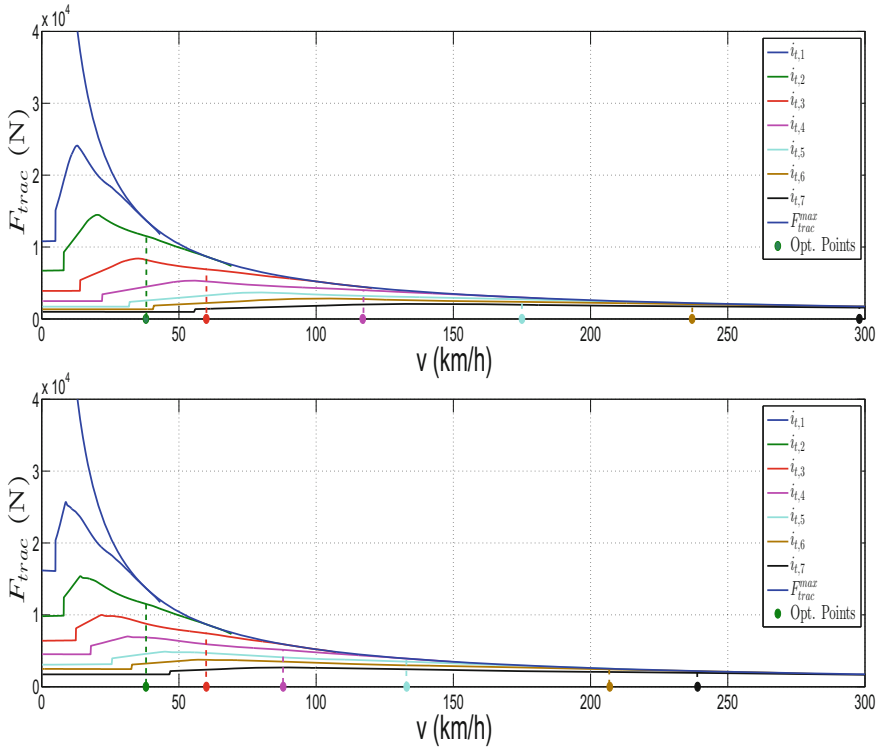


Fig. 13.14 Comparison of the traction forces of each gear. *Upper plot* shows the vehicle configuration with the best fuel economy (SDPI $= 0.5671$ and $\beta(t_f) = 4.4639$ (l)); *Lower plot* shows the vehicle configuration with the highest agility (SDPI $= 0.6175$ and $\beta(t_f) = 4.5571$ (l)). The *colored dots* indicate the optimization points for the acceleration in each gear

temperature $\overline{\tilde{\vartheta}}_{cw}$ of both vehicle configurations look quite similar. The temperature increase of the coolant water is relatively slow because of frequent electrical driving. It is clear that those energy management strategies are not optimal with respect to emissions.

The vehicle agility measured by the SDPI criterion can be imagined as a measure of how good the theoretical traction force can be approximated in each engaged gear using the maximal torque of the aggregated ICE and MG at a vehicle speed v. One can clearly see from Fig. 13.14 that the vehicle configuration with the highest agility approximates the total traction force better than the vehicle configuration with the best fuel economy. Especially, in the lower speed region below idle speed the traction force generated by the MG only is visibly larger. A further indicator for a good approximation are the optimization points, which are in the configuration with the highest agility shifted to the left.

It is remarkable that the switching structure provided by both control strategies are nearly insensitive to parameter variations, which suggests that the control strategies are robust. This needs to be further investigated in the next section.

13.5 Post Optimal Parametric Sensitivity Analysis

Solving the MOPD problem provided us valuable information about the sizing process of the components to achieve a hybrid vehicle design that is optimal with respect to the specifications. The optimal vehicle configurations found so far can be further used to gain even more information. For instance, to gain knowledge of how a vehicle design responds to perturbations in the design parameters. The key are here parameter sensitivities, which are powerful in analyzing the system behavior.

To calculate the parameter sensitivities two vehicle configurations are selected from Fig. 13.8. The first configuration is nearby the Pareto front, the second configuration is far away from the Pareto front. The parameters of the chosen vehicle configurations are listed in Table 13.3.

An approximation of the sensitivity differentials of the cost function, the continuous-valued controls, the continuous states, and some important vehicle measures can be obtained after computing the optimal solution of the nominal problem NLP($\mathbf{p}_0$) of $\mathcal{P}_{12}$, where $\mathbf{p} := \left[p_1, p_2, \ldots, p_{19}\right]^T$ is defined as the model parameter vector. Problem $\mathcal{P}_{12}$ must be re-optimized using the SQP method with an exact Hessian matrix. The re-optimization with the exact Hessian matrix is computational demanding but mandatory in order to fulfill the optimality conditions. It is therefore important to implement a compressed calculation of the Hessian of the Lagrangian as discussed in Sect. 9.2.4.3.

Once the optimal solution with the exact Hessian is obtained, Theorem 2.6 guarantees that a solution

Table 13.3 Vehicle configuration nearby the Pareto front

Design parameter	Conf. 1	Conf. 2
Rated power P_{mg}^{max} of MG (kW)	40	26.2
Speed ratio b_{mg} of MG (–)	0.31	0.438
Gear step φ_1 (–)	1.6131	2.0191
Gear step φ_2 (–)	1.5505	1.8312
Gear step φ_3 (–)	1.3859	1.4104
Gear step φ_4 (–)	1.4369	1.3230
Gear step φ_5 (–)	1.3787	1.1256
Gear step φ_6 (–)	1.3776	1.4357

$$\overline{\mathbf{y}}(\mathbf{p}) = \begin{bmatrix} \overline{\tilde{\mathbf{T}}}_{ice}(\mathbf{p}) \\ \overline{\tilde{\mathbf{T}}}_{brk}(\mathbf{p}) \\ \overline{\boldsymbol{\varsigma}}(\mathbf{p}) \\ \overline{\tilde{\boldsymbol{\xi}}}(\mathbf{p}) \\ \overline{\tilde{\boldsymbol{\omega}}}_{gbx}(\mathbf{p}) \\ \overline{\tilde{\mathbf{T}}}_{gbx}(\mathbf{p}) \end{bmatrix}$$

exists. The sensitivity differentials $(\mathrm{d}\overline{\tilde{\mathbf{T}}}^{[k]}_{ice,j}/\mathrm{d}\mathbf{p})(\mathbf{p}_0)$, $(\mathrm{d}\overline{\tilde{\mathbf{T}}}^{[k]}_{brk,j}/\mathrm{d}\mathbf{p})(\mathbf{p}_0)$, $(\mathrm{d}\varsigma_j/\mathrm{d}\mathbf{p})(\mathbf{p}_0)$, $(\mathrm{d}\overline{\tilde{\boldsymbol{\xi}}}^{[k]}_{j}/\mathrm{d}\mathbf{p})(\mathbf{p}_0)$, $(\mathrm{d}\overline{\tilde{\boldsymbol{\omega}}}^{[k]}_{gbx,j}/\mathrm{d}\mathbf{p})(\mathbf{p}_0)$, and $(\mathrm{d}\overline{\tilde{\mathbf{T}}}^{[k]}_{gbx,j}/\mathrm{d}\mathbf{p})(\mathbf{p}_0)$ are directly obtained from Corollary 2.2. The sensitivity differential of the cost function is obtained from Corollary 2.4.

In order to assess the parameter sensitivities the maximal sensitivity and mean sensitivity are introduced. The measures are described by

$$s_{max} = \operatorname{sgn}\left(\max\left\{\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0)\cdot \Delta\mathbf{p}\right\}\right)\cdot\left\|\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0)\cdot\Delta\mathbf{p}\right\|_\infty$$

and

$$s_{mean} = \frac{1}{N_t}\cdot\sum_{k=0}^{N_t}\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0)\cdot\Delta\mathbf{p},$$

where $\mathbf{y}(\cdot)$ represents the optimal solution function.

In order to compare the influence of the different parameters, it is important to disturb only one parameter at each evaluation. Thus, each parameter is increased by only 1% of the nominal value (i.e., $\Delta p_{0,i} = 0.01\cdot p_{0,i}$) by the rule

$$p_i = p_{0,i} + \Delta p_{0,i}.$$

In Table 13.4 the five largest sensitivities for vehicle configuration 1 and for the ARTEMIS drive cycle are summarized. The sensitivities are sorted w.r.t. the relative values.

Certainly, Table 13.4 needs some discussion. The highest impact on the fuel consumption have the vehicle speed v and the drag coefficient a_2. The latter one weights the quadratic increase of the resistance force with the vehicle speed. This means, that for the highly dynamic drive cycle the major influence on the fuel consumption is performed by design parameters (p_9, p_8, p_4, and p_6), which are independent from energy management. Just the choice of the initial state of charge has a small effect on the fuel efficiency. These results are not surprising for a drive cycle with a large speed and acceleration spectrum (cf. Fig. 10.30). In case of low dynamic drive cycles, the influence of $\xi(t_0)$ and $\xi(t_f)$ becomes more relevant as has been shown in the work of Schäfer [28].

Table 13.4 The five largest sensitivities of the objective function, the continuous-valued controls, and the continuous state for a disturbance of 1% of the nominal parameter value. The calculations are performed for the vehicle configuration 1 and for the ARTEMIS drive cycle

	No.	Parameter	Disturbance Δp_i	Absolute maximum of the sensitivities s_{max}		Average of the sensitivities s_{mean}	
				Absolute	Relative	Absolute	relative
$\mathrm{d}\phi/\mathrm{d}\mathbf{p} \cdot \Delta\mathbf{p}$	1.	$p_9 : \mathrm{sc}.v$	0.010000	0.035269	1.9107%	–	–
	2.	$p_8 : a_2$	0.000238	0.008232	0.4460%	–	–
	3.	$p_4 : m$	16.450000	0.007465	0.4044%	–	–
	4.	$p_6 : a_0$	0.000785	0.005299	0.2871%	–	–
	5.	$p_{11} : \xi(t_0)$	0.005000	−0.001759	−0.0953%	–	–
$\mathrm{d}\bar{\tilde{\mathbf{T}}}^{[k]}_{ice,j}/\mathrm{d}\mathbf{p} \cdot \Delta\mathbf{p}$	1.	$p_{11} : \xi(t_0)$	0.005000	−6.017674	−16.4189%	−0.113537	−0.1052%
	2.	$p_9 : \mathrm{sc}.v$	0.010000	2.725457	2.5636%	1.089569	1.0093%
	3.	$p_{12} : \xi(t_f)$	0.005000	1.818968	2.5502%	0.102549	0.0950%
	4.	$p_{16} : i_{t_4}$	0.044714	−0.645859	−1.7622%	−0.091022	−0.0843%
	5.	$p_5 : r_{wh}$	0.003080	1.899554	1.5635%	1.010704	0.9362%
$\mathrm{d}\bar{\tilde{\mathbf{T}}}^{[k]}_{brk,j}/\mathrm{d}\mathbf{p} \cdot \Delta\mathbf{p}$	1.	$p_9 : \mathrm{sc}.v$	0.01	−3.526717	40.5377%	−0.003054	9.8214%
	2.	$p_4 : m$	16.450000	−1.958081	22.5071%	−0.001635	5.2566%
	3.	$p_{10} : \mathrm{sc}.T_{mg}$	0.010000	1.729357	−19.8780%	0.001540	−4.9530%
	4.	$p_8 : a_2$	0.000238	0.141726	−1.6291%	0.000146	−0.4704%
	5.	$p_6 : a_0$	0.000785	0.122715	−1.4105%	0.000170	−0.5482%
$\mathrm{d}\varsigma_j/\mathrm{d}\mathbf{p} \cdot \Delta\mathbf{p}$	1.	$p_{19} : i_{t_7}$	0.016384	0.000525	2.2841%	−0.000000	<[−0.0000%
	2.	$p_9 : \mathrm{sc}.v$	0.010000	0.002353	0.1207%	0.000000	<0.0000%
	3.	$p_{11} : \xi(t_0)$	0.005000	−0.002271	−0.1136%	−0.000000	<0.0000%
	4.	$p_4 : m$	16.450000	0.001587	0.0814%	0.000000	<0.0000%
	4.	$p_5 : r_{wh}$	0.002958	0.001454	0.0746%	−0.000000	<0.0000%

(continued)

Table 13.4 (continued)

	No.	Parameter	Disturbance Δp_i	Absolute maximum of the sensitivities s_{max}		Average of the sensitivities s_{mean}	
				Absolute	Relative	Absolute	relative
$\mathrm{d}\bar{\tilde{\boldsymbol{\xi}}}_j^{[k]}/\mathrm{d}\mathbf{p}\cdot\Delta\mathbf{p}$	1.	$p_{12}:\xi(t_f)$	0.005000	0.501022	1.0387%	0.033651	0.0905%
	2.	$p_{11}:\xi(t_0)$	0.005000	0.500560	1.0207%	0.046042	0.1238%
	3.	$p_9:\mathrm{sc}.v$	0.010000	−0.189159	−0.9763%	0.006195	0.0167%
	4.	$p_1:Q_{bat}$	180.000000	0.135385	0.6988%	0.001042	0.0028%
	5.	$p_{10}:\mathrm{sc}.T_{mg}$	0.010000	0.116654	0.6021%	−0.000275	−0.0007%
$\mathrm{d}\bar{\tilde{\boldsymbol{\omega}}}_{gbx,j}^{[k]}/\mathrm{d}\mathbf{p}\cdot\Delta\mathbf{p}$	1.	$p_5:r_{wh}$	0.002958	−44.158950	−1.0003%	−12.960732	−1.0000%
	2.	$p_{17}:i_{t_5}$	0.031119	31.356994	1.0000%	2.356548	0.1818%
	3.	$p_{16}:i_{t_4}$	0.044714	44.144781	1.0000%	1.620858	0.1251%
	4.	$p_{14}:i_{t_2}$	0.096087	24.795318	1.0000%	0.685270	0.0529%
	5.	$p_{18}:i_{t_6}$	0.022571	25.896579	1.0000%	2.852519	0.2201%
$\mathrm{d}\bar{\tilde{\mathbf{T}}}_{gbx,j}^{[k]}/\mathrm{d}\mathbf{p}\cdot\Delta\mathbf{p}$	1.	$p_4:m$	16.450000	−2.131149	9.5074%	0.143773	0.4596%
	2.	$p_8:a_2$	0.000238	1.385211	−6.1796%	0.167323	0.5349%
	3.	$p_6:a_0$	0.000785	0.642181	−2.8649%	0.125447	0.4010%
	4.	$p_9:\mathrm{sc}.v$	0.010000	2.083920	1.0488%	0.346172	1.1066%
	5.	$p_{15}:i_{t_3}$	0.061971	−0.932212	−1.0007%	−0.004936	−0.0158%

The continuous-valued control $\overline{\tilde{\mathbf{T}}}_{ice}$ is mainly sensitive with respect to the changes in the initial and final state of charge. An increase in the vehicle speed plays also an important role because of the higher resistance force. Interestingly, the fourth gear ratio has the highest impact from all gear ratios on the engine torque. However, an increase of 1% of this gear ratio has only negligible impact on the fuel consumption.

The continuous-valued control $\overline{\tilde{\mathbf{T}}}_{brk}$ has the largest relative change, which is the result of very small optimal values. The absolute changes of this control have no significant influence on energy management.

Especially important are the sensitivities of the switching intervals $\boldsymbol{\varsigma}$, since a change in the length of the switching intervals can result in a modification of the control strategy. One can observe that the switching intervals are robust w.r.t. all disturbed parameters. The maximum absolute change is in the range of 2 ms.

The gearbox input speed $\overline{\tilde{\boldsymbol{\omega}}}_{gbx}$ is clearly dominated by the changes in the wheel radius and the gear ratios. If measured by the maximum sensitivity, the fifth gear ratio has the highest influence from all gear ratios on the gearbox input speed. If measured by the averaged sensitivity, the sixth gear ratio has the highest influence from all gear ratios on the gearbox input speed.

An increase of 1% of the vehicle mass increases the negative torque of $\overline{\tilde{\mathbf{T}}}_{gbx}$ and thus the mechanical break torque. Surprisingly, the three largest sensitivities with the parameters p_4, p_8, and p_6 have only minor effects on the engine torque $\overline{\tilde{\mathbf{T}}}_{ice}$. This means, that the optimal engine torque is robust against small disturbances of these parameters if the MG is not constrained in the torque direction and can compensate these perturbations.

Obviously, a small perturbation of 1% of the gear ratios and the rated power of the MG have only minor effects on the fuel consumption. But what happens for larger perturbations? In order to answer this question we have to approximate the confidence region of each sensitivity parameter. Using the theory discussed in Sect. 2.4.4, the confidence region is rendered by the set of active constraints, which are kept unchanged.

We obtain for the vehicle configuration 1 the confidence regions as shown in Table 13.5. One can observe from this table that some design parameters have lower and upper limits, which are fairly small. This means, that the optimal solution $\mathbf{y}^*$ has values, which are close to the constraints that change the active set. A remedy to enlarge these limits is to identify the involving constraints and to adapt these if possible at all. Using the lower and upper limits from Table 13.5, the maximal effect on the objective function can be calculated as shown in Table 13.6.

One can notice from Table 13.6 that if we set the auxiliary power p_3 to zero then we save approx. 4.5% of fuel. Unfortunately, this is not an option for a proper vehicle design, but gives us the hint to operate the electrical auxiliary devices only if needed.

With regard to the design parameters, an increase of the rated power P_{mg}^{max} and the gear ratios $i_{t_1} - i_{t_7}$ increases the fuel consumption. Instead, a decrease of the rated power and the gear ratios reduces only slightly the fuel consumption. This allows us to infer that vehicle configuration 1 is indeed an optimal vehicle configuration nearby the Pareto front.

Table 13.5 Lower and upper limits of the confidence regions for perturbation of the design parameters. The calculations are performed for the vehicle configuration 1 and for the ARTEMIS drive cycle

Parameter	$\mathbf{p}_0$	p_{min}		p_{max}	
		Absolute	Relative	Absolute	Relative
$p_1 : Q_{bat}$	18000.000000	17999.152273	−0.0047%	18003.395003	0.0189%
$p_2 : R_{bat}$	0.160600	0.000000	−100.0000%	1.334537	730.9694%
$p_3 : P_{aux}$	225.500000	0.000000	−100.0000%	622.000861	175.8319%
$p_4 : m$	1645.000000	1571.911947	−4.4430%	1736.571050	5.5666%
$p_5 : r_{wh}$	0.295800	0.289728	−2.0527%	0.296772	0.3285%
$p_6 : a_0$	0.078480	0.071866	−8.4278%	0.119776	52.6202%
$p_7 : a_1$	0.000245	0.000187	−23.9065%	0.000948	286.6503%
$p_8 : a_2$	0.023800	0.022967	−3.5004%	0.036031	51.3924%
$p_9 : \text{sc}.v$	1.000000	0.996711	−0.3289%	1.031787	3.1787%
$p_{10} : \text{sc}.P_{mg}^{max}$	1.000000	0.940047	−5.9953%	1.050307	5.0307%
$p_{11} : \xi(t_0)$	0.500000	0.433622	−13.2757%	0.525467	5.0935%
$p_{12} : \xi(t_f)$	0.500000	0.434703	−13.0594%	0.730564	46.1127%
$p_{13} : i_{t_1}$	15.500000	0.000000	−100.0000%	57.709697	272.3206%
$p_{14} : i_{t_2}$	9.608700	6.774686	−29.4943%	14.982137	55.9226%
$p_{15} : i_{t_3}$	6.197100	5.836386	−5.8207%	8.990280	45.0724%
$p_{16} : i_{t_4}$	4.471400	4.440582	−0.6892%	6.077389	35.9169%
$p_{17} : i_{t_5}$	3.111900	3.014668	−3.1245%	4.621720	48.5176%
$p_{18} : i_{t_6}$	2.257100	2.249682	−0.3286%	2.422387	7.3230%
$p_{19} : i_{t_7}$	1.638400	1.631894	−0.3971%	1.688223	3.0409%

Table 13.6 Lower and upper limits of the sensitivities of the objective function. The calculations are performed for the vehicle configuration 1 and for the ARTEMIS drive cycle

Parameter	s_{max} of the lower limit		s_{max} of the upper limit	
	Absolute	Relative	Absolute	Relative
$p_1 : Q_{bat}$	0.00000012	0.0000%	−0.00000047	−0.0000%
$p_2 : R_{bat}$	−0.06949813	−3.7650%	0.18589175	10.0706%
$p_3 : P_{aux}$	−0.08265268	−4.4777%	0.10619492	5.7531%
$p_4 : m$	−0.03316868	−1.7969%	0.04155660	2.2513%
$p_5 : r_{wh}$	0.00221054	0.1198%	−0.00035382	−0.0192%
$p_6 : a_0$	−0.04466250	−2.4196%	0.27885754	15.1070%
$p_7 : a_1$	−0.03365011	−1.8230%	0.40348034	21.8584%
$p_8 : a_2$	−0.02881488	−1.5610%	0.42305457	22.9188%
$p_9 : \text{sc}.v$	−0.01159976	−0.6284%	0.11210965	6.0735%
$p_{10} : \text{sc}.P_{mg}^{max}$	−0.00544826	−0.2952%	0.00457164	0.2477%
$p_{11} : \xi(t_0)$	0.02335170	1.2651%	−0.00895937	−0.4854%
$p_{12} : \xi(t_f)$	−0.02165741	−1.1733%	0.07647240	4.1429%
$p_{13} : i_{t_1}$	−0.00004000	−0.2200%	0.01088200	0.5895%
$p_{14} : i_{t_2}$	−0.00176085	−0.0954%	0.00333867	0.1809%
$p_{15} : i_{t_3}$	−0.00039315	−0.0213%	0.00304432	0.1649%
$p_{16} : i_{t_4}$	−0.00012092	−0.0066%	0.00630128	0.3414%
$p_{17} : i_{t_5}$	−0.00038057	−0.0206%	0.00590944	0.3201%
$p_{18} : i_{t_6}$	−0.00008959	−0.0049%	0.00199637	0.1082%
$p_{19} : i_{t_7}$	−0.00011858	−0.0064%	0.00090808	0.0492%

13.6 Further Work

13.6.1 Speedup of the Algorithm

For the evaluation of a population size of 150 individuals for 75 generations the computational demand can be very high and can take weeks on standard computer hardware even with parallelization. It seems obvious that parameter sensitivities can be used to accelerate this process.

If vehicle configuration 2 is selected and the lower and upper limits of the sensitivities of the objective function are calculated then it sounds reasonable to expect that the rated power P_{mg}^{max} and the gear ratios $i_{t_2} - i_{t_7}$ should be increased to lower the fuel consumption. The calculated parameter sensitivities in Table 13.7 matches with the expectations.

One strategy could be to double the population size virtually and to solve the first half of the population using the MOPD procedure from Sect. 13.3. The second half of the population is solved using the optimal solutions from the first half as blueprints. It seems reasonable to use the lower and upper limits of the sensitivities

Table 13.7 Lower and upper limits of the sensitivities of the objective function (only for the design parameters). The calculations are performed for the vehicle configuration 2 and for the ARTEMIS drive cycle

Parameter	s_{max} of the lower limit		s_{max} of the upper limit	
	Absolute	Relative	Absolute	Relative
p_{10} : sc.P_{mg}^{max}	0.00000000	0.0000%	−0.00256460	−0.1438%
p_{14} : i_{t_2}	0.00000000	0.0000%	−0.00000009	−0.0000%
p_{15} : i_{t_3}	0.00000000	0.0000%	−0.00037202	−0.0209%
p_{16} : i_{t_4}	0.00000000	0.0000%	−0.00129533	−0.0726%
p_{17} : i_{t_5}	0.00000001	0.0000%	−0.00215873	−0.1210%
p_{18} : i_{t_6}	0.00000000	0.0000%	−0.00040805	−0.0229%
p_{19} : i_{t_7}	0.00000001	0.0000%	−0.00002093	−0.0012%

of the objective function (only the parameters from Table 13.7 are necessary) to adjust the design parameters. The corresponding solution trajectories can be approximated by perturbed solution trajectories using the following equations:

$$\phi\left(\overline{\tilde{\mathbf{u}}}^{*}(\mathbf{p}),\mathbf{p}\right) \approx \phi\left(\overline{\tilde{\mathbf{u}}}^{*}\right) + \frac{\mathrm{d}\phi}{\mathrm{d}\mathbf{p}}\left(\overline{\tilde{\mathbf{u}}}^{*},\mathbf{p}_0\right)\Delta\mathbf{p}$$

$$\overline{\tilde{\mathbf{T}}}_{ice,j}^{[k]}(\mathbf{p}) \approx \left(\overline{\tilde{\mathbf{T}}}_{ice,j}^{*}\right)_{[k]} + \frac{\mathrm{d}\overline{\tilde{\mathbf{T}}}_{ice,j}^{[k]}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0)\,\Delta\mathbf{p}$$

$$\overline{\tilde{\mathbf{T}}}_{brk,j}^{[k]}(\mathbf{p}) \approx \left(\overline{\tilde{\mathbf{T}}}_{brk,j}^{*}\right)_{[k]} + \frac{\mathrm{d}\overline{\tilde{\mathbf{T}}}_{brk,j}^{[k]}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0)\,\Delta\mathbf{p}$$

$$\varsigma_j(\mathbf{p}) \approx \varsigma_j^{*} + \frac{\mathrm{d}\varsigma_j}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0)\,\Delta\mathbf{p}$$

$$\overline{\tilde{\boldsymbol{\xi}}}_{j}^{[k]}(\mathbf{p}) \approx \left(\overline{\tilde{\boldsymbol{\xi}}}_{j}^{*}\right)_{[k]} + \frac{\mathrm{d}\overline{\tilde{\boldsymbol{\xi}}}_{j}^{[k]}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0)\,\Delta\mathbf{p}$$

$$\overline{\tilde{\boldsymbol{\omega}}}_{gbx,j}^{[k]}(\mathbf{p}) \approx \left(\overline{\tilde{\boldsymbol{\omega}}}_{gbx,j}^{*}\right)_{[k]} + \frac{\mathrm{d}\overline{\tilde{\boldsymbol{\omega}}}_{gbx,j}^{[k]}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0)\,\Delta\mathbf{p}$$

$$\overline{\tilde{\mathbf{T}}}_{gbx,j}^{[k]}(\mathbf{p}) \approx \left(\overline{\tilde{\mathbf{T}}}_{gbx,j}^{*}\right)_{[k]} + \frac{\mathrm{d}\overline{\tilde{\mathbf{T}}}_{gbx,j}^{[k]}}{\mathrm{d}\mathbf{p}}(\mathbf{p}_0)\,\Delta\mathbf{p}$$

for $j = 1,\ldots,N_{t_3}-1$ and $k = 0,\ldots,N_t$ as discussed in Sect. 2.4.3. However, the approximation of the perturbed solution is only allowed for disturbances, which do not change the set of active constraints. This restriction is necessary since the Hessian of the Lagrangian is invalid and render the sensitivity differentials incorrect.

It is expected that this modification should speed up the MOPD notably.

13.6.2 Increase of Model Complexity

As we have already mentioned, it is sometimes necessary to increase the model complexity in order to consider some effects on the choice of the design parameters. For instant, the evolution of raw emissions or even of end-of-pipe emissions. From our experience, it is not advisable to solve complex model aggregation directly with the proposed optimization methodology. A more appealing approach could be *homotopy* by starting the multi-objective powertrain design with simple models first to obtain the Pareto front in an acceptable time span. Then, selecting some promising configurations from this front and using the solutions as initial conditions for further exploration with more feature-rich models should keep the computational effort manageable. These configuration candidates can then be extended with more features like thermodynamics (see Sect. 10.5.2) or a model predictive control strategy.

13.7 Bibliographical Notes

Optimal design studies have investigated different PHEV configurations (Liu [18]; Kaushal et al. [14]), battery sizing (Shiau et al. [31]), engine and motor sizing (Assanis et al. [1]; Patil et al. [21]), etc.

This optimization problem has been seen by many authors as an optimization problem in a dynamic environment. Cook et al. [7] solved this optimization problem by fixing the controller structure and applying a *genetic algorithm* (GA). The system dynamics has been ignored by this procedure.

Different optimization techniques have been applied in the context of optimal design to find the most efficient powertrain sizing. For instance, in Boehme et al. [3, 4], a MOGA has been used to find the suboptimal gear ratios, battery capacity, and MG rated power with respect to fuel consumption only. A comparison between a former version of the MOPD approach with the MOGA approach has been conducted in Boehme et al. [5, 6]. In Sundström et al. [34], a DP approach is applied to optimize non-convex, mixed-integer powertrain models, whereas a convex optimization scheme for component sizing is proposed in Johannesson et al. [12]. A direct collocation method using DIRCOL and a BB method has been investigated by Jörg et al. [13] for designing the electrical components like MG and supercap of a hybrid vehicle with CVT.

The question about the optimal battery size for HEVs or PHEVs has been skipped in this chapter. Patil [22] treated this topic for battery sizing for PHEVs using a combined optimization approach to evaluate optimal battery sizes for minimum life cycle CO_2 emissions. The CO_2 impacts associated with a certain battery's life can be categorized into CO_2 from materials and manufacturing, use cases, and recycling phases. The amount of CO_2 resulting from the manufacturing phase ranges from 90 to 120 (kg/kWh), as reported in Sullivan et al. [33]. The author assembled the cost function of the optimization problem using the CO_2 impact of the battery production

per day, the CO_2 produced from fuel usage, and the CO_2 corresponding to the energy mix of the grid charging. Investigations on the effects of the battery capacity on the fuel economy can be found in Neglur and Ferdowsi [20], Moura et al. [19]. Moura et al. [19] showed that an optimized energy control strategy can reduce energy consumption and hence reduces battery capacity requirements.

The battery's state of health has been considered in Johannesson et al. [12] for the entire life span of a hybrid bus. Hu et al. [11] proposed a convex programming paradigm for optimizing a combination of lithium-ion cells and supercapacitors for a hybrid bus. A dynamic state of health model is then used to examine the replacement effects on the new hybrid energy storage system.

The requirements on the battery concerning, all-electric range, drive cycle, and control strategy have been investigated by Rousseau et al. [26]. Battery aging effects can have a tremendous effect on the hybrid vehicle design and have been investigated by different authors, among them Serrao et al. [30], Smith et al. [32], and Sciarretta et al. [29]. The effect of different driving patterns on the optimal sizing of battery, MG, and engine of a series of PHEVs is studied by Pourabdollah et al. [24].

A crucial point of the vehicle design is the feasibility. A vehicle candidate with optimized parameters can only be regarded as valid if all the design constraints can be fulfilled. Kolmanovsky et al. [16] performed a multi-objective assessment of the powertrain capability using a DP algorithm to check if its design targets and constraints are met.

Sensitivities between control strategy parameters and fuel economy have been performed based on correlation in Rousseau et al. [27]. Parameter sensitivities for some important drive cycles have been investigated by Schäfer [28]. The author showed that parameter sensitivities with different ranking and height are yielded for different drive cycles.

Drivability aspects have been considered by Koprubasi et al. [17]. The authors proposed a separated design process for energy management and damping control for the oscillations induced by gear shifts or tip-in/tip-out excitations. Pisu et al. [23] treated the drivability objective as a control objective to design a decoupling control to achieve smooth gear shiftings and minimal driveline vibrations. Barbarisi et al. [2] also treated the state of charge as an individual control component, hence entirely decoupling $\xi(\cdot)$ from fuel management and drivability control.

Emission constraints in the design of hybrid vehicles have been considered by many authors; among them Kleimaier [15].

References

1. Assanis D, Delagrammatikas G, Fellini R, Filipi Z, Liedtke J, Michelena N, Papalambros P, Reyes D, Rosenbaum D, Sales A et al (1999) Optimization approach to hybrid electric propulsion system design. J Struct Mech 27(4):393–421
2. Barbarisi O, Westervelt ER, Vasca F, Rizzoni G (2005) Power management decoupling control for a hybrid electric vehicle. In: Proceedings of the 44th decision and control conference, IEEE, pp 2012–2017

3. Boehme T, Metwally O, Becker B, Meinhardt N, Rucht M, Rabba H (2012) A simulation-based comparison of different power split configurations with respect to the system efficiency. In: SAE world congress. Technical paper 2012-01-0438. doi:10.4271/2012-01-0438
4. Boehme TJ, Becker B, Ruben-Weck M, Rothschuh M, Boldt A, Rollinger C, Butz R, Rabba H (2013) Optimal design strategies for different hybrid powertrain configurations assessed with European drive cycles. In: SAE world congress, Technical paper 2013-01-1751. doi:10.4271/2013-01-1751
5. Boehme TJ, Frank B, Schori M, Jeinsch T (2014a) Multi-objective optimal powertrain design of parallel hybrid vehicles with respect to fuel consumption and driving performance. In: Proceedings of the 2014 European control conference (ECC). IEEE, pp 1017–1023
6. Boehme TJ, Rothschuh M, Frank B, Schultalbers M, Schori M, Jeinsch T (2014) Multi-objective optimal design of parallel plug-in hybrid powertrain configurations with respect to fuel consumption and driving performance. SAE Int J Alt Power 3(2):176–192. doi:10.4271/2014-01-1158
7. Cook R, Molina-Cristobal A, Parks G, Correa CO, Clarkson PJ (2007) Multi-objective optimisation of a hybrid electric vehicle: drive train and driving strategy. Springer
8. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6:182–197
9. Ehsani M, Gao Y, Emadi A (2010) Modern electric, hybrid electric, and fuel cell vehicles. Fundamentals, theory, and design, 2nd edn. CRC Press
10. Guzzella L, Sciarretta A (2005) Vehicle propulsion systems. Introduction to modeling and optimization. Springer, Berlin
11. Hu X, Johannesson L, Murgovski N, Egardt B (2015) Longevity-conscious dimensioning and power management of the hybrid energy storage system in a fuel cell hybrid electric bus. Appl Energy 137:913–924
12. Johannesson L, Murgovski N, Ebbesen S, Egardt B, Gelso E, Hellgren J (2013) Including a battery state of health model in the HEV component sizing and optimal control problem. In: Proceedings of the 7th IFAC symposium on advances in automotive control, Tokyo, Japan, pp 388–393
13. Jörg A et al (2010) Optimale Auslegung und Betriebsführung von Hybridfahrzeugen. PhD thesis, Technische Universität München
14. Kaushal N, Shiau CSN, Michalek JJ (2009) Optimal plug-in hybrid electric vehicle design and allocation for diverse charging patterns. In: ASME 2009 international design engineering technical conferences and computers and information in engineering conference. American Society of Mechanical Engineers, pp 899–908
15. Kleimaier A (2004) Optimale Betriebsführung von Hybridfahrzeugen. PhD thesis, Technische Universität München
16. Kolmanovsky IV, Sivashankar SN, Sun J (2005) Optimal control-based powertrain feasibility assessment: a software implementation perspective. In: Proceedings of the American control conference. IEEE, pp 4452–4457
17. Koprubasi K, Morbitzer J, Westervelt E, Rizzoni G (2006) Toward a framework for the hybrid control of a multi-mode hybrid-electric driveline. In: Proceedings of the American control conference. Minneapolis, IEEE, pp 3296–3301
18. Liu J (2007) Modeling, configuration and control optimization of power-split hybrid vehicles. PhD thesis, The University of Michigan
19. Moura SJ, Callaway DS, Fathy HK, Stein JL (2010) Tradeoffs between battery energy capacity and stochastic optimal power management in plug-in hybrid electric vehicles. J Power Sources 195(9):2979–2988
20. Neglur S, Ferdowsi M (2009) Effect of battery capacity on the performance of plug-in hybrid electric vehicles. In: Vehicle power and propulsion conference 2009, VPPC'09. IEEE, pp 649–654
21. Patil R, Adornato B, Filipi Z (2010) Design optimization of a series plug-in hybrid electric vehicle for real-world driving conditions. SAE Int J Engines 3(2010-01-0840):655–665

22. Patil RM (2012) Combined design and control optimization: application to optimal PHEV design and control for multiple objectives. PhD thesis, The University of Michigan
23. Pisu P, Koprubasi K, Rizzoni G (2005) Energy management and drivability control problems for hybrid electric vehicles. In: Proceedings of the 44th IEEE conference on decision and control. IEEE, pp 1824–1830
24. Pourabdollah M, Grauers A, Egardt B (2013) Effect of driving patterns on components sizing of a series PHEV. In: Proceedings of the 7th IFAC symposium on advances in automotive control, Tokyo, Japan, pp 17–22
25. Rechs M, Menne R, Tielkes U, Pingen B (2002) Torque Boost - Drehmomenterhhung und Verbrauchsreduzierung im realen Fahrbetrieb. In: 23. International Wiener Motorensymposium
26. Rousseau A, Shidore N, Carlson R, Freyermuth V (2007) Research on PHEV battery requirements and evaluation of early prototypes. Technical report, Argonne National Laboratory
27. Rousseau A, Pagerit S, Gao DW (2008) Plug-in hybrid electric vehicle control strategy parameter optimization. J Asian Electr Veh 6(2):1125–1133
28. Schäfer R (2014) Gemischt-ganzzahlige Optimalsteuerung, Sensitivitätsanalyse und Echtzeitoptimierung von Parallel-Hybridfahrzeugen. Master's thesis, Universität Bremen
29. Sciarretta A, di Domenico D, Pognant-Gros P, Zito G (2014) Optimal energy management of automotive battery systems including thermal dynamics and aging. In: Optimization and optimal control in automotive systems. Springer, pp 219–236
30. Serrao L, Onori S, Sciaretta A, Guezennec Y, Rizzoni G (2011) Optimal energy management of hybrid electric vehicles including battery aging. In: Proceedings of the 2011 American control conference. IEEE, San Francisco, pp 2125–2130
31. Shiau CSN, Kaushal N, Hendrickson CT, Peterson SB, Whitacre JF, Michalek JJ (2010) Optimal plug-in hybrid electric vehicle design and allocation for minimum life cycle cost, petroleum consumption, and greenhouse gas emissions. J Mech Des 132(9):1–11
32. Smith K, Earleywine M, Wood E, Neubauer J, Pesaran A (2012) Comparison of plug-in hybrid electric vehicle battery life across geographies and drive cycles. In: SAE world congress, Technical paper 2012-01-0666. doi:10.4271/2012-01-0666
33. Sullivan J, Gaines L et al (2010) A review of battery life-cycle analysis: state of knowledge and critical needs. Technical report, Argonne National Laboratory (ANL)
34. Sundström O, Guzzella L, Soltic P (2010) Torque-assist hybrid electric powertrain sizing: from optimal control towards a sizing law. IEEE Trans Control Syst Technol 18. doi:10.1109/TCST.2009.2030173

Part VI
Appendix

Chapter 14
Graph Theoretical Fundamentals for Sparse Matrices

We only state the graph theoretical concepts, which are in the scope of this book. A comprehensive introduction to graph theory can be found in the textbooks of Diestel [1], George et al. [2], Golumbic [3], and Wilson [4].

If we deal with sparse matrices graph theoretical methods play a great role, because the structure of a sparse matrix can be visualized as a graph. Therefore, we review some fundamentals of graph theory, which are needed to apply the proposed methods.

Important set operations are as follows:

- **size of the set**: $|\cdot|$;
- **union of sets**: $\mathcal{B}_1 \cup \mathcal{B}_2 := \{v \mid v \in \mathcal{B}_1 \vee v \in \mathcal{B}_2\}$;
- **intersection of sets**: $\mathcal{B}_1 \cap \mathcal{B}_2 := \{v \mid v \in \mathcal{B}_1 \wedge v \in \mathcal{B}_2\}$; and
- **difference of two sets**: $\mathcal{B}_1 \setminus \mathcal{B}_2 := \{v \mid v \in \mathcal{B}_1 \wedge v \notin \mathcal{B}_2\}$.

Definition 14.1 (*Undirected Graph*) An undirected graph $G(\mathcal{V}, \mathcal{B})$ is defined by a set of vertices $\mathcal{V}$ and a set of edges $\mathcal{B} \subseteq [\mathcal{V}]^2$, such that the set of edges $\mathcal{B}$ consists of 2-element subsets of the set of vertices $\mathcal{V}$. △

A simple undirected graph is illustrated in Fig. 14.1.

Definition 14.2 (*Directed Graph*) A directed graph $G(\mathcal{V}, \mathcal{A})$ is defined by a set of vertices $\mathcal{V}$ and a set of edges $\mathcal{A} \subseteq [\mathcal{V}]^2$, such that the set of edges $\mathcal{A}$ consists of 2-element subsets of the set of vertices $\mathcal{V}$, but in contrast to an undirected graph the edges have an *orientation* and therefore $uv \in \mathcal{A}$ is not the same as $vu \in \mathcal{A}$. The edges are visualized as arrows. △

A simple directed graph is shown in Fig. 14.2.

Definition 14.3 (*Adjacency*) Two vertices which are linked by an edge are said to be *adjacent*. △

Definition 14.4 (*Adjacency set*) For a vertex $v \in \mathcal{V}$ the set $adj(v)$ consists of all vertices, which are adjacent to v, and is called *adjacency set*. △

T.J. Böhme and B. Frank, *Hybrid Systems, Optimal Control and Hybrid Vehicles*,
Advances in Industrial Control, DOI 10.1007/978-3-319-51317-1_14

Definition 14.5 (*Complete Graph*) A graph $G(\mathcal{V}, \mathcal{B})$ is called *complete graph*, if all its vertices are adjacent. △

Definition 14.6 (*Path*) A *path* is a subgraph $P(\mathcal{V}_p, \mathcal{B}_p) \subseteq G(\mathcal{V}, \mathcal{B})$ with

$$\mathcal{V}_p = \{v_0, v_1, \ldots, v_k\} \subseteq \mathcal{V}, \qquad \mathcal{B}_p = \{v_0v_1, v_1v_2, \ldots, v_{k-1}v_k\} \subseteq \mathcal{B}$$

where all v_i for $i = 1, \ldots, k$ are distinct. The vertices v_0 and v_k are linked by the path $P(\mathcal{V}_p, \mathcal{B}_p)$. The number of edges k of the path $P(\mathcal{V}_p, \mathcal{B}_p)$ is the *length of the path*. △

Definition 14.7 (*Cycle*) A path $P(\mathcal{V}_p, \mathcal{B}_p)$ together with the additional edge v_kv_0 is called a *cycle* $C := P(\mathcal{V}_p, \mathcal{B}_p \cup v_kv_0$ with length $k + 1$. A graph which does not contain a cycle as a subgraph is called *acyclic*. △

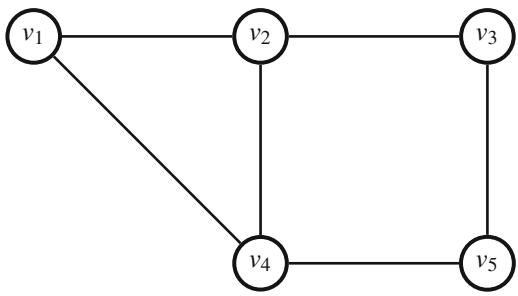


Fig. 14.1 A simple undirected graph $G(\mathcal{V}, \mathcal{B})$ with the vertices $\mathcal{V} = \{v_1, v_2, v_3, v_4, v_5\}$ and edges $\mathcal{B} = \{v_1v_2, v_1v_4, v_2v_4, v_2v_3, v_3v_5, v_4v_5\} = \{v_2v_1, v_4v_1, v_4v_2, v_3v_2, v_5v_3, v_5v_4\}$

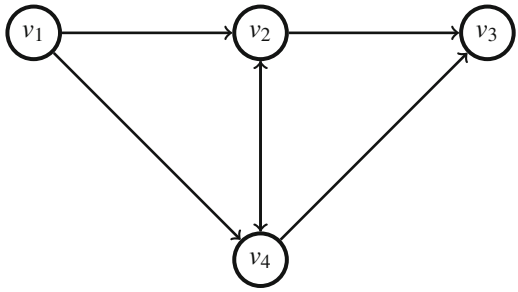


Fig. 14.2 A simple directed graph $G(\mathcal{V}, \mathcal{A})$ with the vertices $\mathcal{V} = \{v_1, v_2, v_3, v_4\}$ and the edges $\mathcal{A} = \{v_1v_4, v_1v_2, v_2v_3, v_2v_4, v_4v_2, v_4v_3\}$

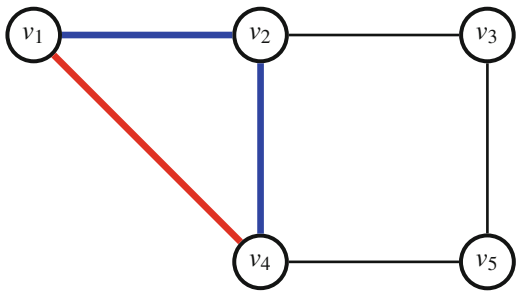


Fig. 14.3 Illustration of a path of length 2 (*blue edges*) and a cycle (*blue* and *red edges*)

An illustration of a path and a cycle is shown in Fig. 14.3.

Definition 14.8 (*Chord*) An edge, which is not an element of a cycle C, but which links two vertices of C, is called a *chord* of that cycle. △

Definition 14.9 (*Chordal Graph*) A graph $G(\mathcal{V}, \mathcal{B})$ is *chordal* if every cycle of length greater than three has a chord. △

Definition 14.10 (*Clique*) Any maximal set of vertices $\mathcal{C}_{cl} \subseteq \mathcal{V}$ of a graph $G(\mathcal{V}, \mathcal{B})$ that is complete in $G(\mathcal{V}, \mathcal{B})$ is called a *clique*. Thus, a clique is not completely contained in another clique. △

The blue edges in Fig. 14.4 show a circle and the blue-dashed edge between v_2 and v_5 is a chord of that circle. Together with this chord the graph is chordal and contains the three cliques $\mathcal{C}_{cl,1} = \{v_1, v_2, v_4\}$, $\mathcal{C}_{cl,2} = \{v_2, v_4, v_5\}$, and $\mathcal{C}_{cl,3} = \{v_2, v_3, v_5\}$.

Definition 14.11 (*Simplicial vertex*) A vertex $v \in \mathcal{V}$ is called simplicial if its adjacency set $adj(v)$ is a clique. △

Definition 14.12 (*Connected Graph*) A graph $G(\mathcal{V}, \mathcal{B})$ is said to be *connected*, if any two vertices are linked by a path in $G(\mathcal{V}, \mathcal{B})$. If any two vertices are not linked by a path in $G(\mathcal{V}, \mathcal{B})$, the graph is said to be *disconnected* (see Fig. 14.5). △

A graph $G(\mathcal{V}, \mathcal{B})$ can be represented as a matrix called the adjacency matrix.

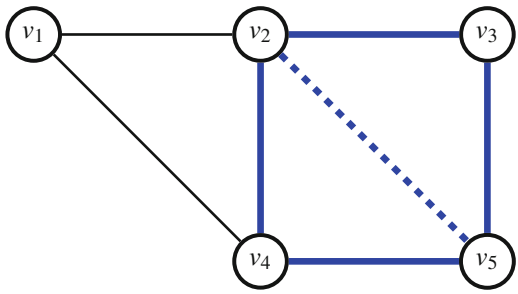


Fig. 14.4 A chordal graph

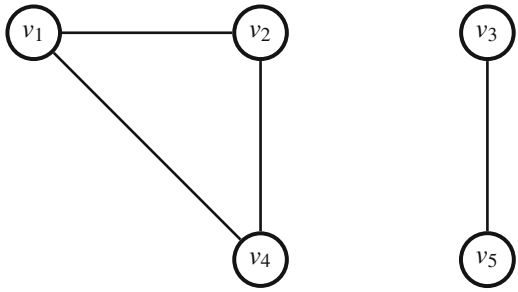


Fig. 14.5 A disconnected graph

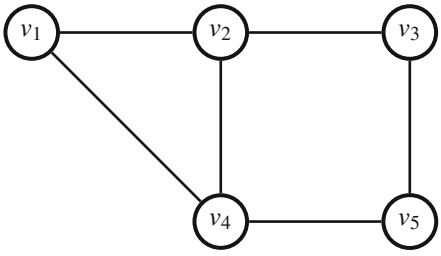
Definition 14.13 (*Adjacency Matrix*) The symmetric *adjacency matrix* $\mathbf{A}$ with dimension $N_v \times N_v$ and entries $a_{i,j}$ in the i-th row and j-th column of an undirected graph $G(\mathcal{V}, \mathcal{B})$ with an N_v-dimensional set of vertices $\mathcal{V} = \{v_1, v_2, \ldots, v_{N_v}\}$ is defined by

$$a_{i,j} = \begin{cases} 1, & \text{if } v_i v_j \in \mathcal{B}, \\ 0, & \text{otherwise.} \end{cases}$$

△

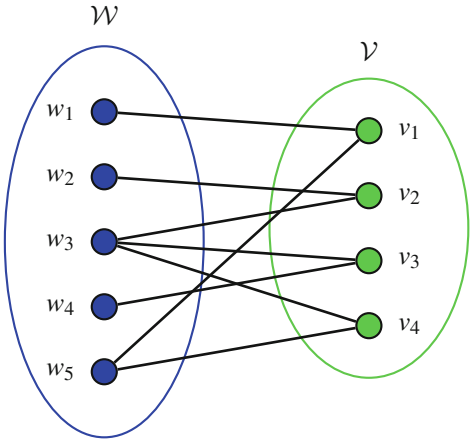
The adjacency matrix can also be derived from directed graphs. Then, transposing of the adjacency matrix $\mathbf{A}$ changes the direction of the edges. An adjacency graph with a corresponding adjacency matrix can be seen in Fig. 14.6.

Definition 14.14 (*Bipartite Graph*) A graph $G(\mathcal{W}, \mathcal{V}, \mathcal{B})$ with a set of edges $\mathcal{B}$ and two distinct sets of vertices $\mathcal{W}$ and $\mathcal{V}$ is called *bipartite graph* if every edge $e \in \mathcal{B}$ connects a vertex from the set $\mathcal{W}$ with a vertex from the set $\mathcal{V}$. △



$$\mathbf{A} = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

Fig. 14.6 The graph from Fig. 14.1 and its adjacency matrix



$$\mathbf{B} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

Fig. 14.7 A bipartite graph and its biadjacency matrix $\mathbf{B}$

Definition 14.15 (*Biadjacency Matrix*) The *biadjacency matrix* **B** with dimension $N_w \times N_v$ and entries $b_{i,j}$ in the i-th row and j-th column of a bipartite graph $G(\mathcal{W}, \mathcal{V}, \mathcal{B})$ with a N_w-dimensional set of vertices $\mathcal{W}$ and a N_v-dimensional set of vertices $\mathcal{V}$ is defined by

$$b_{i,j} = \begin{cases} 1, & \text{if } w_i v_j \in \mathcal{B}, \\ 0, & \text{otherwise.} \end{cases}$$

So, the vertices w_i correspond to the i-th row and the vertices v_j to the j-th column of the matrix **B**. △

In Fig. 14.7 a biadjacency graph with its biadjacency matrix is illustrated.

Definition 14.16 (*Vertex Coloring*) A map $c : \mathcal{V} \to \mathcal{J}$ is called a *vertex coloring* of a graph $G(\mathcal{V}, \mathcal{B})$, if $c(u) \neq c(v)$ for all adjacent $u \in \mathcal{V}$ and $v \in \mathcal{V}$. The elements of $\mathcal{J}$ are the available *colors*. A *vertex coloring problem* is the problem of finding a coloring $c(v)$ with the least possible number of colors $|\mathcal{J}|$. △

Definition 14.17 (*Distance-k Coloring*) A *distance-k coloring* of a graph $G(\mathcal{V}, \mathcal{B})$ is a map $c : \mathcal{V} \to \mathcal{J}$, for which $c(v_0) \neq c(v_k)$ for every path $P(\mathcal{V}_p, \mathcal{B}_p) \subseteq G(\mathcal{V}, \mathcal{B})$ of length k with

$$\mathcal{V}_p = \{v_0, \ldots, v_k\} \subseteq \mathcal{V}, \qquad \mathcal{B}_p = \{v_0 v_1, \ldots, v_{k-1} v_k\} \subseteq \mathcal{B}.$$

A *distance-k coloring problem* is the problem of finding a distance-k coloring $c(v)$ with the least possible number of colors $|\mathcal{J}|$. △

References

1. Diestel R (2005) Graph theory. Grad texts in math (2005)
2. George A, Gilbert JR, Liu JW (1993) Graph theory and sparse matrix computation, vol 56. Springer
3. Golumbic MC (2004) Algorithmic graph theory and perfect graphs, vol 57. Elsevier
4. Wilson R (1996) Introduction to graph theory. Longman

Index

T.J. Böhme and B. Frank, *Hybrid Systems, Optimal Control and Hybrid Vehicles*,
Advances in Industrial Control, DOI 10.1007/978-3-319-51317-1