Numerical Solution of Quasi-Linear Differential-Algebraic Equations and Industrial Simulation of Multibody Systems

vorgelegt von

Dipl.-Math.techn. Andreas Steinbrecher

von der Fakultät II - Mathematik und Naturwissenschaften
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften
- Dr. rer. nat. -

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender:	Prof. Dr. Dirk Ferus
Berichter:	Prof. Dr. Volker Mehrmann
Berichter:	Prof. Dr. Peter Kunkel
Berichter:	Prof. Dr. Martin Arnold

Tag der wissenschaftlichen Aussprache: 09.03.2006

Berlin 2006
D 83

Acknowledgment

This thesis is devoted to the three most important women in my life,

my mother Marlies Steinbrecher,
my sister Christin Steinbrecher, and
my partner in life Nicolle Bräunig.

At this point, I would like to express my gratitude to my advisor Prof. Dr. Volker Mehrmann for initiating this work and for his valuable criticism and discussions, his continual encouragement, and his endurance, but also for the freedom he has given me in my research. Furthermore, I would like to thank Prof. Dr. Peter Kunkel for valuable hints and discussions on the topic of differential-algebraic equations and the strangeness concept and, I would like thank Prof. Dr. Martin Arnold for interesting and fruitful discussions with respect to multibody systems.
I have enjoyed the convenient and unique working environment provided of the members of the research field Numerical Analysis of the Technical University Berlin. Therefore, my thanks go to all my colleagues who were always willing to lend me an open ear to all intents and purposes. In particular, I would like to thank Simone Bächle, Martin Bodestedt, Falk Ebert, Dr. Michael Karow, Dr. Christian Mehl, Sonja Schlauch, Ingo Seufer, and Prof. Dr. Caren Tischendorf for fruitful discussions, for the proofreading of this thesis, and in particular, for their hints and advice concerning the English language.
Also I have to express my deep gratitude to my mother, my sister, and my partner in life for their endurance, encouragement, and support over the years. Without my family's encouragement, love and support, this work would never have been accomplished.

Zusammenfassung

In dieser Dissertationsschrift wird die numerische Integration von allgemeinen quasi-linearen differentiell-algebraischen Gleichungen (DAEs) hinsichtlich der numerischen Simulation von Mehrkörpersystemen untersucht. Basierend auf den Resultaten wurden zwei neue Algorithmen zur numerischen Integration allgemeiner Bewegungsgleichungen, wie sie in der industriellen Simulation vorherrschen, entwickelt und implementiert. Im Vordergrund dieser Arbeit stehen insbesondere drei Schwerpunkte, die nachfolgend erläutert werden.

Der erste Schwerpunkt liegt in der Untersuchung quasi-linearer DAEs in ihrer allgemeinsten Form. Diese können sowohl über- als auch unterbestimmt sein und ihre führende Matrix und rechte Seite hängen dabei sowohl vom Zustand als auch von einer vorgegebenen Steuerfunktion ab. Als Grundlage der Untersuchung wird eine iterativ arbeitende Prozedur entwickelt, die eine Analyse der DAE ausschließlich basierend auf relevanten Gleichungen sowie deren Ableitungen erlaubt. Hierdurch ist der Aufwand gegenüber anderen Analysekonzepten deutlich reduziert. Diese Prozedur ermöglicht die Bestimmung grundlegender Eigenschaften quasi-linearer DAEs, insbesondere des maximalen Zwanglevels, was bei regulären DAEs dem Differentiationsindex gleich kommt. Basierend auf dieser Prozedur wird eine Regularisierungsmethode für allgemeine quasi-lineare DAEs entwickelt, die auf eine äquivalente DAE führt, welche aufgrund ihrer vorteilhaften Eigenschaften als Basis einer robusten numerischen Integration sehr geeignet ist. Basierend auf impliziten Runge-Kutta-Methoden wird ein Diskretisierungsverfahren entwickelt, welches in effizienter und einfacher Weise die Regularisierung der DAE und das Lösen der entstehenden linearen Systeme miteinander verbindet.

Die Untersuchung allgemeiner Bewegungsgleichungen für allgemeine mechanische Mehrkörpersysteme bildet den zweiten Schwerpunkt. Die Ergebnisse vorhergegangener Untersuchungen hinsichtlich allgemeiner quasi-linearer DAEs werden auf die Bewegungsgleichungen unter größtmöglicher Ausnutzung von Strukturen der Bewegungsgleichungen angewandt. Mit Hilfe der Prozedur werden die grundlegenden Eigenschaften der Bewegungsgleichungen bestimmt und die Existenz und Eindeutigkeit von Lösungen untersucht. Zudem wird eine Regularisierung erarbeitet, die in einfacher und effizienter Weise durchführbar ist und als Grundlage einer effizienten und robusten numerischen Integration dient.

Ausgehend von den Resultaten der Untersuchung der Bewegungsgleichungen werden als dritter Schwerpunkt zwei neue Integrationsalgorithmen entwickelt, welche die Integration allgemeiner, in industriellen Anwendungen vorherrschender Bewegungsgleichungen ermöglichen. Dabei wird, aufbauend auf einer Kombination aus entwickelter Regularisierungstechnik und Diskretisierung, unter Ausnutzung der vorherrschenden Struktur eine sowohl robuste als auch effiziente Integration ermöglicht. Versteckte Zwangsbedingungen wurden dabei ebenso berücksichtigt wie eventuelle Lösungsinvarianten. Abschließend wird die Effizienz und Robustheit dieser numerischen Algorithmen anhand mehrerer Anwendungsbeispiele demonstriert und mit herkömmlichen numerischen Algorithmen verglichen.

Abstract

In this thesis we discuss the numerical integration of general quasi-linear differential-algebraic equations (DAEs) in view of the numerical simulation of multibody systems. Based on the obtained results we develop and implement two new numerical algorithms for the numerical integration of general equations of motion as they appear in industrial applications. This thesis is focused on three topics as elucidated in the following.

The first topic involves the consideration of quasi-linear DAEs in their most general form which may be underdetermined or overdetermined. Their leading matrix and the right-hand side depend on the state as well as on a given control function. As a base for the analytical considerations we develop an iterative procedure which enables us to investigate the quasi-linear DAE. This procedure is only based on relevant equations and their derivatives. This fact reduces the effort significantly in comparison with the use of other analysis concepts. This procedure allows the determination of characteristic properties of a quasi-linear DAE, in particular, the maximal constraint level which corresponds to the differentiation index in case of regular DAEs. Furthermore, based on this procedure, a regularization technique for general quasi-linear DAEs is developed. This regularization technique yields an equivalent DAE which is suited for a robust numerical integration because of its favorable properties. Based on implicit Runge-Kutta methods we develop a discretization technique which connects in an efficient and simple way the developed regularization technique with the solution of the occuring linear algebraic systems.

The investigation of general equations of motion for general mechanical multibody systems is the second topic of this thesis. The obtained results with respect to general quasi-linear DAEs will be applied to the equations of motion by maximal exploitation of their structure. By use of the procedure the characteristic properties of the equations of motion will be determined and the existence and the uniqueness of a solution will be discussed. Furthermore, a simple and efficient regularization technique adapted to the equations of motion will be developed. This serves as a base for an efficient and robust numerical integration of the equations of motion.

The third topic in this thesis is the development and the implementation of two new integration algorithms for the numerical integration of equations of motion in their general form as discussed previously. These algorithms are based on the combination of the regularization and discretization technique discussed above and exploit the structure of the equations of motion. Therefore, they allow an efficient and robust numerical integration. Hidden constraints as well as possible solution invariants are respected. Concluding, the efficiency and robustness of both algorithms applied to several examples are demonstrated in comparison with other commonly used numerical algorithms.

Contents

Abbreviations

BDF	backward differential formulas, BDF methods
DAE	differential-algebraic equation, see Chapter 3
d-index	differentiation index, see Definition 3.2.3
EoM	equations of motion, , see Chapter 4
EoM_0	s-index-0-formulation of the equations of motion, see (4.82)
EoM_1	s-index-1-formulation of the equations of motion, see (4.81)
GGL formulation	Gear-Gupta-Leimkuhler formulation, see Section 4.6.2.1 and [68]
`NACCPT`	number of accepted steps during the integration process
`NBSUB`	number of backward substitution the integration process
`NCRJCT`	number of rejections caused from convergence failures during the integration process
`NEDEC`	number of complete decompositions during the integration process
`NEOM`	number of function evaluations of the right-hand side of the equations of motion during the integration process
`NERJCT`	number of rejections caused from error test failures during the integration process
`NJAC`	number of Jacobian evaluations of the right-hand side of the equations of motion during the integration process
`NMAS`	number of mass matrix evaluations of the equations of motion during the integration process
`NPDEC`	number of predecompositions during the integration process
`NSEL`	number of determinations of the selector during the integration process
ODE	ordinary differential equations, see (3.1a)
pEoM_1	projected-s-index-1-formulation of the equations of motion, see (4.123)
p-index	perturbation index, see Definition 3.2.13
psfEoM	projected-strangeness-free-formulation of the equations of motion, see Section 4.6.2.3
s-index	strangeness index, see Definition 3.2.9
SV-decomposition	singular value decomposition, see Lemma A.2.12
UDE	underlying differential equations, see Definition 3.5.23
uODE	underlying ordinary differential equations, see Definition 3.5.23

Notation

$\dot{x}, \ddot{x}, \dddot{x}, x^{(i)}$	total derivative(s) of $x(t)$ with respect to t , i.e., $\dot{x}(t) = \frac{\mathrm{d}}{\mathrm{d}t}x(t)$, $\ddot{x}(t) = \frac{\mathrm{d}^2}{\mathrm{d}t^2}x(t)$, $\dddot{x}(t) = \frac{\mathrm{d}^3}{\mathrm{d}t^3}x(t)$, $x^{(i)}(t) = \frac{\mathrm{d}^i}{\mathrm{d}t^i}x(t)$, see Notation 2.0.1
$\cdot_{,\cdot} = \frac{\partial \cdot}{\partial \cdot}$	partial derivative, notation with comma-operator, e.g., the partial derivative of $g(x, y)$ with respect to x is denoted by $g_{,x}(x, y) = \frac{\partial g}{\partial x}(x, y)$, see Notation 2.0.1
A^+	Moore-Penrose pseudo-inverse, see Definition A.2.15
$\otimes$	Kronecker product, see Definition A.2.19
$\|\| \cdot \|\|$	norm, in particular, the Hölder norm, see Definition A.1.5
$\|\| \cdot \|\|_{sc}$	weighted root square norm, see (5.2)
$[\![\ ]\!]$	tensor product, see Definition 2.2.1
$0_n, 0_{mn}$	zero matrix in $\mathbb{R}^{n,n}$ and in $\mathbb{R}^{m,n}$, respectively

$\mathfrak{A}_l(t)$	right-hand side matrix of the derivative array of a linear DAE, see (3.10b)
$\hat{b}_C, \hat{b}_D$	right-hand side of the selected linear systems (3.132) and (3.133)
$\widetilde{b}_C, \widetilde{b}_D$	right-hand side of the unselected linear systems (3.137) and (3.138)
$\hat{B}$	part of the leading matrix associated with the differential part of the selected linear systems (3.132) and (3.133)
$\widetilde{B}$	part of the leading matrix associated with the differential part of the unselected linear systems (3.137) and (3.138)
$\hat{C}$	leading matrix associated with the constraint part of the selected linear systems (3.132) and (3.133)
$\widetilde{C}$	leading matrix associated with the constraint part of the unselected linear systems (3.137) and (3.138)
$\mathcal{C}, \mathcal{C}^i, \mathcal{C}^\infty$	set of continuous and i-times continuously differentiable mappings, respectively, see Definitions A.1.4 and A.1.10
$\mathbb{C}$	set of complex numbers
$\hat{D}$	part of the leading matrix associated with the differential part of the selected linear systems (3.132) and (3.133)
$\widetilde{D}$	part of the leading matrix associated with the differential part of the unselected linear systems (3.137) and (3.138)
$\mathfrak{E}_l(t)$	leading matrix of the derivative array of a linear DAE, see (3.10a)
$\mathfrak{F}_l$	derivative array of order l, see Definition 3.2.1
$\breve{\mathfrak{F}}_l$	reduced derivative array of order l, see Definition 3.2.1
$\widetilde{\mathfrak{F}}_l$	minimal reduced derivative array of order l, see Definition 3.5.39
$g(p, s, u)$	holonomic constraints on position level, see (4.43g)
$g^I(p, v, s, u)$	holonomic constraints on velocity level, see (4.50)
$g^{I\!I}(p, v, r, w, s, \lambda, \mu, u)$	holonomic constraints on acceleration level, see (4.51)
$G(p, s, u)$	holonomic constraint matrix, see (4.18) and Lemma 4.1.19
G_λ	abbreviation $G_\lambda = f_{,\lambda} - f_{,w} d_{,w}^{-1} d_{,\lambda} - Z^T G^T$, see (4.46)
$\breve{h}(p, v, s, u)$	nonholonomic constraints on velocity level, we have $\breve{h}(p, v, s, u) = H(p, s, u)Z(p)v + h(p, s, u)$, see (4.43f)
$h^I(p, v, r, w, s, \lambda, \mu, u)$	nonholonomic constraints on acceleration level, see (4.53a)
$H(p, s, u)$	nonholonomic constraint matrix, see (4.18)
H_μ	abbreviation $H_\mu = f_{,\mu} - f_{,w} d_{,w}^{-1} d_{,\mu} - Z^T H^T$, see (4.46)
$I,\ I_n$	identity matrix ($\in \mathbb{R}^{n,n}$)
$\mathbb{I} = [t_0, t_f]$	domain of the independent variable t of differential equations and equations of motion
$\mathcal{I}_k$	$\mathcal{I} = \begin{bmatrix} I_n & 0_n & \cdots & 0_n \end{bmatrix}^T \in \mathbb{R}^{(k+1)n,n}$, $I_n, 0_n \in \mathbb{R}^{n,n}$, see Hypothesis 3.2.6
$\mathfrak{K}_l(t)$	inhomogeneity of the derivative array of a linear DAE, see (3.10c)
$\mathbb{L}_\nu$	set of solutions of the derivative array $\mathfrak{F}_\nu$ of order ν, see (3.13)
$\widetilde{\mathbb{M}}_0,\ \widetilde{\mathbb{M}}_i$	manifold of level 0 and hidden manifolds of level $i = 1, 2, ...$, see Procedure 3.5.11
m_1^i, m_2^i	dimension of the differential part and the constraint part (i.e., hidden constraints) of the intermediate DAE (3.40) in Procedure 3.5.11, respectively.
$\hat{m}_C, \hat{m}_D$	dimension of the constraint part and of the differential part of the discretized projected-strangeness free formulation, respectively, see Page 73

$\widetilde{m}_C, \widetilde{m}_D$	dimension of the constraint part and of the differential part of the discretized reduced derivative array, respectively, see Page 74
m_e	number of equations describing solution invariants, see (4.28)
$\mathbb{M}_i$	solution manifold up to level i, i.e., , $\mathbb{M}_i = \bigcap_{j=0}^{i} \widetilde{\mathbb{M}}_i$, $i \in \mathbb{N}_0$, see Procedure 3.5.11
$\mathbb{M},\ \mathbb{M}_p,\ \mathbb{M}_v,\ \mathbb{M}_a$	solution, position, velocity, acceleration manifold
n	number of state variables of a system of DAEs
n_f	degrees of freedom, Remark 4.1.8
n_{f_p}	positional degrees of freedom, see Definition 4.1.7
n_{f_v}	motional degrees of freedom, see Definition 4.1.7
n_p	number of position variables of a system of equations of motion
n_r	number of hydraulic position variables of a system of equations of motion
n_s	number of contact variables of a system of equations of motion
n_u	number of control variables of a system of equations of motion
n_w	number of additional variables of a system of equations of motion
n_v	number of velocity variables of a system of equations of motion
n_λ	number of Lagrange-multipliers with respect to holonomic constraints of a system of equations of motion
n_μ	number of Lagrange-multipliers with respect to nonholonomic constraints of a system of equations of motion
$\mathbb{N}$	set of natural numbers (excluding 0)
$\mathbb{N}_0$	set of natural numbers (including 0)
$\mathcal{O}$	Landau symbol, see Definition A.1.11
$p(t)$	position variables of a system of equations of motion depending on t
$r(t)$	hydraulic variables of a system of equations of motion depending on t
$r_G,\ r_H$	rank of G and rank of H respectively, i.e., $r_g = \operatorname{rank}(G)$, $r_H = \operatorname{rank}(H)$, see (4.45)
$\overset{*}{Q}_R, Q_R, Q_I$	index reducing decomposition matrices, see (3.163b), (3.165b), and (3.168b), respectively
$\mathbb{R}$	set of real numbers
$s(t)$	contact variables of a system of equations of motion depending on t
S_C, S_D	kinematic and dynamic selector, respectively, see Definitions 3.5.43 and 3.5.44
$\widetilde{S}_C, \widetilde{S}_D$	discrete kinematic and discrete dynamic selector, respectively, see Definitions 3.5.67 and 3.5.68
S_p, S_v	dynamic selector concerning the kinematical and the dynamical equations of motion, respectively, see Section 4.6.2.3 and Condition (4.110)
S_λ, S_μ	kinematic selector concerning the holonomic constraints and the nonholonomic constraints, respectively, see Section 4.2 and Condition (4.66)
$\mathbb{S}(x,\epsilon), \bar{\mathbb{S}}(x,\epsilon)$	open and closed sphere or neighborhood, see Definition A.1.6
t	independent variable of differential equations and equations of motion
$t_0,\ t_f$	initial point and final point of the domain $\mathbb{I} = [t_0, t_f]$ of the independent variable t

$u(t)$	control variable of a system of DAEs or equations of motion depending on t, see (3.2) or (4.40), (4.41), (4.43)
$\mathfrak{u}^i(t)$	vector of control variables and its derivatives, i.e., $\mathfrak{u}^i = [u^T, \dot{u}^T, ..., (u^{(i)})^T]^T$, see Notation 3.1.4
$\hat{U}^*$	particular control, see Section 3.5.4.2
$\mathbb{U}$	set of admissible control variables u
$\mathbb{U}^{(i)}$	set of admissible i-th derivatives $u^{(i)}$ of the control variables u, see Notation 3.1.4
$\mathbb{U}^i$	direct sum of the sets $\mathbb{U}^{(j)}$, $j = 0, ..., i$, see Notation 3.1.4
$v(t)$	velocity variables of a system of equations of motion depending on t
$\mathbb{V}$	variety, see Definition 2.3.4
$w(t)$	additional variables of a system of equations of motion depending on t
$x(t)$	state variable of a system of DAEs or equations of motion depending on t, see (3.2) or (4.40)(4.40), (4.41), (4.43)
$\mathfrak{x}^i(t)$	vector of state variables and its derivatives, i.e., $\mathfrak{x}^i = [x^T, \dot{x}^T, ..., (x^{(i)})^T]^T$, see Notation 3.1.4
X^*	particular state, see Section 3.5.4.2
$X_i, \mathfrak{X}, X_i', \mathfrak{X}'$	stages of the Runge-Kutta method, see Section 3.5.4.1
$\mathbb{X}$	set of admissible unknown variables x
$\mathbb{X}^{(i)}$	set of admissible i-th derivatives $x^{(i)}$ of the unknown variables x, see Notation 3.1.4
$\mathbb{X}^i$	direct sum of the sets $\mathbb{X}^{(j)}$, $j = 0, ..., i$, see Notation 3.1.4
$Y_i, \mathfrak{Y}, Y_i', \mathfrak{Y}'$	shifted stages of the Runge-Kutta method, see Section 3.5.4.1
$Z_i, \mathfrak{Z}$	transformed stages of the Runge-Kutta method, see Section 3.5.4.1
$\mathbb{Z}$	set of integers
$\gamma(t)$	drift function of the holonomic constraints depending on t, see Definition 4.4.1
$\zeta(t)$	Lagrange-multipliers with respect to constraints depending on t, $\zeta \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{n_\zeta})$, see (4.15)
$\eta(t)$	drift function of the nonholonomic constraints depending on t, see Definition 4.4.1
$\vartheta(t)$	additional Lagrange-multipliers for the GGL formulation depending on t, see (4.103)
$\lambda(t)$	Lagrange-multipliers with respect to holonomic constraints of a system of equations of motion depending on t, $\lambda \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{n_\lambda})$, see (4.18)
$\mu(t)$	Lagrange-multipliers with respect to nonholonomic constraints of a system of equations of motion depending on t, $\mu \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{n_\mu})$, see (4.18)
ν_c	maximal constraint level, see Definition 3.5.27
ν_d	differentiation index, see Definition 3.2.3
ν_g	global index, see Section 3.2
ν_n	index of nilpotency, see Section 3.2
ν_p	perturbation index, see Definition 3.2.13
ν_s	strangeness index, see Definition 3.2.9
ν_t	tractability index, see Section 3.2
ν_u	uniform index, see Section 3.2

Chapter 1

Introduction

Mechanical systems have played an important role for the development of human civilization. Simple mechanical systems like the lever, the wheel, or the cable winch were the first attainments which facilitated the construction of large buildings and monuments. Further developments to more complex mechanical systems, like doors, windows, simple bridges, transportation vehicles, or ships were the consequence. However, all these mechanical systems were built without detailed knowledge about their dynamical behavior just by trial and error.
But the interest in the understanding and description of dynamical processes was growing, such that in the 17th century arose a new field of natural science, the *analytical mechanics*. Analytical mechanics considers nonrelativistic mechanics of mass points and rigid bodies. The most famous celebrities related to analytical mechanics are Galilei[1], Huygens[2], and foremost, Newton[3].
The so-called *Newtonian mechanics* is mainly based on the three Newton axioms which were published in 1687 in [128]. The central equation of the Newton mechanics is the second Newton axiom $F = m\ddot{x}$ which describes the relation between a force F acting on a mass point with mass m and its acceleration $\ddot{x}$. This was the first time in the history of natural science that a mathematical foundation for the description and mathematical computation of mechanical observations and experiments were avaliable. However, from the current point of view, the Newtonian mechanics has two disadvantages. First, for the description of constrained motions, the constraint forces must be computed very laboriously. Therefore, the establishing of the equations is very difficult. Secondly, the transformation from cartesian coordinates to other coordinates is rather complicated.
In 1788, Lagrange[4] published a new concept of analytical mechanics in [110] which avoids these disadvantages of the Newtonian mechanics. Today it is known as *Lagrangian mechanics* and it is more abstract than Newtonian mechanics. The principle items of the Lagrangian mechanics are the Lagrange equations of type one and type two. These equations form the basis for further developments in the mechanics. Because of their simplicity and their simple extension to holonomic as well as nonholonomic constraints in general, they are one of the most suitable formalisms to determine the *equations of motion*, i.e., the equations which describe the motion of mechanical systems.
The derivation of the Lagrange equations is based (besides the second Newton ax-

[1] Galileo Galilei (born 1564 in Arcetri, Italy - died 1642 in Arcetri, Italy)
[2] Christiaan Huygens (born 1629 in The Hague, Netherlands - died 1695 in The Hague, Netherlands)
[3] Sir Isaac Newton (born 1643 in Woolsthorpe, Lincolnshire, England - died 1727 in London, England)
[4] Joseph-Louis Lagrange (born 1736 in Turin, Sardinia-Piedmont (now Italy) - died 1813 in Paris, France)

iom) on the *d'Alembert's*[5] *principle of virtual displacements*, e.g., see [17, 148, 174, 180], that results in the fact that constraint forces do not introduce any work in the mechanical system, a fact that cannot be derived from Newtonian mechanics. By application of d'Alembert's principle the constraint forces disappear from the equations.
The Lagrangian mechanics can also be derived from the *Hamilton*[6] *principle of least action*, e.g., see [17, 148, 174, 180]. Hamilton developed this variational principle in 1823. It is based on the fact that the motion of a mechanical system acts in an extreme way. In particular, the Hamilton principle is equivalent to the Lagrange equations of type two.
However, the derivation of the model equations describing the motion of mechanical systems is not the largest problem in the investigation of mechanical systems. Rather, the solution of these model equations was and often still is a difficult problem. Furthermore unfortunately, an analytical solution is only in rarest cases possible.
Independent of the progress of mathematical modeling of mechanical systems and the difficulties in the solution of the model equations, the progress in the development and in the construction of more and more complex mechanical systems was unrestricted. Already in 1769 James Watt[7] constructed the first industrially applicable steam engine, which is seen as the first important invention of the 18th century. In 1771 Richard Trevithick constructed the first locomotive, in 1886 Karl Benz[8] constructed the first automobile, and 1903 was the beginning of modern aeronautics with the first motorized flight by Orville[9] und Wilbur[10] Wright.
But the problem in the computation of the dynamical behavior of complex mechanical systems were unchanged or impossible because of the amount and complexity of computation. In 1941 Konrad Zuse[11] constructed the first electro-mechanical computer which in its modern form offered the possibility to master the amount and the complexity of the accruing computations in a numerical way.
In the 60s of the 20th century the demand for more complex and more detailed models increased. In particular, in the aircraft and space industry such models were of growing interest. This demand was favored by the development of more powerful computers which made it possible to handle such large models as they are necessary in these technologies. The *multibody system approach* provides the basic methodology [86, 147, 154] for modeling such mechanical systems. During this period a new research field of mechanics arose, the *multibody dynamics*. Favored by the computer technology, the first numerical formalisms for the description of multibody systems were developed and presented in [91].
In the 80s the first software packages were developed which include the modeling as well as the numerical simulation and animations, see [154, 160].
In the last decades the need for automatic multibody system simulators in industry increased. Therefore, much work has been spent in the numerical treatment of *differential-algebraic equations* describing the motion of multibody systems. Considerable advances in the theory and development of computational methods have brought a variety of codes that have been designed for the automatic modeling, analysis, and numerical integration of such systems, see, e.g., [3, 33, 85, 98, 122, 131, 144, 150, 154, 164].

[5] Jean d'Alembert (born 1717 in Paris, France - died 1783 Paris, France)
[6] Sir William Rowan Hamilton (born 1805 in Dublin, Ireland - died 1865 in Dunsink near Dublin, Ireland)
[7] James Watt (born 1736 in Greenock, Schottland - died 1819 in Heathfield, England)
[8] Karl Friedrich Benz (born 1844 in Karlsruhe, Germany - died 1929 in Ladenburg, Germany)
[9] Orville Wright (born 1871 in Dayton, Ohio, USA - died 1948 in Dayton, Ohio, USA)
[10] Wilbur Wright (born 1867 in Millville, Indiana, USA - died 1912 in Dayton, Ohio, USA)
[11] Konrad Zuse (born 1910 in Berlin-Wilmersdorf, Germany - died 1995 in Hünfeld, Germany)

Today, mechanical multibody systems are used in many applications like vehicle dynamics, aeronautics, biomechanics, and robotics, e.g., see the journal 'Multibody System Dynamics' published from Springer Science+Business Media B.V. and also, in particular, [11, 86, 147, 154, 166]. While the kinematical and dynamical modeling of mechanical systems is well developed and automated, see [1, 41, 147, 156], the numerical integration of the arising model equations is still a large field under consideration.
Since in any constrained mechanical system joints connecting bodies restrict their relative motion and impose constraints on the generalized coordinates, the model equations, e.g., derived from Lagrangian mechanics, form a system of differential-algebraic equations, which combine differential and algebraic equations. While the numerical behavior of ordinary differential equations is well understood, the analytical and numerical behavior of differential-algebraic equations as well as their theoretical background are more complicated and quite different from ordinary differential equations, see [32, 66, 69, 133].
Obviously, the class of ordinary differential equations is a special case of differential-algebraic equations and their numerical treatment is well understood. Furthermore, it was observed that certain classes of differential-algebraic equations create slight difficulties, while the treatment of other classes of differential-algebraic equations is more difficult. Therefore, it became necessary to classify differential-algebraic equations with respect to their difficulties arising in their analytical and numerical treatment. This classification was done by introduction of several index concepts [25, 66, 69, 73, 79, 82, 100, 102, 132, 138, 145], which offer a measure of difficulty in the treatment of differential-algebraic equations. It was observed that differential-algebraic equations of lower index behave similar as ordinary differential equations. Consequently, a large variety of numerical methods and software is developed to integrate lower index problems, e.g., [45, 79, 90, 135]. But the situation is quite different for higher index problems. Here, some problems like order reduction, drift, hidden constraints, consistency of initial values, and instabilities complicate the treatment of such systems. Unfortunately, the differential-algebraic equations arising from mechanical systems are such higher index problems.
A simple discretization of higher index problems typically did not lead to success and it was necessary to transform the higher index problems to lower index problems with more convenient properties regarding the numerical treatment. This process is called *regularization of differential-algebraic equations*. Consequently, regarding higher index differential-algebraic equations it becomes necessary to consider discretization methods in connection with some regularization methods. In the last three decades large effort has been spent in order to investigate differential-algebraic equations arising from mechanical systems and to avoid the numerical difficulties mentioned above. Also software has been developed which exploits the special structure of the systems arising in mechanical systems in several ways, e.g., [22, 118, 163, 164].
The aim of this thesis is to consider differential-algebraic equations in quasi-linear form of an arbitrary high index concerning numerical integration combined with a regularization strategy. In particular, these considerations are important and very helpful for the numerical integration of differential-algebraic equations arising in mechanical systems. These facts enable the development of a numerical code which integrates the differential-algebraic equations arising in mechanical systems in an efficient and stable way. Based on the investigations and the developed regularization of quasi-linear differential-algebraic equations two new codes will be developed. Furthermore, the efficiency of these codes will be demonstrated via several mechanical examples.

This thesis is organized as follows. Chapter 2 contains some preliminary results.

In Section 2.1 we will investigate smooth matrix operations that play an important role in the investigation of differential-algebraic equations and, in particular, in the treatment of the model equations arising in mechanical systems. Mainly, we will extend known results regarding a singular-value-like decomposition for matrix valued functions depending on several variables. Furthermore, the investigation of general multibody systems demands the analysis and numerical solution of systems of nonlinear equations. Here, the classification and regularization of redundant systems of nonlinear equations are important for further investigations, see Section 2.3. In Section 2.2 we will develop a convenient notation for the frequently used tensor-vector multiplication. Furthermore, frequently used definitions and tools from functional analysis and linear algebra are introduced in Sections A.1 and A.2, respectively.
Chapter 3 is concerned with differential-algebraic equations. In this chapter we will review and develop the most relevant topics for the analytical and numerical treatment of differential-algebraic equations with respect to the numerical integration of the equations of motion of multibody systems. In Section 3.1 we will review some preliminaries. Furthermore, in Section 3.2 we will review several index concepts of differential-algebraic equations, in particular, the so-called differentiation index (d-index) and the strangeness index (s-index). Since we will not restrict our investigations to regular differential-algebraic equations where, in particular, the number of equations equals the number of unknowns, it is necessary that we will base our investigations on the s-index which generalizes the d-index to such systems which are allowed to be over- or underdetermined. In our considerations the overdetermined case is of great importance. Furthermore, in Section 3.3 we will consider different types of linearizations of differential-algebraic equations.
A main topic of this thesis is the regularization of differential-algebraic equations, in particular, such of quasi-linear form with state depending leading matrix. We will give an overview of certain commonly used regularization techniques in Section 3.4. In particular, we will discuss the still widely used regularization by differentiation of the constraints and the recently developed strangeness concept. Subsequently, in Section 3.5 we will investigate quasi-linear differential-algebraic equations. In that section we will develop a procedure which can be used as general tool for the treatment of quasi-linear differential-algebraic equations. By use of this procedure it is possible to analyze quasi-linear differential-algebraic equations and to define the solution manifold, hidden constraints, the minimal reduced derivative array, and the maximal constraint level. Furthermore, this procedure provides an approach for a regularization technique of quasi-linear differential-algebraic equations of an arbitrary index which is similar to the strangeness concept but less technical in its execution. Furthermore, the regularization of general quasi-linear differential-algebraic equations based on this concept is discussed in detail. This regularization technique may be used for the numerical integration, e.g., by use of Runge[12]- Kutta[13] methods.
We will discuss the numerical integration of quasi-linear differential-algebraic equations in Section 3.5.4, i.e., we will develop a discretization scheme by use of Runge-Kutta methods applied to quasi-linear differential-algebraic equations, subsequently, we will discuss the efficient solution of the arising nonlinear systems by use of a simplified Newton iteration method. In particular, we will discuss the relation between decomposition techniques for the linear systems arising in the simplified Newton iteration process and the developed regularization technique. Based on these considerations, we will present a decomposition technique which corresponds to the discretization of the regularized quasi-linear DAE and therefore, is suitable for the use of the numerical integration of quasi-linear differential-algebraic equations of

[12]Carle David Tolmé Runge (born 1856 in Bremen, Germany - died 1927 in Göttingen, Germany)
[13]Martin Wilhelm Kutta (born 1867 in Pitschen, Upper Silesia (now Byczyna, Poland) - died 1944 in Fürstenfeldbruck, Germany)

arbitrary high index. Chapter 3 will conclude with an overview over numerical integration methods that are suited for several types of differential-algebraic systems, see Section 3.6.
In Chapter 4 we discuss equations of motion of multibody systems arising in industrial applications. In a large part of the literature equations of motion in standard form are discussed including the dynamical equations of motion according to some holonomic constraints. But in industrial applications more complex equations arise which include friction effects, contact force laws, dynamical force elements, nonholonomic constraints, and in some cases redundant constraints. Therefore, we will focus our investigations on the most general case including all these features. In Section 4.1.1 we will briefly discuss the modeling procedure according to Hamilton's principle of least action with respect to unconstrained motions. This principle leads to the Euler[14] equations, which form the equations of motion of a free multibody system as a set of ordinary differential equations. Afterwards, in Section 4.1.2, we will consider certain types of constraints, in particular, holonomic and nonholonomic constraints, which restrict the free motion. The consideration of constraints leads to the Euler-Lagrange equations as a set of differential-algebraic equations. The modeling procedure will be illustrated by several examples. Another important aspect in the numerical integration of dynamical systems are solution invariants, e.g., conservation of the total energy or conservation of the impulse, which are satisfied by every solution of the equations of motion. Unfortunately the numerical solution in general does not satisfy the solution invariants and, therefore, has to be considered very carefully as discussed in Section 4.1.4.
The deal of detail in the model equations representing the dynamical behavior of mechanical systems may be quite different depending on the necessity for different aspects of the motion and on the wish for information about the motion of mechanical systems. Furthermore, the equations of motion of mechanical systems provide a high measure of structure which should be exploited during the (numerical) integration process. Therefore, the model equations should be classified into several classes depending on the degree of information. In Section 4.1.5 a classification of the equations of motion is given. It ranges from the very simple form of the state space equations up to the general form of the equations of motion containing holonomic as well as nonholonomic constraints which may be redundant. Furthermore, contact points and hydraulic force elements are of great interest in the investigation of industrial applications. Subsequent to the investigation of the modeling of mechanical systems, we will investigate the properties of the equations of motion of the most general form with respect to an efficient and robust numerical integration in Section 4.2. In particular, in that section we will introduce several assumptions which guarantee a good behavior of the equations of motion in view of holonomic and nonholonomic constraints which may be possibly redundant. Furthermore, the procedure which is introduced in Section 3.5 for general quasi-linear differential-algebraic equations will be executed for the equations of motion in view of their regularity properties. Moreover, in Section 4.2 we will discuss the existence and the uniqueness of the solution of the equations of motion. In Section 4.3 briefly we will discuss the linearization of the equations of motion. One important property of the equations of motion is their higher index which complicates a direct numerical integration. A very popular and widely used (but not recommended) technique for the reduction of the index is just to replace the constraints by its derivatives. This leads to a lowered index but to a less restricted solution which yields an unfortunate behavior of the numerical solutions, i.e., the numerical solution drifts away from the set of solutions. This drift-off phenomenon will be discussed in detail in Section 4.4. With respect to the results of Sections 4.2 and 4.4 in Section 4.5 we

[14]Leonhard Euler (born 1707 in Basel, Switzerland - died 1783 in Petersburg, Russia)

postulate two paradigms which assume important properties of the model equations obtained from the modeling process and certain important properties of the used algorithms for the (numerical) integration of the equations of motion. Regarding the numerical integration of the equations of motion it is important to use a suitable formulation of the equations of motion which is obtained by several regularization techniques as basis for a suitable discretization method. In Section 4.6 such regularization techniques will be discussed. In particular, we direct our attention to the so-called Baumgarte stabilization and the so-called Gear-Gupta-Leimkuhler formulation of the equations of motion. Both will be generalized to the general form of the equations of motion, considered here. Furthermore, we will present a new regularization technique for the equations of motion which bases on a technique which determines all algebraic constraints that restrict the solution and defines the so-called solution manifold combined with the selection of the differential part, which describes the dynamical behavior inside the solution manifold. In this way the so-called projected-strangeness-free formulation of the equations of motion will be determined. This corresponds to an equivalent formulation of the equations of motion which has locally the same set of solutions and is very convenient for the numerical discretization using numerical methods suitable for the numerical integration of stiff ordinary differential equations. Chapter 4 concludes with an overview over numerical integration methods which are suited for the numerical simulation of mechanical systems.
Based on the obtained results we will present two new numerical integration methods `GEOMS` and `GMKSSOL` for the dynamical simulation of mechanical systems in Chapter 5. The algorithms `GEOMS` and `GMKSSOL` are based on the discretization of the projected-strangeness-free formulation of the equations of motion by use of the implicit Runge-Kutta method of type Radau[15] IIa of order 5. These algorithms are constructed for the efficient and robust numerical integration of the equations of motion of the most general form which often is investigated in industrial applications, including dynamical force elements, contact points, holonomic as well as nonholonomic constraints which may be redundant. Furthermore, the algorithm `GEOMS` offers the possibility to respect additionally provided information of invariant solutions such that these can be satisfied in an accurate way also in the numerical solution. In Sections 5.1 and 5.2 we will describe the features of the codes `GEOMS` and `GMKSSOL`, respectively, and in Section 5.3 we will present several numerical experiments which demonstrate the applicability and the performance of the code.
Finally, in Chapter 6 we will summarize and discuss the obtained results. Furthermore, we will point out several open problems that should be investigated in the future.

[15]Rodolphe Radau (born 1835 - died 1911, french astronomer and mathematician)

Chapter 2

Preliminaries

In this chapter we will discuss topics of importance for the consideration of nonlinear differential-algebraic equations, see Chapter 3, and of nonlinear model equations of mechanical systems, see Chapter 4. With respect to both topics, very important aspects are smooth matrix operations, e.g., smooth matrix decompositions, and the investigation of the properties of systems of nonlinear equations, see Section 2.1 and Section 2.3, respectively. Furthermore, in Section 2.2 we will introduce a simple notation for particular topics of the tensor calculus as they are useful for further considerations. Fundamentals associated with nonlinear functional analysis and fundamentals associated with linear algebra are reviewed in the appendix, see Section A.1 and Section A.2, respectively.
In the first instance we will introduce the following notation.

Notation 2.0.1 Let f be a differentiable function $f : \mathbb{X} \to \mathbb{R}^m$, $\mathbb{X} \subset \mathbb{R}^n$, see Definition A.1.9, and let x be a differentiable function $x : \mathbb{I} \to \mathbb{R}^n$, $\mathbb{I} \subset \mathbb{R}$ open. The (total) derivative of $x(t)$ with respect to t is denoted by $\dot{x}(t) = \mathrm{d}x(t)/\mathrm{d}t$. Furthermore, higher (total) derivatives with respect to t are denoted by $\ddot{x}(t) = \mathrm{d}^2x(t)/\mathrm{d}t^2$, $\dddot{x}(t) = \mathrm{d}^3x(t)/\mathrm{d}t^3$, and $x^{(i)}(t) = \mathrm{d}^ix(t)/\mathrm{d}t^i$ for $i \in \mathbb{N}_0$. Note the convention $x^{(0)}(t) = x(t)$. The (partial) derivative of $f(x)$ with respect to x is denoted by $f_{,x}(x) = \frac{\partial}{\partial x} f(x)$. The same notation is used for differentiable matrix functions. □

Notation 2.0.2 a) The continuous differentiability of functions and the set of continuously differentiable functions is defined in Definition A.1.10. Let $f : \mathbb{X} \times \mathbb{Y} \to \mathbb{R}^m$, $\mathbb{X} \subset \mathbb{R}^{n_x}$, and $\mathbb{Y} \subset \mathbb{R}^{n_y}$, then $f \in \mathcal{C}^{l_1,l_2}(\mathbb{X} \times \mathbb{Y}, \mathbb{R}^m)$ if $f(\cdot, y) \in \mathcal{C}^{l_1}(\mathbb{X}, \mathbb{R}^m)$ for all $y \in \mathbb{Y}$ and $f(x, \cdot) \in \mathcal{C}^{l_2}(\mathbb{Y}, \mathbb{R}^m)$ for all $x \in \mathbb{X}$.
b) If the smoothness of a function $f : \mathbb{X} \times \mathbb{Y} \to \mathbb{R}^m$ with respect to certain components is unknown or unimportant, e.g., with respect to the second component, then we will use the notation $f \in \mathcal{C}^{l_1,\cdot}(\mathbb{X} \times \mathbb{Y}, \mathbb{R}^m)$, and vice versa. □

Notation 2.0.3 Let $A = [a_{ij}] \in \mathbb{R}^{l,m}$ and $b = [b_i] \in \mathbb{R}^n$. Furthermore, let $I = [i_1, i_2, ..., i_p]$, $J = [j_1, j_2, ..., j_q]$, and $K = [k_1, k_2, ..., k_r]$ be three index vectors with $i_s \in \{1, ..., l\}$ for $s = 1, ..., p \leq l$, $j_s \in \{1, ..., m\}$ for $s = 1, ..., q \leq m$, and $k_s \in \{1, ..., n\}$ for $s = 1, ..., r \leq n$. Then we use the notation A_{IJ} and b_K for

$$A_{IJ} = \begin{bmatrix} a_{i_1j_1} & a_{i_1j_2} & \cdots & a_{i_1j_q} \\ a_{i_2j_1} & a_{i_2j_2} & \cdots & a_{i_2j_q} \\ \vdots & \vdots & \ddots & \vdots \\ a_{i_pj_1} & a_{i_pj_2} & \cdots & a_{i_pj_q} \end{bmatrix} \quad \text{and} \quad b_K = \begin{bmatrix} b_{k_1} \\ b_{k_2} \\ \vdots \\ b_{k_r} \end{bmatrix},$$

respectively. □

2.1 Smooth matrix operations

During the analysis and numerical solution of differential-algebraic equations and, in particular, of model equations of mechanical systems we will need some matrix operations with respect to smooth matrix functions. These matrix operations must fulfill some smoothness requirements. Smooth matrix operations like matrix decompositions are considered for example in [46, 73, 99, 146]. In this section we will recall some notation and important facts and we will generalize and extend some known results to matrix functions depending on more than one variable.
In the following considerations we will restrict ourselves to the space $\mathbb{R}^n$, although the following statements may also be valid in more general setting.

Lemma 2.1.1 *Let $A \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{n,n})$ where $\mathbb{X} \subset \mathbb{R}^{n_x}$ is an open subset. Furthermore, let $A(x)$ be nonsingular for all $x \in \mathbb{X}$. Then $A^{-1} \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{n,n})$.*

Proof: Since the determinant of $A(x)$ is nonzero for all $x \in \mathbb{X}$, the proof follows directly from the smoothness of the determinant of $A(x)$ and its minors and therefore of the adjugate matrix of $A(x)$, see Lemma A.2.8. □

Proposition 2.1.2 *Let $A \in \mathcal{C}(\mathbb{X}, \mathbb{R}^{n,n})$ and let $\mathbb{X} \subset \mathbb{R}^{n_x}$ be an open subset. Suppose that $A(x_0)$ is nonsingular for $x_0 \in \mathbb{X}$. Then, there exists an $\epsilon > 0$ sufficiently small with $A(x)$ nonsingular for all $x \in \mathbb{X}$ with $||x - x_0|| < \epsilon$.*

Proof: The proof follows from Lemma 2.1.1 using Lemma A.1.7. □

Lemma 2.1.3 *Suppose that $G \in \mathcal{C}(\mathbb{X}, \mathbb{R}^{m,n})$, $\mathbb{X} \subset \mathbb{R}^{n_x}$, $\operatorname{rank}(G(x)) = r$ for all $x \in \mathbb{X}$ and let $x_0 \in \mathbb{X}$. Furthermore, let $S \in \mathbb{R}^{r,m}$ be given such that*

$$\operatorname{rank}(SG(x_0)) = r.$$

Then there exists an $\epsilon > 0$ such that

$$SG \in \mathcal{C}(\mathbb{X}, \mathbb{R}^{r,n})$$

and $SG(x)$ has full (column) rank r for all $x \in \mathbb{X}$ with $||x - x_0|| < \epsilon$.

Proof: Define $A(x) = SG(x)G^T(x)S^T$, then the proof follows from Proposition 2.1.2. □

Theorem 2.1.4 *Let $A \in \mathcal{C}^l(\mathbb{I}, \mathbb{R}^{m,n})$ where $\mathbb{I} = [t_0, t_f] \subset \mathbb{R}$. Furthermore, assume that $\operatorname{rank}(A(t)) = r$ for all $t \in \mathbb{I}$. Then, there exist smooth matrix functions $U \in \mathcal{C}^l(\mathbb{I}, \mathbb{R}^{m,m})$ and $V \in \mathcal{C}^l(\mathbb{I}, \mathbb{R}^{n,n})$ such that*

$$U^T(t)A(t)V(t) = \begin{bmatrix} \Sigma(t) & 0 \\ 0 & 0 \end{bmatrix} \tag{2.1}$$

where $\Sigma \in \mathcal{C}^l(\mathbb{I}, \mathbb{R}^{r,r})$ is nonsingular for all $t \in \mathbb{I}$.

Proof: Since $\operatorname{rank}(A(t)) = r$ is constant for all $t \in \mathbb{I}$, for each $\tau_0 \in \mathbb{I}$ there exist index sets $I \in \mathbb{N}^r$ and $J \in \mathbb{N}^r$ such that the minor $[A]_{IJ}(\tau_0)$ (of r-th order) is nonzero, whereas all minors of higher order than r vanish. Furthermore, because of the smoothness of the minor $[A]_{IJ}(\tau_0)$, it does not vanish in a neighborhood

$\mathbb{S}(\tau_0, \epsilon_0) \subset \mathbb{I}$ where $\mathbb{S}(\tau_0, \epsilon_0) = \{\tau \in \mathbb{I} \ : \ ||\tau - \tau_0|| < \epsilon_0\}$, $\epsilon_0 > 0$, see Lemma 2.1.1. Therefore, there exist permutation matrices $P_0 = \left[\begin{array}{cc} P_{01} & P_{02} \end{array} \right] \in \mathbb{R}^{m,m}$ with $P_{01} \in \mathbb{R}^{m,r}$, $P_{02} \in \mathbb{R}^{m,m-r}$ and $Q_0 = \left[\begin{array}{cc} Q_{01} & Q_{02} \end{array} \right] \in \mathbb{R}^{n,n}$ with $Q_{01} \in \mathbb{R}^{n,r}$, $Q_{02} \in \mathbb{R}^{n,n-r}$ defined by the index sets I and J such that

$$\left[\begin{array}{c} P_{01}^T \\ P_{02}^T \end{array} \right] A(\tau) \left[\begin{array}{cc} Q_{01} & Q_{02} \end{array} \right] = \left[\begin{array}{cc} A_{IJ}(\tau) & A_{I\bar{J}}(\tau) \\ A_{\bar{I}J}(\tau) & A_{\bar{I}\bar{J}}(\tau) \end{array} \right],$$

with $A_{IJ}(\tau)$ nonsingular for all $\tau \in \mathbb{S}(\tau_0, \epsilon_0)$. With

$$U^T(\tau) = \left[\begin{array}{c} P_{01}^T \\ -A_{\bar{I}J}(\tau) A_{IJ}^{-1}(\tau) P_{01}^T + P_{02}^T \end{array} \right] \tag{2.2}$$

and

$$V(\tau) = \left[\begin{array}{cc} Q_{01} & -Q_{01} A_{IJ}^{-1}(\tau) A_{I\bar{J}}(\tau) + Q_{02} \end{array} \right], \tag{2.3}$$

we have a locally smooth decomposition

$$U^T(\tau) A(\tau) V(\tau) = \left[\begin{array}{cc} \Sigma(\tau) & 0 \\ 0 & 0 \end{array} \right],$$

with nonsingular $U \in \mathcal{C}^l(\mathbb{S}(\tau_0, \epsilon_0), \mathbb{R}^{m,m})$, $V \in \mathcal{C}^l(\mathbb{S}(\tau_0, \epsilon_0), \mathbb{R}^{n,n})$ and $\Sigma(\tau) = A_{IJ}(\tau)$ with $\Sigma \in \mathcal{C}^l(\mathbb{S}(\tau_0, \epsilon_0), \mathbb{R}^{r,r})$. The smoothness of U and V follows by (2.2), (2.3), and Lemma 2.1.1.

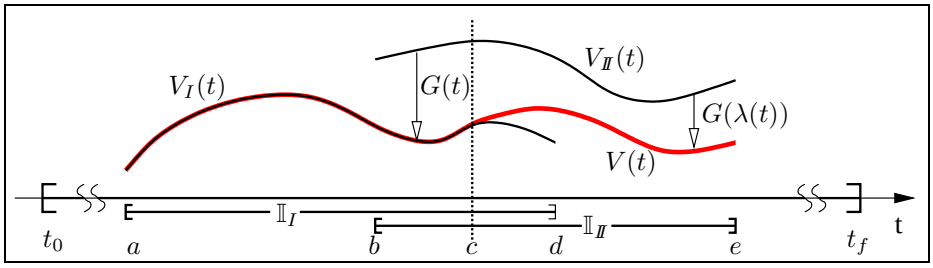

Figure 2.1: Smooth overlapping

Now, if we have two distinct points τ_I and $\tau_{I\!I}$ with $\mathbb{S}(\tau_I, \epsilon_I) \cap \mathbb{S}(\tau_{I\!I}, \epsilon_{I\!I}) \neq \emptyset$ we can define smooth decompositions of A for each interval $\mathbb{S}(\tau_I, \epsilon_I) = \mathbb{I}_I = (a, d)$ and $\mathbb{S}(\tau_{I\!I}, \epsilon_{I\!I}) = \mathbb{I}_{I\!I} = (b, e)$ separately, where $a < b < d < e$, see Figure 2.1. Let us denote them by subscripts I and $I\!I$ such that we have

$$U_I^T(\tau) A(\tau) V_I(\tau) = \left[\begin{array}{cc} \Sigma_I(\tau) & 0 \\ 0 & 0 \end{array} \right]$$

for all $\tau \in \mathbb{S}(\tau_I, \epsilon_I)$ and

$$U_{I\!I}^T(\tau) A(\tau) V_{I\!I}(\tau) = \left[\begin{array}{cc} \Sigma_{I\!I}(\tau) & 0 \\ 0 & 0 \end{array} \right]$$

for all $\tau \in \mathbb{S}(\tau_{I\!I}, \epsilon_{I\!I})$. Furthermore, let $c \in (b, d)$ be given. We define the nonsingular matrix functions

$$G(\tau) = \left[\begin{array}{cc} G_{11}(\tau) & G_{12}(\tau) \\ G_{21}(\tau) & G_{22}(\tau) \end{array} \right] \quad \text{and} \quad H(\tau) = \left[\begin{array}{cc} H_{11}(\tau) & H_{12}(\tau) \\ H_{21}(\tau) & H_{22}(\tau) \end{array} \right]$$

on $\tau \in [b,d]$ such that $V_I(\tau) = V_{I\!I}(\tau)G(\tau)$ and $U_I(\tau) = U_{I\!I}(\tau)H(\tau)$, we get $G_{12}(\tau) = 0$ and $H_{12}(\tau) = 0$, since range(V_{I2}) = range$(V_{I\!I2})$ and range(U_{I2}) = range$(U_{I\!I2})$, respectively. Furthermore, defining a smooth function $\lambda \in \mathcal{C}^l(\mathbb{I}_{I\!I}, [b,d])$ such that

$$\lambda(t) \begin{cases} = t, & \text{for } t \in [b,c], \\ \in [b,d], & \text{for } t \in [c,e], \end{cases}$$

with the smooth matrix functions $V \in \mathcal{C}^l(\mathbb{I}_I \cup \mathbb{I}_{I\!I}, \mathbb{R}^{n,n})$ and $U \in \mathcal{C}^l(\mathbb{I}_I \cup \mathbb{I}_{I\!I}, \mathbb{R}^{m,m})$ defined by

$$V(t) = \begin{cases} V_I(t), & \text{for } t \in [a,b], \\ V_I(t) = V_{I\!I}(t)G(t), & \text{for } t \in [b,c], \\ V_{I\!I}(t)G(\lambda(t)), & \text{for } t \in [c,e], \end{cases}$$

and

$$U(t) = \begin{cases} U_I(t), & \text{for } t \in [a,b], \\ U_I(t) = U_{I\!I}(t)H(t), & \text{for } t \in [b,c], \\ U_{I\!I}(t)H(\lambda(t)), & \text{for } t \in [c,e] \end{cases}$$

we get the desired smooth decomposition (2.1) of $A(t)$ for all $t \in \mathbb{I}_I \cup \mathbb{I}_{I\!I}$ with $\Sigma \in \mathcal{C}^l(\mathbb{I}_I \cup \mathbb{I}_{I\!I}, \mathbb{R}^{r,r})$ given by

$$\Sigma(t) = \begin{cases} \Sigma_I(t), & \text{for } t \in [a,b], \\ \Sigma_I(t) = H_{11}^T(t)\Sigma_{I\!I}(t)G_{11}(t), & \text{for } t \in [b,c], \\ H_{11}^T(\lambda(t))\Sigma_{I\!I}(t)G_{11}(\lambda(t)), & \text{for } t \in [c,e]. \end{cases}$$

Because of the existence of a finite covering of $\mathbb{I}$ by such $\mathbb{S}(\tau_i, \epsilon_i) = \mathbb{I}_i$, $i = 1, ..., s$, see Lemma A.1.14, the statement follows. □

Theorem 2.1.5 *Let $A \in \mathcal{C}^l(\mathbb{Y}, \mathbb{R}^{m,n})$ where $\mathbb{Y}$ is a convex subset defined by*

$$\mathbb{Y} = ([y_1^a, y_1^b] \times \cdots \times [y_{n_y}^a, y_{n_y}^b]) \subset \mathbb{R}^{n_y}. \tag{2.4}$$

Furthermore, suppose that rank$(A(y)) = r$ *for all $y \in \mathbb{Y}$. Then there exist $U \in \mathcal{C}^l(\mathbb{Y}, \mathbb{R}^{m,m})$ and $V \in \mathcal{C}^l(\mathbb{Y}, \mathbb{R}^{n,n})$ such that*

$$U^T(y)A(y)V(y) = \begin{bmatrix} \Sigma(y) & 0 \\ 0 & 0 \end{bmatrix} \tag{2.5}$$

where $\Sigma \in \mathcal{C}^l(\mathbb{Y}, \mathbb{R}^{r,r})$ is nonsingular for all $y \in \mathbb{Y}$.

Proof: We will only present a sketch of the proof. We discretize $\mathbb{I}_i = [y_i^a, y_i^b]$ by $\mathbb{I}_i^D = \{y_i^a = y_i^0, y_i^1, ..., y_i^{n_i} = y_i^b\}$ for all $i = 1, ..., n_y$ such that for every subset $\mathbb{Y} \cap \bigotimes_{i=1}^{n_y} (y_i^{k_i} - \epsilon_{k_i}, y_i^{k_i+1} + \epsilon_{k_i})$ for all possible $k_i \in \{0, ..., n_i - 1\}$, $i = 1, ..., n_y$, with appropriately given ϵ_{k_i}, we have a locally smooth decomposition (2.5) because of the smoothness and the constant rank of A.
Recursively, for $s = 1, ..., n_y$, in an analogous way as in the proof of Lemma 2.1.4 determine the smooth decomposition for all subsets intersecting in direction y_s resulting in local smooth decompositions (2.5) on the subsets

$$\mathbb{Y} \cap \left(\bigotimes_{i=1}^{s} \mathbb{I}_i \otimes \bigotimes_{i=s+1}^{n_y} [y_i^{k_i} - \epsilon_{k_i}, y_i^{k_i+1} + \epsilon_{k_i}] \right).$$

When the recursion terminates, we have found a global decomposition (2.5) on $\mathbb{Y} \cap \bigotimes_{i=1}^{n_y} \mathbb{I}_i = \mathbb{Y}$. □

Theorem 2.1.6 *Let $A \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{m,n})$ and let $\mathbb{X} \subset \mathbb{R}^{n_x}$ be $\mathcal{C}^l$-diffeomorphic to a subset $\mathbb{Y}$ as defined in (2.4). Furthermore, suppose that* $\operatorname{rank}(A(x)) = r$ *for all $x \in \mathbb{X}$. Then there exist a nonsingular matrix function $U \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{m,m})$ and a nonsingular matrix function $V \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{n,n})$ such that*

$$U^T(x)A(x)V(x) = \begin{bmatrix} \Sigma(x) & 0 \\ 0 & 0 \end{bmatrix}$$

for all $x \in \mathbb{X}$, where $\Sigma \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{r,r})$ is nonsingular for all $x \in \mathbb{X}$.

Proof: Since $\mathbb{X}$ is $\mathcal{C}^l$-diffeomorphic to a set $\mathbb{Y}$ as defined in (2.4), there exists a $\mathcal{C}^l$-diffeomorphism $g \in \mathcal{C}^l(\mathbb{Y}, \mathbb{X})$. Furthermore, we have $A(x) = A(g(y)) = \widetilde{A}(y)$ with $\widetilde{A} \in \mathcal{C}^l(\mathbb{Y}, \mathbb{R}^{m,n})$. From Theorem 2.1.5 the existence of a smooth decomposition

$$\widetilde{U}^T(y)\widetilde{A}(y)\widetilde{V}(y) = \begin{bmatrix} \widetilde{\Sigma}(y) & 0 \\ 0 & 0 \end{bmatrix},$$

follows with $y \in \mathbb{Y}$. Consequently, with $U(x) = \widetilde{U}(g^{-1}(x)) = \widetilde{U}(y)$, $V(x) = \widetilde{V}(g^{-1}(x)) = \widetilde{V}(y)$, and $\Sigma(x) = \widetilde{\Sigma}(g^{-1}(x)) = \widetilde{\Sigma}(y)$, we get a globally smooth decomposition of $A(x)$ on $\mathbb{X}$. □

In [73] the smoothness of orthogonal projections onto the kernel and onto the range of a matrix function $A \in \mathcal{C}^l(\mathbb{I}, \mathbb{R}^{n,n})$ is considered.
In the following, we will generalize these results to matrix functions $A \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{m,n})$ depending on more than one independent variable in parameterizable subsets $\mathbb{X}$ of $\mathbb{R}^{n_x}$.

Lemma 2.1.7 *Let $A \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{m,n})$ and let $\mathbb{X} \subset \mathbb{R}^{n_x}$ be $\mathcal{C}^l$-diffeomorphic to a subset $\mathbb{Y}$ as defined in (2.4). Furthermore, suppose that* $\operatorname{rank}(A(x)) = r$ *for all $x \in \mathbb{X}$. Then the orthogonal projections onto the kernel, cokernel, range, and corange of A, respectively, are uniquely defined and as smooth as A, i.e.,*

$$\begin{array}{rclrcl} P_{\operatorname{range}(A)} & \in & \mathcal{C}^l(\mathbb{Y}, \mathbb{R}^{m,m}), & P_{\operatorname{corange}(A)} & \in & \mathcal{C}^l(\mathbb{Y}, \mathbb{R}^{m,m}), \\ P_{\ker(A)} & \in & \mathcal{C}^l(\mathbb{Y}, \mathbb{R}^{n,n}), & P_{\operatorname{coker}(A)} & \in & \mathcal{C}^l(\mathbb{Y}, \mathbb{R}^{n,n}). \end{array} \tag{2.6}$$

Proof: From Theorem 2.1.6, we have the smooth decomposition of A on $\mathbb{X}$ with

$$\begin{aligned} U(x) &= \begin{bmatrix} U_1(x) & U_2(x) \end{bmatrix}, \text{ with } U_1(x) \in \mathbb{R}^{m,r},\ U_2(x) \in \mathbb{R}^{m,m-r}, \\ V(x) &= \begin{bmatrix} V_1(x) & V_2(x) \end{bmatrix}, \text{ with } V_1(x) \in \mathbb{R}^{n,r},\ V_2(x) \in \mathbb{R}^{n,n-r} \end{aligned}$$

for all $x \in \mathbb{X}$. Using Lemma A.2.11 we get the uniquely defined orthogonal projections onto the kernel, cokernel, range, and corange of A, respectively, from (1.2) as follows.

$$P_{\operatorname{range}(A)}(x) = U_1(x)(U_1(x)^T U_1(x))^{-1} U_1(x)^T, \tag{2.7a}$$
$$P_{\operatorname{corange}(A)}(x) = U_2(x)(U_2(x)^T U_2(x))^{-1} U_2(x)^T, \tag{2.7b}$$
$$P_{\ker(A)}(x) = V_2(x)(V_2(x)^T V_2(x))^{-1} V_2(x)^T, \tag{2.7c}$$
$$P_{\operatorname{coker}(A)}(x) = V_1(x)(V_1(x)^T V_1(x))^{-1} V_1(x)^T. \tag{2.7d}$$

The smoothness of the projections (2.7), i.e., (2.6), follows from Lemma 2.1.1. □

Lemma 2.1.8 *Let $A \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{m,n})$ and let $\mathbb{X} \subset \mathbb{R}^{n_x}$ be $\mathcal{C}^l$-diffeomorphic to a subset $\mathbb{Y}$ as defined in (2.4). Furthermore, let* $\operatorname{rank}(A(x)) = r$ *for all $x \in \mathbb{X}$. Then $A^+A \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{n,n})$ and $AA^+ \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{m,m})$.*

Proof: The proof follows directly from Lemma A.2.16 using Lemma 2.1.7. □

Next, we generalize the *moving frame algorithm* of [146] to general matrix functions $A \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{m,n})$ independent of an underlying function, i.e., it is not necessary that there exists a function $a \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^m)$, $\mathbb{X} \subset \mathbb{R}^{n_x}$, $l \geq 1$ such that $A(x) = \partial a(x)/\partial x$ and $n = n_x$.

Algorithm 2.1.9 (Smooth kernel of a smooth matrix function)
Let $A \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{m,n})$ and let $\mathbb{X} \subset \mathbb{R}^{n_x}$ be $\mathcal{C}^l$-diffeomorphic to a subset $\mathbb{Y}$ as defined in (2.4). Furthermore, suppose that $\operatorname{rank}(A(x)) = r$ for all $x \in \mathbb{X}$. Let $x_0 \in \mathbb{X}$ be given. Moreover, assume that we have chosen some method for computing a matrix $\widetilde{K}(x)$ with orthonormal columns that span the kernel of $A(x)$ at any point $x \in \mathbb{X}$. Of course, $\widetilde{K}$ is not expected to be continuous in x. Then in order to compute a smooth kernel function K at x proceed as follows.

$$1) \quad \text{compute } \widetilde{K}_0 = \widetilde{K}^T(x)\widetilde{K}(x_0) \tag{2.8a}$$

$$2) \quad \text{compute the singular value decomposition } \widetilde{K}_0 = C\Sigma B^T \tag{2.8b}$$

$$3) \quad \text{compute } K(x) = \widetilde{K}(x)CB^T \tag{2.8c}$$

The kernel function K is called *orthogonal moving frame* and the following Lemma 2.1.10 shows that $K \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{n,n-r})$ with $\operatorname{range}(K(x)) = \ker(A(x))$ for all $x \in \mathbb{X}$.

Lemma 2.1.10 *Let $A \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{m,n})$ and let $\mathbb{X} \subset \mathbb{R}^{n_x}$ be $\mathcal{C}^l$-diffeomorphic to a subset $\mathbb{Y}$ as defined in (2.4). Furthermore, suppose that* $\operatorname{rank}(A(x)) = r$ *for all $x \in \mathbb{X}$. Then the matrix function K defined in Algorithm 2.1.9 by (2.8c) is as smooth as A, i.e., $K \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{n,n-r})$, with* $\operatorname{range}(K(x)) = \ker(A(x))$ *for all $x \in \mathbb{X}$.*

Proof: From (2.8b) we get the polar decomposition, see Lemma A.2.13, of $\widetilde{K}_0$ in the form

$$\widetilde{K}_0 = QH, \tag{2.9}$$

with $Q = CB^T$ given in (2.8b) and $H = B\Sigma B^T$. It follows that $\widetilde{K}_0^T\widetilde{K}_0 = H^TQ^TQH = H^TH = H^2$, and from (2.8a) we get

$$H = (\widetilde{K}_0^T\widetilde{K}_0)^{1/2} = (\widetilde{K}^T(x_0)\widetilde{K}(x)\widetilde{K}^T(x)\widetilde{K}(x_0))^{1/2} \tag{2.10}$$

is nonsingular. Because of the nonsingularity, we get from (2.9) that $Q = \widetilde{K}_0H^{-1}$, i.e., with (2.8a) and (2.10) we have

$$Q = \widetilde{K}^T(x)\widetilde{K}(x_0)(\widetilde{K}^T(x_0)\widetilde{K}(x)\widetilde{K}^T(x)\widetilde{K}(x_0))^{-1/2}. \tag{2.11}$$

The orthogonal projection $P_{\operatorname{coker}(A)}(x)$ onto the cokernel of $A(x)$ is unique and from Lemma 2.1.7 it follows that $P_{\operatorname{coker}(A)} \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{n,n})$. In addition, we have from Lemma A.2.14 that $P_{\operatorname{coker}(A)}(x) = \widetilde{K}(x)\widetilde{K}^T(x)$. Therefore, from (2.11) we get

$$K(x) = \widetilde{K}(x)Q = P_{\operatorname{coker}(A)}(x)\widetilde{K}(x_0)(\widetilde{K}^T(x_0)P_{\operatorname{coker}(A)}(x)\widetilde{K}(x_0))^{-1/2}.$$

Since $\widetilde{K}^T(x_0)$ is independent of x and $P_{\operatorname{coker}(A)} \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{n,n})$, the assertion follows. □

Lemma 2.1.11 *Let $A \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{m,n})$ and let $\mathbb{X} \subset \mathbb{R}^{n_x}$ be $\mathcal{C}^l$-diffeomorphic to a subset $\mathbb{Y}$ as defined in (2.4). Furthermore, suppose that* $\operatorname{rank}(A(x)) = r$ *for all $x \in \mathbb{X}$. Then there exists $B \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{m,m-n})$ such that the matrix $\begin{bmatrix} A(x) & B(x) \end{bmatrix}$ is nonsingular for all $x \in \mathbb{X}$.*

Proof: From Lemma 2.1.10 the existence of a matrix function $K \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^{n,m-n})$ with $\text{range}(K(x)) = \ker(A^T(x))$ for all $x \in \mathbb{X}$ follows. Using $B(x) = K(x)$, the assertion follows. □

Lemma 2.1.12 *Let $B \in \mathcal{C}^{l_1,l_2}(\mathbb{X}_1 \times \mathbb{X}_2, \mathbb{R}^{m,n})$ with $l_1, l_2 \in \mathbb{N}_0$, $m \geq n$ and let $\text{rank}(B(x,y)) = n$ for all $(x,y) \in \mathbb{X}_1 \times \mathbb{X}_2$. Then there exists $V \in \mathcal{C}^{l_1,l_2}(\mathbb{X}_1 \times \mathbb{X}_2, \mathbb{R}^{n,m})$ such that*

$$V(x,y)B(x,y) \text{ is nonsingular for every } (x,y) \in \mathbb{X}_1 \times \mathbb{X}_2. \tag{2.12}$$

Proof: Setting $V(x,y) = B^T(x,y)$ with $V \in \mathcal{C}^{l_1,l_2}(\mathbb{X}_1 \times \mathbb{X}_2, \mathbb{R}^{n,m})$, we obtain that $V(x,y)B(x,y) = B^T(x,y)B(x,y)$ is nonsingular for every (x,y), since $B(x,y)$ has full rank for every $(x,y) \in \mathbb{X}_1 \times \mathbb{X}_2$. □

2.2 Tensor calculus

In the following chapters very often we have to investigate matrix valued functions and its derivatives. In particular, the differentiation of a matrix valued function $A(x)$ of size $m \times n$ with respect to $x \in \mathbb{R}^{n_x}$ leads to a tensor of level three. Furthermore, we have to treat the total derivative of $A(x(t))$ with respect to t. This leads to a tensor-vector product. Therefore, in the following, we will introduce a notation for the efficient dealing with such products and we will investigate some of its properties.

Definition 2.2.1 (Tensor-vector product of higher level) *Let $T \in \mathbb{R}^{m,n_1,\ldots,n_l}$ be a tensor of level $l+1$, i.e.,*

$$T = [T_{ij_1j_2\ldots j_l}], \qquad i = 1,\ldots,m,\ j_1 = 1,\ldots,n_1,\ \ldots,\ j_l = 1,\ldots,n_l,$$

and let $x^k \in \mathbb{R}^{n_{l-s+k}}$, $k = 1,\ldots,s \leq l$ be vectors. Then we use the notation

$$v = T[\![x^1, x^2, \ldots, x^s]\!] = [\sum_{j_l=1}^{n_l} \cdots \sum_{j_{l-s+1}=1}^{n_{l-s+1}} T_{ij_1\ldots j_{l-s}j_{l-s+1}\ldots j_l} x^1_{j_{l-s+1}} \cdot \ldots \cdot x^s_{j_l}]$$

for the tensor-vector product*, where $v = [v_{ij_1\ldots j_{l-s}}] \in \mathbb{R}^{m,n_1,\ldots,n_{l-s}}$ is a tensor of level $l-s+1$.*

Note, that the tensor-vector product is performed in such a way that the sum is taken over the s last indices of the tensor T, as illustrated in the following example.

Example 2.2.2 Let $T \in \mathbb{R}^{m,n_j,n_k,n_l}$ be a tensor of level 4, i.e.,

$$T = [T_{ijkl}], \qquad i = 1,\ldots,m,\ j = 1,\ldots,n_j,\ k = 1,\ldots,n_k,\ l = 1,\ldots,n_l,$$

and let $x \in \mathbb{R}^{n_k}$ and $y \in \mathbb{R}^{n_l}$ be two vectors. Then we have

$$v = T[\![x,y]\!] = [\sum_{l=1}^{n_l} \sum_{k=1}^{n_k} T_{ijkl} x_k y_l]$$

where $v = [v_{ij}] \in \mathbb{R}^{m,n_j}$ is a tensor of level 2. □

Notation 2.2.3 a) In addition to the complete tensor-vector product, we introduce a partial tensor-vector product in which we will use dots as place holders. For example, we have for the tensor $T \in \mathbb{R}^{m,n_1,...,n_4}$ of level 5 and two vectors $x^k \in \mathbb{R}^{n_k}$, $k = 2, 3$ the relation

$$v = T[\![\cdot, x^2, x^3, \cdot]\!] = [\sum_{j_3=1}^{n_3} \sum_{j_2=1}^{n_2} T_{ij_1j_2j_3j_4} x^2_{j_2} x^3_{j_3}],$$

where $v = [v_{ij_1j_4}] \in \mathbb{R}^{m,n_1,n_4}$ is a tensor of level 3.
b) If $T \in \mathbb{R}^{m,n}$ is a matrix, i.e., a tensor of level 2, and $x \in \mathbb{R}^n$ is a vector, then the matrix-vector product can be written as

$$Tx = T[\![x]\!].$$

c) Furthermore, if $T \in \mathcal{C}^1(\mathbb{Y}, \mathbb{R}^{m,n_1,...,n_l})$, $z \in \mathbb{R}^{n_{l+1}}$, and $x^k \in \mathbb{R}^{n_k}$, $k = 1, ..., l$ are vectors, independent of $y \in \mathbb{Y} \subset \mathbb{R}^{n_{l+1}}$, then we use the notation

$$\begin{aligned} (T(y)[\![x^1, x^2, ..., x^l]\!])_{,y} z &= T_{,y}(y)[\![x^1, x^2, ..., x^l, z]\!], \\ (T(y)[\![x^1, x^2, ..., x^l]\!])_{,y} &= T_{,y}(y)[\![x^1, x^2, ..., x^l, \cdot]\!]. \end{aligned}$$

□

Notation 2.2.4 If $T \in \mathbb{R}^{m,n_1,...,n_l}$ is a tensor of level $l+1$ and depends on $y \in \mathbb{R}^{n_{l+1}}$ then $T_{,y} \in \mathbb{R}^{m,n_1,...,n_l,n_{l+1}}$ is a tensor of level $l + 2$. In particular, for $l = 0$ and $z \in \mathbb{R}^{n_1}$ we have the relation $T_{,y} \in \mathbb{R}^{m,n_1}$ and the product with z can be written as

$$T_{,y} z = T_{,y}[\![z]\!].$$

In addition, for an arbitrary $l \in \mathbb{N}$, assuming that z^i does not depend on y^j for all $i, j = 1, ..., k$, then we have the relation

$$\begin{aligned} ((...((T_{,y^1} z^1)_{,y^2} z^2)...)_{,y^k} z^k) &= T_{,y^1 y^2 ... y^k}[\![z^1, z^2, ..., z^k]\!] \\ &= [\sum_{r_k=1}^{n_k} \cdots \sum_{r_1=1}^{n_1} \frac{\partial^k T_{ij_1...j_l}}{\partial y^1_{r_1} \cdots \partial y^k_{r_k}} z^1_{r_1} \cdots z^k_{r_k}]. \end{aligned}$$

□

Lemma 2.2.5 *Let $T \in \mathcal{C}^2(\mathbb{R}^{m_1} \times \mathbb{R}^{m_2}, \mathbb{R}^{m,n_1,...,n_l})$, then*

$$\check{T}(z^1, z^2, y^1, y^2) = T_{,z^1 z^2}(z^1, z^2)[\![y^1, y^2]\!] = T_{,z^2 z^1}(z^1, z^2)[\![y^2, y^1]\!]$$

and

$$\check{T} \in \mathcal{C}^0(\mathbb{R}^{m_1} \times \mathbb{R}^{m_2} \times \mathbb{R}^{m_1} \times \mathbb{R}^{m_2}, \mathbb{R}^{m,n_1,...,n_l})$$

with $z^i, y^i \in \mathbb{R}^{m_i}$ for $i = 1, 2$.

Proof: The proof follows directly from Notation 2.2.4 and the Schwarz[1] Theorem, e.g., see [56, 89]. □

[1] Herrmann Amandus Schwarz (born 1843 in Hermsdorf, Silesia (now Poland) - died 1921 in Berlin, Germany)

2.3 Systems of nonlinear equations and manifolds

In this section we will consider the analytical properties of nonlinear systems of equations

$$f(x) = 0, \tag{2.13}$$

where $f \in \mathcal{C}^l(\mathbb{R}^n, \mathbb{R}^m)$ is a nonlinear system of functions mapping a vector $x \in \mathbb{R}^n$ to a vector $y = f(x) \in \mathbb{R}^m$, defined by a system of m nonlinear equations

$$y = f(x) = \begin{bmatrix} f_1(x_1, ..., x_n) \\ \vdots \\ f_m(x_1, ..., x_n) \end{bmatrix}.$$

For the investigation of nonlinear differential-algebraic equations, see Chapter 3, and for nonlinear model equations of mechanical systems, see Chapter 4, solvability results for systems of nonlinear equations (2.13) are of great importance. The following two theorems, see [16], give conditions for the (local) solvability of such systems of equations.

Theorem 2.3.1 (Implicit Function Theorem) *Let $f \in \mathcal{C}^1(\mathbb{S}((x_0, y_0), \epsilon_0), \mathbb{R}^r)$ and let $\mathbb{S}((x_0, y_0), \epsilon_0)$ be an open neighborhood of the point (x_0, y_0) in $\mathbb{X} \times \mathbb{Y}$ with $\mathbb{X} \subset \mathbb{R}^r$ and $\mathbb{Y} \subset \mathbb{R}^m$. If $f(x_0, y_0) = 0$ and if the matrix $f_{,x}(x_0, y_0)$ is nonsingular, then there exist a neighborhood $\mathbb{S}(x_0, \epsilon_0^x)$ of the point x_0 in $\mathbb{X}$, a neighborhood $\mathbb{S}(y_0, \epsilon_0^y)$ of the point y_0 in $\mathbb{Y}$, and a unique function $\varphi : \mathbb{S}(y_0, \epsilon_0^y) \to \mathbb{S}(x_0, \epsilon_0^x)$ such that $\varphi(y_0) = x_0$ and $f(\varphi(y), y) = 0$ for all $y \in \mathbb{S}(y_0, \epsilon_0^y)$. Furthermore, φ is continuously differentiable in $\mathbb{S}(y_0, \epsilon_0^y)$, and its derivative is given by*

$$\begin{aligned} \varphi'(y) &= -\left(\frac{\partial}{\partial x} f(\varphi(y), y)\right)^{-1} \frac{\partial}{\partial y} f(\varphi(y), y) \\ &= -f_{,x}^{-1}(\varphi(y), y) f_{,y}(\varphi(y), y). \end{aligned} \tag{2.14}$$

Proof: See [16, 47]. □

Theorem 2.3.2 (General Implicit Function Theorem)
Let $f \in \mathcal{C}^1(\mathbb{S}((x_0, y_0), \epsilon_0), \mathbb{R}^s)$ and let $\mathbb{S}((x_0, y_0), \epsilon_0)$ be an open neighborhood of the point (x_0, y_0) in $\mathbb{X} \times \mathbb{Y}$ with $\mathbb{X} \subset \mathbb{R}^r$ and $\mathbb{Y} \subset \mathbb{R}^m$. If $f(x_0, y_0) = 0$, if the matrix $f_{,x}(x_0, y_0)$ has rank r, and if the matrix $\begin{bmatrix} f_{,x}(x_0, y_0) & f_{,y}(x_0, y_0) \end{bmatrix}$ has rank r on $\mathbb{S}((x_0, y_0), \epsilon_0)$, then there exist a neighborhood $\mathbb{S}(x_0, \epsilon_0^x)$ of the point x_0 in $\mathbb{X}$, a neighborhood $\mathbb{S}(y_0, \epsilon_0^y)$ of the point y_0 in $\mathbb{Y}$, and a unique function $\varphi : \mathbb{S}(y_0, \epsilon_0^y) \to \mathbb{S}(x_0, \epsilon_0^x)$ such that $\varphi(y_0) = x_0$ and $f(\varphi(y), y) = 0$ for all $y \in \mathbb{S}(y_0, \epsilon_0^y)$. Furthermore, φ is continuously differentiable in $\mathbb{S}(y_0, \epsilon_0^y)$.

Proof: See [16]. □

The regularity of certain systems of nonlinear equations is a very important fact in the investigation of differential-algebraic equations and the model equations of mechanical systems. Therefore, in the following we will develop some basic tools regarding regularity and redundancy of functions defined by a system of equations. Since a nonlinear function $f : x \mapsto y$ is defined by the system of equations $y = f(x)$ we use the same notation for both.

Definition 2.3.3 (Regular point) *Let $f \in \mathcal{C}^1(\mathbb{X}, \mathbb{R}^m)$, let $\mathbb{X}$ be open in $\mathbb{R}^n$, and let $n \geq m$. A point $x_0 \in \mathbb{X}$ is called* regular *with respect to the function f if the total derivative $\frac{\mathrm{d}}{\mathrm{d}x} f(x_0)$ of f with respect to x evaluated at the point x_0 has full rank m.*

In the following we will define the terms *variety* and *manifold*, see [16].

Definition 2.3.4 (Variety) *Let $f \in \mathcal{C}^1(\mathbb{X}, \mathbb{R}^m)$ and let $\mathbb{X}$ be an open subset of $\mathbb{R}^n$. Then the subset $\mathbb{V}$ of $\mathbb{R}^n$, consisting of those points $x \in \mathbb{X}$ for which $f(x) = 0$ is satisfied is called a* variety defined by f.
If each point of a variety $\mathbb{V}$ has a neighborhood in $\mathbb{R}^n$ on which the matrix $f_{,x}(x)$ has constant rank $r < n$ and if that rank r is the same for all points in $\mathbb{V}$, then the variety defined by f is called a differentiable variety *in $\mathbb{R}^n$.*

Definition 2.3.5 (Singular point) *Let $f \in \mathcal{C}^1(\mathbb{X}, \mathbb{R}^m)$ define a variety $\mathbb{S} \subset \mathbb{X}$ and let $\mathbb{X}$ be an open subset of $\mathbb{R}^n$. Then a point $x \in \mathbb{S}$ is called a* singular point *if there does not exist a neighborhood $\mathbb{S}(x, \epsilon) = \{y \in \mathbb{S} \ : \ ||x - y|| < \epsilon\}$ in $\mathbb{S}$ with $\epsilon > 0$ on which the matrix $f_{,x}(y)$ has constant rank r for all points $y \in \mathbb{S}(x, \epsilon)$. Otherwise, the point x is called a* nonsingular point.

Definition 2.3.6 (Manifold) *We say that $\mathbb{M} \subset \mathbb{R}^n$ is a* manifold *of dimension r if every point $x \in \mathbb{M}$ has an open neighborhood in $\mathbb{M}$ which is homeomorphic to an open subset of $\mathbb{R}^r$.*

Definition 2.3.7 (Neighborhood of a subset) *Let $\mathbb{X} \in \mathbb{R}^n$. Then $\mathbb{X}^\epsilon = \{x \in \mathbb{R}^n \ : \ \text{there exists an } y \in \mathbb{X} \text{ with } ||x - y|| < \epsilon\}$ defines a* neighborhood of the subset $\mathbb{X}$.

Lemma 2.3.8 *Let $f \in \mathcal{C}^1(\mathbb{X}, \mathbb{R}^m)$ and let $\mathbb{X}$ be an open subset of $\mathbb{R}^n$. Let $\mathbb{M}$ be a variety defined by f and let $\operatorname{rank}(f_{,x}(x)) = r = \text{const}$ for all $x \in \mathbb{M}$. Then the variety $\mathbb{M}$ defined by f is a manifold.*

Proof: Since $\operatorname{rank}(f_{,x}(x)) = r$ is constant for all $x \in \mathbb{M}$, for every $x_0 \in \mathbb{M}$ there exist index sets $I \in \mathbb{N}^r$ and $J \in \mathbb{N}^r$ such that the minor $[f_{,x}]_{IJ}(x_0)$ (of r-th order) is nonzero, whereas all minors of higher order than r vanish. Furthermore, because of the smoothness of the minor $[f_{,x}]_{IJ}$, it does not vanish in a neighborhood $\mathbb{S}(x_0, \epsilon_0) \subset \mathbb{M}$ with $\mathbb{S}(x_0, \epsilon_0) = \{x \in \mathbb{M} \ : \ ||x - x_0|| < \epsilon_0\}$, $\epsilon_0 > 0$, see Lemma 2.1.1. From the General Implicit Function Theorem 2.3.2 we get the existence of a function $\varphi(x_{\bar{J}})$ such that $f(\varphi(x_{\bar{J}}), x_{\bar{J}}) = 0$ for all $x_{\bar{J}} \in \mathbb{R}^{n-r}$. Therefore, for every point x_0 and its corresponding neighborhood $\mathbb{S}(x_0, \epsilon_0)$ we have an homeomorphism $\xi \ : \ \mathbb{R}^{n-r} \to \mathbb{R}^n$ with

$$\xi(x_{\bar{J}}) = \begin{bmatrix} \varphi(x_{\bar{J}}) \\ x_{\bar{J}} \end{bmatrix}.$$

The assertion follows from Definition 2.3.6. □

Remark 2.3.9 Manifolds do not contain any singular points. □

Definition 2.3.10 (Redundancy of a function and a system of equations) *Let $f \in \mathcal{C}^1(\mathbb{X}, \mathbb{R}^m)$ where $\mathbb{X}$ is open in $\mathbb{R}^n$. The function f is called* nonredundant *over $\mathbb{X}$ if $n \geq m$ and every point $x \in \mathbb{X}$ is regular with respect to f. Otherwise the function is called* redundant.
Equivalently, a system of equations $f(x) = 0$ is called redundant *or* nonredundant *if the function f is redundant or nonredundant, respectively.*

For nonredundant functions $f \in \mathcal{C}^1(\mathbb{X}, \mathbb{R}^m)$ over $\mathbb{X}$ the rank condition $\operatorname{rank}(f_{,x}(x)) = m$ is satisfied for all $x \in \mathbb{X}$. Furthermore, one has to distinguish between certain types of redundancy of a system of equations.

Definition 2.3.11 (Uniformly redundant function, rank of a function) *Let $f \in \mathcal{C}^1(\mathbb{X}, \mathbb{R}^m)$ be a redundant function and let $\mathbb{X}$ be open in $\mathbb{R}^n$. The function f is called* uniformly redundant *over $\mathbb{X}$ if the rank of the total derivative of $f(x)$ with respect to x is constant for all points $x_0 \in \mathbb{X}$, i.e.,*

$$\operatorname{rank}(\frac{\mathrm{d}}{\mathrm{d}x} f(x_0)) \equiv r = \text{const}$$

with $r < m$ for all $x_0 \in \mathbb{X}$. Equivalently, a system of equations $f(x) = 0$ is called uniformly redundant *if the function f is uniformly redundant. The quantity r is called the* (uniform) rank of the function f *or of the system of equations $f(x) = 0$ over $\mathbb{X}$.*

Definition 2.3.12 (Noncontradictory system of equations) *A system of equations $f(x) = 0$ with $f : \mathbb{X} \to \mathbb{R}^m$ is called* noncontradictory *if there exists at least one $x_0 \in \mathbb{X}$ satisfying $f(x_0) = 0$.*

Notation 2.3.13 In the following, we will use the term rank both for a matrix (or a matrix function) as well as for a system of equations. The rank of the system of equations $f(x) = 0$ corresponds to the maximal number of nonredundant equations. In particular, this corresponds to the rank of the total derivative in the set of solutions, i.e.,

$$\operatorname{rank}(f) = \operatorname{rank}(\frac{\mathrm{d}}{\mathrm{d}x} f(x)).$$

□

Lemma 2.3.14 *Let $f \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^m)$ with $l \geq 1$ be a uniformly redundant function with rank r, where $\mathbb{X}$ is open in $\mathbb{R}^n$ and $\mathbb{X}_0 = \{x \in \mathbb{R}^n : f(x) = 0\} \subset \mathbb{X}$. Furthermore, let $f(x) = 0$ be a noncontradictory system of equations. Then, there exists a matrix function $S \in \mathcal{C}^{l-1}(\mathbb{X}, \mathbb{R}^{r,m})$ such that the function Sf is nonredundant and the respective system of equations*

$$S(x)f(x) = 0 \tag{2.15}$$

is noncontradictory for all $x \in \mathbb{X}_0$ with the same local solution set as $f(x) = 0$, i.e., there exists an $\epsilon > 0$ such that for every solution y of (2.15) with $f(y) \neq 0$ we have $||y - x|| > \epsilon$ for all x satisfying (2.15).

Proof: We have $f_{,x} \in \mathcal{C}^{l-1}(\mathbb{X}, \mathbb{R}^{m,n})$. By Theorem 2.1.6 we get the existence of a matrix function $S \in \mathcal{C}^{l-1}(\mathbb{X}, \mathbb{R}^{r,m})$ such that $Sf_{,x} \in \mathcal{C}^{l-1}(\mathbb{X}, \mathbb{R}^{r,n})$ has full rank r for all $x \in \mathbb{X}_0$. Furthermore, with $f(x) = 0$ for all $x \in \mathbb{X}_0$ we have that the partial derivative of the function $S(x)f(x)$ has the form

$$\frac{\partial}{\partial x}(S(x)f(x)) = S_{,x}(x)[\![\cdot, f(x)]\!] + S(x)f_{,x}(x) = S(x)f_{,x}(x), \tag{2.16}$$

see also [138]. Then the system of equations $S(x)f(x) = 0$ is nonredundant by Definition 2.3.10. Since $f(x) = 0$ is noncontradictory it follows that $S(x)f(x) = 0$ is noncontradictory.
Since $f_{,x}(x)$ has constant rank r for all $x \in \mathbb{X}$, we get from Lemma 2.3.8 that the set $\mathbb{M}_f = \{x \in \mathbb{X} : f(x) = 0\}$ is a manifold of dimension $n - r$, i.e., it does not have singular points. By the construction of $S(x)$ the matrix $\frac{\partial}{\partial x}(S(x)f(x))$ has constant maximal rank r for all $x \in \mathbb{M}_f$. Therefore, it follows from Lemma 2.3.8 that Sf with $x \in \mathbb{M}_f$ describes a manifold $\mathbb{M}_{Sf} = \{x \in \mathbb{X} : S(x)f(x) = 0\}$ of dimension $n - r$. Because of $\mathbb{M}_f \subset \mathbb{M}_{Sf}$ and $\dim(\mathbb{M}_f) = \dim(\mathbb{M}_{Sf}) = n - r$ we get the assertion. □

Unfortunately, in (2.15) the solution set of $0 = S(x)f(x)$ is not identical to the solution set of $0 = f(x)$ in general. This is illustrated by the following example.

Example 2.3.15 Let us consider the set of equations $0 = f(x)$ with

$$f(x) = \begin{bmatrix} x \\ x \end{bmatrix}$$

and $\mathbb{X} = \mathbb{R}$. Obviously, $f(x)$ is redundant and noncontradictory (and even linear). The solution set is $\mathbb{M}_f = \{0\}$ and $\operatorname{rank}(f_{,x}(x)) = 1$ for all $x \in \mathbb{R}$. The matrix

$$S(x) = \begin{bmatrix} \cos(\frac{\pi}{4} + x) & \sin(\frac{\pi}{4} + x) \end{bmatrix} \in \mathbb{R}^{1,2}$$

has full rank, i.e., $\operatorname{rank}(S(x)) = 1$ for all $x \in \mathbb{R}$, and we obtain

$$0 = S(x)f(x) = (\cos(\frac{\pi}{4} + x) + \sin(\frac{\pi}{4} + x))x,$$

with the solution set $\mathbb{M}_{Sf} = \{0, \pi/2 + k\pi \text{ with } k \in \mathbb{Z}\}$. □

Remark 2.3.16 Let $l \in \mathbb{N}$. If $f \in \mathcal{C}^l(\mathbb{X}, \mathbb{R}^m)$ with $\mathbb{X} \in \mathbb{R}^n$ and $y \in \mathcal{C}^l(\mathbb{I}, \mathbb{R}^n)$ where $y(t) \in \mathbb{X}_0 = \{x \in \mathbb{R}^n \ : \ f(x) = 0\}$ for all $t \in \mathbb{I}$, then we have

$$\frac{\mathrm{d}^i}{\mathrm{d}t^i} f(y(t)) = 0$$

and with a matrix function $S \in \mathcal{C}^{l-1}(\mathbb{X}, \mathbb{R}^{r,m})$ as in Lemma 2.3.14 we have

$$\frac{\mathrm{d}^i}{\mathrm{d}t^i}(S(y(t))f(y(t))) = S(y(t))\frac{\mathrm{d}^i}{\mathrm{d}t^i} f(y(t))$$

for all $i \in \{0, ..., l-1\}$ and if defined also for $i = l$. □

Lemma 2.3.17 *Let $f \in \mathcal{C}^1(\mathbb{X} \times \mathbb{U}, \mathbb{R}^m)$ be a set of nonlinear functions which are uniformly redundant with respect to x. Furthermore, let the set of equations $f(x, u) = 0$ be noncontradictory and let $r_f = \operatorname{rank}(f_{,x}(x, u))$ for all $(x, u) \in \mathbb{M} = \{(x, u) \ : \ f(x, u) = 0\}$. Then for every matrix function $S \in \mathcal{C}^0(\mathbb{X} \times \mathbb{U}, \mathbb{R}^{m,m})$ such that*

$$S(x, u)f_{,x}(x, u) = \begin{bmatrix} S_1(x, u) \\ S_2(x, u) \end{bmatrix} f_{,x}(x, u) = \begin{bmatrix} S_1(x, u)f_{,x}(x, u) \\ 0 \end{bmatrix}$$

for all $(x, u) \in \mathbb{M}$ and with $S_1 \in \mathcal{C}^0(\mathbb{X} \times \mathbb{U}, \mathbb{R}^{r_f, m})$ and for all functions $x \in \mathcal{C}^1(\mathbb{I}, \mathbb{R}^n)$ and $u \in \mathcal{C}^1(\mathbb{I}, \mathbb{R}^{n_u})$ with $(x(t), u(t)) \in \mathbb{M}$ for all $t \in \mathbb{I} = [t_0, t_f]$ it follows that $S_2(x, u)\frac{\mathrm{d}}{\mathrm{d}t} f(x(t), u(t)) = 0$ for all $t \in \mathbb{I} = [t_0, t_f]$.

Proof: Since $0 = f(x, u)$ is noncontradictory, it follows that

$$\begin{bmatrix} f_{,x} & f_{,u} \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{u} \end{bmatrix} = 0$$

for all functions $x(t)$ and $u(t)$ with $(x(t), u(t)) \in \mathbb{M}$ for all $t \in \mathbb{I} = [t_0, t_f]$. It follows

$$\begin{bmatrix} S_1(x, u) \\ S_2(x, u) \end{bmatrix} \begin{bmatrix} f_{,x} & f_{,u} \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{u} \end{bmatrix} = 0$$

and

$$\begin{bmatrix} S_1(x, u)f_{,x} & S_1(x, u)f_{,u} \\ 0 & S_2(x, u)f_{,u} \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{u} \end{bmatrix} = 0$$

for all functions $x(t)$ and $u(t)$ with $(x(t), u(t)) \in \mathbb{M}$ for all $t \in \mathbb{I} = [t_0, t_f]$. Thus, it follows that

$$S_2(x, u)f_{,u}\dot{u} = 0$$

for all functions $x(t)$ and $u(t)$ with $(x(t), u(t)) \in \mathbb{M}$ for all $t \in \mathbb{I} = [t_0, t_f]$. In particular, it follows that

$$S_2(x,u)\frac{\mathrm{d}}{\mathrm{d}t}f(x(t),u(t)) = S_2(x,u)(f_{,x}\dot{x} + f_{,u}\dot{u}) = 0.$$

□

The numerical solution of systems of nonlinear equations has been investigated extensively in the literature, e.g., see [42, 94, 130].
The numerical solution of systems of nonlinear equations is necessary for the simulation of multibody systems, but not the main topic of the present work. Therefore, here only the most relevant results have been presented.

Algorithm 2.3.18 (Newton method) Let $f : \mathbb{X} \to \mathbb{R}^n$ be a continuously differentiable map with $\mathbb{X} \subset \mathbb{R}^n$ open and convex and let $x^0 \in \mathbb{X}$ be given. Suppose that $f_{,x}(x)$ is nonsingular for each $x \in \mathbb{X}$. Then proceed as follows.

0) set $k = 0$ (2.17a)
1) solve $\mathfrak{N}(x^k)\Delta x^k = -f(x^k)$ for Δx^k with $\mathfrak{N}(x^k) = f_{,x}(x^k)$ (2.17b)
2) set $x^{k+1} = x^k + \Delta x^k$ (2.17c)
3) increase k by 1 and continue with 1) (2.17d)

The matrix function $\mathfrak{N}$ defined in (2.17b) is called *Newton iteration matrix*.

Lemma 2.3.19 *Let $f : \mathbb{X} \to \mathbb{R}^n$ be a continuously differentiable map with $\mathbb{X} \subset \mathbb{R}^n$ open and convex. Suppose that $f_{,x}(x)$ is nonsingular for each $x \in \mathbb{X}$. Assume that the affine covariant Lipschitz[2] condition*

$$||f_{,x}^{-1}(x)(f_{,x}(y) - f_{,x}(x))(y-x)|| \quad \leq \quad \omega||y-x||^2$$

is satisfied for all $x, y \in \mathbb{X}$. Let $f(x) = 0$ have a solution x^*. For the initial guess x^0 assume that $\bar{\mathbb{S}}(x^*, ||x^0 - x^*||) \subset \mathbb{X}$ and that

$$\omega||x^0 - x^*|| < 2.$$

Then the sequence $\{x^k\}$ determined by use of the Newton method in Algorithm 2.3.18 remains in the open ball $\mathbb{S}(x^*, ||x^0 - x^*||)$ and converges to x^* at an estimated rate

$$||x^{k+1} - x^*|| \leq \frac{\omega}{2}||x^k - x^*||^2.$$

Moreover, the solution x^* is unique in the open ball $\mathbb{S}(x^*, 2/\omega)$.

Proof: See [42]. □

Algorithm 2.3.20 (Simplified Newton method) Let $f : \mathbb{X} \to \mathbb{R}^n$ be a continuously differentiable map with $\mathbb{X} \subset \mathbb{R}^n$ open and convex and let $x^0 \in \mathbb{X}$ be given. Suppose that $f_{,x}(x^0)$ is nonsingular for the initial guess $x^0 \in \mathbb{X}$. Then proceed as follows.

0) set $k = 0$ and $\mathfrak{N}_0 = f_{,x}(x^0)$
1) solve $\mathfrak{N}_0\Delta x^k = -f(x^k)$ for Δx^k
2) set $x^{k+1} = x^k + \Delta x^k$
3) increase k by 1 and continue with 1)

[2]Rudolf Otto Sigismund Lipschitz (1832 Königsberg, Germany (now Kaliningrad, Russia) - 1903 Bonn, Germany)

Lemma 2.3.21 *Let $f : \mathbb{X} \to \mathbb{R}^n$ be a continuously differentiable map with $\mathbb{X} \subset \mathbb{R}^n$ open and convex. Suppose that $f_{,x}(x^0)$ is nonsingular for the initial guess $x^0 \in \mathbb{X}$. Assume that the affine covariant Lipschitz condition*

$$||f_{,x}^{-1}(x^0)(f_{,x}(x) - f_{,x}(x^0))|| \quad \leq \quad \omega_0||x - x^0||$$

is satisfied for all $x \in \mathbb{X}$. Let

$$h_0 = \omega_0||\Delta x^0|| \leq \frac{1}{2}$$

and define

$$t^* = 1 - \sqrt{1 - 2h_0}, \qquad \rho = \frac{t^*}{\omega_0}.$$

Moreover, assume that $\bar{\mathbb{S}}(x^0, \rho) \subset \mathbb{X}$. Then the sequence $\{x^k\}$ determined by use of the simplified Newton method in Algorithm 2.3.20 remains in $\bar{\mathbb{S}}(x^0, \rho)$ and converges to some x^* with $f(x^*) = 0$. The convergence rate can be estimated by

$$\frac{||x^{k+1} - x^k||}{||x^k - x^{k-1}||} \leq \frac{1}{2}(t_k + t_{k-1}), \ k = 1, 2, ...$$

and

$$||x^k - x^*|| \leq \frac{t^* - t_k}{\omega_0}, \ k = 0, ...$$

with $t_0 = 0$ and

$$t_{k+1} = h_0 + \frac{1}{2}t_k^2, \ k = 0, 1, ... \ .$$

Proof: See [42]. □

Chapter 3

Differential-Algebraic Equations

In recent years, differential equations in combination with algebraic constraints, called *differential-algebraic equations* (DAEs) have gained more and more attention in the modeling of dynamical processes. In particular, the modeling of complex mechanical systems leads to differential equations in connection with algebraic constraints. While the theory for the numerical integration of purely differential equations is well developed and a large collection of appropriate solvers are available, the situation regarding DAEs is more difficult and more complex. In recent years large efforts have been made to investigate the analytical as well as the numerical behavior of DAEs. This has lead to a better understanding of DAEs, e.g., see [7, 25, 43, 73, 79, 82, 105, 172, 173]. While the analytical and numerical behavior of certain classes of DAEs is well developed, the analytical as well as the numerical treatment of general nonlinear DAEs or even of quasi-linear DAEs is not yet clear and many questions of this topic are still untreated or unsolved.
In this chapter we will review and develop important facts on DAEs in view of the numerical treatment of multibody systems. In particular, we will review the index concept as well as some regularization techniques. As basis for the treatment of multibody systems we will discuss in detail quasi-linear DAEs with respect to their analytical and numerical properties as well as with respect to their numerical treatment with Runge-Kutta methods.
In Section 3.1 we will review some important facts and definitions which are helpful for further investigations. Subsequently, in Section 3.2 we will discuss the index concept for general nonlinear DAEs which allows a classification of DAEs with respect to the degree of difficulties arising in their analytical and numerical treatment. In particular, the recently developed strangeness concept will be considered in detail. In Section 3.3 we will discuss the linearization of DAEs along an arbitrary trajectory and its relation to certain properties of DAEs, in particular, with respect to the index. Subsequently, in Section 3.4 we will present some important regularization techniques for DAEs of higher index. In Section 3.5 we will investigate quasi-linear DAEs in detail. As first instance, we will consider strangeness-free quasi-linear DAEs in Section 3.5.1. In particular, in Section 3.5.2 we will present a procedure which provides a general tool for the investigation of general quasi-linear DAEs of an arbitrary index, in particular, this procedure offers the possibility for the determination of the index as well as the definition of the hidden constraints, the maximal constraint level, and the solution manifold. Furthermore, this procedure can be used as basis for a regularization method suited for general quasi-linear DAEs as discussed in Section 3.5.3. Concluding that section, in Section 3.5.4 we

will discuss the numerical integration of quasi-linear DAEs via Runge-Kutta methods based on the regularization developed in Section 3.5.3. In particular, we will investigate the discretization, the numerical solution of the arising nonlinear stage equations, and in particular, the efficient numerical solution of the linear systems arising in the integration process. Concluding this chapter we will give an overview of available and commonly used numerical methods for the numerical integration of DAEs in Section 3.6.

3.1 Preliminaries

In this section we will review some fundamental facts of importance for further investigations for differential equations. Consider an initial value problem for the *ordinary differential equations* (ODEs)

$$\dot{x}(t) = f(x(t), t) \tag{3.1a}$$

on the domain $\mathbb{I} = [t_0, t_f]$ with the initial values

$$x(t_0) = x_0 \in \mathbb{R}^n. \tag{3.1b}$$

Here, the vector function x depending on t of size n denotes the unknown variables and f is a vector function of the same size n depending on x and t. The existence and uniqueness of the solutions follows from the Theorem of Picard[1] and Lindelöf[2].

Theorem 3.1.1 (Picard and Lindelöf) *Let $f(x,t)$ be continuous on the compact rectangle*

$$\mathcal{R} = \{(x,t) \ : \ |t - t_0| < a, \ ||x - x_0|| < b\}, \qquad (a, b > 0),$$

and let $f(x,t)$ be Lipschitz continuous with respect to x with the Lipschitz constant L, i.e.,

$$||f(x,t) - f(y,t)|| \leq L||x - y|| \quad \textit{for all } (x,t), (y,t) \in \mathcal{R}.$$

Then there exists a unique solution $x(t)$ of the initial value problem (3.1) on

$$\mathbb{I} = [t_0 - \alpha, t_0 + \alpha],$$

where

$$\alpha = \min\left(a, \frac{b}{M}\right) \quad \textit{with } M = \max_{(x,t)\in\mathcal{R}} ||f(x,t)||.$$

Proof: The proof is given in [88]. □

For the numerical simulation of multibody systems, i.e., the numerical integration of the model equations of mechanical systems, see Chapters 4 and 5, DAEs are of great importance. Therefore, let us discuss the initial value problem for the nonlinear DAE

$$0 = F(x(t), \dot{x}(t), u(t)), \tag{3.2a}$$

with $F : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{U} \to \mathbb{R}^m$ on the domain $\mathbb{I} = [t_0, t_f]$ with initial values

$$x(t_0) = x_0 \in \mathbb{R}^n. \tag{3.2b}$$

[1] Charles Emile Picard (born 1856 in Paris, France - died 1941 in Paris, France)

[2] Ernst Leonard Lindelöf (born 1870 in Helsingfors, Russian Empire (now Helsinki, Finland) - died 1946 in Helsinki, Finland)

Note that the right-hand side of (3.2a) depends on an additional variable $u(t) \in \mathbb{U} \subset \mathbb{R}^{n_u}$, which represent *control variables* which are given as function of t. In particular, if the investigated system is a nonautonomous system, then there exists an explicit dependence on t and t will be modeled as a component of the control u, for example $u_1(t) = t$.
Special forms of DAEs which play an important role in our investigations are *linear DAEs with variable coefficients*

$$E(t)\dot{x}(t) = A(t)x(t) + k(t), \tag{3.3}$$

where $E, A \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{m,n})$, $k \in \mathcal{C}(\mathbb{I}, \mathbb{R}^m)$, *quasi-linear DAEs* of the form

$$E(x(t), u(t))\dot{x}(t) = k(x(t), u(t)), \tag{3.4}$$

where $E \in \mathcal{C}(\mathbb{X} \times \mathbb{U}, \mathbb{R}^{m,n})$, $k \in \mathcal{C}(\mathbb{X} \times \mathbb{U}, \mathbb{R}^m)$, $\mathbb{X} \subset \mathbb{R}^n$ and DAEs in partitioned form

$$\begin{aligned} 0 &= F_1(x(t), \dot{x}(t), y(t), u(t)), && \text{(3.5a)} \\ 0 &= F_2(x(t), y(t), u(t)), && \text{(3.5b)} \end{aligned}$$

with $F_1 \in \mathcal{C}(\mathbb{X} \times \mathbb{X}^{(1)} \times \mathbb{Y} \times \mathbb{U}, \mathbb{R}^{m_1})$, $F_2 \in \mathcal{C}(\mathbb{X} \times \mathbb{Y} \times \mathbb{U}, \mathbb{R}^{m_2})$, $\mathbb{X}, \mathbb{X}^{(1)} \subset \mathbb{R}^{n_x}$, $\mathbb{Y} \subset \mathbb{R}^{n_y}$, and $\mathbb{U} \subset \mathbb{R}^{n_u}$ all in the domain $\mathbb{I} = [t_0, t_f]$ and with initial values (3.2b). Quasi-linear DAEs will be considered in more detail in Section 3.5.

Definition 3.1.2 (Solution of DAEs) *Consider a system of differential-algebraic equations of form (3.2a). A function $x : \mathbb{I} \to \mathbb{R}^n$ is called a* solution of DAE *(3.2a) if x is sufficiently continuously differentiable and satisfies (3.2a) pointwise. In addition it is called a* solution of the initial value problem *(3.2) if x is a solution of (3.2a) and satisfies (3.2b).*

Definition 3.1.3 (Consistency of DAEs, consistency of (initial) values) *The DAE (3.2a) is called* consistent *if there exists a solution of the DAE (3.2a). Values $y \in \mathbb{X} \subset \mathbb{R}^n$ are called* consistent to the DAE *(3.2a) if there exists at least one solution x of the DAE (3.2a) and a $t \in \mathbb{I}$ with $x(t) = y$. The initial values (3.2b) are called* consistent to the DAE *(3.2a) if there exists at least one solution x of the initial value problem (3.2).*

Notation 3.1.4 With respect to the unknown variables $x \in \mathbb{X} \subset \mathbb{R}^n$ we will use the notation

$$\mathfrak{x}^i = \begin{bmatrix} x \\ \dot{x} \\ \vdots \\ x^{(i)} \end{bmatrix} \in \mathbb{X}^i \subset \mathbb{R}^{(i+1)n},$$

with $\mathbb{X}^0 = \mathbb{X}$ and $\mathbb{X}^i = \mathbb{X}^{i-1} \times \mathbb{X}^{(i)}$, $i \in \mathbb{N}$. The set $\mathbb{X}^{(i)}$ denotes the set of admissible $x^{(i)}(t) = \frac{\mathrm{d}^i}{\mathrm{d}t^i}x(t)$. With respect to the control variables $u \in \mathbb{U} \subset \mathbb{R}^{n_u}$ we will use the same notation, i.e.,

$$\mathfrak{u}^i = \begin{bmatrix} u \\ \dot{u} \\ \vdots \\ u^{(i)} \end{bmatrix} \in \mathbb{U}^i \subset \mathbb{R}^{(i+1)n_u},$$

with $\mathbb{U}^{-1} = \mathbb{U}^0 = \mathbb{U}$ and $\mathbb{U}^i = \mathbb{U}^{i-1} \times \mathbb{U}^{(i)}$, $i \in \mathbb{N}$. The set $\mathbb{U}^{(i)}$ denotes the set of admissible $u^{(i)}(t) = \frac{\mathrm{d}^i}{\mathrm{d}t^i}u(t)$. Furthermore, we will use the convention $\mathfrak{u}^{-1} = \mathfrak{u}^0 = u$. □

3.2 The index of differential-algebraic equations

The analytical and numerical treatment of DAEs is quite different and more complicated than the one of ODEs, see [133] and in addition [25, 73, 82, 105]. The investigation of DAEs requires a suitable classification of DAEs according to certain degrees of difficulty. This requirement leads to the (independent) development of several index concepts for the classification of different types of DAEs.
Today, the index concept plays a key role in the numerical analysis of DAEs. The index of a DAE provides a measure of difficulty in the analytical as well as in the numerical solution. In this section we give a short overview over different index concepts. For more detailed investigations of the different index concepts and its relation to each other we refer to the literature cited below.
First attempts for a better understanding of DAEs were done by investigating linear DAEs (3.3) with constant coefficients $E, A \in \mathbb{R}^{n,n}$. The investigation of such linear DAEs is related to the investigation of matrix pairs, see [61]. If a transformation of the linear DAE, i.e., scaling with a nonsingular matrix $P \in \mathbb{R}^{n,n}$ and transforming $x(t) = Qy(t)$ with a nonsingular matrix $Q \in \mathbb{R}^{n,n}$, leads to a linear DAE of the form

$$\begin{bmatrix} I & 0 \\ 0 & N \end{bmatrix} \dot{y}(t) = \begin{bmatrix} J & 0 \\ 0 & I \end{bmatrix} y(t) + Pk(t), \tag{3.6}$$

where N is a nilpotent Jordan[3] block matrix with $N^{\nu_n} = 0$ and $N^{\nu_n - 1} \neq 0$, the linear DAE (3.3) with constant coefficients is said to be a DAE of *index of nilpotency* ν_n. In the style of matrix pairs the index of nilpotency is also known as the *Kronecker*[4] *index* and the DAE (3.6) is said to be in *Kronecker canonical form*. For more details see [25, 61, 82, 105].
A generalization of the index of nilpotency concerning for linear DAEs (3.3) with variable coefficients $E, A \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{n,n})$ is done in [69] and leads to the so-called *global index* ν_g. The idea is similar to the index of nilpotency except that the transformation matrices P and Q are assumed to be continuous and continuously differentiable, respectively, i.e., $P \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{n,n})$, $Q \in \mathcal{C}^1(\mathbb{I}, \mathbb{R}^{n,n})$, and the matrix J is allowed to depend on t, i.e., $J \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{n_1,n_1})$. Then, if the index of nilpotency is constant over $\mathbb{I}$ the global index ν_g corresponds to the index of nilpotency, i.e., $\nu_g = \nu_n$.
Furthermore, in [69] a further generalization to general nonlinear DAEs of the form (3.2a) is given in the way that a system is said to have *uniform index* ν_u if the index of nilpotency of the linear(ized) DAE (3.3) with constant coefficients $E = F_{,\dot{x}}(x, \dot{x}, u)$ and $A = F_{,x}(x, \dot{x}, u)$ is independent of the points of evaluation. Then $\nu_u = \nu_n$.
Rabier and Rheinboldt investigated differential-algebraic equations from a geometrical point of view. They consider a DAE as an ODE on manifolds which are given by the constraints which restrict the solution. Based on this geometrical point of view in [138, 145] the *geometrical index* was introduced.
Griepentrog and März developed in [73] the *tractability index* (t-index) for linear DAEs (3.3). An other important class of DAEs are those with *properly stated leading term* introduced in [120]. DAEs with properly stated leading term are given in the form

$$E(t)\frac{\mathrm{d}}{\mathrm{d}t}(D(t)x(t)) = k(x,t)$$

[3] Marie Ennemond Camille Jordan (born 1838 in La Croix-Rousse, Lyon, France - died 1922 in Paris, France)
[4] Leopold Kronecker (1823 Legnica, Poland - 1891 Berlin, Germany)

with $E \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{n,l})$, $D \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{l,n})$, $\ker(E(t)) \oplus \operatorname{range}(D(t)) = \mathbb{R}^l$ for all $t \in \mathbb{I}$, and there exists a projector $R \in \mathcal{C}^1(\mathbb{I}, \mathbb{R}^{l,l})$ such that

$$\operatorname{range}(R(t)) = \operatorname{range}(D(t)),\ \ker(R(t)) = \ker(E(t))$$

for all $t \in \mathbb{I}$. While the tractability index for linear DAEs with properly stated leading term is defined in [120] the generalizations to nonlinear DAEs with properly stated leading term are proposed in [121]. The aim of the tractability index is the construction of matrix chains for the decoupling of the DAE into characteristic components. This decoupling is not developed in view of numerical methods, rather, it is suited for the analytical investigation of the DAE as well as its discretizations. Furthermore, it is useful for the determination of consistent initial values, see [54, 84], as well as for the determination of the smoothness requirements to the solution and system matrices and vectors. In particular, the matrices of the constructed matrix chains allow the analysis of smoothness requirements for the solution.
As a general approach to index and structural analysis of general DAEs $F(x, \dot{x}) = 0$, Campbell introduced the *derivative array*, which summarizes the original equation and all its derivatives up to a certain order l in one large system, see [29, 32, 103]. The derivative array with respect to (3.2a) is defined as follows.

Definition 3.2.1 (Derivative array) *Let $\mathbb{X}, \mathbb{X}^{(1)} \subset \mathbb{R}^n$ and $\mathbb{U} \subset \mathbb{R}^{n_u}$ be given and let $F : (x(t), \dot{x}(t), u(t)) \mapsto F(x(t), \dot{x}(t), u(t))$ be a function in $\mathcal{C}^s(\mathbb{X} \times \mathbb{X}^{(1)} \times \mathbb{U}, \mathbb{R}^m)$. Then, the associated* derivative array of order l *with respect to (3.2a), with $l \leq s$, has the form*

$$0 = \mathfrak{F}_l(x, \dot{x}, ..., x^{(l+1)}, \mathfrak{u}^l) = \begin{bmatrix} F(x, \dot{x}, u) \\ \frac{\mathrm{d}}{\mathrm{d}t} F(x, \dot{x}, u) \\ \vdots \\ \frac{\mathrm{d}^l}{\mathrm{d}t^l} F(x, \dot{x}, u) \end{bmatrix}, \tag{3.7}$$

where $\mathfrak{u}^l(t) = \begin{bmatrix} u^T(t) & \cdots & (u^{(l)}(t))^T \end{bmatrix}^T \in \mathbb{U}^l$.
A reduced derivative array of level l *is denoted by $\check{\mathfrak{F}}_l(x, \dot{x}, ..., x^{(l+1)}, \mathfrak{u}^l)$ and does contain the function $F(x, \dot{x}, u)$ and some of its derivatives, i.e.,*

$$0 = \check{\mathfrak{F}}_l(x, \dot{x}, ..., x^{(l+1)}, \mathfrak{u}^l) = \begin{bmatrix} F(x, \dot{x}, u) \\ \frac{\mathrm{d}}{\mathrm{d}t} F_{I_1}(x, \dot{x}, u) \\ \vdots \\ \frac{\mathrm{d}^l}{\mathrm{d}t^l} F_{I_l}(x, \dot{x}, u) \end{bmatrix} \tag{3.8}$$

with certain index vectors I_j, $j = 1, ..., l$, see Notation 2.0.3 for the terms $F_{I_1}, ..., F_{I_l}$.

Remark 3.2.2 Consider the derivative array $0 = \mathfrak{F}_l$ of level l, see (3.7). If we compute the derivative array of level 1 of $0 = \mathfrak{F}_l$, i.e.,

$$0 = \begin{bmatrix} \mathfrak{F}_l(x, \dot{x}, ..., x^{(l+1)}, \mathfrak{u}^l) \\ \frac{\mathrm{d}}{\mathrm{d}t} \mathfrak{F}_l(x, \dot{x}, ..., x^{(l+1)}, \mathfrak{u}^l) \end{bmatrix} = \begin{bmatrix} F(x, \dot{x}, u) \\ \vdots \\ \frac{\mathrm{d}^l}{\mathrm{d}t^l} F(x, \dot{x}, u) \\ \hline \frac{\mathrm{d}^1}{\mathrm{d}t^1} F(x, \dot{x}, u) \\ \vdots \\ \frac{\mathrm{d}^{l+1}}{\mathrm{d}t^{l+1}} F(x, \dot{x}, u) \end{bmatrix},$$

then its last block equations (consisting of the last m equations $0 = \frac{\mathrm{d}^{l+1}}{\mathrm{d}t^{l+1}} F(x, \dot{x}, u)$) correspond to the last block equations (consisting of the last m equations) of the

derivative array

$$0 = \mathfrak{F}_{l+1} = \begin{bmatrix} F(x, \dot{x}, u) \\ \vdots \\ \frac{\mathrm{d}^l}{\mathrm{d}t^l} F(x, \dot{x}, u) \\ \frac{\mathrm{d}^{l+1}}{\mathrm{d}t^{l+1}} F(x, \dot{x}, u) \end{bmatrix}$$

of level $l+1$ of the original DAE (3.2a). □

From Definition 3.2.1 for general nonlinear DAEs we get the derivative array for a linear DAE (3.3) in the following form, see also [100]. Let $E(t)$, $A(t)$, and $k(t)$ in (3.3) be sufficiently smooth. Formal differentiation of (3.3) leads to linear differential-algebraic equations of the form

$$\mathfrak{E}^l(t)\dot{\mathfrak{x}}^l(t) = \mathfrak{A}^l(t)\mathfrak{x}^l(t) + \mathfrak{K}^l(t) \tag{3.9}$$

for any l, where

$$(\mathfrak{E}^l(t))_{ij} = \binom{i}{j} E^{(i-j)}(t) - \binom{i}{j+1} A^{(i-j-1)}(t), \tag{3.10a}$$

$$(\mathfrak{A}^l(t))_{ij} = \begin{cases} A^{(i)}(t) & \text{for } i = 0, ..., l,\ j = 0, \\ 0 & \text{else,} \end{cases} \tag{3.10b}$$

$$(\mathfrak{K}^l(t))_i = k^{(i)}(t), \tag{3.10c}$$

$i, j = 0, ..., l$. We use the convention that $\binom{i}{j} = 0$ for $i < 0,\ j < 0$ or $j > i$. The matrix pairs $(\mathfrak{E}^l, \mathfrak{A}^l)$ for $l \in \mathbb{N}_0$ are called *inflated pairs*.
Based on the derivative array (3.7) the *differentiation index* (d-index) is defined as follows, see [31, 66, 67].

Definition 3.2.3 (Differentiation index) *Suppose that (3.2a) is a solvable system of differential-algebraic equations on an open set Ω. Let $\mathfrak{F}_l(x, \dot{x}, z, \mathfrak{u}^l)$ be the derivative array defined by (3.7). Let $\dot{x}$ be considered locally as an algebraic variable y. If ν_d is the smallest integer l such that y is uniquely determined by x, $\mathfrak{u}^l$, and*

$$0 = \mathfrak{F}_l(x, y, z, \mathfrak{u}^l)$$

for all consistent values then we call ν_d the differentiation index *(d-index) of the system of differential-algebraic equations (3.2a).*

Unfortunately, the d-index in this form is not suitable for the investigation of general DAEs. The two following examples demonstrate this fact.

Example 3.2.4 Let us consider the algebraic equation

$$x^2(t) = 0 \tag{3.11}$$

for $t \in \mathbb{I} = [t_0, t_f]$ as a very special case of a DAE. Following Definition 3.2.3 this DAE has d-index two, although, one would expect that the d-index is one because it is an algebraic equation. Let us explain this fact. The first derivative with respect to t of the DAE yields $2x(t)\dot{x}(t) = 0$ which is not uniquely solvable for $\dot{x}(t)$, since only $x(t) = 0$ is consistent. Further differentiation yields $2\dot{x}^2(t) + 2x(t)\ddot{x}(t) = 0$, which is uniquely solvable for $\dot{x}(t)$. Because of $x(t) = 0$ we get $\dot{x}(t) = 0$ which yields the correct solution for the consistent initial value $x(t_0) = 0$. Therefore, the derivative array of level at least two is necessary to determine $\dot{x}(t)$ uniquely in $x(t)$ and t for all consistent values. It follows that the DAE (3.11) has d-index $\nu_d = 2$. The reason for this effect is the existence of a multiple root of the algebraic equation.
Analogously, it is possible to show that the equation $x^n(t) = 0$ has d-index $\nu_d = n$.
□

Example 3.2.5 Consider the DAE

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & x_1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} x_1 x_2 \\ x_2 - 1 \\ x_3 - 1 \end{bmatrix} \tag{3.12}$$

with $\mathbb{X} = \mathbb{R}^3$ and $t \in \mathbb{I}$. From the last equation we have $x_3(t) = 1$ and it follows that $\dot{x}_3(t) = 0$. Inserting this into the second equation we get $x_2(t) = 1$. Inserting this into the first equation we get $\dot{x}_1(t) = x_1(t)$ and the solution of (3.12) is given by

$$x_1(t) = c_1 e^t, \qquad x_2(t) = 1, \qquad x_3(t) = 1.$$

The set of consistent values according to Definition 3.2.3 is given by $\{(x_1, x_2, x_3) \in \mathbb{R}^3 \ : \ x_2 = 1, \ x_3 = 1\}$. Therefore, at least two differentiations, i.e., the derivative array of level at least two, are necessary such that $\dot{x}(t)$ can be determined uniquely from $x(t)$ and t for all consistent values. It follows that the DAE (3.12) has d-index $\nu_d = 2$. On the other hand, there is only one differentiation necessary, if the initial value is $x(t_0) = \begin{bmatrix} 0 & 1 & 1 \end{bmatrix}^T$ with corresponding solution $x(t) = \begin{bmatrix} 0 & 1 & 1 \end{bmatrix}^T$. □

In [132] Pantelides developed an algorithm to determine a set of conditions that consistent initial values of DAEs (3.2a) have to satisfy. Later, this algorithm was applied to determine the so-called *structural index*. It is known that the structural index may be less than the d-index and it has been claimed that the structural index cannot exceed the d-index, see [25]. But, in [142] it is shown that in contrast to previous results in the literature the structural index of a DAE with constant coefficients of d-index 1 may be arbitrarily high.
Recently, a new concept for analytical and numerical treatment of DAEs has been developed by Kunkel and Mehrmann, see [102, 103, 104, 105]. This concept is known as *strangeness concept* and we review this concept applied to linear and nonlinear DAEs of arbitrarily high index.
Consider a general linear DAE with variable coefficients (3.3) together with an initial condition (3.2b) on the domain $t \in \mathbb{I} = [t_0, t_f]$. In [102, 103], it is explained how to determine an equivalent formulation of the DAE (3.3) with $m = n$ which contains the same solution set as the DAE (3.3) but has a more suitable structure. In [104, 106] this approach is generalized to arbitrary m and n.
Let $E(t)$, $A(t)$, and $k(t)$ in (3.3) be sufficiently smooth and let the derivative array be defined as in (3.9). Based on the inflated pair (3.10), the idea is to classify linear DAEs with the help of the following hypothesis for linear DAEs with variable coefficients, see [102, 104, 106].

Hypothesis 3.2.6 *Consider a general linear DAE (3.3) with variable coefficients. Then there exist integers ν, a, d, and v such that the inflated pairs $(\mathfrak{E}^\nu, \mathfrak{A}^\nu)$ and $(\mathfrak{E}^{\nu+1}, \mathfrak{A}^{\nu+1})$ associated with (E, A) have the following properties.*
1) For all $t \in \mathbb{I}$ we have $\operatorname{corank}(\mathfrak{E}^{\nu+1}(t)) - \operatorname{corank}(\mathfrak{E}^\nu(t)) = v$.
2) For all $t \in \mathbb{I}$ we have $\operatorname{rank}(\mathfrak{E}^\nu(t)) = (\nu+1)m - a - v$ *such that there exists a smooth matrix function Z_{23} of size $(\nu+1)m \times a + v$ and full rank with $Z_{23}^T(t)\mathfrak{E}^\nu(t) = 0$.*
3) For all $t \in \mathbb{I}$ we have $\operatorname{rank}(\mathfrak{A}^\nu(t)\mathcal{I}_\nu) = a$, *with $\mathcal{I}_\nu = \begin{bmatrix} I_n & 0_n & \cdots & 0_n \end{bmatrix}^T \in \mathbb{R}^{(\nu+1)n,n}$, such that without loss of generality Z_{23} may be partitioned as $Z_{23} = \begin{bmatrix} Z_2 & Z_3 \end{bmatrix}$ where Z_2 of size $(\nu+1)m \times a$ and maximal rank such that $\hat{A}_2 = Z_2^T \mathfrak{A}^\nu \mathcal{I}_\nu$ has full rank a and $Z_3^T \mathfrak{A}^\nu \mathcal{I}_\nu = 0$. This implies the existence of a smooth matrix function T_2 of size $n \times d$, $d = m - a - v$ and maximal rank satisfying $\hat{A}_2 T_2 = 0$.*
4) For all $t \in \mathbb{I}$ we have $\operatorname{rank}(E(t)T_2(t)) = d$, *such that there exists a smooth matrix function Z_1 of size $m \times d$ with* $\operatorname{rank}(\hat{E}_1(t)) = d$ *for all $t \in \mathbb{I}$, where $\hat{E}_1 = Z_1^T E$.*

The existence of the smooth matrix-valued functions Z_{23}, T_2, and Z_1 is ensured by Theorem 2.1.6.
Because the model equations of general multibody systems below are nonlinear we have to review the results according to the strangeness concept regarding general nonlinear DAEs (3.2a), see [103, 104, 106].
The set of solutions of the derivative array $\mathfrak{F}_\nu$ of order ν (3.7) corresponding to general nonlinear DAEs (3.2a) is defined as

$$\mathbb{L}_\nu = \{(z_0, ..., z_{\nu+1}, \mathbb{u}^\nu) \in \mathbb{R}^n \times ... \times \mathbb{R}^n \times \mathbb{U}^\nu \ : \ \mathfrak{F}_\nu(z_0, ..., z_{\nu+1}, \mathbb{u}^\nu) = 0\}. \quad (3.13)$$

The following hypothesis was stated in [104] for $u(t) = t$. Note, that we will use the convention that $\operatorname{corank}(\mathfrak{F}_{-1,x}) = 0$.

Hypothesis 3.2.7 *Consider a general nonlinear DAE (3.2a). There exist integers ν, r, a, d, and v such that $\mathbb{L}_\nu$ is not empty, and the following properties are satisfied.*
1) The set $\mathbb{L}_\nu \subset \mathbb{R}^{(\nu+2)n+(\nu+1)n_u}$ forms a manifold of dimension $(\nu+2)n + (\nu+1)n_u - r$.
2) We have $\operatorname{rank}(\mathfrak{F}_{\nu,x\dot{x}...x^{(\nu+1)}}) = r$ on $\mathbb{L}_\nu$.
3) We have $\operatorname{corank}(\mathfrak{F}_{\nu,x\dot{x}...x^{(\nu+1)}}) - \operatorname{corank}(\mathfrak{F}_{\nu-1,x\dot{x}...x^{(\nu)}}) = v$ on $\mathbb{L}_\nu$.
4) We have $\operatorname{rank}(\mathfrak{F}_{\nu,\dot{x}...x^{(\nu+1)}}) = r - a$ on $\mathbb{L}_\nu$, such that there exist smooth matrix functions Z_2 and T_2 defined on $\mathbb{L}_\nu$ of size $(\nu+1)m \times a$ and $n \times n - a$, respectively, having full rank and satisfying $Z_2^T \mathfrak{F}_{\nu,\dot{x}...x^{(\nu+1)}} = 0$, $\operatorname{rank}(Z_2^T \mathfrak{F}_{\nu,x}) = a$, and $Z_2^T \mathfrak{F}_{\nu,x} T_2 = 0$ on $\mathbb{L}_\nu$.
5) We have $\operatorname{rank}(F_{,\dot{x}} T_2) = d = m - a - v$ on $\mathbb{L}_\nu$, such that there exists a smooth matrix function Z_1 defined on $\mathbb{L}_\nu$ of size $m \times d$ having full rank and satisfying $Z_1^T F_{,\dot{x}} T_2$ having full rank on $\mathbb{L}_\nu$.

Remark 3.2.8 It is important to note that the term "smooth" in connection to the matrix functions Z_1, Z_2, and T_2 means smoothness of this matrix functions along a solution with respect to t, i.e., smooth totally depending on t along a solution, even if the matrix functions formally are defined on $\mathbb{L}_\nu$. □

For linear systems (3.3) the Hypothesis 3.2.7 reduces to the Hypothesis 3.2.6. The assumptions on the DAE (3.2a) made in the Hypothesis are not as restrictive as they are made for the considerations with respect to the solvability in [31]. Both hypotheses allow the consideration of redundancies and underdeterminedness. Furthermore, the constant rank assumptions are not required in a neighborhood of the solution in the entire space but only in a submanifold. In addition less smoothness of the function F is required. For more details we refer to [104, 105].
Based on the Hypothesis 3.2.7 (and 3.2.6) the *strangeness index* (s-index) is defined as follows.

Definition 3.2.9 (Strangeness index, strangeness-free) *If the right-hand side $F(x, \dot{x}, u)$ of a general nonlinear DAE (3.2a) satisfies Hypothesis 3.2.7, then the* strangeness index *(s-index), denoted by ν_s, is defined to be the smallest integer ν for which $F(x, \dot{x}, u)$ satisfies Hypothesis 3.2.7.*
If the DAE $F(x, \dot{x}, u) = 0$ has vanishing s-index, i.e., $\nu_s = 0$, then the DAE is called strangeness-free.

Remark 3.2.10 a) Assume that the coefficients $E(t)$, $A(t)$ of (3.3) satisfy Hypothesis 3.2.6. Then the least ν for which the linear DAE (3.3) satisfies Hypothesis 3.2.6 is the s-index, i.e., $\nu_s = \nu$, of the linear DAE (3.3). A DAE (3.2a) with $m = n$ and $v = 0$ is called regular.
b) The quantities r, d, a, v, and, in particular, ν_s are called *characteristic quantities* of the DAE, see [102].

c) If any $\mathbb{L}_i = \emptyset$ then the DAE (3.2a) does not satisfy Hypothesis 3.2.7 because from $\mathbb{L}_i = \emptyset$ it follows that $\mathbb{L}_j = \emptyset$ for all $j > i$ and this is not a manifold, see Definition 2.3.6. In particular, this means that DAEs with contradictory algebraic constraints are excluded by the Hypothesis 3.2.7. □

Remark 3.2.11 The s-index generalizes the d-index to overdetermined and underdetermined systems. Furthermore, the relation between the d-index and the s-index for linear DAEs with variable coefficients of the form (3.3) is discussed in [101]. If the d-index as well as the s-index are well defined, than it is shown that locally the relation

$$\nu_s = \begin{cases} 0 & \text{for } \nu_d = 0, \\ \nu_d - 1 & \text{for } \nu_d > 0 \end{cases}$$

between the s-index and the d-index is satisfied. □

Example 3.2.12 Let us consider the DAE (3.12) which is already investigated in Example 3.2.5 in view of its d-index. Let us investigate the DAE (3.12) in view of the Hypothesis 3.2.7. Let us start with $\nu = 0$. For $\nu = 0$ we get that the set of solutions $\mathbb{L}_0 = \{(x, \dot{x}) \in \mathbb{R}^3 \times \mathbb{R}^3 \ : \ \dot{x}_1 = x_1 x_2,\ x_1 \dot{x}_3 = x_2 + 1,\ x_3 = 1\}$ forms a manifold of dimension $3 = (\nu + 2)n + (\nu + 1)n_u - r$ with $m = n = 3$, $n_u = 0$, and $r = 3$. Furthermore, we have that the derivative array $\mathfrak{F}_0$ of level 0 equals the DAE, i.e., we have

$$0 = \mathfrak{F}_0 = \begin{bmatrix} \dot{x}_1 - x_1 x_2 \\ x_1 \dot{x}_3 - x_2 + 1 \\ x_3 - 1 \end{bmatrix}$$

and we get the partial derivative as

$$\mathfrak{F}_{0,x\dot{x}} = \begin{bmatrix} -x_2 & -x_1 & 0 & 1 & 0 & 0 \\ \dot{x}_3 & -1 & 0 & 0 & 0 & x_1 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

with $\operatorname{rank}(\mathfrak{F}_{0,x\dot{x}}) = 3 = r$. Furthermore, we get $\operatorname{corank}(\mathfrak{F}_{0,x\dot{x}}) - \operatorname{corank}(\mathfrak{F}_{-1,x}) = 0 - 0 = 0 = v$. In point 4) of Hypothesis 3.2.7 we have the condition $\operatorname{rank}(\mathfrak{F}_{0,\dot{x}}) = r - a$, but we have that

$$\operatorname{rank}(\mathfrak{F}_{0,\dot{x}}) = \operatorname{rank}(F_{,\dot{x}}) = \operatorname{rank}(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & x_1 \\ 0 & 0 & 0 \end{bmatrix}) = \begin{cases} 2 & \text{for } x_1 \neq 0 \\ 1 & \text{for } x_1 = 0 \end{cases}$$

and therefore $\operatorname{rank}(\mathfrak{F}_{0,\dot{x}}) \neq \text{const}$ and the Hypothesis 3.2.7 is not satisfied for $\nu = 0$. Increasing ν by one yields the derivative array

$$\mathfrak{F}_1 = \begin{bmatrix} \dot{x}_1 - x_1 x_2 \\ x_1 \dot{x}_3 - x_2 + 1 \\ x_3 - 1 \\ \ddot{x}_1 - \dot{x}_1 x_2 - x_1 \dot{x}_2 \\ \dot{x}_1 \dot{x}_3 + x_1 \ddot{x}_3 - \dot{x}_2 \\ \dot{x}_3 \end{bmatrix}$$

of level 1 with its derivative

$$\mathfrak{F}_{1,x\dot{x}\ddot{x}} = \begin{bmatrix} -x_2 & -x_1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ \dot{x}_3 & -1 & 0 & 0 & 0 & x_1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\dot{x}_2 & -\dot{x}_1 & 0 & -x_2 & -x_1 & 0 & 1 & 0 & 0 \\ \ddot{x}_3 & 0 & 0 & \dot{x}_3 & -1 & \dot{x}_1 & 0 & 0 & x_1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

We get that the set of solutions $\mathbb{L}_1$ forms a manifold of dimension $3 = (\nu+2)n + (\nu+1)n_u - r$ with $r = 6$. Furthermore, we have $\operatorname{rank}(\mathfrak{F}_{1,x\dot{x}\ddot{x}}) = 6 = r$ and we get $\operatorname{corank}(\mathfrak{F}_{1,x\dot{x}\ddot{x}}) - \operatorname{corank}(\mathfrak{F}_{0,x\dot{x}}) = 0 - 0 = 0 = v$. From point 4) of Hypothesis 3.2.7 we get that $\operatorname{rank}(\mathfrak{F}_{0,\dot{x}\ddot{x}}) = 4 = r - a = 6 - a$ and it follows $a = 2$ and the existence of the matrix functions Z_2^T of size 2×6 and T_2 of size 3×1 having full rank and satisfying $Z_2^T \mathfrak{F}_{1,\dot{x}\ddot{x}} = 0$, $\operatorname{rank}(Z_2^T \mathfrak{F}_{1,x}) = a = 2$, and $Z_2^T \mathfrak{F}_{1,x} T_2 = 0$, e.g., with

$$Z_2^T = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -x_1 \end{bmatrix} \quad \text{and} \quad T_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}. \tag{3.14a}$$

Note that from $0 = \mathfrak{F}_1$ it follows that $\dot{x}_3 = 0$ and, therefore, we have $x_2 = 1$. In the last point of Hypothesis 3.2.7 the condition $\operatorname{rank}(F_{,x} T_2) = d = m - a - v = 1$ has to be satisfied. We get

$$\operatorname{rank}(F_{,x} T_2) = \operatorname{rank}\Big(\begin{bmatrix} -x_2 \\ 0 \\ 0 \end{bmatrix} = \operatorname{rank}\Big(\begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} = 1$$

and consequently the existence of the matrix function Z_1^T of size 1×3 having full rank and satisfying $\operatorname{rank}(Z_1^T F_{,x} T_2) = d = 1$, e.g., with

$$Z_1 = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}. \tag{3.14b}$$

Therefore, the DAE (3.12) has s-index $\nu_s = 1$. □

While the previous index concepts are more or less based on the structure of the DAE, in [79, 82] the so-called *perturbation index* (p-index) is introduced, which is a measure of the sensitivity of the solution with respect to perturbations of the DAE.

Definition 3.2.13 (Perturbation index) *The differential-algebraic equation (3.2a) with $u(t) = t$ has* perturbation index *(p-index) ν_p along a solution $x(t)$ on $t \in \mathbb{I}$, if ν_p is the smallest integer such that, for all solutions $y(t)$ of the perturbed differential-algebraic equation*

$$F(y, \dot{y}, t) = \delta(t)$$

with the defect $\delta(t)$ there exists on $\mathbb{I}$ an estimate

$$||y(t) - x(t)|| \leq C\Big(||y(t_0) - x(t_0)|| + \max_{t_0 \leq \xi \leq t} ||\delta(t)|| + ... + \max_{t_0 \leq \xi \leq t} ||\delta^{(\nu_p - 1)}(t)|| \Big),$$

whenever the expression on the right-hand side is sufficiently small.

Example 3.2.14 Consider a set of differential-algebraic equations of the form

$$\dot{x}_1 = k_1(x_1, x_2), \tag{3.15a}$$
$$0 = k_2(x_1, x_2) \tag{3.15b}$$

with $x_1 \in \mathbb{X}_1 \subset \mathbb{R}^{n_1}$, $x_2 \in \mathbb{X}_2 \subset \mathbb{R}^{n_2}$, $\mathbb{X}_1 \times \mathbb{X}_2 = \mathbb{X}$, and $t \in \mathbb{I}$. This special case is called *semi-explicit DAE*. If $k_{2,x_2}(x_1, x_2)$ has a bounded inverse for all $(x_1, x_2) \in \{(x_1, x_2) \in \mathbb{X}_1 \times \mathbb{X}_2 \ : \ 0 = k_2(x_1, x_2)\}$, then the DAE has d-index 1, s-index 0, and p-index 1. □

The class of semi-explicit DAEs of form (3.15) has been extensively investigated and is well understood, see for instance [25, 73, 79]. In the fully implicit case the situation is much more different. As a rule of thumb, the higher the index of a DAE is, the more complicated is its numerical analysis, and the more careful one has to be in the numerical solution of the problem. The difficulty in the numerical solution of high index problems is discussed in [25, 66, 69, 73, 79, 82, 103, 104, 105, 133, 134].

Lemma 3.2.15 *If the nonlinear DAE (3.2a) with $m = n$ is strangeness-free, i.e., has vanishing s-index, and* $\operatorname{rank}(F_{,x\dot{x}}(x,\dot{x},u)) = n$ *for all* $(x,\dot{x},u) \in \mathbb{L}_0$*, then the matrix pair* $(F_{,\dot{x}}(x,\dot{x},u), F_{,x}(x,\dot{x},u))$ *is regular, i.e., there exists* $\tau \in \mathbb{R}$ *such that* $F_{,\dot{x}}(x,\dot{x},u) + \tau F_{,x}(x,\dot{x},u)$ *is a nonsingular matrix for all* $(x,\dot{x},u) \in \mathbb{L}_0$*, see (3.13).*

Proof: Since the DAE is assumed to be strangeness-free, it satisfies Hypothesis 3.2.7 with $\nu = \nu_s = 0$. Therefore, the associated derivative array is the original DAE, i.e., $\mathfrak{F}_0(x,\dot{x},u) = F(x,\dot{x},u)$. From $\operatorname{rank}(F_{,x\dot{x}}(x,\dot{x},u)) = n$ it follows that $v = 0$, and for $a = n - \operatorname{rank}(F_{,\dot{x}})$ and $d = n - a$ we have the existence of a matrix function Z_2 of size $n \times a$ satisfying $Z_2^T F_{,\dot{x}} = 0$ and $\operatorname{rank}(Z_2^T F_{,x}) = a$, and the existence of a matrix function T_2 of size $n \times d$ satisfying $Z_2^T F_{,x} T_2 = 0$. Furthermore, there exists a matrix function Z_1 of size $n \times d$ satisfying $\operatorname{rank}(Z_1^T F_{,\dot{x}} T_2) = \operatorname{rank}(F_{,\dot{x}} T_2) = d$. Therefore, we have that $Z_1^T F_{,\dot{x}} T_2$ of size $d \times d$ and that $Z_2^T F_{,x} T_1$ of size $a \times a$ are nonsingular, with $T_1 \in \mathbb{R}^{n,a}$ such that $\begin{bmatrix} T_1 & T_2 \end{bmatrix}$ is nonsingular. For $0 \neq \tau \in \mathbb{R}$ we get

$$\begin{aligned} &\begin{bmatrix} Z_1^T \\ \frac{1}{\tau} Z_2^T \end{bmatrix} \begin{bmatrix} F_{,\dot{x}} + \tau F_{,x} \end{bmatrix} \begin{bmatrix} T_1 & T_2 \end{bmatrix} \\ &= \begin{bmatrix} Z_1^T F_{,\dot{x}} T_1 + \tau Z_1^T F_{,x} T_1 & Z_1^T F_{,\dot{x}} T_2 + \tau Z_1^T F_{,x} T_2 \\ Z_2^T F_{,x} T_1 & 0 \end{bmatrix}. \end{aligned} \tag{3.16}$$

Due to the nonsingularity of $Z_1^T F_{,\dot{x}} T_2$ and $Z_2^T F_{,x} T_1$, we get from Proposition 2.1.2 the result for $0 \neq \tau \in \mathbb{R}$ sufficiently small, i.e., the nonsingularity of the matrix on the right-hand side in (3.16). Furthermore, from (3.16) it follows that the matrices $\begin{bmatrix} Z_1 & \frac{1}{\tau} Z_2 \end{bmatrix}$ and, in particular, $\begin{bmatrix} F_{,\dot{x}} + \tau F_{,x} \end{bmatrix}$ are nonsingular for sufficiently small $\tau \neq 0$. Then, we get the regularity of the matrix pair $(F_{,\dot{x}}(x,\dot{x},u), F_{,x}(x,\dot{x},u))$. □

Remark 3.2.16 If the s-index of the DAE is higher than zero the statement of Lemma 3.2.15 does not hold. In [100] the linear DAE (3.3) with

$$E(t) = \begin{bmatrix} 0 & 0 \\ 1 & -t \end{bmatrix}, \quad A(t) = \begin{bmatrix} -1 & t \\ 0 & 0 \end{bmatrix}$$

is given. This DAE is of s-index one with a unique solution, but the matrix pair $(F_{,\dot{x}}(x,\dot{x},u), F_{,x}(x,\dot{x},u)) = (E(t), -A(t))$ is singular for all t. □

Remark 3.2.17 In general, additionally to the constraints explicitly occurring in the DAE, the solution of higher index DAEs is restricted by constraints which are hidden in the DAE, i.e., they are not explicitly stated. These constraints impose additional consistency conditions on the initial values and provoke severe difficulties in the direct numerical integration of DAEs of higher index, see [25, 66, 69, 73, 79, 82, 133, 134, 166]. The consideration of such hidden constraints is done in detail for quasi-linear DAEs in Section 3.5. □

3.3 Linearization of differential-algebraic equations

Let us discuss the linearization of DAEs (3.2a) in a function space along a reference trajectory $\bar{x}(t)$ which does not necessarily have to be a solution of the original DAE (3.2a). The linearization in function space is called *quasi-linearization*, see [30, 52]. For $x(t) = \bar{x}(t) + \hat{x}(t)$ we get from (3.2a) that

$$0 = F(\bar{x}(t) + \hat{x}(t), \dot{\bar{x}}(t) + \dot{\hat{x}}(t), u).$$

From the Taylor[5] expansion we get

$$0 = F(\bar{x}(t), \dot{\bar{x}}(t), u) + F_{,x}(\bar{x}(t), \dot{\bar{x}}(t), u)\hat{x} + F_{,\dot{x}}(\bar{x}(t), \dot{\bar{x}}(t), u)\dot{\hat{x}}(t) + \Phi.$$

Here, Φ sums up all expressions which contain higher order terms, i.e., which contain terms with $\hat{x}^i\dot{\hat{x}}^j$ with $i + j \leq 2$. Neglecting the higher order terms Φ, we get a linearization of (3.2a) in the form (3.3) with

$$E(t) = F_{,\dot{x}}(\bar{x}(t), \dot{\bar{x}}(t), u),\ A(t) = -F_{,x}(\bar{x}(t), \dot{\bar{x}}(t), u),\ k(t) = -F(\bar{x}(t), \dot{\bar{x}}(t), u), \quad (3.17)$$

and x in (3.3) corresponds to $\hat{x}$.

Remark 3.3.1 Note that in the case of $\bar{x}(t)$ being a solution of (3.2a) the inhomogeneity $k(t)$ equals zero, and in the case that $\bar{x}(t)$ and $u(t)$ are constant with respect to t we get a linear DAE with constant matrices E and A. □

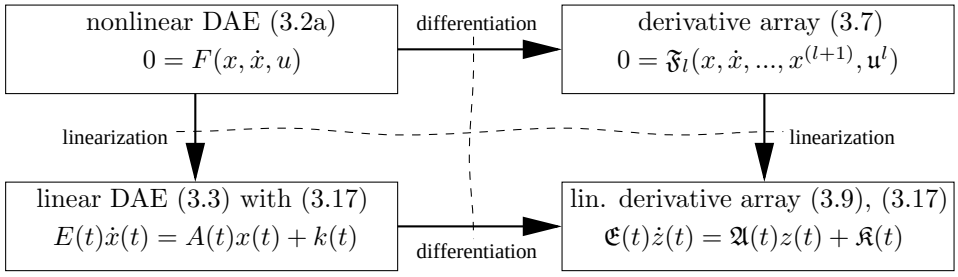

Figure 3.1: Linearization and differentiation of nonlinear DAEs

The linearization of DAEs along solutions of (3.2a) is discussed in [30]. There, it is shown that the linearization of the derivative array (3.7) along a constant solution yields the same as the derivative array based on the linearization along a constant solution of the nonlinear DAE (3.2a). However, a more general result can be obtained independent of the kind of chosen reference trajectory, see Figure 3.1 and the following Lemma 3.3.2.

Lemma 3.3.2 *Let the derivative array (3.7) of the DAE (3.2a) and the linearization of this derivative array along an arbitrary reference trajectory $\bar{x}$ and its derivatives $\bar{x}^{(i)}$ with respect to t be well defined. Then the derivative array (3.9) of the linearized DAE (3.3) with (3.17) is well defined and identical to the linearized derivative array of the original DAE (3.2a) along the reference trajectory $\bar{x}$ and its derivatives $\bar{x}^{(i)}$ with respect to t.*

Proof: Consider the DAE (3.2a) and its derivative array (3.7) of level 1, i.e., with $l = 1$,

$$0 = \mathfrak{F}_1(x, \dot{x}, \ddot{x}, \mathfrak{u}^1) = \begin{bmatrix} F(x, \dot{x}, u) \\ F_{,x}(x, \dot{x}, u)\dot{x} + F_{,\dot{x}}(x, \dot{x}, u)\ddot{x} + F_{,u}(x, \dot{x}, u)\dot{u} \end{bmatrix}.$$

For $x = \bar{x} + \hat{x}$, the linearization along the reference trajectory $\bar{x}(t)$ yields

$$\begin{aligned} 0 \ = \ & \begin{bmatrix} \bar{F}_{,x} & \bar{F}_{,\dot{x}} & 0 \\ (\bar{F}_{,x}\dot{\bar{x}} + \bar{F}_{,\dot{x}}\ddot{\bar{x}} + \bar{F}_{,u}\dot{u})_{,x} & (\bar{F}_{,x}\dot{\bar{x}} + \bar{F}_{,\dot{x}}\ddot{\bar{x}} + \bar{F}_{,u}\dot{u})_{,\dot{x}} & \bar{F}_{,\dot{x}} \end{bmatrix} \begin{bmatrix} \hat{x} \\ \dot{\hat{x}} \\ \ddot{\hat{x}} \end{bmatrix} \\ & + \begin{bmatrix} \bar{F} \\ \bar{F}_{,x}\dot{\bar{x}} + \bar{F}_{,\dot{x}}\ddot{\bar{x}} + \bar{F}_{,u}\dot{u} \end{bmatrix}. \end{aligned} \quad (3.18)$$

[5]Brook Taylor (born 1685 in Edmonton, Middlesex, England - died 1731 in Somerset House, London, England)

Note that the partial derivatives of the function F in (3.18) are evaluated along the reference trajectory $\bar{x}$ and its derivative $\dot{\bar{x}}$. This is denoted by a bar on top, i.e., $\bar{F}_{,x} = F_{,x}(\bar{x}, \dot{\bar{x}}, u)$, $\bar{F}_{,\dot{x}} = F_{,\dot{x}}(\bar{x}, \dot{\bar{x}}, u)$, and $\bar{F}_{,u} = F_{,u}(\bar{x}, \dot{\bar{x}}, u)$. For the first two entries of the last block row we get with Notation 2.2.3 and Lemma 2.2.5 that

$$\begin{aligned}(\bar{F}_{,x}\dot{\bar{x}} + \bar{F}_{,\dot{x}}\ddot{\bar{x}} + \bar{F}_{,u}\dot{u})_{,x} &= (\bar{F}_{,x}\dot{\bar{x}})_{,x} + (\bar{F}_{,\dot{x}}\ddot{\bar{x}})_{,x} + (\bar{F}_{,u}\dot{u})_{,x} \\ &= \bar{F}_{,xx}[\![\dot{\bar{x}}, \cdot]\!] + \bar{F}_{,\dot{x}x}[\![\ddot{\bar{x}}, \cdot]\!] + \bar{F}_{,ux}[\![\dot{u}, \cdot]\!] \\ &= \bar{F}_{,xx}[\![\cdot, \dot{\bar{x}}]\!] + \bar{F}_{,x\dot{x}}[\![\cdot, \ddot{\bar{x}}]\!] + \bar{F}_{,xu}[\![\cdot, \dot{u}]\!] \\ &= \frac{\mathrm{d}}{\mathrm{d}t}\bar{F}_{,x}\end{aligned}$$

and

$$\begin{aligned}(\bar{F}_{,x}\dot{\bar{x}} + \bar{F}_{,\dot{x}}\ddot{\bar{x}} + \bar{F}_{,u}\dot{u})_{,\dot{x}} &= (\bar{F}_{,x}\dot{\bar{x}})_{,\dot{x}} + (\bar{F}_{,\dot{x}}\ddot{\bar{x}})_{,\dot{x}} + (\bar{F}_{,u}\dot{u})_{,\dot{x}} \\ &= \bar{F}_{,x\dot{x}}[\![\dot{\bar{x}}, \cdot]\!] + \bar{F}_{,x} + \bar{F}_{,\dot{x}\dot{x}}[\![\ddot{\bar{x}}, \cdot]\!] + \bar{F}_{,u\dot{x}}[\![\dot{u}, \cdot]\!] \\ &= \bar{F}_{,\dot{x}x}[\![\cdot, \dot{\bar{x}}]\!] + \bar{F}_{,x} + \bar{F}_{,\dot{x}\dot{x}}[\![\cdot, \ddot{\bar{x}}]\!] + \bar{F}_{,\dot{x}u}[\![\cdot, \dot{u}]\!] \\ &= \frac{\mathrm{d}}{\mathrm{d}t}\bar{F}_{,\dot{x}} + \bar{F}_{,x}.\end{aligned}$$

Therefore, the linearization (3.18) of the derivative array is in the form

$$0 = \begin{bmatrix} \bar{F}_{,x} & \bar{F}_{,\dot{x}} & 0 \\ \frac{\mathrm{d}}{\mathrm{d}t}\bar{F}_{,x} & \frac{\mathrm{d}}{\mathrm{d}t}\bar{F}_{,\dot{x}} + \bar{F}_{,x} & \bar{F}_{,\dot{x}} \end{bmatrix} \begin{bmatrix} \hat{x} \\ \dot{\hat{x}} \\ \ddot{\hat{x}} \end{bmatrix} + \begin{bmatrix} \bar{F} \\ \bar{F}_{,x}\dot{\bar{x}} + \bar{F}_{,\dot{x}}\ddot{\bar{x}} + \bar{F}_{,u}\dot{u} \end{bmatrix},$$

which is equivalent to

$$\begin{aligned}\begin{bmatrix} \bar{F}_{,\dot{x}} & 0 \\ \frac{\mathrm{d}}{\mathrm{d}t}\bar{F}_{,\dot{x}} + \bar{F}_{,x} & \bar{F}_{,\dot{x}} \end{bmatrix} \begin{bmatrix} \dot{\hat{x}} \\ \ddot{\hat{x}} \end{bmatrix} &= \begin{bmatrix} -\bar{F}_{,x} & 0 \\ -\frac{\mathrm{d}}{\mathrm{d}t}\bar{F}_{,x} & 0 \end{bmatrix} \begin{bmatrix} \hat{x} \\ \dot{\hat{x}} \end{bmatrix} \\ &\quad - \begin{bmatrix} \bar{F} \\ \bar{F}_{,x}\dot{\bar{x}} + \bar{F}_{,\dot{x}}\ddot{\bar{x}} + \bar{F}_{,u}\dot{u} \end{bmatrix}.\end{aligned} \tag{3.19}$$

With (3.17) this corresponds to

$$\begin{bmatrix} E(t) & 0 \\ \dot{E}(t) - A(t) & E(t) \end{bmatrix} \begin{bmatrix} \dot{\hat{x}} \\ \ddot{\hat{x}} \end{bmatrix} = \begin{bmatrix} A(t) & 0 \\ \dot{A}(t) & 0 \end{bmatrix} \begin{bmatrix} \hat{x} \\ \dot{\hat{x}} \end{bmatrix} + \begin{bmatrix} k(t) \\ \dot{k}(t) \end{bmatrix},$$

which is exactly the derivative array of level 1 for the linearized DAE (3.3) with (3.17).
The proof for arbitrary $l > 1$ follows inductively according to Remark 3.2.2. □

Remark 3.3.3 a) Note that until this point the linearization along an arbitrary trajectory has been considered. In principle, it is not necessary to require that the reference trajectory is a solution. However, one has to be very careful with respect to the choice of the reference trajectory. It is important that all characteristic quantities (see Remark 3.2.10b) of the DAE along the reference trajectory are the same as along the set of solutions $\mathbb{L}_{\nu_s}$. In particular, the d-index is not invariant under linearization along a trajectory. A counterexample is given in [30] and another one is given in the following Example 3.3.4.
b) The linearization along a solution, in particular, along an equilibrium state (which is constant), offers the possibility for the investigation of local stability properties of DAEs, see [73, 119, 175]. □

Example 3.3.4 Consider the DAE in Example 3.2.5. The linearization along the solution $\bar{x} = \begin{bmatrix} 0 & 1 & 1 \end{bmatrix}^T$ yields

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\hat{x}}_1 \\ \dot{\hat{x}}_2 \\ \dot{\hat{x}}_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \\ \hat{x}_3 \end{bmatrix},$$

which is of d-index one and not two. □

3.4 Regularization of differential-algebraic equations of higher index

It is known that, in general, the direct discretization of higher index DAEs may lead to wrong results, e.g., suffering from *drift-off phenomena*, instabilities, or order reduction, see [25, 73, 79, 82, 105, 133, 134]. Therefore, in the numerical integration of DAEs not only the discretization schemes have to be analyzed but also a variety of equivalent transformations of the DAE often called stabilization or regularization techniques. In the following, we will discuss some stabilization techniques for nonlinear DAEs of the form (3.2a).

One of the first ideas to stabilize DAEs was given by Baumgarte in [18] for DAEs arising in multibody dynamics (see Section 4.6.1.2) and for the stabilization of ODEs with invariants in [19]. Furthermore, Gear considered the stabilization of ODEs with invariants [65]. Further ideas are for instance lowering the index by differentiation of the constraints [25, 66, 82, 164], singularly perturbed problems [97, 114, 117], and the recently developed strangeness concept [102, 103, 104, 105].

3.4.1 Regularization by differentiation

A first idea for the regularization of a DAE of higher index is to lower the index by replacing the constraints (if explicitly available) by their derivative(s) with respect to t. In fact, it is possible to lower the index in this way but differentiation neglects some constants which are deleted by the differentiation in the constraints. Neglecting these constants leads to the so-called drift-off phenomenon, see [25, 59, 82, 164, 167, 170]. This phenomenon means that due to round-off errors or inconsistent initial values with respect to the original DAE the solution drifts away from the set of solutions during the numerical integration. The obtained approximation to the solution is consistent with the regularized DAE but it is not consistent with the originally given DAE.

The drift-off phenomenon is discussed in more detail with respect to the model equations arising in multibody dynamics in Section 4.4.

A way out of this dilemma is given in [66]. There, the idea is to differentiate the constraints as many times as it is necessary to determine an ODE in the unknowns x and its derivative $\dot{x}$ from the reduced derivative array (3.8). As mentioned above, explicit information about the solution restrictions is lost but the constraints remain as invariants in the obtained ODE. In order to enforce the numerical solution to satisfy all constraints, the idea is to consider the obtained ODE and all constraints simultaneously. But this leads to an overdetermined system, although it is consistent for solutions of the original DAE. To avoid this overdeterminacy, the introduction of additional variables is proposed in [65]. It is shown that the solution associated with these additional variables is zero.

Furthermore, it is not necessary to differentiate the constraints as many times as one needs to get an ODE. Rather, it is possible to lower the index by a certain number of differentiations. This does not lead to an ODE, but to a DAE of lower

index than the original DAE.
One should note that this procedure may not always be practical or suitable, but in some problems the structure of the equations is such that the manipulations are simple. For instance, the structure of the model equations of mechanical systems makes it possible to carry out these manipulations in a simple way which leads to the so-called Gear-Gupta-Leimkuhler formulation, see Section 4.6.2.1.

3.4.2 The strangeness-concept

According to Hypothesis 3.2.6 we define the linear DAE

$$\begin{bmatrix} \hat{E}_1(t) \\ 0 \\ 0 \end{bmatrix} \dot{x}(t) = \begin{bmatrix} \hat{A}_1(t) \\ \hat{A}_2(t) \\ 0 \end{bmatrix} x(t) + \begin{bmatrix} \hat{k}_1(t) \\ \hat{k}_2(t) \\ \hat{k}_3(t) \end{bmatrix}, \tag{3.20}$$

where

$$\begin{aligned} \hat{E}_1 = Z_1^T E, \quad \hat{A}_1 = Z_1^T A, \quad & \hat{k}_1 = Z_1^T k, \\ \hat{A}_2 = Z_2^T \mathfrak{A}^{\nu_s}, \quad & \hat{k}_2 = Z_2^T \mathfrak{K}^{\nu_s}, \\ & \hat{k}_3 = Z_3^T \mathfrak{K}^{\nu_s}. \end{aligned}$$

By construction it follows that this DAE is strangeness-free. Furthermore, it follows from Hypothesis 3.2.6 that all solutions of the linear DAE (3.3) are also solutions of (3.20), and in [102] it has been shown that in addition all solutions of (3.20) are also solutions of (3.3). Therefore, both formulations are equivalent in the sense that they have the same solution set and (3.20) represents a regularization of the linear DAE (3.3). This yields the following definition.

Definition 3.4.1 (Linear equivalent strangeness-free formulation) *If the linear DAE (3.3) satisfies the Hypothesis 3.2.6 with s-index ν_s, then the associated linear DAE (3.20) is called* equivalent strangeness-free formulation *of the linear DAE (3.3).*

With respect to general nonlinear DAEs (3.2a) we can define an analogous formulation as follows.

Definition 3.4.2 (Equivalent strangeness-free formulation) *If the nonlinear DAE (3.2a) satisfies the Hypothesis 3.2.7 with s-index ν_s, then the associated DAE*

$$0 = Z_1^T F(x(t), \dot{x}(t), u(t)), \tag{3.21a}$$
$$0 = Z_2^T \mathfrak{F}_{\nu_s}(x(t), \dot{x}(t), ..., x^{(\nu_s+1)}(t), \mathfrak{u}^{\nu_s}(t)) \tag{3.21b}$$

is called equivalent strangeness-free formulation *of the nonlinear DAE (3.2a).*

Even though the derivative array $\mathfrak{F}_{\nu_s}$ of level ν_s does depend in addition to the unknown variables x, its derivative $\dot{x}$, and the control variables u also on $\ddot{x}, ..., x^{(\nu_s+1)}$ and $\dot{u}, ..., u^{(\nu_s)}$, the equivalent strangeness-free formulation only depends on the unknown variables x, its derivative $\dot{x}$, and the control variables u. This is obtained by the particular choice of the matrix function Z_2 as stated in Hypothesis 3.2.7.

Remark 3.4.3 In [104] two theorems are presented which prove that the solution set of the DAE (3.2a) and the solution set of (3.21) locally are identical. Here, the term "locally" means locally with respect to all variables x, $\dot{x}$, and u in the set of solutions $\mathbb{L}_{\nu_s}$.
However, together with Remark 3.2.8 it cannot be guaranteed in general that the two solution sets are equal. Indeed caused from the particular global choice of the matrix

functions Z_1 and Z_2 smoothly depending on x, $\dot{x}$, ..., $x^{(\nu_s+1)}$, and $\mathfrak{u}^{\nu_s}$ independent of a solution the solution set of the equivalent strangeness-free formulation (3.21) can be larger than the solution set of the original DAE (3.2a). □

Example 3.4.4 Let us consider the DAE (3.12), i.e.,

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & x_1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} x_1 x_2 \\ x_2 - 1 \\ x_3 - 1 \end{bmatrix}, \tag{3.22}$$

with $\mathbb{X} = \mathbb{R}^3$ and $t \in \mathbb{I}$ which is already investigated in Example 3.2.5 in view of its d-index and in Example 3.2.12 in view of its s-index and where we have found that the DAE (3.22) has d-index $\nu_d = 2$ and s-index $\nu_s = 1$. Furthermore, in Example 3.2.12 we did already determine the matrix functions Z_1^T and Z_2^T in (3.14). Therefore, we obtain the equivalent strangeness-free formulation (3.21) of the DAE (3.22) by

$$0 = \begin{bmatrix} Z_1^T F \\ \hline Z_2^T \mathfrak{F}_1 \end{bmatrix} = \begin{bmatrix} \dot{x}_1 - x_1 x_2 \\ \hline x_3 - 1 \\ (x_1 \dot{x}_3 - x_2 + 1) - x_1(\dot{x}_3) \end{bmatrix} = \begin{bmatrix} \dot{x}_1 - x_1 x_2 \\ \hline x_3 - 1 \\ -x_2 + 1 \end{bmatrix}$$

consisting of $d = 1$ differential equations and $a = 2$ algebraic equations. □

3.5 Quasi-linear differential-algebraic equations

In this section we will investigate in detail quasi-linear DAEs of the form

$$E(x(t), u(t))\dot{x}(t) = k(x(t), u(t)), \tag{3.23}$$

where $E \in \mathcal{C}(\mathbb{X} \times \mathbb{U}, \mathbb{R}^{m,n})$ is called *leading matrix of the quasi-linear DAE* and $k \in \mathcal{C}(\mathbb{X} \times \mathbb{U}, \mathbb{R}^m)$ is called *right-hand side of the quasi-linear DAE*, $\mathbb{X} \subset \mathbb{R}^n$ and $\mathbb{U} \subset \mathbb{R}^{n_u}$. Furthermore, we call $x \in \mathcal{C}^1(\mathbb{I}, \mathbb{R}^n)$ the *unknown variables* and the functions $u \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{n_u})$ are given control variables.
Furthermore, we will investigate *semi-implicit DAEs* of the form

$$E_1(x(t), u(t))\dot{x}(t) = k_1(x(t), u(t)), \tag{3.24a}$$
$$0 = k_2(x(t), u(t)) \tag{3.24b}$$

as special case of a quasi-linear DAE (3.23), where $E_1 \in \mathcal{C}(\mathbb{X} \times \mathbb{U}, \mathbb{R}^{m_1,n})$, $k_1 \in \mathcal{C}(\mathbb{X} \times \mathbb{U}, \mathbb{R}^{m_1})$, $k_2 \in \mathcal{C}(\mathbb{X} \times \mathbb{U}, \mathbb{R}^{m_2})$, and $x \in \mathcal{C}^1(\mathbb{I}, \mathbb{R}^n)$. If the leading matrix in (3.24a) satisfies $E_1(x, u) = \begin{bmatrix} I & 0 \end{bmatrix}$ then the DAE (3.24) corresponds to a semi-explicit DAE (3.15).
In view of the treatment of the model equations of mechanical systems, see Chapter 4, the class of DAEs in *Hessenberg*[6] *form* plays an important role. A quasi-linear DAE (3.23) is in Hessenberg form of order $r \geq 2$ if it can be written as

$$\dot{x}_1 = k_1(x_1(t), x_2(t), ..., x_{r-1}(t), x_r(t), u(t)), \tag{3.25a}$$
$$\dot{x}_i = k_i(x_{i-1}(t), ..., x_{r-1}(t), u(t)), \qquad i = 2, ..., r-1, \tag{3.25b}$$
$$0 = k_r(x_{r-1}(t), u(t)) \tag{3.25c}$$

with $(\partial k_r/\partial x_{r-1}) \cdot (\partial k_{r-1}/\partial x_{r-2}) \cdot ... \cdot (\partial k_2/\partial x_1) \cdot (\partial k_1/\partial x_r)$ nonsingular for all consistent x and $k_i \in \mathcal{C}(\mathbb{X} \times \mathbb{U}, \mathbb{R}^{m_i})$ for $i = 1, ..., r$, $x_i \in \mathcal{C}^1(\mathbb{I}, \mathbb{R}^{m_i})$ for $i = 1, ..., r-1$, and $x_r \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{m_r})$. It is known that DAEs in Hessenberg form of order r have d-index $\nu_d = r$ and s-index $\nu_s = r - 1$, see [25, 82, 105].

[6] Gerhard Hessenberg (born 1874 in Frankfurt, Germany - died 1925 in Berlin, Germany)

3.5.1 Strangeness-free quasi-linear differential-algebraic equations

In this section we will discuss some important results concerning strangeness-free quasi-linear DAEs.

Lemma 3.5.1 *Suppose that (3.24) is a solvable system of semi-implicit differential-algebraic equations. Let the partial derivative $k_{2,x}(x,u)$ be defined. Then, the system (3.24) is strangeness-free if*

$$m_1 = \operatorname{rank}(E_1(x,u)) = \text{const}, \ r_C = \operatorname{rank}(k_{2,x}(x,u)) = \text{const}, \tag{3.26}$$

and

$$\operatorname{rank}(\begin{bmatrix} E_1(x,u) \\ k_{2,x}(x,u) \end{bmatrix}) = m_1 + r_C \tag{3.27}$$

for all $(x,u) \in \mathbb{M}$, where

$$\mathbb{M} = \{(x,u) \in \mathbb{X} \times \mathbb{U} \ : \ k_2(x,u) = 0\}. \tag{3.28}$$

Proof: In the following we will show that the semi-implicit DAE (3.24) with the stated assumptions satisfies the Hypothesis 3.2.7 for $\nu = 0$. Then from Definition 3.2.9 it follows that the semi-implicit DAE (3.24) is strangeness-free.
Therefore, we have to consider

$$\mathfrak{F}_0 = F = \begin{bmatrix} E_1\dot{x} - k_1 \\ -k_2 \end{bmatrix}.$$

From the constant rank condition (3.26) we get that $\mathbb{L}_0$ defined in (3.13) is a manifold of dimension $2n + n_u - r$ with $r = m_1 + r_C$, see point 1) of the Hypothesis 3.2.7. The point 2) of the Hypothesis 3.2.7 is satisfied since

$$\operatorname{rank}(\mathfrak{F}_{0,x\dot{x}}) = \operatorname{rank}(\begin{bmatrix} E_{1,x}[\![\dot{x},\cdot]\!] - k_{1,x} & E_1 \\ k_{2,x} & 0 \end{bmatrix}) = m_1 + r_C = r.$$

From 3) of Hypothesis 3.2.7 we get that $v = m_2 - r_C$, since $\operatorname{corank}(\mathfrak{F}_{0,x\dot{x}}) = m_2 - r_C$ and $\operatorname{corank}(\mathfrak{F}_{-1,x}) = 0$ by convention. Furthermore, from point 4) of Hypothesis 3.2.7 we get $a = r_C$ since $\operatorname{rank}(\mathfrak{F}_{0,\dot{x}}) = m_1 = r - a$ and we get the existence of the matrix function

$$Z_2^T = \begin{bmatrix} 0 & \widetilde{Z}_2^T \end{bmatrix}$$

of size $r_C \times m$ with $\operatorname{rank}(\widetilde{Z}_2^T k_{2,x}) = r_C = a$. Furthermore, we get the existence of a matrix function T_2 of size $n \times n - r_C$ such that $\operatorname{range}(T_2) = \ker(k_{2,x})$. In the last point 5) of Hypothesis 3.2.7 the condition

$$\operatorname{rank}(F_{,\dot{x}} T_2) = d = m - a - v$$

which corresponds to

$$\operatorname{rank}(\begin{bmatrix} E_1 \\ 0 \end{bmatrix} T_2) = d = m_1$$

has to be satisfied for all $(x,\dot{x},u) \in \mathbb{L}_0$. This and therefore, the assertion follows from (3.27). □

Remark 3.5.2 The condition (3.27) is equivalent to $\text{range}(E_1^T) \cap \text{range}(k_{2,x}^T) = \{0\}$ for all $(x,u) \in \mathbb{M} = \{(x,u) \in \mathbb{X} \times \mathbb{U} \ : \ k_2(x,u) = 0\}$. □

Lemma 3.5.3 *Suppose that the quasi-linear DAE (3.23) with* $\text{rank}(E(x,u)) = r_E = \text{const}$ *for all consistent values* (x,u) *is a solvable DAE. Then, the system (3.23) is strangeness-free if there exists a nonsingular matrix function*

$$S(x,u) = \begin{bmatrix} S_1(x,u) \\ S_2(x,u) \end{bmatrix} \in \mathcal{C}(\mathbb{X} \times \mathbb{U}, \mathbb{R}^{m,m}) \tag{3.29}$$

with $S_1(x,u)$ *is of size* $r_E \times m$ *with*

$$\begin{aligned} r_E &= \text{rank}(S_1(x,u)E(x,u)), && \text{(3.30a)} \\ r_C &= \text{rank}((S_2(x,u)k(x,u))_{,x}), && \text{(3.30b)} \\ 0 &= S_2(x,u)E(x,u), && \text{(3.30c)} \end{aligned}$$

and

$$\text{rank}\begin{pmatrix} S_1(x,u)E(x,u) \\ (S_2(x,u)k(x,u))_{,x} \end{pmatrix} = r_E + r_C \tag{3.31}$$

for all $(x,u) \in \mathbb{M}$*, where*

$$\mathbb{M} = \{(x,u) \in \mathbb{X} \times \mathbb{U} \ : \ S_2(x,u)k(x,u) = 0\}. \tag{3.32}$$

Proof: From the existence of the matrix function $S(x,u)$ it follows that the scaled DAE $S(x,u)E(x,u)\dot{x} = S(x,u)k(x,u)$ corresponds to a semi-implicit DAE (3.24) satisfying the conditions of Lemma 3.5.1. Therefore, the assertion follows from Lemma 3.5.1. □

Remark 3.5.4 Quasi-linear DAEs (3.23) satisfying Lemma 3.5.3 have d-index 1 if $r_E + r_C = m = n$ and $r_E < m$ and they have d-index 0 if $r_E = m = n$. In the latter case, the quasi-linear DAE (3.23) corresponds to an ODE in implicit form. □

Remark 3.5.5 A necessary but not sufficient condition for the semi-implicit DAE (3.24) to be strangeness-free is that the matrices $E_1(x,u)$ and $k_{2,x}(x,u)$ have constant rank for all $(x,u) \in \mathbb{M}$. Note that it is not necessary for the matrices to have constant rank outside of $\mathbb{M}$. Consider the following example. □

Example 3.5.6 An example for a semi-implicit DAE is

$$\begin{aligned} \begin{bmatrix} x_2 & -x_1 \end{bmatrix} \dot{x} &= k_1(x), \\ 0 &= \begin{bmatrix} x_1^2 + x_2^2 - 1 \end{bmatrix}. \end{aligned}$$

Since

$$\begin{bmatrix} E_1(x,u) \\ k_{2,x}(x,u) \end{bmatrix} = \begin{bmatrix} x_2 & -x_1 \\ 2x_1 & 2x_2 \end{bmatrix} \tag{3.33}$$

satisfies condition (3.27) for all $x \in \mathbb{M} = \{x = (x_1, x_2) \ : \ 0 = x_1^2 + x_2^2 - 1\}$, this DAE is actually strangeness-free although the matrix (3.33) is singular for $x = 0$, since $x = 0 \notin \mathbb{M}$. In particular, this system is actually of d-index one since the matrix (3.33) is nonsingular for all $x \in \mathbb{M}$. □

We have seen in Examples 3.2.14 and 3.5.6 that the solution x of a DAE (3.2a) is restricted to a subset $\mathbb{M}$ by certain constraints contained in the DAE. This manifold is called the *solution manifold*. For a strangeness-free semi-implicit DAE (3.24) the solution manifold $\mathbb{M}$ is defined by the constraints $k_2(x,u)=0$, see (3.28).
In general, the solution manifold for a nonlinear DAE (3.2a) is defined by all constraints, i.e., constraints occurring explicitly in the DAE and additional constraints which are contained but hidden in the DAE, i.e., they are not stated as equations, as it is in Examples 3.2.14 and 3.5.6, see Definition 3.5.32 below for details. In particular, DAEs of higher index may contain constraints which are not explicitly given in the DAE as shown in the following example.

Example 3.5.7 Consider a semi-explicit DAE of the form

$$\begin{aligned} \dot{x}_1 &= k_1(x_1,x_2,u), && \text{(3.34a)}\\ 0 &= k_2(x_1,u) && \text{(3.34b)} \end{aligned}$$

with $x_1 \in \mathcal{C}^1(\mathbb{I},\mathbb{R}^{n_1})$, $x_2 \in \mathcal{C}(\mathbb{I},\mathbb{R}^{n_2})$, $k_1 \in \mathcal{C}(\mathbb{X}_1\times\mathbb{X}_2\times\mathbb{U},\mathbb{R}^{n_1})$, $k_2 \in \mathcal{C}^1(\mathbb{X}_1\times\mathbb{U},\mathbb{R}^{n_2})$, $\mathbb{X}_1 \subset \mathbb{R}^{n_1}$, and $\mathbb{X}_2 \subset \mathbb{R}^{n_2}$. The solution has to satisfy the constraints (3.34b) which are explicitly stated in the DAE and in addition the solution has to satisfy the constraints

$$0 = k_{2,x_1}(x_1,u)k_1(x_1,x_2,u) + k_{2,u}(x_1,u)\dot{u} = \frac{\mathrm{d}}{\mathrm{d}t}k_2(x_1,u), \qquad \text{(3.34c)}$$

which arise from the first derivative of (3.34b) with respect to t and are not explicitly stated in the DAE, but implicitly. Therefore, the solution has to lie in the manifold

$$\begin{aligned} \mathbb{M} = \{(x_1,x_2,\mathfrak{u}^1) \in \mathbb{X}_1\times\mathbb{X}_2\times\mathbb{U}^1 \;:\; & 0 = k_2(x_1,u),\\ & 0 = k_{2,x_1}(x_1,u)k_1(x_1,x_2,u) + k_{2,u}(x_1,u)\dot{u}\} \end{aligned}$$

defined by both kinds of constraints.
In the case that $k_{2,x_1}(x_1,u)k_{1,x_2}(x_1,x_2,u)$ has a bounded inverse for all $(x_1,x_2,\mathfrak{u}^1) \in \mathbb{M}$, the DAE has d-index $\nu_d=2$, p-index $\nu_p=2$, s-index $\nu_s=1$, and t-index $\nu_t=2$. Furthermore, if $n_1=n_2$ then the solution is completely determined by the algebraic constraints (3.34b) and (3.34c) such that actually the DAE (3.34) corresponds to an algebraic system. □

3.5.2 Analysis of quasi-linear differential-algebraic equations

In the following, we will investigate quasi-linear DAEs with respect to their analytical properties, like constraints and their solution manifold. We will develop a tool for the analysis of quasi-linear DAEs in form of an iterative procedure. In preparation we need the following two lemmata.

Lemma 3.5.8 *Suppose that (3.23) is a solvable system of quasi-linear differential-algebraic equations and let $\mathbb{X}\subset\mathbb{R}^n$ be such that $x(t)\in\mathbb{X}$ for all $t\in\mathbb{I}$ and for all solutions x. Then, the solution x of (3.23) is invariant under nonsingular transformations, i.e., (3.23) has in $\mathbb{X}\times\mathbb{U}$ the same solution set as*

$$Z(x,u)E(x,u)\dot{x} = Z(x,u)k(x,u), \qquad \text{(3.35)}$$

where $Z(x,u)$ is nonsingular for all $(x,u)\in\mathbb{X}\times\mathbb{U}$.

Proof: Every solution of (3.23) is also a solution of (3.35). On the other hand, $x(t)$ is a solution of (3.35) if $Z(x,u)(E(x,u)\dot{x}-k(x,u))=0$. Because of the nonsingularity of $Z(x,u)$ for all $(x,u)\in\mathbb{X}\times\mathbb{U}$ this is satisfied if and only if $E(x,u)\dot{x}-k(x,u)=0$ and we get the assertion. □

Remark 3.5.9 Note, that if the transformation matrix $Z(x,u)$ is singular, then the solution set of (3.35) may be larger than the solution set of (3.23). □

Lemma 3.5.10 *Suppose that (3.24) is a solvable system of semi-implicit differential-algebraic equations and let $\mathbb{X} \subset \mathbb{R}^n$ be such that $x(t) \in \mathbb{X}$ for all $t \in \mathbb{I}$ and for all solutions x. Furthermore, let the initial values $x(t_0) = x_0$ be consistent. Then, the solution of (3.24) is invariant under differentiation of the constraints, i.e., a solution of the initial value problem (3.24) with the initial values $x(t_0) = x_0$ is also a solution of*

$$\begin{bmatrix} E_1(x,u) \\ k_{2,x}(x,u) \end{bmatrix} \dot{x} = \begin{bmatrix} k_1(x,u) \\ -k_{2,u}(x,u)\dot{u} \end{bmatrix} \tag{3.36}$$

with the initial values $x(t_0) = x_0$, and vice versa.

Proof: If x is a solution of (3.24) with initial values $x(t_0) = x_0$ it is also a solution of (3.36) with initial values $x(t_0) = x_0$. On the other hand, setting $\kappa(t) = k_2(x(t), u(t))$ we get the constraints in the form $\kappa(t) = 0$ and its first derivative with respect to t as

$$\dot{\kappa}(t) = 0. \tag{3.37}$$

The solution of (3.37) is given by $\kappa(t) = \kappa_0$ with $\kappa_0 = k_2(x_0, u(t_0))$. From the consistency of the initial values we get $\kappa_0 = 0$ and, therefore, every $x(t)$ which satisfies the first derivative (3.37) with respect to t of the constraints (3.24b) with $x(t_0) = x_0$ also satisfies the condition $\kappa(t) = 0$ which corresponds to the constraints of (3.24) such that we get the assertion. □

In the following we will present a procedure for the analysis of quasi-linear DAEs of the form (3.23), with $x \in \mathbb{X}$. For the control variables $u \in \mathbb{U}$ we will use the Notation 3.1.4.

Procedure 3.5.11 Consider the quasi-linear DAE (3.23). Assume that the leading matrix $E(x,u)$ is continuous with respect to the first component, i.e., $E \in \mathcal{C}^{0,\cdot}(\mathbb{X} \times \mathbb{U}, \mathbb{R}^{m,n})$.

Initialization: Set $i = 0$, $E^0(x,u) = E(x,u)$, $k^0(x,u) = k(x,u)$, and $\mathbb{M}_{-1} = \mathbb{X}$.

Iteration step:

0) Start the iteration step with

$$E^i(x, \mathfrak{u}^{i-1})\dot{x} = k^i(x, \mathfrak{u}^i). \tag{3.38}$$

Note that in the case of $i = 0$, the DAE (3.38) is identical with the DAE (3.23) and note the convention that $\mathfrak{u}^{-1} = \mathfrak{u}^0 = u$.

I) Suppose, there exists a matrix function $Z^i(x, \mathfrak{u}^{i-1})$ which is nonsingular for all $(x, \mathfrak{u}^{i-1}) \in \mathbb{M}_{i-1}$ and continuous with respect to its first component, i.e.,

$Z^i \in \mathcal{C}^{0,\cdot}(\mathbb{M}_{i-1}, \mathbb{R}^{m,m})$, such that the following criteria are satisfied.

$$1)\ \ Z^i(x,\mathfrak{u}^{i-1})E^i(x,\mathfrak{u}^{i-1}) = \begin{bmatrix} \widetilde{E}_1^i(x,\mathfrak{u}^{i-1}) \\ 0 \end{bmatrix} \tag{3.39a}$$

$$2)\ \ Z^i(x,\mathfrak{u}^{i-1})k^i(x,\mathfrak{u}^i) = \begin{bmatrix} \widetilde{k}_1^i(x,\mathfrak{u}^i) \\ \widetilde{k}_2^i(x,\mathfrak{u}^i) \end{bmatrix} \tag{3.39b}$$

$$3)\ \ \widetilde{E}_1^i \text{ is of size } m_1^i \times n \text{ with } m_1^i = \max_{(x,\mathfrak{u}^i)\in\mathbb{M}_i} (\operatorname{rank}(E^i(x,\mathfrak{u}^{i-1}))) \tag{3.39c}$$

$$4)\ \ \widetilde{k}_j^i \text{ are of sizes } m_j^i,\ j=1,2, \quad \text{with } m_1^i + m_2^i = m \tag{3.39d}$$

$$5)\ \ \operatorname{rank}(\widetilde{E}_1^i(x,\mathfrak{u}^{i-1})) = \operatorname{rank}(E^i(x,\mathfrak{u}^{i-1})) \text{ for all } (x,\mathfrak{u}^i) \in \mathbb{M}_i \tag{3.39e}$$

$$6)\ \ \widetilde{\mathbb{M}}_i = \{(x,\mathfrak{u}^i) \in \mathbb{X}\times\mathbb{U}^i \ :\ 0 = \widetilde{k}_2^i(x,\mathfrak{u}^i)\} \tag{3.39f}$$

$$7)\ \ \mathbb{M}_i = (\mathbb{M}_{i-1}\times\mathbb{U}^{(i)}) \cap \widetilde{\mathbb{M}}_i \subset \mathbb{X}\times\mathbb{U}^i \tag{3.39g}$$

II) Multiply both sides of the quasi-linear DAE (3.38) with Z^i from the left to obtain the intermediate DAE

$$\begin{bmatrix} \widetilde{E}_1^i(x,\mathfrak{u}^{i-1}) \\ 0 \end{bmatrix} \dot{x} = \begin{bmatrix} \widetilde{k}_1^i(x,\mathfrak{u}^i) \\ \widetilde{k}_2^i(x,\mathfrak{u}^i) \end{bmatrix}. \tag{3.40}$$

In this semi-implicit form the DAE is partitioned into the differential part $\widetilde{E}_1^i(x,\mathfrak{u}^{i-1})\dot{x} = \widetilde{k}_1^i(x,\mathfrak{u}^i)$ and the constraints

$$0 = \widetilde{k}_2^i(x,\mathfrak{u}^i). \tag{3.41}$$

Let us call these *constraints of level i*. Together with the constraints up to level $i-1$ they restrict the solution x into the *constraint set of level i* $\mathbb{M}_i$.

III) If

$$m_2^i = 0 \quad \text{or} \quad \widetilde{k}_2^i(x,\mathfrak{u}^i) = 0 \tag{3.42}$$

for all $(x,\mathfrak{u}^i) \in \mathbb{M}_{i-1}\times\mathbb{U}^{(i)}$, then the procedure terminates with $\nu = i$.

IV) If the right-hand side $\widetilde{k}_2^i(x,\mathfrak{u}^i)$ is contradictory or not continuously differentiable on $\mathbb{M}_i$, then the procedure terminates without result.

V) Replace the constraints (3.41) by its derivative with respect to t and transfer the DAE (3.40) to the form

$$\begin{bmatrix} \widetilde{E}^i(x,\mathfrak{u}^{i-1}) \\ \widetilde{k}_{2,x}^i(x,\mathfrak{u}^i) \end{bmatrix} \dot{x} = \begin{bmatrix} \widetilde{k}_1^i(x,\mathfrak{u}^i) \\ -\widetilde{k}_{2,\mathfrak{u}^i}^i(x,\mathfrak{u}^i)\dot{\mathfrak{u}}^i \end{bmatrix}. \tag{3.43}$$

VI) The DAE (3.43) is a quasi-linear DAE in the form (3.38) for $i+1$ instead of i. Set

$$E^{i+1}(x,\mathfrak{u}^i) = \begin{bmatrix} \widetilde{E}_1^i(x,\mathfrak{u}^{i-1}) \\ \widetilde{k}_{2,x}^i(x,\mathfrak{u}^i) \end{bmatrix}$$

and

$$k^{i+1}(x,\mathfrak{u}^{i+1}) = \begin{bmatrix} \widetilde{k}_1^i(x,\mathfrak{u}^i) \\ -\widetilde{k}_{2,\mathfrak{u}^i}^i(x,\mathfrak{u}^i)\dot{\mathfrak{u}}^i \end{bmatrix},$$

increase i by one, and proceed with step *I*).

□

Remark 3.5.12 a) If the rank of the leading matrix in (3.38) is constant, i.e., $\operatorname{rank}(E^i(x,\mathfrak{u}^{i-1})) = r_i$ for all $(x,\mathfrak{u}^{i-1}) \in \mathbb{M}_{i-1}$ and $\mathbb{M}_{i-1} \subset \mathbb{X} \times \mathbb{U}^{i-1}$ is $\mathcal{C}^1$-diffeomorphic to a set $\mathbb{Y}$ defined in (2.4) then the existence of the transformation matrix Z^i follows from Theorem 2.1.6 and it can be computed with Algorithm 2.1.9.
b) If the rank of the leading matrix in (3.38) is not constant, it is often possible to obtain such a transformation matrix by the following process.

1. Set $\mathbb{M}_i = \mathbb{M}_{i-1} \times \mathbb{U}^{(i)}$.
2. Compute $m_1^i = \max\limits_{(x,\mathfrak{u}^i)\in\mathbb{M}_i} (\operatorname{rank}(E^i(x,\mathfrak{u}^{i-1})))$ and $m_2^i = m - m_1^i$.
3. If possible, compute a nonsingular matrix function $Z^i \in \mathcal{C}^{0,\cdot}(\mathbb{M}_{i-1}, \mathbb{R}^{m,m})$, such that the following criteria are satisfied.

   $i)$ $\quad Z^i(x,\mathfrak{u}^{i-1})E^i(x,\mathfrak{u}^{i-1}) = \begin{bmatrix} \widetilde{E}_1^i(x,\mathfrak{u}^{i-1}) \\ 0 \end{bmatrix}$

   $\ddot{u})$ $\quad Z^i(x,\mathfrak{u}^{i-1})k^i(x,\mathfrak{u}^i) = \begin{bmatrix} \widetilde{k}_1^i(x,\mathfrak{u}^i) \\ \widetilde{k}_2^i(x,\mathfrak{u}^i) \end{bmatrix}$

   $\dddot{m})$ $\quad \widetilde{E}_1^i$ is of size $m_1^i \times n$, $\widetilde{k}_j^i$ are of sizes m_j^i, $j = 1,2$

   $\dot{v})$ $\quad \operatorname{rank}(\widetilde{E}_1^i(x,\mathfrak{u}^{i-1})) = \operatorname{rank}(E^i(x,\mathfrak{u}^{i-1}))$ for all $(x,\mathfrak{u}^i) \in \mathbb{M}_i$
4. Set $\widetilde{\mathbb{M}}_i = \{(x,\mathfrak{u}^i) \in \mathbb{X} \times \mathbb{U}^i \ : \ 0 = \widetilde{k}_2^i(x,\mathfrak{u}^i)\}$.
5. Set $\mathbb{M}_i = (\mathbb{M}_{i-1} \times \mathbb{U}^{(i)}) \cap \widetilde{\mathbb{M}}_i \subset \mathbb{X} \times \mathbb{U}^i$.
6. If $\max\limits_{(x,\mathfrak{u}^i)\in\mathbb{M}_i} (\operatorname{rank}(E^i(x,\mathfrak{u}^{i-1}))) < m_1^i$ then proceed with Step 2. Otherwise, the process terminates with Z^i computed in Step 3.

c) If in any iteration step i of Procedure 3.5.11 the constraints (3.41) contain some trivially satisfied equations, i.e., constraints which are trivially satisfied for all $(x,\mathfrak{u}^i) \in \mathbb{M}_{i-1} \times \mathbb{U}^{(i)}$, it is not necessary to consider these trivially satisfied equations in the further iteration process of the procedure, rather, they are kept unchanged in the constraints (3.41) from one iteration step i to the next iteration step $i+1$. □

Remark 3.5.13 a) In the case that the matrix $E(x,u)$ is nonsingular for all $(x,u) \in \mathbb{X} \times \mathbb{U}$, the quasi-linear DAE (3.23) corresponds to an implicit ODE and Procedure 3.5.11 terminates with $\nu = 0$.
b) The termination criterion $\widetilde{k}_2^i(x,\mathfrak{u}^i) = 0$ in (3.42) means that the remaining constraints do not restrict the state x and therefore, also every subsequent derivative of $0 = \widetilde{k}_2^i(x,\mathfrak{u}^i)$ with respect to t does not restrict the state x.
c) It follows from Procedure 3.5.11 that some components of the right-hand side of the quasi-linear DAE (3.23) have to be ν-times continuously differentiable with respect to its first argument.
d) Note that we consider quasi-linear DAEs (3.4) with m not necessarily equal to n. Therefore, the constraints (3.41) may be redundant. □

Remark 3.5.14 The Procedure 3.5.11 can be seen as a generalization of the Silverman structure algorithm introduced in [162] which is designed for the inversion of linear control problems with constant coefficients as well as a generalization of the modification of the Silverman structure algorithm for a certain class of nonlinear control problems introduced in [168]. □

Example 3.5.15 Consider again the semi-explicit DAE (3.34), i.e.,

$$\begin{aligned} \dot{x}_1 &= k_1(x_1, x_2, u), \\ 0 &= k_2(x_1, u) \end{aligned}$$

with $x_1 \in \mathcal{C}^1(\mathbb{I}, \mathbb{R}^{n_1})$, $x_2 \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{n_2})$, $k_1 \in \mathcal{C}(\mathbb{X}_1 \times \mathbb{X}_2 \times \mathbb{U}, \mathbb{R}^{n_1})$, $k_2 \in \mathcal{C}^1(\mathbb{X}_1 \times \mathbb{U}, \mathbb{R}^{n_2})$, $\mathbb{X}_1 \subset \mathbb{R}^{n_1}$, and $\mathbb{X}_2 \subset \mathbb{R}^{n_2}$. Following Procedure 3.5.11 we have

$$E^0 = \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix}, \; k^0 = \begin{bmatrix} k_1(x_1, x_2, u) \\ k_2(x_1, u) \end{bmatrix}.$$

With the transformation matrix $Z^0 = I$ we get $\widetilde{E}_1^0 = \begin{bmatrix} I & 0 \end{bmatrix}$, $\widetilde{k}_1^0 = k_1$, and $\widetilde{k}_2^0 = k_2$. The constraints of level 0 are given by $0 = \widetilde{k}_2^0 = k_2$ and therefore, we get $\mathbb{M}_0 = \widetilde{\mathbb{M}}_0 = \{(x, u) \in \mathbb{X} \times \mathbb{U} \; : \; 0 = k_2\}$. Differentiation of the constraints of level 0 with respect to t yields $0 = k_{2,x_1}(x_1, u)\dot{x}_1 + k_{2,u}(x_1, u)\dot{u}$ and we get the transformed DAE $E^1(x, u)\dot{x} = k^1(x, \mathfrak{u}^1)$ with

$$E^1(x, u) = \begin{bmatrix} I & 0 \\ k_{2,x_1}(x_1, u) & 0 \end{bmatrix}, \; k^1(x, \mathfrak{u}^1) = \begin{bmatrix} k_1(x_1, x_2, u) \\ -k_{2,u}(x_1, u)\dot{u} \end{bmatrix}.$$

The transformation with the nonsingular transformation matrix

$$Z^1(x, u) = \begin{bmatrix} I & 0 \\ -k_{2,x}(x_1, u) & I \end{bmatrix}$$

leads to

$$\begin{aligned} \widetilde{E}_1^1(x, u) &= \begin{bmatrix} I & 0 \end{bmatrix}, \\ \widetilde{k}_1^1(x, \mathfrak{u}^1) &= \begin{bmatrix} k_1(x_1, x_2, u) \end{bmatrix}, \\ \widetilde{k}_2^1(x, \mathfrak{u}^1) &= \begin{bmatrix} -k_{2,x}(x_1, u)k_1(x_1, x_2, u) - k_{2,u}(x_1, u)\dot{u} \end{bmatrix}. \end{aligned}$$

Now we have determined the constraints of level 1 by

$$0 = \widetilde{k}_2^1(x, \mathfrak{u}^1) = -k_{2,x_1}(x_1, u)k_1(x_1, x_2, u) - k_{2,u}(x_1, u)\dot{u}. \tag{3.44}$$

Furthermore, we get

$$\widetilde{\mathbb{M}}_1 = \{(x, \mathfrak{u}^1) \in \mathbb{X} \times \mathbb{U}^1 \; : \; 0 = -k_{2,x_1}k_1 - k_{2,u}\dot{u}\}$$

and the constraint set of level 1 by

$$\mathbb{M}_1 = \{(x, \mathfrak{u}^1) \in \mathbb{X} \times \mathbb{U}^1 \; : \; 0 = k_2, \; 0 = -k_{2,x_1}k_1 - k_{2,u}\dot{u}\}.$$

Since the constraints of level 1 (3.44) are not satisfied for all $(x, \mathfrak{u}^1) \in \mathbb{M}_0 \times \mathbb{U}^{(1)}$, we have to continue the procedure. Differentiation of the constraints (3.44) with respect to t yields

$$\begin{aligned} 0 = \; & -(k_{2,x_1}(x_1, u)k_1(x_1, x_2, u))_{,x_1}\dot{x}_1 - k_{2,x_1}(x_1, u)k_{1,x_2}(x_1, x_2, u)\dot{x}_2 \\ & -(k_{2,x_1}(x_1, u)k_1(x_1, x_2, u))_{,u}\dot{u} - (k_{2,u}(x_1, u)\dot{u})_{,x_1}\dot{x}_1 \\ & -(k_{2,u}(x_1, u)\dot{u})_{,u}\dot{u} - k_{2,u}(x_1, u)\ddot{u} \end{aligned}$$

and we get the transformed DAE $E^2(x, \mathfrak{u}^1)\dot{x} = k^2(x, \mathfrak{u}^2)$ with

$$\begin{aligned} E^2(x, \mathfrak{u}^1) &= \begin{bmatrix} I & 0 \\ (k_{2,x_1}k_1)_{,x_1} + (k_{2,u}\dot{u})_{,x_1} & k_{2,x_1}k_{1,x_2} \end{bmatrix}, \\ k^2(x, \mathfrak{u}^2) &= \begin{bmatrix} k_1 \\ -(k_{2,x_1}k_1)_{,u}\dot{u} - (k_{2,u}\dot{u})_{,u}\dot{u} - k_{2,u}\ddot{u} \end{bmatrix}. \end{aligned}$$

If $k_{2,x_1}k_{1,x_2}$ is nonsingular for all $(x_1, x_2, \mathfrak{u}^1) \in \mathbb{M}_1$, then $m_2^2 = 0$ and the procedure terminates with $\nu = 2$. □

Remark 3.5.16 Note that the procedure does not require that the rank of the leading matrix $\widetilde{E}_1^\nu(x, \mathfrak{u}^{\nu-1})$ in (3.40) of the termination step $i = \nu$ has to be constant, i.e., it is not necessary that

$$\operatorname{rank}(\widetilde{E}_1^\nu(x, \mathfrak{u}^{\nu-1})) = \text{const}, \quad \text{for all } (x, \mathfrak{u}^{\nu-1}) \in \mathbb{M}_{\nu-1}, \tag{3.45a}$$

or that the rank of the constraints of level i has to be constant, i.e.,

$$\operatorname{rank}(\widetilde{k}_{2,x}^i(x, \mathfrak{u}^i)) = \text{const}, \quad \text{for all } (x, \mathfrak{u}^i) \in \mathbb{M}_i, \tag{3.45b}$$

for $i = 0, ..., \nu - 1$, as shown in the following Example 3.5.17.
Of course, quasi-linear DAEs (3.23) that do not satisfy (3.45) are not in general suitable for numerical integration. But the numerical integration only gets into trouble if the solution (x, u) passes such singularities, in which the rank of the leading matrix in (3.40) of the termination step $i = \nu$ or the rank of the constraints of level i in (3.40) is not constant. As long as the solution, which is determined by the initial values, does not pass such singularities the numerical integration process will not get into trouble. Therefore, we will not exclude such DAEs from our investigations. Rather, during the numerical integration process, if the solution passes such singularities, the numerical algorithm should detect such singularities and react in an appropriate way, see Example 5.3.4 in Section 5.3. □

Example 3.5.17 Consider the differential-algebraic equation

$$(x(t) - 1)\dot{x}(t) = (x(t) - 1)x(t). \tag{3.46}$$

For this DAE the Procedure 3.5.11 terminates in the first step, i.e., $\nu = i = 0$, since there do not exist any constraints. Obviously, with $\nu = 0$ we get for $\widetilde{E}_1^\nu(x, u) = (x - 1)$ that

$$\operatorname{rank}(\widetilde{E}_1^\nu(x, \mathfrak{u}^{\nu-1})) = \begin{cases} 0, & \text{if } x = 1, \\ 1, & \text{else.} \end{cases}$$

Therefore, this DAE has a singularity at $x = 1$. The solution set is given by $\{x \in \mathcal{C}^1(\mathbb{I}, \mathbb{R}) \ : \ x(t) = 1 \text{ or } x(t) = ce^t \text{ with } c \in \mathbb{R}\}$. If the initial value $x(t_0)$ is already larger than one, i.e., $x(t_0) > 1$, the solution does not pass $x = 1$ inside the integration interval $\mathbb{I} = [t_0, t_f]$ with $t_f > t_0$. On the other hand, if the initial value is smaller than one, i.e., $x(t_0) < 1$, and the integration interval is small enough, the singularity again will not be met. If the state space $\mathbb{X}$ does include the singularity, but the particular solution determined by the initial values and the integration interval does not pass through this singularity then it would be bad to exclude this DAE in advance from the investigations.
Note, that the DAE (3.46) does not satisfy the Hypothesis 3.2.7 because the set of solutions $\mathbb{L}_i$ is not a manifold for any i. □

Proposition 3.5.18 *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu$ in (3.42). Then, $m_2^i \leq m_2^{i-1}$ for all $i = 1, ..., \nu$.*

Proof: The proof follows immediately from Procedure 3.5.11. □

Proposition 3.5.19 *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu$ in (3.42) and assume that the constraint sets $\mathbb{M}_i$ are manifolds, i.e., $\operatorname{rank}(\widetilde{k}_{2,x}^i(x, \mathfrak{u}^i)) = \text{const}$ for all $(x, \mathfrak{u}_i) \in \mathbb{M}_i$, then we have*

$$\dim(\mathbb{M}_i) < \dim(\mathbb{M}_{i-1} \times \mathbb{U}^{(i)})$$

for all $i = 0, ..., \nu - 1$.

Proof: The proof follows immediately from the construction of the constraint sets $\mathbb{M}_i$ in Procedure 3.5.11 and the fact that $m_2^i > 0$ for all $i \in \mathbb{N}_0$ with $i < \nu$. □

Note that this proposition cannot be stated for general constraint sets $\mathbb{M}_i$ (without being a manifold) because the term "dimension" is not defined for general constraint sets possibly containing singular points.
With this proposition it can be guaranteed that Procedure 3.5.11 terminates within a finite number of iteration steps.

Lemma 3.5.20 *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu$ in (3.42). Then the differential part in (3.40) and the constraints in (3.40), i.e., the hidden constraints (3.59), are determined by*

$$\widetilde{E}_1^i(x, \mathfrak{u}^{i-1})\dot{x} - \widetilde{k}_1^i(x, \mathfrak{u}^i) = \sum_{j=0}^{i} D_j^i \frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x} - k), \tag{3.47a}$$

$$-\widetilde{k}_2^i(x, \mathfrak{u}^i) = \sum_{j=0}^{i} A_j^i \frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x} - k), \tag{3.47b}$$

with $D_0^0 = Z_1^0$, $A_0^0 = Z_2^0$, and the recursion

$$D_0^i = Z_{11}^i D_0^{i-1} + Z_{12}^i \dot{A}_0^{i-1}, \tag{3.48a}$$

$$D_j^i = Z_{11}^i D_j^{i-1} + Z_{12}^i \dot{A}_j^{i-1} + Z_{12}^i A_{j-1}^{i-1}, \qquad j = 1, ..., i-1, \tag{3.48b}$$

$$D_i^i = Z_{12}^i A_{i-1}^{i-1}, \tag{3.48c}$$

$$A_0^i = Z_{21}^i D_0^{i-1} + Z_{22}^i \dot{A}_0^{i-1}, \tag{3.49a}$$

$$A_j^i = Z_{21}^i D_j^{i-1} + Z_{22}^i \dot{A}_j^{i-1} + Z_{22}^i A_{j-1}^{i-1}, \qquad j = 1, ..., i-1, \tag{3.49b}$$

$$A_i^i = Z_{22}^i A_{i-1}^{i-1}, \tag{3.49c}$$

where Z_k^0, $k \in \{1, 2\}$ are defined such that

$$Z^0 = \begin{bmatrix} Z_1^0 \\ Z_2^0 \end{bmatrix} \text{ is nonsingular} \tag{3.50a}$$

and satisfies (3.39) for $i = 0$ and Z_{kl}^i, $k, l \in \{1, 2\}$ are defined such that

$$Z^i = \begin{bmatrix} Z_{11}^i & Z_{12}^i \\ Z_{21}^i & Z_{22}^i \end{bmatrix} \text{ is nonsingular} \tag{3.50b}$$

and satisfy (3.39) for $i = 1, ..., \nu$.

Proof: The proof will be done inductively. Let us start with $i = 0$. From Procedure 3.5.11 we get

$$\widetilde{E}_1^0\dot{x} - \widetilde{k}_1^0 = Z_1^0(E\dot{x} - k) = \sum_{j=0}^{0} D_j^0 \frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x} - k)$$

which corresponds to (3.47a) for $i = 0$. Furthermore, for the algebraic constraints we have

$$-\widetilde{k}_2^0 = Z_2^0(E\dot{x} - k) = \sum_{j=0}^{0} A_j^0 \frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x} - k)$$

which corresponds to (3.47b) for $i = 0$. Let us assume that the relations (3.47a) and (3.47b) are satisfied for certain $i \geq 0$ and let us proceed one step of Procedure 3.5.11. Starting in substep *II*) we have

$$\begin{aligned} \widetilde{E}_1^i \dot{x} - \widetilde{k}_1^i &= \sum_{j=0}^{i} D_j^i \frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x} - k), \\ -\widetilde{k}_2^i &= \sum_{j=0}^{i} A_j^i \frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x} - k). \end{aligned}$$

Differentiation of the algebraic constraints with respect to t yields

$$\widetilde{E}_1^i \dot{x} - \widetilde{k}_1^i = \sum_{j=0}^{i} D_j^i \frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x} - k), \tag{3.51a}$$

$$\begin{aligned} -\widetilde{k}_{2,x}^i \dot{x} - \widetilde{k}_{2,\mathrm{u}^i}^i \dot{\mathrm{u}}^i &= \sum_{j=0}^{i} \dot{A}_j^i \frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x} - k) + \sum_{j=0}^{i} A_j^i \frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}}(E\dot{x} - k) \qquad (3.51b) \\ &= \dot{A}_0^i (E\dot{x} - k) + \sum_{j=0}^{i-1} (\dot{A}_{j+1}^i + A_j^i) \frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}}(E\dot{x} - k) \\ &\quad + A_i^i \frac{\mathrm{d}^{i+1}}{\mathrm{d}t^{i+1}}(E\dot{x} - k). \end{aligned}$$

Selecting the differential and the algebraic part by use of the transformation matrix

$$Z^{i+1} = \begin{bmatrix} Z_{11}^{i+1} & Z_{12}^{i+1} \\ Z_{21}^{i+1} & Z_{22}^{i+1} \end{bmatrix}$$

with

$$\begin{bmatrix} Z_{11}^{i+1} & Z_{12}^{i+1} \\ Z_{21}^{i+1} & Z_{22}^{i+1} \end{bmatrix} \begin{bmatrix} \widetilde{E}_1^i \\ -\widetilde{k}_{2,x}^i \end{bmatrix} = \begin{bmatrix} \widetilde{E}_1^{i+1} \\ 0 \end{bmatrix}$$

yields

$$0 = \begin{bmatrix} \widetilde{E}_1^{i+1} \dot{x} - \widetilde{k}_1^{i+1} \\ -\widetilde{k}_2^{i+1} \end{bmatrix} = \begin{bmatrix} Z_{11}^{i+1} & Z_{12}^{i+1} \\ Z_{21}^{i+1} & Z_{22}^{i+1} \end{bmatrix} \begin{bmatrix} \widetilde{E}_1^i \dot{x} - \widetilde{k}_1^1 \\ -\widetilde{k}_{2,x}^i \dot{x} - \widetilde{k}_{2,\mathrm{u}^i}^i \dot{\mathrm{u}}^i \end{bmatrix}.$$

With (3.51) we get

$$\begin{aligned} & \widetilde{E}_1^{i+1} \dot{x} - \widetilde{k}_1^{i+1} \\ = \; & Z_{11}^{i+1} \sum_{j=0}^{i} D_j^i \frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x} - k) \\ & + Z_{12}^{i+1} \Big(\dot{A}_0^i (E\dot{x} - k) + \sum_{j=0}^{i-1} (\dot{A}_{j+1}^i + A_j^i) \frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}}(E\dot{x} - k) + A_i^i \frac{\mathrm{d}^{i+1}}{\mathrm{d}t^{i+1}}(E\dot{x} - k) \Big) \\ = \; & (Z_{11}^{i+1} D_0^i + Z_{12}^{i+1} \dot{A}_0^i)(E\dot{x} - k) \\ & + \sum_{j=1}^{i} (Z_{11}^{i+1} D_j^i + Z_{12}^{i+1} \dot{A}_j^i + Z_{12}^{i+1} A_{j-1}^i) \frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x} - k) + Z_{12}^{i+1} A_i^i \frac{\mathrm{d}^{i+1}}{\mathrm{d}t^{i+1}}(E\dot{x} - k) \end{aligned}$$

and

$$\begin{aligned}
&-\widetilde{k}_2^{i+1}\\
&= Z_{21}^{i+1}\sum_{j=0}^{i} D_j^i \frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\\
&\quad + Z_{22}^{i+1}\Big(\dot{A}_0^i(E\dot{x}-k)+\sum_{j=0}^{i-1}(\dot{A}_{j+1}^i+A_j^i)\frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}}(E\dot{x}-k)+A_i^i\frac{\mathrm{d}^{i+1}}{\mathrm{d}t^{i+1}}(E\dot{x}-k)\Big)\\
&= (Z_{21}^{i+1}D_0^i+Z_{22}^{i+1}\dot{A}_0^i)(E\dot{x}-k)\\
&\quad + \sum_{j=1}^{i}(Z_{21}^{i+1}D_j^i+Z_{22}^{i+1}\dot{A}_j^i+Z_{22}^{i+1}A_{j-1}^i)\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)+Z_{22}^{i+1}A_i^i\frac{\mathrm{d}^{i+1}}{\mathrm{d}t^{i+1}}(E\dot{x}-k).
\end{aligned}$$

Hence, for the differential part we get the recursion

$$\begin{aligned}
\widetilde{E}_1^i\dot{x}-\widetilde{k}_1^i &= (Z_{11}^iD_0^{i-1}+Z_{12}^i\dot{A}_0^{i-1})(E\dot{x}-k)\\
&\quad +\sum_{j=1}^{i-1}(Z_{11}^iD_j^{i-1}+Z_{12}^i\dot{A}_j^{i-1}+Z_{12}^iA_{j-1}^{i-1})\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\\
&\quad +Z_{12}^iA_{i-1}^{i-1}\frac{\mathrm{d}^i}{\mathrm{d}t^i}(E\dot{x}-k)
\end{aligned}$$

which corresponds to (3.47a) and for the algebraic part we get the recursion

$$\begin{aligned}
-\widetilde{k}_2^i &= (Z_{21}^iD_0^{i-1}+Z_{22}^i\dot{A}_0^{i-1})(E\dot{x}-k)\\
&\quad +\sum_{j=1}^{i-1}(Z_{21}^iD_j^{i-1}+Z_{22}^i\dot{A}_j^{i-1}+Z_{22}^iA_{j-1}^{i-1})\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\\
&\quad +Z_{22}^iA_{i-1}^{i-1}\frac{\mathrm{d}^i}{\mathrm{d}t^i}(E\dot{x}-k)
\end{aligned}$$

which corresponds to (3.47b). □

Remark 3.5.21 Note that the matrix functions A_j^i and D_j^i formally depend on the variables x up to its ith derivative $x^{(i)}$ with respect to t and on the control variables and its derivatives, i.e., on $\mathfrak{u}^i$. In particular, we have $A_j^i = A_j^i(x,\dot{x},...,x^{(i)},\mathfrak{u}^i)$ and $D_j^i = D_j^i(x,\dot{x},...,x^{(i)},\mathfrak{u}^i)$. □

Lemma 3.5.22 *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i=\nu$ in (3.42). Then for all $i=0,...,\nu-1$ we have*

$$\sum_{j=0}^{i} A_j^i\left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}} = 0 \quad \text{for } l=1,...,\nu, \tag{3.52a}$$

$$\sum_{j=0}^{i} D_j^i\left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}} = \begin{cases} \widetilde{E}_1^i & \text{for } l=1,\\ 0 & \text{for } l=2,...,\nu, \end{cases} \tag{3.52b}$$

where D_j^i and A_j^i are defined in (3.48) and (3.49), respectively.

Proof: The proof proceeds by induction. For $i=0$ we get that (3.52a) simplifies to

$$A_0^0(E\dot{x}-k)_{,x^{(l)}} = 0 \quad \text{for } l=1,...,\nu,$$

with $A_0^0 = Z_2^0$ and (3.52b) simplifies to

$$D_0^0(E\dot{x}-k)_{,x^{(l)}} = \begin{cases} \widetilde{E}_1^i & \text{for } l=1, \\ 0 & \text{for } l=2,...,\nu, \end{cases}$$

with $D_0^0 = Z_1^0$. Since $E(x,u)$ and $k(x,u)$ do not depend on derivatives with respect to t of x these identities are satisfied. Furthermore, since also the hidden constraints do not depend on derivatives with respect to t of x by construction, we get

$$0 = \left(-\widetilde{k}_2^i(x,\mathfrak{u}^i)\right)_{,x^{(l)}}$$

for $(\mathfrak{x}^\nu, \mathfrak{u}^{\nu-1}) \in \mathbb{L}_{\nu-1}$. From (3.47b) it follows that

$$0 = \sum_{j=0}^{i} A_{j,x^{(l)}}^i [\![\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k), \cdot]\!] + \sum_{j=0}^{i} A_j^i \left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}}.$$

Therefore, from the induction hypothesis for (3.52a) it follows that

$$0 = \sum_{j=0}^{i} A_{j,x^{(l)}}^i [\![\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k), \cdot]\!]$$

for all $l = 1,...,\nu$. Furthermore, since $\widetilde{E}_1^i$ and $\widetilde{k}_1^i$ do not depend on derivatives with respect to t of x, by construction, see (3.40), we have

$$(\widetilde{E}_1^i\dot{x} - \widetilde{k}_1^i)_{,x^{(l)}} = \begin{cases} \widetilde{E}_1^i & \text{for } l=1, \\ 0 & \text{for } l=2,...,\nu. \end{cases} \tag{3.53}$$

With (3.47a) it follows on $\mathbb{L}_{\nu-1}$ that

$$\begin{aligned} \left(\widetilde{E}_1^i\dot{x} - \widetilde{k}_1^i\right)_{,x^{(l)}} &= \left(\sum_{j=0}^{i} D_j^i \frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}} \\ &= \sum_{j=0}^{i} D_{j,x^{(l)}}^i [\![\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k), \cdot]\!] + \sum_{j=0}^{i} D_j^i \left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}}. \end{aligned}$$

From (3.53) and the induction hypothesis for (3.52b) it follows that

$$0 = \sum_{j=0}^{i} D_{j,x^{(l)}}^i [\![\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k), \cdot]\!]$$

for all $l = 1,...,\nu-1$.
Let us assume that (3.52) is satisfied for i and consider now the next induction step, i.e., we show (3.52) for $i+1$. We have

$$\begin{aligned} \sum_{j=0}^{i+1} A_j^{i+1} \left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}} &= A_0^{i+1}(E\dot{x}-k)_{,x^{(l)}} + \sum_{j=1}^{i} A_j^{i+1} \left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}} \\ &\quad + A_{i+1}^{i+1} \left(\frac{\mathrm{d}^{i+1}}{\mathrm{d}t^{i+1}}(E\dot{x}-k)\right)_{,x^{(l)}}. \end{aligned}$$

From the recursion (3.49) with respect to A_j^i we get

$$\begin{aligned}
&\sum_{j=0}^{i+1} A_j^{i+1} \left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}} \\
&\quad = (Z_{21}^{i+1}D_0^i + Z_{22}^{i+1}\dot{A}_0^i)(E\dot{x}-k)_{,x^{(l)}} \\
&\qquad + \sum_{j=1}^{i}(Z_{21}^{i+1}D_j^i + Z_{22}^{i+1}\dot{A}_j^i + Z_{22}^{i+1}A_{j-1}^i)\left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}} \\
&\qquad + (Z_{22}^{i+1}A_i^i)\left(\frac{\mathrm{d}^{i+1}}{\mathrm{d}t^{i+1}}(E\dot{x}-k)\right)_{,x^{(l)}}
\end{aligned}$$

and it follows that

$$\begin{aligned}
&\sum_{j=0}^{i+1} A_j^{i+1} \left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}} \\
&\quad = Z_{21}^{i+1}D_0^i(E\dot{x}-k)_{,x^{(l)}} + Z_{22}^{i+1}\dot{A}_0^i(E\dot{x}-k)_{,x^{(l)}} \\
&\qquad + Z_{21}^{i+1}\sum_{j=1}^{i} D_j^i\left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}} + Z_{22}^{i+1}\sum_{j=1}^{i}\dot{A}_j^i\left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}} \\
&\qquad + Z_{22}^{i+1}\sum_{j=1}^{i} A_{j-1}^i\left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}} + Z_{22}^{i+1}A_i^i\left(\frac{\mathrm{d}^{i+1}}{\mathrm{d}t^{i+1}}(E\dot{x}-k)\right)_{,x^{(l)}} \\
&\quad = Z_{21}^{i+1}\sum_{j=0}^{i} D_j^i\left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}} \qquad (3.54) \\
&\qquad + Z_{22}^{i+1}\sum_{j=0}^{i}\left(\dot{A}_j^i\left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}} + A_j^i\left(\frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}}(E\dot{x}-k)\right)_{,x^{(l)}}\right).
\end{aligned}$$

For $\widetilde{k}_2^i(x,\mathfrak{u}^i)$ we have

$$\begin{aligned}
-\left(\dot{\widetilde{k}}_2^i(x,\mathfrak{u}^i)\right)_{,x^{(l)}} &= \left(-\widetilde{k}_{2,x}^i(x,\mathfrak{u}^i)\dot{x} - \widetilde{k}_{2,\mathfrak{u}^i}^i(x,\mathfrak{u}^i)\dot{\mathfrak{u}}^i\right)_{,x^{(l)}} \\
&= \begin{cases} -\widetilde{k}_{2,x}^i(x,\mathfrak{u}^i) & \text{for } l=1, \\ 0 & \text{for } l=2,...,\nu-1. \end{cases}
\end{aligned}$$

Furthermore, we have

$$\begin{aligned}
-\dot{\widetilde{k}}_2^i(x,\mathfrak{u}^i)_{,x^{(l)}} &= \left(\frac{\mathrm{d}}{\mathrm{d}t}(-\widetilde{k}_2^i(x,\mathfrak{u}^i))\right)_{,x^{(l)}} = \left(\frac{\mathrm{d}}{\mathrm{d}t}(\sum_{j=0}^{i} A_j^i\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k))\right)_{,x^{(l)}} \\
&= \sum_{j=0}^{i}\dot{A}_{j,x^{(l)}}^i[\![\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k),\cdot]\!] + \sum_{j=0}^{i}\dot{A}_j^i\left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}} \\
&\quad + \sum_{j=0}^{i} A_{j,x^{(l)}}^i[\![\frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}}(E\dot{x}-k),\cdot]\!] + \sum_{j=0}^{i} A_j^i\left(\frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}}(E\dot{x}-k)\right)_{,x^{(l)}}.
\end{aligned}$$

With $(\mathfrak{x}^\nu, \mathfrak{u}^{\nu-1}) \in \mathbb{L}_{\nu-1}$ we get

$$\begin{aligned}
&\sum_{j=0}^{i} \dot{A}_j^i \left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}} + \sum_{j=0}^{i} A_j^i \left(\frac{\mathrm{d}^{j+1}}{\mathrm{d}t^{j+1}}(E\dot{x}-k)\right)_{,x^{(l)}} \\
&\quad = -\dot{\widetilde{k}}_2^i(x,\mathfrak{u}^i)_{,x^{(l)}} \\
&\quad = \left(-\widetilde{k}_{2,x}^i(x,\mathfrak{u}^i)\dot{x} - \widetilde{k}_{2,\mathfrak{u}^i}^i(x,\mathfrak{u}^i)\dot{\mathfrak{u}}^i\right)_{,x^{(l)}} \\
&\quad = \begin{cases} -\widetilde{k}_{2,x}^i(x,\mathfrak{u}^i) & \text{for } l=1, \\ 0 & \text{for } l=2,...,\nu \end{cases}
\end{aligned}$$

and thus from (3.52b), (3.54), and (3.50b) it follows that

$$\begin{aligned}
&\sum_{j=0}^{i+1} A_j^{i+1} \left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}} \\
&\quad = \begin{cases} Z_{21}^{i+1}\widetilde{E}_1^i(x,\mathfrak{u}^i) + Z_{22}^{i+1}(-\widetilde{k}_{2,x}^i(x,\mathfrak{u}^i)) & \text{for } l=1, \\ 0 & \text{for } l=2,...,\nu \end{cases} \\
&\quad = 0
\end{aligned}$$

for $i = 1, ..., \nu-1$. Analogously, it follows that

$$\sum_{j=0}^{i+1} D_j^{i+1} \left(\frac{\mathrm{d}^j}{\mathrm{d}t^j}(E\dot{x}-k)\right)_{,x^{(l)}} = \begin{cases} \widetilde{E}_1^{i+1}(x,\mathfrak{u}^i) & \text{for } l=1, \\ 0 & \text{for } l=2,...,\nu \end{cases}$$

for $i = 0, ..., \nu-1$. □

Definition 3.5.23 (underlying (ordinary) differential equation) *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu$ in (3.42) and assume that* $\operatorname{rank}(\widetilde{E}_1^\nu(x,\mathfrak{u}^{\nu-1})) = m_1^\nu = \text{const}$ *for all* $(x,\mathfrak{u}^{\nu-1}) \in \mathbb{M}_{\nu-1}$. *Then, the obtained DAE*

$$\widetilde{E}_1^\nu(x,\mathfrak{u}^{\nu-1})\dot{x} = \widetilde{k}_1^\nu(x,\mathfrak{u}^\nu) \tag{3.55}$$

in (3.40) in step $i = \nu$ is called underlying differential equation *(UDE) of the quasi-linear DAE (3.23).*
If in addition the matrix $\widetilde{E}_1^\nu(x,\mathfrak{u}^{\nu-1})$ is nonsingular for all $(x,\mathfrak{u}^{\nu-1}) \in \mathbb{M}_{\nu-1}$ then the obtained DAE

$$\dot{x} = (\widetilde{E}_1^\nu(x,\mathfrak{u}^{\nu-1}))^{-1}\widetilde{k}_1^\nu(x,\mathfrak{u}^\nu)$$

in (3.40) in step $i = \nu$ is called underlying ordinary differential equation *(uODE) of the quasi-linear DAE (3.23).*

Lemma 3.5.24 *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu$ in (3.42). Then the set of solutions of the intermediately obtained DAEs $E^i(x,u)\dot{x} = k^i(x,u)$ for $i = 0, ..., \nu-1$ with initial values $x(t_0) = x_0$ equals the set of solutions of the original quasi-linear DAE (3.23) with initial values $x(t_0) = x_0$, if the initial values $x(t_0) = x_0$ are consistent to the original DAE (3.23), i.e., if they lie in the solution manifold $\mathbb{M}$.*

Proof: From Lemma 3.5.8 and from Lemma 3.5.10 it follows that the set of solution is unchanged in every step of the Procedure 3.5.11 if the initial values $x(t_0) = x_0$ are consistent to the original DAE (3.23), i.e., if they lie in the solution manifold $\mathbb{M}$. □

Proposition 3.5.25 *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu$ in (3.42). Then the set of solutions of the underlying differential equation (3.55) with initial values $x(t_0) = x_0$ equals the set of solutions of the original quasi-linear DAE (3.23) with initial values $x(t_0) = x_0$, if the initial values $x(t_0) = x_0$ are consistent to the original DAE (3.23), i.e., if they lie in the solution manifold $\mathbb{M}$.*

Proof: According to the Definition 3.5.23 the proof follows from Lemma 3.5.8 and Lemma 3.5.10 considering Procedure 3.5.11. □

Corollary 3.5.26 *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) with $m = n$ terminates in iteration step $i = \nu$ with $m_2^\nu = 0$ in (3.42) and* $\operatorname{rank}(\widetilde{E}_1^\nu(x, \mathfrak{u}^{\nu-1})) = \text{const}$ *for all $(x, \mathfrak{u}^{\nu-1}) \in \mathbb{M}_{\nu-1}$. Then, the DAE (3.40) corresponds to*

$$\widetilde{E}_1^\nu(x, \mathfrak{u}^{\nu-1})\dot{x} = \widetilde{k}_1^\nu(x, \mathfrak{u}^\nu) \tag{3.56}$$

with nonsingular leading matrix $\widetilde{E}_1^\nu(x, \mathfrak{u}^{\nu-1})$ for all $(x, \mathfrak{u}^{\nu-1}) \in \mathbb{M}_{\nu-1}$. Furthermore, (3.56) corresponds to the underlying ODE in implicit form.

Proof: The proof follows immediately from Procedure 3.5.11 in view of Definition 3.5.23. □

Definition 3.5.27 (Maximal constraint level) *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu$ in (3.42). Then the* maximal constraint level*, denoted by ν_c, is defined as $\nu_c = \nu - 1$.*

Note that $\nu_c \geq -1$. In particular, $\nu_c = -1$ means that there do not exist any constraints such that the quasi-linear DAE (3.23) corresponds to a set of differential equations. In the case of $m = n$ the quasi-linear DAE (3.23) corresponds to an ODE in implicit form.

Example 3.5.28 Consider again the semi-explicit DAE (3.34). If $k_{2,x_1} k_{1,x_2}$ is nonsingular for all $(x_1, x_2, \mathfrak{u}^1) \in \mathbb{M}_1$, then $m_2^2 = 0$ and the Procedure 3.5.11 terminates with $\nu = 2$ as shown in Example 3.5.15. Therefore, from Definition 3.5.27 we get that the DAE (3.34) has maximal constraint level $\nu_c = 1$. □

Lemma 3.5.29 *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) with $m = n$ terminates with $m_2^\nu = 0$ in iteration step $i = \nu = \nu_c + 1$ in (3.42) with maximal constraint level ν_c and* $\operatorname{rank}(\widetilde{E}_1^\nu(x, \mathfrak{u}^{\nu-1})) = \text{const}$ *for all $(x, \mathfrak{u}^{\nu-1}) \in \mathbb{M}_{\nu-1}$, then the d-index of the DAE (3.23) is well defined by $\nu_d = \nu_c + 1$.*

Proof: From Definition 3.2.3 we get the assertion with respect to the d-index in view of the Procedure 3.5.11 and Corollary 3.5.26 with the relation (3.47a) of Lemma 3.5.20. □

Lemma 3.5.30 *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu = \nu_c + 1$ in (3.42) with maximal constraint level ν_c, $m_2^\nu = 0$, and* $\operatorname{rank}(\widetilde{E}_1^\nu(x, \mathfrak{u}^{\nu-1})) = \text{const}$ *for all* $(x, \mathfrak{u}^{\nu-1}) \in \mathbb{M}_{\nu-1}$. *Furthermore, let* $\widetilde{k}_2^i(x, \mathfrak{u}^i)$, $i = 0, ..., \nu_c$ *with* $\operatorname{rank}(\widetilde{k}_2^i(x, \mathfrak{u}^i)) = \text{const}$ *for all* $(x, \mathfrak{u}^{\nu-1}) \in \mathbb{M}_{\nu-1}$ *be vector valued functions of size* m_2^i, *respectively, as defined in Procedure 3.5.11. Then,* $m \leq n$ *and*

$$\operatorname{rank}\left(\begin{bmatrix} E(x,u) \\ \widetilde{k}_{2,x}^0(x, \mathfrak{u}^0) \\ \vdots \\ \widetilde{k}_{2,x}^{\nu_c}(x, \mathfrak{u}^{\nu_c}) \end{bmatrix}\right) = m. \tag{3.57}$$

Proof: The proof follows immediately from Procedure 3.5.11. □

Lemma 3.5.31 *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu = \nu_c + 1$ in (3.42) with maximal constraint level ν_c, $m = n + m_2^\nu$, and* $\operatorname{rank}(\widetilde{E}_1^\nu(x, \mathfrak{u}^{\nu-1})) = \text{const}$ *for all* $(x, \mathfrak{u}^{\nu-1}) \in \mathbb{M}_{\nu-1}$. *Furthermore, let* $\widetilde{k}_2^i(x, \mathfrak{u}^i)$, $i = 0, ..., \nu_c$ *with* $\operatorname{rank}(\widetilde{k}_2^i(x, \mathfrak{u}^i)) = \text{const}$ *for all* $(x, \mathfrak{u}^{\nu-1}) \in \mathbb{M}_{\nu-1}$ *be vector valued functions of size* m_2^i, *respectively, as defined in Procedure 3.5.11. Then,*

$$\operatorname{rank}\left(\begin{bmatrix} E(x,u) \\ \widetilde{k}_{2,x}^0(x, \mathfrak{u}^0) \\ \vdots \\ \widetilde{k}_{2,x}^{\nu_c}(x, \mathfrak{u}^{\nu_c}) \end{bmatrix}\right) = n, \tag{3.58}$$

i.e., the matrix in (3.58) has full (column) rank.

Proof: The proof follows immediately from Procedure 3.5.11. □

With Procedure 3.5.11 we are able to define important quantities of the quasi-linear DAE (3.23), like the solution manifold, the hidden constraints, and the minimal reduced derivative array, which are of great interest for the development of numerical methods.

Definition 3.5.32 (Hidden constraints of quasi-linear DAEs) *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu = \nu_c + 1$ in (3.42) with maximal constraint level ν_c. The algebraic constraints (3.41)*

$$0 = \widetilde{k}_2^j(x, \mathfrak{u}^j), \ j = 1, ..., \nu_c \tag{3.59}$$

are called hidden constraints of level j of the quasi-linear DAE *(3.23).*

Remark 3.5.33 The maximal constraint level corresponds to the highest level of existing (hidden) constraints of the quasi-linear DAE (3.23) which equals the highest level of the derivative array which is necessary for the determination of all (hidden) constraints.
In general, the maximal constraint level ν_c does not coincide with the s-index ν_s, as shown in the following Examples 3.5.34 and 3.5.35. Rather the s-index is an upper bound for the maximal constraint level.

In particular, if the quasi-linear DAE (3.23) has s-index ν_s and the matrix function Z_2^T of size $a \times (\nu_s + 1)m$ in Hypothesis 3.2.7 has the form

$$Z_2^T = \left[\begin{array}{cccc} Z_{20}^T & \cdots & Z_{2\nu_s-1}^T & Z_{2\nu_s}^T \end{array} \right], \tag{3.60}$$

where Z_{2i}^T has the size $a \times m$ for all $i = 0, ..., \nu_s$ and $Z_{2\nu_s}^T \neq 0$, then all derivatives included in the derivative array of level ν_s, i.e., the derivatives of the DAE up to order ν_s, are necessary for the determination of the algebraic part containing the (hidden) constraints $\widetilde{k}_2^i$ in (3.41) which are extracted from the derivative array $\mathfrak{F}_{\nu_s}$ by use of the matrix function Z_2^T and contains all (hidden) constraints. Therefore, we have the maximal constraint level $\nu_c = \nu_s$. Otherwise, if the matrix function Z_2^T of size $a \times (\nu_s + 1)m$ in Hypothesis 3.2.7 has the form (3.60) with $Z_{2\nu_s}^T = 0$, then the derivative of the DAE of order ν_s is not used for the determination of the algebraic part and we have the maximal constraint level $\nu_c < \nu_s$. □

Example 3.5.34 Consider the DAE

$$\begin{aligned} \dot{x} &= 0, && \text{(3.61a)} \\ 0 &= x && \text{(3.61b)} \end{aligned}$$

which is already in semi-implicit form (3.40) with $\widetilde{E}_1^0 = \left[\begin{array}{c} 1 \end{array} \right]$ and the constraint

$$0 = \widetilde{k}_2^0 = x$$

which defines the manifolds $\widetilde{\mathbb{M}}_0 = \mathbb{M}_0 = \{x \in \mathbb{R} \ : \ x = 0\}$. Following Procedure 3.5.11 by differentiation of the constraint with respect to t we get the DAE $E^1\dot{x} = k^1$ with

$$E^1 = \left[\begin{array}{c} 1 \\ 1 \end{array} \right] \quad \text{and} \quad k^1 = \left[\begin{array}{c} 0 \\ 0 \end{array} \right]. \tag{3.62}$$

Transformation with the transformation matrix

$$Z^1 = \left[\begin{array}{cc} 1 & 0 \\ -1 & 1 \end{array} \right]$$

leads to the $\widetilde{E}_1^1 = \left[\begin{array}{c} 1 \end{array} \right]$ and the constraint

$$0 = \widetilde{k}_2^0 = 0$$

which is trivially satisfied. Therefore, Procedure 3.5.11 terminates with $\nu = 1$ and we get the maximal constraint level $\nu_c = \nu - 1 = 0$. Obviously, this corresponds to the highest level of the (hidden) constraints because there does not exist any hidden constraint but only the constraint $0 = \widetilde{k}_2^0$.
On the other hand, the DAE (3.61) is not strangeness-free but satisfies the Hypothesis 3.2.7 with $\nu = 1$, i.e., it has s-index $\nu_s = 1$, while the DAE $E^1\dot{x} = k^1$ with (3.62) obtained after one iteration step of Procedure 3.5.11 has s-index 0. □

Example 3.5.35 Consider the DAE

$$\left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{array} \right] \dot{x} = \left[\begin{array}{ccc} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{array} \right] x. \tag{3.63a}$$

The DAE is already in semi-implicit form (3.40) with $i = 0$ and

$$\widetilde{E}_1^0 = \left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \end{array} \right], \ \widetilde{k}_1^0 = \left[\begin{array}{ccc} 0 & 1 & 0 \\ 1 & 0 & 0 \end{array} \right] x, \ \widetilde{k}_2^0 = \left[\begin{array}{ccc} 1 & 0 & 0 \end{array} \right] x.$$

Therefore, we have the constraint $0 = \widetilde{k}_2^0 = x_1$ of level 0 and the manifolds $\widetilde{\mathbb{M}}_0 = \mathbb{M}_0 = \{x \in \mathbb{R}^3 \ : \ x_1 = 0\}$. By differentiation of the constraint we get the DAE (3.38) with $i = 1$ and

$$E^1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix}, \ k^1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} x. \tag{3.64}$$

Transformation with the transformation matrix

$$Z^1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

leads to the DAE (3.40) in semi-implicit form with $i = 1$ and

$$\widetilde{E}_1^1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \ \widetilde{k}_1^1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} x, \ \widetilde{k}_2^1 = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} x.$$

Therefore, we have the constraint $0 = \widetilde{k}_2^1 = x_2$ of level 1 and the manifolds $\widetilde{\mathbb{M}}_1 = \{x \in \mathbb{R}^3 \ : \ x_2 = 0\}$ and $\mathbb{M}_1 = \{x \in \mathbb{R}^3 \ : \ x_1 = 0, \ x_2 = 0\}$. Since the constraint of level 1 is not trivially satisfied for all $x \in \mathbb{M}_0$, we have to continue the procedure. By differentiation of the constraint of level 1 we get the DAE (3.38) with $i = 2$ and

$$E^2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \end{bmatrix}, \ k^2 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} x. \tag{3.65}$$

Transformation with

$$Z^2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

leads to the DAE (3.40) in semi-implicit form with $i = 2$ and

$$\widetilde{E}_1^2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \ \widetilde{k}_1^2 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} x, \ \widetilde{k}_2^2 = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x.$$

Therefore, we have the constraint $0 = \widetilde{k}_2^2 = x_1$ but this is trivially satisfied for all $x \in \mathbb{M}_1$. Therefore, Procedure 3.5.11 terminates with $\nu = 2$ and we get the maximal constraint level $\nu_c = \nu - 1 = 1$. Obviously, again this corresponds to the highest level of the (hidden) constraints.
On the other hand, the DAE (3.63) has s-index $\nu_s = 2$, while the DAE $E^1\dot{x} = k^1$ with (3.64) obtained after the first iteration step of Procedure 3.5.11 as well as the DAE $E^2\dot{x} = k^2$ with (3.65) obtained after the second iteration step of Procedure 3.5.11 have also s-index $\nu_s = 2$. In particular, in this example the s-index does not decrease from one iteration step to the next, in contrast to Examples 3.5.15 and 3.5.34. □

Remark 3.5.36 a) Hidden constraints impose consistency conditions on the initial values and provoke severe difficulties for the direct numerical integration of DAEs of higher index. For more details the reader is referred to [7, 25, 66, 69, 82, 133].
b) If the investigated quasi-linear DAE (3.23) does contain redundant constraints, in particular, the hidden constraints $0 = \widetilde{k}_2^i(x, \mathfrak{u}^i)$ for $i = (0,)1, ..., \nu_c$ are also redundant with respect to the state x. Therefore, they impose in addition to the restrictions on the state x also conditions on the control variables u and its derivatives with respect to t. □

Remark 3.5.37 Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu = \nu_c + 1$ in (3.42) with maximal constraint level ν_c. Furthermore, let the (hidden) constraints $0 = \widetilde{k}_2^i(x, \mathfrak{u}^i),\ i = 1, ..., \nu_c$ be defined in (3.41). Then, if the initial values $x(t_0) = x_0$ are consistent, they satisfy all (hidden) constraints, i.e.,

$$0 = \widetilde{k}_2^i(x_0, \mathfrak{u}^i(t_0))$$

for all $i = 0, ..., \nu_c$. □

Definition 3.5.38 (Solution manifold of quasi-linear DAEs) *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu = \nu_c + 1$ in (3.42) with maximal constraint level ν_c and* $\operatorname{rank}(\widetilde{k}_{2,x}^i(x, \mathfrak{u}^i)) =$ const *for all $(x, \mathfrak{u}^{\nu_c}) \in \mathbb{M}_{\nu_c}$ and $i = 0, ..., \nu_c$ in Procedure 3.5.11. The* solution manifold of the quasi-linear DAE *(3.23) is determined from all (hidden) constraints (3.59) of level $i = 0, ..., \nu_c$ and is defined by*

$$\mathbb{M} = \mathbb{M}_{\nu_c} = \{(x, \mathfrak{u}^{\nu_c}) \in \mathbb{X} \times \mathbb{U}^{\nu_c} \ :\ 0 = \widetilde{k}_2^0(x, \mathfrak{u}^0),\ ...,\ 0 = \widetilde{k}_2^{\nu_c}(x, \mathfrak{u}^{\nu_c})\}.$$

The derivative array $\mathfrak{F}_{\nu_c}$ of order ν_c defined in Definition 3.2.1 contains all necessary information about the solution manifold, constraints, and hidden constraints. But in general it contains more information than necessary. Unfortunately, the determination of the whole derivative array of order ν_c is often very technical and time consuming, in particular, if we consider a complex DAE of higher index. Procedure 3.5.11 offers the possibility to filter out the following *minimal reduced derivative array* which contains only the necessary information of the DAE.

Definition 3.5.39 (Minimal reduced derivative array) *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu = \nu_c + 1$ in (3.42) with maximal constraint level ν_c. Then the* minimal reduced derivative array of level $l \le \nu_c$ *of the quasi-linear DAE (3.23) is defined by*

$$0 = \widetilde{\mathfrak{F}}_l(x, \dot{x}, \mathfrak{u}^l) = \begin{bmatrix} E(x,u)\dot{x} - k(x,u) \\ \widetilde{k}_2^0(x,u) \\ \widetilde{k}_2^1(x, \mathfrak{u}^1) \\ \vdots \\ \widetilde{k}_2^l(x, \mathfrak{u}^l) \end{bmatrix}, \tag{3.66}$$

where $\widetilde{\mathfrak{F}}_l \in \mathcal{C}(\mathbb{X}^1 \times \mathbb{U}^l, \mathbb{R}^{m + \sum_{i=0}^{l} m_2^i})$, $\mathfrak{u}^i = [u^T, \dot{u}^T, ..., u^{(i)T}]^T$ and where $\widetilde{k}_2^i(x, \mathfrak{u}^i)$, $i = 0, ..., l$, are recursively defined as in Procedure 3.5.11.
In the case $l = \nu_c$ the minimal reduced derivative array $\widetilde{\mathfrak{F}}_{\nu_c}(x, \dot{x}, \mathfrak{u}^{\nu_c})$ is called complete minimal reduced derivative array *of the quasi-linear DAE (3.23).*

Remark 3.5.40 a) The complete minimal reduced derivative array $\widetilde{\mathfrak{F}}_{\nu_c}$ contains the original DAE together with all its hidden constraints. Note that in contrast to the derivative array (3.7) or a reduced derivative array (3.8) the minimal reduced derivative array (3.66) only depends on the control function u and its derivatives up to order ν_c and the unknown variables x and its first derivative $\dot{x}$ with respect to t, but not on higher derivatives $x^{(i)}$, $i > 1$ of the unknown variables x with respect to t.
b) If the investigated quasi-linear DAE (3.23) is in semi-implicit form (3.24) then it is not necessary to write down the explicitly given constraints twice in the upper

block equations in (3.66). Thus, the minimal reduced derivative array associated to a semi-explicit DAE (3.24) may have the form

$$0 = \widetilde{\mathfrak{F}}_l(x, \dot{x}, \mathfrak{u}^l) = \begin{bmatrix} E_1(x,u)\dot{x} - k_1(x,u) \\ \widetilde{k}_2^0(x,u) \\ \widetilde{k}_2^1(x,\mathfrak{u}^1) \\ \vdots \\ \widetilde{k}_2^l(x,\mathfrak{u}^l) \end{bmatrix}. \tag{3.67}$$

c) In particular, the minimal reduced derivative array (3.66) of level l corresponds to a semi-implicit DAE (3.24) of form

$$0 = \widetilde{E}(x,\mathfrak{u}^l)\dot{x} - \widetilde{k}(x,\mathfrak{u}^l) \tag{3.68a}$$

which has the particular form

$$\begin{bmatrix} \widetilde{E}_D(x,\mathfrak{u}^l) \\ 0 \end{bmatrix} \dot{x} = \begin{bmatrix} \widetilde{k}_D(x,\mathfrak{u}^l) \\ \widetilde{k}_C(x,\mathfrak{u}^l) \end{bmatrix} \tag{3.68b}$$

with

$$\widetilde{E}_D(x,\mathfrak{u}^l) = E(x,u), \qquad \widetilde{k}_D(x,\mathfrak{u}^l) = k(x,u), \tag{3.68c}$$

$$\widetilde{k}_C(x,\mathfrak{u}^l) = \begin{bmatrix} \widetilde{k}_2^0(x,u) \\ \widetilde{k}_2^1(x,\mathfrak{u}^1) \\ \vdots \\ \widetilde{k}_2^l(x,\mathfrak{u}^l) \end{bmatrix} \tag{3.68d}$$

and has the same set of solutions as the associated quasi-linear DAE (3.23). This fact is obtained from the Lemmata 3.5.8 and 3.5.10. If the quasi-linear DAE is in semi-implicit form (3.24) we may use $\widetilde{E}_D(x,\mathfrak{u}^l) = E_1(x,u)$ and $\widetilde{k}_D(x,\mathfrak{u}^l) = k_1(x,u)$ instead of the whole DAE in the first block equations. □

Example 3.5.41 Let us get back to Example 3.2.5. The considered DAE is

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & x_1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} x_1 x_2 \\ x_2 - 1 \\ x_3 - 1 \end{bmatrix}, \tag{3.69}$$

with $\mathbb{X} = \mathbb{R}^3$ and $t \in \mathbb{I}$. Note that $\operatorname{rank}(E(x,u))$ is not constant for $x \in \mathbb{R}^3$, since

$$\operatorname{rank}(E(x,u)) = \begin{cases} 2, & \text{for } x_1 \neq 0, \\ 1, & \text{for } x_1 = 0. \end{cases}$$

Nevertheless, there exists a nonsingular matrix $Z^0(x,u)$ such that the criteria (3.39) in step I) of Procedure 3.5.11 are satisfied. In our example we get $Z^0 = I$ and

$$\widetilde{E}_1^0(x,u) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & x_1 \end{bmatrix}, \qquad \widetilde{k}_1^0(x,u) = \begin{bmatrix} x_1 x_2 \\ x_2 - 1 \end{bmatrix},$$

$$\widetilde{k}_2^0(x,u) = \begin{bmatrix} x_3 - 1 \end{bmatrix},$$

with the constraint $0 = \widetilde{k}_2^0(x,u) = x_3 - 1$ which defines the constraint set of level 0

$$\mathbb{M}_0 = \widetilde{\mathbb{M}}_0 = \{(x,u) \in \mathbb{R}^3 \times \mathbb{U} \ : \ x_3 = 1\}$$

and

$$\max_{(x,u)\in\mathbb{M}_0}(\operatorname{rank}(\widetilde{E}_1^0(x,u))) = m_1^0 = 2.$$

Furthermore, we have that the constraint $0 = \widetilde{k}_2^0(x,u)$ is not satisfied automatically, i.e., $\widetilde{k}_2^0(x,u)$ does not vanish on $\mathbb{R}^3 \times \mathbb{U}$ and $\operatorname{rank}(\widetilde{k}_{2,x}^0(x,u)) \equiv m_2^0 = 1$ for all $(x,u) \in \mathbb{M}_0$. From the derivative of $\widetilde{k}_2^0(x,u)$ with respect to t we get the DAE $E^1\dot{x} = k^1$, see (3.38), with

$$E^1(x,u) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & x_1 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad k^1(x,\mathfrak{u}^1) = \begin{bmatrix} x_1x_2 \\ x_2 - 1 \\ 0 \end{bmatrix},$$

where E^1 has constant rank but not full rank for all x. With

$$Z^1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & -x_1 \end{bmatrix}$$

we get the intermediate DAE (3.40) with $i = 1$ and

$$\widetilde{E}_1^1(x,u) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad \widetilde{k}_1^1(x,\mathfrak{u}^1) = \begin{bmatrix} x_1x_2 \\ 0 \end{bmatrix},$$
$$\widetilde{k}_2^1(x,\mathfrak{u}^1) = \begin{bmatrix} x_2 - 1 \end{bmatrix},$$

with the constraint $0 = \widetilde{k}_2^1(x,u) = x_2 - 1$ which defines the subset

$$\widetilde{\mathbb{M}}_1 = \{(x,\mathfrak{u}^1) \in \mathbb{R}^3 \times \mathbb{U}^1 \ : \ x_2 = 1\}$$

and we get the constraint set of level 1

$$\mathbb{M}_1 = (\mathbb{M}_0 \times \mathbb{U}^{(1)}) \cap \widetilde{\mathbb{M}}_1 = \{(x,\mathfrak{u}^1) \in \mathbb{R}^3 \times \mathbb{U}^1 \ : \ x_2 = 1,\ x_3 = 1\}. \tag{3.70}$$

Furthermore, we have that the constraint $0 = \widetilde{k}_2^1(x,\mathfrak{u}^1)$ is not satisfied automatically on $\mathbb{M}_0$ and $\operatorname{rank}(\widetilde{k}_{2,x}^1(x,\mathfrak{u}^1)) \equiv m_2^1 = 1$ for all $(x,\mathfrak{u}^1) \in \mathbb{M}_1$. From the derivative with respect to t of $\widetilde{k}_2^1(x,\mathfrak{u}^1)$ we get the DAE $E^2\dot{x} = k^2$, see (3.38), with

$$E^2(x,\mathfrak{u}^1) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

which has constant rank and full rank for all $(x,\mathfrak{u}^1) \in \mathbb{M}_1$ and the constraints $0 = \widetilde{k}_2^2(x,\mathfrak{u}^2)$ are of size $m_2^2 = 0$. Therefore, the Procedure 3.5.11 terminates with $\nu = 2$, see (3.42). The maximal constraint level then is $\nu_c = \nu - 1 = 1$, the solution manifold is given by $\mathbb{M} = \mathbb{M}_1$ in (3.70), and the complete minimal reduced derivative array is given by

$$0 = \widetilde{\mathfrak{F}}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & x_1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} - \begin{bmatrix} x_1x_2 \\ x_2 - 1 \\ x_3 - 1 \\ x_3 - 1 \\ x_2 - 1 \end{bmatrix}.$$

Note that we used the complete minimal reduced derivative array in form (3.67) even if the original DAE (3.69) is in semi-implicit form. Therefore, the original

constraints of level 0 appear twice. In particular, note that $\operatorname{rank}(\widetilde{E}_1^{\nu}(x,\mathfrak{u}^{\nu-1})) = \text{const}$ for all $(x,\mathfrak{u}^{\nu-1}) \in \mathbb{M}_{\nu-1}$ with $\nu = \nu_c + 1 = 2$.
In Example 3.2.5 we have already found that the d-index of (3.69) is $\nu_d = 2$ and in Example 3.2.12 we have already found that the s-index of (3.69) is $\nu_s = 1$ which coincides with the maximal constraint level $\nu_c = 1$ for this example. □

Example 3.5.42 Consider again the semi-explicit DAE (3.34), i.e.,

$$\begin{aligned} \dot{x}_1 &= k_1(x_1, x_2, u), && \text{(3.71a)} \\ 0 &= k_2(x_1, u). && \text{(3.71b)} \end{aligned}$$

If $k_{2,x_1} k_{1,x_2}$ is nonsingular for all $(x_1, x_2, \mathfrak{u}^1) \in \mathbb{M}_1$, then $m_2^2 = 0$ and the Procedure 3.5.11 terminates with $\nu = 2$ as shown in Example 3.5.15. From Example 3.5.28 we have already the maximal constraint level $\nu_c = 1$. Furthermore, since the Procedure 3.5.11 terminates with $m_2^2 = 0$ and we have $m = n$ and $\operatorname{rank}(\widetilde{E}^2(x,\mathfrak{u}^1)) = \text{const}$ for all $(x,\mathfrak{u}^1) \in \mathbb{M}_1$, we get from Lemma 3.5.29 that the d-index of the DAE (3.71) is well defined and determined by $\nu_d = \nu_c + 1 = 2$. Furthermore, the solution manifold is given by

$$\begin{aligned} \mathbb{M} = \{(x_1, x_2, \mathfrak{u}^1) \in \mathbb{X}_1 \times \mathbb{X}_2 \times \mathbb{U}^1 \ : \ & 0 = k_2(x_1, u), && \text{(3.72)} \\ & 0 = -k_{2,x_1}(x_1, u) k_1(x_1, x_2, u) - k_{2,u}(x_1, u)\dot{u}\} \end{aligned}$$

and the complete minimal reduced derivative array is defined by

$$\widetilde{\mathfrak{F}}_1(x, \dot{x}, \mathfrak{u}^1) = \begin{bmatrix} \dot{x}_1 - k_1(x_1, x_2, u) \\ k_2(x_1, u) \\ k_2(x_1, u) \\ -k_{2,x_1}(x_1, u) k_1(x_1, x_2, u) - k_{2,u}(x_1, u)\dot{u} \end{bmatrix}. \qquad \text{(3.73)}$$

□

3.5.3 Regularization of quasi-linear differential-algebraic equations

As we have seen before, Procedure 3.5.11 provides a tool for the investigation of general quasi-linear DAEs (3.23) with respect to their analytical properties in an iterative way, based on transformations and differentiation of the necessary algebraic equations. On the other hand Hypothesis 3.2.7 provides a general tool for the investigation of general nonlinear DAEs (3.2a) based on transformations of the whole derivative array (3.7). In the following we will discuss a regularization technique for quasi-linear DAEs (3.23) based on Procedure 3.5.11. In preparation let us define the functions $r^l : \mathbb{M}_l \mapsto \mathbb{R}$, $a^l : \mathbb{M}_l \mapsto \mathbb{R}$, and $d^l : \mathbb{M}_l \mapsto \mathbb{R}$ by

$$\begin{aligned} r^l &= \operatorname{rank}\left(\begin{bmatrix} E(x,u) \\ \widetilde{k}_{2,x}^0(x,u) \\ \widetilde{k}_{2,x}^1(x,\mathfrak{u}^1) \\ \vdots \\ \widetilde{k}_{2,x}^l(x,\mathfrak{u}^l) \end{bmatrix} \right), && \text{(3.74a)} \\ a^l &= \operatorname{rank}\left(\begin{bmatrix} \widetilde{k}_{2,x}^0(x,u) \\ \widetilde{k}_{2,x}^1(x,\mathfrak{u}^1) \\ \vdots \\ \widetilde{k}_{2,x}^l(x,\mathfrak{u}^l) \end{bmatrix} \right), && \text{(3.74b)} \\ d^l &= r^l - a^l && \text{(3.74c)} \end{aligned}$$

for $l \le \nu_c$.

Definition 3.5.43 (Kinematic selector) *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu = \nu_c + 1$ in (3.42) with maximal constraint level ν_c. Let a^{ν_c} be defined by (3.74b) with $l = \nu_c$ and let $a^{\nu_c} = \text{const}$ for all $(x, \mathfrak{u}^{\nu_c}) \in \mathbb{M}$. Then a matrix function $S_C(x, \mathfrak{u}^{\nu_c})$ of size $a^{\nu_c} \times \sum_{i=0}^{\nu_c} m_2^i$ that satisfies*

$$\operatorname{rank}(S_C(x, \mathfrak{u}^{\nu_c}) \begin{bmatrix} \widetilde{k}_{2,x}^0(x, u) \\ \widetilde{k}_{2,x}^1(x, \mathfrak{u}^1) \\ \vdots \\ \widetilde{k}_{2,x}^{\nu_c}(x, \mathfrak{u}^{\nu_c}) \end{bmatrix}) = a^{\nu_c} \tag{3.75}$$

for all $(x, \mathfrak{u}^{\nu_c}) \in \mathbb{M}$ (see Definition 3.5.38), is called kinematic selector *of the quasi-linear DAE (3.23).*

Definition 3.5.44 (Dynamic selector) *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu = \nu_c + 1$ in (3.42) with maximal constraint level ν_c. Let r^{ν_c} and d^{ν_c} be defined by (3.74) with $l = \nu_c$ and let $r^{\nu_c} = \text{const}$ and $d^{\nu_c} = \text{const}$ for all $(x, \mathfrak{u}^{\nu_c}) \in \mathbb{M}$. Then a matrix function $S_D(x, \mathfrak{u}^{\nu_c})$ of size $d^{\nu_c} \times m$ that satisfies*

$$\operatorname{rank}(\begin{bmatrix} S_D(x, \mathfrak{u}^{\nu_c}) E(x, u) \\ \widetilde{k}_{2,x}^0(x, u) \\ \widetilde{k}_{2,x}^1(x, \mathfrak{u}^1) \\ \vdots \\ \widetilde{k}_{2,x}^{\nu_c}(x, \mathfrak{u}^{\nu_c}) \end{bmatrix}) = r^{\nu_c} \tag{3.76}$$

for all $(x, \mathfrak{u}^{\nu_c}) \in \mathbb{M}$ (see Definition 3.5.38), is called dynamic selector *of the quasi-linear DAE (3.23).*

Remark 3.5.45 In the investigation of semi-implicit DAEs (3.24) the definition of the dynamic selectors may be based on $E_1(x, u)$ instead of $E(x, u)$, see (3.76). □

Remark 3.5.46 a) The dynamic selector and the kinematic selector of a quasi-linear DAE are not uniquely defined.
b) Because of this nonuniqueness, it is possible to choose the selectors piecewise constant, see Lemma 2.1.3. This fact is of great advantage and importance for the numerical integration because it offers the possibility to reduce the amount of work for the regularization of the DAE. □

Lemma 3.5.47 *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu = \nu_c + 1$ in (3.42) with maximal constraint level ν_c. Let r^{ν_c} and d^{ν_c} be defined by (3.74) with $l = \nu_c$ and let $r^{\nu_c} = \text{const}$ and $d^{\nu_c} = \text{const}$ for all $(x, \mathfrak{u}^{\nu_c}) \in \mathbb{M}$. Then, the DAE*

$$0 = \begin{bmatrix} S_D(x, \mathfrak{u}^{\nu_c})(E(x, u)\dot{x} - k(x, u)) \\ \widetilde{k}_2^0(x, u) \\ \widetilde{k}_2^1(x, \mathfrak{u}^1) \\ \vdots \\ \widetilde{k}_2^{\nu_c}(x, \mathfrak{u}^{\nu_c}) \end{bmatrix}, \tag{3.77}$$

with a dynamic selector $S_D(x, \mathfrak{u}^{\nu_c})$ as defined in Definition 3.5.44 has at most maximal constraint level 0 and is strangeness-free.

Proof: With $\hat{u} = \mathfrak{u}^{\nu_c}$, the DAE (3.77) corresponds to a semi-implicit DAE (3.24) with $E_1(x,\hat{u}) = S_D(x,\mathfrak{u}^{\nu_c})E(x,u)$, $k_1(x,\hat{u}) = S_D(x,\mathfrak{u}^{\nu_c})k(x,u)$, and $k_2(x,\hat{u}) = \begin{bmatrix} -(\widetilde{k}_2^0(x,u))^T & \cdots & -(\widetilde{k}_2^{\nu_c}(x,\mathfrak{u}^{\nu_c}))^T \end{bmatrix}^T$ satisfying the assumptions of Lemma 3.5.1 arising from the construction of the dynamic selector S_D. Therefore, the assertion follows from Lemma 3.5.1. □

Proposition 3.5.48 *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu = \nu_c + 1$ in (3.42) with maximal constraint level ν_c. Let r^{ν_c}, a^{ν_c}, and d^{ν_c} be defined by (3.74) with $l = \nu_c$ and let $r^{\nu_c} = \text{const}$, $a^{\nu_c} = \text{const}$, and $d^{\nu_c} = \text{const}$ for all $(x,\mathfrak{u}^{\nu_c}) \in \mathbb{M}$. Then, the DAE*

$$0 = \begin{bmatrix} S_D(x,\mathfrak{u}^{\nu_c})(E(x,u)\dot{x} - k(x,u)) \\ S_C(x,\mathfrak{u}^{\nu_c}) \begin{bmatrix} \widetilde{k}_2^0(x,u) \\ \widetilde{k}_2^1(x,\mathfrak{u}^1) \\ \vdots \\ \widetilde{k}_2^{\nu_c}(x,\mathfrak{u}^{\nu_c}) \end{bmatrix} \end{bmatrix}, \tag{3.78}$$

with a dynamic selector $S_D(x,\mathfrak{u}^{\nu_c})$ as defined in Definition 3.5.44 and a kinematic selector $S_C(x,\mathfrak{u}^{\nu_c})$ as defined in Definition 3.5.43 has at most maximal constraint level 0 and is strangeness-free.

Proof: The proof follows analogous to the proof of Lemma 3.5.47. □

Remark 3.5.49 Since the s-index is an upper bound of the maximal constraint level, see Remark 3.5.33, it follows that the maximal constraint level of the DAEs (3.77) and (3.78) is at most 0. In the case that no constraints exist in the DAE (3.23), the maximal constraint level of (3.23) as well as of (3.77) and (3.78) is -1. □

Definition 3.5.50 (Projected-strangeness-free form of a quasi-linear DAE) *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu = \nu_c + 1$ in (3.42) with maximal constraint level ν_c. Let r^{ν_c}, a^{ν_c}, and d^{ν_c} be defined by (3.74) with $l = \nu_c$ and let $r^{\nu_c} = \text{const}$, $a^{\nu_c} = \text{const}$, and $d^{\nu_c} = \text{const}$ for all $(x,\mathfrak{u}^{\nu_c}) \in \mathbb{M}$. Then, a* projected-strangeness-free formulation *of the quasi-linear DAE (3.23) is defined by (3.77) with a dynamic selector $S_D(x,\mathfrak{u}^{\nu_c})$ as defined in Definition 3.5.44. Furthermore, a* projected-strangeness-free formulation with selected constraints *of the quasi-linear DAE (3.23) is defined by (3.78) with a dynamic selector $S_D(x,\mathfrak{u}^{\nu_c})$ as defined in Definition 3.5.44 and a kinematic selector $S_C(x,\mathfrak{u}^{\nu_c})$ as defined in Definition 3.5.43.*

Remark 3.5.51 In the case of the investigation of semi-implicit DAEs (3.24) the definition of the projected-strangeness-free form (3.77) may be based on the differential part $0 = E_1(x,u)\dot{x} - k_1(x,u)$ of the semi-implicit DAE instead of the whole DAE $0 = E(x,u)\dot{x} - k(x,u)$, see (3.77). Note that in this case the definition of the dynamic selector, see Definition 3.5.44, also has to be based on $E_1(x,u)$ instead of $E(x,u)$.
Furthermore, the same remark also applies to the projected-strangeness-free form with selected constraints (3.78) □

Theorem 3.5.52 *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu = \nu_c + 1$ in (3.42) with maximal constraint level ν_c. Let r^{ν_c} and d^{ν_c} be defined by (3.74) with $l = \nu_c$ and let $r^{\nu_c} = \text{const}$ and $d^{\nu_c} = \text{const}$ for all $(x, \mathfrak{u}^{\nu_c}) \in \mathbb{M}$. Then, every solution of the projected-strangeness-free formulation (3.77) associated to the quasi-linear DAE (3.23) with initial values $(x(t_0), \mathfrak{u}^{\nu_c}(t_0)) = (x_0, \mathfrak{u}_0^{\nu_c}) \in \mathbb{M}$ solves also the original quasi-linear DAE (3.23) with initial values $(x(t_0), \mathfrak{u}^{\nu_c}(t_0)) = (x_0, \mathfrak{u}_0^{\nu_c}) \in \mathbb{M}$.*

Proof: Consider the quasi-linear DAE (3.23) with respect to Procedure 3.5.11. Let us define in advance

$$K^i = \begin{bmatrix} \widetilde{k}_{2,x}^0 \\ \vdots \\ \widetilde{k}_{2,x}^{i-1} \end{bmatrix}, \ \widetilde{k}_{12}^i = \begin{bmatrix} \widetilde{k}_{2,u}^0 \dot{u} \\ \vdots \\ \widetilde{k}_{2,\mathfrak{u}^{i-1}}^{i-1} \dot{\mathfrak{u}}^{i-1} \end{bmatrix}, \ \widetilde{k}_2^{0\cdots i} = \begin{bmatrix} \widetilde{k}_2^0 \\ \vdots \\ \widetilde{k}_2^i \end{bmatrix}$$

for $i = 0, ..., \nu_c$ such that $0 = K^i \dot{x} + \widetilde{k}_{12}^i = \frac{\mathrm{d}}{\mathrm{d}t} \widetilde{k}_2^{0\cdots i}$ corresponds to the derivative with respect to t of the (hidden) constraints $\widetilde{k}_2^j$ for $j = 0, ..., i-1$. With $m_{12}^i = \sum_{j=0}^{i} m_2^j$, the matrix functions K^i are of size $m_{12}^{i-1} \times n$ and have rank $r_{12}^i = \operatorname{rank}(K^i)$. Furthermore, the vector functions $\widetilde{k}_{12}^i$ are of size m_{12}^{i-1} and the vector functions $\widetilde{k}_2^{0\cdots i}$ are of size m_{12}^i. Note that for $i = 0$ the matrix functions K^i and the vector functions $\widetilde{k}_{12}^i$ do not occur, or in particular, they are of size $0 \times n$ and 0, respectively, and $r_{12}^0 = 0$.
Procedure 3.5.11 implies the existence of a nonsingular transformation matrix

$$Z^0 = \begin{bmatrix} S^0 \\ Z_2^0 \end{bmatrix}$$

which transforms the DAE $E\dot{x} = k$ by scaling with Z^0 to the intermediate DAE (3.40) of partitioned form

$$\begin{aligned} S^0 E \dot{x} &= S^0 k, && \text{(3.79a)} \\ K^0 \dot{x} &= -\widetilde{k}_{12}^0, && \text{(3.79b)} \\ 0 &= \widetilde{k}_2^0 && \text{(3.79c)} \end{aligned}$$

with $S^0 E$ of size $m_{11}^0 \times n$ and $\operatorname{rank}(S^0 E) = m_{11}^0$ and

$$\operatorname{rank}\left(\begin{bmatrix} S^0 E \\ K^0 \end{bmatrix} \right) = m_1^0 = m_{11}^0 + r_{12}^0.$$

In view of Procedure 3.5.11 we have the intermediate DAE (3.40) with $i = 0$ and

$$E_1^0 = \begin{bmatrix} S^0 E \\ K^0 \end{bmatrix} \quad \text{and} \quad k_1^0 = \begin{bmatrix} S^0 k \\ -\widetilde{k}_{12}^0 \end{bmatrix}.$$

Note that we have formally introduced $K^0 \dot{x} = -\widetilde{k}_{12}^0$ even if this corresponds to a vanishing number of equations. Since the nonuniqueness of the transformation matrix functions Z^i in step I) of Procedure 3.5.11 lies in the block-column-wise scaling with nonsingular matrix functions, without loss of generality we can choose the nonsingular transformation matrix Z^{i+1} as

$$Z^{i+1} = \begin{bmatrix} S^{i+1} & 0 & 0 \\ 0 & I_{m_{12}^{i-1}} & 0 \\ 0 & 0 & I_{m_2^i} \\ Z_{41}^{i+1} & Z_{42}^{i+1} & Z_{43}^{i+1} \end{bmatrix} \tag{3.80}$$

satisfying

$$Z_{41}^{i+1} S^i \cdots S^0 E + Z_{42}^{i+1} K^i + Z_{43}^{i+1} \widetilde{k}_{2,x}^i = 0 \tag{3.81}$$

and

$$\operatorname{rank}\left(\begin{bmatrix} S^{i+1} S^i \cdots S^0 E \\ K^i \\ \widetilde{k}_{2,x}^i \end{bmatrix} \right) = m_{11}^{i+1} + \operatorname{rank}\left(\begin{bmatrix} K^i \\ \widetilde{k}_{2,x}^i \end{bmatrix} \right), \tag{3.82}$$

where the matrix on the left-hand side of (3.82) is of size $m_{11}^{i+1} + m_{12}^i \times n$ with $m_{11}^{i+1} = \operatorname{rank}(\begin{bmatrix} S^{i+1} S^i \cdots S^0 E \end{bmatrix})$, i.e., $S^{i+1} S^i \cdots S^0 E$ has full (column) rank. By use of nonsingular transformation matrix functions Z^i of the form (3.80) in every subsequent step of Procedure 3.5.11, the intermediate DAE (3.40) has the structure

$$\begin{aligned} S^i \cdots S^0 E \dot{x} &= S^i \cdots S^0 k, && \text{(3.83a)} \\ K^i \dot{x} &= -\widetilde{k}_{12}^i, && \text{(3.83b)} \\ 0 &= \widetilde{k}_2^i. && \text{(3.83c)} \end{aligned}$$

Let us keep in mind the following associated DAE

$$\begin{aligned} S^i \cdots S^0 E \dot{x} &= S^i \cdots S^0 k, && \text{(3.84a)} \\ 0 &= \widetilde{k}_2^{0 \cdots i-1}, && \text{(3.84b)} \\ 0 &= \widetilde{k}_2^i && \text{(3.84c)} \end{aligned}$$

which consists of all hidden constraints of level $0, ..., i$ and a selection of the original DAE. Let us consider the iteration step from i, i.e., (3.83), to $i+1 \leq \nu_c + 1$. From the derivative of the constraints (3.83c), using (3.83) we get the DAE $E^{i+1} \dot{x} = k^{i+1}$, see (3.43), by

$$\begin{aligned} S^i \cdots S^0 E \dot{x} &= S^i \cdots S^0 k, && \text{(3.85a)} \\ K^i \dot{x} &= -\widetilde{k}_{12}^i, && \text{(3.85b)} \\ \widetilde{k}_{2,x}^i \dot{x} &= -\widetilde{k}_{2,\mathrm{u}^i}^i \dot{\mathrm{u}}^i && \text{(3.85c)} \end{aligned}$$

By scaling with Z^{i+1} of the form (3.80), we get from (3.85) the intermediate DAE (3.40) for $i+1$ again in partitioned form

$$\begin{aligned} S^{i+1} S^i \cdots S^0 E \dot{x} &= S^{i+1} S^i \cdots S^0 k, && \text{(3.86a)} \\ K^i \dot{x} &= -\widetilde{k}_{12}^i, && \text{(3.86b)} \\ \widetilde{k}_{2,x}^i \dot{x} &= -\widetilde{k}_{2,\mathrm{u}^i}^i \dot{\mathrm{u}}^i, && \text{(3.86c)} \\ 0 &= \widetilde{k}_2^{i+1} && \text{(3.86d)} \end{aligned}$$

with

$$\widetilde{k}_2^{i+1} = Z_{41}^{i+1} S^i \cdots S^0 k - Z_{42}^{i+1} \widetilde{k}_{12}^i - Z_{43}^{i+1} \widetilde{k}_{2,\mathrm{u}^i}^i \dot{\mathrm{u}}^i \tag{3.87}$$

and its associated DAE

$$\begin{aligned} S^{i+1} \cdots S^0 E \dot{x} &= S^{i+1} \cdots S^0 k, && \text{(3.88a)} \\ 0 &= \widetilde{k}_2^{0 \cdots i-1}, && \text{(3.88b)} \\ 0 &= \widetilde{k}_2^i, && \text{(3.88c)} \\ 0 &= \widetilde{k}_2^{i+1}. && \text{(3.88d)} \end{aligned}$$

The relation between the DAE (3.86) and (3.88) is that the equations (3.86b) and (3.86c) are the first derivative with respect to t of (3.88b) and (3.88c), respectively. In particular, (3.88) forms the DAE explicitly consisting of all hidden constraints of level $0, ..., i+1$ and a selection of the original quasi-linear DAE (3.23).
In the following we will show that a solution of the associated DAE (3.88) solves also the associated DAE (3.84). Then, recursively it follows that a solution of the projected-strangeness-free formulation (3.77) of the quasi-linear DAE (3.23) also solves the DAE (3.79) and therefore, also solves the original quasi-linear DAE (3.23). Furthermore, the matrix $S^{\nu_c}S^{\nu_c-1}\cdots S^0$ satisfies the conditions of Definition 3.5.44. Therefore, the dynamic selector S_D can by chosen as $S_D = S^{\nu_c}S^{\nu_c-1}\cdots S^0$, except for scaling with a nonsingular matrix.
From the derivative of (3.88b) with respect to t we get $K^i\dot{x} = -\widetilde{k}^i_{12}$ and, therefore, it follows that

$$Z^{i+1}_{42}K^i\dot{x} = -Z^{i+1}_{42}\widetilde{k}^i_{12}. \tag{3.89}$$

Furthermore, from the derivative of (3.88c) with respect to t we get $\widetilde{k}^i_{2,x}\dot{x} = -\widetilde{k}^i_{2,\mathrm{u}^i}\dot{\mathrm{u}}^i$ and it follows that

$$Z^{i+1}_{43}\widetilde{k}^i_{2,x}\dot{x} = -Z^{i+1}_{43}\widetilde{k}^i_{2,\mathrm{u}^i}\dot{\mathrm{u}}^i. \tag{3.90}$$

Therefore, from equation (3.88a) and the sum of (3.89) and (3.90) we get

$$\begin{aligned} S^{i+1}S^i\cdots S^0E\dot{x} &= S^{i+1}S^i\cdots S^0k, && \text{(3.91a)}\\ (-Z^{i+1}_{42}K^i - Z^{i+1}_{43}\widetilde{k}^i_{2,x})\dot{x} &= Z^{i+1}_{42}\widetilde{k}^i_{12} + Z^{i+1}_{43}\widetilde{k}^i_{2,\mathrm{u}^i}\dot{\mathrm{u}}^i. && \text{(3.91b)}\end{aligned}$$

On the other hand, the equation (3.88d) with (3.87) and the trivial equation $S^{i+1}S^i\cdots S^0k = S^{i+1}S^i\cdots S^0k$ yield

$$\begin{aligned} S^{i+1}S^i\cdots S^0k &= S^{i+1}S^i\cdots S^0k, && \text{(3.92a)}\\ Z^{i+1}_{41}S^i\cdots S^0k &= Z^{i+1}_{42}\widetilde{k}^i_{12} + Z^{i+1}_{43}\widetilde{k}^i_{2,\mathrm{u}^i}\dot{\mathrm{u}}^i. && \text{(3.92b)}\end{aligned}$$

From (3.91) and (3.92) it follows because of the same right-hand side that

$$\begin{aligned} S^{i+1}S^i\cdots S^0E\dot{x} &= S^{i+1}S^i\cdots S^0k,\\ (-Z^{i+1}_{42}K^i - Z^{i+1}_{43}\widetilde{k}^i_{2,x})\dot{x} &= Z^{i+1}_{41}S^i\cdots S^0k\end{aligned}$$

which is equivalent to

$$\begin{bmatrix} S^{i+1}S^i\cdots S^0E \\ -Z^{i+1}_{42}K^i - Z^{i+1}_{43}\widetilde{k}^i_{2,x} \end{bmatrix}\dot{x} = \begin{bmatrix} S^{i+1} \\ Z^{i+1}_{41} \end{bmatrix} S^i\cdots S^0k.$$

Because of the nonsingularity of the matrix on the right-hand side, see (3.80), this is equivalent to

$$\begin{bmatrix} S^{i+1} \\ Z^{i+1}_{41} \end{bmatrix}^{-1}\begin{bmatrix} S^{i+1}\cdots S^0E \\ -Z^{i+1}_{42}K^i - Z^{i+1}_{43}\widetilde{k}^i_{2,x} \end{bmatrix}\dot{x} = S^i\cdots S^0k. \tag{3.93}$$

Furthermore, we have from (3.81) that

$$\begin{bmatrix} S^{i+1} \\ Z^{i+1}_{41} \end{bmatrix} S^i\cdots S^0E = \begin{bmatrix} S^{i+1}S^i\cdots S^0E \\ Z^{i+1}_{41}S^i\cdots S^0E \end{bmatrix} = \begin{bmatrix} S^{i+1}S^i\cdots S^0E \\ -Z^{i+1}_{42}K^i - Z^{i+1}_{43}\widetilde{k}^i_{2,x} \end{bmatrix}.$$

Therefore, we obtain

$$\begin{bmatrix} S^{i+1} \\ Z^{i+1}_{41} \end{bmatrix}^{-1}\begin{bmatrix} S^{i+1}S^i\cdots S^0E \\ -Z^{i+1}_{42}K^i - Z^{i+1}_{43}\widetilde{k}^i_{2,x} \end{bmatrix} = S^i\cdots S^0E$$

and it follows from (3.93) that

$$S^i \cdots S^0 E\dot{x} = S^i \cdots S^0 k.$$

Together with (3.88b) and (3.88c) it follows that every solution of (3.88) also solves (3.84) and we get the assertion by the recursion mentioned above. □

Remark 3.5.53 The projected-strangeness-free formulation (3.77) without selecting nonredundant constraints corresponds to a regularization of the quasi-linear DAE (3.23) because both formulations have the same set of solutions (see Theorem 3.5.52), because the reduced maximal constraint level is reduced (see Remark 3.5.49), and because the formulation (3.77) is strangeness-free (see Lemma 3.5.47). But (3.77) is in general not suitable for the numerical integration because of the (possible) redundancies in the constraints.
On the other hand, if the quasi-linear DAE (3.23) contains redundant algebraic constraints, the projected-strangeness-free formulation (3.78) with selected nonredundant constraints possibly has more solutions than the original quasi-linear DAE (3.23) since a selection of a set of nonredundant constraints by use of a kinematic selector could lead to an enlargement of the set of solutions, see Example 2.3.15. But nevertheless, by such a selection the set of solutions is increased in a way such that the (possibly) new obtained solutions are separated from the solution manifold of the original quasi-linear DAE (3.23), see Lemma 2.3.14. Furthermore, in the numerical treatment the numerical evaluation of the selectors is done along the specific solution $x(t)$ which is determined by the given initial values $x(t_0) = x_0$. Therefore, the dynamic selector as well as the kinematic selector are evaluated as function only depending on t and not explicitely on x, i.e, we have $S_D(t) = S_D(x(t), \mathfrak{u}^{\nu_c}(t))$ and $S_C(t) = S_C(x(t), \mathfrak{u}^{\nu_c}(t))$ as matrix functions depending on t. Hence, the projected-strangeness-free formulation (3.78) with selected nonredundant constraints can serve as basis for the numerical integration because of the separation of the additional solutions from the solutions of (3.23).
Note that in case of nonredundant constraints the formulations (3.77) and (3.78) are identical except for a nonlinear scaling of the constraints. □

Remark 3.5.54 Note that if we use Procedure 3.5.11 for the determination of the projected-strangeness-free formulation (3.77), it is only necessary to determine the derivatives of the algebraic constraints. The derivatives of the differential parts need not be determined because they have no influence on the projected-strangeness-free formulation (3.77). In contrast, if we use Hypothesis 3.2.7, it is necessary to determine the derivatives of the whole DAE. Therefore, the use of Hypothesis 3.2.7 for the determination of the equivalent strangeness-free formulation (3.21) of a quasi-linear DAE (3.23) is more involved than the use of Procedure 3.5.11. Even if the leading matrix E of the quasi-linear DAE (3.23) has constant rank with respect to x, then the determination of the transformation matrices Z^i in (3.39) is more simple than for leading matrices E having nonconstant rank. Therefore, the execution of the Procedure 3.5.11 becomes still less involved than the use of the Hypothesis 3.2.7. On the other hand, the Procedure 3.5.11 is restricted to quasi-linear DAEs only, while the Hypothesis 3.2.7 is suited for general nonlinear DAEs. □

Remark 3.5.55 The projected-strangeness-free formulation (3.77) corresponds to a regularization of the quasi-linear DAE (3.23), since the projected-strangeness-free formulation is strangeness-free and leads to the same solution for prescribed consistent initial values. Therefore, the projected-strangeness-free formulation is

useful for the numerical integration by use of numerical algorithms suited for stiff ODEs like implicit Runge-Kutta methods or methods based on *backward differential formulas* (BDF methods).
The Procedure 3.5.11 also offers the possibility to regularize the DAE only up to a certain level, i.e., because of the inductive nature of Procedure 3.5.11 it is possible to stop it prematurely in (3.42), say with $\nu = \widetilde{\nu}_c + 1 < \nu_c + 1$. Assume that $r^{\widetilde{\nu}_c}$, $a^{\widetilde{\nu}_c}$, and $d^{\widetilde{\nu}_c}$ are defined by (3.74) with $l = \widetilde{\nu}_c$ and assume that $r^{\widetilde{\nu}_c} = \text{const}$, $a^{\widetilde{\nu}_c} = \text{const}$, and $d^{\widetilde{\nu}_c} = \text{const}$ for all $(x, \mathfrak{u}^{\widetilde{\nu}_c}) \in \mathbb{M}$. In this case the formulation (3.77) with $\widetilde{\nu}_c$ instead of ν_c and with a dynamic selector $S_D(x, \mathfrak{u}^{\widetilde{\nu}_c})$ of size $d^{\widetilde{\nu}_c - 1} \times m$ with $d^{\widetilde{\nu}_c - 1} = r^{\widetilde{\nu}_c - 1} - a^{\widetilde{\nu}_c - 1}$, $a^{\widetilde{\nu}_c - 1}$ defined in (3.74b) and $r^{\widetilde{\nu}_c - 1}$ defined by (3.74c) satisfying

$$\operatorname{rank}\left(\begin{bmatrix} S_D(x, \mathfrak{u}^{\nu_s})E(x,u) \\ \widetilde{k}^0_{2,x}(x,u) \\ \widetilde{k}^1_{2,x}(x, \mathfrak{u}^1) \\ \vdots \\ \widetilde{k}^{\widetilde{\nu}_c - 1}_{2,x}(x, \mathfrak{u}^{\widetilde{\nu}_c - 1}) \end{bmatrix}\right) = r^{\widetilde{\nu}_c - 1}$$

for all $(x, \mathfrak{u}^{\widetilde{\nu}_c}) \in \mathbb{M}_{\widetilde{\nu}_c}$ has maximal constraint level $\nu_c - \widetilde{\nu}_c > 0$ and contains, therefore, hidden constraints up to the maximal level $\nu_c - \widetilde{\nu}_c$. But nevertheless, this corresponds to a regularization because of the lowering of the index and the determination of the hidden constraints up to level $\widetilde{\nu}_c$. We call this an *incomplete regularization*. From the proof of Theorem 3.5.52 it follows that $S_D = S^{\widetilde{\nu}_c - 1} \cdots S^0$ and that every solution of an incomplete regularization (3.77) with $\widetilde{\nu}_c$ instead of ν_c also solves the original quasi-linear DAE (3.23).
Furthermore, according to Remark 3.5.53 the formulation (3.78) with $\widetilde{\nu}_c$ instead of ν_c and with a kinematic selector $S_C(x, \mathfrak{u}^{\widetilde{\nu}_c})$ defined in Definition 3.5.43 with $\widetilde{\nu}_c$ instead of ν_c can serve as basis for the numerical integration. Furthermore, note that an incomplete regularization is only possible for $\widetilde{\nu}_c > 0$ since a regularization bases on additional information obtained from the derivative of constraints with respect to t. □

In the following Example 3.5.56 we will demonstrate the regularization to the projected-strangeness-free formulation as well as the incomplete regularization of a semi-explicit DAE in Hessenberg form (3.25) of order 3.

Example 3.5.56 Let us consider the semi-explicit DAE

$$\begin{aligned} \dot{x}_1 &= k_1(x_1, x_2, u), && \text{(3.94a)} \\ \dot{x}_2 &= k_2(x_1, x_2, x_3, u), && \text{(3.94b)} \\ 0 &= k_3(x_1, u) && \text{(3.94c)} \end{aligned}$$

with $x_1 \in \mathcal{C}^1(\mathbb{I}, \mathbb{X}_1)$, $x_2 \in \mathcal{C}^1(\mathbb{I}, \mathbb{X}_2)$, $x_3 \in \mathcal{C}(\mathbb{I}, \mathbb{X}_3)$, $u \in \mathcal{C}(\mathbb{I}, \mathbb{U})$, $k_1 \in \mathcal{C}^1(\mathbb{X}_1 \times \mathbb{X}_2 \times \mathbb{U}, \mathbb{R}^{n_1})$, $k_2 \in \mathcal{C}(\mathbb{X}_1 \times \mathbb{X}_2 \times \mathbb{X}_3 \times \mathbb{U}, \mathbb{R}^{n_2})$, $k_3 \in \mathcal{C}^2(\mathbb{X}_1 \times \mathbb{U}, \mathbb{R}^{n_3})$, $\mathbb{X}_1 \subset \mathbb{R}^{n_1}$, $\mathbb{X}_2 \subset \mathbb{R}^{n_2}$, $\mathbb{X}_3 \subset \mathbb{R}^{n_3}$, and assume that $k_{3,x_1} k_{1,x_2} k_{2,x_3}$ is nonsingular. Note that the DAE (3.94) in this form does not correspond to the form (3.25) but by permuting the meaning of x_1 with x_2 and k_1 with k_2 the DAE (3.94) can be written in form (3.25). Therefore, the DAE (3.94) is in Hessenberg form of order 3.
Following Procedure 3.5.11 we have

$$E^0 = \begin{bmatrix} I_{m_1} & 0 & 0 \\ 0 & I_{m_2} & 0 \\ 0 & 0 & 0 \end{bmatrix}, \; k^0 = \begin{bmatrix} k_1 \\ k_2 \\ k_3 \end{bmatrix}.$$

With transformation matrix $Z^0 = I$ we get the intermediate DAE (3.40) with $i = 0$ and

$$\begin{aligned} \widetilde{E}_1^0 &= \begin{bmatrix} I_{m_1} & 0 & 0 \\ 0 & I_{m_2} & 0 \end{bmatrix}, \quad \widetilde{k}_1^0 = \begin{bmatrix} k_1 \\ k_2 \end{bmatrix}, \\ & \qquad\qquad \widetilde{k}_2^0 = \begin{bmatrix} k_3 \end{bmatrix}. \end{aligned}$$

The constraints of level 0 are given by $0 = \widetilde{k}_2^0 = k_3$ and therefore, we get $\mathbb{M}_0 = \widetilde{\mathbb{M}}_0 = \{(x,u) \in \mathbb{X} \times \mathbb{U} \ : \ 0 = k_3\}$. Differentiation of the constraints of level 0 with respect to t yields $0 = k_{3,x_1}\dot{x}_1 + k_{3,u}\dot{u}$ and we get the transformed DAE $E^1(x,u)\dot{x} = k^1(x,\mathfrak{u}^1)$ with

$$E^1 = \begin{bmatrix} I_{m_1} & 0 & 0 \\ 0 & I_{m_2} & 0 \\ k_{3,x_1} & 0 & 0 \end{bmatrix}, \ k^1 = \begin{bmatrix} k_1 \\ k_2 \\ -k_{3,u}\dot{u} \end{bmatrix}.$$

The transformation with the transformation matrix

$$Z^1(x,u) = \begin{bmatrix} I_{m_1} & 0 & 0 \\ 0 & I_{m_2} & 0 \\ -k_{3,x_1} & 0 & I_{m_3} \end{bmatrix}$$

leads to the intermediate DAE (3.40) with $i = 1$ and

$$\begin{aligned} \widetilde{E}_1^1 &= \begin{bmatrix} I_{m_1} & 0 & 0 \\ 0 & I_{m_2} & 0 \end{bmatrix}, \quad \widetilde{k}_1^1 = \begin{bmatrix} k_1 \\ k_2 \end{bmatrix}, \\ & \qquad\qquad \widetilde{k}_2^1 = \begin{bmatrix} -k_{3,x_1}k_1 - k_{3,u}\dot{u} \end{bmatrix}. \end{aligned}$$

Now we have determined the constraints of level 1 by $0 = \widetilde{k}_2^1 = k_{3,x_1}k_1 + k_{3,u}\dot{u}$. Furthermore, we get $\widetilde{\mathbb{M}}_1 = \{(x,\mathfrak{u}^1) \in \mathbb{X} \times \mathbb{U}^1 \ : \ 0 = k_{3,x_1}k_1 + k_{3,u}\dot{u}\}$ and the constraint set of level 1 by $\mathbb{M}_1 = \{(x,\mathfrak{u}^1) \in \mathbb{X} \times \mathbb{U}^1 \ : \ 0 = k_3, \ 0 = k_{3,x_1}k_1 + k_{3,u}\dot{u}\}$. Since the constraints of level 1 are not satisfied for all $(x,\mathfrak{u}^1) \in \mathbb{M}_0 \times \mathbb{U}^{(1)}$, we have to continue the procedure. Differentiation of the constraints of level 1 with respect to t yields

$$0 = (k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,x_1}\dot{x}_1 + k_{3,x_1}k_{1,x_2}\dot{x}_2 + (k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,\mathfrak{u}^1}\dot{\mathfrak{u}}^1$$

and we get the transformed DAE $E^2(x,\mathfrak{u}^1)\dot{x} = k^2(x,\mathfrak{u}^2)$ with

$$E^2 = \begin{bmatrix} I_{m_1} & 0 & 0 \\ 0 & I_{m_2} & 0 \\ (k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,x_1} & k_{3,x_1}k_{1,x_2} & 0 \end{bmatrix}, \ k^2 = \begin{bmatrix} k_1 \\ k_2 \\ -(k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,\mathfrak{u}^1}\dot{\mathfrak{u}}^1 \end{bmatrix}.$$

The transformation with the transformation matrix

$$Z^2(x,\mathfrak{u}^1) = \begin{bmatrix} I_{m_1} & 0 & 0 \\ 0 & I_{m_2} & 0 \\ -(k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,x_1} & -k_{3,x_1}k_{1,x_2} & I_{m_3} \end{bmatrix}$$

leads to the intermediate DAE (3.40) with $i = 2$ and

$$\begin{aligned} \widetilde{E}_1^2 &= \begin{bmatrix} I_{m_1} & 0 & 0 \\ 0 & I_{m_2} & 0 \end{bmatrix}, \\ \widetilde{k}_1^2 &= \begin{bmatrix} k_1 \\ k_2 \end{bmatrix}, \\ \widetilde{k}_2^2 &= \begin{bmatrix} -(k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,x_1}k_1 - k_{3,x_1}k_{1,x_2}k_2 - (k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,\mathfrak{u}^1}\dot{\mathfrak{u}}^1 \end{bmatrix}. \end{aligned}$$

Now we have determined the constraints of level 2 by $0 = \widetilde{k}_2^2 = (k_{3,x_1}k_1)_{,x_1}k_1 + k_{3,x_1}k_{1,x_2}k_2 + (k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,\mathrm{u}^1}\dot{\mathrm{u}}^1$. Furthermore, we get $\widetilde{\mathbb{M}}_2 = \{(x,\mathrm{u}^2) \in \mathbb{X}\times\mathbb{U}^2 : 0 = (k_{3,x_1}k_1)_{,x_1}k_1 + k_{3,x_1}k_{1,x_2}k_2 + (k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,\mathrm{u}^1}\dot{\mathrm{u}}^1\}$ and the constraint set of level 2 by $\mathbb{M}_2 = \{(x,\mathrm{u}^2) \in \mathbb{X}\times\mathbb{U}^2 : 0 = k_3,\ 0 = k_{3,x_1}k_1 + k_{3,u}\dot{u},\ 0 = (k_{3,x_1}k_1)_{,x_1}k_1 + k_{3,x_1}k_{1,x_2}k_2 + (k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,\mathrm{u}^1}\dot{\mathrm{u}}^1\}$. Since the constraints of level 2 are still not satisfied for all $(x,\mathrm{u}^2) \in \mathbb{M}_1\times\mathbb{U}^{(2)}$, we have to continue the procedure. Differentiation of the constraints of level 2 with respect to t yields

$$\begin{aligned} 0 \;=\; & ((k_{3,x_1}k_1)_{,x_1}k_1 + k_{3,x_1}k_{1,x_2}k_2 + (k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,\mathrm{u}^1}\dot{\mathrm{u}}^1)_{,x_1}\dot{x}_1 \\ & +((k_{3,x_1}k_1)_{,x_1}k_1 + k_{3,x_1}k_{1,x_2}k_2 + (k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,\mathrm{u}^1}\dot{\mathrm{u}}^1)_{,x_2}\dot{x}_2 \\ & +k_{3,x_1}k_{1,x_2}k_{2,x_3}\dot{x}_3 \\ & +((k_{3,x_1}k_1)_{,x_1}k_1 + k_{3,x_1}k_{1,x_2}k_2 + (k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,\mathrm{u}^1}\dot{\mathrm{u}}^1)_{,\mathrm{u}^2}\dot{\mathrm{u}}^2 \end{aligned}$$

and we get the transformed DAE $E^3(x,\mathrm{u}^2)\dot{x} = k^3(x,\mathrm{u}^3)$ with

$$\begin{aligned} E^3 \;&=\; \begin{bmatrix} I_{m_1} & 0 & 0 \\ 0 & I_{m_2} & 0 \\ E^3_{31} & E^3_{32} & k_{3,x_1}k_{1,x_2}k_{2,x_3} \end{bmatrix}, \\ k^3 \;&=\; \begin{bmatrix} k_1 \\ k_2 \\ -((k_{3,x_1}k_1)_{,x_1}k_1 + k_{3,x_1}k_{1,x_2}k_2 + (k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,\mathrm{u}^1}\dot{\mathrm{u}}^1)_{,\mathrm{u}^2}\dot{\mathrm{u}}^2 \end{bmatrix} \end{aligned}$$

with

$$\begin{aligned} E^3_{31} \;&=\; ((k_{3,x_1}k_1)_{,x_1}k_1 + k_{3,x_1}k_{1,x_2}k_2 + (k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,\mathrm{u}^1}\dot{\mathrm{u}}^1)_{,x_1}, \\ E^3_{32} \;&=\; ((k_{3,x_1}k_1)_{,x_1}k_1 + k_{3,x_1}k_{1,x_2}k_2 + (k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,\mathrm{u}^1}\dot{\mathrm{u}}^1)_{,x_2}. \end{aligned}$$

Because of the assumed nonsingularity of $k_{3,x_1}k_{1,x_2}k_{2,x_3}$ we have $m_2^3 = 0$ and the Procedure 3.5.11 terminates with $\nu = 3$. Therefore, the DAE (3.94) has maximal constraint level $\nu_c = \nu - 1 = 2$ and because of the nonsingularity of the matrix function E^3 for all $(x,\mathrm{u}^2) \in \mathbb{M} = \mathbb{M}_2$ the DAE (3.94) has in addition d-index $\nu_d = \nu = 3$. The s-index of the DAE (3.94) is $\nu_s = 2$ which can be verified by Hypothesis 3.2.7. The solution manifold is given by

$$\begin{aligned} \mathbb{M} = \{(x,\mathrm{u}^2) \in \mathbb{X}\times\mathbb{U}^2 \;:\; & 0 = k_3, \\ & 0 = k_{3,x_1}k_1 + k_{3,u}\dot{u}, \\ & 0 = (k_{3,x_1}k_1)_{,x_1}k_1 + k_{3,x_1}k_{1,x_2}k_2 + (k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,\mathrm{u}^1}\dot{\mathrm{u}}^1\}. \end{aligned}$$

The complete minimal reduced derivative array is defined by

$$\widetilde{\mathfrak{F}}_2(x,\dot{x},\mathrm{u}^2) = \begin{bmatrix} \dot{x}_1 - k_1 \\ \dot{x}_2 - k_2 \\ k_3 \\ k_3 \\ k_{3,x_1}k_1 + k_{3,u}\dot{u} \\ (k_{3,x_1}k_1)_{,x_1}k_1 + k_{3,x_1}k_{1,x_2}k_2 + (k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,\mathrm{u}^1}\dot{\mathrm{u}}^1 \end{bmatrix}.$$

Because of the assumed nonsingularity of $k_{3,x_1}k_{1,x_2}k_{2,x_3}$, we get from Definition 3.5.43 the kinematic selector as identity, i.e., $S_C = I_{3m_3}$. Furthermore, according to Definition 3.5.44 we can choose the dynamic selector as

$$S_D = \begin{bmatrix} S_{D11} & 0 & 0 \\ 0 & S_{D22} & 0 \end{bmatrix}$$

such that

$$\begin{bmatrix} S_{D11} \\ k_{3,x_1} \end{bmatrix} \text{ and } \begin{bmatrix} S_{D22} \\ k_{3,x_1}k_{1,x_2} \end{bmatrix}$$

are nonsingular, and we get for the DAE (3.94) the associated projected-strangeness-free formulation

$$0 = \begin{bmatrix} S_{D11}(\dot{x}_1 - k_1) \\ S_{D22}(\dot{x}_2 - k_2) \\ k_3 \\ k_{3,x_1}k_1 + k_{3,u}\dot{u} \\ (k_{3,x_1}k_1)_{,x_1}k_1 + k_{3,x_1}k_{1,x_2}k_2 + (k_{3,x_1}k_1 + k_{3,u}\dot{u})_{,\mathfrak{u}^1}\dot{\mathfrak{u}}^1 \end{bmatrix}.$$

In the following let us discuss an incomplete regularization of level 1. This means that we stop the Procedure 3.5.11 already with $i - 1 = \widetilde{\nu}_c = 1$. Then, we get the (incomplete) reduced derivative array

$$\widetilde{\mathfrak{F}}_1(x, \dot{x}, \mathfrak{u}^1) = \begin{bmatrix} \dot{x}_1 - k_1 \\ \dot{x}_2 - k_2 \\ k_3 \\ k_3 \\ k_{3,x_1}k_1 + k_{3,u}\dot{u} \end{bmatrix}.$$

From (3.74c) we get

$$r^{\widetilde{\nu}_c - 1} = \operatorname{rank}\left(\begin{bmatrix} E \\ \widetilde{k}^0_{2,x} \end{bmatrix}\right) = \operatorname{rank}\left(\begin{bmatrix} I_{m_1} & 0 & 0 \\ 0 & I_{m_2} & 0 \\ 0 & 0 & 0 \\ k_{3,x_1} & 0 & 0 \end{bmatrix}\right) = m_1 + m_2$$

and $a^{\widetilde{\nu}_c - 1}$ is given from (3.74b) with $\widetilde{\nu}_c = 1$ instead of ν_c by $a^{\widetilde{\nu}_c - 1} = m_3$. Because of the full rank of the constraints which follows from the nonsingularity of $k_{3,x_1}k_{1,x_2}k_{2,x_3}$ we can again choose the kinematic selector as the identity, i.e., $S_C(x, \mathfrak{u}^1) = I_{2m_3}$. According to Remark 3.5.55, then the dynamic selector

$$S_D = \begin{bmatrix} S_{D1} & S_{D2} & S_{D3} \end{bmatrix}$$

of size $d^{\widetilde{\nu}_c - 1} \times m$ with $d^{\widetilde{\nu}_c - 1} = r^{\widetilde{\nu}_c - 1} - a^{\widetilde{\nu}_c - 1} = m_1 + m_2 - m_3$ has to be determined such that

$$\operatorname{rank}\left(\begin{bmatrix} S_{D1} & S_{D2} & S_{D3} \\ k_{3,x_1} & 0 & 0 \end{bmatrix}\right) = r^{\widetilde{\nu}_c - 1} = m_1 + m_2.$$

In particular, this can be reached by

$$S_{D1} = \begin{bmatrix} S_{D11} \\ 0 \end{bmatrix},\ S_{D2} = \begin{bmatrix} 0 \\ I_{m_2} \end{bmatrix},\ S_{D3} = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

with S_{D11} of size $m_1 - m_3 \times m_1$ chosen such that again the matrix

$$\begin{bmatrix} S_{D11} \\ k_{3,x_1} \end{bmatrix}$$

is nonsingular. With respect to the DAE (3.94) this leads to

$$0 = \begin{bmatrix} S_{D11}(\dot{x}_1 - k_1) \\ \dot{x}_2 - k_2 \\ k_3 \\ k_{3,x_1}k_1 + k_{3,u}\dot{u} \end{bmatrix} \tag{3.95}$$

with maximal constraint level $\nu_c = 1$ and which has s-index $\nu_s = 1$ which can be checked by use of the Hypothesis 3.2.7. Let us call the DAE (3.95) the associated *projected-s-index-1 formulation* of the quasi-linear DAE (3.94). □

Example 3.5.57 Consider the differential-algebraic equation (3.34), see Example 3.5.7. Assume that $k_{2,x_1}k_{1,x_2}$ is nonsingular. In Example 3.5.42 we did execute the Procedure 3.5.11, where we did determine its maximal constraint level $\nu_c = 1$. Furthermore, the solution manifold $\mathbb{M}$ is given in (3.72) and the complete minimal reduced derivative array $\widetilde{\mathfrak{F}}_1$ is given in (3.73).
Following Definition 3.5.43 we can choose the kinematic selector as

$$S_C(x, \mathfrak{u}^1) = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}$$

and following Definition 3.5.44 we have the dynamic selector

$$S_D = \begin{bmatrix} S_{D1} & S_{D2} \end{bmatrix}$$

of size $n_1 - n_2 \times n_1 + n_2$ with $S_{D2} = 0$ and such that

$$\operatorname{rank}\left(\begin{bmatrix} S_{D1} & 0 \\ k_{2,x_1} & 0 \\ (k_{2,x_1}k_1)_{,x_1} & k_{2,x_1}k_{1,x_2} \end{bmatrix} \right) = n_1 + n_2.$$

Hence, the projected-strangeness-free form of the DAE (3.34) is given by

$$\begin{bmatrix} S_{D1} & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \dot{x} = \begin{bmatrix} S_{D1}k_1 \\ k_2 \\ k_{2,x_1}k_1 + k_{2,u}\dot{u} \end{bmatrix} \tag{3.96}$$

with S_{D1} chosen such that the matrix

$$\begin{bmatrix} S_{D1} \\ k_{2,x_1} \end{bmatrix}$$

is nonsingular. In the case of $n_1 = n_2$ the dynamic selector S_D is of size $0 \times n_1 + n_2$ and then the projected-strangeness-free form (3.96) corresponds to a set of two algebraic equations in accordance with Example 3.5.7. □

Example 3.5.58 Consider the DAE

$$\begin{bmatrix} (2x_1 + x_2) & x_1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_1^2 + x_1x_2 \end{bmatrix} \tag{3.97}$$

with $\mathbb{X} = \mathbb{R}^2$ and $t \in \mathbb{I}$. Following Procedure 3.5.11 we get the constraint $0 = \widetilde{k}_2^0(x) = x_1^2 + x_1x_2$ which does not define a manifold because $\operatorname{rank}(\widetilde{k}_{2,x}^0(x))$ is not constant for all x satisfying this constraint, see Lemma 2.3.8. Nevertheless, we get the constraint set of level 0 as $\mathbb{M}_0 = \{x \in \mathbb{R}^2 \ : \ x_1^2 + x_1x_2 = 0\}$. For the quantities r^0, a^0, and d^0 we get from (3.74)

$$r^0 = \begin{cases} 0 & \text{for } x_1 = x_2 = 0, \\ 1 & \text{else,} \end{cases} \qquad a^0 = \begin{cases} 0 & \text{for } x_1 = x_2 = 0, \\ 1 & \text{else,} \end{cases} \qquad d^0 = 0,$$

where r^0 and a^0 are not constant in $\mathbb{M}_0$. By differentiating the constraint we get

$$\begin{bmatrix} (2x_1 + x_2) & x_1 \\ (2x_1 + x_2) & x_1 \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ 0 \end{bmatrix}$$

and elimination yields

$$\begin{bmatrix} (2x_1 + x_2) & x_1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ -x_1 \end{bmatrix}.$$

Obviously, the hidden constraint $0 = \widetilde{k}_2^1(x) = x_1$ occurs, i.e., $m_2^1 = 1 \neq 0$ and the hidden constraint is not satisfied automatically for all $x \in \mathbb{M}_0$. Therefore, the procedure does not terminate and we get the constraint set of level 1

$$\mathbb{M}_1 = \{x \in \mathbb{R}^2 \ : \ x_1^2 + x_1 x_2 = 0, \ x_1 = 0\} = \{x \in \mathbb{R}^2 \ : \ x_1 = 0\}.$$

For the quantities r^1, a^1, and d^1 we get from (3.74)

$$r^1 = 1, \qquad a^1 = 1, \qquad d^1 = 0,$$

which are constant in $\mathbb{M}_1$. From further differentiation of the constraint we get

$$\begin{bmatrix} (2x_1 + x_2) & x_1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ 0 \end{bmatrix}$$

which is equivalent to

$$\begin{bmatrix} x_2 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

for all $x \in \mathbb{M}_1$. Elimination yields

$$\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{3.98}$$

The last equation corresponds to a constraint $0 = \widetilde{k}_2^2 = 0$, which is satisfied for all $x \in \mathbb{M}_1$ because of $x_1 = 0$. Thus, the procedure terminates in iteration step $\nu = 2$ in (3.42) and we get the maximal constraint level $\nu_c = 1$.
From Definition 3.5.50 we get the projected-strangeness-free form (3.77) of the DAE (3.97) by

$$0 = \begin{bmatrix} x_1^2 + x_1 x_2 \\ x_1 \end{bmatrix}.$$

Obviously, the solution is not uniquely determined. We get $x_1 = 0$ and x_2 arbitrary as set of solutions for the DAE (3.97). Note that because of the singularity of the constraints of level 0 this example does not satisfy Hypothesis 3.2.7. □

Example 3.5.59 Returning to Example 3.5.41

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & x_1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} x_1 x_2 \\ x_2 - 1 \\ x_3 - 1 \end{bmatrix},$$

with $\mathbb{X} = \mathbb{R}^3$ and $t \in \mathbb{I}$. With a kinematic selector $S_C = I_2$ and a dynamic selector $S_D = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$ we get the projected strangeness-free form of the DAE as

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} x_1 x_2 \\ x_3 - 1 \\ x_2 - 1 \end{bmatrix}.$$

□

Example 3.5.60 The stirred tank: In [157] a stirred liquid-liquid dispersion, i.e., a tank filled with two immiscible fluids is considered which are stirred so that one of the phases disperses into the other one by building droplets. The modeling of the flow field is done by Navier[7]-Stokes[8] equations for incompressible fluids and the dispersed phase is modeled by the general population balanced equation. For the detailed considerations we refer to [157]. After some (simplifying) assumptions and semi-discretization with respect to the space variables a linear DAE of the form

$$\begin{bmatrix} I & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{u}(t) \\ \dot{m}(t) \\ \dot{p}(t) \end{bmatrix} = \begin{bmatrix} G(t) & 0 & C \\ 0 & R(t) & 0 \\ \widetilde{C} & 0 & 0 \end{bmatrix} \begin{bmatrix} u(t) \\ m(t) \\ p(t) \end{bmatrix} + \begin{bmatrix} f_1(t) \\ f_2(t) \\ f_3(t) \end{bmatrix} \tag{3.99}$$

is obtained, where $R \in \mathcal{C}^1(\mathbb{I}, \mathbb{R}^{n_m,n_m})$ is nonsingular for all $t \in \mathbb{I}$, $\widetilde{C}, C^T \in \mathbb{R}^{n_p,n_u}$, $G \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{n_u,n_u})$, $f_1 \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{n_u})$, $f_2 \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{n_m})$, and $f_3 \in \mathcal{C}^1(\mathbb{I}, \mathbb{R}^{n_p})$. Following Procedure 3.5.11 we get the intermediate DAE (3.40) with $i = 0$ and

$$\widetilde{E}_1^0 = \begin{bmatrix} I & 0 & 0 \end{bmatrix}, \qquad \widetilde{k}_1^0 = \begin{bmatrix} G(t) & 0 & C \end{bmatrix} \begin{bmatrix} u(t) \\ m(t) \\ p(t) \end{bmatrix} + \begin{bmatrix} f_1(t) \end{bmatrix}, \tag{3.100a}$$

$$\widetilde{k}_2^0 = \begin{bmatrix} 0 & R(t) & 0 \\ \widetilde{C} & 0 & 0 \end{bmatrix} \begin{bmatrix} u(t) \\ m(t) \\ p(t) \end{bmatrix} + \begin{bmatrix} f_2(t) \\ f_3(t) \end{bmatrix}. \tag{3.100b}$$

The derivative of the constraints $0 = \widetilde{k}_2^0$ with respect to t together with the differential part of the original DAE formally leads to

$$\begin{bmatrix} I & 0 & 0 \\ 0 & -R(t) & 0 \\ -\widetilde{C} & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{u}(t) \\ \dot{m}(t) \\ \dot{p}(t) \end{bmatrix} = \begin{bmatrix} G(t) & 0 & C \\ 0 & \dot{R}(t) & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u(t) \\ m(t) \\ p(t) \end{bmatrix} + \begin{bmatrix} f_1(t) \\ \dot{f}_2(t) \\ \dot{f}_3(t) \end{bmatrix}.$$

Multiplication by

$$Z^1 = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ \widetilde{C} & 0 & I \end{bmatrix}$$

from the left yields the intermediate DAE (3.40) with $i = 1$ and

$$\widetilde{E}_1^1 = \begin{bmatrix} I & 0 & 0 \\ 0 & -R(t) & 0 \end{bmatrix}, \tag{3.101a}$$

$$\widetilde{k}_1^1 = \begin{bmatrix} G(t) & 0 & C \\ 0 & \dot{R}(t) & 0 \end{bmatrix} \begin{bmatrix} u(t) \\ m(t) \\ p(t) \end{bmatrix} + \begin{bmatrix} f_1(t) \\ \dot{f}_2(t) \end{bmatrix}, \tag{3.101b}$$

$$\widetilde{k}_2^1 = \begin{bmatrix} \widetilde{C}G(t) & 0 & \widetilde{C}C \end{bmatrix} \begin{bmatrix} u(t) \\ m(t) \\ p(t) \end{bmatrix} + \begin{bmatrix} \widetilde{C}f_1(t) + \dot{f}_3(t) \end{bmatrix}. \tag{3.101c}$$

[7] Claude Louis Marie Henri Navier (born 1785 in Dijon, France - died 1836 in Paris, France)

[8] George Gabriel Stokes (born 1819 in Skreen, County Sligo, Ireland - died 1903 in Cambridge, England)

One further differentiation of the constraints $0 = \widetilde{k}_2^1$ with respect to t yields, together with the differential part,

$$\begin{aligned}\begin{bmatrix} I & 0 & 0 \\ 0 & -R(t) & 0 \\ -\widetilde{C}G(t) & 0 & -\widetilde{C}C \end{bmatrix} \begin{bmatrix} \dot{u}(t) \\ \dot{m}(t) \\ \dot{p}(t) \end{bmatrix} \\ = \begin{bmatrix} G(t) & 0 & C \\ 0 & \dot{R}(t) & 0 \\ \frac{\mathrm{d}}{\mathrm{d}t}(\widetilde{C}G(t)) & 0 & \frac{\mathrm{d}}{\mathrm{d}t}(\widetilde{C}C) \end{bmatrix} \begin{bmatrix} u(t) \\ m(t) \\ p(t) \end{bmatrix} + \begin{bmatrix} f_1(t) \\ \dot{f}_2(t) \\ \frac{\mathrm{d}}{\mathrm{d}t}(\widetilde{C}f_1(t) + \dot{f}_3(t)) \end{bmatrix}.\end{aligned} \tag{3.102}$$

Recall that $R(t)$ is assumed to be nonsingular for all $t \in \mathbb{I}$. If $\widetilde{C}C$ is nonsingular as well, then the leading matrix on the left-hand side of (3.102) is nonsingular and (3.102) corresponds to the underlying ODE in implicit form associated with the original DAE (3.99). Therefore, we have $m_2^2 = 0$ and Procedure 3.5.11 terminates with $\nu = 2$, the maximal constraint level is $\nu_c = 1$ and the d-index is $\nu_d = 2$. The (hidden) constraints are given by $0 = k_2^i$, $i = 0, 1$ in (3.100b) and (3.101c). Furthermore, the solution manifold is given by

$$\begin{aligned}\mathbb{M} = \{(u, m, p, t) \in \mathbb{R}^{n_u} \times \mathbb{R}^{n_m} \times \mathbb{R}^{n_p} \times \mathbb{I} : \\ 0 = \begin{bmatrix} 0 & R(t) & 0 \\ \widetilde{C} & 0 & 0 \end{bmatrix} \begin{bmatrix} u(t) \\ m(t) \\ p(t) \end{bmatrix} + \begin{bmatrix} f_2(t) \\ f_3(t) \end{bmatrix}, \\ 0 = \begin{bmatrix} \widetilde{C}G(t) & 0 & \widetilde{C}C \end{bmatrix} \begin{bmatrix} u(t) \\ m(t) \\ p(t) \end{bmatrix} + \begin{bmatrix} \widetilde{C}f_1(t) + \dot{f}_3(t) \end{bmatrix}\}\end{aligned}$$

and the associated projected-strangeness-free formulation is determined by

$$\left[\begin{array}{ccc} S_D & 0 & 0 \\ \hline 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline 0 & 0 & 0 \end{array}\right] \begin{bmatrix} \dot{u}(t) \\ \dot{m}(t) \\ \dot{p}(t) \end{bmatrix} = \left[\begin{array}{ccc} S_D G(t) & 0 & S_D C \\ \hline 0 & R(t) & 0 \\ \widetilde{C} & 0 & 0 \\ \hline \widetilde{C}G(t) & 0 & \widetilde{C}C \end{array}\right] \begin{bmatrix} u(t) \\ m(t) \\ p(t) \end{bmatrix} + \left[\begin{array}{c} S_D f_1(t) \\ \hline f_2(t) \\ f_3(t) \\ \hline \widetilde{C}f_1(t) + \dot{f}_3(t) \end{array}\right],$$

with S_D chosen such that

$$\left[\begin{array}{ccc} S_D & 0 & 0 \\ \hline 0 & R(t) & 0 \\ \widetilde{C} & 0 & 0 \\ \hline \widetilde{C}G(t) & 0 & \widetilde{C}C \end{array}\right]$$

is nonsingular. Since $R(t)$ and $\widetilde{C}C$ are assumed to be nonsingular, this condition reduces to the condition that

$$\begin{bmatrix} S_D \\ \widetilde{C} \end{bmatrix}$$

is nonsingular. This is achieved for example by $S_D = (\widetilde{C}^-)^T$ with $\mathrm{range}(\widetilde{C}^-) = \ker(\widetilde{C})$. □

In the following sections we will discuss the discretization of differential-algebraic equations of this quasi-linear form via Runge-Kutta methods.

3.5.4 Runge-Kutta method applied to quasi-linear differential-algebraic equations

In this section we will present an approach for the numerical integration of an initial value problem of general quasi-linear DAEs of the form

$$\hat{E}(x,\hat{u})\dot{x} = \hat{k}(x,\hat{u}), \tag{3.103}$$

by use of an implicit Runge-Kutta method. We are interested in a numerical integration on the domain $\mathbb{I} = [t_0, t_f]$ with given initial values $x(t_0) = x_0$ where $\hat{E} \in \mathcal{C}(\mathbb{X} \times \hat{\mathbb{U}}, \mathbb{R}^{\hat{m},n})$ and $\hat{k} \in \mathcal{C}(\mathbb{X} \times \hat{\mathbb{U}}, \mathbb{R}^{\hat{m}})$ with $\hat{m}, n \in \mathbb{N}_0$, $\mathbb{X} \subset \mathbb{R}^n$, $\hat{\mathbb{U}} \subset \mathbb{R}^{n_{\hat{u}}}$. Furthermore, we will focus on the class of semi-implicit DAEs

$$\begin{bmatrix} \hat{E}_D(x,\hat{u}) \\ 0 \end{bmatrix} \dot{x} = \begin{bmatrix} \hat{k}_D(x,\hat{u}) \\ \hat{k}_C(x,\hat{u}) \end{bmatrix}, \tag{3.104}$$

which arise frequently in industrial applications like the simulation of electrical circuits and, in particular, in multibody dynamics, with $\hat{E}_D \in \mathcal{C}(\mathbb{X} \times \hat{\mathbb{U}}, \mathbb{R}^{\hat{m}_D,n})$, $\hat{k}_D \in \mathcal{C}(\mathbb{X} \times \hat{\mathbb{U}}, \mathbb{R}^{\hat{m}_D})$, and $\hat{k}_C \in \mathcal{C}(\mathbb{X} \times \hat{\mathbb{U}}, \mathbb{R}^{\hat{m}_C})$ with $\hat{m}_D, \hat{m}_C, n \in \mathbb{N}_0$, $\mathbb{X} \subset \mathbb{R}^n$, $\hat{\mathbb{U}} \subset \mathbb{R}^{n_{\hat{u}}}$, and $\hat{m}_D + \hat{m}_C = \hat{m}$.
The presented discretization technique is formally not restricted to quasi-linear DAEs (3.23) of certain index or to quasi-linear DAEs with $m = n$, rather, formally it works for $m = n$ as well as for $m \neq n$ and for quasi-linear DAEs of arbitrary index. Therefore, we have to consider a general control variable $\hat{u} \in \hat{\mathbb{U}}$ which not necessarily corresponds to u. Rather, depending on the used formulation as basis for the numerical integration, it may correspond to $\mathfrak{u}^i$, $i = 0, ..., \nu_c$ in case of the use of an incomplete regularization, see Remark 3.5.55, or in particular, it may corresponds to $\mathfrak{u}^{\nu_c}$ in case of the use of the projected-strangeness-free formulation (3.77) of the quasi-linear DAE.
The following investigations mainly focus on two different types of semi-implicit DAEs and their associated linear systems which we will denote by use of a hat or a tilde on top of the functions. First, the systems denoted by a hat are associated to the regularized DAE (mostly we focus on the projected-strangeness-free formulation with selected constraints (3.78)). In the case of nonredundant constraints these systems have the same dimension as the original quasi-linear DAE (3.23). Secondly, the systems denoted by a tilde are associated to the minimal reduced derivative array (3.68). In general these systems contain more equations than the original quasi-linear DAE (3.23).

Remark 3.5.61 In [136], the numerical solution of nonlinear DAEs in partitioned form (3.5) is considered. The conditioning of the iteration matrix is considered and scaling strategies are proposed. In particular, in [115] it is shown that the condition number of the iteration matrix for a DAE of form (3.5) with d-index ν_d is of order $\mathcal{O}(1/h^{\nu_d})$.
The investigation of strangeness-free DAEs, i.e., of s-index 0 or at most d-index 1, is relevant for our investigation with respect to the numerical integration of regularized model equations of multibody systems, see Chapter 4. For the strangeness-free DAEs in semi-explicit form a scaling of the algebraic constraints with $1/h$ is recommended in [136]. In this case and if the DAE is regular then the iteration matrix tends for $h \to 0$ to a nonsingular matrix instead of a singular matrix in the case without scaling.
While the round-off errors for d-index one systems can be reduced by scaling, the situation for higher index problems is different. In [136], it is shown that the algebraic variables associated with a semi-explicit d-index two problem are influenced by a round-off error up to order $\mathcal{O}(1/h^2)$ without scaling and up to order $\mathcal{O}(1/h)$

with the proposed scaling. In the case of semi-explicit d-index three systems occurring in multibody dynamics the situation is even more delicate. Without scaling, the round-off error is of order $\mathcal{O}(1/h^3)$ and with scaling of $\mathcal{O}(1/h^2)$, which is not satisfactory. □

As mentioned in previous sections, e.g., in Sections 3.2 and 3.4, the direct discretization of higher index DAEs is not advisable such that we will base the numerical integration on the discretization of the projected-strangeness-free form (3.77) of the quasi-linear DAE (3.23) as regularization developed in Section 3.5.3, see Definition 3.5.50. In principle, the projected-strangeness-free form (3.77) has the form (3.103) with

$$\hat{E}(x,\hat{u}) = S^*(x,\hat{u})\widetilde{E}(x,\hat{u}) \quad \text{and} \quad \hat{k}(x,\hat{u}) = S^*(x,\hat{u})\widetilde{k}(x,\hat{u}), \tag{3.105}$$

where $\widetilde{E} \in \mathcal{C}(\mathbb{X} \times \hat{\mathbb{U}}, \mathbb{R}^{\widetilde{m},n})$ and $\widetilde{k} \in \mathcal{C}(\mathbb{X} \times \hat{\mathbb{U}}, \mathbb{R}^{\widetilde{m}})$ are given from $\widetilde{E}(x,\hat{u})\dot{x} = \widetilde{k}(x,\hat{u})$ corresponding to the (complete) minimal reduced derivative array (3.68) and $S^*(x,\hat{u})$ corresponds to the selectors, i.e.,

$$S^*(x,\hat{u}) = \begin{bmatrix} S_D(x,\hat{u}) & 0 \\ 0 & S_C(x,\hat{u}) \end{bmatrix}$$

defined in Definitions 3.5.43 and 3.5.44. Furthermore, in Remark 3.5.46b it is mentioned that the selectors can be chosen piecewise constant such that we will consider the special case that S^* of size $\hat{m} \times \widetilde{m}$ is independent of x and u.
In the numerical integration of quasi-linear DAEs (3.23), and in particular, of the model equations of multibody systems (see Chapter 4), by use of the regularization approach which is introduced in Sections 3.5.3 and 4.6.2.3, the regularized DAEs have the form (3.104) with $\hat{E}_D(x,\hat{u}) = S_D(x,\hat{u})\widetilde{E}_D(x,u)$, $\hat{k}_D(x,\hat{u}) = S_D(x,\hat{u})\widetilde{k}_D(x,\hat{u})$, and $\hat{k}_C(x,\hat{u}) = S_C(x,\hat{u})\widetilde{k}_C(x,\hat{u})$, where $\widetilde{E}_D \in \mathcal{C}(\mathbb{X} \times \hat{\mathbb{U}}, \mathbb{R}^{\widetilde{m}_D,n})$, $\widetilde{k}_D \in \mathcal{C}(\mathbb{X} \times \hat{\mathbb{U}}, \mathbb{R}^{\widetilde{m}_D})$, and $\widetilde{k}_C \in \mathcal{C}(\mathbb{X} \times \hat{\mathbb{U}}, \mathbb{R}^{\widetilde{m}_C})$, are given from $\widetilde{E}(x,\hat{u})\dot{x} = \widetilde{k}(x,\hat{u})$ corresponding to the minimal reduced derivative array (3.68) and $\widetilde{S}_D \in \mathcal{C}(\mathbb{X} \times \hat{\mathbb{U}}, \mathbb{R}^{\hat{m}_D,\widetilde{m}_D})$, $\widetilde{S}_C \in \mathcal{C}(\mathbb{X} \times \hat{\mathbb{U}}, \mathbb{R}^{\hat{m}_C,\widetilde{m}_C})$, with $\hat{m}_D, \hat{m}_C, \widetilde{m}_D, \widetilde{m}_C, n \in \mathbb{N}_0$, $\mathbb{X} \subset \mathbb{R}^n$, $\hat{\mathbb{U}} \subset \mathbb{R}^{n_{\hat{u}}}$.
We will focus, in particular, on the class of *selected semi-implicit DAEs* of the form

$$\begin{bmatrix} S_D(x,\hat{u})\widetilde{E}_D(x,u) \\ 0 \end{bmatrix} \dot{x} = \begin{bmatrix} S_D(x,\hat{u})\widetilde{k}_D(x,\hat{u}) \\ S_C(x,\hat{u})\widetilde{k}_C(x,\hat{u}) \end{bmatrix}, \tag{3.106}$$

which arise in the regularization process introduced in Section 3.5.3 of the projected-strangeness-free form with selected constraints (3.78) of quasi-linear DAEs (3.77).

3.5.4.1 Discretization

The discretization of quasi-linear DAEs of the form (3.103) via an s-stage Runge-Kutta method can be defined by a Butcher-tableau given in Table 3.1, where $A = [a_{ij}]_{i,j=1,\ldots,s} \in \mathbb{R}^{s,s}$ denotes the *Runge-Kutta matrix*, $b = [b_i]_{i=1,\ldots,s} \in \mathbb{R}^s$ denotes the *weight vector*, and $c = [c_i]_{i=1,\ldots,s} \in \mathbb{R}^s$ denotes the *node vector*, with

$$c_i = \sum_{j=1}^{s} a_{ij}. \tag{3.107}$$

We assume in the following considerations that the Runge-Kutta matrix is invertible and we are interested in a particular Runge-Kutta step from t_0 to $t_1 = t_0 + h$. We assume, that the initial values at t_0 are consistently given by x_0, i.e., $(x_0, \hat{u}(t_0)) \in \mathbb{M}$,

c_1	a_{11}	$\cdots$	a_{1s}
$\vdots$	$\vdots$	$\ddots$	$\vdots$
c_s	a_{s1}	$\cdots$	a_{ss}
	b_1	$\cdots$	b_s

Table 3.1: Butcher tableau for implicit Runge-Kutta methods

see Definition 3.5.38. The discretization of the quasi-linear DAE (3.103) yields the nonlinear algebraic system

$$\begin{aligned} \hat{E}(X_i, \hat{u}(t_0 + c_i h))X_i' &= \hat{k}(X_i, \hat{u}(t_0 + c_i h)), \quad i = 1, ..., s, \\ X_i &= x_0 + h \sum_{j=1}^{s} a_{ij} X_j', \qquad i = 1, ..., s. \end{aligned} \tag{3.108}$$

This system determines the *Runge-Kutta stages* $X_i \in \mathbb{R}^n$ and $X_i' \in \mathbb{R}^n$, $i = 1, ..., s$, which approximate the solution $x(t_0+c_i h)$ and its derivative $\dot{x}(t_0+c_i h)$, respectively, see [79]. The approximation x_1 of the solution $x(t_1)$ at t_1 is to be determined by

$$x_1 = x_0 + h \sum_{j=1}^{s} b_j X_j'. \tag{3.109}$$

By introducing the *shifted Runge-Kutta stages* $Y_i = X_i - x_0$ and $Y_i' = X_i'$, which is preferable in numerical implementations because of the smaller absolute values, we get

$$\begin{aligned} \hat{E}(x_0 + Y_i, \hat{u}(t_0 + c_i h))Y_i' &= \hat{k}(x_0 + Y_i, \hat{u}(t_0 + c_i h)), \quad i = 1, ..., s, \\ Y_i &= h \sum_{j=1}^{s} a_{ij} Y_j', \qquad\qquad\qquad\quad i = 1, ..., s, \\ x_1 &= x_0 + h \sum_{j=1}^{s} b_j Y_j'. \end{aligned}$$

Using the Kronecker product, see Definition A.2.19, and introducing

$$\hat{\mathfrak{E}}(\mathfrak{Y}, \hat{\mathfrak{U}}) = \operatorname{diag}(\hat{E}_1, ..., \hat{E}_s), \tag{3.110}$$

$$\hat{\mathfrak{k}}(\mathfrak{Y}, \hat{\mathfrak{U}}) = [\hat{k}_1^T, ..., \hat{k}_s^T]^T, \tag{3.111}$$

where

$$\begin{aligned} \hat{E}_i &= \hat{E}(x_0 + Y_i, \hat{u}(t_0 + c_i h)), \ i = 1, ..., s, \\ \hat{k}_i &= \hat{k}(x_0 + Y_i, \hat{u}(t_0 + c_i h)), \ i = 1, ..., s, \\ \mathfrak{Y} &= [Y_1^T, ..., Y_s^T]^T, \\ \mathfrak{Y}' &= [{Y_1'}^T, ..., {Y_s'}^T]^T, \\ \hat{\mathfrak{U}} &= [\hat{u}^T(t_0 + c_1 h), ..., \hat{u}^T(t_0 + c_s h)]^T, \end{aligned}$$

it follows that

$$\hat{\mathfrak{E}}(\mathfrak{Y}, \hat{\mathfrak{U}})\mathfrak{Y}' = \hat{\mathfrak{k}}(\mathfrak{Y}, \hat{\mathfrak{U}}), \tag{3.112a}$$

$$\mathfrak{Y} = h(A \otimes I_n)\mathfrak{Y}', \tag{3.112b}$$

$$x_1 = x_0 + h(b^T \otimes I_n)\mathfrak{Y}'. \tag{3.112c}$$

Using the properties of the Kronecker product, see Lemma A.2.20, we obtain from the second equation that

$$\mathfrak{Y}' = \frac{1}{h}(A \otimes I_n)^{-1}\mathfrak{Y} = \frac{1}{h}(A^{-1} \otimes I_n)\mathfrak{Y}$$

and hence we get for (3.112) that

$$\begin{aligned} 0 &= \hat{\mathfrak{E}}(\mathfrak{Y}, \hat{\mathfrak{U}})(A^{-1} \otimes I_n)\mathfrak{Y} - h\hat{\mathfrak{k}}(\mathfrak{Y}, \hat{\mathfrak{U}}), && (3.113a) \\ x_1 &= x_0 + (b^T A^{-1} \otimes I_n)\mathfrak{Y}. && (3.113b) \end{aligned}$$

The most expensive part in the determination of the next iterate x_1 is the solution of the *nonlinear stage equation* (3.113a). Note that $\hat{u}(t)$ and therefore also $\hat{\mathfrak{U}}$ are given. Let us denote the right-hand side of (3.113a) by

$$\hat{\mathfrak{G}}(\mathfrak{Y}, \hat{\mathfrak{U}}) = \hat{\mathfrak{E}}(\mathfrak{Y}, \hat{\mathfrak{U}})(A^{-1} \otimes I_n)\mathfrak{Y} - h\hat{\mathfrak{k}}(\mathfrak{Y}, \hat{\mathfrak{U}}). \tag{3.114}$$

Therefore, the nonlinear stage equation corresponds to

$$0 = \hat{\mathfrak{G}}(\mathfrak{Y}, \hat{\mathfrak{U}}).$$

As mentioned above, formally the discretization technique is not restricted to quasi-linear DAEs (3.23) of certain index or to quasi-linear DAEs with $m = n$, rather, formally it works for $m = n$ as well as for $m \neq n$ and for quasi-linear DAEs of arbitrary index, but as discussed in previous sections, higher index DAEs contain hidden constraints which impose additional consistency conditions on the initial values and provoke severe difficulties for the direct numerical integration of DAEs of higher index, see [25, 66, 69, 73, 79, 82, 133, 134, 166]. Therefore, in general, higher index DAEs are not suitable for the direct discretization.

3.5.4.2 Numerical solution of the nonlinear stage equation arising from the discretization

In this section we will discuss the numerical solution of the nonlinear stage equation (3.113a) for general quasi-linear DAEs (3.103), semi-implicit DAEs (3.104), and selected semi-implicit DAEs (3.106). The Newton method, see Algorithm 2.3.18 and [42, 93, 141], applied to the nonlinear stage equation (3.113a) with (3.114) can be summarized as

$$\begin{array}{c} \mathfrak{Y}^0 \text{ given starting value,} \\ \left.\begin{array}{lcl} \text{solve} & \mathfrak{N}(\mathfrak{Y}^k, \hat{\mathfrak{U}})\Delta\mathfrak{Y}^k = -\hat{\mathfrak{G}}(\mathfrak{Y}^k, \hat{\mathfrak{U}}) & \text{for } \Delta\mathfrak{Y}^k, \\ \text{set} & \mathfrak{Y}^{k+1} = \mathfrak{Y}^k + \Delta\mathfrak{Y}^k, & \end{array}\right\} \; k = 0, 1, ..., \end{array}$$

where the Newton iteration matrix $\mathfrak{N}$ is defined as

$$\mathfrak{N}(\mathfrak{Y}, \hat{\mathfrak{U}}) = \frac{\partial}{\partial \mathfrak{Y}}\hat{\mathfrak{G}}(\mathfrak{Y}, \hat{\mathfrak{U}}).$$

The Newton method will be carried out until a certain termination criterion is satisfied, e.g., see Section 5.1.3.5. From (3.113a), the Jacobi[9] matrix of $\hat{\mathfrak{G}}(\mathfrak{Y}, \hat{\mathfrak{U}})$ with respect to $\mathfrak{Y}$ is given by

$$\frac{\partial}{\partial \mathfrak{Y}}\hat{\mathfrak{G}}(\mathfrak{Y}, \hat{\mathfrak{U}}) = (\hat{\mathfrak{E}}(\mathfrak{Y}, \hat{\mathfrak{U}})(A^{-1} \otimes I_n)\mathfrak{Y})_{,\mathfrak{Y}} - h\hat{\mathfrak{k}}_{,\mathfrak{Y}}(\mathfrak{Y}, \hat{\mathfrak{U}}).$$

[9] Carl Gustav Jacob Jacobi (born 1804 in Potsdam, Prussia (now Germany) - died 1851 in Berlin, Germany)

By using a fixed approximation of the Jacobi matrix for several or all steps of the Newton iteration process, we get the simplified Newton method, see Algorithm 2.3.20. Let us use the Jacobi matrix $\mathfrak{N} = \mathfrak{N}(\mathfrak{Y}, \hat{\mathfrak{U}}) \equiv \frac{\partial}{\partial \mathfrak{Y}} \hat{\mathfrak{G}}(\mathfrak{Y}^*, \hat{\mathfrak{U}}^*)$ evaluated at $\mathfrak{Y}^*$ and $\hat{\mathfrak{U}}^*$ which we assume to be a regular point $(X^*, \hat{U}^*)$ with respect to all (hidden) constraints of (3.103), where X^* corresponds to a particular state and $\hat{U}^*$ to a particular control, such that

$$\mathfrak{Y}^* = \begin{bmatrix} Y^* \\ \vdots \\ Y^* \end{bmatrix} \in \mathbb{R}^{sn}, \; Y^* = X^* - x_0, \quad \text{and } \hat{\mathfrak{U}}^* = \begin{bmatrix} \hat{U}^* \\ \vdots \\ \hat{U}^* \end{bmatrix} \in \mathbb{R}^{sn_{\hat{u}}}, \tag{3.115}$$

is such an approximation. The regularity of the point $(X^*, \hat{U}^*)$ with respect to all (hidden) constraints of (3.103) is defined in Definition 2.3.3. It follows that

$$\begin{aligned} \mathfrak{N} &= (\hat{\mathfrak{E}}(\mathfrak{Y}^*, \hat{\mathfrak{U}}^*)(A^{-1} \otimes I_n)\mathfrak{Y}^*)_{,\mathfrak{Y}} - h\hat{\mathfrak{k}}_{,\mathfrak{Y}}(\mathfrak{Y}^*, \hat{\mathfrak{U}}^*) \\ &= (A^{-1} \otimes (\hat{E}^*_{,x}[\![Y^*]\!] + \hat{E}^*)) - h(I_s \otimes \hat{k}^*_{,x}), \end{aligned} \tag{3.116}$$

with $\hat{E}^* = \hat{E}(X^*, U^*)$, $\hat{E}^*_{,x} = \hat{E}_{,x}(X^*, U^*)$, $\hat{k}^* = \hat{k}(X^*, U^*)$, and $\hat{k}^*_{,x} = \hat{k}_{,x}(X^*, U^*)$. If the leading matrix $\hat{E}(x, u)$ is independent of x such that $\hat{E}(x, u) \equiv \hat{E}(u)$, i.e., $\hat{E}_{,x}(u) = 0$, or in the case of $X^* = x_0$, i.e., $Y^* = X^* - x_0 = 0$, we obtain the particular approximation

$$\mathfrak{N}_0 = (A^{-1} \otimes \hat{E}^*) - h(I_s \otimes \hat{k}^*_{,x}).$$

From (3.113a) and by use of (3.116) we get a simplified Newton method in the form

$$\left.\begin{array}{c} \mathfrak{Y}^0 \text{ given starting value,} \\ \text{solve } \big((A^{-1} \otimes (\hat{E}^*_{,x}[\![Y^*]\!] + \hat{E}^*)) - h(I_s \otimes \hat{k}^*_{,x})\big)\Delta\mathfrak{Y}^k = -\hat{\mathfrak{G}}(\mathfrak{Y}^k, \hat{\mathfrak{U}}) \text{ for } \Delta\mathfrak{Y}^k, \\ \text{set } \mathfrak{Y}^{k+1} = \mathfrak{Y}^k + \Delta\mathfrak{Y}^k, \end{array}\right\} k = 0, 1, \ldots . \tag{3.117}$$

Here, the equation (3.117) corresponds to an $s\hat{m} \times sn$ dimensional linear system which has to be solved in every Newton iteration step. Let us follow an approach proposed by Butcher and introduced in [27] to decouple the linear system in several smaller subsystems to reduce the amount of computation. The approach uses a similarity transformation of the inverse $A^{-1} \in \mathbb{R}^{s,s}$ of the Runge-Kutta matrix such that

$$T^{-1}A^{-1}T = \Sigma \;\text{ or equivalently }\; A^{-1} = T\Sigma T^{-1}, \tag{3.118}$$

where $T \in \mathbb{R}^{s,s}$ is an invertible appropriate transformation matrix with

$$T = \begin{bmatrix} \tau_1 \\ \vdots \\ \tau_s \end{bmatrix} \in \mathbb{R}^{s,s} \;\text{ and }\; T^{-1} = \begin{bmatrix} \bar{\tau}_1 \\ \vdots \\ \bar{\tau}_s \end{bmatrix} \in \mathbb{R}^{s,s},$$

$\tau_i, \bar{\tau}_i \in \mathbb{R}^{1,s}$. The matrix $\Sigma \in \mathbb{R}^{s,s}$ should be of a convenient structure such that the amount of computation in solving the linear system (3.117) will be reduced. It would be optimal if Σ were a diagonal matrix, but in the real case the best we can expect is a real Schur[10] form which is upper block triangular with diagonal blocks of size 1×1 or 2×2, see [62].
The linear equation (3.117) in the Newton iteration then becomes

$$\big((T\Sigma T^{-1} \otimes (\hat{E}^*_{,x}[\![Y^*]\!] + \hat{E}^*)) - h(TT^{-1} \otimes \hat{k}^*_{,x})\big)\Delta\mathfrak{Y}^k = -\hat{\mathfrak{G}}(\mathfrak{Y}^k, \hat{\mathfrak{U}})$$

[10] Issai Schur (born 1875 in Mogilyov, Mogilyov province, Russian Empire (now Belarus) - died 1941 in Tel Aviv, Palestine (now Israel))

which is equivalent to

$$\begin{aligned}\big((\Sigma\otimes(\hat{E}^*_{,x}[\![Y^*]\!]+\hat{E}^*))-h(I_s\otimes\hat{k}^*_{,x})\big)(T^{-1}\otimes I_n)\Delta\mathfrak{Y}^k \\ =-(T^{-1}\otimes I_n)\hat{\mathfrak{G}}(\mathfrak{Y}^k,\hat{\mathfrak{U}}).\end{aligned}\tag{3.119}$$

By defining the *transformed Runge-Kutta stages* $\mathfrak{Z}$

$$\mathfrak{Z}=(T^{-1}\otimes I_n)\mathfrak{Y}\ \text{ or equivalently }\ \mathfrak{Y}=(T\otimes I_n)\mathfrak{Z}$$

and

$$\Delta\mathfrak{Z}=(T^{-1}\otimes I_n)\Delta\mathfrak{Y}\ \text{ or equivalently }\ \Delta\mathfrak{Y}=(T\otimes I_n)\Delta\mathfrak{Z}$$

with

$$\mathfrak{Z}=\begin{bmatrix}Z_1\\ \vdots\\ Z_s\end{bmatrix},\ \Delta\mathfrak{Z}=\begin{bmatrix}\Delta Z_1\\ \vdots\\ \Delta Z_s\end{bmatrix},$$

we obtain from (3.119) and (3.114) that

$$\begin{aligned}\big((\Sigma\otimes(\hat{E}^*_{,x}[\![Y^*]\!]+\hat{E}^*))-h(I_s\otimes\hat{k}^*_{,x})\big)\Delta\mathfrak{Z}^k \\ =-(T^{-1}\otimes I_n)\hat{\mathfrak{G}}((T\otimes I_n)\mathfrak{Z}^k,\hat{\mathfrak{U}})\end{aligned}\tag{3.120a}$$

with

$$\hat{\mathfrak{G}}((T\otimes I_n)\mathfrak{Z}^k,\hat{\mathfrak{U}})=\hat{\mathfrak{E}}((T\otimes I_n)\mathfrak{Z}^k,\hat{\mathfrak{U}})(A^{-1}T\otimes I_n)\mathfrak{Z}^k-h\hat{\mathfrak{k}}((T\otimes I_n)\mathfrak{Z}^k,\hat{\mathfrak{U}}).\tag{3.120b}$$

From (3.110) and (3.111) it follows that

$$\begin{aligned}&\hat{\mathfrak{G}}((T\otimes I_n)\mathfrak{Z}^k,\hat{\mathfrak{U}})\\ &=\begin{bmatrix}\hat{E}(x_0+(\tau_1\otimes I_n)\mathfrak{Z}^k,\hat{u}(t_0+c_1h))&&\\ &\ddots&\\ &&\hat{E}(x_0+(\tau_s\otimes I_n)\mathfrak{Z}^k,\hat{u}(t_0+c_sh))\end{bmatrix}(A^{-1}T\otimes I_n)\mathfrak{Z}^k\\ &-h\begin{bmatrix}\hat{k}(x_0+(\tau_1\otimes I_n)\mathfrak{Z}^k,\hat{u}(t_0+c_1h))\\ \vdots\\ \hat{k}(x_0+(\tau_s\otimes I_n)\mathfrak{Z}^k,\hat{u}(t_0+c_sh))\end{bmatrix}.\end{aligned}\tag{3.121}$$

Let us assume that the inverse Runge-Kutta matrix A^{-1} and therefore, the matrix Σ in (3.118) are diagonalizable over $\mathbb{C}$, which corresponds to

$$\Sigma=\operatorname{diag}(\gamma_1,...,\gamma_{n_R},\Gamma_1,...,\Gamma_{n_I})\ \text{with}\ \Gamma_i=\begin{bmatrix}\alpha_i&-\beta_i\\ \beta_i&\alpha_i\end{bmatrix},\ i=1,...,n_I,\tag{3.122}$$

where $\gamma_i\in\mathbb{R}$ for $i=1,...,n_R$, $\alpha_i,\beta_i\in\mathbb{R}$ for $i=1,...,n_I$, and $n_R+2n_I=s$. Then, the linear system (3.120a) is decoupled into n_R subsystems

$$\big(\gamma_i(\hat{E}^*_{,x}[\![Y^*]\!]+\hat{E}^*)-h\hat{k}^*_{,x}\big)\Delta Z^k_i=-(\bar{\tau}_i\otimes I_n)\hat{\mathfrak{G}}((T\otimes I_n)\mathfrak{Z}^k,\hat{\mathfrak{U}}),\tag{3.123}$$

for $i=1,...,n_R$ of dimension $\hat{m}\times n$ and into n_I subsystems

$$\begin{aligned}&\begin{bmatrix}\alpha_i(\hat{E}^*_{,x}[\![Y^*]\!]+\hat{E}^*)-h\hat{k}^*_{,x}&-\beta_i(\hat{E}^*_{,x}[\![Y^*]\!]+\hat{E}^*)\\ \beta_i(\hat{E}^*_{,x}[\![Y^*]\!]+\hat{E}^*)&\alpha_i(\hat{E}^*_{,x}[\![Y^*]\!]+\hat{E}^*)-h\hat{k}^*_{,x}\end{bmatrix}\begin{bmatrix}\Delta Z^k_{n_R+2i-1}\\ \Delta Z^k_{n_R+2i}\end{bmatrix}\\ &=-(\begin{bmatrix}\bar{\tau}_{n_R+2i-1}\\ \bar{\tau}_{n_R+2i}\end{bmatrix}\otimes I_n)\hat{\mathfrak{G}}((T\otimes I_n)\mathfrak{Z}^k,\hat{\mathfrak{U}})\end{aligned}\tag{3.124}$$

for $i = 1, ..., n_I$ of dimension $2\hat{m} \times 2n$. From the nonsingularity of the Runge-Kutta matrix A we have $\gamma_i \neq 0$ for $i = 1, ..., n_R$ and $\alpha_i^2 + \beta_i^2 \neq 0$ for $i = 1, ..., n_I$. Then the linear system (3.117) of dimension $s\hat{m} \times sn$ is decoupled into n_R linear systems (3.123) of dimension $\hat{m} \times n$ and n_I linear systems (3.124) of dimension $2\hat{m} \times 2n$. Furthermore, apart from the coefficients α_i, β_i, and γ_i the leading matrix consists of the same submatrices $(\hat{E}^*_{,x}[\![Y^*]\!] + \hat{E}^*)$ and $\hat{k}^*_{,x}$ for all systems (3.123) for $i = 1, ..., n_R$ and for all systems (3.124) for $i = 1, ..., n_I$. This fact is very convenient with respect to the amount of computation needed for the solution of these linear systems and with respect to the amount of storage, since only these submatrices have to be stored.

Remark 3.5.62 It is important to note that all manipulations of the nonlinear stage equation may be done in advance, i.e., before the implementation of the algorithm. In particular, the entries of the matrices A, A^{-1}, T, T^{-1}, Σ, and $A^{-1}T$ are independent of the considered DAE. They only depend on the used Runge-Kutta method such that they can be determined and stored or implemented independent of the integrated DAE. □

In the case of the discretization of the semi-implicit DAE of the form (3.104) we get the decoupled subsystems (3.123) and (3.124) in the following special form such that we have n_R subsystems

$$\begin{bmatrix} \gamma_i \hat{D} + h\hat{B} \\ \hat{C} \end{bmatrix} \xi_i = \begin{bmatrix} \hat{b}_{Di} \\ \hat{b}_{Ci} \end{bmatrix} \tag{3.125a}$$

for $i = 1, ..., n_R$ of dimension $\hat{m} \times n$ with $\xi_i = \Delta Z_i^k$ and

$$\hat{D} = (\hat{E}^*_{D,x}[\![Y^*]\!] + \hat{E}^*_D), \qquad \hat{B} = -\hat{k}^*_{D,x}, \qquad \hat{C} = -\hat{k}^*_{C,x}, \tag{3.125b}$$

and

$$\hat{b}_{Di} = -(\bar{\tau}_i \otimes I_n)\hat{\mathfrak{G}}_D((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}), \tag{3.125c}$$

$$\hat{b}_{Ci} = -\frac{1}{h}(\bar{\tau}_i \otimes I_n)\hat{\mathfrak{G}}_C((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}), \tag{3.125d}$$

where

$$\hat{\mathfrak{G}}_D((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}) = \Big(\hat{\mathfrak{E}}_D((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}})(A^{-1}T \otimes I_n)\mathfrak{Z}^k - h\hat{\mathfrak{k}}_D((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}})\Big),$$
$$\hat{\mathfrak{G}}_C((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}) = \Big(-h\hat{\mathfrak{k}}_C((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}})\Big),$$

$$\hat{\mathfrak{E}}_D((T \otimes I_n)\mathfrak{Z}, \hat{\mathfrak{U}}) = \mathrm{diag}(\hat{E}_{D1}, ..., \hat{E}_{Ds}), \quad \hat{E}_{Di} = \hat{E}_D(x_0 + (\tau_i \otimes I_n)\mathfrak{Z}, \hat{u}(t_0 + c_i h)),$$
$$\hat{\mathfrak{k}}_D((T \otimes I_n)\mathfrak{Z}, \hat{\mathfrak{U}}) = [\hat{k}_{D1}^T, ..., \hat{k}_{Ds}^T]^T, \quad \hat{k}_{Di} = \hat{k}_D(x_0 + (\tau_i \otimes I_n)\mathfrak{Z}, \hat{u}(t_0 + c_i h)),$$
$$\hat{\mathfrak{k}}_C((T \otimes I_n)\mathfrak{Z}, \hat{\mathfrak{U}}) = [\hat{k}_{C1}^T, ..., \hat{k}_{Cs}^T]^T, \quad \hat{k}_{Ci} = \hat{k}_C(x_0 + (\tau_i \otimes I_n)\mathfrak{Z}, \hat{u}(t_0 + c_i h)).$$

Note that the second part of this linear system is scaled by $1/h$, as suggested in Remark 3.5.61 and [136]. From this scaling it follows that the leading matrix in (3.125) tends to

$$\begin{bmatrix} \gamma_i \hat{D} \\ \hat{C} \end{bmatrix}$$

if $h \to 0$ and keeps its full rank, in the case of regular systems it remains nonsingular. Without scaling, the lower part would also tend to 0 and the condition number would increase for small step sizes.
Furthermore, from (3.124) we have the n_I subsystems

$$\left[\begin{array}{c|c} \alpha_i \hat{D} + h\hat{B} & -\beta_i \hat{D} \\ \hat{C} & 0 \\ \hline \beta_i \hat{D} & \alpha_i \hat{D} + h\hat{B} \\ 0 & \hat{C} \end{array}\right] \begin{bmatrix} \xi_{1i} \\ \xi_{2i} \end{bmatrix} = \left[\begin{array}{c} \hat{b}_{1Di} \\ \hat{b}_{1Ci} \\ \hline \hat{b}_{2Di} \\ \hat{b}_{2Ci} \end{array}\right] \tag{3.126a}$$

for $i = 1, ..., n_I$ of dimension $2\hat{m} \times 2n$ with $\xi_{1i} = \Delta Z^k_{n_R+2i-1}$, $\xi_{2i} = \Delta Z^k_{n_R+2i}$, $\hat{D}$, $\hat{B}$, and $\hat{C}$ as in (3.125b) and

$$\hat{b}_{1Di} = -(\bar{\tau}_{n_R+2i-1} \otimes I_n)\hat{\mathfrak{G}}_D((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}), \tag{3.126b}$$

$$\hat{b}_{1Ci} = -\frac{1}{h}(\bar{\tau}_{n_R+2i-1} \otimes I_n)\hat{\mathfrak{G}}_C((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}), \tag{3.126c}$$

$$\hat{b}_{2Di} = -(\bar{\tau}_{n_R+2i} \otimes I_n)\hat{\mathfrak{G}}_D((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}), \tag{3.126d}$$

$$\hat{b}_{2Ci} = -\frac{1}{h}(\bar{\tau}_{n_R+2i} \otimes I_n)\hat{\mathfrak{G}}_C((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}). \tag{3.126e}$$

As mentioned above we will consider the numerical integration of quasi-linear DAEs in combination with the regularization technique developed in Section 3.5.3. This regularization technique yields a selected quasi-linear DAE (3.103) with (3.105) or in particular, a selected semi-implicit DAE (3.106) which we will investigate in the following. Regarding (3.105), with a constant selector S^* for the whole interval $[t_0, t_0 + h]$ from (3.121) we get

$$\hat{\mathfrak{G}}((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}) = (I_s \otimes S^*)\widetilde{\mathfrak{G}}((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}) \tag{3.127a}$$

with

$$\begin{aligned} &\widetilde{\mathfrak{G}}((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}) \\ &\quad = \widetilde{\mathfrak{E}}((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}})(A^{-1}T \otimes I_n)\mathfrak{Z}^k - h\widetilde{\mathfrak{k}}((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}), \end{aligned} \tag{3.127b}$$

and

$$\begin{aligned} \widetilde{\mathfrak{E}}((T \otimes I_n)\mathfrak{Z}, \hat{\mathfrak{U}}) &= \operatorname{diag}(\widetilde{E}_1, ..., \widetilde{E}_s), \\ \widetilde{\mathfrak{k}}((T \otimes I_n)\mathfrak{Z}, \hat{\mathfrak{U}}) &= [\widetilde{k}_1^T, ..., \widetilde{k}_s^T]^T, \\ \widetilde{E}_i &= \widetilde{E}(x_0 + (\tau_i \otimes I_n)\mathfrak{Z}, \hat{u}(t_0 + c_i h)), \\ \widetilde{k}_i &= \widetilde{k}(x_0 + (\tau_i \otimes I_n)\mathfrak{Z}, \hat{u}(t_0 + c_i h)). \end{aligned}$$

Hence, the subsystem (3.123) has the selected form

$$S^*\big(\gamma_i(\widetilde{E}^*_{,x}[\![Y^*]\!] + \widetilde{E}^*) - h\widetilde{k}^*_{,x}\big)\Delta Z^k_i = -S^*(\bar{\tau}_i \otimes I_n)\widetilde{\mathfrak{G}}((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}), \tag{3.128}$$

with $\widetilde{E}^* = \widetilde{E}(X^*, U^*)$, $\widetilde{E}^*_{,x} = \widetilde{E}_{,x}(X^*, U^*)$, $\widetilde{k}^* = \widetilde{k}(X^*, U^*)$, and $\widetilde{k}^*_{,x} = \widetilde{k}_{,x}(X^*, U^*)$. Furthermore, it follows from (3.127a) that the subsystem (3.124) has the selected form

$$\begin{aligned} &\begin{bmatrix} S^* & 0 \\ 0 & S^* \end{bmatrix}\begin{bmatrix} \alpha_i(\widetilde{E}^*_{,x}[\![Y^*]\!]+\widetilde{E}^*)-h\widetilde{k}^*_{,x} & -\beta_i(\widetilde{E}^*_{,x}[\![Y^*]\!]+\widetilde{E}^*) \\ \beta_i(\widetilde{E}^*_{,x}[\![Y^*]\!]+\widetilde{E}^*) & \alpha_i(\widetilde{E}^*_{,x}[\![Y^*]\!]+\widetilde{E}^*)-h\widetilde{k}^*_{,x} \end{bmatrix}\begin{bmatrix} \Delta Z^k_{n_R+2i-1} \\ \Delta Z^k_{n_R+2i} \end{bmatrix} \\ &\quad = -\begin{bmatrix} S^* & 0 \\ 0 & S^* \end{bmatrix}\Big(\begin{bmatrix} \bar{\tau}_{n_R+2i-1} \\ \bar{\tau}_{n_R+2i} \end{bmatrix} \otimes I_n\Big)\widetilde{\mathfrak{G}}((T \otimes I_m)\mathfrak{Z}^k, \hat{\mathfrak{U}}) \end{aligned} \tag{3.129}$$

with $\widetilde{\mathfrak{G}}((T \otimes I_m)\mathfrak{Z}, \hat{\mathfrak{U}})$ as in (3.127b).
Regarding the selected semi-implicit DAE (3.106) also let us assume that $S_D(x, u)$ and $S_C(x, u)$ are piecewise constant and, in particular, constant on the whole interval $[t_0, t_0+h]$, i.e., $S_D(x, u) = S^*_D$ and $S_C(x, u) = S^*_C$, see Remark 3.5.46. Therefore, from (3.128) for $i = 1, ..., n_R$ we get the linear systems (3.125) in the form

$$\begin{bmatrix} \gamma_i S^*_D\widetilde{D} + hS^*_D\widetilde{B} \\ S^*_C\widetilde{C} \end{bmatrix}\xi_i = \begin{bmatrix} S^*_D\widetilde{b}_{Di} \\ S^*_C\widetilde{b}_{Ci} \end{bmatrix} \tag{3.130a}$$

with $\xi_i = \Delta Z_i^k$ and

$$\widetilde{D} = (\widetilde{E}^*_{D,x}[\![Y^*]\!] + \widetilde{E}^*_D), \qquad \widetilde{B} = -\widetilde{k}^*_{D,x}, \qquad \widetilde{C} = -\widetilde{k}^*_{C,x}, \tag{3.130b}$$

and

$$\widetilde{b}_{Di} = -(\bar{\tau}_i \otimes I_n)\widetilde{\mathfrak{G}}_D((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}), \tag{3.130c}$$

$$\widetilde{b}_{Ci} = -\frac{1}{h}(\bar{\tau}_i \otimes I_n)\widetilde{\mathfrak{G}}_C((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}), \tag{3.130d}$$

with

$$\begin{aligned}
&\widetilde{\mathfrak{G}}_D((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}) = \Big(\widetilde{\mathfrak{E}}_D((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}})(A^{-1}T \otimes I_n)\mathfrak{Z}^k - h\widetilde{\mathfrak{k}}_D((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}})\Big),\\
&\widetilde{\mathfrak{G}}_C((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}) = \Big(- h\widetilde{\mathfrak{k}}_C((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}})\Big),\\
&\widetilde{\mathfrak{E}}_D((T \otimes I_n)\mathfrak{Z}, \hat{\mathfrak{U}}) = \mathrm{diag}(\widetilde{E}_{D1}, ..., \widetilde{E}_{Ds}), \quad \widetilde{E}_{Di} = \widetilde{E}_D(x_0 + (\tau_i \otimes I_n)\mathfrak{Z}, \hat{u}(t_0 + c_i h)),\\
&\widetilde{\mathfrak{k}}_D((T \otimes I_n)\mathfrak{Z}, \hat{\mathfrak{U}}) = [\widetilde{k}^T_{D1}, ..., \widetilde{k}^T_{Ds}]^T, \qquad \widetilde{k}_{Di} = \widetilde{k}_D(x_0 + (\tau_i \otimes I_n)\mathfrak{Z}, \hat{u}(t_0 + c_i h)),\\
&\widetilde{\mathfrak{k}}_C((T \otimes I_n)\mathfrak{Z}, \hat{\mathfrak{U}}) = [\widetilde{k}^T_{C1}, ..., \widetilde{k}^T_{Cs}]^T, \qquad \widetilde{k}_{Ci} = \widetilde{k}_C(x_0 + (\tau_i \otimes I_n)\mathfrak{Z}, \hat{u}(t_0 + c_i h)).
\end{aligned}$$

Note that again the second part of this linear system is scaled by $1/h$, as suggested in Remark 3.5.61 and [136]. Furthermore, from (3.129) we get the linear system

$$\left[\begin{array}{c|c} \alpha_i S^*_D\widetilde{D} + hS^*_D\widetilde{B} & -\beta_i S^*_D\widetilde{D} \\ S^*_C\widetilde{C} & 0 \\ \hline \beta_i S^*_D\widetilde{D} & \alpha_i S^*_D\widetilde{D} + hS^*_D\widetilde{B} \\ 0 & S^*_C\widetilde{C} \end{array}\right] \left[\begin{array}{c} \xi_{1i} \\ \xi_{2i} \end{array}\right] = \left[\begin{array}{c} S^*_D\widetilde{b}_{1Di} \\ S^*_C\widetilde{b}_{1Ci} \\ \hline S^*_D\widetilde{b}_{2Di} \\ S^*_C\widetilde{b}_{2Ci} \end{array}\right] \tag{3.131a}$$

for $i = 1, ..., n_I$ with $\xi_{1i} = \Delta Z^k_{n_R+2i-1}$, $\xi_{2i} = \Delta Z^k_{n_R+2i}$, $\widetilde{D}$, $\widetilde{B}$, and $\widetilde{C}$ as in (3.130b) and

$$\widetilde{b}_{1Di} = -(\bar{\tau}_{n_R+2i-1} \otimes I_n)\widetilde{\mathfrak{G}}_D((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}), \tag{3.131b}$$

$$\widetilde{b}_{1Ci} = -\frac{1}{h}(\bar{\tau}_{n_R+2i-1} \otimes I_n)\widetilde{\mathfrak{G}}_C((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}), \tag{3.131c}$$

$$\widetilde{b}_{2Di} = -(\bar{\tau}_{n_R+2i} \otimes I_n)\widetilde{\mathfrak{G}}_D((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}), \tag{3.131d}$$

$$\widetilde{b}_{2Ci} = -\frac{1}{h}(\bar{\tau}_{n_R+2i} \otimes I_n)\widetilde{\mathfrak{G}}_C((T \otimes I_n)\mathfrak{Z}^k, \hat{\mathfrak{U}}). \tag{3.131e}$$

Remark 3.5.63 a) By use of Runge-Kutta methods for the discretization of quasi-linear DAEs (3.103), the solution of the $s\hat{m} \times sn$ dimensional linear system (3.117) reduces to the solution of a set of linear systems consisting of n_R systems of the form (3.123) of size $\hat{m} \times n$ and n_I systems of the form (3.124) of size $2\hat{m} \times 2n$.
The discretization of semi-implicit DAEs of the form (3.104) reduces to the solution of a set of n_R linear systems of the form (3.125) of size $\hat{m} \times n$ and n_I linear systems of the form (3.126a) of size $2\hat{m} \times 2n$ which exploit the partitioned structure.
Furthermore, in the special case (3.106), where $S_D(x, \hat{u}) = S^*_D$ and $S_C(x, \hat{u}) = S^*_C$ are constant on the interval $[t_0, t_0 + h]$, this discretization of (3.106) reduces to the solution of a set of n_R linear systems of the form (3.130a) of size $\hat{m} \times n$ and n_I linear systems of the form (3.131a) of size $2\hat{m} \times 2n$ which offer the special partitioned structure.
We will discuss the efficient numerical solution of the latter systems in detail in Section 3.5.4.3.
b) Note, that the matrices in the linear systems (3.130a) and (3.131a) contain the same submatrices $\hat{D}$, $\hat{B}$, and $\hat{C}$ independent of i. Merely, the coefficients α_i, β_i, and γ_i and the right-hand sides depend on i. This property can be exploited in the numerical solution of these systems as shown in Section 3.5.4.3. □

Remark 3.5.64 If we use BDF methods [25, 37, 64, 81, 125, 173] as discretization method for the numerical solution of the initial value problem of the quasi-linear DAE (3.103) we obtain the linear systems to be solved in every integration step of the same type as (3.123). Analogously, the discretization of the semi-implicit DAE (3.104) leads to a linear system of type (3.125) and the discretization of the selected semi-implicit DAE (3.106) leads to a linear system of type (3.130a), where in every case coefficients γ_i and the right-hand side are adapted to the used discretization method. □

Remark 3.5.65 Recently, a third class of discretization techniques, the so-called *general linear methods* are widely studied, see [28, 159, 179]. The class of general linear methods forms a generalization of Runge-Kutta methods and of BDF methods and contains both as special discretization techniques. Also if we use general linear methods as discretization method for the numerical solution of the initial value problem of the quasi-linear DAE (3.103) or of the semi-implicit DAE (3.104) or of the selected semi-implicit DAE (3.106) we obtain the linear systems of type (3.123), (3.124) or (3.125), (3.126a) or (3.130a), (3.131a), respectively, to solve in every integration step, where in every case the coefficients α_1, β_1, and γ_i and the right-hand side are adapted to the used discretization method. □

3.5.4.3 Numerical solution of special linear systems arising from discretization methods

In the previous sections we have investigated the discretization of the quasi-linear DAE (3.103) via Runge-Kutta methods. We have seen that the numerical integration of the quasi-linear DAE (3.103) needs the efficient solution of the linear systems (3.123) and (3.124). In the numerical treatment of semi-implicit DAEs (3.104) and, in particular, in the numerical treatment of selected semi-implicit DAEs (3.106), the existing structure can be exploited. This will be discussed in detail in the following. Recall, that in the case of semi-implicit DAEs (3.104) we have to solve linear systems of two different types that contains the same submatrices $\hat{C} \in \mathbb{R}^{\hat{m}_C,n}$ and $\hat{D}, \hat{B} \in \mathbb{R}^{\hat{m}_D,n}$. First, we have to solve linear systems of the form

$$\begin{bmatrix} \hat{C} \\ \gamma\hat{D} + h\hat{B} \end{bmatrix} \xi = \begin{bmatrix} \hat{b}_C \\ \hat{b}_D \end{bmatrix} \tag{3.132}$$

of size $\hat{m} \times n$, for $i = 1, ..., n_R$ and $\gamma = \gamma_i$, $\hat{b}_C = \hat{b}_{Ci} \in \mathbb{R}^{\hat{m}_C}$, $\hat{b}_D = \hat{b}_{Di} \in \mathbb{R}^{\hat{m}_D}$ with $\hat{m}_C + \hat{m}_D = \hat{m}$. This linear system corresponds to (3.125) where we did permute the block rows and omitted the index i for the sake of simplicity. Let us call this type of system *selected linear system of type 1* or *selected system of type 1* in short. Furthermore, let us call the first block equation $\hat{C}\xi = \hat{b}_C$ of (3.132) the *algebraic part* of (3.132) because of its correspondence to the algebraic constraints of the considered DAE (3.104) which restrict the solution to be in the solution manifold $\mathbb{M}$. Furthermore, the second block equation $(\gamma\hat{D} + h\hat{B})\xi = \hat{b}_D$ of (3.132) will be called the *differential part* of (3.132) because of its correspondence to the differential equations of the considered DAE (3.104), which describe the dynamics of the system along the solution manifold $\mathbb{M}$. Secondly, we have to solve linear systems of the form

$$\left[\begin{array}{c|c} \hat{C} & 0 \\ \alpha\hat{D} + h\hat{B} & -\beta\hat{D} \\ \hline 0 & \hat{C} \\ \beta\hat{D} & \alpha\hat{D} + h\hat{B} \end{array}\right] \begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix} = \left[\begin{array}{c} \hat{b}_{1C} \\ \hat{b}_{1D} \\ \hline \hat{b}_{2C} \\ \hat{b}_{2D} \end{array}\right] \tag{3.133}$$

of size $2\hat{m} \times 2n$ for $i = 1, ..., n_I$ and $\alpha = \alpha_i$, $\beta = \beta_i$, $\hat{b}_{1C} = \hat{b}_{1Ci} \in \mathbb{R}^{\hat{m}_C}$, $\hat{b}_{2C} = \hat{b}_{2Ci} \in \mathbb{R}^{\hat{m}_C}$, $\hat{b}_{1D} = \hat{b}_{1Di} \in \mathbb{R}^{\hat{m}_D}$, and $\hat{b}_{2D} = \hat{b}_{2Di} \in \mathbb{R}^{\hat{m}_D}$. This coincides with

(3.126a) when permuting the block rows and omitting the index i. Analogously, let us call this system *selected linear system of type 2* or *selected system of type 2* in short and let us call the first and third block equations $\hat{C}\xi_1 = \hat{b}_{1C}$ and $\hat{C}\xi_2 = \hat{b}_{2C}$ of (3.133) the algebraic part of (3.133). The second and fourth block equations $(\alpha\hat{D}+h\hat{B})\xi_1 - \beta\hat{D}\xi_2 = \hat{b}_{1D}$ and $\beta\hat{D}\xi_1 + (\alpha\hat{D}+h\hat{B})\xi_2 = \hat{b}_{2D}$ of (3.133) will be called the differential part of (3.133).
Recall, that the linear systems (3.132) and (3.133) arise from the discretization of the projected-strangeness-free form with selected constraints (3.78).
In the following we will consider general matrix decompositions of a matrix $T \in \mathbb{R}^{q,l}$ of the form

$$T = Q\bar{T}P$$

with nonsingular matrices $Q \in \mathbb{R}^{q,q}$ and $P \in \mathbb{R}^{l,l}$. We will call them *decomposition* in short. Typical examples are the LU decomposition, the QR decomposition, or the SV decomposition, see [72, 171]. Note that we will not distinguish between the particular transformation matrices regarding the decomposition of the selected linear systems (3.132) and (3.133) even if both systems are of different size. We will denote the transformation matrices by $\hat{Q}$, $\widetilde{Q}$, and $\hat{P}$, and $\widetilde{P}$. From the context it becomes clear which size and properties they have to have.
Let us start our discussion regarding the selected linear system of type 1 (3.132). A decomposition of the matrix in (3.132) leads to the linear system

$$\hat{R}\hat{\zeta} = \hat{b}, \tag{3.134a}$$
$$\xi = \hat{P}\hat{\zeta} \tag{3.134b}$$

with

$$\hat{R} = \begin{bmatrix} \hat{Q}_1 & \hat{Q}_2 \end{bmatrix} \begin{bmatrix} \hat{C} \\ \gamma\hat{D} + h\hat{B} \end{bmatrix} \hat{P}, \tag{3.134c}$$
$$\hat{b} = \begin{bmatrix} \hat{Q}_1 & \hat{Q}_2 \end{bmatrix} \begin{bmatrix} \hat{b}_C \\ \hat{b}_D \end{bmatrix}, \tag{3.134d}$$

where $\hat{Q} = \begin{bmatrix} \hat{Q}_1 & \hat{Q}_2 \end{bmatrix} \in \mathbb{R}^{\hat{m},\hat{m}}$ and $\hat{P} \in \mathbb{R}^{n,n}$ are nonsingular with $\hat{Q}_1 \in \mathbb{R}^{\hat{m},\hat{m}_C}$, $\hat{Q}_2 \in \mathbb{R}^{\hat{m},\hat{m}_D}$. If the decomposition is a LU decomposition or QR decomposition, respectively, then, in particular, $\hat{R} \in \mathbb{R}^{\hat{m},n}$ is an upper triangular matrix or in case of the SV decomposition even a diagonal matrix. We will not restrict the terms "triangular" or "diagonal" to square matrices, but rather we will use these terms also for rectangular matrices. The linear system (3.132) can be solved for ξ by backward substitution of (3.134a) and transformation of $\hat{\zeta}$ to ξ with $\hat{P}$ as in (3.134b). If the linear system (3.132) arises from the discretization of an arbitrary semi-implicit DAE (3.104), the matrix $\hat{R}$ does not necessarily have full rank.
As already mentioned, we will investigate the numerical treatment of quasi-linear DAEs or semi-implicit DAEs as combination of the regularization technique developed in Section 3.5.3 and the discretization technique developed in Section 3.5.4.1. In contrast to the case of discretization of general semi-implicit DAEs (3.104), the matrix in (3.132), and therefore also the matrix $\hat{R}$ in (3.134), have full rank for all sufficiently small h and, in particular, for $h \to 0$, when they arise from the discretization of the projected-strangeness-free form with selected constraints (3.78) of a quasi-linear DAE (3.103). In the case of nonredundant constraints the matrix (3.132) has full rank $\min(m, n)$.
In the case that the regularized DAE has the form (3.78) which corresponds to the selected semi-implicit DAEs (3.106), the discretization discussed in Section 3.5.4.1

leads to the selected linear system of type 1 of the form (3.132) with

$$\hat{D} = S_D^* \widetilde{D}, \;\; \hat{B} = S_D^* \widetilde{B}, \;\; \hat{b}_D = S_D^* \widetilde{b}_D, \;\; \text{and} \;\; \hat{C} = S_C^* \widetilde{C}, \;\; \hat{b}_C = S_C^* \widetilde{b}_C, \quad (3.135)$$

compare with (3.130a). We have $\widetilde{C} \in \mathbb{R}^{\widetilde{m}_C,n}$, $\widetilde{D}, \widetilde{B} \in \mathbb{R}^{\widetilde{m}_D,n}$ and $S_C^* \in \mathbb{R}^{r_C,\widetilde{m}_C}$, $S_D^* \in \mathbb{R}^{r_D,\widetilde{m}_D}$. In particular, from the regularization (3.78) it follows that $\hat{m}_C = r_C$ and $\hat{m}_D = r_D$ with

$$r_C = \operatorname{rank}(\widetilde{C}) \;\; \text{and} \;\; r_D = \operatorname{rank}(\begin{bmatrix} \widetilde{C} \\ \widetilde{D} \end{bmatrix}) - r_C \quad (3.136)$$

and $r_C + r_D \leq n$. With respect to the relations (3.135), the matrix $\hat{R}$ and the right-hand side $\hat{b}$ in the decomposition (3.134) are given by

$$\begin{aligned} \hat{R} &= \begin{bmatrix} \hat{Q}_1 S_C^* & \hat{Q}_2 S_D^* \end{bmatrix} \begin{bmatrix} \widetilde{C} \\ \gamma\widetilde{D} + h\widetilde{B} \end{bmatrix} \hat{P}, \\ \hat{b} &= \begin{bmatrix} \hat{Q}_1 S_C^* & \hat{Q}_2 S_D^* \end{bmatrix} \begin{bmatrix} \widetilde{b}_C \\ \widetilde{b}_D \end{bmatrix}. \end{aligned}$$

This decomposition can also be interpreted as a decomposition of special type of the linear system of the form

$$\begin{bmatrix} \widetilde{C} \\ \gamma\widetilde{D} + h\widetilde{B} \end{bmatrix} \xi = \begin{bmatrix} \widetilde{b}_C \\ \widetilde{b}_D \end{bmatrix} \quad (3.137)$$

of the size $\widetilde{m} \times n$ for $i = 1, ..., n_R$ and $\gamma = \gamma_i$, $\widetilde{b}_C = \widetilde{b}_{Ci} \in \mathbb{R}^{\widetilde{m}_C}$, $\widetilde{b}_D = \widetilde{b}_{Di} \in \mathbb{R}^{\widetilde{m}_D}$, $\widetilde{C} \in \mathbb{R}^{\widetilde{m}_C,n}$, $\widetilde{D}, \widetilde{B} \in \mathbb{R}^{\widetilde{m}_D,n}$, and $\widetilde{m} = \widetilde{m}_C + \widetilde{m}_D$. The linear system (3.137) is called *unselected linear system of type 1*, or *unselected system of type 1* in short.
An analogous observation can be made for the decomposition of the linear system (3.133) which can be interpreted also as a decomposition of special type of the linear system of the form

$$\left[\begin{array}{c|c} \widetilde{C} & 0 \\ \alpha\widetilde{D} + h\widetilde{B} & -\beta\widetilde{D} \\ \hline 0 & \widetilde{C} \\ \beta\widetilde{D} & \alpha\widetilde{D} + h\widetilde{B} \end{array}\right] \begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix} = \left[\begin{array}{c} \widetilde{b}_{1C} \\ \widetilde{b}_{1D} \\ \hline \widetilde{b}_{2C} \\ \widetilde{b}_{2D} \end{array}\right] \quad (3.138)$$

of the size $2\widetilde{m} \times 2n$ for $i = 1, ..., n_I$ and $\alpha = \alpha_i$, $\beta = \beta_i$, $\widetilde{b}_{1C} = \widetilde{b}_{1Ci} \in \mathbb{R}^{\widetilde{m}_C}$, $\widetilde{b}_{2C} = \widetilde{b}_{2Ci} \in \mathbb{R}^{\widetilde{m}_C}$, $\widetilde{b}_{1D} = \widetilde{b}_{1Di} \in \mathbb{R}^{\widetilde{m}_D}$, $\widetilde{b}_{2D} = \widetilde{b}_{2Di} \in \mathbb{R}^{\widetilde{m}_D}$, $\widetilde{C} \in \mathbb{R}^{\widetilde{m}_C,n}$, and $\widetilde{D}, \widetilde{B} \in \mathbb{R}^{\widetilde{m}_D,n}$. Analogously, the linear system (3.138) is called *unselected linear system of type 2*, or *unselected system of type 2* in short.
In the numerical treatment of quasi-linear DAEs (3.103) by use of Runge-Kutta methods described in the Sections 3.5.4.1 and 3.5.4.2 in combination with the regularization to the projected-strangeness-free form with selected constraints (3.78) described in Section 3.5, the linear systems (3.137) and (3.138) arise from the discretization of the complete minimal reduced derivative array (3.66).
On the other hand, an arbitrary decomposition applied directly to the unselected linear system of type 1 (3.137) without respecting the regularization to the projected-strangeness-free form would lead to

$$\begin{bmatrix} \widetilde{R} \\ 0 \end{bmatrix} \widetilde{\zeta} = \begin{bmatrix} \widetilde{b} \\ \widetilde{r} \end{bmatrix}, \quad (3.139a)$$

$$\xi = \widetilde{P}\widetilde{\zeta} \quad (3.139b)$$

with nonsingular $\widetilde{Q} \in \mathbb{R}^{\widetilde{m},\widetilde{m}}$ and $\widetilde{P} \in \mathbb{R}^{n,n}$ such that

$$\begin{bmatrix} \widetilde{R} \\ 0 \end{bmatrix} = \begin{bmatrix} \widetilde{Q}_{11} & \widetilde{Q}_{12} \\ \widetilde{Q}_{21} & \widetilde{Q}_{22} \end{bmatrix} \begin{bmatrix} \widetilde{C} \\ \gamma\widetilde{D} + h\widetilde{B} \end{bmatrix} \widetilde{P}, \tag{3.139c}$$

$$\begin{bmatrix} \widetilde{b} \\ \widetilde{r} \end{bmatrix} = \begin{bmatrix} \widetilde{Q}_{11} & \widetilde{Q}_{12} \\ \widetilde{Q}_{21} & \widetilde{Q}_{22} \end{bmatrix} \begin{bmatrix} \widetilde{b}_C \\ \widetilde{b}_D \end{bmatrix}, \tag{3.139d}$$

$$\widetilde{Q} = \begin{bmatrix} \widetilde{Q}_{11} & \widetilde{Q}_{12} \\ \widetilde{Q}_{21} & \widetilde{Q}_{22} \end{bmatrix} \tag{3.139e}$$

with $\widetilde{Q}_{11} \in \mathbb{R}^{\hat{m}_C+\hat{m}_D,\widetilde{m}_C}$, $\widetilde{Q}_{12} \in \mathbb{R}^{\hat{m}_C+\hat{m}_D,\widetilde{m}_D}$, $\widetilde{R} \in \mathbb{R}^{\hat{m}_C+\hat{m}_D,n}$, and $\widetilde{b} \in \mathbb{R}^{\hat{m}_C+\hat{m}_D}$. In particular, $\widetilde{R}$ has full rank, i.e., $\operatorname{rank}(\widetilde{R}) = \hat{m}_C + \hat{m}_D$ and is an upper triangular matrix in the case of LU decomposition or QR decomposition or in the case of an SV decomposition even a diagonal matrix. The unselected linear system of type 1 (3.137) could then be solved for ξ by backward substitution of the upper block row of (3.139a) and transformation of $\widetilde{\zeta}$ to ξ with $\widetilde{P}$ as in (3.139b).
Similarly, an arbitrary decomposition applied directly to the unselected linear system of type 2 (3.138) without respecting the regularization to the projected-strangeness-free form would lead to

$$\begin{bmatrix} \widetilde{R} \\ 0 \end{bmatrix} \widetilde{\zeta} = \begin{bmatrix} \widetilde{b} \\ \widetilde{r} \end{bmatrix}, \tag{3.140a}$$

$$\begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix} = \begin{bmatrix} \widetilde{P}_1 \\ \widetilde{P}_2 \end{bmatrix} \widetilde{\zeta} \tag{3.140b}$$

with nonsingular $\widetilde{Q} \in \mathbb{R}^{2\widetilde{m},2\widetilde{m}}$ and $\widetilde{P} \in \mathbb{R}^{2n,2n}$,

$$\begin{bmatrix} \widetilde{R} \\ 0 \end{bmatrix} = \begin{bmatrix} \widetilde{Q}_{11} & \widetilde{Q}_{12} & \widetilde{Q}_{13} & \widetilde{Q}_{14} \\ \widetilde{Q}_{21} & \widetilde{Q}_{22} & \widetilde{Q}_{23} & \widetilde{Q}_{24} \end{bmatrix} \left[\begin{array}{c|c} \alpha\widetilde{D} + h\widetilde{B} & -\beta\widetilde{D} \\ \widetilde{C} & 0 \\ \hline \beta\widetilde{D} & \alpha\widetilde{D} + h\widetilde{B} \\ 0 & \widetilde{C} \end{array}\right] \begin{bmatrix} \widetilde{P}_1 \\ \widetilde{P}_2 \end{bmatrix}, \tag{3.140c}$$

$$\begin{bmatrix} \widetilde{b} \\ \widetilde{r} \end{bmatrix} = \begin{bmatrix} \widetilde{Q}_{11} & \widetilde{Q}_{12} & \widetilde{Q}_{13} & \widetilde{Q}_{14} \\ \widetilde{Q}_{21} & \widetilde{Q}_{22} & \widetilde{Q}_{23} & \widetilde{Q}_{24} \end{bmatrix} \left[\begin{array}{c} \widetilde{b}_{1C} \\ \widetilde{b}_{1D} \\ \hline \widetilde{b}_{2C} \\ \widetilde{b}_{2D} \end{array}\right], \tag{3.140d}$$

$$\widetilde{Q} = \begin{bmatrix} \widetilde{Q}_{11} & \widetilde{Q}_{12} & \widetilde{Q}_{13} & \widetilde{Q}_{14} \\ \widetilde{Q}_{21} & \widetilde{Q}_{22} & \widetilde{Q}_{23} & \widetilde{Q}_{24} \end{bmatrix}, \tag{3.140e}$$

$$\widetilde{P} = \begin{bmatrix} \widetilde{P}_1 \\ \widetilde{P}_2 \end{bmatrix} \tag{3.140f}$$

with $\widetilde{Q}_{11} \in \mathbb{R}^{2\hat{m}_C+2\hat{m}_D,\widetilde{m}_C}$, $\widetilde{Q}_{12} \in \mathbb{R}^{2\hat{m}_C+2\hat{m}_D,\widetilde{m}_D}$, $\widetilde{Q}_{13} \in \mathbb{R}^{2\hat{m}_C+2\hat{m}_D,\widetilde{m}_C}$, $\widetilde{Q}_{14} \in \mathbb{R}^{2\hat{m}_C+2\hat{m}_D,\widetilde{m}_D}$, $\widetilde{P}_i \in \mathbb{R}^{n,2n}$, $i = 1, 2$, $\widetilde{R} \in \mathbb{R}^{2\hat{m}_C+2\hat{m}_D,2n}$, and $\widetilde{b} \in \mathbb{R}^{2\hat{m}_C+2\hat{m}_D}$. In particular, $\widetilde{R}$ has full rank, i.e., $\operatorname{rank}(\widetilde{R}) = 2\hat{m}_C + 2\hat{m}_D$ and is an upper triangular matrix in the case of LU decomposition or QR decomposition or in the case of an SV decomposition even a diagonal matrix. The unselected linear system of type 2 (3.138) could then be solved for ξ_1 and ξ_2 by backward substitution of the upper block row of (3.140a) and transformation $\widetilde{\zeta}$ to ξ_1 and ξ_2 with $\widetilde{P}$ as in (3.140b).

Remark 3.5.66 From the solvability of the considered quasi-linear DAE (3.23) the solvability of the minimal reduced derivative array (3.66) follows. Unfortunately, the

discretization of the derivative array causes discretization errors, which leads to an overdetermined system which is in general not solvable, because the discretization of the (hidden) constraints are in general slightly contradictory to the discretization of the differential equations of the quasi-linear DAE (3.23). Here, "slightly" means that in general a solution of the discretized (hidden) constraints causes small residuals in the discretized differential equations. Therefore, the quantities $\widetilde{r}$ in (3.139) and (3.140) corresponds to such residuals caused from discretization errors and we have $||\widetilde{r}|| \ll 1$ for small step sizes h and consistent reference points (X^*, U^*).
Furthermore, the residual $\widetilde{r}$ can be used for the validation of the consistency of the differential-algebraic equations. □

In the following it turns out that an arbitrary decomposition of the unselected systems (3.137) and (3.138) could lead to unexpected effects or even to wrong results as discussed in the following and shown in Example 3.5.77. Therefore, we will now investigate the relation of the decomposition of the selected linear systems (3.132), (3.133) and the decomposition of the unselected linear systems (3.137), (3.138).

Definition 3.5.67 (Discrete kinematic selector) *Let the discretized complete minimal reduced derivative array be given by (3.137) and (3.138). Then a matrix* $\widetilde{S}_C \in \mathbb{R}^{r_C, \widetilde{m}_C}$ *with* $r_C = \operatorname{rank}(\widetilde{C})$ *satisfying*

$$\operatorname{rank}(\widetilde{S}_C \widetilde{C}) = r_C \tag{3.141}$$

is called discrete kinematic selector *of the discretized complete minimal reduced derivative array.*

Definition 3.5.68 (Discrete dynamic selector) *Let the discretized complete minimal reduced derivative array be given by (3.137) and (3.138). Then a matrix* $\widetilde{S}_D \in \mathbb{R}^{r_D, \widetilde{m}_D}$ *with*

$$r_D = \operatorname{rank}\left(\begin{bmatrix} \widetilde{C} \\ \widetilde{D} \end{bmatrix} \right) - \operatorname{rank}(\widetilde{C}) \tag{3.142}$$

satisfying

$$\operatorname{rank}\left(\begin{bmatrix} \widetilde{C} \\ S_D \widetilde{D} \end{bmatrix} \right) = \operatorname{rank}\left(\begin{bmatrix} \widetilde{C} \\ \widetilde{D} \end{bmatrix} \right) \tag{3.143}$$

is called discrete dynamic selector *of the discretized complete minimal reduced derivative array.*

Lemma 3.5.69 *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step* $i = \nu = \nu_c + 1$ *in (3.42) with maximal constraint level* ν_c. *Let* a^{ν_c} *be defined by (3.74b) with* $l = \nu_c$ *and let* $a^{\nu_c} = \text{const}$ *for all* $(x, \mathfrak{u}^{\nu_c}) \in \mathbb{M}$. *Furthermore, let the discretized complete minimal reduced derivative array be given by (3.137) and (3.138). Then every constant kinematic selector* $S_C = S_C(X^*, \mathfrak{u}^{\nu_c *})$ *(see Definition 3.5.43) forms a discrete kinematic selector* $\widetilde{S}_C$ *(see Definition 3.5.67). Conversely, every discrete kinematic selector* $\widetilde{S}_C$ *forms a constant kinematic selector* $S_C = S_C(X^*, \mathfrak{u}^{\nu_c *})$ *in a neighborhood* $\mathbb{S}((X^*, \mathfrak{u}^{\nu_c *}), \epsilon)$ *for sufficiently small* $\epsilon > 0$.

Proof: From the discretization of the complete minimal reduced derivative array (3.66), see also (3.68) and (3.130b), we get

$$\widetilde{C} = \begin{bmatrix} -\widetilde{k}^0_{2,x}(X^*, U^*) \\ \vdots \\ -\widetilde{k}^{\nu_c}_{2,x}(X^*, \mathfrak{u}^{\nu_c *}) \end{bmatrix} \in \mathbb{R}^{\widetilde{m}_C, n} \tag{3.144}$$

with $\widetilde{m}_C = \sum_{i=0}^{\nu_c} m_2^i$. With (3.74b) we get $a^{\nu_c} = r_C$ with $r_C = \operatorname{rank}(\widetilde{C})$. Therefore, a kinematic selector S_C defined in Definition 3.5.43 and a discrete kinematic selector $\widetilde{S}_C$ defined in Definition 3.5.67 have the same size $r_C \times \widetilde{m}_C$. Furthermore, it follows that the conditions (3.75) and (3.141) are equivalent and we get the assertion. □

Lemma 3.5.70 *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu = \nu_c + 1$ in (3.42) with maximal constraint level ν_c. Let a^{ν_c} and d^{ν_c} be defined by (3.74) with $l = \nu_c$ and let $a^{\nu_c} = \text{const}$ and $d^{\nu_c} = \text{const}$ for all $(x, \mathfrak{u}^{\nu_c}) \in \mathbb{M}$. Furthermore, let the discretized complete minimal reduced derivative array be given by (3.137) and (3.138) and let $Y^* = X^* - x_0 = \mathcal{O}(h)$. Then every constant dynamic selector $S_D = S_D(X^*, \mathfrak{u}^{\nu_c *})$ (see Definition 3.5.44) forms a discrete dynamic selector $\widetilde{S}_D$ (see Definition 3.5.68). Conversely, every discrete dynamic selector $\widetilde{S}_D$ forms a constant dynamic selector $S_D = S_D(X^*, \mathfrak{u}^{\nu_c *})$ in a neighborhood $\mathbb{S}((X^*, \mathfrak{u}^{\nu_c *}), \epsilon)$ for sufficiently small $\epsilon > 0$ and sufficiently small h.*

Proof: From the discretization of the minimal reduced derivative array (3.66), see also (3.68) and (3.130b), we get

$$\widetilde{D} = E_{,x}(X^*, U^*)[\![Y^*]\!] + E(X^*, U^*). \tag{3.145}$$

Furthermore, $\widetilde{D}$ depends smoothly on Y^* and for $h \to 0$ we have $Y^* \to 0$. Therefore, from (3.144) and (3.145) we get for sufficiently small h that

$$\operatorname{rank}(\widetilde{D}) = \operatorname{rank}(E(X^*, U^*)) \tag{3.146a}$$

and with (3.144)

$$\operatorname{rank}\left(\begin{bmatrix} \widetilde{D} \\ \widetilde{C} \end{bmatrix}\right) = \operatorname{rank}\left(\begin{bmatrix} E(X^*, U^*) \\ \widetilde{k}_{2,x}^0(X^*, U^*) \\ \vdots \\ \widetilde{k}_{2,x}^{\nu_c}(X^*, \mathfrak{u}^{\nu_c *}) \end{bmatrix}\right). \tag{3.146b}$$

From Definition 3.5.44 it follows with (3.144), (3.142), (3.145), and $a^{\nu_c} = r_C$ (see proof of Lemma 3.5.69) that $d^{\nu_c} = r_D$. Then, from Definition 3.5.44 and from Definition 3.5.68 we get for sufficiently small h that a dynamic selector S_D and a discrete dynamic selector $\widetilde{S}_D$, respectively, have the same size $r_D \times \widetilde{m}_D$. Furthermore, we get from (3.146) for sufficiently small h that the conditions (3.76) and (3.143) are equivalent and we get the assertion. □

Definition 3.5.71 (Index reducing decomposition matrix) *If there exist a discrete kinematic selector $\widetilde{S}_C \in \mathbb{R}^{r_C, \widetilde{m}_C}$ and a discrete dynamic selector $\widetilde{S}_D \in \mathbb{R}^{r_D, \widetilde{m}_D}$ with r_C and r_D defined by (3.136) such that for a nonsingular matrix $\widetilde{Q} \in \mathbb{R}^{\widetilde{m}, \widetilde{m}}$ with*

$$\widetilde{Q} = \begin{bmatrix} \widetilde{Q}_{11} & \widetilde{Q}_{12} \\ \widetilde{Q}_{21} & \widetilde{Q}_{22} \end{bmatrix} \in \mathbb{R}^{\widetilde{m}, \widetilde{m}} \tag{3.147}$$

and $\widetilde{Q}_{11} \in \mathbb{R}^{r_C + r_D, \widetilde{m}_C}$, $\widetilde{Q}_{12} \in \mathbb{R}^{r_C + r_D, \widetilde{m}_D}$, and $\widetilde{m} = \widetilde{m}_C + \widetilde{m}_D$ it holds that with

$$\widetilde{Q}_{11} = \hat{Q}_1 \widetilde{S}_C \quad \text{and} \quad \widetilde{Q}_{12} = \hat{Q}_2 \widetilde{S}_D$$

the matrix

$$\hat{Q} = \left[\begin{array}{cc} \hat{Q}_1 & \hat{Q}_2 \end{array} \right]$$

is nonsingular and for a nonsingular matrix $\widetilde{Q} \in \mathbb{R}^{2\widetilde{m},2\widetilde{m}}$ *with*

$$\widetilde{Q} = \left[\begin{array}{cccc} \widetilde{Q}_{11} & \widetilde{Q}_{12} & \widetilde{Q}_{13} & \widetilde{Q}_{14} \\ \widetilde{Q}_{21} & \widetilde{Q}_{22} & \widetilde{Q}_{23} & \widetilde{Q}_{24} \end{array} \right] \in \mathbb{R}^{2\widetilde{m},2\widetilde{m}} \tag{3.148}$$

and $\widetilde{Q}_{11}, \widetilde{Q}_{13} \in \mathbb{R}^{2r_C+2r_D,\widetilde{m}_C}$, $\widetilde{Q}_{12}, \widetilde{Q}_{14} \in \mathbb{R}^{2r_C+2r_D,\widetilde{m}_D}$, *and* $\widetilde{m} = \widetilde{m}_C + \widetilde{m}_D$ *it holds that with*

$$\widetilde{Q}_{11} = \hat{Q}_1 S_C, \qquad \widetilde{Q}_{12} = \hat{Q}_2 S_D, \qquad \widetilde{Q}_{13} = \hat{Q}_3 S_C, \qquad \widetilde{Q}_{14} = \hat{Q}_4 S_D$$

the matrix

$$\hat{Q} = \left[\begin{array}{cccc} \hat{Q}_1 & \hat{Q}_2 & \hat{Q}_3 & \hat{Q}_4 \end{array} \right]$$

is nonsingular, then the matrix (3.147) is called an index reducing decomposition matrix of type 1 *of the unselected linear system of type 1 (3.137) and the matrix (3.148) is called an* index reducing decomposition matrix of type 2 *of the unselected linear system of type 2 (3.138).*

Definition 3.5.72 (Index reducing decomposition) *A decomposition (3.139) and (3.140) of the discretized complete minimal reduced derivative array of the form (3.137) or (3.138), respectively, is called* index reducing decomposition *if the matrices* $\widetilde{Q}$ *in (3.139) and (3.140) are index reducing decomposition matrices, see Definition 3.5.71.*

Lemma 3.5.73 *Let the linear system (3.137) be given and let* r_C *and* r_D *be defined by (3.136). Furthermore, let*

$$\widetilde{Q} = \left[\begin{array}{cc} \widetilde{Q}_{11}^1 & 0 \\ \widetilde{Q}_{11}^2 \widetilde{Q}_{11}^1 & \widetilde{Q}_{12}^2 \\ \hline \widetilde{Q}_{21} & \widetilde{Q}_{22} \end{array} \right] \in \mathbb{R}^{\widetilde{m},\widetilde{m}} \tag{3.149}$$

be a nonsingular matrix with $\widetilde{Q}_{11}^1 \in \mathbb{R}^{r_C,\widetilde{m}_C}$, $\widetilde{Q}_{11}^2 \in \mathbb{R}^{r_D,r_C}$, *and* $\widetilde{Q}_{12}^2 \in \mathbb{R}^{r_D,\widetilde{m}_D}$ *satisfying*

$$\left[\begin{array}{cc} \widetilde{Q}_{11}^1 & 0 \\ \widetilde{Q}_{11}^2 \widetilde{Q}_{11}^1 & \widetilde{Q}_{12}^2 \\ \hline \widetilde{Q}_{21} & \widetilde{Q}_{22} \end{array} \right] \left[\begin{array}{c} \widetilde{C} \\ \gamma \widetilde{D} + h \widetilde{B} \end{array} \right] = \left[\begin{array}{c} \widetilde{R} \\ \hline 0 \end{array} \right], \tag{3.150}$$

where $\widetilde{R} \in \mathbb{R}^{r_C+r_D,n}$ *has full rank* $\operatorname{rank}(\widetilde{R}) = r_C + r_D$. *Then, for sufficiently small* h, *the matrix* $\widetilde{Q}$ *is an index reducing decomposition matrix, see Definition 3.5.71.*

Proof: The submatrix $\widetilde{Q}_{11}^1$ forms a discrete kinematic selector, since (3.150) implies (3.141). Furthermore, for sufficiently small h the submatrix $\widetilde{Q}_{12}^2$ forms a discrete dynamic selector, since (3.150) implies (3.143). Therefore, by defining

$$\widetilde{S}_C = \widetilde{Q}_{11}^1 \text{ and } \widetilde{S}_D = \widetilde{Q}_{12}^2 \tag{3.151}$$

we have

$$\widetilde{Q}_{11} = \left[\begin{array}{c} \widetilde{Q}_{11}^1 \\ \widetilde{Q}_{11}^2 \widetilde{Q}_{11}^1 \end{array} \right] = \left[\begin{array}{c} I_{r_C} \\ \widetilde{Q}_{11}^2 \end{array} \right] \widetilde{S}_C = \hat{Q}_1 \widetilde{S}_C$$

and

$$\widetilde{Q}_{12} = \begin{bmatrix} 0 \\ \widetilde{Q}_{12}^2 \end{bmatrix} = \begin{bmatrix} 0 \\ I_{r_D} \end{bmatrix} \widetilde{S}_D = \hat{Q}_2 \widetilde{S}_D.$$

With

$$\hat{Q} = \begin{bmatrix} \hat{Q}_1 & \hat{Q}_2 \end{bmatrix} = \begin{bmatrix} I_{r_C} & 0 \\ \widetilde{Q}_{11}^2 & I_{r_C} \end{bmatrix}$$

we get that $\hat{Q}$ is nonsingular. From Definition 3.5.71 the assertion follows. □

Remark 3.5.74 In particular, it follows from Lemma 3.5.73 that by use of a decomposition (3.139) with a nonsingular matrix $\widetilde{Q}$ of the form (3.149) the discrete selectors $\widetilde{S}_C$ and $\widetilde{S}_D$ can be obtained directly from $\widetilde{Q}$ by (3.151). □

Theorem 3.5.75 *Assume that Procedure 3.5.11 applied to the quasi-linear DAE (3.23) terminates in iteration step $i = \nu = \nu_c + 1$ in (3.42) with maximal constraint level ν_c. Let a^{ν_c} and d^{ν_c} be defined by (3.74) with $l = \nu_c$ and let $a^{\nu_c} = \text{const}$ and $d^{\nu_c} = \text{const}$ for all $(x, \mathfrak{u}^{\nu_c}) \in \mathbb{M}$. Let the related complete minimal reduced derivative array $\widetilde{\mathfrak{F}}_{\nu_c}$ (3.66) be given with $E(x(t), u(t)) \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{m,n})$ and $\widetilde{k}_{2,x}^i(x(t), \mathfrak{u}^i(t)) \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{m_2^i})$ for $i = 0, ..., \nu_c$. Furthermore, let the discrete dynamic selector $\widetilde{S}_D$ and the discrete kinematic selector $\widetilde{S}_C$ be obtained in (3.151) from an index reducing decomposition of the discretized complete minimal reduced derivative array (given by the unselected linear systems (3.137) and (3.138)) at the point (X^*, U^*). Then, for sufficiently small step size h, the constant dynamic selector $S_D(x, \mathfrak{u}^{\nu_c}) = \widetilde{S}_D$ and the constant kinematic selector $S_C(x, \mathfrak{u}^{\nu_c}) = \widetilde{S}_C$ define a regularization of the quasi-linear DAE (3.23) in form of the projected-strangeness-free form with selected constraints (3.78) in a neighborhood of the point (X^*, U^*).*

Proof: The discretization of the complete minimal reduced derivative array (3.66) leads to the unselected linear systems (3.137) and (3.138). The index reducing decomposition (see Definition 3.5.72) of the linear system (3.137) has the form (3.139) with $\begin{bmatrix} \widetilde{Q}_{11} & \widetilde{Q}_{12} \end{bmatrix} = \begin{bmatrix} \hat{Q}_1 S_C & \hat{Q}_2 S_D \end{bmatrix}$ and $\begin{bmatrix} \hat{Q}_1 & \hat{Q}_2 \end{bmatrix}$ is nonsingular, see Definition 3.5.71. Therefore, we obtain with $S_D(x, \mathfrak{u}^{\nu_c}) = \widetilde{S}_D$ and $S_C(x, \mathfrak{u}^{\nu_c}) = \widetilde{S}_C$ the dynamic selector and the kinematic selector, respectively, which satisfy Definitions 3.5.43 and 3.5.44 in a neighborhood $\mathbb{S}((X^*, U^*), \epsilon) = \{(x, \mathfrak{u}^{\nu_s}) : ||(x - X^*, \mathfrak{u}^{\nu_s} - U^*)|| < \epsilon\}$ of the regular point (X^*, U^*) in accordance with Lemmata 3.5.69 and 3.5.70.
Hence, the selectors $S_D(x, \mathfrak{u}^{\nu_s}) = \widetilde{S}_D$ and $S_C(x, \mathfrak{u}^{\nu_s}) = \widetilde{S}_C$ define a regularization of the quasi-linear DAE (3.23) to the projected strangeness-free DAE (3.78) for $t \in (t^* - h, t^* + h)$ with $(x(t), \mathfrak{u}^{\nu_s}(t)) \in \mathbb{S}((X^*, U^*), \epsilon)$. □

Remark 3.5.76 It follows from Theorem 3.5.75 that in view of the nonuniqueness of the kinematic and dynamic selectors, see Definitions 3.5.43 and 3.5.44, the index reducing decomposition of the unselected linear systems (3.137) and (3.138) arising from the discretization of the complete minimal reduced derivative array (3.66) already contains a certain choice of the kinematic and dynamic selectors such that the decomposition of the selected linear systems (3.132) and (3.133) arising from the discretization of the regularized quasi-linear DAE and the index reducing decomposition of the unselected linear systems (3.137) and (3.138) are equivalent in the sense that both solutions are identical. In particular, this means that the sequence of regularization and discretization can be exchanged, see Figure 3.2. □

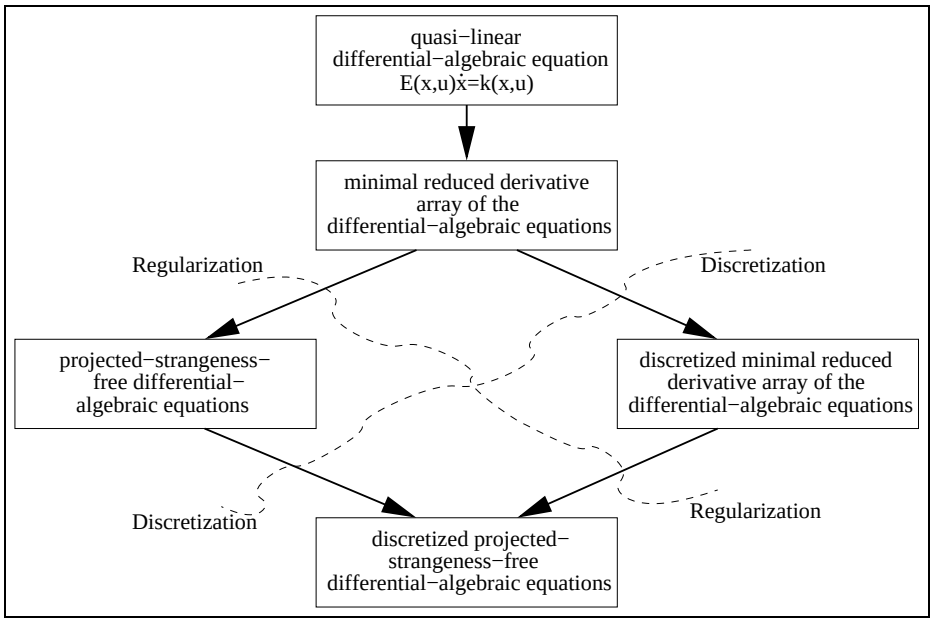

Figure 3.2: Discretization and regularization of quasi-linear DAEs

Example 3.5.77 Consider again the differential-algebraic equation introduced in (3.34), see Example 3.5.7. Assume that $n_1 = n_2 = 1$. The Procedure 3.5.11 is already executed in Example 3.5.15 and we did determine in Example 3.5.42 the maximal constraint level $\nu_c = 1$ and the complete minimal reduced derivative array (3.73) in form of the semi-implicit DAE

$$\begin{aligned} \widetilde{E}_D(x, \mathfrak{u}^1)\dot{x} &= \widetilde{k}_D(x, \mathfrak{u}^1), \\ 0 &= \widetilde{k}_C(x, \mathfrak{u}^1) \end{aligned}$$

with

$$\widetilde{E}_D(x, \mathfrak{u}^1) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \qquad \widetilde{k}_D(x, \mathfrak{u}^1) = \begin{bmatrix} k_1 \\ k_2 \end{bmatrix},$$
$$\widetilde{k}_C(x, \mathfrak{u}^1) = \begin{bmatrix} k_2 \\ -k_{2,x_1} k_1 - k_{2,u}\dot{u} \end{bmatrix}.$$

The discretization of this overdetermined DAE leads to a linear system (3.137) of the form

$$\left[\begin{array}{cc} -k_{2,x_1} & 0 \\ (k_{2,x_1}k_1 + k_{2,u}\dot{u})_{,x_1} & k_{2,x_1}k_{1,x_2} \\ \hline \gamma - hk_{1,x_1} & -hk_{1,x_2} \\ -hk_{2,x_1} & 0 \end{array}\right] \begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix} = \left[\begin{array}{c} \widetilde{b}_{C1} \\ \widetilde{b}_{C2} \\ \hline \widetilde{b}_{D1} \\ \widetilde{b}_{D2} \end{array}\right] \tag{3.152}$$

compare with (3.125), which has to be solved in every Newton iteration step inside every integration step. From (3.136) we get $r_C = 2$ and $r_D = 0$ because of $k_{2,x_1} \neq 0$ and $k_{2,x_1}k_{1,x_2} \neq 0$, see Example 3.5.7. Obviously, this linear system is overdetermined and in general not solvable because of discretization errors. Therefore, it is to be solved up to a remaining residual which mainly depends on the discretization and the step size. Let us use the LU decomposition for simplicity.

If we choose the matrix $\widetilde{Q}$ in the decomposition (3.139) as

$$\widetilde{Q} = \left[\begin{array}{cc|cc} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 1 & q_{32} & q_{33} & 0 \\ 0 & hq_{32} & hq_{33} & 1 \end{array}\right]$$

with

$$\begin{aligned} q_{32} &= hk_{2,x_1}/(h(k_{2,x_1}k_1 + k_{2,u}\dot{u})_{,x_1} + k_{2,x_1}(\gamma - hk_{1,x_1})), \\ q_{33} &= k_{2,x_1}k_{2,x_1}/(h(k_{2,x_1}k_1 + k_{2,u}\dot{u})_{,x_1} + k_{2,x_1}(\gamma - hk_{1,x_1})) \end{aligned}$$

and multiply the equation (3.152) from the left with $\widetilde{Q}$, then we get

$$\left[\begin{array}{cc} (k_{2,x_1}k_1 + k_{2,u}\dot{u})_{,x_1} & +k_{2,x_1}k_{1,x_2} \\ \gamma - hk_{1,x_1} & -hk_{1,x_2} \\ \hline 0 & 0 \\ 0 & 0 \end{array}\right] \left[\begin{array}{c} \xi_1 \\ \xi_2 \end{array}\right] = \left[\begin{array}{c} \widetilde{b}_{C2} \\ \widetilde{b}_{D1} \\ \hline \widetilde{b}_{C1} + q_{32}\widetilde{b}_{C2} + q_{33}\widetilde{b}_{D1} \\ hq_{32}\widetilde{b}_{C2} + hq_{33}\widetilde{b}_{D1} + \widetilde{b}_{D2} \end{array}\right]. \quad (3.153)$$

Obviously, the upper two by two submatrix of the leading matrix in (3.153) is nonsingular for small h because of $\gamma \neq 0$ and $k_{2,x_1}k_{1,x_2} \neq 0$, and the equation can be solved for ξ_1 and ξ_2. But, this numerical solution only depends on the second and the third equation of (3.152) which correspond to the discretization of the differential equation of (3.34) and of the first time derivative of the constraint equation of (3.34). In particular, this means that we actually use an index reduced formulation of the DAE which is obtained just by replacing the constraint (3.34b) by its first time derivative (3.34c) which is not advisable because of the drift-off phenomenon, see Section 3.4.1. Therefore, this choice of the transformation matrix $\widetilde{Q}$ is not a good choice and on the other hand the transformation matrix $\widetilde{Q}$ is not an index reducing decomposition matrix, see Definition 3.5.71.
A better choice of the transformation matrix $\widetilde{Q}$ is for example

$$\widetilde{Q} = \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline \widetilde{q}_{31} & \widetilde{q}_{32} & 1 & 0 \\ \widetilde{q}_{41} & \widetilde{q}_{42} & 0 & 1 \end{array}\right]$$

with

$$\begin{aligned} \widetilde{q}_{31} &= \frac{\gamma k_{2,x_1} - hk_{1,x_1}k_{2,x_1} + h(k_{2,x_1}k_1 + k_{2,u}\dot{u})_{,x_1}}{k_{2,x_1}k_{2,x_1}}, & \widetilde{q}_{32} &= \frac{h}{k_{2,x_1}}, \\ \widetilde{q}_{41} &= -h, & \widetilde{q}_{42} &= 0. \end{aligned}$$

Note that this choice of $\widetilde{Q}$ corresponds to an index reducing decomposition matrix with $S_C = I_2 \in \mathbb{R}^{2,2}$ and $S_D \in \mathbb{R}^{0,2}$. see Definition 3.5.71. By multiplication of the equation (3.152) from the left with $\widetilde{Q}$ only the discretized algebraic constraints remain in considerations, which is good since the original DAE (3.34) actually corresponds to two algebraic equations, see Example 3.5.7 with $n_1 = n_2 = 1$. Therefore, the discretization of the projected-strangeness-free form (3.96) is equivalent to the index reducing decomposition of the discretization of the complete minimal reduced derivative array (3.73). □

We have seen that the numerical integration of the quasi-linear DAE (3.23) needs the efficient solution (via index reducing decompositions) of an overdetermined linear system of the form

$$\left[\begin{array}{c} C \\ \gamma D + hB \end{array}\right] \xi = \left[\begin{array}{c} b_C \\ b_D \end{array}\right], \quad (3.154)$$

with $C \in \mathbb{R}^{m_C,n}$, $D, B \in \mathbb{R}^{m_D,n}$, $b_C \in \mathbb{R}^{m_C}$, $b_D \in \mathbb{R}^{m_D}$, $\xi \in \mathbb{R}^n$, and $m_C + m_D = m$. This linear system corresponds to (3.123). Furthermore, we have to solve the second linear system of the form

$$\left[\begin{array}{c|c} C & 0 \\ \alpha D + hB & -\beta D + hB \\ \hline 0 & C \\ \beta D + hB & \alpha D + hB \end{array}\right] \left[\begin{array}{c} \xi_1 \\ \hline \xi_2 \end{array}\right] = \left[\begin{array}{c} b_{1C} \\ b_{1D} \\ \hline b_{2C} \\ b_{2D} \end{array}\right], \tag{3.155}$$

where the matrices B, C, D are the same as in the system (3.154) and $b_{jC} \in \mathbb{R}^{m_C}$, $b_{jD} \in \mathbb{R}^{m_D}$, $\xi_j \in \mathbb{R}^n$, $j = 1, 2$.
In the following we will discuss the efficient solution of both linear systems exploiting the given structure. The decomposition of all unselected linear systems, i.e., the unselected linear systems of type 1 for $\gamma = \gamma_i$ for $i = 1, ..., n_R$ and the unselected linear systems of type 2 for $\alpha = \alpha_i$, $\beta = \beta_i$ for $i = 1, ..., n_I$, will be done in the following way. First, we will pick out arbitrarily one of the unselected linear systems of type 1 and perform a decomposition as described below. Then we will use the information of the transformation matrices $\overset{*}{Q}$ and $\overset{*}{P}$ for the decomposition of the remaining unselected linear systems of type 1 and for the decomposition of the unselected linear systems of type 2 with the aim to perform an index reducing decomposition.
From the properties of an index reducing decomposition we have the condition that the algebraic part of the linear system (3.137) has to be satisfied exactly. Therefore, we decompose only the algebraic part in the first step. Let the matrices

$$\overset{*}{Q}_C = \left[\begin{array}{c} \overset{*}{Q}{}^1_C \\ \overset{*}{Q}{}^2_C \end{array}\right] \in \mathbb{R}^{m_C,m_C} \quad \text{and} \quad \overset{*}{P} = \left[\begin{array}{cc} \overset{*}{P}_1 & \check{P}_2 \end{array}\right] \in \mathbb{R}^{n,n}$$

be nonsingular with $\overset{*}{Q}{}^1_C \in \mathbb{R}^{r_C,m_C}$, $\overset{*}{Q}{}^2_C \in \mathbb{R}^{m_C-r_C,m_C}$, $\overset{*}{P}_1 \in \mathbb{R}^{n,r_C}$, and $\check{P}_2 \in \mathbb{R}^{n,n-r_C}$ such that

$$\overset{*}{R}_C = \overset{*}{Q}{}^1_C C \overset{*}{P}_1 \in \mathbb{R}^{r_C,r_C} \text{ is nonsingular and } \overset{*}{Q}{}^2_C C = 0 \in \mathbb{R}^{m_C-r_C,n}. \tag{3.156}$$

Then, by scaling the constraint part with $\overset{*}{Q}_C$ and transforming with $\xi = \overset{*}{P}\zeta$ we get from (3.154) the equivalent equation

$$\left[\begin{array}{cc} \overset{*}{R}_C & \overset{*}{V}_C \check{P}_2 \\ 0 & 0 \\ (\gamma D + hB)\overset{*}{P}_1 & (\gamma D + hB)\check{P}_2 \end{array}\right] \overset{*}{P}{}^{-1} \xi = \left[\begin{array}{c} \overset{*}{Q}{}^1_C b_C \\ \overset{*}{Q}{}^2_C b_C \\ b_D \end{array}\right] \tag{3.157}$$

with

$$\overset{*}{V}_C = \overset{*}{Q}{}^1_C C \in \mathbb{R}^{r_C,n}. \tag{3.158}$$

The first r_C components of the vector $\zeta = \overset{*}{P}{}^{-1}\xi$ can be determined from the algebraic part because of the nonsingularity of the matrix $\overset{*}{R}_C$. Hence, the first r_C columns of the differential part can be eliminated by use of block Gauß[11] elimination. Let $\overset{*}{L}_D, \overset{*}{L}_B \in \mathbb{R}^{m_D,r_C}$ be such that,

$$\overset{*}{L}_D \overset{*}{R}_C + D\overset{*}{P}_1 = 0 \quad \text{and} \quad \overset{*}{L}_B \overset{*}{R}_C + B\overset{*}{P}_1 = 0. \tag{3.159}$$

Then by scaling with the nonsingular matrix

$$\left[\begin{array}{ccc} I & 0 & 0 \\ 0 & I & 0 \\ \gamma\overset{*}{L}_D + h\overset{*}{L}_B & 0 & I \end{array}\right]$$

[11] Carl Friedrich Gauß (born 1777 in Braunschweig, Germany - died 1855 in Göttingen, Germany)

we get

$$\begin{bmatrix} \mathring{R}_C & \mathring{V}_C\check{P}_2 \\ 0 & 0 \\ 0 & (\gamma\mathring{V}_D + h\mathring{V}_B)\check{P}_2 \end{bmatrix} \mathring{P}^{-1}\xi = \begin{bmatrix} \mathring{Q}_C^1 b_C \\ \mathring{Q}_C^2 b_C \\ (\gamma\mathring{L}_D + h\mathring{L}_B)\mathring{Q}_C^1 b_C + b_D \end{bmatrix}, \quad (3.160a)$$

where

$$\mathring{V}_D = D + \mathring{L}_D\mathring{V}_C \ \text{ and } \ \mathring{V}_B = B + \mathring{L}_D\mathring{V}_C. \quad (3.160b)$$

Finally, let us decompose the remaining differential part. Let

$$\mathring{Q}_D = \begin{bmatrix} \mathring{Q}_D^1 \\ \mathring{Q}_D^2 \end{bmatrix} \in \mathbb{R}^{m_D,m_D}$$

be nonsingular with $\mathring{Q}_D^1 \in \mathbb{R}^{r_D,m_D}$ and $\mathring{Q}_D^2 \in \mathbb{R}^{m_D-r_D,m_D}$ and let $\check{P}_2 = \begin{bmatrix} \mathring{P}_2 & \mathring{P}_3 \end{bmatrix}$ with $\mathring{P}_2 \in \mathbb{R}^{n,r_D}$ and $\mathring{P}_3 \in \mathbb{R}^{n,n-r_C-r_D}$ of full rank such that $\mathring{P} = \begin{bmatrix} \mathring{P}_1 & \mathring{P}_2 & \mathring{P}_3 \end{bmatrix}$ is nonsingular,

$$\mathring{R}_R = \mathring{Q}_D^1(\gamma\mathring{V}_D + h\mathring{V}_B)\mathring{P}_2 \in \mathbb{R}^{r_D,r_D} \text{ is nonsingular,}$$

and

$$\mathring{Q}_D^2(\gamma\mathring{V}_D + h\mathring{V}_B)\begin{bmatrix} \mathring{P}_2 & \mathring{P}_3 \end{bmatrix} = 0 \in \mathbb{R}^{m_D-r_D,n-r_C}.$$

Then by scaling the differential part with $\mathring{Q}_D$ and transforming $\xi = \mathring{P}\zeta$ we obtain

$$\begin{bmatrix} \mathring{R}_C & \mathring{V}_C\mathring{P}_2 & \mathring{V}_C\mathring{P}_3 \\ 0 & 0 & 0 \\ 0 & \mathring{R}_R & \mathring{V}_R \\ 0 & 0 & 0 \end{bmatrix} \mathring{P}^{-1}\xi = \begin{bmatrix} \mathring{Q}_C^1 b_C \\ \mathring{Q}_C^2 b_C \\ \mathring{Q}_D^1((\gamma\mathring{L}_D + h\mathring{L}_B)\mathring{Q}_C^1 b_C + b_D) \\ \mathring{Q}_D^2((\gamma\mathring{L}_D + h\mathring{L}_B)\mathring{Q}_C^1 b_C + b_D) \end{bmatrix} \quad (3.161)$$

with

$$\mathring{V}_R = \mathring{Q}_D^1(\gamma\mathring{V}_D + h\mathring{V}_B)\mathring{P}_3 \in \mathbb{R}^{r_D,n-r_C-r_D}. \quad (3.162)$$

Summarizing these three steps, together with a row permutation, we get the transformation of the linear system (3.154) in the form

$$\mathring{Q}_R \begin{bmatrix} C \\ \gamma D + hB \end{bmatrix} \mathring{P}\mathring{P}^{-1}\xi = \mathring{Q}_R \begin{bmatrix} b_C \\ b_D \end{bmatrix} \quad (3.163a)$$

with

$$\mathring{Q}_R = \left[\begin{array}{cc} \mathring{Q}_C^1 & 0 \\ \mathring{Q}_D^1(\gamma\mathring{L}_D + h\mathring{L}_B)\mathring{Q}_C^1 & \mathring{Q}_D^1 \\ \hline \mathring{Q}_C^2 & 0 \\ \mathring{Q}_D^2(\gamma\mathring{L}_D + h\mathring{L}_B)\mathring{Q}_C^1 & \mathring{Q}_D^2 \end{array}\right], \quad (3.163b)$$

$$\mathring{P} = \begin{bmatrix} \mathring{P}_1 & \mathring{P}_2 & \mathring{P}_3 \end{bmatrix}. \quad (3.163c)$$

This yields an equivalent system to the system (3.154) of the form

$$\left[\begin{array}{cc|c} \mathring{R}_C & \mathring{V}_C\mathring{P}_2 & \mathring{V}_C\mathring{P}_3 \\ 0 & \mathring{R}_R & \mathring{V}_R \\ \hline 0 & 0 & 0 \\ 0 & 0 & 0 \end{array}\right] \zeta = \left[\begin{array}{c} \mathring{Q}_C^1 b_C \\ \mathring{Q}_D^1((\gamma\mathring{L}_D + h\mathring{L}_B)\mathring{Q}_C^1 b_C + b_D) \\ \hline \mathring{Q}_C^2 b_C \\ \mathring{Q}_D^2((\gamma\mathring{L}_D + h\mathring{L}_B)\mathring{Q}_C^1 b_C + b_D) \end{array}\right], \quad (3.164a)$$

$$\xi = \mathring{P}\zeta. \quad (3.164b)$$

A solution ξ of (3.154) can be obtained by solving the linear equation (3.164a) for ζ by use of the first two block rows of (3.164a) with subsequent transformation of ζ to ξ by the equation (3.164b). If the last block column of the leading matrix of (3.164a) vanishes, then the solution of (3.154) is unique.

Remark 3.5.78 The matrix $\mathring{Q}_R$ given in (3.163b) is an index reducing decomposition matrix, see Definition 3.5.71 and Lemma 3.5.73. Therefore, the decomposition of (3.154) given by (3.163) is a suitable decomposition of the discretized minimal reduced derivative array (3.66) regarding the regularization of the quasi-linear DAE (3.23) to the projected-strangeness-free formulation with selected constraints (3.78). Furthermore, we get the discrete kinematic selector as $\widetilde{S}_C = \mathring{Q}_C^1$ and the discrete dynamic selector $\widetilde{S}_D = \mathring{Q}_D^1$ in accordance with Remark 3.5.74. Both can be used as locally constant kinematic and dynamic selectors, respectively, for the regularization of the quasi-linear DAE (3.23) to the projected-strangeness-free formulation with selected constraints (3.78). □

For the remaining unselected linear systems of type 1 we can determine the transformation of the linear system (3.154) in the form

$$Q_R \begin{bmatrix} C \\ \gamma D + hB \end{bmatrix} P_R P_R^{-1} \xi = Q_R \begin{bmatrix} b_C \\ b_D \end{bmatrix} \tag{3.165a}$$

with

$$Q_R = \left[\begin{array}{cc} \mathring{Q}_C^1 & 0 \\ Q_D^1 \mathring{Q}_D^1 (\gamma \mathring{L}_D + h \mathring{L}_B) \mathring{Q}_C^1 & Q_D^1 \mathring{Q}_D^1 \\ \hline \mathring{Q}_C^2 & 0 \\ Q_D^2 (\gamma \mathring{L}_D + h \mathring{L}_B) \mathring{Q}_C^1 & Q_D^2 \end{array}\right], \tag{3.165b}$$

$$P_R = \begin{bmatrix} \mathring{P}_1 & \mathring{P}_2 & \mathring{P}_3 \end{bmatrix}, \tag{3.165c}$$

and the matrix

$$\left[\begin{array}{c} Q_D^1 \mathring{Q}_D^1 \\ \hline Q_D^2 \end{array}\right] \tag{3.166}$$

is chosen nonsingular with $Q_D^1 \mathring{Q}_D^1 \in \mathbb{R}^{r_D, m_D}$, $Q_D^2 \in \mathbb{R}^{m_D - r_D, m_D}$, and $Q_D^1 \in \mathbb{R}^{r_D, r_D}$ nonsingular such that

$$\left[\begin{array}{c} Q_D^1 \mathring{Q}_D^1 \\ \hline Q_D^2 \end{array}\right] \begin{bmatrix} \gamma \mathring{V}_D + h \mathring{V}_B \end{bmatrix} \begin{bmatrix} \mathring{P}_2 & \mathring{P}_3 \end{bmatrix} = \left[\begin{array}{cc} R_R & V_R \\ \hline 0 & 0 \end{array}\right]$$

with $R_R \in \mathbb{R}^{r_D, r_D}$ nonsingular. This yields an equivalent system to the system (3.154) of the form

$$\left[\begin{array}{cc|c} \mathring{R}_C & \mathring{V}_C \mathring{P}_2 & \mathring{V}_C \mathring{P}_3 \\ 0 & R_R & V_R \\ \hline 0 & 0 & 0 \\ 0 & 0 & 0 \end{array}\right] \zeta = \left[\begin{array}{c} \mathring{Q}_C^1 b_C \\ Q_D^1 \mathring{Q}_D^1 ((\gamma \mathring{L}_D + h \mathring{L}_B) \mathring{Q}_C^1 b_C + b_D) \\ \hline \mathring{Q}_C^2 b_C \\ Q_D^2 ((\gamma \mathring{L}_D + h \mathring{L}_B) \mathring{Q}_C^1 b_C + b_D) \end{array}\right], \tag{3.167a}$$

$$\xi = P_R \zeta. \tag{3.167b}$$

Remark 3.5.79 Since the transformation matrices $\mathring{Q}_C$, $\mathring{L}_D$, $\mathring{L}_B$, $\mathring{P}$ as well as the resulting matrix products $\mathring{R}_C$, $\mathring{V}_C \mathring{P}_2$, and $\mathring{V}_C \mathring{P}_3$ are known from the particular linear system of type 1 (3.154) and since the last two block columns in the linear

system (3.167a) will be ignored for the solution of the linear system of type 1 (3.154), the amount of computation for the decomposition of the remaining linear systems of type 1 (3.154) reduces to the determination of a nonsingular transformation matrix Q_D^1 such that

$$Q_D^1\Big(\overset{*}{Q}{}_D^1(\gamma\overset{*}{V}_D + h\overset{*}{V}_B)\overset{*}{P}_2\Big) = R_R \in \mathbb{R}^{r_D,r_D}$$

is nonsingular and preferably an upper triangular matrix. □

Let us now consider the linear system (3.155). Inspired by the decomposition of the linear system (3.154) we define a transformation of the system (3.155) as

$$\overset{*}{Q}_I \left[\begin{array}{c|c} C & 0 \\ \alpha D + hB & -\beta D + hB \\ \hline 0 & C \\ \beta D + hB & \alpha D + hB \end{array}\right] P_I P_I^{-1} \left[\begin{array}{c} \xi_1 \\ \hline \xi_2 \end{array}\right] = \overset{*}{Q}_I \left[\begin{array}{c} b_{11} \\ b_{12} \\ \hline b_{21} \\ b_{22} \end{array}\right], \tag{3.168a}$$

with

$$Q_I = \left[\begin{array}{cc|cc} \overset{*}{Q}{}_C^1 & 0 & 0 & 0 \\ 0 & 0 & \overset{*}{Q}{}_C^1 & 0 \\ \begin{pmatrix} Q_{D11}^1\overset{*}{Q}{}_D^1 L_\alpha \\ +Q_{D12}^1\overset{*}{Q}{}_D^1 L_\beta \end{pmatrix}\overset{*}{Q}{}_C^1 & Q_{D11}^1\overset{*}{Q}{}_D^1 & \begin{pmatrix} Q_{D11}^1\overset{*}{Q}{}_D^1 L_{-\beta} \\ +Q_{D12}^1\overset{*}{Q}{}_D^1 L_\alpha \end{pmatrix}\overset{*}{Q}{}_C^1 & Q_{D12}^1\overset{*}{Q}{}_D^1 \\ \begin{pmatrix} Q_{D21}^1\overset{*}{Q}{}_D^1 L_\alpha \\ +Q_{D22}^1\overset{*}{Q}{}_D^1 L_\beta \end{pmatrix}\overset{*}{Q}{}_C^1 & Q_{D21}^1\overset{*}{Q}{}_D^1 & \begin{pmatrix} Q_{D21}^1\overset{*}{Q}{}_D^1 L_{-\beta} \\ +Q_{D22}^1\overset{*}{Q}{}_D^1 L_\alpha \end{pmatrix}\overset{*}{Q}{}_C^1 & Q_{D22}^1\overset{*}{Q}{}_D^1 \\ \hline \overset{*}{Q}{}_C^2 & 0 & 0 & 0 \\ 0 & 0 & \overset{*}{Q}{}_C^2 & 0 \\ \begin{pmatrix} Q_{D11}^2 L_\alpha \\ +Q_{D12}^2 L_\beta \end{pmatrix}\overset{*}{Q}{}_C^1 & Q_{D11}^2 & \begin{pmatrix} Q_{D11}^2 L_{-\beta} \\ +Q_{D12}^2 L_\alpha \end{pmatrix}\overset{*}{Q}{}_C^1 & Q_{D12}^2 \\ \begin{pmatrix} Q_{D21}^2 L_\alpha \\ +Q_{D22}^2 L_\beta \end{pmatrix}\overset{*}{Q}{}_C^1 & Q_{D21}^2 & \begin{pmatrix} Q_{D21}^2 L_{-\beta} \\ +Q_{D22}^2 L_\alpha \end{pmatrix}\overset{*}{Q}{}_C^1 & Q_{D22}^2 \end{array}\right], \tag{3.168b}$$

$$P_I = \begin{bmatrix} \overset{*}{P}_1 & 0 & \overset{*}{P}_2 & 0 & \overset{*}{P}_3 & 0 \\ 0 & \overset{*}{P}_1 & 0 & \overset{*}{P}_2 & 0 & \overset{*}{P}_3 \end{bmatrix} \tag{3.168c}$$

and

$$L_\alpha = (\alpha\overset{*}{L}_D + h\overset{*}{L}_B),\ L_\beta = (\beta\overset{*}{L}_D + h\overset{*}{L}_B),\ L_{-\beta} = (-\beta\overset{*}{L}_D + h\overset{*}{L}_B).$$

Let the matrix

$$\left[\begin{array}{cc} Q_{D11}^1\overset{*}{Q}{}_D^1 & Q_{D12}^1\overset{*}{Q}{}_D^1 \\ \hline Q_{D21}^1\overset{*}{Q}{}_D^1 & Q_{D22}^1\overset{*}{Q}{}_D^1 \\ \hline Q_{D11}^2 & Q_{D12}^2 \\ Q_{D21}^2 & Q_{D22}^2 \end{array}\right],$$

be chosen with $Q_{Dij}^1\overset{*}{Q}{}_D^1 \in \mathbb{R}^{r_D,m_D}$, $Q_{Dij}^2 \in \mathbb{R}^{m_D-r_D,m_D}$, $i,j = 1,2$ and

$$\begin{aligned} &\left[\begin{array}{cc} Q_{D11}^1\overset{*}{Q}{}_D^1 & Q_{D12}^1\overset{*}{Q}{}_D^1 \\ \hline Q_{D21}^1\overset{*}{Q}{}_D^1 & Q_{D22}^1\overset{*}{Q}{}_D^1 \\ \hline Q_{D11}^2 & Q_{D12}^2 \\ Q_{D21}^2 & Q_{D22}^2 \end{array}\right] \begin{bmatrix} \alpha\overset{*}{V}_D + h\overset{*}{V}_B & -\beta\overset{*}{V}_D + h\overset{*}{V}_B \\ \beta\overset{*}{V}_D + h\overset{*}{V}_B & \alpha\overset{*}{V}_D + h\overset{*}{V}_B \end{bmatrix} \begin{bmatrix} \overset{*}{P}_2 & 0 & \overset{*}{P}_3 & 0 \\ 0 & \overset{*}{P}_2 & 0 & \overset{*}{P}_3 \end{bmatrix} \\ &= \left[\begin{array}{cc|cc} R_{I1} & V_{I12} & V_{I13} & V_{I14} \\ 0 & R_{I2} & V_{I23} & V_{I24} \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right], \end{aligned} \tag{3.169a}$$

where

$$\begin{bmatrix} R_{I1} & V_{I12} \\ 0 & R_{I2} \end{bmatrix} \in \mathbb{R}^{2r_D,2r_D} \tag{3.169b}$$

is upper triangular and nonsingular. Then, system (3.155) is equivalent to

$$\left[\begin{array}{cccc|cc} \overset{*}{R}_C & 0 & \overset{*}{V}_C\overset{*}{P}_2 & 0 & \overset{*}{V}_C\overset{*}{P}_3 & 0 \\ 0 & \overset{*}{R}_C & 0 & \overset{*}{V}_C\overset{*}{P}_2 & 0 & \overset{*}{V}_C\overset{*}{P}_3 \\ 0 & 0 & R_{I1} & V_{I12} & V_{I13} & V_{I14} \\ 0 & 0 & 0 & R_{I2} & V_{I23} & V_{I24} \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \zeta \tag{3.170a}$$

$$= \left[\begin{array}{c} \overset{*}{Q}{}^1_C b_{1C} \\ \overset{*}{Q}{}^1_C b_{2C} \\ \left(\begin{array}{c} (Q^1_{D11}\overset{*}{Q}{}^1_D(\alpha\overset{*}{L}_D + h\overset{*}{L}_B) + Q^1_{D12}\overset{*}{Q}{}^1_D(\beta\overset{*}{L}_D + h\overset{*}{L}_B))\overset{*}{Q}{}^1_C b_{1C} \\ +(Q^1_{D11}\overset{*}{Q}{}^1_D(-\beta\overset{*}{L}_D + h\overset{*}{L}_B) + Q^1_{D12}\overset{*}{Q}{}^1_D(\alpha\overset{*}{L}_D + h\overset{*}{L}_B))\overset{*}{Q}{}^1_C b_{2C} \\ +Q^1_{D11}\overset{*}{Q}{}^1_D b_{1D} + Q^1_{D12}\overset{*}{Q}{}^1_D b_{2D} \end{array}\right) \\ \left(\begin{array}{c} (Q^1_{D21}\overset{*}{Q}{}^1_D(\alpha\overset{*}{L}_D + h\overset{*}{L}_B) + Q^1_{D22}\overset{*}{Q}{}^1_D(\beta\overset{*}{L}_D + h\overset{*}{L}_B))Q^1_C b_{1C} \\ +(Q^1_{D21}\overset{*}{Q}{}^1_D(-\beta\overset{*}{L}_D + h\overset{*}{L}_B) + Q^1_{D22}\overset{*}{Q}{}^1_D(\alpha\overset{*}{L}_D + h\overset{*}{L}_B))Q^1_C b_{2C} \\ +Q^1_{D21}\overset{*}{Q}{}^1_D b_{1D} + Q^1_{D22}\overset{*}{Q}{}^1_D b_{2D} \end{array}\right) \\ \hline \overset{*}{Q}{}^2_C b_{1C} \\ \overset{*}{Q}{}^2_C b_{2C} \\ \left(\begin{array}{c} (Q^2_{D11}(\alpha\overset{*}{L}_D + h\overset{*}{L}_B) + Q^2_{D12}(\beta\overset{*}{L}_D + h\overset{*}{L}_B))\overset{*}{Q}{}^1_C b_{1C} + Q^2_{D11}b_{1D} \\ +(Q^2_{D11}(-\beta\overset{*}{L}_D + h\overset{*}{L}_B) + Q^2_{D12}(\alpha\overset{*}{L}_D + h\overset{*}{L}_B))\overset{*}{Q}{}^1_C b_{2C} + Q^2_{D12}b_{2D} \end{array}\right) \\ \left(\begin{array}{c} (Q^2_{D21}(\alpha\overset{*}{L}_D + h\overset{*}{L}_B) + Q^2_{D22}(\beta\overset{*}{L}_D + h\overset{*}{L}_B))\overset{*}{Q}{}^1_C b_{1C} + Q^2_{D21}b_{1D} \\ +(Q^2_{D21}(-\beta\overset{*}{L}_D + h\overset{*}{L}_B) + Q^2_{D22}(\alpha\overset{*}{L}_D + h\overset{*}{L}_B))\overset{*}{Q}{}^1_C b_{2C} + Q^2_{D22}b_{2D} \end{array}\right) \end{array}\right],$$

$$\begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix} = P_I\zeta. \tag{3.170b}$$

Solutions ξ_1 and ξ_2 can be obtained by solving the linear equation (3.170a) for ζ via the first four block rows of (3.170a) with subsequent transformation of ζ to ξ_1 and ξ_2 by the equation (3.170b). If the last two block columns of the leading matrix of (3.170a) vanish, then the solutions ξ_1 and ξ_2 are unique.

Remark 3.5.80 Since the transformation matrices $\overset{*}{Q}_C$, $\overset{*}{L}_D$, $\overset{*}{L}_B$, $\overset{*}{P}$ as well as the resulting matrix products $\overset{*}{R}_C$, $\overset{*}{V}_C\overset{*}{P}_2$, and $\overset{*}{V}_C\overset{*}{P}_3$ are known from the particular linear system of type 1 (3.154) and since the last four block columns in the linear system (3.170a) will be ignored for the solution of the linear system of type 2 (3.155), the amount of computation for the decomposition of the linear systems of type 2 (3.155) reduces to the determination of the nonsingular transformation matrix

$$\begin{bmatrix} Q^1_{D11} & Q^1_{D12} \\ Q^1_{D21} & Q^1_{D22} \end{bmatrix}$$

which must satisfy that

$$\begin{bmatrix} Q^1_{D11} & Q^1_{D12} \\ Q^1_{D21} & Q^1_{D22} \end{bmatrix} \begin{bmatrix} \overset{*}{Q}{}^1_D(\alpha \overset{*}{V}_D + h\overset{*}{V}_B)\overset{*}{P}_2 & \overset{*}{Q}{}^1_D(-\beta \overset{*}{V}_D + h\overset{*}{V}{}^2_B)\overset{*}{P}_2 \\ \overset{*}{Q}{}^1_D(\beta \overset{*}{V}_D + h\overset{*}{V}_B)\overset{*}{P}_2 & \overset{*}{Q}{}^1_D(\alpha \overset{*}{V}_D + h\overset{*}{V}{}^2_B)\overset{*}{P}_2 \end{bmatrix} = \begin{bmatrix} R_{I1} & V_{I12} \\ 0 & R_{I2} \end{bmatrix} \in \mathbb{R}^{2r_D, 2r_D}$$

is nonsingular and preferably an upper triangular matrix. □

Remark 3.5.81 The matrix Q_I given in (3.168b) is an index reducing decomposition matrix, see Definition 3.5.71. Therefore, the decomposition of (3.155) given by (3.168) is a suitable decomposition of the discretized minimal reduced derivative array (3.66) for the regularization of the quasi-linear DAE (3.23) to the projected-strangeness-free formulation (3.77). □

Remark 3.5.82 It is important to note that the transformation matrices $\overset{*}{Q}_C$, $\overset{*}{L}_D$, $\overset{*}{L}_B$, and $\overset{*}{P}$ as well as the resulting matrix products $\overset{*}{R}_C$, $\overset{*}{V}_C\overset{*}{P}_2$, and $\overset{*}{V}_C\overset{*}{P}_3$ are already known from the treatment of the particular linear system (3.125). Nevertheless, the matrices $\overset{*}{Q}_C$, $\overset{*}{L}_D$, $\overset{*}{L}_B$, $\overset{*}{P}$, $\overset{*}{R}_C$, $\overset{*}{V}_C\overset{*}{P}_2$, and $\overset{*}{V}_C\overset{*}{P}_3$ depend only on the submatrices C, D, and B, but not on the coefficients α, β, or the step size h. Therefore, it is not necessary to recompute these matrices for the remaining linear systems (3.125) and (3.126a).
In particular, this means that the matrices $\overset{*}{Q}_C$, $\overset{*}{L}_D$, $\overset{*}{L}_B$, $\overset{*}{P}$, $\overset{*}{R}_C$, $\overset{*}{V}_C\overset{*}{P}_2$, and $\overset{*}{V}_C\overset{*}{P}_3$ have to be determined only once for the whole simplified Newton process and they are usable for all linear systems obtained by the discretization of the minimal reduced derivative array. Therefore, let us denote the transformations done starting with (3.154) up to (3.160a) as *predecomposition process*.
In this way the amount of computation can be reduced. Instead of the decomposition of n_R linear systems of type 1 (3.154) of size $\widetilde{m} \times n$ and n_I linear systems of type 2 (3.155) of size $2\widetilde{m} \times 2n$ we have to decompose only one linear system of type 1 (3.154) of size $\widetilde{m} \times n$ and $n_R - 1$ linear systems of size $r_D \times r_D$ and n_I linear systems of size $2r_D \times 2r_D$, see Remarks 3.5.79 and 3.5.80, respectively. □

3.5.4.4 Stability of Runge-Kutta methods

Stability is a very important fact in the numerical treatment of (stiff) ordinary differential equations as well as of differential-algebraic equations. In the following we will review and discuss several commonly used stability concepts. For more details see for example [63, 74, 82, 173].
With respect to the numerical treatment, the notion of stability refers to the behavior of the numerical solution of differential equations whose solution shows a stable behavior. In particular, if the solution of an initial value problem has the property

$$||y(t+h)|| \leq ||y(t)|| \text{ for all } h > 0, \tag{3.171}$$

this should be reflected also in the numerical solution. For the investigation of the stability of numerical methods for ODEs Dahlquist[12] introduced in [38] the equation

$$\dot{x} = \lambda x \text{ with } \lambda \in \mathbb{C} \text{ and } \mathrm{Re}(\lambda) < 0 \tag{3.172}$$

as test equation. The solution of this equation is given by

$$x(t) = ce^{\lambda t}$$

[12]Germund Dahlquist (born 1925 in Uppsala, Sweden - died in Stockholm, Sweden 2005)

with a constant c determined by the initial value. This function $x(t)$ has the property (3.171) because of the negative real part of λ.
Beside the property (3.171) with respect to (3.172) one is also interested in the property that

$$\lim_{h\mathrm{Re}(\lambda)\to-\infty} y(t+h) = 0 \tag{3.173}$$

which should also be reflected in the numerical solution.
The application of a Runge-Kutta method given by the Butcher tableau in Table 3.1 to Dahlquist's equation (3.172) for a particular integration step from t_0 to $t_1 = t_0+h$ yields

$$\begin{aligned} X_i &= x_0 + \lambda h \sum_{j=1}^{s} a_{ij} X_j, \quad i = 1, ..., s, \\ x_1 &= x_0 + \lambda h \sum_{j=1}^{s} b_j X_j \end{aligned}$$

which is equivalent to

$$x_1 = R(z)x_0$$

with $z = \lambda h$ and

$$R(z) = 1 - zb^T(I_s - zA)^{-1}\mathbb{1}, \tag{3.174}$$

where $\mathbb{1} = \left[\begin{array}{ccc} 1 & \cdots & 1 \end{array}\right]^T \in \mathbb{R}^s$.

Definition 3.5.83 (Stability function of a Runge-Kutta method) *The function $R(z)$ defined in (3.174) is called* stability function of a Runge-Kutta method *defined by Table 3.1.*

Definition 3.5.84 (A-stability and A(α)-stability) *A Runge-Kutta method is called* A-stable*, if*

$$|R(z)| \le 1 \quad \textit{for all} \quad z \in \mathbb{C} \textit{ with } \mathrm{Re}(z) \le 0$$

and it is called A(α)-stable*, if*

$$|R(z)| \le 1 \quad \textit{for all} \quad z \in \mathbb{C} \textit{ with } \mathrm{Re}(z) \le 0 \textit{ and } |\arg(z) - \pi| \le \alpha$$

with $\alpha \in (0, \pi/2)$.

A-stability corresponds to the fact that independent of the step size h the decreasing behavior (3.171) is also reflected in the numerical solution.
Let us again consider equation (3.172). If the parameter λ is increasing in its absolute value then this differential equation becomes more and more stiff and corresponds to the singularly perturbed system

$$\frac{1}{\lambda}\dot{x} = x.$$

Hence, if $\lambda \to \infty$ we have in the limit the equation

$$0 = x$$

which corresponds to an (differential-)algebraic equation. With $z = \lambda h \to \infty$ this shows that for the numerical treatment of differential-algebraic equations by use of Runge-Kutta methods the limit of the stability function $R(z)$ at infinity plays a very decisive role. For invertible Runge-Kutta matrices A we have

$$R(\infty) = 1 - b^T A^{-1}\mathbb{1} \tag{3.175}$$

with $\mathbb{1} = \left[\begin{array}{ccc} 1 & \cdots & 1 \end{array}\right]^T \in \mathbb{R}^s$.

Definition 3.5.85 (strong A-stability) *A Runge-Kutta method is called* strongly A-stable*, if it is A-stable and if*

$$\lim_{\mathrm{Re}(z)\to-\infty} |R(z)| < 1.$$

Definition 3.5.86 (L-stability) *A Runge-Kutta method is called* L-stable*, if it is A-stable and if*

$$\lim_{\mathrm{Re}(z)\to-\infty} R(z) = 0.$$

If a Runge-Kutta method is L-stable then the properties (3.171) as well as (3.173) are reflected in the numerical solution.
With respect to stability properties of numerical methods there exists a large number of further stability concepts like AN-, B-, BN-, D-, I-stability. For more details, we refer the reader to the literature, e.g., [38, 40, 58, 63, 82, 173, 178].
Another important aspect in the numerical treatment of DAEs is the consistency of the numerically computed solution. Although, in particular, in the case of strangeness-free semi-implicit DAEs (3.24) all constraints are given in an explicit way and therefore, the consistency of the stages X_i by use of an implicit Runge-Kutta method is guaranteed, the consistency of the numerical solution x_1 at the point t_1 cannot be guaranteed, since x_1 is given by (3.109). Therefore, Runge-Kutta methods which automatically determine consistent numerical solutions are to be preferred, since in this case no additional effort has to be performed to force the numerical solution onto the solution manifold.
Note that the original quasi-linear DAE (3.23) does in general not contain all constraints in an explicit way, but the projected-strangeness-free formulation (3.77) is strangeness-free and of semi-implicit form (3.24) and therefore, it contains all constraints including the hidden constraints as equations in the algebraic part. Therefore, by use of this regularized formulation, the stages X_i of an integration step (say from t_0 to $t_1 = t_0 + h$) are automatically consistent, i.e., they satisfy all constraints.

Proposition 3.5.87 *If an implicit Runge-Kutta method with nonsingular A satisfies one of the conditions*

$$a_{sj} = b_j, \qquad j = 1, ..., s, \tag{3.176a}$$
$$a_{i1} = b_1, \qquad i = 1, ..., s \tag{3.176b}$$

then $R(\infty) = 0$.

Proof: See [82]. □

Each of the conditions (3.176) make A-stable methods L-stable. Methods satisfying the condition (3.176a) are called *stiffly accurate* [137]. In particular, those methods are important for the numerical treatment of differential-algebraic equations, because from (3.176a) it follows that the numerical solution x_1 coincides with the last stage X_s, i.e., $x_1 = X_s$. Therefore, it can be guaranteed that the numerical solution is consistent for strangeness-free semi-implicit DAEs.

3.5.4.5 Convergence of Runge-Kutta methods applied to quasi-linear differential-algebraic equations

Due to the discretization of the DAE we introduce errors in the numerical solution which are defined as follows.

Definition 3.5.88 (Local discretization error) *The difference*

$$\delta x_h(t) = x_1 - x(t+h)$$

between the exact solution $x(t+h)$ of the differential-algebraic equations (3.2a) and the corresponding numerical solution x_1 obtained after one integration step from t to $t+h$ with exact initial values $x_0 = x(t)$ is called local discretization error.

In [79] the convergence of Runge-Kutta methods applied to semi-explicit DAEs in the form (3.15) or quasi-linear DAE (3.103) with constant leading matrix $\hat{E}(x, \hat{u}) \equiv \hat{E}$ are investigated. The considered quasi-linear DAE (3.103) or the semi-implicit DAE (3.104) are in general not in this form and the given results cannot be applied directly to the DAEs of interest. Therefore, we first consider transformations of quasi-linear DAEs to semi-explicit DAEs which conserve the numerical solution, and subsequently, we apply the results stated in [79].
In [134] results on the order of implicit Runge-Kutta methods applied to differential-algebraic equations are presented. In particular, it is shown that the obtained order of implicit Runge-Kutta methods applied to differential-algebraic equations differs from the obtained order of numerical integration methods by application to ordinary differential equations. Let us denote the obtained order by application to ODEs as the *classical order*. Therefore, Runge-Kutta methods have to be investigated with respect to several types of DAEs. This is done in [79] for semi-explicit DAEs of d-index one, two, and three. Let us recall the results obtained for Runge-Kutta methods applied to semi-explicit DAEs of the form

$$\begin{aligned} \dot{x}_1 &= k_1(x_1, x_2), && \text{(3.177a)} \\ 0 &= k_2(x_1) && \text{(3.177b)} \end{aligned}$$

of d-index 2. The following theorem is stated in [79].

Lemma 3.5.89 *Suppose that a Runge-Kutta method defined by the Butcher tableau, see Table 3.1, satisfies the conditions*

$$\begin{aligned} B(p): &\quad \sum_{i=1}^{s} b_i c_i^{k-1} = \tfrac{1}{k} \qquad \text{for } k = 1, ..., p, \\ C(q): &\quad \sum_{j=1}^{s} a_{ij} c_j^{k-1} = \tfrac{c_i^k}{k} \qquad \text{for } k = 1, ..., q \text{ and all } i = 1, ..., s \end{aligned}$$

with $p \geq q+1$ and $q \geq 1$. Then by the discretization of the semi-explicit DAE (3.177) of d-index 2 the local error $\delta x_{1h}(t)$ of the x_1-component is of magnitude

$$\delta x_{1h}(t) = \mathcal{O}(h^{q+1}), \;\; P(t)\delta x_{1h}(t) = \mathcal{O}(h^{q+2}),$$

with

$$P(t) = I - (k_{1,x_2}(k_{2,x_1}k_{1,x_2})^{-1}k_{2,x_1})(x_1(t), x_2(t)). \tag{3.178}$$

Proof: See [79]. □

Theorem 3.5.90 *Consider a system of DAEs of the form (3.177). Suppose that*

$$||(k_{2,x_1}k_{1,x_2})^{-1}|| \leq L \tag{3.179}$$

holds in a neighborhood of the solution $x(t)$ of (3.177) and that the initial values are consistent. If the Runge-Kutta matrix A, see Table 3.1, is invertible, $|R(\infty)| < 1$ (see (3.175)), and if the local error $\delta x_h(t)$ satisfies

$$\delta x_{1h}(t) = \mathcal{O}(h^r), \;\; P\delta x_{1h}(t) = \mathcal{O}(h^{r+1})$$

with P given by (3.178) then with respect to the x_1-component of (3.177) the Runge-Kutta method is convergent of order r, i.e.,

$$x_{1n} - x_1(t_n) = \mathcal{O}(h^r) \quad \text{for } t_n = nh \leq \text{const.}$$

Proof: See [79]. □

Let us formally introduce a transformation of the quasi-linear DAE (3.103) to a semi-explicit DAE of the form (3.177) of d-index 2. Based on the obtained semi-explicit DAE we will investigate the convergence of the Runge-Kutta methods applied to the quasi-linear DAE (3.103).

Lemma 3.5.91 *Suppose that the initial value problem (3.2) is a solvable DAE of d-index ν_d. Then*

$$\begin{aligned} \dot{x} &= y, \\ 0 &= F(x, y, t), \end{aligned}$$

together with the initial values $x(t_0) = x_0$ and $y(t_0) = y_0 = \dot{x}(t_0)$ is a semi-explicit DAE of d-index $\nu_d + 1$ with the same solution for $x(t)$ as in (3.2).

Proof: See [25]. □

Remark 3.5.92 As a rule of thumb, it is mentioned in [25] that the semi-explicit case of a given d-index ν_d is much like the general case of d-index $\nu_d - 1$, i.e., a nonlinear DAE of d-index one behaves like a semi-explicit DAE of d-index two. □

Lemma 3.5.93 *The initial value problem of the quasi-linear DAE (3.103) with initial values $x(t_0)$ for $t \in \mathbb{I} = [t_0, t_f]$ is equivalent to the initial value problem of the autonomous quasi-linear DAE*

$$\begin{bmatrix} \hat{E}(y_1, \hat{u}(y_2)) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \end{bmatrix} = \begin{bmatrix} \hat{k}(y_1, \hat{u}(y_2)) \\ 1 \end{bmatrix} \tag{3.180}$$

with initial values

$$y_1(t_0) = x(t_0), \; y_2(t_0) = t_0$$

for $t \in \mathbb{I} = [t_0, t_f]$ in the sense that the solution $y_1(t)$ of (3.180) is identical to the solution $x(t)$ of (3.103).

Proof: With $y_1(t) = x(t)$, $y_2(t) = t$ the assertion follows immediately. □

Lemma 3.5.94 *Suppose that the initial value problem (3.103) is a solvable DAE of d-index $\nu_d = 1$ and that the matrix $\hat{E}(x, \hat{u})$ has constant rank $r_{\hat{E}} = \operatorname{rank}(\hat{E}(x, \hat{u}))$ for all possible $(x, \hat{u})$. Let $y = \begin{bmatrix} y_1^T & y_2^T \end{bmatrix}^T = \begin{bmatrix} \begin{bmatrix} x_1^T & x_2^T \end{bmatrix} & t \end{bmatrix}^T \in \mathbb{R}^{n+1}$ and a decomposition of the matrix $\hat{E}(y_1, \hat{u}(y_2))$ be given by*

$$\begin{aligned} \hat{E}(y_1, \hat{u}(y_2)) &= S^{-1}(y_1, y_2) \begin{bmatrix} I_{r_{\hat{E}}} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} T_{11}(y_1, y_2) & T_{12}(y_1, y_2) \\ T_{21}(y_1, y_2) & T_{22}(y_1, y_2) \end{bmatrix}, \\ S(y_1, y_2) &= \begin{bmatrix} S_1(y_1, y_2) \\ S_2(y_1, y_2) \end{bmatrix}, \\ x &= \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \mathbb{R}^n, \; x_1 \in \mathbb{R}^{r_{\hat{E}}}, \; x_2 \in \mathbb{R}^{n - r_{\hat{E}}}. \end{aligned} \tag{3.181}$$

Then the DAE

$$\begin{aligned} \dot{y} &= \hat{f}(y,z), && \text{(3.182a)}\\ 0 &= \hat{g}(y) && \text{(3.182b)} \end{aligned}$$

with

$$\begin{aligned} \hat{f}(y,z) &= \begin{bmatrix} T_{11}^{-1}(y_1,y_2)\big(S_1(y_1,y_2)\hat{k}(y_1,\hat{u}(y_2)) - T_{12}(y_1,y_2)z\big) \\ z \\ 1 \end{bmatrix}, && \text{(3.182c)}\\ \hat{g}(y) &= S_2(y_1,y_2)\hat{k}(y_1,\hat{u}(y_2)) && \text{(3.182d)} \end{aligned}$$

together with the initial values $y(t_0) = \begin{bmatrix} x_1^T(t_0) & x_2^T(t_0) & t_0 \end{bmatrix}^T$ *and* $z(t_0) = \dot{x}_2(t_0)$ *is a semi-explicit DAE (3.15) of d-index* $\nu_d = 2$ *with the same solution for* x *as in (3.103).*

Proof: From Lemma 3.5.93 we get that the solution $y_1(t)$ of (3.180) is identical to the solution $x(t)$ of (3.103).
Furthermore, we get with (3.181) from (3.180) that

$$\begin{bmatrix} T_{11}(y_1,y_2) & T_{12}(y_1,y_2) & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{y}_{11} \\ \dot{y}_{12} \\ \dot{y}_2 \end{bmatrix} = \begin{bmatrix} S_1(y_1,y_2)\hat{k}(y_1,\hat{u}(y_2)) \\ S_2(y_1,y_2)\hat{k}(y_1,\hat{u}(y_2)) \\ 1 \end{bmatrix}.$$

By introducing $z = \dot{y}_{12}$ it follows that

$$\begin{bmatrix} T_{11}(y_1,y_2) & 0 & 0 & 0 \\ 0 & I_{n-r_{\hat{E}}} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{y}_{11} \\ \dot{y}_{12} \\ \dot{y}_2 \\ \dot{z} \end{bmatrix} = \begin{bmatrix} S_1(y_1,y_2)\hat{k}(y_1,\hat{u}(y_2)) - T_{12}(y_1,y_2)z \\ z \\ 1 \\ S_2(y_1,y_2)\hat{k}(y_1,\hat{u}(y_2)) \end{bmatrix}.$$

Since the matrix $T(y_1,y_2)$ is nonsingular, apart from permutation of columns we may assume that $T_{11}(y_1,y_2)$ is nonsingular. Hence, we get the assertion by multiplication of the first block row with $T_{11}^{-1}(y_1,y_2)$. □

Remark 3.5.95 The DAE (3.182) is called *augmented semi-explicit DAE*. In terms of x and $y_2 = t$ it has the form

$$\begin{aligned} \dot{x}_1 &= T_{11}^{-1}(x,y_2)\big(S_1(x,y_2)\hat{k}(x,\hat{u}(y_2)) - T_{12}(x,y_2)z\big), && \text{(3.183a)}\\ \dot{x}_2 &= z, && \text{(3.183b)}\\ \dot{y}_2 &= 1, && \text{(3.183c)}\\ 0 &= S_2(x,y_2)\hat{k}(x,\hat{u}(y_2)). && \text{(3.183d)} \end{aligned}$$

□

Lemma 3.5.96 *Suppose that the quasi-linear DAE (3.103) is a solvable DAE. Then the direct discretization of the quasi-linear DAE (3.103) and of the augmented semi-explicit DAE (3.182) using Runge-Kutta methods defined by the Butcher tableau in Table 3.1, yields the same numerical solution* x_i *as approximation of* $x(t_i)$.

Proof: The direct discretization of (3.182), i.e., of (3.183), leads to

$$\begin{aligned}
X'_{1i} &= T_{11}^{-1}(X_i, Y_{2i})(S_1(X_i, Y_{2i})\hat{k}(X_i, \hat{u}(Y_{2i})) - T_{12}(X_i, Y_{2i})Z_i), && \text{(3.184a)}\\
X'_{2i} &= Z_i, && \text{(3.184b)}\\
Y'_{2i} &= 1, && \text{(3.184c)}\\
0 &= S_2(X_i, Y_{2i})\hat{k}(X_i, \hat{u}(Y_{2i})), && \text{(3.184d)}\\
X_i &= x_0 + h\sum_{j=1}^{s} a_{ij}X'_j, && \text{(3.184e)}\\
Y_{2i} &= y_{20} + h\sum_{j=1}^{s} a_{ij}Y'_{2j}, && \text{(3.184f)}\\
Z_i &= z_0 + h\sum_{j=1}^{s} a_{ij}Z'_j, && \text{(3.184g)}
\end{aligned}$$

for $i = 1, ..., s$. We did already substitute $Y_{1i} = X_i = \begin{bmatrix} X_{1i}^T & X_{2i}^T \end{bmatrix}^T$ and $Y'_{1i} = X'_i = \begin{bmatrix} (X'_{1i})^T & (X'_{2i})^T \end{bmatrix}^T$. From $y_{20} = t_0$, (3.184c) and (3.184f) with (3.107) we get $Y_{2i} = t_0 + c_i h$. Furthermore, replacing Z_i by X'_{2i} in (3.184a) we get from (3.184a), (3.184d), and (3.184e)

$$\begin{aligned}
T_{11}(X_i, t_0+c_ih)X'_{1i} + T_{12}(X_i, t_0+c_ih)X'_{2i} &= S_1(X_i, t_0+c_ih)\hat{k}(X_i, \hat{u}(t_0+c_ih)),\\
0 &= S_2(X_i, t_0+c_ih)\hat{k}(X_i, \hat{u}(t_0+c_ih)),\\
X_i &= x_0 + h\sum_{j=1}^{s} a_{ij}X'_j
\end{aligned}$$

independent of Y_{2i} and Z_i, which is equivalent to

$$\begin{aligned}
&\begin{bmatrix} S_1(X_i, t_0 + c_ih) \\ S_2(X_i, t_0 + c_ih) \end{bmatrix}^{-1} \begin{bmatrix} I_{r_E} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} T_{11}(X_i, t_0 + c_ih) & T_{12}(X_i, t_0 + c_ih) \\ T_{21}(X_i, t_0 + c_ih) & T_{22}(X_i, t_0 + c_ih) \end{bmatrix} X'_i\\
&\quad = \hat{k}(X_i, \hat{u}(t_0 + c_ih)),\\
X_i &= x_0 + h\sum_{j=1}^{s} a_{ij}X'_j.
\end{aligned}$$

With (3.181) these equations correspond exactly to the direct discretization (3.108) of the quasi-linear DAE (3.23), independent of Y_{2i}, Y'_{2i}, Z_i and Z'_i. □

Corollary 3.5.97 *The numerical approximation x_k of $x(t_k)$ by use of Runge-Kutta methods is invariant under the transformation of the DAE (3.103) to its augmented semi-explicit DAE (3.182), i.e., the numerical integration yields the same solution for the x-component.*

Proof: The proof follows immediately from Lemma 3.5.96. □

Theorem 3.5.98 *Suppose that the initial value problem (3.103) is a solvable DAE and the matrix $\hat{E}(x, \hat{u})$ has constant rank $r_{\hat{E}} = \operatorname{rank}(\hat{E}(x, \hat{u}))$ for all possible $(x, \hat{u})$. Furthermore, suppose that there exists a nonsingular matrix*

$$S(x, \hat{u}) = \begin{bmatrix} S_1(x, \hat{u}) \\ S_2(x, \hat{u}) \end{bmatrix}$$

with $S_1(x,\hat{u})$ is of size $r_{\hat{E}} \times \hat{m}$ such that

$$\| \begin{bmatrix} S_1(x,\hat{u})\hat{E}(x,\hat{u}) \\ (S_2(x,\hat{u})\hat{k}(x,\hat{u}))_{,x} \end{bmatrix}^{-1} \| \leq L \tag{3.185}$$

holds in a neighborhood of the solution $(x(t),\hat{u}(t))$ of (3.103) and that the initial values are consistent. If the Runge-Kutta matrix A is invertible, $|R(\infty)| < 1$ (see (3.175)) and if the local error $\delta x_h(t)$ satisfies

$$\delta x_h(t) = \mathcal{O}(h^r),\ P\delta x_h(t) = \mathcal{O}(h^{r+1})$$

with $P(x)$ given by (3.178), then the Runge-Kutta method (3.108) is convergent of order r, i.e.,

$$x_n - x(t_n) = \mathcal{O}(h^r) \quad \textit{for } t_n = nh \leq \text{const.}$$

Proof: With Lemma 3.5.3 it follows from (3.185) that the quasi-linear DAE (3.103) has d-index $\nu_d = 1$ or $\nu_d = 0$. Therefore, from Lemma 3.5.94 with (3.181) we have $S_1\hat{E} = \begin{bmatrix} T_{11} & T_{12} \end{bmatrix}$ and we get from (3.185) that

$$\| \begin{bmatrix} T_{11} & T_{12} \\ (S_2\hat{k})_{,x_1} & (S_2\hat{k})_{,x_2} \end{bmatrix}^{-1} \| \leq L.$$

With T_{11} nonsingular we get

$$\|T_{11}^{-1}\| \leq L_1 \ \text{ and } \ \|((S_2\hat{k})_{,x_2} - (S_2\hat{k})_{,x_1}T_{11}^{-1}T_{12})^{-1}\| \leq L_2.$$

The second inequality is equivalent to

$$\|(\begin{bmatrix} (S_2\hat{k})_{,x_1} & (S_2\hat{k})_{,x_2} & (S_2\hat{k})_{,\hat{u}}\dot{\hat{u}} \end{bmatrix} \begin{bmatrix} -T_{11}^{-1}T_{12} \\ I \\ 0 \end{bmatrix})^{-1}\| \leq L_2$$

which corresponds exactly to the condition (3.179) in Theorem 3.5.90 respecting the DAE (3.103) in its augmented form (3.182).
Hence, the proof follows from Theorem 3.5.90 applied to the augmented semi-explicit DAE (3.182), since the numerical solution of the semi-implicit DAE (3.104) and its augmented semi-explicit DAE (3.182) produces the same result, see Lemma 3.5.96. □

From the convergence results in [79] applied to semi-explicit DAEs of d-index 2, we get the convergence results listed in Table 3.2 for Runge-Kutta methods applied to the quasi-linear DAE (3.103) of d-index $\nu_d = 1$. Compare with Table 2.2 in [79].

3.5.4.6 Important classes of Runge-Kutta methods

Certain classes of implicit Runge-Kutta methods are well investigated and very popular in the numerical integration of DAEs. In particular, these are *Gauß methods*, *Radau methods*, and *Lobatto*[13] *methods*.
Gauß methods are collocation methods based on the Gauß-quadrature formulas, see [82, 172]. The s-stage Gauß method is of classical order $2s$ but the numerical integration of strangeness-free quasi-linear DAEs in general leads to a convergence rate of only $s+1$ for odd s and a convergence rate of s for even s, see Table 3.2. Furthermore, the Gauß methods are A-stable and B-stable but they are not L-stable. From the choice of the nodes, i.e., of the parameters c_i, it follows that no stage

[13] Rehuel Lobatto (born 1797 in Amsterdam, Netherlands - died 1866 in Delft, Netherlands)

X_i coincides with the approximation x_1 at the end of the current integration step. Rather, the approximation x_1 is determined as linear combination of the stages X_i, see (3.109). While the consistency of the stages of the Gauß method applied to strangeness-free semi-implicit DAEs of the form (3.104) can be guaranteed, the consistency of the approximation x_1 is in general not insured. This facts makes the Gauß methods impracticable for the numerical integration of differential-algebraic equations.

The s-stage Radau methods are of classical order $2s-1$ which is slightly lower than the classical order of the Gauß method. But the order of convergence rate of Radau IIa methods for strangeness-free quasi-linear differential-algebraic equations is still $2s-1$ which is, in general, larger than the convergence rate of the Gauß methods for strangeness-free semi-implicit DAEs. Furthermore, the Radau methods have a large number of advantages in contrast to the Gauß methods. The stability properties of Radau methods are excellent, i.e., Radau methods are A-, B- as well as L-stable. In particular, the L-stability is of great importance for the numerical treatment of differential-algebraic equations, see Section 3.5.4.4. Furthermore, for the node vector c of the Radau Ia methods we have that $c_1 = 0$. This means that the first stage X_1 coincides with the initial point of the current integration step, i.e., $x_0 = X_1$. On the other hand, with respect to Radau IIa methods we have that $c_s = 1$ and furthermore, $a_{si} = b_i$ for $i = 1, ..., s$, i.e., Radau IIa methods are stiffly accurate, see Section 3.5.4.4. In particular, we have that the last stage X_s coincides with the final point of the current integration step, i.e., $x_1 = X_s$. From this fact the consistency of the approximation x_i obtained from a Radau Ia and IIa method applied to strangeness-free semi-implicit DAEs of the form (3.104) follows from the consistency of the stages which is insured because of the explicit occurrence of all constraints. The simplest representative of the Radau IIa methods is the implicit Euler method.

The s-stage Lobatto IIIa, IIIb, and IIIc methods are of classical order $2s-2$, and the numerical integration of strangeness-free quasi-linear differential-algebraic equations lead to the same rate of convergence, see Table 3.2. Lobatto IIIa, IIIb, and IIIc methods are A-stable but not L-stable, while Lobatto IIIa and IIIc methods are in addition stiffly accurate. For the numerical treatment of mechanical systems Lobatto methods are investigated in [151, 152].

Method	stages	rate of convergence	stability
Gauß	$s \begin{cases} \text{odd} \\ \text{even} \end{cases}$	$\begin{cases} s+1 \\ s \end{cases}$	A-, B-stable
Radau Ia	s	s	A-, B-, L-stable
Radau IIa	s	$2s-1$	A-, B-, L-stable
Lobatto IIIa	s	$2s-2^*$	A-stable
Lobatto IIIc	s	$2s-2$	A-stable

*) Result for Lobatto IIIa is proven for $s = 2, 3$ and conjectured for larger s (see [79])

Table 3.2: Properties of implicit Runge-Kutta methods applied to strangeness-free semi-implicit DAEs

These facts make Radau methods and, in particular, the Radau IIa methods excellent candidates for the numerical integration of initial value problems for strangeness-free quasi-linear differential-algebraic equations of form (3.104).

3.6 Numerical methods and software: an overview

The numerical treatment of initial value problems for ordinary differential equations of the form (3.1) is well developed, see [34, 35, 63, 74, 75, 81, 161, 173]. Thus, there exists a large collection of efficient algorithms which are suitable for the numerical integration of ordinary differential equations. As discussed in Sections 3.4 and 4.6, the numerical treatment of differential-algebraic equations, in particular, of the model equations of multibody systems, is more difficult and should combine a discretization scheme with suitable regularization techniques, because of the order reduction phenomenon, drift-off effects, or instabilities caused by the higher index of DAEs.

Therefore, not all numerical methods that are suitable for ordinary differential equations are also suitable for the numerical treatment of differential-algebraic equations. In [25, 77, 82] suitable methods for the numerical treatment of differential-algebraic equations are investigated. In particular, the drift-off phenomenon and therefore, the consistency of the approximate solution plays an important role in the accuracy and stability of numerical algorithms.

With respect to the numerical treatment of differential-algebraic equations implicit or half implicit numerical methods are of great interest. Therein, three classes of methods play a major role. These are multistep methods, one step methods, and extrapolation methods.

For multistep methods, discretization and implementation is not very difficult. Also, the solution of the nonlinear systems arising in every integration step is relatively cheap. On the other hand, the development of a step size control is complicated and this control is not very flexible in its choice of the step sizes, see [70, 71]. Furthermore, the convergence behavior of multistep methods with variable step size for nonlinear DAEs is not completely understood, not even for quasi-linear DAEs of the form (3.103) with state depending leading matrix. Very popular representatives of multistep methods are the BDF methods. But, they have the disadvantage that they are A-stable only for $k \leq 2$, A(α)-stable only for $k \leq 6$, and may not be stable for $k > 6$, see [81, 173].

For implicit Runge-Kutta methods, as representative for one step methods, the discretization of differential-algebraic equations and its implementation is very technical and the solution of the nonlinear systems arising in every integration step is expensive. On the other hand, the development of a suitable and efficient step size control is easy and the control is very flexible in the choice of the step sizes, see [39, 63, 78, 82, 169], with respect to the accuracy of the numerical solution and to the convergence of the Newton iteration method for solving the nonlinear systems. Furthermore, the convergence behavior of implicit Runge-Kutta methods at least for quasi-linear DAEs of the form (3.103) with state depending leading matrix is well understood, see Section 3.5.4.5.

The available software is mostly suited for special classes of DAEs and does often not contain a regularization technique. Therefore, general DAEs, in particular, higher index DAEs, first have to be regularized and the regularized form has to be provided to the numerical algorithms. In the following we give an an overview over available and commonly used numerical algorithms for the numerical integration of DAEs.

The software package `DASSL`[14] [25, 135] is based on backward differentiation formulas with step size and order control. It is designed to integrate nonlinear differential-algebraic equations of the form (3.2a) of d-index one. This code is widely used and works efficiently for nonstiff systems. Furthermore, the numerical algorithm `DASPK`[15] [26, 177] is designed for large scale DAEs. It also bases on backward dif-

[14] `DASSL - http://www.engineering.ucsb.edu/~cse/software.html`

[15] `DASPK - http://www.engineering.ucsb.edu/~cse/software.html`

ferentiation formulas with step size and order control but in contrast to DASSL the numerical solution of the arising linear systems is done via iterative methods. The latest version of DASPK includes in addition a sensitivity analysis and root finding. It is known that BDF methods are suitable for the numerical integration of DAEs (3.2a) with d-index at most one, see [25]. Unfortunately, the direct discretization of higher index problems by use of BDF methods may lead to wrong results or may not produce a solution at all.

GELDA[16] [107] is a software package for the numerical integration of linear DAEs (3.3) with variable coefficients. While most of the standard integration methods are suitable only for regular strangeness-free DAEs, i.e., for DAEs of d-index not higher than one, GELDA is suitable for the numerical integration of linear DAEs of arbitrary index. The implementation of GELDA is based on the combination of regularization, see Section 3.4.2, which transforms the linear DAE (3.3) into a linear equivalent strangeness-free DAE (3.20) with the same solution set, followed by discretization of the regularized strangeness-free DAE by use of the Runge-Kutta scheme RADAU5 [79, 82] as well as the BDF method DASSL [25, 135]. The user has to provide the necessary number of derivatives of the whole system matrices.

A nonlinear version of GELDA is available and called GENDA[17] [109]. GENDA is suitable for general nonlinear DAEs (3.2a) of arbitrarily high index. The code GENDA combines the regularization technique reviewed in Section 3.4.2 with the discretization of an equivalent strangeness-free formulation (3.21) of the DAE by use of BDF methods. The user has to provide the necessary number of derivatives of the whole DAE (3.2a), in particular, the whole derivative array $\mathfrak{F}_{\nu_s}$ of level ν_s, and in addition the characteristic quantities. For more details we refer to [103, 109].

Both numerical algorithms GELDA and GENDA are included in a recently developed MATLAB[18] toolbox, see [108].

The subroutine LSODI[19] [90] is designed to solve initial value problems in quasi-linear form (3.103) with $u(t) = t$ by use of backward differentiation formulas for the discretization.

RADAU5[20] [79, 82] is designed for the direct numerical integration of quasi-linear DAEs of the form

$$E\dot{x} = k(x, u) \tag{3.186}$$

where E a constant, possibly singular matrix. It is based on the implicit Runge-Kutta method of Radau IIa type of order 5. In addition to RADAU5 the numerical algorithm RADAUP[21] is avaliable. It bases on the implicit Radau IIa method of order 5, 9, or 13. In [79] and [82] it has been proved that convergence of the Radau IIa method can be guaranteed for semi-explicit DAEs of d-index 1 and d-index 2. In addition, convergence for this method applied to semi-explicit DAEs in Hessenberg form (3.94) of d-index 3 is investigated in [79]. Furthermore, it is shown in [79] how to transform the DAE (3.186) into semi-explicit form (3.15) and, moreover, that Runge-Kutta methods are invariant under this transformation, see also Section 3.5.4.5. Because of this invariance, the convergence results proposed in [79, 82] are valid for semi-explicit DAEs (3.15) as well as quasi-linear DAEs (3.186) with constant leading matrix. Therefore, RADAU5 allows the numerical integration of quasi-linear DAEs (3.186) up to a differentiation index 3.

However, the application of RADAU5 to quasi-linear DAEs of the form (3.103) is not possible because of the state depending leading matrix $E(x)$ which is in general not

[16] GELDA - http://www.math.tu-berlin.de/numerik/mt/NumMat/Software/GELDA/
[17] GENDA - http://www.math.tu-berlin.de/numerik/mt/NumMat/Software/GENDA/
[18] MATLAB - http://www.mathworks.com/
[19] LSODI - http://www.netlib.org/alliant/ode/prog/lsodi.f
[20] RADAU5 - http://www.unige.ch/∼hairer/software.html
[21] RADAUP - http://www.unige.ch/∼hairer/software.html

constant. A way out is the introduction of new variables and the transformation to a semi-explicit DAE of the form

$$\begin{aligned} \dot{x} &= y, \\ 0 &= E(x,u)y - k(x,u). \end{aligned}$$

Unfortunately, this increases the d-index by one, see [25], but Runge-Kutta methods are still applicable if the d-index of the original quasi-linear DAE does not exceed 2, see [79].

`LIMEX`[22] [44, 45] and `SEULEX`[23] [82] are extrapolation integrators for the solution of quasi-linear DAEs of the form (3.186) with $u(t) = t$ and d-index lower or equal one. These discretization methods base on the linearly implicit Euler discretization combined with extrapolation. The codes offer the possibility to compute consistent initial values and dense output. Both codes use the same numerical method, but a different implementation.

Another approach to integrate the system of differential-algebraic equations is the so-called *dynamic iteration* or *waveform relaxation*. In [123, 124] the convergence of the dynamic iteration for the analytical solution of the subsystems in every iteration step is investigated. In addition, in [20] the convergence of the dynamic iteration concerning the numerical solution of the subsystems in form of ODEs by use of Runge-Kutta methods in every iteration step is investigated. Furthermore, in [8, 49] the dynamic iteration applied to coupled strangeness-free DAEs is discussed.

[22] `LIMEX - http://www.zib.de/Numerik/numsoft/CodeLib/ivpode.en.html`
[23] `SEULEX - http://www.unige.ch/~hairer/software.html`

Chapter 4

Multibody Systems

The dynamical behavior of *multibody systems* (MBS) is of great importance in many fields of mechanical engineering, like robotics, road and rail vehicle construction, air and space craft design, see [52, 86, 154, 155, 156]. Furthermore, today the need and the demand for mechanical systems with high speed motions is increasing, e.g., the desire for high speed cars, high speed trains, high speed aircraft, high speed ferries, or high speed elevators. Because of this high speed motion of mechanical systems effects arising from huge inertial or centrifugal forces cannot be neglected.
From the dynamical point of view a multibody system is regarded as a number of mass points and rigid or elastic bodies, subject to possibly existing interconnections and constraints of various kinds, e.g., joints, springs, dampers, and actuators. In particular, rigid bodies are regarded as a collection of particles which are kept at invariant distances of each other.
As a basis for the investigations of the dynamical behavior of multibody systems, the *equations of motion* (EoM) provide the tool for modeling the relevant dynamical properties. The equations of motion as well as the necessary derivatives of the constraints with respect to t can be generated in a systematic way by *multibody formalisms* based on the principles of classical mechanics [53, 112, 149, 154]. The equations of motion usually form a nonlinear system of differential-algebraic equations (DAEs) with a very special structure that can and should be exploited in the numerical solution [52, 82].
The efficient and robust numerical integration of these equations is a challenging problem in the development of simulation packages, because dynamical simulation is a frequently used and one of the most time consuming analysis methods for MBS models. In this chapter we will discuss properties and tools which are important for the efficient numerical integration of the equations of motion. We will not restrict ourselves to classical constrained mechanical systems but consider the more complex model equations that are currently used in state-of-the-art multibody system simulation packages.
We will mainly consider multibody systems from the dynamical point of view. In Section 4.1 we will discuss the modeling procedure of MBS. In a large part of literature, equations of motion in standard form including the dynamical equations of motion subject to some holonomic constraints are discussed in detail. But, in industrial applications more complex equations arise which include friction effects, contact force laws, dynamical force elements, nonholonomic constraints, and in some cases the existing constraints are redundant. Therefore, we will focus our investigation on the most general case that includes all of these features. Furthermore, we will discuss solution invariants of several mechanical systems, and we will give a classification of several forms of equations of motion in form of modeling levels.
In Section 4.2 we will consider equations of motion with possibly redundant con-

straints with respect to their dynamical properties and, in particular, with respect to the existence and partial uniqueness of solutions.
It is well known that the direct numerical integration of the equations of motion leads to several numerical difficulties due to the higher index of the equations of motion. Therefore, we will discuss the numerical properties of several formulations of the equations of motion in Section 4.4. In particular, we will discuss the drift-off phenomenon.
Based on the obtained results, in Section 4.5 we will postulate two paradigms, one for the modeling of multibody systems and one for numerical integration methods for the model equations of dynamical systems.
As mentioned in the previous paragraphs, numerical integration of equations of motion is a nontrivial problem such that a numerical method should combine a discretization method with a suitable regularization technique. Therefore, we will review some important regularization techniques in Section 4.6 and extend them to the general form of the equations of motion investigated in this thesis. Furthermore, we will introduce a new regularization technique for the equations of motion, following a novel approach that has originally been developed for the time integration of general nonlinear differential-algebraic equations of arbitrarily high index.
Concluding, we will give a short overview of numerical integration methods designed for certain types of equations of motion in Section 4.7 and we will discuss their applicability.

4.1 Modeling of multibody systems

In this section we will briefly discuss the modeling of multibody systems. For more details the reader is referred to the large variety of literature on this topic, e.g., [1, 12, 17, 48, 57, 83, 147, 148, 174, 180].

4.1.1 Free motion of multibody systems

For a multibody system, let a set of generalized coordinates or *position variables* p, depending on t, be given such that the position (and the orientation) of all bodies is uniquely specified. The classical approach (see, e.g., [57]) for the modeling and simulation of multibody systems is using *minimal coordinates* p. In this approach a minimal number of coordinates is used to describe the motion of the multibody system. Possible kinematic constraints, for example those resulting from joints, which restrict the motion are eliminated. Other frequently used aproaches base on the *relative coordinates*, on the *absolute coordinates*, or both, i.e., *mixed coordinates*. While the absolute coordinates describe the position and orientation of the several bodies with respect to a fixed inertial coordinate system, the relative coordinates describe the position and orientation of a body relatively to an other body, for instance the distances and angles between different bodies.
Note that the choice of the coordinates p is not restricted to absolute coordinates, but the choice of relative coordinates is possible and often advantageous, in particular, in the investigation of more complex multibody systems. In the case of a free motion, it is necessary that the coordinates are nonredundant, i.e., each of them is freely selectable, independent of other coordinates.
Let the coordinate vector p consist of arbitrary coordinates which describe the positions of all bodies involved in the multibody system. Let the derivative with respect to t of the position variables $\dot{p} = \mathrm{d}p/\mathrm{d}t$ denote the *generalized velocity* of the MBS. Furthermore, let us assume that these positions are unrestricted in their choice. Let $T(p, \dot{p})$ denote the *kinetic energy* which is assumed to be given as a function of the arguments p and $\dot{p}$. From the definition, see [17, 180], it follows that the kinetic

energy $T(p, \dot{p})$ of a multibody system is in general nonnegative. In particular, if only masses with a positive value are allowed then the kinetic energy is strictly positive for $\dot{p} \neq 0$. Therefore, $T(p, \dot{p})$ is a positive semi-definite (positive definite in case of nonvanishing masses) quadratic form in $\dot{p}$.
The acting force on a moving mass point of a multibody system in $\mathbb{R}^3$ can be described by a vector function

$$\varphi(q) \ : \ \mathbb{R}^3 \to \mathbb{R}^3$$

which describes a field of forces. Here, $q \in \mathbb{R}^3$ denotes the position of the mass point. A special field of forces is the so-called *conservative field of forces*, which has the property that the work done by the forces of the field only depends on the initial and the final configuration of the multibody system but is independent of the path between both configurations. It satisfies the condition

$$\frac{\partial \varphi_i}{\partial q_j} - \frac{\partial \varphi_j}{\partial q_i} = 0$$

for $i, j = 1, 2, 3$. Following an idea of Lagrange, the force $\varphi(q)$ can be expressed as the gradient of a scalar function $U(q) \ : \ \mathbb{R}^3 \to \mathbb{R}$, i.e.,

$$\varphi(q) = -\frac{\partial U(q)}{\partial q}.$$

Following Green[1] and Gauß the scalar function U is called *potential* or *potential energy*, see [148]. Gravity is a conspicuous example for a conservative field of forces. The work to move a body from the height h_0 to h_1 is $-mg(h_1 - h_0)$ done by gravity, where m is the mass of the body and g is the gravitational acceleration. This work does not depend on the path by which the body is moved from one height to the other.
If the multibody system is influenced by a conservative field, we have to take the potential energy into account. Let the potential energy of the whole multibody system be denoted by $U(p)$. Mechanical systems which are only influenced by forces of conservative fields of forces are called *conservative mechanical systems*.
The functions $T(p, \dot{p})$ and $U(p)$ are functions in p and $\dot{p}$ and are known from the configuration of the multibody system.
The motion of an arbitrary conservative mechanical system described by the position variables p and the generalized velocities $\dot{p}$ satisfies the *Hamilton principle of the least action*, e.g., see [17, 148, 174, 180], which reads as

$$\int_{t_0}^{t_f} T(p(t), \dot{p}(t)) - U(p(t)) \, \mathrm{d}t = \text{extremal}. \tag{4.1}$$

This integral is called *action integral*, with the *Lagrangian* or *Lagrange function*

$$L(p(t), \dot{p}(t)) = T(p(t), \dot{p}(t)) - U(p(t)). \tag{4.2}$$

By use of techniques from variational calculus, see [96], from (4.1) we get a necessary condition for the extremity in the form

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{p}}\right) - \frac{\partial L}{\partial p} = 0. \tag{4.3}$$

These equations are called the *Euler equations* see [36, 55, 148].
In this thesis, we will not restrict our investigations to multibody systems influenced

[1] Georg Green (born 1793 in Nottingham, England - died 1841 in Sneinton, England)

by a conservative field of forces. Rather, we take into account work which is done by other forces that we call *external forces*. Let us denote the work done by the external forces for an arbitrary displacement ∂p by $Q(p,t)\partial p$. In the case of free motion, the external forces contain *applied forces* $Q_a(p,t)$ only, i.e., $Q(p,t) = Q_a(p,t)$. Then the equations of motion which determine the motion of the multibody system may be written as

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{p}}\right) - \frac{\partial L}{\partial p} = Q(p,t). \tag{4.4}$$

From (4.4), one then obtains the equations of motion in *state space form*

$$M(p)\ddot{p} = \widetilde{f}(p,\dot{p},t) + Q(p,t) \qquad (= f(p,\dot{p},t)). \tag{4.5}$$

The matrix $M(p) = T_{,\dot{p}\dot{p}}$ is called the mass matrix. Since the kinetic energy $T(p,\dot{p})$ is a positive semi-definite quadratic form in $\dot{p}$, the mass matrix is positive semi-definite. If only masses with a positive value are allowed the mass matrix is actually strictly positive definite. and does not depend on $\dot{p}$. The right-hand side $f(p,\dot{p},t)$ of (4.5) contains gravitational forces and gyroscopic forces (both contained in $\widetilde{f}(p,\dot{p},t)$) as well as the external forces $Q(p,t)$.
The state space form of the equations of motion (4.5) is an ordinary differential equation (ODE) in implicit form of second order that can be reduced to a first order ODE in implicit form. Then, it can be treated in the usual way according to the theory and numerical methods for ODEs. From the numerical analysis point of view, the numerical integration of the equations of motion in state space form is well developed and there exists a large collection of efficient and robust solvers for nonstiff as well as for stiff ODEs, e.g., see [28, 34, 63, 74, 75, 81, 82, 161].

Example 4.1.1 The mathematical pendulum: A mathematical pendulum of length $L > 0$ represents a point mass which moves without friction along a vertical circle of radius L under gravity, see Figure 4.1 and [13]. Let us use the angle $\varphi(t)$ to

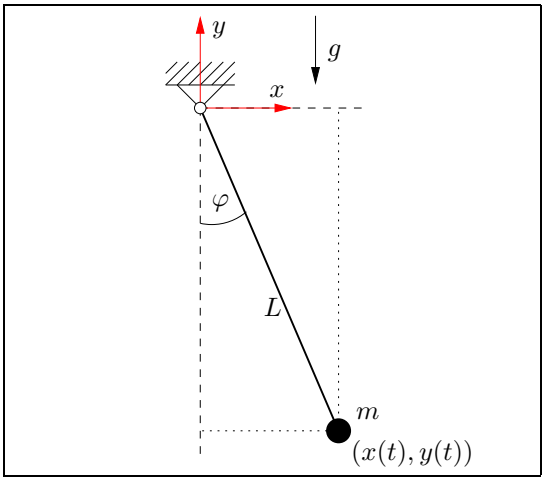

Figure 4.1: Topology of the mathematical pendulum

describe the configuration depending on t. The kinetic energy is given by $T(\varphi,\dot{\varphi}) = \frac{1}{2}mL^2\dot{\varphi}^2$. Furthermore, the potential energy is given by $U(\varphi) = -mgL\cos(\varphi)$. By use of the Lagrange function (4.2) $L(\varphi,\dot{\varphi}) = \frac{1}{2}mL^2\dot{\varphi}^2 + mgL\cos(\varphi)$, from the Euler equations (4.3) we get

$$L\ddot{\varphi} = -g\sin(\varphi)$$

as the equations of motion in state space form. □

The determination of the state space form plays a key role in the classical approach. In [52] it is described how to determine and to compute the state space form for linear or linearized equations of motion. Furthermore, the determination and computation of the state space form of the general equations of motion is considered in [60] and [147]. Here, it becomes clear, that for general nonlinear mechanical systems, especially those with closed loops, the state space form can only be established locally, and that the complete process of reduction to state space form is quite laborious and therefore, very time consuming. For instance the example of a wheel set [11] features so-called closed kinematical loops. This example cannot be modeled as a system of ordinary differential equations in minimal coordinates.

4.1.2 Constraint motion of multibody systems

As discussed in the previous section, in the case of a free motion it is necessary that the coordinates are nonredundant. Unfortunately, a choice of nonredundant coordinates p is in general very difficult, very technical, or not possible, in particular, if multibody systems with kinematical closed loops are considered. Therefore, in modern approaches (see e.g. [52, 53, 149, 154, 155, 156, 164]), multibody systems are modeled by use of nonminimal coordinates in connection with constraints.
In this section we will discuss the modeling of constrained multibody systems in this way. We have to distinguish between two important types of constraints. Let us illustrate these with a simple example.

Example 4.1.2 The rolling ball: Let us consider the motion of a ball with given radius r, see Figure 4.2. The position coordinates are subject to constraints arising from the fact that the ball has to move while being in contact with a given two-dimensional plane in the three dimensional space, e.g., see [17, 129, 180]. The

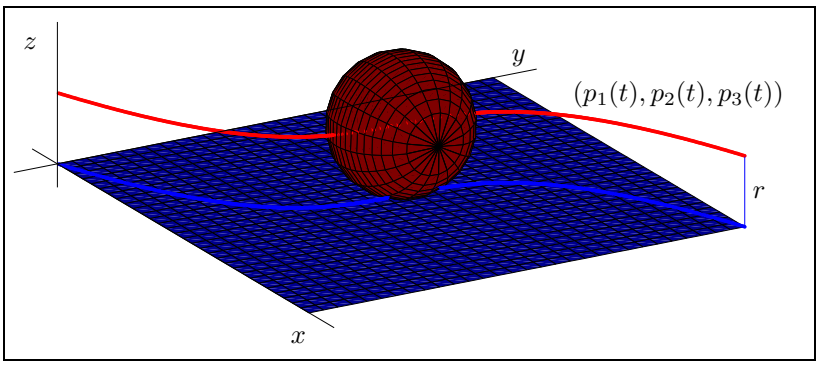

Figure 4.2: Rolling ball on surface

configuration is described by three position coordinates, say $p_1(t)$, $p_2(t)$, and $p_3(t)$, and three orientation coordinates, say $p_4(t)$, $p_5(t)$, and $p_6(t)$. The ball can take any position, as long as it is in contact with the plane. Therefore, the position is restricted by a constraint of the form

$$0 = g(p, t). \tag{4.6}$$

In addition to this constraint it is obvious that its first total derivative with respect to t has to be satisfied, too, i.e.,

$$0 = g_{,p}(p, t)\dot{p} + g_{,t}(p, t). \tag{4.7}$$

If the plane is smooth in the sense that sliding is possible, any displacement δp from the configuration p to the configuration $p + \delta p$ with $g(p + \delta p, t + \delta t) = 0$ is possible in an arbitrary way, i.e., for example just changing the orientation without changing the position is allowed. But if the plane is perfectly rough this is no longer possible. In this case just perfect rolling is possibly. Therefore, the displacements are mutually connected and have to satisfy a condition of the form

$$0 = \breve{h}(p, \dot{p}, t) \tag{4.8}$$

in addition to the constraint (4.7). While the constraint (4.6) restricts the positions and the constraints (4.7) restrict the displacements according to the possible positions, the constraints (4.8) additionally restrict the possible displacements, i.e., velocities, of the considered system without being based on any restrictions of the positions. This suggests that we have to distinguish carefully these two types of restrictions for the multibody system. □

Let us define the two types of restrictions mentioned in Example 4.1.2, the so-called *holonomic constraints* and the so-called *nonholonomic constraints*, e.g., see [17, 129, 147].

Definition 4.1.3 (Holonomic and nonholonomic constraints) *Constraints which only depend on the position variables p of the multibody system, i.e., which are of the form*

$$0 = g(p, t), \tag{4.9}$$

are called holonomic constraints *(on position level).*
Constraints which depend on the derivative $\dot{p}$ with respect to t of the position variables p, i.e., which are of the form

$$0 = \breve{h}(p, \dot{p}, t), \tag{4.10}$$

and for which no functions $g(p, t)$ with $\frac{\mathrm{d}}{\mathrm{d}t} g(p, t) = \breve{h}(p, \dot{p}, t)$ exist, are called nonholonomic constraints *(on velocity level).*

Remark 4.1.4 a) The constraints arising from (4.9) by differentiation with respect to t, i.e.,

$$0 = g_{,p}(p, t)\dot{p} + g_{,t}(p, t) \tag{4.11}$$

are called *holonomic constraints on velocity level* and the constraints arising from (4.9) by one more differentiation with respect to t, i.e.,

$$0 = g_{,pp}(p, t)[\![\dot{p}, \dot{p}]\!] + 2g_{,pt}(p, t)\dot{p} + g_{,p}(p, t)\ddot{p} + g_{,tt}(p, t)$$

are called *holonomic constraints on acceleration level*. Furthermore, the constraints arising from (4.10) by differentiation with respect to t, i.e.,

$$0 = \breve{h}_{,p}(p, \dot{p}, t)\dot{p} + \breve{h}_{,\dot{p}}(p, \dot{p}, t)\ddot{p} + \breve{h}_{,t}(p, \dot{p}, t)$$

are called *nonholonomic constraints on acceleration level*.
b) If the holonomic constraints (4.9) explicitly depend on t, then we have a so-called *kinematical excitation*. In case of the explicit dependency of the nonholonomic constraints (4.10) on t we have a so-called *motional excitation*. □

Remark 4.1.5 a) Holonomic constraints restrict the position as well as the motion of a multibody system with the same number of constraints. Nonholonomic

constraints restrict the velocity only, independent of associated restrictions to the position.
b) Note that if constraints do depend on the velocities they are not automatically nonholonomic.
c) In [24, 83, 147], it is discussed that nonholonomic constraints are linear in the velocity, i.e., they have the form

$$0 = H(p,t)\dot{p} + h(p,t). \tag{4.12}$$

d) Examples of multibody systems with nonholonomic constraints are the sliding of blades, knives, or skates (without scratching sidewards), or the rolling of balls (see Example 4.1.2) or cylinders. □

Definition 4.1.6 (Holonomic and nonholonomic multibody system) *If the motion of a multibody system is restricted by nonholonomic constraints, the multibody system is called* nonholonomic *or* nonholonomic system*. Otherwise it is called* holonomic *or* holonomic system*. If there do not exist any holonomic constraints but only nonholonomic constraints, the system is called* purely nonholonomic *or* purely nonholonomic system*.*

Since we have different types of constraints which restrict the coordinates p and its derivative $\dot{p}$ with respect to t in different ways, we have to distinguish between two types of degrees of freedom. In [147] the positional and motional degrees of freedom are introduced.

Definition 4.1.7 (Positional and motional degrees of freedom) *The* positional degrees of freedom *are defined as the degrees of freedom of choice of the position coordinates p and are denoted by n_{f_p}.*
The motional degrees of freedom *are defined as the degrees of freedom of the choice of the motion denoted by $\dot{p}$ and are denoted by n_{f_v}.*

Remark 4.1.8 If the positional and the motional degrees of freedom are identical, we will not distinguish them and call them *degrees of freedom* and we will denote them by n_f. □

Remark 4.1.9 In [129] the notation *geometrical degrees of freedom* and *kinematical degrees of freedom* is used instead of positional and motional degrees of freedom, respectively. The meaning is the same. □

Remark 4.1.10 The number of positional degrees of freedom is influenced by the number of nonredundant holonomic constraints which restrict the positions of the multibody system and therefore, the number of holonomic constraints reduces the positional degrees of freedom. Furthermore, the number of motional degrees of freedom is influenced by the number of nonredundant nonholonomic constraints in addition to the number of nonredundant holonomic constraints. Therefore, the number of motional degrees of freedom is less than or equal to the number of positional degrees of freedom

$$n_{f_v} \leq n_{f_p},$$

e.g., see Example 4.1.2. □

There exists a lot of literature, where holonomic as well as nonholonomic constraints are defined and discussed, e.g., [17, 24, 36, 52, 129, 147] but the number of references which additionally consider the modeling of nonholonomic systems, e.g., [126, 180], and furthermore which investigate the dynamical behavior of nonholonomic systems

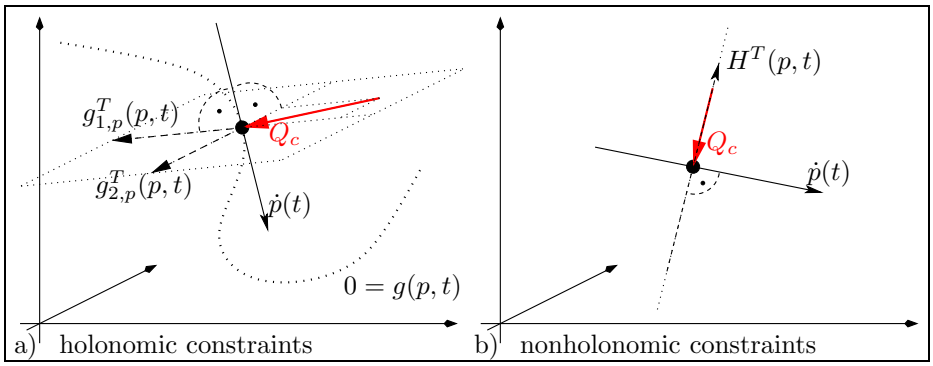

Figure 4.3: Holonomic and nonholonomic constraints and the respective constraint forces for the example of one mass point in three dimensional space with two holonomic constraints and one nonholonomic constraint

is quite small, e.g., [140]. In this work we will discuss the modeling of nonholonomic systems, see below, as well as their numerical integration, see Chapter 5.
For the investigation of the modeling process let us assume that the considered multibody system is restricted by constraints of the form

$$0 = K(p,t)\dot{p} + k(p,t), \tag{4.13}$$

regardless if they are of holonomic or of nonholonomic nature. In the case of a holonomic system (4.13) corresponds to (4.11). In the case of a purely nonholonomic system (4.13) corresponds to (4.12). Otherwise (4.13) corresponds to a system of equations consisting of equations of the form (4.11) and (4.12). We then have

$$K(p,t) = \begin{bmatrix} H(p,t) \\ g_{,p}(p,t) \end{bmatrix}, \; k(p,t) = \begin{bmatrix} h(p,t) \\ g_{,t}(p,t) \end{bmatrix}.$$

In general, the investigated constrained multibody system is equivalent with respect to its resulting motion to a free multibody system with additional forces, say $Q_c(p,t)$, depending on the position p and on t, which force the motion to satisfy the constraints and do not perform any work on the system. Since the work of the system performed by $Q_c(p,t)$ is defined as

$$W = \int Q_c(p,t) \; \mathrm{d}p(t),$$

the force $Q_c(p,t)$ does not perform any work on the system if and only if $Q_c(p,t)$ is perpendicular to all possible directions of motion, i.e., to $\dot{p}(t)$, according to the constraints (4.13) for all $t \in \mathbb{I}$, see Figure 4.3. Thus, we get

$$Q_c^T(p,t)\dot{p}(t) \stackrel{!}{=} 0. \tag{4.14}$$

Investigations of Lagrange [110] show that the force Q_c can be expressed as

$$Q_c = -K^T(p,t)\zeta, \tag{4.15}$$

with unknown functions $\zeta \in \mathcal{C}(\mathbb{I}, \mathbb{R}^{n_\zeta})$, which are called *Lagrange multipliers*. Adding the *constraint forces* $Q_c(p,t)$ to the external forces in the equations of motion (4.5) to force the motion to satisfy the restrictions (4.13), we get equation (4.4) in the form

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{p}}\right) - \frac{\partial L}{\partial p} = Q_a(p,t) + Q_c(p,t), \tag{4.16}$$

with $Q_a(p,t)+Q_c(p,t) = Q(p,t)$. The equations (4.16) are known as *Euler-Lagrange equations*, see [36, 180]. From the Euler-Lagrange equations, it follows that

$$M(p)\ddot{p} = \widetilde{f}(p,\dot{p},t) + Q_a(p,t) + Q_c(p,t).$$

The applied force $Q_a(p,t)$ is a known function and the constraint forces $Q_c(p,t)$ are defined by (4.15) with still unknown Lagrange multipliers ζ. But the consideration of the constraints delivers the information to determine the Lagrange multipliers, such that the equations

$$\begin{aligned} M(p)\ddot{p} &= f(p,\dot{p},t) - K^T(p,t)\zeta, && \text{(4.17a)} \\ 0 &= K(p,t)\dot{p} + k(p,t), && \text{(4.17b)} \end{aligned}$$

with $f(p,\dot{p},t) = \widetilde{f}(p,\dot{p},t) + Q_a(p,t)$ are sufficient to determine all unknowns $p(t)$ and $\zeta(t)$. Recall that the constraints (4.17a) contain holonomic constraints (4.9) as well as nonholonomic constraints (4.12). Explicit distinction of the holonomic and nonholonomic constraints in their original form yields the equations of motion of a constraint multibody system including holonomic constraints as well as nonholonomic constraints in the form

$$\begin{aligned} M(p)\ddot{p} &= f(p,\dot{p},t) - G^T(p,t)\lambda - H^T(p,t)\mu, && \text{(4.18a)} \\ 0 &= H(p,t)\dot{p} + h(p,t), && \text{(4.18b)} \\ 0 &= g(p,t), && \text{(4.18c)} \end{aligned}$$

with the *nonholonomic constraint matrix* $H(p,t)$ and the *holonomic constraint matrix* $G(p,t)$ defined by

$$G(p,t) = g_{,p}(p,t).$$

The columns of the matrices G^T and H^T describe the inaccessible directions of the motion, see Figure 4.3.
The equations of motion in the form (4.18) are known as *descriptor form for nonholonomic systems*. Note that, in general, the *descriptor form for holonomic systems* of the form

$$\begin{aligned} M(p)\ddot{p} &= f(p,\dot{p},t) - G^T(p,t)\lambda, && \text{(4.19a)} \\ 0 &= g(p,t), && \text{(4.19b)} \end{aligned}$$

is used as basis for investigations in the literature. The form (4.18) represents an extension of the generally used descriptor form (4.19). Nevertheless, in the present work we will focus on general nonholonomic systems such that mainly the descriptor form (4.18) for nonholonomic systems will be used.

Remark 4.1.11 In principle, it is possible (at least locally) to solve the constraints of the descriptor form (4.19) and to determine the (local) state space form (4.5). But the complete process of reduction to state space form is quite laborious, time consuming, and subject to the solution of nonlinear systems.
Even in the case when a determination of a set of minimal coordinates is possible, the convenient structure of the equations of motion in descriptor form, like sparsity, banded system matrices, or the constant mass matrix, is usually lost, such that the evaluation effort for the system matrices or the right-hand side of the equations of motion as well as the time consumption for algebraic transformations is increasing in an inconvenient way. On the other hand the descriptor form (4.18) bypasses topological analysis. Often the technical interpretation of the full set of nonminimal coordinates p with respect to the features and the dynamical behavior of the

mechanical system is simpler than for minimal coordinates.
In [164] the advantages and the disadvantages of several choices of the position variables, e.g., in minimal or nonminimal and in absolute or relative coordinates, is discussed in more detail. Already in [64], the authors dissuade from the usage of the state space form as basis of numerical computations because of the loss of sparsity and structure in comparison to the descriptor form. Furthermore, both the choice of the coordinates and the formulation in descriptor or state space form strongly influence the computational complexity of the equations of motion. For more details we refer to [52, 60, 147, 167]. □

4.1.3 Examples for mechanical systems

Example 4.1.12 The mathematical pendulum: Again, let us consider the mathematical pendulum, see Figure 4.1. In contrast to Example 4.1.1, now we choose absolute coordinates $p = \begin{bmatrix} x & y \end{bmatrix}^T$ denoting the position of the mass m in the two dimensional space $\mathbb{R}^2$ for the description of the configuration of the pendulum. The equations of motion of first order have the form

$$\begin{bmatrix} \dot{p}_1 \\ \dot{p}_2 \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}, \tag{4.20a}$$

$$\begin{bmatrix} m & 0 \\ 0 & m \end{bmatrix} \begin{bmatrix} \dot{v}_1 \\ \dot{v}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ -mg \end{bmatrix} - \begin{bmatrix} 2p_1 \\ 2p_2 \end{bmatrix} \begin{bmatrix} \lambda_1 \end{bmatrix}, \tag{4.20b}$$

$$0 = \begin{bmatrix} p_1^2 + p_2^2 - L^2 \end{bmatrix}. \tag{4.20c}$$

The constraints on velocity level and on acceleration level are given by

$$0 = \begin{bmatrix} 2p_1v_1 + 2p_2v_2 \end{bmatrix}, \tag{4.20d}$$

$$0 = \begin{bmatrix} 2v_1^2 + 2v_2^2 - 2p_2g - \frac{4}{m}(p_1^2 + p_2^2)\lambda_1 \end{bmatrix}, \tag{4.20e}$$

respectively, and

$$G(p,t) = \begin{bmatrix} 2p_1 & 2p_2 \end{bmatrix} \tag{4.21}$$

defines the holonomic constraint matrix. □

Example 4.1.13 The lolly: In this example let us introduce an extension of the mathematical pendulum. The multibody system consists of one point mass in the three dimensional space rotating about the origin with a fixed distance L and sliding on a surface given by $0 = g_2$, see Figure 4.4. The gravitational force is directed downwards along the z-axis and presses the point mass on the surface. The equations of motion of first order have the form

$$\begin{bmatrix} \dot{p}_1 \\ \dot{p}_2 \\ \dot{p}_3 \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}, \tag{4.22a}$$

$$\begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & m \end{bmatrix} \begin{bmatrix} \dot{v}_1 \\ \dot{v}_2 \\ \dot{v}_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} - \begin{bmatrix} 2p_1 & 4(p_1^2 + p_2^2 - 1)p_1 \\ 2p_2 & 4(p_1^2 + p_2^2 - 1)p_2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix}, \tag{4.22b}$$

$$0 = \begin{bmatrix} p_1^2 + p_2^2 - 1 \\ e^{-\beta(p_1^2+p_2^2-1)^2} - 1 - p_3 \end{bmatrix}. \tag{4.22c}$$

Note that we have introduced the parameter β in the equations of motion which influences the shape of the surface. With increasing parameter β, the summit becomes sharper. □

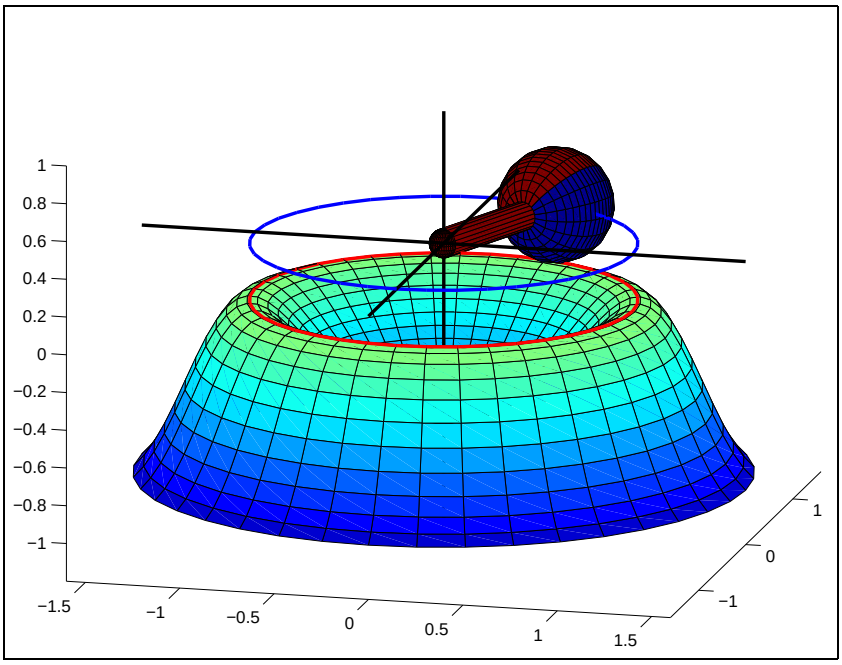

Figure 4.4: Topology of the lolly

Example 4.1.14 The nonlinear truck model: In [167] a planar nonlinear model of a truck is introduced as benchmark example. In Figure 4.5 (thanks to Bernd Simeon for providing this figure) the topology as well as the coordinates, bodies, joints, and force elements are depicted. The model consists of eleven coordinates p_i, $i = 1, ..., 11$ describing the motion of seven rigid bodies and one Lagrange multiplier λ_1, see Table 4.1.

Body		Coordinate	
1	rear wheel	p_1	vertical motion
2	front wheel	p_2	vertical motion
3	truck chassis	p_3	vertical motion
		p_4	rotation about y-axis
4	engine	p_5	vertical motion
		p_6	rotation about y-axis
5	driver cabin	p_7	vertical motion
		p_8	rotation about y-axis
6	driver seat	p_9	vertical motion
7	loading area	p_{10}	vertical motion
		p_{11}	rotation about y-axis
		λ_1	Lagrange multiplier with respect to the joint between loading area and truck chassis

Table 4.1: Nonlinear truck model

We omit to specify the equations of motion in detail and refer to [167] instead. Note that the equations of motion of the truck model are badly scaled, since the solution of the Lagrange multiplier λ_1 is of magnitude 10^4 but the solution of the other independent variables p and v are of magnitude 10^{-2}. □

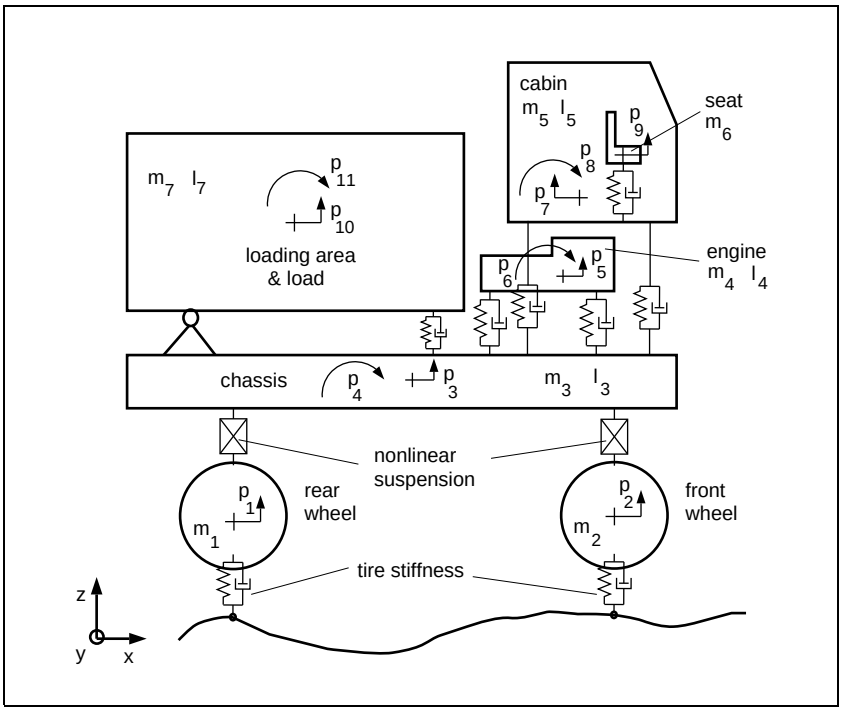

Figure 4.5: Topology of the truck

Example 4.1.15 Slider crank: In this example we will model a slider crank as depicted in Figure 4.6.

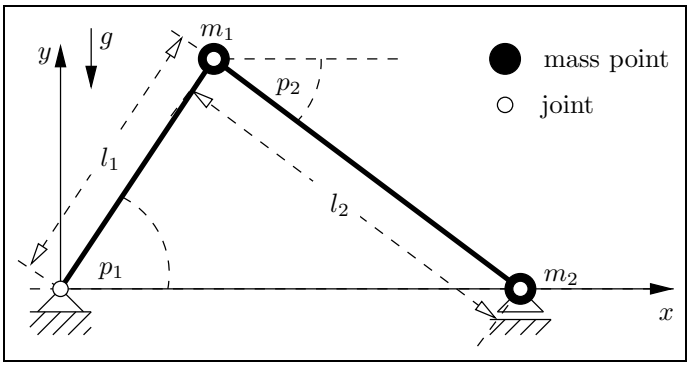

Figure 4.6: Topology of the slider crank

We will discuss two different types of equations of motion. First, the equations of motion with nonredundant constraints

$$\begin{bmatrix} \dot{p}_1 \\ \dot{p}_2 \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}, \tag{4.23a}$$

$$M(p) \begin{bmatrix} \dot{v}_1 \\ \dot{v}_2 \end{bmatrix} = \begin{bmatrix} -(m_1+m_2)gl_1\cos(p_1) - m_2 l_1 l_2 v_2^2 \sin(p_1+p_2) \\ m_2 g l_2 \cos(p_2) - m_2 l_1 l_2 v_1^2 \sin(p_1+p_2) \end{bmatrix} - \begin{bmatrix} l_1\cos(p_1) \\ -l_2\cos(p_2) \end{bmatrix} \begin{bmatrix} \lambda_1 \end{bmatrix}, \tag{4.23b}$$

$$0 = \begin{bmatrix} l_1\sin(p_1) - l_2\sin(p_2) \end{bmatrix}, \tag{4.23c}$$

where

$$M(p) = \begin{bmatrix} (m_1+m_2)l_1^2 & -m_2 l_1 l_2 \cos(p_1+p_2) \\ -m_2 l_1 l_2 \cos(p_1+p_2) & m_2 l_2^2 \end{bmatrix}, \tag{4.23d}$$

and secondly, the equations of motion with redundant constraints

$$\begin{bmatrix} \dot{p}_1 \\ \dot{p}_2 \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}, \tag{4.24a}$$

$$M(p) \begin{bmatrix} \dot{v}_1 \\ \dot{v}_2 \end{bmatrix} = \begin{bmatrix} -(m_1+m_2) g l_1 \cos(p_1) - m_2 l_1 l_2 v_2^2 \sin(p_1+p_2) \\ m_2 g l_2 \cos(p_2) - m_2 l_1 l_2 v_1^2 \sin(p_1+p_2) \end{bmatrix} - G^T(p,t) \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{bmatrix}, \tag{4.24b}$$

$$0 = \begin{bmatrix} l_1 \sin(p_1) - l_2 \sin(p_2) \\ r_{1x} + r_{2x} - r_{3x} \\ r_{1y} + r_{2y} - r_{3y} \end{bmatrix}, \tag{4.24c}$$

where

$$\begin{aligned} r_{1x} &= l_1 \cos(p_1), & r_{1y} &= l_1 \sin(p_1), \\ r_{2x} &= l_1 \cos(p_1) + l_2 \cos(p_2), & r_{2y} &= l_1 \sin(p_1) + l_2 \sin(p_2), \\ r_{3x} &= 2l_1 \cos(p_1) + l_2 \cos(p_2), & r_{3y} &= 2l_1 \sin(p_1) + l_2 \sin(p_2), \end{aligned}$$

and with mass matrix (4.23d) and constraint matrix

$$G(p,t) = \begin{bmatrix} l_1 \cos(p_1) & -l_2 \cos(p_2) \\ -(-2l_1 \sin(p_1) + 2l_1 \sin(p_1)) & -(-l_2 \sin(p_2) + l_2 \sin(p_2)) \\ -(2l_1 \cos(p_1) - 2l_1 \cos(p_1)) & -(-l_2 \sin(p_2) + l_2 \sin(p_2)) \end{bmatrix}. \tag{4.24d}$$

The generation of the equations of motion (4.24) with redundant constraints may be obtained via use of tools for the automatic generation of the model equations, see [112]. □

Example 4.1.16 Double four joint mechanism: Let us consider the mechanism depicted in Figure 4.7. It consists of four rigid bodies which are linked together by use of rotational joints. The lengths of the bodies are denoted by L_i, $i = 1, ..., 4$ with $L_1 = L_2 = L_3$, the masses by m_i, $i = 1, ..., 4$, and the gravitational acceleration by g. The position variables are given by the absolute orientation $p_i = \varphi_i$, $i = 1, ..., 4$. The masses of bodies $1, 2, 3$ are placed at the end of the rods and the mass of the fourth body is placed in its center. The position of the mass m_4 is given by (x_4, y_4) with respect to the reference frame.
Let us follow the approach of Lagrange and Euler to model this mechanical system. The kinetic energy is given by

$$\begin{aligned} T(\varphi, \dot{\varphi}) &= \frac{1}{2} J_1 \dot{\varphi}_1^2 + \frac{1}{2} J_2 \dot{\varphi}_2^2 + \frac{1}{2} J_3 \dot{\varphi}_3^2 + \frac{1}{2} m_4 (\dot{x}_4^2 + \dot{y}_4^2) \\ &= \frac{1}{2}(J_1 + m_4 L_1^2)\dot{\varphi}_1^2 + \frac{1}{2} J_2 \dot{\varphi}_2^2 + \frac{1}{2} J_3 \dot{\varphi}_3^2 + \frac{1}{8} L_4^2 m_4 \dot{\varphi}_4^2 \\ &\quad + \frac{1}{2} m_4 L_1 L_4 \cos(\varphi_1 - \varphi_4) \dot{\varphi}_1 \dot{\varphi}_4 \end{aligned}$$

and the potential energy is given by

$$\begin{aligned} U(\varphi) &= g(m_1 L_1 \sin(\varphi_1) + m_2 L_2 \sin(\varphi_2) + m_3 L_3 \sin(\varphi_3) \\ &\quad + m_4 (L_1 \sin(\varphi_1) + \frac{L_4}{2} \sin(\varphi_4))). \end{aligned}$$

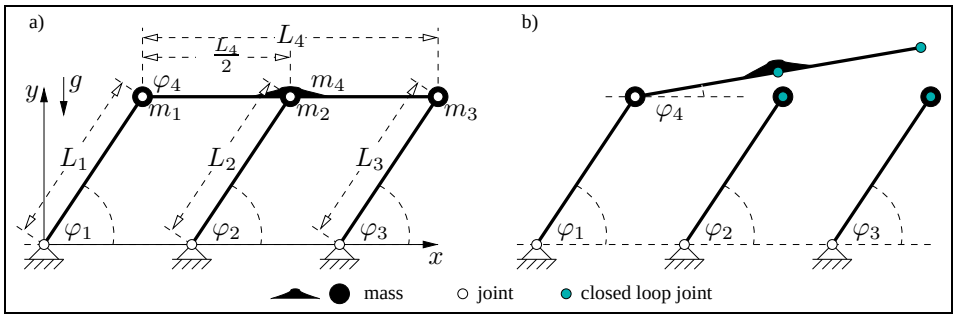

Figure 4.7: Topology of the double four joint mechanism

Furthermore, the motion of the mechanism is restricted such that the closed loop joints, see Figure 4.7b, fit together. The constraints are given by

$$\begin{aligned}
0 = \quad g_1(\varphi) &= L_1\cos(\varphi_1) + \frac{L_4}{2}\cos(\varphi_4) - L_2\cos(\varphi_2) - \frac{L_4}{2},\\
0 = \quad g_2(\varphi) &= L_1\sin(\varphi_1) + \frac{L_4}{2}\sin(\varphi_4) - L_2\sin(\varphi_2),\\
0 = \quad g_3(\varphi) &= L_1\cos(\varphi_1) + L_4\cos(\varphi_4) - L_3\cos(\varphi_3) - L_4,\\
0 = \quad g_4(\varphi) &= L_1\sin(\varphi_1) + L_4\sin(\varphi_4) - L_3\sin(\varphi_3).
\end{aligned}$$

With the Lagrange function (4.2) and the constraint forces $Q_c = -G^T\lambda$ with

$$G = \begin{bmatrix} -L_1\sin(\varphi_1) & L_2\sin(\varphi_2) & 0 & -\frac{L_4}{2}\sin(\varphi_4) \\ L_1\cos(\varphi_1) & -L_2\cos(\varphi_2) & 0 & \frac{L_4}{2}\cos(\varphi_4) \\ -L_1\sin(\varphi_1) & 0 & L_3\sin(\varphi_3) & -L_4\sin(\varphi_4) \\ L_1\cos(\varphi_1) & 0 & -L_3\cos(\varphi_3) & L_4\cos(\varphi_4) \end{bmatrix} \tag{4.25}$$

from the Euler-Lagrange equation (4.16) in form

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{\varphi}}\right) - \frac{\partial L}{\partial \varphi} = -G^T\lambda$$

we obtain with

$$\begin{aligned}
0 \quad &= \quad \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{p}}\right) - \frac{\partial L}{\partial p} \\
&= \begin{bmatrix} (J_1 + m_4L_1^2) & 0 & 0 & +\frac{1}{2}m_4L_1L_4\cos(\varphi_1-\varphi_4) \\ 0 & J_2\ddot{\varphi}_2 & 0 & 0 \\ 0 & 0 & J_3\ddot{\varphi}_3 & 0 \\ \frac{1}{2}m_4L_1L_4\cos(\varphi_1-\varphi_4) & 0 & 0 & \frac{1}{4}L_4^2m_4 \end{bmatrix} \begin{bmatrix} \ddot{\varphi}_1 \\ \ddot{\varphi}_2 \\ \ddot{\varphi}_3 \\ \ddot{\varphi}_4 \end{bmatrix} \\
&\quad + \begin{bmatrix} \frac{1}{2}m_4L_1L_4\sin(\varphi_1-\varphi_4)\dot{\varphi}_4^2 \\ 0 \\ 0 \\ -\frac{1}{2}m_4L_1L_4\sin(\varphi_1-\varphi_4)\dot{\varphi}_1^2 \end{bmatrix} + \begin{bmatrix} g(m_1+m_4)L_1\cos(\varphi_1) \\ gm_2L_2\cos(\varphi_2) \\ gm_3L_3\cos(\varphi_3) \\ gm_4\frac{L_4}{2}\cos(\varphi_4) \end{bmatrix}
\end{aligned}$$

the equations of motion together with the constraints in the form

$$\begin{bmatrix} (J_1+m_4L_1^2) & 0 & 0 & +\frac{1}{2}m_4L_1L_4\cos(\varphi_1-\varphi_4) \\ 0 & J_2\ddot{\varphi}_2 & 0 & 0 \\ 0 & 0 & J_3\ddot{\varphi}_3 & 0 \\ \frac{1}{2}m_4L_1L_4\cos(\varphi_1-\varphi_4) & 0 & 0 & \frac{1}{4}L_4^2m_4 \end{bmatrix} \begin{bmatrix} \ddot{\varphi}_1 \\ \ddot{\varphi}_2 \\ \ddot{\varphi}_3 \\ \ddot{\varphi}_4 \end{bmatrix}$$

$$= \begin{bmatrix} -g(m_1+m_4)L_1\cos(\varphi_1) - \frac{1}{2}m_4L_1L_4\sin(\varphi_1-\varphi_4)\dot{\varphi}_4^2 \\ -gm_2L_2\cos(\varphi_2) \\ -gm_3L_3\cos(\varphi_3) \\ -gm_4\frac{L_4}{2}\cos(\varphi_4) + \frac{1}{2}m_4L_1L_4\sin(\varphi_1-\varphi_4)\dot{\varphi}_1^2 \end{bmatrix} \tag{4.26a}$$

$$- \begin{bmatrix} -L_1\sin(\varphi_1) & L_1\cos(\varphi_1) & -L_1\sin(\varphi_1) & L_1\cos(\varphi_1) \\ L_2\sin(\varphi_2) & -L_2\cos(\varphi_2) & 0 & 0 \\ 0 & 0 & L_3\sin(\varphi_3) & -L_3\cos(\varphi_3) \\ -\frac{L_4}{2}\sin(\varphi_4) & \frac{L_4}{2}\cos(\varphi_4) & -L_4\sin(\varphi_4) & L_4\cos(\varphi_4) \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \end{bmatrix},$$

$$0 = \begin{bmatrix} L_1\cos(\varphi_1) + \frac{L_4}{2}\cos(\varphi_4) - L_2\cos(\varphi_2) \\ L_1\sin(\varphi_1) + \frac{L_4}{2}\sin(\varphi_4) - L_2\sin(\varphi_2) - \frac{L_4}{2} \\ L_1\cos(\varphi_1) + L_4\cos(\varphi_4) - L_3\cos(\varphi_3) \\ L_1\sin(\varphi_1) + L_4\sin(\varphi_4) - L_3\sin(\varphi_3) - L_4 \end{bmatrix}. \tag{4.26b}$$

At first glance one might assume that this mechanism is a forced mechanism. But it is clear, that this mechanism reacts like a pendulum with one degree of freedom. Therefore, the constraints must be redundant. A check of the rank of the constraint matrix G (4.25) yields

$$\operatorname{rank}(G) = \begin{cases} 3 & \text{if } p \text{ satisfies (4.26b)}, \\ 4 & \text{else.} \end{cases}$$

Since the rank of the constraint matrix G is three along every solution, this four joint mechanism has one degree of freedom. □

Example 4.1.17 The skateboard: In this example we will consider a skateboarder who is rolling on a flat horizontal surface. The simplified topology is illustrated in Figure 4.8.

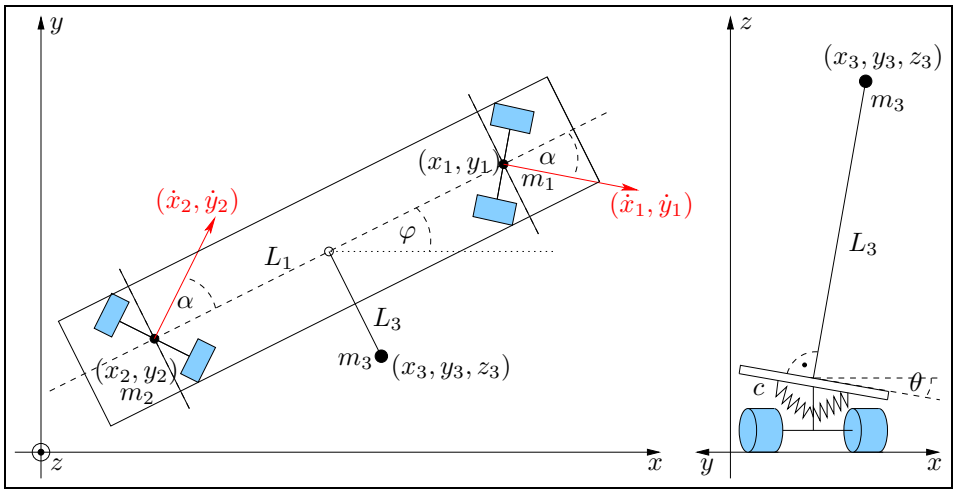

Figure 4.8: Topology of skateboard

Let us model the board such that the whole mass of the board is split into the masses m_1 and m_2 located in the center of both axes whose positions are defined by (x_1, y_1)

and (x_2, y_2), respectively. Furthermore, the orientation of the board is defined by the angle φ which gives the direction of the motion and by the angle θ which defines the bank of the board. It is assumed that the wheels of the skateboard do not lose the contact to the surface, i.e., to the x-y-plane. The skateboard is constructed such that both axes are pivoted such that the axes are changing their relative direction denoted by the angle α with respect to the center axis of the board if the board banks. This yields a self stabilizing effect which is strongly influenced by the relation between α and θ, which is assumed to be

$$\alpha = a\theta$$

with the banking coefficient a.
Let us assume that the skateboarder is very unexperienced such that he is modeled just as the mass m_3 with position (x_3, y_3, z_3) which is located at a distance of l_3 perpendicular to the board as shown in the right of Figure 4.8.
If the position variables $p = \begin{bmatrix} x_1 & y_1 & x_2 & y_2 & \varphi & x_3 & y_3 & z_3 & \theta \end{bmatrix}^T$ could move freely in three dimensional space, the equations of motions would be given by

$$M\ddot{p} = f(p)$$

with

$$M = \begin{bmatrix} m_1 & & & & & & & & \\ & m_1 & & & & & & & \\ & & m_2 & & & & & & \\ & & & m_2 & & & & & \\ & & & & J_1 & & & & \\ & & & & & m_3 & & & \\ & & & & & & m_3 & & \\ & & & & & & & m_3 & \\ & & & & & & & & J_2 \end{bmatrix}, \quad f(p, \dot{p}) = \begin{bmatrix} -d_1\dot{x}_1 \\ -d_1\dot{y}_1 \\ -d_2\dot{x}_2 \\ -d_2\dot{y}_2 \\ -d_\varphi\dot{\varphi} \\ -d_3\dot{x}_3 \\ -d_3\dot{y}_3 \\ -m_3 g - d_3\dot{z}_3 \\ -c\theta - d_\theta\dot{\theta} \end{bmatrix},$$

where the stiffness of the spring, the damping, and the gravitational acceleration are given by c, d_1, d_2, d_3, d_φ, d_θ, and g, respectively. Furthermore, J_1 and J_2 denote the inertia of the board with respect to the rotations φ and θ, respectively. But obviously the positions are restricted in its choice by holonomic constraints $0 = g(p)$ with

$$\begin{aligned} g_1 &= x_2 + L_1\cos(\varphi) - x_1, \\ g_2 &= y_2 + L_1\sin(\varphi) - y_1, \\ g_3 &= (x_1 + x_2)/2 + L_3\cos(\varphi - \pi/2)\sin(\theta) - x_3, \\ g_4 &= (y_1 + y_2)/2 + L_3\sin(\varphi - \pi/2)\sin(\theta) - y_3, \\ g_5 &= L_3\cos(\theta) - z_3. \end{aligned}$$

From the holonomic constraints we get the holonomic constraint matrix

$$G(p) = \begin{bmatrix} -1 & 0 & 1 & 0 & -L_1\sin(\varphi) & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & L_1\cos(\varphi) & 0 & 0 & 0 & 0 \\ 1/2 & 0 & 1/2 & 0 & -L_3\sin(\varphi_\perp)\sin(\theta) & -1 & 0 & 0 & L_3\cos(\varphi_\perp)\cos(\theta) \\ 0 & 1/2 & 0 & 1/2 & L_3\cos(\varphi_\perp)\sin(\theta) & 0 & -1 & 0 & L_3\sin(\varphi_\perp)\cos(\theta) \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -L_3\sin(\theta) \end{bmatrix}$$

with $\varphi_\perp = \varphi - \pi/2$.
Furthermore, let us assume that the skateboard is not allowed to slide on the surface.

In particular, this means that the the motion of the wheels is perpendicular to its axes. This leads to the nonholonomic constraints $0 = \breve{h}(p, \dot{p})$ with

$$\begin{aligned}
\breve{h}_1 &= \begin{bmatrix} x_1 - x_2 & y_1 - y_2 \end{bmatrix} \begin{bmatrix} \cos(\pi/2 - \alpha) & \sin(\pi/2 - \alpha) \\ -\sin(\pi/2 - \alpha) & \cos(\pi/2 - \alpha) \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{y}_1 \end{bmatrix}, \\
\breve{h}_2 &= \begin{bmatrix} x_1 - x_2 & y_1 - y_2 \end{bmatrix} \begin{bmatrix} \cos(\pi/2 + \alpha) & \sin(\pi/2 + \alpha) \\ -\sin(\pi/2 + \alpha) & \cos(\pi/2 + \alpha) \end{bmatrix} \begin{bmatrix} \dot{x}_2 \\ \dot{y}_2 \end{bmatrix}.
\end{aligned}$$

They have the form $\breve{h}(p, \dot{p}) = H(p)\dot{p}$ with the nonholonomic constraint matrix $H(p)$ with

$$H^T(p) = \begin{bmatrix}
\Delta x \cos(\pi/2 - \alpha) - \Delta y \sin(\pi/2 - \alpha) & 0 \\
\Delta x \sin(\pi/2 - \alpha) + \Delta y \cos(\pi/2 - \alpha) & 0 \\
0 & \Delta x \cos(\pi/2 + \alpha) - \Delta y \sin(\pi/2 + \alpha) \\
0 & \Delta x \sin(\pi/2 + \alpha) + \Delta y \cos(\pi/2 + \alpha) \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0
\end{bmatrix}$$

with $\Delta x = (x_1 - x_2)$ and $\Delta y = (y_1 - y_2)$. Hence, the equations of motion for the skateboard have the form

$$\begin{aligned}
\dot{p} &= v, && \text{(4.27a)} \\
M\dot{v} &= f(p, v) - G^T(p)\lambda - H^T(p)\mu, && \text{(4.27b)} \\
0 &= g(p), && \text{(4.27c)} \\
0 &= H(p)v && \text{(4.27d)}
\end{aligned}$$

with the holonomic constraint force $G^T\lambda$ and the nonholonomic constraint force $H^T\mu$ and their associated Lagrange multipliers λ and μ, respectively. □

4.1.4 Solution invariants

Many motions of mechanical systems have known solution invariants, i.e., relations which are satisfied along any motion of the mechanical system, like the invariance of the total energy, momentum, or impulse. Let us denote the m_e equations describing such solution invariants by

$$0 = e(p, v, s, u). \tag{4.28}$$

In particular, conservative multibody systems are energy conserving. In this case the total energy is constant along every motion of the system. Let us consider this fact. The equations of motion for constraint conservative systems are given by the Euler-Lagrange equations (4.16) without applied forces, i.e., $Q_a(p, t) = 0$. The constraint forces are given by (4.15) and we get

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{p}}\right) - \frac{\partial L}{\partial p} = -K^T(p, t)\zeta \tag{4.29}$$

which describes the solution behavior. From the Lagrange function (4.2) we get (4.29) in the form

$$\frac{\mathrm{d}}{\mathrm{d}t}T_{,\dot{p}} - T_{,p} + U_{,p} = -K^T\zeta,$$

which is satisfied by every solution of (4.29). Therefore, every solution satisfies

$$\dot{p}^T \frac{\mathrm{d}}{\mathrm{d}t} T_{,\dot{p}} - \dot{p}^T T_{,p} + \dot{p}^T U_{,p} = -\dot{p}^T K^T \zeta.$$

From (4.14) it follows that $\dot{p}^T K^T \zeta = 0$, and we get

$$\dot{p}^T \frac{\mathrm{d}}{\mathrm{d}t} T_{,\dot{p}} - \dot{p}^T T_{,p} + \dot{p}^T U_{,p} = 0$$

which is equivalent to

$$\frac{\mathrm{d}}{\mathrm{d}t}(\dot{p}^T T_{,\dot{p}}) - \dot{T} + \dot{U} = 0. \tag{4.30}$$

Furthermore, we have that the kinetic energy is a quadratic form in $\dot{p}$ (see page 111), i.e.,

$$T(p, \dot{p}) = \dot{p}^T \Theta(p) \dot{p}.$$

We obtain that

$$\dot{p}^T T_{,\dot{p}} = \dot{p}^T \frac{\partial}{\partial \dot{p}} (\dot{p}^T \Theta(p) \dot{p}) = \dot{p}^T (2\Theta(p)\dot{p}) = 2T$$

and from (4.30) we get the conservation of the total energy in the form

$$\dot{T} + \dot{U} = 0.$$

In the numerical integration of the equations of motion it is often desirable to conserve these solution invariants in an explicit way, because in general the numerical solution does not satisfy the solution invariants of the equations of motion. In [65] a method for maintaining solution invariants in the numerical solution of ODEs is considered. The idea is to introduce a regularization term into the ODE such that the solution manifold defined by the solution invariants becomes attracting. Furthermore, the preservation of solution invariants of so-called Hamiltonian systems is considered in [80, 151] in detail.
Let us discuss the conservation of the total energy in the example of the mathematical pendulum.

Example 4.1.18 The mathematical pendulum: The equations of motion of the mathematical pendulum in descriptor form, see Figure 4.1, are developed in Example 4.1.12.
Since the pendulum is only influenced by the gravitational field of forces, i.e., by a conservative field of forces, and since it is not affected by other applied forces, it represents a mechanical system which conserves the total energy. This total energy is given by

$$E(p, v) = \frac{1}{2} m (v_1^2 + v_2^2) + m g p_2 \tag{4.31}$$

and is conserved such that

$$0 = E(p(t), v(t)) - E(p_0, v_0) = e(p, v) \tag{4.32}$$

for $t \in \mathbb{I}$ and every solution of the equations of motion (4.20).
In Figure 4.9 the total energy in the numerical solution is illustrated. The numerical solution is computed with `RADAU5` [79, 82] for different formulations, see Section 4.4, `ODASSL` [59, 60], `MEXAX` [118], and `HEDOP5` [6]. Obviously, the numerical solutions

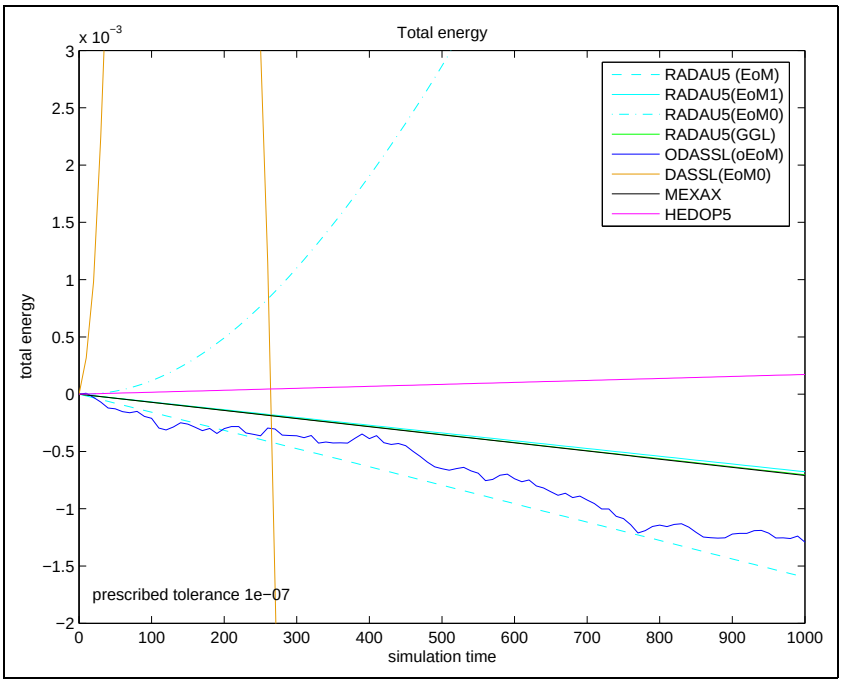

Figure 4.9: Conservation of the total energy by numerical solutions

are far from being constant.
Let us consider the holonomic constraints (4.20c) and their derivatives, which restrict the motion of the pendulum in a nonredundant way, in comparison to the conservation of the total energy (4.32). We have

$$\begin{aligned} 0 &= p_1^2 + p_2^2 - L^2, && \text{(4.33a)} \\ 0 &= 2p_1v_1 + 2p_2v_2, && \text{(4.33b)} \\ 0 &= 2v_1^2 + 2v_2^2 - 2p_2g - \frac{4}{m}(p_1^2 + p_2^2)\lambda_1, && \text{(4.33c)} \\ 0 &= \frac{1}{2}m(v_1^2 + v_2^2) + mgp_2 - E_0. && \text{(4.33d)} \end{aligned}$$

The constraints (4.33) are nonredundant for all p, v, and λ satisfying (4.33). In particular, in addition to the holonomic constraints and their derivatives the energy conservation restricts the solution as well. The dimension of the solution manifold with the energy conservation is therefore smaller than without the energy conservation. As shown in Figure 4.9, the restriction according to the energy conservation is in general not satisfied by the numerical solution of the equations of motion. Therefore, it is necessary to insert these restrictions arising from energy conservation in the equations of motion. This yields a stabilization of the solution such that all constraints, i.e., the holonomic constraints and their derivatives as well as the energy conservation are satisfied even in the numerical method.
For further numerical results see Example 5.3.1 in Section 5.3. □

4.1.5 Classification of equations of motion

In [164] a classification of several forms of the equations of motion using modeling levels is given. In this section we will shortly recall the modeling levels 0, 1, and 2, and we will give an extension of this classification by two additional modeling

levels.

Modeling level 0 - Standard formulation: Based on the standard formulation, given by the Lagrange equations of type one in descriptor form for holonomic systems (4.19) and by use of the *velocity variables* $v(t)$ of dimension n_p the *equations of motion of modeling level 0* are given by

$$\begin{aligned} \dot{p} &= v, && \text{(4.34a)} \\ M(p)\dot{v} &= f(p,v,t) - G^T(p)\lambda, && \text{(4.34b)} \\ 0 &= g(p). && \text{(4.34c)} \end{aligned}$$

In connection with initial values

$$p(t_0) = p_0,\ v(t_0) = v_0,\ \lambda(t_0) = \lambda_0 \qquad \text{(4.34d)}$$

we have the initial value problem for the equations of motion of modeling level 0 on the domain $\mathbb{I} = [t_0, t_f]$. The n_p equations (4.34a) are called *kinematical equations of motion*. They accomplish the order reduction of the equations of motion (4.5) or (4.19) from order two to order one.
Furthermore, the equations of motion are affected by the n_λ holonomic constraints (4.34c). From the constraints $g(p) = 0$ one obtains the constraint matrix $G(p) = \frac{\partial g}{\partial p}(p)$ which column-wise contains the inaccessible directions of motion, see Figure 4.3.
The n_v equations (4.34b) are called *dynamical equations of motion*. They follow from the equilibrium of forces and momenta and include the mass matrix $M(p)$, the vector $f(p,v,t)$ of the applied and gyroscopic forces, the constraint matrix $G(p)$ of the holonomic constraints, the associated constraint forces $G^T(p)\lambda$, and the Lagrange multipliers λ. The mass matrix $M(p)$ is positive semi-definite, since the kinetic energy is a positive semi-definite quadratic form, and it includes the inertia properties of the multibody system.
If the mass matrix $M(p)$ is nonsingular for all possible p, the equations of motion of modeling level 0 (4.34) are in Hessenberg form (3.25) of order 3. See also Example 3.5.56.
The solution of the equations of motion (4.34) satisfies in addition to the holonomic constraints (4.34c) its first and its second derivatives with respect to t, i.e.,

$$\begin{aligned} 0 = G(p)v &= g^I(p,v), && \text{(4.35)} \\ 0 = G_{,p}(p)[\![v]\!]v + G(p)M^{-1}(p)(f(p,v,t) - G^T(p)\lambda) &= g^{II}(p,v,\lambda), && \text{(4.36)} \end{aligned}$$

such that every solution of (4.34) lies in the solution manifold

$$\mathbb{M} = \{(p,v,\lambda,t) \in \mathbb{R}^{2n_p+n_\lambda} \times \mathbb{I} \ :\ 0 = g,\ 0 = g^I,\ 0 = g^{II}\}. \qquad \text{(4.37)}$$

The equations of motion (4.20) modeling the mathematical pendulum, see Example 4.1.12, belong to modeling level 0.
Often used forms for the equations of motion of modeling level 0 are the *s-index-1 formulation of modeling level 0*

$$\begin{aligned} \dot{p} &= v, && \text{(4.38a)} \\ M(p)\dot{v} &= f(p,v,t) - G^T(p)\lambda, && \text{(4.38b)} \\ 0 &= g^I(p,v), && \text{(4.38c)} \end{aligned}$$

using the first derivative (4.35) with respect to t instead of the holonomic constraints (4.34c) and the *s-index-0 formulation of modeling level 0*

$$\begin{aligned} \dot{p} &= v, && \text{(4.39a)} \\ M(p)\dot{v} &= f(p,v,t) - G^T(p)\lambda, && \text{(4.39b)} \\ 0 &= g^{II}(p,v,\lambda), && \text{(4.39c)} \end{aligned}$$

using the second derivative (4.36) with respect to t instead of the holonomic constraints (4.34c).
If the mass matrix $M(p,u)$ is nonsingular for all possible p and u, also the equations of motion of modeling level 1 (4.40) are in Hessenberg form (3.25) of order 3. See also Example 3.5.56.

Modeling level 1 - Dynamical force elements, friction, and spatial motion: If the investigated multibody system is influenced by friction effects, the friction is modeled as part of the applied forces Q_a such that f additionally depends on the Lagrange multipliers λ. Furthermore, if the multibody system contains some controls or dynamical force elements, like multibody systems with additional control devices, hydraulic or electromagnetic components, *dynamical force variables* r of size n_r are introduced, which specify the state of such dynamical force elements as in equation (4.40c), see [52].
In the case of spatial multibody systems, which are discussed in [52], an additional feature has to be taken into account. If the equations of motion of second order, like (4.5), (4.18), or (4.19), are reduced to first order, one has to take the relation between the *generalized velocity* $\dot{p}$ and the velocities v into account, see [86]. In order to transform this second order system to an equivalent first order system we introduce a velocity vector v and get the equations (4.40a) with a transformation matrix $Z(p)$ of size $n_p \times n_v$ with $n_p \leq n_v$, that determines the (angular) velocities. The transformation matrix $Z(p)$ is not the identity I_{n_p} if there are rotations in three dimensional space and it can be determined by *Poisson's*[2] *kinematical equations* [1, 52]. In the two dimensional case we have $Z(p) = I_{n_p}$, i.e., $\dot{p} = v$. Note that the transformation matrix $Z(p)$ mainly depends on the choice of the velocity vector. The use of *Eulerian angles*, *Cardano*[3] *angles*, *Tait*[4]*-Bryan angles*, see [1, 86, 147, 180], are common choices of the velocity vector. They yield $n_p = n_v$ and a square transformation matrix $Z(p)$. Unfortunately, in these cases the transformation matrix $Z(p)$ holds singularities, i.e., there exist some configurations given by p such that the transformation matrix becomes singular. If it cannot be insured in advance that the mechanical system does not pass through such configurations a remedy is the use of *quaternions*, often also called *Euler parameters*, which lead to a rectangular transformation matrix $Z(p)$ with $n_p < n_v$. For more details on quaternions see [140, 147].
Summing up, the *equations of motion of modeling level 1* have the form

$$\begin{aligned} \dot{p} &= Z(p)v, && \text{(4.40a)} \\ M(p,u)\dot{v} &= f(p,v,r,\lambda,u) - Z^T(p)G^T(p,u)\lambda, && \text{(4.40b)} \\ \dot{r} &= b(p,v,r,\lambda,u), && \text{(4.40c)} \\ 0 &= g(p,u). && \text{(4.40d)} \end{aligned}$$

Together with initial values

$$p(t_0) = p_0,\ v(t_0) = v_0,\ r(t_0) = r_0,\ \lambda(t_0) = \lambda_0 \qquad \text{(4.40e)}$$

we have the initial value problem for the equations of motion of modeling level 1 on the domain $\mathbb{I} = [t_0, t_f]$. Note that here all functions and system matrices depend on an additional variable u of size n_u, which represents *control variables* that are given as a function of t. In particular, if the investigated system is a nonautonomous

[2]Siméon Denis Poisson (born 1781 in Pithivier, France - died 1840 in Paris, France)
[3]Girolamo Cardano (born 1501 in Pavia, Duchy of Milan (now Italy) - died 1576 in Rome (now Italy))
[4]Peter Guthrie Tait (born 1831 in Dalkeith, Midlothian, Scotland - died 1901 in Edinburgh, Scotland)

system, then there exists an explicit dependency on t. If this is the case, t will be modeled as a component of the control u, for example $u_1(t) = t$.
Note in addition, that in contrast to [164] for reasons of symmetry, the dynamical equations of motion are premultiplied by the transformation matrix $Z(p)$, implicitly contained in M and f.

Modeling level 2 - Contact, force laws and constraints: The equations

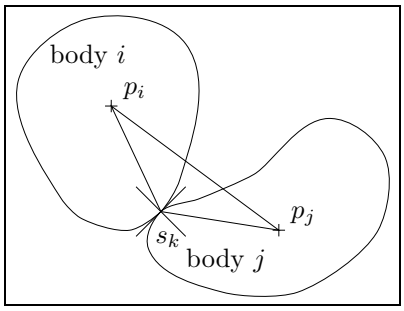

Figure 4.10: Contact point between two bodies

of motion up to modeling level 1 do not yet allow the consideration of additional constraints, e.g., arising from contact problems. Therefore, the equations of motion (4.40) have to be extended by *contact variables* s of size n_s which have to be uniquely determined by the *contact conditions* $0 = c(p, s, u)$ of dimension n_s that describe the relationship between these contact variables s, the position variables p, and the control variables u, see Figure 4.10. Note that because of the assumed uniqueness of s, the partial derivative $c_{,s}(p, s, u)$ is assumed to be nonsingular for all possibly solutions, see the Implicit Function Theorem 2.3.1. Therefore, we have the contact variables as function of the positions and the control variables, i.e., $s = s(p, u)$.
Sometimes, force laws and constraints may be formulated more conveniently using *auxiliary variables* w of size n_w that are implicitly defined by the possibly nonlinear equation (4.41d). Note again that because of the assumed uniqueness of w by (4.41d), the partial derivative $d_{,w}$ is assumed to be nonsingular for all possible solutions, see the Implicit Function Theorem 2.3.1.
The *equations of motion of modeling level 2* have the form

$$\begin{aligned}
\dot{p} &= Z(p)v, && \text{(4.41a)}\\
M(p,u)\dot{v} &= f(p,v,r,w,s,\lambda,u) - Z^T(p)G^T(p,s,u)\lambda, && \text{(4.41b)}\\
\dot{r} &= b(p,v,r,w,s,\lambda,u), && \text{(4.41c)}\\
0 &= d(p,v,r,w,s,\lambda,u), && \text{(4.41d)}\\
0 &= c(p,s,u), && \text{(4.41e)}\\
0 &= g(p,s,u). && \text{(4.41f)}
\end{aligned}$$

Together with initial values

$$p(t_0) = p_0,\ v(t_0) = v_0,\ r(t_0) = r_0,\ w(t_0) = w_0,\ s(t_0) = s_0,\ \lambda(t_0) = \lambda_0 \quad \text{(4.41g)}$$

we have the initial value problem for the equations of motion of modeling level 2 on the domain $\mathbb{I} = [t_0, t_f]$. This formulation of modeling level 2 is proposed in [4]. Note that in contrast to [164] no additional acceleration variables $a = \dot{v}$ are introduced here. The introduction of additional acceleration variables a would yield the equations of motion in the form of a semi-explicit DAE (3.24).
In contrast to the equations of motion of modeling level 0 or 1 the equations of

motion of modeling level 2 (4.41) are no longer in Hessenberg form (3.25) because of the occurrence of the auxiliary variables w and the contact variables s.
In the previous considerations concerning constraints we have found that the constraint forces, given by Q_c in (4.15), do not perform any work on the mechanical system. This is satisfied if the constraint forces are perpendicular to the manifold given by the holonomic constraints (4.41f) with respect to the contact equations (4.41e). Because of the dependency of the contact variables on the positions and the control variables, we have holonomic constraints in the form $0 = g(p, s(p,u), u)$. Therefore, we get a constraint matrix which contains columnwise the inaccessible directions that are perpendicular to the manifold given by the holonomic constraints (4.41f) with respect to the contact equations (4.41e).

Lemma 4.1.19 *Let $c_{,s}(p,s,u)$ be nonsingular and have a bounded inverse $c_{,s}^{-1}(p,s,u)$ for all $(p,s,u) \in \mathbb{R}^{n_p} \times \mathbb{R}^{n_s} \times \mathbb{R}^{n_u}$ satisfying (4.41e) and (4.41f). Then the constraint matrix G in (4.41b) is given by*

$$G(p,u) = \left[\frac{\partial g}{\partial p} - \frac{\partial g}{\partial s} \left(\frac{\partial c}{\partial s} \right)^{-1} \frac{\partial c}{\partial p} \right] (p,u). \tag{4.42}$$

Proof: From (4.41e) it follows from the Implicit Function Theorem 2.3.1 that there exists a function $\xi(p,u)$ such that $0 = c(p, \xi(p,u), u)$. Therefore, we have $s = s(p,u) = \xi(p,u)$, and for the total derivative of $g(p, s(p,u), u)$ with respect to p, it follows that

$$\begin{aligned} G(p,u) &= \frac{\mathrm{d}g}{\mathrm{d}p}(p, s(p,u), u) \\ &= \left[\frac{\partial g}{\partial p} + \frac{\partial g}{\partial s} \frac{\partial s}{\partial p} \right] (p, s(p,u), u) \\ &= \left[\frac{\partial g}{\partial p} - \frac{\partial g}{\partial s} \left(\frac{\partial c}{\partial s} \right)^{-1} \frac{\partial c}{\partial p} \right] (p,u). \end{aligned}$$

The Jacobian $\frac{\partial s}{\partial p}$ has been obtained from (4.41e) by implicit differentiation with respect to p, see the Implicit Function Theorem 2.3.1. □

Modeling level 3 - Nonholonomic constraints: As discussed in Section 4.1.2 there exist some mechanical systems where the motion is not only restricted by holonomic constraints but in addition by nonholonomic constraints, see Example 4.1.2 or Remark 4.1.5d. Therefore, we will introduce the *equations of motion of modeling level 3* containing nonholonomic constraints in addition to the features of the equations of motion of modeling level 2 (4.41). Because of Remark 4.1.5c we introduce nonholonomic constraints in quasi-linear form as in (4.12). We get the equations of motion of modeling level 3 in the form

$$\begin{aligned} \dot{p} &= Z(p)v, && \text{(4.43a)} \\ M(p,u)\dot{v} &= f(p,v,r,w,s,\lambda,\mu,u) && \text{(4.43b)} \\ &\quad -Z^T(p)G^T(p,s,u)\lambda - Z^T(p)H^T(p,s,u)\mu, \\ \dot{r} &= b(p,v,r,w,s,\lambda,\mu,u), && \text{(4.43c)} \\ 0 &= d(p,v,r,w,s,\lambda,\mu,u), && \text{(4.43d)} \\ 0 &= c(p,s,u), && \text{(4.43e)} \\ 0 &= H(p,s,u)Z(p)v + h(p,s,u), && \text{(4.43f)} \\ 0 &= g(p,s,u). && \text{(4.43g)} \end{aligned}$$

Together with initial values

$$\begin{gathered} p(t_0) = p_0,\ v(t_0) = v_0,\ r(t_0) = r_0,\ w(t_0) = w_0, \\ s(t_0) = s_0,\ \lambda(t_0) = \lambda_0,\ \mu(t_0) = \mu_0 \end{gathered} \tag{4.43h}$$

we have the initial value problem for the equations of motion of modeling level 3 on the domain $\mathbb{I} = [t_0, t_f]$. Here, the explicit constraints are called *holonomic constraints on position level* (4.43g) with respect to the contact point equation (4.43e) and the *nonholonomic constraints on velocity level* (4.43f). If the linear dependency of v is not important, then we denote the nonholonomic constraints on velocity level by

$$\breve{h}(p, v, s, u) = H(p, s, u)Z(p)v + h(p, s, u). \tag{4.44}$$

Modeling level 4 - Redundant constraints: Sometimes the multibody system under investigation contains a topological structure which is statically overdetermined. For instance, the Example 4.1.16 offers no unique solution with respect to the Lagrange multipliers, i.e., the joint forces are not uniquely determined. In such cases the constraints are redundant, see Definition 2.3.10. Furthermore, by the modeling of mechanical systems using certain connections via certain types of joints or by use of certain modeling tools redundant constraints may be generated, see [86, 112, 147, 181].
Therefore, the holonomic and nonholonomic constraints of the equations of motion of modeling level 3 are allowed to be redundant. This yields the *equations of motion of modeling level 4* which have the same structure as (4.43) but the holonomic and nonholonomic constraints (4.43g) and (4.43f), respectively, are allowed to form redundant sets of equations. The equations of motion (4.24) modeling the slider crank, see Example 4.1.15, or the equations of motion (4.26) modeling the double four joint mechanism with redundant constraints, see Example 4.1.16, belong to modeling level 4 because of their redundant constraints.

With $n = n_p + n_v + n_r + n_w + n_s + n_\lambda + n_\mu$ and the *state variables* x defined by $x = \begin{bmatrix} p^T & v^T & r^T & w^T & s^T & \lambda^T & \mu^T \end{bmatrix}^T$ of size n, the equations of motions of modeling level 4 (4.43) correspond to a quasi-linear DAE (3.23) with

$$E(x, u) = \begin{bmatrix} I_{n_p} & & & & & & \\ & M(p,u) & & & & & \\ & & I_{n_r} & & & & \\ & & & 0 & & & \\ & & & & 0 & & \\ & & & & & 0 & \\ & & & & & & 0 \end{bmatrix}$$

of size $n \times n$ and

$$k(x, u) = \begin{bmatrix} Z(p)v \\ f(p, v, r, w, s, \lambda, \mu, u) - Z^T(p)G^T(p, s, u)\lambda - Z^T(p)H^T(p, s, u)\mu \\ b(p, v, r, w, s, \lambda, \mu, u) \\ d(p, v, r, w, s, \lambda, \mu, u) \\ c(p, s, u) \\ H(p, s, u)Z(p)v + h(p, s, u) \\ g(p, s, u) \end{bmatrix}$$

of size n. In the present work we will concentrate on the equations of motion of modeling level 3 and 4.

Remark 4.1.20 In the case of nonredundant or uniformly redundant constraints let us define

$$r_G = \operatorname{rank}(G(p,s,u)) \quad \text{and} \quad r_H = \operatorname{rank}(H(p,s,u)) \tag{4.45}$$

for all $(p,s,u) \in \mathbb{M}$ (see Lemma 4.2.14 below). In the case of nonredundant constraints we have $r_G = n_\lambda$ and $r_H = n_\mu$. In particular, we have $r_G \leq n_p$ and $r_G + r_H \leq n_v$ independent on the redundancies of the constraints. In the special case that $r_G = n_p$ the motion of the mechanical system is completely determined by the constraints. Such mechanical systems are called *forced mechanical systems*. □

Remark 4.1.21 Because of high speed motions of mechanical systems, effects arising from deformation of structural elements sometimes cannot be neglected. Then, the consideration of (partially) elastic structural components of mechanical systems is necessary. In [165] mechanical systems with elastic bodies are investigated and it is pointed out that after the space discretization of the elastic effects the equations of motion for elastic multibody systems have the same structure as those of rigid multibody systems. For this reason they can be treated in an analogous way as discussed below. One should be aware though that the dimension of the systems then is typically very large. □

4.2 Analysis of the equations of motion

In this section we will analyze the equations of motion of modeling level 4 (4.43) with possibly redundant constraints (4.43g) and (4.43f). In particular, we will determine the solution manifold, consistency conditions, the minimal reduced derivative array, and we will consider the existence and uniqueness of a solution of the equations of motion.
Before we start to investigate the regularity of the equations of motion let us anticipate some assumptions. In the investigations below it will become clear why these assumptions are justified and necessary. In the following let us use the abbreviations

$$G_\lambda = Z^T G^T - f_{,\lambda} + f_{,w} d_{,w}^{-1} d_{,\lambda}, \tag{4.46}$$

$$H_\mu = Z^T H^T - f_{,\mu} + f_{,w} d_{,w}^{-1} d_{,\mu}. \tag{4.47}$$

Assumption 4.2.1 *Consider the equations of motion of modeling level 4 (4.43). Then the matrices*

$$\text{a)} \quad d_{,w}, \tag{4.48a}$$

$$\text{b)} \quad c_{,s}, \tag{4.48b}$$

are assumed to be nonsingular and are assumed to have a bounded inverse for all $(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^2) \in \mathbb{M}$ (see Lemma 4.2.14 below) and furthermore, we assume that

$$\text{c)} \quad \operatorname{rank}\begin{pmatrix} M & G_\lambda & H_\mu \\ GZ & 0 & 0 \\ HZ & 0 & 0 \end{pmatrix} = n_v + r_G + r_H \tag{4.48c}$$

for all $(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^2) \in \mathbb{M}$ with r_G and r_H defined in (4.45). Furthermore, it is assumed that

$$d \in \mathcal{C}^1(\mathbb{M}, \mathbb{R}^{n_w}),\ c \in \mathcal{C}^1(\mathbb{M}, \mathbb{R}^{n_s}),\ \breve{h} \in \mathcal{C}^2(\mathbb{M}, \mathbb{R}^{n_\mu}),\ g \in \mathcal{C}^3(\mathbb{M}, \mathbb{R}^{n_\lambda}).$$

With respect to $\mathfrak{u}^2$ see Notation 3.1.4.

Remark 4.2.2 Note that in the case of nonredundant constraints, i.e., in the case of the equations of motion of modeling level 3 (4.43), the Assumption (4.48c) corresponds to the assumption that

$$\text{c)} \quad \begin{bmatrix} M & G_\lambda & H_\mu \\ GZ & 0 & 0 \\ HZ & 0 & 0 \end{bmatrix} \text{ is nonsingular} \tag{4.49}$$

and has a bounded inverse for all $(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^2) \in \mathbb{M}$ (see Lemma 4.2.14 below). In particular, in the case of the equations of motion of modeling level 0 (4.34) this corresponds to the nonsingularity of the matrix

$$\begin{bmatrix} M & G^T \\ G & 0 \end{bmatrix},$$

which is assumed in many references, e.g., [6, 23, 118, 164]. □

Remark 4.2.3 a) In the case of $n_s = 0$ or $n_w = 0$ it follows that $\partial c/\partial s \in \mathbb{R}^{0,0}$ and $\partial d/\partial w \in \mathbb{R}^{0,0}$, respectively. Since the empty matrix represents the identity mapping from $\mathbb{R}^0$ into $\mathbb{R}^0$ we interpret it as nonsingular.
b) Assumption (4.49) guarantees that the constraints are not redundant. □

In the following we will restrict our considerations to nonsingular mass matrices M. If in addition to Assumption 4.2.1 we assume the nonsingularity of the mass matrix M for all $(p, u) \in \mathbb{M}$ (see Lemma 4.2.14 below) we obtain from the contact equations (4.43e), and the auxiliary equations (4.43d), via the Implicit Function Theorem 2.3.1, see (2.14), that

$$\begin{aligned} \dot{s} &= -c_{,s}^{-1} c_{,p} \dot{p} - c_{,s}^{-1} c_{,u} \dot{u} \\ &= -c_{,s}^{-1} c_{,p} Z v - c_{,s}^{-1} c_{,u} \dot{u} \end{aligned}$$

and

$$\begin{aligned} \dot{w} &= -d_{,w}^{-1} (d_{,p} \dot{p} + d_{,v} \dot{v} + d_{,r} \dot{r} + d_{,s} \dot{s} + d_{,\lambda} \dot{\lambda} + d_{,\mu} \dot{\mu} + d_{,u} \dot{u}) \\ &= -d_{,w}^{-1} \big(d_{,p} Z v + d_{,v} M^{-1} (f - Z^T G^T \lambda - Z^T H^T \mu) + d_{,r} b \\ &\qquad - d_{,s} c_{,s}^{-1} (c_{,p} Z v + c_{,u} \dot{u}) + d_{,\lambda} \dot{\lambda} + d_{,\mu} \dot{\mu} + d_{,u} \dot{u} \big). \end{aligned}$$

Using the equations of motion of modeling level 4 (4.43), the first and second derivatives with respect to t of the holonomic constraints (4.43g) are given by

$$\begin{aligned} 0 &= g^I(p, v, s, \mathfrak{u}^1) && \text{(4.50a)} \\ &= G \dot{p} + (g_{,u} - g_{,s} c_{,s}^{-1} c_{,u}) \dot{u} && \text{(4.50b)} \\ &= GZv + (g_{,u} - g_{,s} c_{,s}^{-1} c_{,u}) \dot{u} && \text{(4.50c)} \end{aligned}$$

and

$$\begin{aligned} 0 &= g^{I\!I}(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^2) && \text{(4.51a)} \\ &= g^I_{,p} \dot{p} + g^I_{,v} \dot{v} + g^I_{,s} \dot{s} + g^I_{,\mathfrak{u}^1} \dot{\mathfrak{u}}^1 && \text{(4.51b)} \\ &= (g^I_{,p} - g^I_{,s} c_{,s}^{-1} c_{,p}) Z v + GZ \dot{v} + g^I_{,\mathfrak{u}^1} \dot{\mathfrak{u}}^1 - g^I_{,s} c_{,s}^{-1} c_{,u} \dot{u} && \text{(4.51c)} \\ &= (g^I_{,p} - g^I_{,s} c_{,s}^{-1} c_{,p}) Z v + GZ M^{-1} (f - Z^T G^T \lambda - Z^T H^T \mu) && \text{(4.51d)} \\ &\qquad + g^I_{,\mathfrak{u}^1} \dot{\mathfrak{u}}^1 - g^I_{,s} c_{,s}^{-1} c_{,u} \dot{u}. \end{aligned}$$

The first and second derivatives with respect to t of the holonomic constraints on position level are called *holonomic constraints on velocity level* (4.50a) with respect to the contact point equation (4.43e) and *holonomic constraints on acceleration level* (4.51a), respectively. Differentiating the holonomic constraints (4.43g) three times yields

$$
\begin{aligned}
0 &= g^{\text{III}}(p,v,r,w,s,\lambda,\mu,\dot\lambda,\dot\mu,\mathfrak{u}^3) && (4.52a)\\
&= \frac{\mathrm{d}^3}{\mathrm{d}t^3} g(p,s,u) && (4.52b)\\
&= g^{\text{II}}_{,p}\dot p + g^{\text{II}}_{,v}\dot v + g^{\text{II}}_{,r}\dot r + g^{\text{II}}_{,w}\dot w + g^{\text{II}}_{,s}\dot s + g^{\text{II}}_{,\lambda}\dot\lambda + g^{\text{II}}_{,\mu}\dot\mu + g^{\text{II}}_{,\mathfrak{u}^2}\dot{\mathfrak{u}}^2 && (4.52c)\\
&= \big(g^{\text{II}}_{,p} Z v + g^{\text{II}}_{,v} M^{-1}(f - Z^T G^T\lambda - Z^T H^T \mu) + g^{\text{II}}_{,r} b && (4.52d)\\
&\quad - g^{\text{II}}_{,w} d_{,w}^{-1}(d_{,p} Z v + d_{,v} M^{-1}(f - Z^T G^T\lambda - Z^T H^T\mu) + d_{,r} b\\
&\qquad + d_{,s} c_{,s}^{-1}(c_{,p} Z v - c_{,u}\dot u) + d_{,u}\dot u)\\
&\quad - g^{\text{II}}_{,s} c_{,s}^{-1}(c_{,p} Z v + c_{,u}\dot u) + g^{\text{II}}_{,\mathfrak{u}^2}\dot{\mathfrak{u}}^2\big)\\
&\quad - GZM^{-1}(Z^T G^T - f_{,\lambda} + f_{,w} d_{,w}^{-1} d_{,\lambda})\dot\lambda\\
&\quad - GZM^{-1}(Z^T H^T - f_{,\mu} + f_{,w} d_{,w}^{-1} d_{,\mu})\dot\mu\\
&= \widetilde{g}^{\text{III}}(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^3) - GZM^{-1} G_\lambda\dot\lambda - GZM^{-1} H_\mu\dot\mu, && (4.52e)
\end{aligned}
$$

with G_λ and H_μ given in (4.46). The first derivative with respect to t of the nonholonomic constraints (4.43f) is given by

$$
\begin{aligned}
0 &= h^I(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^1) && (4.53a)\\
&= \frac{\mathrm{d}}{\mathrm{d}t}(H(p,s,u)Z(p)v + h(p,s,u)) && (4.53b)\\
&= \breve h_{,p}\dot p + HZ\dot v + \breve h_{,s}\dot s + \breve h_{,u}\dot u && (4.53c)\\
&= (\breve h_{,p} - \breve h_{,s} c_{,s}^{-1} c_{,p}) Z v + HZ\dot v + (\breve h_{,u} - \breve h_{,s} c_{,s}^{-1} c_{,u})\dot u && (4.53d)\\
&= (\breve h_{,p} - \breve h_{,s} c_{,s}^{-1} c_{,p}) Z v + HZM^{-1}(f - Z^T G^T\lambda - Z^T H^T\mu) && (4.53e)\\
&\quad + (\breve h_{,u} - \breve h_{,s} c_{,s}^{-1} c_{,u})\dot u,
\end{aligned}
$$

which are called *nonholonomic constraints on acceleration level* (4.53a) with respect to the contact point equation (4.43e). Furthermore, differentiating the nonholonomic constraints (4.43f) twice yields

$$
\begin{aligned}
0 &= h^{\text{II}}(p,v,r,w,s,\lambda,\mu,u,\dot\lambda,\dot\mu,\mathfrak{u}^2) && (4.54a)\\
&= \frac{\mathrm{d}^2}{\mathrm{d}t^2}(H(p,s,u)Z(p)v + h(p,s,u)) && (4.54b)\\
&= h^I_{,p}\dot p + h^I_{,v}\dot v + h^I_{,r}\dot r + h^I_{,w}\dot w + h^I_{,s}\dot s + h^I_{,\lambda}\dot\lambda + h^I_{,\mu}\dot\mu + h^I_{,\mathfrak{u}^1}\dot{\mathfrak{u}}^1 && (4.54c)\\
&= \big(h^I_{,p} Z v + h^I_{,v} M^{-1}(f - Z^T G^T\lambda - Z^T H^T\mu) + h^I_{,r} b && (4.54d)\\
&\quad - h^I_{,w} d_{,w}^{-1}(d_{,p} Z v + d_{,v} M^{-1}(f - Z^T G^T\lambda - Z^T H^T\mu) + d_{,r} b\\
&\qquad + d_{,s} c_{,s}^{-1}(c_{,p} Z v - c_{,u}\dot u) + d_{,u}\dot u)\\
&\quad - h^I_{,s} c_{,s}^{-1}(c_{,p} Z v + c_{,u}\dot u) + h^I_{,\mathfrak{u}^1}\dot{\mathfrak{u}}^1\big)\\
&\quad - HZM^{-1}(Z^T G^T - f_{,\lambda} + f_{,w} d_{,w}^{-1} d_{,\lambda})\dot\lambda\\
&\quad - HZM^{-1}(Z^T H^T - f_{,\mu} + f_{,w} d_{,w}^{-1} d_{,\mu})\dot\mu\\
&= \widetilde{h}^{\text{II}}(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^2) - HZM^{-1} G_\lambda\dot\lambda - HZM^{-1} H_\mu\dot\mu, && (4.54e)
\end{aligned}
$$

with G_λ and H_μ given in (4.46). The holonomic constraints on velocity level and on acceleration level in form (4.50c) and (4.51d), respectively, as well as the nonholonomic constraints on acceleration level in form (4.53e) will turn out to bee the

hidden constraints of the equations of motion as we will see in Procedure 4.2.21, below.
With a nonsingular mass matrix M we impose a more restrictive assumption as follows.

Assumption 4.2.4 *Consider the equations of motion of modeling level 4 (4.43). Then the matrices*

$$\text{a)} \quad d_{,w}, \tag{4.55a}$$
$$\text{b)} \quad c_{,s}, \tag{4.55b}$$
$$\text{c)} \quad M \tag{4.55c}$$

are assumed to be nonsingular and are assumed to have a bounded inverse for all $(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^2) \in \mathbb{M}$ *(see Lemma 4.2.14 below) and furthermore, we assume that*

$$\text{d)} \quad \operatorname{rank}\left(\begin{bmatrix} GZM^{-1}G_\lambda & GZM^{-1}H_\mu \\ HZM^{-1}G_\lambda & HZM^{-1}H_\mu \end{bmatrix}\right) = r_G + r_H \tag{4.55d}$$

for all $(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^2) \in \mathbb{M}$. *Furthermore, it is assumed that*

$$d \in \mathcal{C}^1(\mathbb{M}, \mathbb{R}^{n_w}),\ c \in \mathcal{C}^1(\mathbb{M}, \mathbb{R}^{n_s}),\ \breve{h} \in \mathcal{C}^2(\mathbb{M}, \mathbb{R}^{n_\mu}),\ g \in \mathcal{C}^3(\mathbb{M}, \mathbb{R}^{n_\lambda}).$$

The difference between Assumption 4.2.1 and Assumption 4.2.4 lies only in the fact that the mass matrix M is assumed to be nonsingular in the latter case.

Remark 4.2.5 a) The condition (4.55d) implies that the constraints are either nonredundant or uniformly redundant on $\mathbb{M}$.
b) Note that in the case of nonredundant constraints, i.e., in the case of the equations of motion of modeling level 3 (4.43), Assumption (4.55d) corresponds to the assumption that the matrix

$$\text{d)} \quad \begin{bmatrix} GZM^{-1}G_\lambda & GZM^{-1}H_\mu \\ HZM^{-1}G_\lambda & HZM^{-1}H_\mu \end{bmatrix} \text{ is nonsingular} \tag{4.56}$$

and has a bounded inverse for all $(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^2) \in \mathbb{M}$ (see Lemma 4.2.14 below).
In particular, in the case of the equations of motion of modeling level 0 (4.34) this corresponds to the nonsingularity of the matrix $GM^{-1}G^T$. If M is positive definite, then the nonsingularity of $GM^{-1}G^T$ is equivalent to the full rank of the constraint matrix, i.e., that $G(p)$ has full row rank, see (4.34). This corresponds to the so-called *Grübler condition* [59].
c) Note that in (4.56) we have

$$\begin{bmatrix} GZM^{-1}G_\lambda & GZM^{-1}H_\mu \\ HZM^{-1}G_\lambda & HZM^{-1}H_\mu \end{bmatrix} = \begin{bmatrix} g^{\text{II}}_{,\lambda} - g^{\text{II}}_{,w}d^{-1}_{,w}d_{,\lambda} & g^{\text{II}}_{,\mu} - g^{\text{II}}_{,w}d^{-1}_{,w}d_{,\mu} \\ h^{I}_{,\lambda} - h^{I}_{,w}d^{-1}_{,w}d_{,\lambda} & h^{I}_{,\mu} - h^{I}_{,w}d^{-1}_{,w}d_{,\mu} \end{bmatrix}.$$

□

Remark 4.2.6 a) The Assumptions (4.55c), (4.55d) correspond to Assumption (4.48c) in the case of a nonsingular mass matrix.
b) Note that if w does not occur, then Assumption (4.55d) reduces to

$$\operatorname{rank}\left(\begin{bmatrix} h^{I}_{,\mu} & h^{I}_{,\lambda} \\ g^{\text{II}}_{,\mu} & g^{\text{II}}_{,\lambda} \end{bmatrix}\right) = r_G + r_H$$

for all $(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^2) \in \mathbb{M}$. □

Definition 4.2.7 (Quasi-regular equations of motion) *Equations of motion of modeling level 4 (4.43) satisfying Assumption 4.2.1 with noncontradictory constraints (4.43f) and (4.43g) are called* quasi-regular equations of motion.

Definition 4.2.8 (Regular equations of motion) *Equations of motion of modeling level 4 (4.43) satisfying Assumptions (4.48a),(4.48b) and (4.49) are called* regular equations of motion.

Remark 4.2.9 Analogously to the definitions above, if the mass matrix is nonsingular for all possible p and u then the equations of motion of modeling level 4 (4.43) are quasi-regular if they satisfy Assumption 4.2.4, and they are regular if they satisfy Assumptions (4.55a)-(4.55c) and (4.56) □

Here and in the following, we will often suppress the dependency on p, v, r, w, s, λ, μ, and u in the notation unless we want to focus on some of those dependencies.

Lemma 4.2.10 *Let the equations of motion of modeling level 3 (4.43) satisfy Assumptions (4.55a)-(4.55c) and (4.56). Then the matrices*

$$G,\ H,\ \begin{bmatrix} G \\ H \end{bmatrix},\ \begin{bmatrix} GZ \\ HZ \end{bmatrix},\ \begin{bmatrix} G_\lambda & H_\mu \end{bmatrix}$$

have full rank for all $(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^2) \in \mathbb{M}$.

Proof: The matrix in (4.56) can be written as

$$\begin{bmatrix} G \\ H \end{bmatrix} ZM^{-1} \begin{bmatrix} G_\lambda & H_\mu \end{bmatrix} = \begin{bmatrix} GZ \\ HZ \end{bmatrix} M^{-1} \begin{bmatrix} G_\lambda & H_\mu \end{bmatrix}.$$

From the nonsingularity of the matrix in (4.56) for all $(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^2) \in \mathbb{M}$, it follows that the matrices

$$\begin{bmatrix} G \\ H \end{bmatrix},\ \begin{bmatrix} GZ \\ HZ \end{bmatrix},\ \begin{bmatrix} G_\lambda & H_\mu \end{bmatrix}$$

have full rank. Furthermore, we have that $\begin{bmatrix} G^T & H^T \end{bmatrix}^T$ is of size $n_\lambda + n_\mu \times n_v$ with $n_\lambda + n_\mu \leq n_v$. Therefore, it follows in addition that G and H have full (row) rank. □

Lemma 4.2.11 *Let the equations of motion of modeling level 4 (4.43) be quasi-regular, i.e., they satisfy Assumptions 4.2.4 with noncontradictory constraints (4.43f) and (4.43g). Then*

$$\operatorname{rank}(G) = r_G,\ \operatorname{rank}(H) = r_H,\ \operatorname{rank}(\begin{bmatrix} G \\ H \end{bmatrix}) = r_G + r_H,$$
$$\operatorname{rank}(\begin{bmatrix} GZ \\ HZ \end{bmatrix}) = r_G + r_H,\ \operatorname{rank}(\begin{bmatrix} G_\lambda & H_\mu \end{bmatrix}) = r_G + r_H$$

for all $(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^2) \in \mathbb{M}$.

Proof: The proof is analogous to the proof of Lemma 4.2.10. □

Lemma 4.2.12 *Let the equations of motion of modeling level 4 (4.43) satisfy Assumptions (4.55a)-(4.55c) with noncontradictory constraints (4.43f) and (4.43g). Then the state* $(p(t), v(t), r(t), w(t), s(t), \lambda(t), \mu(t)) \in \mathbb{M}$ *is a regular point if and only if the matrices*

$$G, \begin{bmatrix} HZM^{-1}G_\lambda & HZM^{-1}H_\mu \\ GZM^{-1}G_\lambda & GZM^{-1}H_\mu \end{bmatrix}, \begin{bmatrix} GZ \\ HZ \end{bmatrix} \tag{4.57}$$

have full rank for all $(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^2) \in \mathbb{M}$.

Proof: The solution manifold is given by (4.65). The partial derivative of its constraints with respect to w, λ, μ, v, s, and p, after some block row changes is given by

$$\begin{bmatrix} d_{,w} & d_{,\lambda} & d_{,\mu} & d_{,v} & d_{,s} & d_{,p} \\ GZM^{-1}f_{,w} & GZM^{-1}(f_{,\lambda}{-}Z^TG^T) & GZM^{-1}(f_{,\mu}{-}Z^TH^T) & g_v^{I\!I} & g_{,s}^{I\!I} & g_{,p}^{I\!I} \\ HZM^{-1}f_{,w} & HZM^{-1}(f_{,\lambda}{-}Z^TG^T) & HZM^{-1}(f_{,\mu}{-}Z^TH^T) & h_{,v}^I & h_{,s}^I & h_{,p}^I \\ 0 & 0 & 0 & HZ & \breve{h}_{,s} & \breve{h}_{,p} \\ 0 & 0 & 0 & GZ & g_{,s}^I & g_{,p}^I \\ 0 & 0 & 0 & 0 & c_{,s} & c_{,p} \\ 0 & 0 & 0 & 0 & g_{,s} & g_{,p} \end{bmatrix}. \tag{4.58}$$

Note that the partial derivatives with respect to r are not relevant, because r is not restricted by $\mathbb{M}$. Block Gauß elimination leads to

$$\begin{bmatrix} d_{,w} & d_{,\lambda} & d_{,\mu} & d_{,v} & d_{,s} & d_{,p} \\ 0 & -GZM^{-1}G_\lambda & -GZM^{-1}H_\mu & g_v^{I\!I} - B_G d_{,v} & g_{,s}^{I\!I} - B_G d_{,s} & g_{,p}^{I\!I} - B_G d_{,p} \\ 0 & -HZM^{-1}G_\lambda & -HZM^{-1}H_\mu & h_{,v}^I - B_H d_{,v} & h_{,s}^I - B_H d_{,s} & h_{,p}^I - B_H d_{,p} \\ 0 & 0 & 0 & HZ & \breve{h}_{,s} & \breve{h}_{,p} \\ 0 & 0 & 0 & GZ & g_{,s}^I & g_{,p}^I \\ 0 & 0 & 0 & 0 & c_{,s} & c_{,p} \\ 0 & 0 & 0 & 0 & 0 & G \end{bmatrix}$$

with $B_G = GZM^{-1}f_{,w}d_{,w}^{-1}$ and $B_H = HZM^{-1}f_{,w}d_{,w}^{-1}$. From Assumptions (4.55a) and (4.55b) we have the nonsingularity of $d_{,w}$ and $c_{,s}$ and we get the full rank of the matrix (4.58) if and only if the matrices in (4.57) have full rank for all $(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^2) \in \mathbb{M}$. By Definition 2.3.3 the assertion follows. □

Remark 4.2.13 For the equations of motion of modeling level 3 (4.43), i.e., in particular, with nonredundant constraints, we obtain a justification for Assumptions (4.55a)-(4.55c) and (4.56) from Lemmata 4.2.10 and 4.2.12. Furthermore, for the equations of motion of modeling level 4 (4.43), i.e., in particular, with possibly redundant constraints, constant rank of the matrices in Assumption 4.2.4 are of great importance for the numerical treatment. This gives a justification for Assumption 4.2.4 from Lemmata 4.2.11 and 4.2.12. □

If the equations of motion of modeling level 4 (4.43) satisfy Assumption 4.2.4, then we have that

$$\frac{\partial}{\partial \begin{bmatrix} s^T & p^T \end{bmatrix}^T} \begin{bmatrix} c \\ g \end{bmatrix} = \begin{bmatrix} c_{,s} & c_{,p} \\ g_{,s} & g_{,p} \end{bmatrix} \tag{4.59}$$

has rank $n_s + r_G$, which follows by block Gauß elimination and Lemma 4.1.19. Furthermore, the auxiliary variables w are uniquely determined by equation (4.43d) via the Implicit Function Theorem 2.3.1. Therefore, the state variables x (in particular, the position variables p, the auxiliary variables w, and the contact variables s) are restricted to the $(n_p+n_v+n_r+(n_\lambda-r_G)+n_\mu+n_u)$-dimensional *position manifold*

$$\begin{aligned} \mathbb{M}_p &= \{(p,v,r,w,s,\lambda,\mu,u) \in \mathbb{R}^n \times \mathbb{U} : \; 0 = d(p,v,r,w,s,\lambda,\mu,u), \\ &\qquad\qquad\qquad\qquad\qquad\quad 0 = c(p,s,u), \\ &\qquad\qquad\qquad\qquad\qquad\quad 0 = g(p,s,u)\} \\ &\subset \mathbb{R}^n \times \mathbb{U}. \end{aligned} \tag{4.60}$$

Furthermore, with respect to the constraints on velocity level, we have that

$$\frac{\partial}{\partial v} \begin{bmatrix} g^I \\ HZv + h \end{bmatrix} = \begin{bmatrix} GZ \\ HZ \end{bmatrix} \tag{4.61}$$

has rank $r_G + r_H$. Therefore, the state variables x (in particular, the velocity variables v) are restricted onto the $(n_p+n_v+n_r+n_w+n_s+(n_\lambda-r_G)+(n_\mu-r_H)+2n_u)$-dimensional *velocity manifold*

$$\begin{aligned} \mathbb{M}_v &= \{(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^1) \in \mathbb{R}^n \times \mathbb{U}^1 \; : \\ &\qquad\qquad 0 = H(p,s,u)Z(p)v + h(p,s,u), \\ &\qquad\qquad 0 = g^I(p,v,s,\mathfrak{u}^1)\} \\ &\subset \mathbb{R}^n \times \mathbb{U}^1. \end{aligned} \tag{4.62}$$

Furthermore, it follows from Assumption 4.55d that

$$\frac{\partial}{\partial [\lambda^T \; \mu^T]^T} \begin{bmatrix} g^{I\!I} \\ h^I \end{bmatrix} = - \begin{bmatrix} GZM^{-1}G_\lambda & GZM^{-1}H_\mu \\ HZM^{-1}G_\lambda & HZM^{-1}H_\mu \end{bmatrix} \tag{4.63}$$

has rank $r_G + r_H$. Therefore, the holonomic and nonholonomic constraints on acceleration level form an additional $(n_p+n_v+n_r+n_w+n_s+(n_\lambda-r_G)+(n_\mu-r_H)+3n_u)$-dimensional manifold. Let us call it the *acceleration manifold*

$$\begin{aligned} \mathbb{M}_a &= \{(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^2) \in \mathbb{R}^n \times \mathbb{U}^2 \\ &\qquad\qquad : \; 0 = h^I(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^1), \\ &\qquad\qquad \quad 0 = g^{I\!I}(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^2)\} \\ &\subset \mathbb{R}^n \times \mathbb{U}^2. \end{aligned} \tag{4.64}$$

In summary, the solution $(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^2)$ has to satisfy all constraints, i.e., all constraints given in (4.60), (4.62), and (4.64). Therefore, we have

$$(p(t),v(t),r(t),w(t),s(t),\lambda(t),\mu(t),\mathfrak{u}^2(t)) \overset{!}{\in} (\mathbb{M}_p \times \mathbb{U}^{(1)} \times \mathbb{U}^{(2)}) \cap (\mathbb{M}_v \times \mathbb{U}^{(2)}) \cap \mathbb{M}_a$$

for all $t \in \mathbb{I}$. This allows to introduce the solution manifold.

Lemma 4.2.14 (Solution Manifold of the equations of motion) *The* solution manifold *of the equations of motion of modeling level 4 (4.43) satisfying Assump-*

tion 4.2.4 is given by

$$\begin{aligned}\mathbb{M} = \{(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^2) \in \mathbb{R}^n \times \mathbb{U}^2 \ : \ 0 &= d(p,v,r,w,s,\lambda,\mu,u), \\ 0 &= c(p,s,u), \\ 0 &= H(p,s,u)Z(p)v + h(p,s,u), \\ 0 &= g(p,s,u), \\ 0 &= h^I(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^1), \\ 0 &= g^I(p,v,s,\mathfrak{u}^1), \\ 0 &= g^{I\!I}(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^2)\}.\end{aligned} \tag{4.65}$$

Proof: The proof follows from the considerations above or by applying Procedure 3.5.11 for the equations of motion, see (4.71), (4.74), and (4.77). □

Remark 4.2.15 Note that in particular, only p, v, w, s, λ, and μ are restricted by constraints. The dynamical force variables r are unrestricted. □

Lemma 4.2.16 (Positional and motional degrees of freedom) *Let the equations of motion of modeling level 4 (4.43) satisfy Assumption 4.2.4 with noncontradictory constraints (4.43f) and (4.43g). Then the number of positional degrees of freedom is $n_{f_p} = n_p - r_G$ and the number of motional degrees of freedom is $n_{f_v} = n_v - r_G - r_H$.*

Proof: The freedom of choice in the position variables is $n_{f_p} = n_p - r_G$ because of the rank of the matrix (4.59). On the other hand the freedom of choice in the velocity variables is $n_{f_v} = n_v - r_G - r_H$ because of the rank of the matrix (4.61). Then the assertion follows with Definition 4.1.7. □

Remark 4.2.17 In the case that $r_H = 0$, we have $n_f = n_{f_p} = n_{f_v}$ degrees of freedom of the multibody system. □

Suppose that the constraints, i.e., both holonomic and nonholonomic constraints, are uniformly redundant and that $g(p,s(p,u),u) \in \mathcal{C}^2(\mathbb{M}_p, \mathbb{R}^{n_\lambda})$ and $H(p,s(p,u),u) \in \mathcal{C}^1(\mathbb{M}_p, \mathbb{R}^{n_\mu,n_p})$. Then, from Lemma 2.1.4 we obtain the existence of nonsingular transformation matrices

$$\begin{bmatrix} S_\lambda(p,u) \\ \bar{S}_\lambda(p,u) \end{bmatrix} \in \mathcal{C}^1(\mathbb{M}_p, \mathbb{R}^{n_\lambda,n_\lambda}) \quad \text{and} \quad \begin{bmatrix} S_\mu(p,u) \\ \bar{S}_\mu(p,u) \end{bmatrix} \in \mathcal{C}^1(\mathbb{M}_p, \mathbb{R}^{n_\mu,n_\mu}) \tag{4.66a}$$

with $S_\lambda(p,u) \in \mathcal{C}^1(\mathbb{M}_p, \mathbb{R}^{r_G,n_\lambda})$ and $S_\mu(p,u) \in \mathcal{C}^1(\mathbb{M}_p, \mathbb{R}^{r_H,n_\mu})$ such that the matrices

$$\begin{bmatrix} S_\lambda(p,u) \\ \bar{S}_\lambda(p,u) \end{bmatrix} G(p,s(p,u),u) = \begin{bmatrix} \widetilde{G}(p,s(p,u),u) \\ 0 \end{bmatrix} \tag{4.66b}$$

and

$$\begin{bmatrix} S_\mu(p,u) \\ \bar{S}_\mu(p,u) \end{bmatrix} H(p,s(p,u),u) = \begin{bmatrix} \widetilde{H}(p,s(p,u),u) \\ 0 \end{bmatrix}. \tag{4.66c}$$

Here, $\widetilde{G}(p,s(p,u),u) \in \mathcal{C}^1(\mathbb{M}_p, \mathbb{R}^{r_G,n_p})$ and $\widetilde{H}(p,s(p,u),u) \in \mathcal{C}^1(\mathbb{M}_p, \mathbb{R}^{r_H,n_v})$ have full (row) rank, i.e., the matrix function $S_\lambda(p,u)g(p,s(p,u),u)$ and the matrix function $S_\mu(p,u)(H(p,s(p,u),u)Z(p)v + h(p,s(p,u),u))$ are nonredundant with respect

to p and v, respectively. These selectors S_λ and S_μ are (nonuniquely) defined by the conditions

$$\operatorname{rank}(S_\lambda(p,u)G(p,s(p,u),u)) = r_G,\ S_\lambda(p,u) \in \mathcal{C}^1(\mathbb{M}_p, \mathbb{R}^{r_G,n_\lambda}), \tag{4.67}$$
$$\operatorname{rank}(S_\mu(p,u)H(p,s(p,u),u)) = r_H,\ S_\mu(p,u) \in \mathcal{C}^1(\mathbb{M}_p, \mathbb{R}^{r_H,n_\mu}), \tag{4.68}$$

with r_G, r_H defined in (4.45) and $s(p,u)$ defined by (4.43e). Furthermore, the following lemmata show that these selectors applied to the hidden constraints on velocity level (4.50c) and applied to the hidden constraints on acceleration level (4.51d) and (4.53e) yield nonredundant constraints.

Lemma 4.2.18 (Nonredundant selected constraints on velocity level) *Let the holonomic constraints on velocity level be given by (4.50c) with $g \in \mathcal{C}^1(\mathbb{M}_p, \mathbb{R}^{n_\lambda})$ using (4.43e). Let them be possibly redundant but noncontradictory. Then a selector $S_\lambda(p,u)$ satisfying (4.67) applied to the holonomic constraints on velocity level $0 = g^I(p,v,s,\mathfrak{u}^1)$ yields the nonredundant selected holonomic constraints on velocity level*

$$0 = S_\lambda(p,u)g^I(p,v,s,\mathfrak{u}^1)$$

for the velocity variables v.

Proof: The selected constraints on velocity level, together with (4.50c), are given by

$$S_\lambda(p,u)g^I(p,v,s(p,u),\mathfrak{u}^1) = S_\lambda(p,u)\big(GZv + (g_{,u} - g_{,s}c_{,s}^{-1}c_{,u})\dot{u}\big).$$

The total derivative with respect to v yields

$$\frac{\mathrm{d}}{\mathrm{d}v}(S_\lambda(p,u)g^I(p,v,s(p,u),\mathfrak{u}^1)) = S_\lambda(p,u)G(p,s(p,u),u)Z(p).$$

Because of condition (4.67) and the full rank of the matrix $Z(p) \in \mathbb{R}^{n_p,n_v}$, i.e., $\operatorname{rank}(Z(p)) = n_p$, it follows that

$$\operatorname{rank}\big(\frac{\mathrm{d}}{\mathrm{d}v}S_\lambda(p,u)g^I(p,v,s(p,u),\mathfrak{u}^1)\big) = r_G$$

and the assertion follows from Definition 2.3.10. □

Lemma 4.2.19 (Nonredundant selected constraints on acceleration level) *Let the holonomic constraints on acceleration level be given by (4.51d) with $g \in \mathcal{C}^2(\mathbb{M}_p, \mathbb{R}^{n_\lambda})$ using (4.43e). Furthermore, let the nonholonomic constraints on acceleration level be given by (4.53e) with $\breve{h} \in \mathcal{C}^1(\mathbb{M}_p, \mathbb{R}^{n_\mu})$ with (4.43e). Then selectors $S_\lambda(p,u)$ and $S_\mu(p,u)$ satisfying (4.67) and (4.68), respectively, applied to the constraints on acceleration level $0 = g^{I\!I}(p,v,r,w,s(p,u),\lambda,\mu,\mathfrak{u}^2)$ and $0 = h^I(p,v,r,w,s(p,u),\lambda,\mu,\mathfrak{u}^1)$ yield the nonredundant selected constraints on acceleration level*

$$0 = \begin{bmatrix} S_\lambda(p,u)g^{I\!I}(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^2) \\ S_\mu(p,u)h^I(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^1) \end{bmatrix}$$

for the Lagrange multipliers λ and μ.

Proof: The selected holonomic constraints on acceleration level together with (4.51d) are

$$\begin{aligned} &S_\lambda(p,u)g^{I\!I}(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^2) \\ &\quad = S_\lambda(p,u)\big((g^I_{,p} - g^I_{,s}c^{-1}_{,s}c_{,p})Zv + GZM^{-1}(f - Z^TG^T\lambda - Z^TH^T\mu) \\ &\qquad\qquad + g^I_{,\mathfrak{u}^1}\dot{\mathfrak{u}}^1 - g^I_{,s}c^{-1}_{,s}c_{,u}\dot{u}\big). \end{aligned}$$

Furthermore, the selected nonholonomic constraints on acceleration level together with (4.53e) are

$$\begin{aligned} &S_\mu(p,u)h^I(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^1) \\ &\quad = S_\mu(p,u)\big((\breve{h}_{,p} - \breve{h}_{,s}c^{-1}_{,s}c_{,p})Zv + HZM^{-1}(f - Z^TG^T\lambda - Z^TH^T\mu) \\ &\qquad\qquad + (\breve{h}_{,u} - \breve{h}_{,s}c^{-1}_{,s}c_{,u})\dot{u}\big). \end{aligned}$$

The total derivative of the acceleration constraints with respect to λ and μ yields

$$\frac{\mathrm{d}}{\mathrm{d}\begin{bmatrix} \lambda^T & \mu^T \end{bmatrix}^T}\begin{bmatrix} S_\lambda g^{I\!I} \\ S_\mu h^I \end{bmatrix} = \begin{bmatrix} -S_\lambda GZM^{-1}G_\lambda & -S_\lambda GZM^{-1}H_\mu \\ -S_\mu HZM^{-1}G_\lambda & -S_\mu HZM^{-1}H_\mu \end{bmatrix}.$$

Because of condition (4.67), the regularity condition (4.55d) is given by

$$\operatorname{rank}\left(\begin{bmatrix} M & G_\lambda & H_\mu \\ S_\lambda GZ & 0 & 0 \\ S_\mu HZ & 0 & 0 \end{bmatrix}\right) = n_v + r_G + r_H.$$

By use of block Gauß elimination it follows that

$$\operatorname{rank}\left(\begin{bmatrix} M & G_\lambda & H_\mu \\ 0 & -S_\lambda GZM^{-1}G_\lambda & -S_\lambda GZM^{-1}H_\mu \\ 0 & -S_\mu HZM^{-1}G_\lambda & -S_\mu HZM^{-1}H_\mu \end{bmatrix}\right) = n_v + r_G + r_H$$

and we get

$$\operatorname{rank}\left(\begin{bmatrix} -S_\lambda GZM^{-1}G_\lambda & -S_\lambda GZM^{-1}H_\mu \\ -S_\mu HZM^{-1}G_\lambda & -S_\mu HZM^{-1}H_\mu \end{bmatrix}\right) = r_G + r_H. \tag{4.69}$$

Note that the matrix in (4.69) is of size $r_G + r_H \times n_\lambda + n_\mu$ with $r_G + r_H \leq n_\lambda + n_\mu$. Then the assertion follows from Definition 2.3.10. □

Remark 4.2.20 Note that if there are redundant constraints, then the selected constraints on acceleration level are not sufficient to determine the Lagrange multiplier uniquely. Therefore, the solution of the equations of motion with redundant constraints is not unique. In Lemma 4.2.30 and in the Theorems 4.2.32 and 4.2.33 below we discuss the existence and the uniqueness of a solution of the equations of motion of modeling level 4 (4.43). □

In preparation for further investigations associated with the equations of motion let us apply Procedure 3.5.11 to the equations of motion of modeling level 4 (4.43) with nonsingular mass matrix M, i.e., satisfying Assumption 4.2.4.

Procedure 4.2.21 Let the equations of motion of modeling level 4 (4.43) satisfy Assumption 4.2.4. Furthermore, let the holonomic constraints (4.43g) be nonredundant or uniformly redundant, noncontradictory, and three times continuously

differentiable and let the nonholonomic constraints (4.43f) be nonredundant or uniformly redundant, noncontradictory, and twice continuously differentiable.
According to Procedure 3.5.11 it follows that

$$E^0(x,u)\dot{x} = k^0(x,u) \tag{4.70}$$

with

$$E^0(x,u) = E(x,u) = \begin{bmatrix} I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & M & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$k^0(x,u) = k(x,u) = \begin{bmatrix} Zv \\ f - Z^T G^T \lambda - Z^T H^T \mu \\ b \\ d \\ c \\ HZv + h \\ g \end{bmatrix},$$

where $x = \begin{bmatrix} p^T & v^T & r^T & w^T & s^T & \lambda^T & \mu^T \end{bmatrix}^T$. Because of the assumed nonsingularity of the mass matrix $M(p,u)$, we have $E(x,u)$ already in partitioned form. Therefore, we have $Z^0 = I$ and we get

$$\widetilde{k}_1^0 = \begin{bmatrix} Zv \\ f - Z^T G^T \lambda - Z^T H^T \mu \\ b \end{bmatrix},$$

$$\widetilde{k}_2^0 = \begin{bmatrix} d \\ c \\ HZv + h \\ g \end{bmatrix}. \tag{4.71}$$

In particular, we obtain the algebraic part (4.71) which is nonredundant or uniformly redundant, noncontradictory, and continuously differentiable. Furthermore, we obtain the manifold

$$\mathbb{M}_0 = \widetilde{\mathbb{M}}_0 = \{x \in \mathbb{R}^n \times \mathbb{U} \ : \ 0 = d, \ 0 = c, \ 0 = h, \ 0 = g\}. \tag{4.72}$$

By differentiation of $\widetilde{k}_2^0$ with respect to t, we get the transformed DAE according to (3.43) in the form

$$E^1(x,u)\dot{x} = k^1(x, \mathfrak{u}^1), \tag{4.73}$$

with

$$E^1(x,u) = \begin{bmatrix} I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & M & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 \\ d_{,p} & d_{,v} & d_{,r} & d_{,w} & d_{,s} & d_{,\lambda} & d_{,\mu} \\ c_{,p} & 0 & 0 & 0 & c_{,s} & 0 & 0 \\ \breve{h}_{,p} & HZ & 0 & 0 & \breve{h}_{,s} & 0 & 0 \\ g_{,p} & 0 & 0 & 0 & g_{,s} & 0 & 0 \end{bmatrix},$$

$$k^1(x,\mathfrak{u}^1) = \begin{bmatrix} Zv \\ f - Z^T G^T \lambda - Z^T H^T \mu \\ b \\ -d_{,u}\dot{u} \\ -c_{,u}\dot{u} \\ -\breve{h}_{,u}\dot{u} \\ -g_{,u}\dot{u} \end{bmatrix},$$

and $\breve{h} = HZv + h$, see (4.44). By Assumption 4.2.4, we have the nonsingularity of M, $d_{,w}$, and $c_{,s}$. Therefore, we get a transformation matrix

$$Z^1(x,u) = \begin{bmatrix} I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & I_{n_v} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 \\ -d_{,p} & -d_{,v}M^{-1} & -d_{,r} & I & 0 & 0 & 0 \\ -c_{,p} & 0 & 0 & 0 & I & 0 & 0 \\ \breve{h}_{,p} - \breve{h}_{,s}c_{,s}^{-1}c_{,p} & HZM^{-1} & 0 & 0 & \breve{h}_{,s}c_{,s}^{-1} & -I & 0 \\ G & 0 & 0 & 0 & g_{,s}c_{,s}^{-1} & 0 & -I \end{bmatrix}.$$

Premultiplication with this transformation matrix yields the DAE in partitioned form

$$\widetilde{E}^1(x,u)\dot{x} = \widetilde{k}^1(x,\mathfrak{u}^1),$$

with

$$\widetilde{E}^1(x,u) = \begin{bmatrix} I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & M & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & d_{,w} & d_{,s} & d_{,\lambda} & d_{,\mu} \\ 0 & 0 & 0 & 0 & c_{,s} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\widetilde{k}^1(x,\mathfrak{u}^1) = \begin{bmatrix} Zv \\ f - Z^T G^T \lambda - Z^T H^T \mu \\ b \\ -d_{,p}Zv - d_{,v}M^{-1}(f - Z^T G^T \lambda - Z^T H^T \mu) - d_{,r}b - d_{,u}\dot{u} \\ -c_{,p}Zv - c_{,u}\dot{u} \\ h^I \\ g^I \end{bmatrix}.$$

With respect to the terms h^I and g^I compare with (4.50c) and (4.53e). In particular, for the algebraic part we obtain

$$\widetilde{k}_2^1(x,\mathfrak{u}^1) = \begin{bmatrix} h^I \\ g^I \end{bmatrix} \tag{4.74}$$

which is nonredundant or uniformly redundant, noncontradictory, and continuously differentiable. Furthermore, we get the manifold of level 1

$$\widetilde{\mathbb{M}}_1 = \{x \in \mathbb{R}^n \times \mathbb{U}^1 \ : \ 0 = h^I, \ 0 = g^I\}. \tag{4.75}$$

and

$$\begin{aligned} \mathbb{M}_1 &= (\mathbb{M}_0 \times \mathbb{U}^{(1)}) \cap \widetilde{\mathbb{M}}_1 \\ &= \{x \in \mathbb{R}^n \times \mathbb{U}^1 \ : \ 0 = d, \ 0 = c, \ 0 = h, \ 0 = g, \ 0 = h^I, \ 0 = g^I\}. \end{aligned}$$

Further differentiation of the algebraic part $\widetilde{k}_2^1(x, \mathfrak{u}^1)$ according to (3.43) gives

$$E^2(x, \mathfrak{u}^1)\dot{x} = k^2(x, \mathfrak{u}^2), \tag{4.76}$$

with

$$E^2(x, \mathfrak{u}^1) = \begin{bmatrix} I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & M & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & d_{,w} & d_{,s} & d_{,\lambda} & d_{,\mu} \\ 0 & 0 & 0 & 0 & c_{,s} & 0 & 0 \\ h^I_{,p} & h^I_{,v} & h^I_{,r} & h^I_{,w} & h^I_{,s} & h^I_{,\lambda} & h^I_{,\mu} \\ g^I_{,p} & g^I_{,v} & 0 & 0 & g^I_{,s} & 0 & 0 \end{bmatrix},$$

$$k^2(x, \mathfrak{u}^2) = \begin{bmatrix} Zv \\ f - Z^T G^T \lambda - Z^T H^T \mu \\ b \\ -d_{,p} Zv - d_{,v} M^{-1}(f - Z^T G^T \lambda - Z^T H^T \mu) - d_{,r} b - d_{,u}\dot{u} \\ -c_{,p} Zv - c_{,u}\dot{u} \\ -h^I_{,u}\dot{u} \\ -g^I_{,u}\dot{u} \end{bmatrix}.$$

Premultiplication with the transformation matrix

$$Z^2(x, \mathfrak{u}^1) = \begin{bmatrix} I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & I_{n_v} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{n_w} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{n_s} & 0 & 0 \\ S_\mu h^I_{,p} & S_\mu h^I_{,v} M^{-1} & S_\mu h^I_{,r} & S_\mu h^I_{,w} d^{-1}_{,w} & Z^2_{65} & -S_\mu & 0 \\ \bar{S}_\mu h^I_{,p} & \bar{S}_\mu h^I_{,v} M^{-1} & \bar{S}_\mu h^I_{,r} & \bar{S}_\mu h^I_{,w} d^{-1}_{,w} & Z^2_{75} & -\bar{S}_\mu & 0 \\ g^I_{,p} & g^I_{,v} M^{-1} & 0 & 0 & g^I_{,s} c^{-1}_{,s} & 0 & -I \end{bmatrix},$$

with

$$\begin{aligned} Z^2_{65} &= -S_\mu(-h^I_{,s} + h^I_{,w} d^{-1}_{,w} d_{,s}) c^{-1}_{,s} \\ Z^2_{75} &= -\bar{S}_\mu(-h^I_{,s} + h^I_{,w} d^{-1}_{,w} d_{,s}) c^{-1}_{,s} \end{aligned}$$

and with S_μ satisfying (4.68) and $\bar{S}_\mu$ chosen such that the matrix $\begin{bmatrix} S_\mu^T & \bar{S}_\mu^T \end{bmatrix}^T$ is nonsingular and such that $\bar{S}_\mu H = 0$, see (4.66c), yields

$$\widetilde{E}^2(x, \mathfrak{u}^1)\dot{x} = \widetilde{k}^2(x, \mathfrak{u}^2),$$

with

$$\widetilde{E}^2(x,\mathfrak{u}^1) = \begin{bmatrix} I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & M & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & d_{,w} & d_{,s} & d_{,\lambda} & d_{,\mu} \\ 0 & 0 & 0 & 0 & c_{,s} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & S_\mu HZM^{-1}G_\lambda & S_\mu HZM^{-1}H_\mu \\ 0 & 0 & 0 & 0 & 0 & \bar{S}_\mu HZM^{-1}G_\lambda & \bar{S}_\mu HZM^{-1}H_\mu \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & M & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_v} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & d_{,w} & d_{,s} & d_{,\lambda} & d_{,\mu} \\ 0 & 0 & 0 & 0 & c_{,s} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & S_\mu HZM^{-1}G_\lambda & S_\mu HZM^{-1}H_\mu \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\widetilde{k}^2(x,\mathfrak{u}^2) = \begin{bmatrix} Zv \\ f - Z^TG^T\lambda - Z^TH^T\mu \\ b \\ -d_{,p}Zv - d_{,v}M^{-1}(f - Z^TG^T\lambda - Z^TH^T\mu) - d_{,r}b - d_{,u}\dot{u} \\ -c_{,p}Zv - c_{,u}\dot{u} \\ S_\mu\widetilde{h}^{I\!I} \\ \bar{S}_\mu\widetilde{h}^{I\!I} \\ g^{I\!I} \end{bmatrix},$$

where $g^{I\!I}$ is given by (4.51d) and $\widetilde{h}^{I\!I}$ is given by (4.54e). From the fact that the nonholonomic constraints are noncontradictory it follows that $\bar{S}_\mu\widetilde{h}^{I\!I} = 0$ for all $(x,\mathfrak{u}^2) \in \mathbb{M}_1 \times \mathbb{U}^{(2)}$. Therefore, the second last block column is trivially satisfied and no longer of interest, see Remark 3.5.12c. In principle, the condition $\bar{S}_\mu\widetilde{h}^{I\!I} = 0$ corresponds to a consistency condition for the control variable u and its derivative $\dot{u}$.

On the other hand, the algebraic equation $0 = g^{I\!I}$ is not trivially satisfied for all $(x,\mathfrak{u}^2) \in \mathbb{M}_1 \times \mathbb{U}^{(2)}$ such that we have to continue with the procedure. Following Remark 3.5.12c and keeping the trivially satisfied equation $0 = \bar{S}_\mu\widetilde{h}^{I\!I}$ unconsidered for the remainder of the procedure, we obtain the algebraic part

$$0 = \widetilde{k}_2^2(x,\mathfrak{u}^2) = \begin{bmatrix} g^{I\!I} \end{bmatrix}, \tag{4.77}$$

which is nonredundant or uniformly redundant, noncontradictory, and continuously differentiable. Furthermore, we get the manifold

$$\widetilde{\mathbb{M}}_2 = \{x \in \mathbb{R}^n \times \mathbb{U}^2 \ : \ 0 = g^{I\!I}\} \tag{4.78}$$

and

$$\begin{aligned} \mathbb{M}_2 &= (\mathbb{M}_1 \times \mathbb{U}^{(2)}) \cap \widetilde{\mathbb{M}}_2 \\ &= \{x \in \mathbb{R}^n \times \mathbb{U}^2 \ : \ 0 = d,\ 0 = c,\ 0 = h,\ 0 = g,\ 0 = h^I,\ 0 = g^I,\ 0 = g^{I\!I}\}. \end{aligned}$$

Further differentiation of the algebraic part $\widetilde{k}_2^2(x,\mathfrak{u}^2)$ according to (3.43) gives

$$E^3(x,\mathfrak{u}^2)\dot{x} = k^3(x,\mathfrak{u}^3) \tag{4.79}$$

with

$$E^3(x,\mathfrak{u}^2)=\begin{bmatrix} I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & M & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & d_{,w} & d_{,s} & d_{,\lambda} & d_{,\mu}\\ 0 & 0 & 0 & 0 & c_{,s} & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & S_\mu HZM^{-1}G_\lambda & S_\mu HZM^{-1}H_\mu\\ 0 & 0 & 0 & 0 & 0 & 0 & 0\\ g^{I\!I}_{,p} & g^{I\!I}_{,v} & g^{I\!I}_{,r} & g^{I\!I}_{,w} & g^{I\!I}_{,s} & g^{I\!I}_{,\lambda} & g^{I\!I}_{,\mu} \end{bmatrix},$$

$$k^3(x,\mathfrak{u}^3)=\begin{bmatrix} Zv\\ f-Z^TG^T\lambda-Z^TH^T\mu\\ b\\ -d_{,p}Zv-d_{,v}M^{-1}(f-Z^TG^T\lambda-Z^TH^T\mu)-d_{,r}b-d_{,u}\dot{u}\\ -c_{,p}Zv-c_{,u}\dot{u}\\ S_\mu\widetilde{h}^{I\!I}\\ \bar{S}_\mu\widetilde{h}^{I\!I}\\ -g^{I\!I}_{,u}\dot{u} \end{bmatrix}.$$

Premultiplication with the transformation matrix

$$Z^3(x,\mathfrak{u}^2)=\begin{bmatrix} I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & I_{n_v} & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & I_{n_w} & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & I_{n_s} & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & I_{r_H} & 0 & 0\\ S_\lambda g^{I\!I}_{,p} & S_\lambda g^{I\!I}_{,v}M^{-1} & S_\lambda g^{I\!I}_{,r} & S_\lambda g^{I\!I}_{,w}d^{-1}_{,w} & Z^3_{75} & 0 & 0 & -S_\lambda\\ 0 & 0 & 0 & 0 & 0 & 0 & I_{n_\mu-r_H} & 0\\ \bar{S}_\lambda g^{I\!I}_{,p} & \bar{S}_\lambda g^{I\!I}_{,v}M^{-1} & \bar{S}_\lambda g^{I\!I}_{,r} & \bar{S}_\lambda g^{I\!I}_{,w}d^{-1}_{,w} & Z^3_{95} & 0 & 0 & -\bar{S}_\lambda \end{bmatrix}$$

with

$$\begin{aligned} Z^3_{75} &= S_\lambda(g^{I\!I}_{,s}-g^{I\!I}_{,w}d^{-1}_{,w}d_{,s})c^{-1}_{,s}\\ Z^3_{95} &= \bar{S}_\lambda(g^{I\!I}_{,s}-g^{I\!I}_{,w}d^{-1}_{,w}d_{,s})c^{-1}_{,s} \end{aligned}$$

and with S_λ satisfying (4.67) and $\bar{S}_\lambda$ chosen such that the matrix $\begin{bmatrix} S_\lambda^T & \bar{S}_\lambda^T \end{bmatrix}^T$ is nonsingular and such that $\bar{S}_\lambda G=0$, see (4.66b), yields

$$\widetilde{E}^3(x,\mathfrak{u}^2)\dot{x}=\widetilde{k}^3(x,\mathfrak{u}^3),$$

where

$$\widetilde{E}^3(x,\mathfrak{u}^2) = \begin{bmatrix} I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & M & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & d_{,w} & d_{,s} & d_{,\lambda} & d_{,\mu} \\ 0 & 0 & 0 & 0 & c_{,s} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & S_\mu HZM^{-1}G_\lambda & S_\mu HZM^{-1}H_\mu \\ 0 & 0 & 0 & 0 & 0 & S_\lambda GZM^{-1}G_\lambda & S_\lambda GZM^{-1}H_\mu \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \bar{S}_\lambda GZM^{-1}G_\lambda & \bar{S}_\lambda GZM^{-1}H_\mu \end{bmatrix}$$

$$= \begin{bmatrix} I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & M & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & d_{,w} & d_{,s} & d_{,\lambda} & d_{,\mu} \\ 0 & 0 & 0 & 0 & c_{,s} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & S_\mu HZM^{-1}G_\lambda & S_\mu HZM^{-1}H_\mu \\ 0 & 0 & 0 & 0 & 0 & S_\lambda GZM^{-1}G_\lambda & S_\lambda GZM^{-1}H_\mu \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\widetilde{k}^3(x,\mathfrak{u}^3) = \begin{bmatrix} Zv \\ f - Z^TG^T\lambda - Z^TH^T\mu \\ b \\ -d_{,p}Zv - d_{,v}M^{-1}(f - Z^TG^T\lambda - Z^TH^T\mu) - d_{,r}b - d_{,u}\dot{u} \\ -c_{,p}Zv - c_{,u}\dot{u} \\ S_\mu\widetilde{h}^{II} \\ S_\lambda\widetilde{g}^{III} \\ \bar{S}_\mu\widetilde{h}^{II} \\ \bar{S}_\lambda\widetilde{g}^{III} \end{bmatrix}.$$

where $\widetilde{g}^{III}$ is given by (4.52e). From the fact that the holonomic constraints are noncontradictory it follows that $\bar{S}_\lambda\widetilde{g}^{III} = 0$ for all $(x,\mathfrak{u}^3) \in \mathbb{M}_2 \times \mathbb{U}^{(3)}$. Therefore, in addition to the second last column also the last block column is trivially satisfied and we obtain the algebraic part

$$0 = \widetilde{k}_2^3(x,\mathfrak{u}^3) = \begin{bmatrix} \bar{S}_\mu\widetilde{h}^{II} \\ \bar{S}_\lambda\widetilde{g}^{III} \end{bmatrix},$$

which is satisfied for all $(x,\mathfrak{u}^3) = (p,v,r,w,s,\lambda,\mu,\mathfrak{u}^3) \in \mathbb{M}_2 \times \mathbb{U}^{(3)}$ and the Procedure 3.5.11 terminates with $\nu = 3$. □

Lemma 4.2.22 (Maximal constraint level of equations of motion) *Let the equations of motion of modeling level 4 (4.43) satisfy Assumption 4.2.4. Then the equations of motion of modeling level 4 (4.43) form a system of differential-algebraic equations with maximal constraint level* $\nu_c = 2$.

Proof: The proof follows from Procedure 4.2.21 in view of Procedure 3.5.11 and Definition 3.5.27. □

Remark 4.2.23 a) If the equations of motion of modeling level 3 satisfy Assumptions (4.48a),(4.48b) and (4.49) and if there appear holonomic constraints (4.43g), then the equations of motion (4.43) form a system of DAEs of s-index $\nu_s = 2$ and d-index $\nu_d = 3$. This follows from Procedure 4.2.21 with the assumption that the

constraints are nonredundant. In the case of purely nonholonomic systems the equations of motion of modeling level 3 (4.43) satisfying Assumptions (4.48a),(4.48b) and (4.49) form a system of DAEs of s-index $\nu_s = 1$ and d-index $\nu_d = 2$.
b) In the case of equations of motion of modeling level 4 (4.43) satisfying Assumption 4.2.4, with redundant but noncontradictory constraints (4.43f) and (4.43g) having constant rank the d-index is not defined but the s-index is still $\nu_s = 2$ if holonomic constraints (4.43g) appear and the s-index is $\nu_s = 1$ for purely nonholonomic systems. □

Remark 4.2.24 Note that the manifolds $\widetilde{\mathbb{M}}_i$, i=0,1,2, see (4.72), (4.75), and (4.78), do not correspond to the manifolds $\mathbb{M}_p$, $\mathbb{M}_v$, and $\mathbb{M}_a$, see (4.60), (4.62), and (4.64). But the intersection of all manifolds corresponds to the solution manifold (4.65), i.e.,

$$\begin{aligned}\mathbb{M} &= (\mathbb{M}_0 \times \mathbb{U}^{(1)} \times \mathbb{U}^{(2)}) \cap (\mathbb{M}_1 \times \mathbb{U}^{(2)}) \cap \mathbb{M}_2 \\ &= (\mathbb{M}_p \times \mathbb{U}^{(1)} \times \mathbb{U}^{(2)}) \cap (\mathbb{M}_v \times \mathbb{U}^{(2)}) \cap \mathbb{M}_a.\end{aligned}$$

□

Note that after every iteration step in Procedure 4.2.21, the maximal constraint level is reduced by one. In particular, we have the original equations of motion (4.43) or (4.70) with $\nu_c = 2$. After the first iteration step we get (4.73) with $\nu_c = 1$ and after a further iteration step we get (4.76) with $\nu_c = 0$. In particular, (4.76) is already strangeness free. Furthermore, after the third iteration step of Procedure 4.2.21 we get (4.79) with $\nu_c = -1$, i.e., (4.79) is also strangeness-free. In particular, in the case of nonredundant constraints, i.e., in the case of the equations of motion of modeling level 3 (4.43), because of the nonsingularity of the leading matrix $E^3(x, \mathfrak{u}^2)$ in (4.79) the DAE (4.79) corresponds to an ODE in implicit form, the underlying ODE, see Definition 3.5.23.

Lemma 4.2.25 (Complete minimal reduced derivative array of the EoM) *The complete minimal reduced derivative array of the equations of motion of modeling level 4 (4.43) satisfying Assumption 4.2.4 is given by*

$$0 = \tilde{\mathfrak{F}}_2(p, v, r, w, s, \lambda, \mu, \dot{p}, \dot{v}, \dot{r}, \mathfrak{u}^2) = \begin{bmatrix} -\dot{p} + Zv \\ -M\dot{v} + f - Z^T G^T \lambda - Z^T H^T \mu \\ -\dot{r} + b \\ d \\ c \\ HZv + h \\ g \\ h^I \\ g^I \\ g^{II} \end{bmatrix}, \quad (4.80)$$

where the hidden constraints are defined in (4.50), (4.51), and (4.53).

Proof: We get the assertion by Definition 3.5.39 with (4.71), (4.74), and (4.77). □

Because of the semi-implicit structure of the equations of motion of modeling level 4 (4.43) it is not necessary that the constraints of level 0 appear twice in the complete minimal reduced derivative array, as precisely defined in Definition 3.5.39, see Remark 3.5.40b.

Remark 4.2.26 Let the equations of motion of modeling level 4 (4.43) satisfy Assumption 4.2.4 and let the hidden constraints be defined in (4.50), (4.51), and (4.53). Then, if the initial values p_0, v_0, r_0, w_0, s_0, λ_0, and μ_0 are consistent, the constraints

$$\begin{aligned}
0 &= d(p_0, v_0, r_0, w_0, s_0, \lambda_0, \mu_0, u_0),\\
0 &= c(p_0, s_0, u_0),\\
0 &= H(p_0, s_0, u_0)Z(p_0)v_0 + h(p_0, s_0, u_0),\\
0 &= g(p_0, s_0, u_0),\\
0 &= h^I(p_0, v_0, r_0, w_0, s_0, \lambda_0, \mu_0, \mathfrak{u}_0^1),\\
0 &= g^I(p_0, v_0, s_0, \mathfrak{u}_0^1),\\
0 &= g^{II}(p_0, v_0, r_0, w_0, s_0, \lambda_0, \mu_0, \mathfrak{u}_0^2)
\end{aligned}$$

have to be satisfied with $u_0 = u(t_0)$, $\mathfrak{u}_0^1 = \mathfrak{u}^1(t_0)$, and $\mathfrak{u}_0^2 = \mathfrak{u}^2(t_0)$. □

Besides the original equations of motion, several formulations resulting from differentiation of the constraints are frequently used for analytical and numerical investigations. These formulations do not correspond to the formulations (4.73) or (4.76) obtained in Procedure 4.2.21. In the following we will introduce these formulations for the equations of motion of modeling level 4 (4.43).

Strangeness-index-1 formulation: If we use the holonomic constraints on velocity level (4.50a) instead of the holonomic constraints on position level (4.43g) in the equations of motion of modeling level 4 (4.43), then we get the equations of motion in the form

$$\begin{aligned}
\dot{p} &= Zv, && (4.81a)\\
M\dot{v} &= f - Z^T G^T \lambda - Z^T H^T \mu, && (4.81b)\\
\dot{r} &= b, && (4.81c)\\
0 &= d, && (4.81d)\\
0 &= c, && (4.81e)\\
0 &= HZv + h, && (4.81f)\\
0 &= g^I. && (4.81g)
\end{aligned}$$

Let us abbreviate the DAE (4.81) as EoM_1 in order to highlight that it has maximal constraint level one.

Remark 4.2.27 a) The associated EoM_1 (4.81) is a semi-implicit DAE of s-index one. Therefore, it is not strangeness-free. In the case of regular equations of motion the d-index of the associated EoM_1 (4.81) is two.
b) Note that the solution of the initial value problem for the associated EoM_1 (4.81) with the initial values (4.43h) which are consistent to the original equations of motion, see Lemma 4.2.26, is identical to the solution of the initial value problem for the equations of motion of modeling level 4 (4.43), see Lemma 3.5.10.
c) In general, the solution set is larger than the solution set of the original equations of motion, because of the loss of information of the holonomic constraints of position level. In particular, the set of solutions is given by $\{(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^2) \in \mathbb{R}^n \times \mathbb{U}^2 : 0 = d,\ 0 = c,\ 0 = HZv + h,\ 0 = g^I,\ 0 = h^I,\ 0 = g^{II}\}$. □

Strangeness-index-0 formulation: If one substitutes the holonomic constraints on position level (4.43g) in the equations of motion of modeling level 4 (4.43) by the holonomic constraints on acceleration level (4.51a) on the one hand and the nonholonomic constraints on velocity level (4.43f) by the nonholonomic constraints

on acceleration level (4.53a) on the other hand then we get the equations of motion in the form

$$\begin{aligned}
\dot{p} &= Zv, && \text{(4.82a)}\\
M\dot{v} &= f - Z^T G^T \lambda - Z^T H^T \mu, && \text{(4.82b)}\\
\dot{r} &= b, && \text{(4.82c)}\\
0 &= d, && \text{(4.82d)}\\
0 &= c, && \text{(4.82e)}\\
0 &= h^I, && \text{(4.82f)}\\
0 &= g^{I\!I}. && \text{(4.82g)}
\end{aligned}$$

Let us abbreviate the DAE (4.82) as EoM_0 in order to highlight that it has maximal constraint level zero.

Remark 4.2.28 a) The associated EoM_0 (4.82) corresponds to a semi-implicit DAE of s-index zero. Therefore, it is strangeness-free. In the case of regular equations of motion the d-index of the associated EoM_0 (4.82) is one.
b) Note that the solution of the initial value problem for the associated EoM_0 (4.82) with the initial values (4.43h) which are consistent to the original equations of motion, see Lemma 4.2.26, is identical to the solution of the initial value problem for the equations of motion of modeling level 4 (4.43), see Lemma 3.5.10.
c) In comparison to the s-index-1 formulation (4.81) the solution set again is increased because of the loss of information of the holonomic constraints of position level as well as the constraints on velocity level. In particular, the set of solutions is given by $\{(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^2) \in \mathbb{R}^n \times \mathbb{U}^2 \ : \ 0 = d,\ 0 = c,\ 0 = h^I,\ 0 = g^{I\!I}\}$. □

Underlying differential equation: Replacing the auxiliary equations (4.43d) by their derivative with respect to t

$$0 = d^I = \frac{\mathrm{d}}{\mathrm{d}t} d, \tag{4.83}$$

as well as the contact equations (4.43e) by their derivative with respect to t,

$$0 = c^I = \frac{\mathrm{d}}{\mathrm{d}t} c, \tag{4.84}$$

the holonomic constraints (4.43g) by (4.52e) and, finally, the nonholonomic constraints (4.43f) by (4.54e) in the equations of motion of modeling level 4 (4.43), we get the underlying differential equation as defined in Definition 3.5.23, in the form

$$\begin{aligned}
\dot{p} &= Zv, && \text{(4.85a)}\\
M\dot{v} &= f - Z^T G^T \lambda - Z^T H^T \mu, && \text{(4.85b)}\\
\dot{r} &= b, && \text{(4.85c)}\\
d_{,p}\dot{p} + d_{,v}\dot{v} + d_{,r}\dot{r} + d_{,w}\dot{w} + d_{,s}\dot{s} + d_{,\lambda}\dot{\lambda} + d_{,\mu}\dot{\mu} &= -d_{,u}\dot{u}, && \text{(4.85d)}\\
c_{,p}\dot{p} + c_{,s}\dot{s} &= -c_{,u}\dot{u}, && \text{(4.85e)}\\
HZM^{-1}G_\lambda\dot{\lambda} + HZM^{-1}H_\mu\dot{\mu} &= \widetilde{h}^{I\!I}, && \text{(4.85f)}\\
GZM^{-1}G_\lambda\dot{\lambda} + GZM^{-1}H_\mu\dot{\mu} &= \widetilde{g}^{I\!I\!I}. && \text{(4.85g)}
\end{aligned}$$

Remark 4.2.29 a) In the case of regular equations of motion the DAE (4.85) corresponds to the uODE, see Definition 3.5.23, and therefore, it is strangeness-free.
b) Note that the solution of the initial value problem for the associated underlying differential equations (4.85) with the initial values (4.43h), which are consistent to

the original equations of motion, see Lemma 4.2.26, is identical to the solution of the initial value problem for the equations of motion of modeling level 4 (4.43), see Lemma 3.5.10.
c) Because of the structure of the underlying differential equation, the solution is not restricted to a manifold such that the solution set is the entire $\mathbb{R}^n$. □

In [113] Lötstedt discussed the influence of redundant constraints on the existence and the uniqueness of solutions for the equations of motion of modeling level 0 (4.34). It was shown that redundant constraints only influence the uniqueness of the solution of the Lagrange multipliers. The constraint forces as well as the solution p and v are unique. Let us generalize the obtained results to the equations of motion of modeling level 4 (4.43). Furthermore, we will derive the existence and (partial) uniqueness of the solution for the equations of motion of modeling level 4 in Theorem 4.2.32, below.

Lemma 4.2.30 (Partial uniqueness for redundant constraints) *Let the equations of motion of modeling level 4 (4.43) satisfy Assumption 4.2.4 with noncontradictory constraints (4.43f) and (4.43g) having constant rank on $\mathbb{M}$. Furthermore, let the functions $f(p,v,r,w,s,\lambda,\mu,u)$ and $d(p,v,r,w,s,\lambda,\mu,u)$ be linear in w, λ, and μ. Suppose that for given (p,v,r,s,u) satisfying (4.43e) the function $(w,\lambda,\mu) \mapsto b(p,v,r,w,s,\lambda,\mu,u)$ is constant in $(w,\lambda,\mu) \in \{\mathbb{R}^{n_w} \times \mathbb{R}^{n_\lambda} \times \mathbb{R}^{n_\mu} \ : \ 0 = d(p,v,r,w,s,\lambda,\mu,u),\ 0 = g^{\mathrm{II}}(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^2),\ 0 = h^I(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^1)\}$. Then, for given $(p,v,r,\mathfrak{u}^2)$ the forces originating from the Lagrange multipliers λ and μ, i.e., $G_\lambda\lambda + H_\mu\mu$, the generalized velocity $\dot{p}$, the acceleration $\dot{v}$, as well as the contact variables s and $\dot{r}$ are uniquely defined.*

Proof: We have that $\dot{p}$ and s are uniquely determined by (4.43a) and (4.43e), respectively, independent of the Lagrange multipliers λ and μ and independent of the auxiliary variables w.
Furthermore, the functions f and d are linear in w, λ, and μ such that they have the form

$$\begin{aligned} &f(p,v,r,w,s,\lambda,\mu,u) \qquad (4.86)\\ &\quad = \widetilde{f}(p,v,r,s,u) + \widetilde{f}_w(p,v,r,s,u)w + \widetilde{f}_\lambda(p,v,r,s,u)\lambda + \widetilde{f}_\mu(p,v,r,s,u)\mu,\\ &d(p,v,r,w,s,\lambda,\mu,u) \qquad (4.87)\\ &\quad = \widetilde{d}(p,v,r,s,u) + \widetilde{d}_w(p,v,r,s,u)w + \widetilde{d}_\lambda(p,v,r,s,u)\lambda + \widetilde{d}_\mu(p,v,r,s,u)\mu. \end{aligned}$$

From (4.87), (4.43d), and from Assumption (4.55a), we obtain

$$w = -\widetilde{d}_w^{-1}(\widetilde{d} + \widetilde{d}_\lambda\lambda + \widetilde{d}_\mu\mu). \qquad (4.88)$$

From (4.43b) we get, with (4.86) and (4.88), that

$$M\dot{v} = \widetilde{f} - \widetilde{f}_w\widetilde{d}_w^{-1}\widetilde{d} - G_\lambda\lambda - H_\mu\mu.$$

Together with the constraints on acceleration level (4.51c) and (4.53d), we get

$$\begin{bmatrix} M & G_\lambda & H_\mu \\ GZ & 0 & 0 \\ HZ & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{v} \\ \lambda \\ \mu \end{bmatrix} = \begin{bmatrix} \widetilde{f} - \widetilde{f}_w\widetilde{d}_w^{-1}\widetilde{d} \\ \widetilde{g}^{\mathrm{II}} \\ \widetilde{h}^I \end{bmatrix} \qquad (4.89)$$

with

$$\begin{aligned} \widetilde{g}^{\mathrm{II}} &= -(g^I_{,p} - g^I_{,s}c^{-1}_{,s}c_{,p})Zv - g^I_{,\mathfrak{u}^1}\dot{\mathfrak{u}}^1 + g^I_{,s}c^{-1}_{,s}c_{,u}\dot{u},\\ \widetilde{h}^I &= -(\breve{h}_{,p} - \breve{h}_{,s}c^{-1}_{,s}c_{,p})Zv - (\breve{h}_{,u} - \breve{h}_{,s}c^{-1}_{,s}c_{,u})\dot{u}. \end{aligned}$$

Note that the matrix and the right-hand side in (4.89) only depend on $(p, v, r, s, \mathfrak{u}^2)$. Furthermore, the matrix in (4.89) is of size $n_v + n_\lambda + n_\mu \times n_v + n_\lambda + n_\mu$ and has rank $n_v + r_G + r_H$, see Assumption (4.48c). There exist orthogonal matrices $Q_G = \begin{bmatrix} Q_{G1} & Q_{G2} \end{bmatrix} \in \mathbb{R}^{n_\lambda, n_\lambda}$ and $Q_H = \begin{bmatrix} Q_{H1} & Q_{H2} \end{bmatrix} \in \mathbb{R}^{n_\mu, n_\mu}$ such that $G_\lambda \begin{bmatrix} Q_{G1} & Q_{G2} \end{bmatrix} = \begin{bmatrix} \widetilde{G}_\lambda & 0 \end{bmatrix}$ and $H_\mu \begin{bmatrix} Q_{H1} & Q_{H2} \end{bmatrix} = \begin{bmatrix} \widetilde{H}_\mu & 0 \end{bmatrix}$ with $\operatorname{rank}(\widetilde{G}_\lambda) = r_G$ and $\operatorname{rank}(\widetilde{H}_\mu) = r_H$, respectively. With $\lambda = Q_{G1}\widetilde{\lambda}_1 + Q_{G2}\widetilde{\lambda}_2$ and $\mu = Q_{H1}\widetilde{\mu}_1 + Q_{H2}\widetilde{\mu}_2$, (4.89) is equivalent to

$$\begin{bmatrix} M & \widetilde{G}_\lambda & \widetilde{H}_\mu \\ GZ & 0 & 0 \\ HZ & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{v} \\ \widetilde{\lambda}_1 \\ \widetilde{\mu}_1 \end{bmatrix} = \begin{bmatrix} \widetilde{f} - \widetilde{f}_w \widetilde{d}_w^{-1} \widetilde{d} \\ \widetilde{g}^{II} \\ \widetilde{h}^I \end{bmatrix}, \tag{4.90}$$

where the matrix in (4.90) has full (column) rank. According to the General Implicit Function Theorem 2.3.2 we get the uniqueness of $\widetilde{\lambda}_1$ and $\widetilde{\mu}_1$ and, in particular, we get the uniqueness of $\dot{v}$ as solution of (4.90), while $\widetilde{\lambda}_2$ and $\widetilde{\mu}_2$ can be freely chosen. Furthermore, with respect to the forces resulting from the Lagrange multipliers it follows that

$$G_\lambda \lambda + H_\mu \mu = G_\lambda Q_{G1} \widetilde{\lambda}_1 + H_\mu Q_{H1} \widetilde{\mu}_1,$$

which is uniquely determined because of the uniqueness of $\widetilde{\lambda}_1$ and $\widetilde{\mu}_1$. The uniqueness of $\dot{r}$ follows directly from the assumption that the associated right-hand side b is a constant function for all

$$\begin{aligned} (w, \lambda, \mu) \in \{ \mathbb{R}^{n_w} \times \mathbb{R}^{n_\lambda} \times \mathbb{R}^{n_\mu} \quad : \quad & 0 = d(p, v, r, w, s, \lambda, \mu, u), \\ & 0 = g^{II}(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^2), \\ & 0 = h^I(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^1) \}, \end{aligned}$$

i.e., it is invariant under the nonuniqueness of the Lagrange multipliers λ and μ and of the auxiliary variable w. □

Remark 4.2.31 a) If d does not depend on λ or μ, then the auxiliary variables w are also uniquely determined.
b) If f and d do not depend on λ or μ, the matrix G_λ corresponds to $Z^T G^T$ and the matrix H_μ corresponds to $Z^T H^T$ and we get the uniqueness of the constraint forces given by $Z^T G^T \lambda + Z^T H^T \mu$. □

In the following theorems we will discuss the existence and uniqueness of the solution of the equations of motion of modeling level 3 (4.43) as well as of modeling level 4 (4.43).

Theorem 4.2.32 (Existence and partial uniqueness of the solution) *Let the equations of motion of modeling level 4 (4.43) satisfy the Assumption 4.2.4 with noncontradictory constraints (4.43f) and (4.43g) having constant rank on $\mathbb{M}$ and let the initial values (4.43h) be consistent. Furthermore, let the functions $f(p, v, r, w, s, \lambda, \mu, u)$ and $d(p, v, r, w, s, \lambda, \mu, u)$ be of the form (4.86) and (4.87), respectively, i.e., linear in w, λ, and μ, and let the functions $\widetilde{f}$, $\widetilde{f}_w$, $\widetilde{d}$, $\widetilde{d}_w^{-1}$, b, Z, M^{-1}, and $G_\lambda \lambda + H_\mu \mu$ be bounded and continuous on $\mathbb{M}$ and furthermore, Lipschitz continuous on $\mathbb{M}$ with respect to (p, v, r). Suppose that for given (p, v, r, s, u) satisfying (4.43e) the function $(w, \lambda, \mu) \mapsto b(p, v, r, w, s, \lambda, \mu, u)$ is constant in $(w, \lambda, \mu) \in \{ \mathbb{R}^{n_w} \times \mathbb{R}^{n_\lambda} \times \mathbb{R}^{n_\mu} \ : \ 0 = d(p, v, r, w, s, \lambda, \mu, u), \ 0 = g^{II}(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^2), \ 0 = h^I(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^1) \}$.*
Then there exists a unique solution $(p(t), v(t), r(t), s(t))$ of the initial value problem of modeling level 4 (4.43).

Proof: From Lemma 3.5.10 we obtain that the solutions of the initial value problem for the equations of motion of modeling level 4 (4.43) and of the initial value problem for the EoM_0 of modeling level 4 (4.82) are identical for the same initial values. Therefore, the proof reduces to show that the solution $(p(t), v(t), r(t), s(t))$ of (4.82) exists and is unique.
Because of the Assumption (4.55b), from the Implicit Function Theorem 2.3.1 we get the unique solvability of (4.82e) with respect to $s = s(p, u)$ as functions of p and u. Furthermore, we get w from (4.88) and from Lemma 4.2.30 we get the uniqueness of the forces arising from the Lagrange multipliers, i.e., the uniqueness of $G_\lambda\lambda + H_\mu\mu$. Therefore, with Assumption (4.55c) the equations (4.82a)-(4.82c) correspond to the ODE

$$\begin{bmatrix} \dot{p} \\ \dot{v} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} Zv \\ M^{-1}(\widetilde{f} - \widetilde{f}_w\widetilde{d}_w^{-1}\widetilde{d} - G_\lambda\lambda - H_\mu\mu) \\ b \end{bmatrix}. \tag{4.91}$$

From Lemma 4.2.30 we have that the right-hand side of (4.91) is uniquely determined by p, v, r, and u with s satisfying (4.82e), even if the Lagrange multipliers λ, μ or the auxiliary variables w are not unique with $(w, \lambda, \mu) \in \{\mathbb{R}^{n_w} \times \mathbb{R}^{n_\lambda} \times \mathbb{R}^{n_\mu} : 0 = d(p, v, r, w, s, \lambda, \mu, u),\ 0 = g^{II}(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^2),\ 0 = h^I(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^1)\}$. Because of the assumed boundedness, continuity, and Lipschitz continuity of $\widetilde{f}$, $\widetilde{f}_w$, $\widetilde{d}$, $\widetilde{d}_w^{-1}$, b, Z, M^{-1}, and $G_\lambda\lambda + H_\mu\mu$, the conditions of the Theorem of Picard and Lindelöf, see Theorem 3.1.1, are satisfied and the existence of a unique solution for p, v, and r then follows. Furthermore, from Assumption (4.55b), and from the Implicit Function Theorem 2.3.1 we get the unique solution for the contact variables s from (4.82e). □

Theorem 4.2.33 (Existence and uniqueness of the solution of EoM) *Let the equations of motion of modeling level 3 (4.43) be regular, i.e., satisfy Assumptions (4.55a)-(4.55c) and (4.56). Let the initial values (4.43h) be consistent. Furthermore, let the functions $\widetilde{f}$ and $\widetilde{b}$ be defined by*

$$\begin{aligned} \widetilde{f}(p, v, r, \mathfrak{u}^2) &= f(p, v, r, \widetilde{w}, \widetilde{s}, \widetilde{\lambda}, \widetilde{\mu}, u) - Z^T(p)G^T(p, \widetilde{s}, u)\widetilde{\lambda} - Z^T(p)H^T(p, \widetilde{s}, u)\widetilde{\mu}, \\ \widetilde{b}(p, v, r, \mathfrak{u}^2) &= b(p, v, r, \widetilde{w}, \widetilde{s}, \widetilde{\lambda}, \widetilde{\mu}, u), \end{aligned}$$

where $\widetilde{w} = w(p, v, r, \mathfrak{u}^2)$, $\widetilde{s} = s(p, u)$, $\widetilde{\lambda} = \lambda(p, v, r, \mathfrak{u}^2)$, $\widetilde{\mu} = \mu(p, v, r, \mathfrak{u}^2)$ are determined from (4.82d)-(4.82g). Assume further that $\widetilde{f}$ and $\widetilde{b}$ as well as M^{-1} and Z are continuous and bounded on $\mathbb{M}$ and Lipschitz continuous with respect to (p, v, r) on $\mathbb{M}$.
Then there exists a unique solution $(p(t), v(t), r(t), w(t), s(t), \lambda(t), \mu(t))$ of the initial value problem of modeling level 3 (4.43).

Proof: From Lemma 3.5.10 we get that the solutions of the initial value problem for the equations of motion of modeling level 3 (4.43) and of the initial value problem for the EoM_0 of modeling level 3 (4.82) for the same initial values are identical. Therefore, the proof reduces to show that the solution of (4.82) exists and is unique. Because of the Assumptions (4.55a), (4.55b), and (4.56) from the Implicit Function Theorem 2.3.1 we get the unique solvability of (4.82d)-(4.82g) with respect to w, s, λ, and μ as functions of p, v, r, and $\mathfrak{u}^2$. Therefore, with Assumption (4.55c) the equations (4.82a)-(4.82c) correspond to the ODE

$$\begin{bmatrix} \dot{p} \\ \dot{v} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} Z(p)v \\ M^{-1}(p, u)\widetilde{f}(p, v, r, \mathfrak{u}^2) \\ \widetilde{b}(p, v, r, \mathfrak{u}^2) \end{bmatrix}.$$

Because of the assumed boundedness, continuity, and Lipschitz continuity of $\widetilde{f}$, $\widetilde{b}$, Z, and M^{-1} the conditions of the Theorem of Picard and Lindelöf, see Theorem 3.1.1, are satisfied and the existence of a unique solution for p, v, and r then follows. □

4.3 Linear equations of motion and linearization

In this section we will consider linear equations of motion and linearized equations of motion of modeling level 4 (4.43) in function space along a reference trajectory $\bar{p}(t)$, $\bar{v}(t)$, $\bar{r}(t)$, $\bar{w}(t)$, $\bar{s}(t)$, $\bar{\lambda}(t)$, and $\bar{\mu}(t)$, see also Section 3.3. The reference trajectory does not have to be a solution of the equations of motion, see Remarks 3.3.1 and 3.3.3. The linearization of the equations of motion of modeling level 2 (4.41) is discussed in detail in [9]. With the reference trajectory $\bar{x} = \begin{bmatrix} \bar{p}^T & \bar{v}^T & \bar{r}^T & \bar{w}^T & \bar{s}^T & \bar{\lambda}^T & \bar{\mu}^T \end{bmatrix}^T$ we get

$$\begin{array}{lllllllllll} p &=& \bar{p}+\hat{p}, & v &=& \bar{v}+\hat{v}, & r &=& \bar{r}+\hat{r}, & w &=& \bar{w}+\hat{w}, \\ s &=& \bar{s}+\hat{s}, & \lambda &=& \bar{\lambda}+\hat{\lambda}, & \mu &=& \bar{\mu}+\hat{\mu}. \end{array} \tag{4.92}$$

Following the considerations of Section 3.3 we get the linearization of the equations of motion of modeling level 4 (4.43) in the form of (3.3) with (3.17) as

$$E(t)\dot{x} = A(t)x + k(t), \tag{4.93}$$

where

$$E(t) = \begin{bmatrix} I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \bar{M} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, k(t) = \begin{bmatrix} -\dot{\bar{p}} + \bar{Z}\bar{v} \\ -\bar{M}\dot{\bar{v}} + \bar{f} - \bar{Z}^T\bar{G}^T\bar{\lambda} - \bar{Z}^T\bar{H}^T\bar{\mu} \\ -\dot{\bar{r}} + \bar{b} \\ \bar{d} \\ \bar{c} \\ \bar{H}\bar{Z}\bar{v} + \bar{h} \\ \bar{g} \end{bmatrix},$$

$$A(t) = \begin{bmatrix} (\bar{Z}\bar{v})_{,p} & \bar{Z} & 0 & 0 & 0 & 0 & 0 \\ \bar{A}_{pp} & \bar{f}_{,v} & \bar{f}_{,r} & \bar{f}_{,w} & \bar{A}_{ps} & \bar{A}_{p\lambda} & \bar{A}_{p\mu} \\ \bar{b}_{,p} & \bar{b}_{,v} & \bar{b}_{,r} & \bar{b}_{,w} & \bar{b}_{,s} & \bar{b}_{,\lambda} & \bar{b}_{,\mu} \\ \bar{d}_{,p} & \bar{d}_{,v} & \bar{d}_{,r} & \bar{d}_{,w} & \bar{d}_{,s} & \bar{d}_{,\lambda} & \bar{d}_{,\mu} \\ \bar{c}_{,p} & 0 & 0 & 0 & \bar{c}_{,s} & 0 & 0 \\ \bar{A}_{\mu p} & \bar{H}\bar{Z} & 0 & 0 & \bar{A}_{\mu s} & 0 & 0 \\ \bar{g}_{,p} & 0 & 0 & 0 & \bar{g}_{,s} & 0 & 0 \end{bmatrix}, x(t) = \begin{bmatrix} \hat{p} \\ \hat{v} \\ \hat{r} \\ \hat{w} \\ \hat{s} \\ \hat{\lambda} \\ \hat{\mu} \end{bmatrix},$$

with

$$\begin{array}{lll} \bar{A}_{pp} &=& -(\bar{M}\dot{\bar{v}})_{,p} + \bar{f}_{,p} - (\bar{Z}^T\bar{G}^T\bar{\lambda})_{,p} - (\bar{Z}^T\bar{H}^T\bar{\mu})_{,p}, \\ \bar{A}_{ps} &=& \bar{f}_{,s} - \bar{Z}^T(\bar{G}^T\bar{\lambda})_{,s} - (\bar{Z}^T\bar{H}^T\bar{\mu})_{,s}, \\ \bar{A}_{p\lambda} &=& \bar{f}_{,\lambda} - \bar{Z}^T\bar{G}^T, \end{array} \qquad \begin{array}{lll} \bar{A}_{p\mu} &=& \bar{f}_{,\mu} - \bar{Z}^T\bar{H}^T, \\ \bar{A}_{\mu p} &=& (\bar{H}\bar{Z}\bar{v} + \bar{h})_{,p}, \\ \bar{A}_{\mu s} &=& (\bar{H}\bar{Z}\bar{v} + \bar{h})_{,s}. \end{array}$$

The entries of the matrix functions are evaluated at the reference functions $\bar{p}(t)$, $\bar{v}(t)$, $\bar{r}(t)$, $\bar{w}(t)$, $\bar{s}(t)$, $\bar{\lambda}(t)$, and $\bar{\mu}(t)$, respectively. This is denoted by the bar on top of the matrix functions. Recall that the control u is assumed to be given, it remains as a nonlinear function in the matrix functions E, A and the vector function k, which therefore, only depend on t.

Example 4.3.1 The mathematical pendulum: In Example 4.1.12 we derived the equations of motion (4.20) describing the dynamical behavior of the mathematical pendulum. The linearization is given by

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 \\ 0 & 0 & 0 & m & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\hat{p}}_1 \\ \dot{\hat{p}}_2 \\ \dot{\hat{v}}_1 \\ \dot{\hat{v}}_2 \\ \dot{\hat{\lambda}}_1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ -2\bar{\lambda}_1 & 0 & 0 & 0 & -2\bar{p}_1 \\ 0 & -2\bar{\lambda}_1 & 0 & 0 & -2\bar{p}_2 \\ 2\bar{p}_1 & 2\bar{p}_2 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \hat{p}_1 \\ \hat{p}_2 \\ \hat{v}_1 \\ \hat{v}_2 \\ \hat{\lambda}_1 \end{bmatrix} + \begin{bmatrix} -\dot{\bar{p}}_1 + \bar{v}_1 \\ -\dot{\bar{p}}_2 + \bar{v}_2 \\ -m\dot{\bar{v}}_1 - 2\bar{p}_1\bar{\lambda} \\ -m\dot{\bar{v}}_2 - mg - 2\bar{p}_2\bar{\lambda} \\ \bar{p}_1^2 + \bar{p}_2^2 - L^2 \end{bmatrix}.$$

Note that by choosing a reference trajectory with $\begin{bmatrix} \bar{p}_1 & \bar{p}_2 \end{bmatrix}^T$ going through zero, we get a jump of the rank of the constraint matrix G from 1 to 0, which does not occur for any solution satisfying the constraint (4.20c).
Furthermore, if $\begin{bmatrix} \bar{p}_1 & \bar{p}_2 \end{bmatrix}^T = 0$, then the last column of the matrix $A(t)$ becomes zero. Therefore, the Lagrange multiplier is not uniquely defined. See Remark 3.3.3. The d-index as well as the s-index are not defined if $\bar{p}(t)$ goes through zero. □

4.4 Solution submanifold drift and drift stability

In Section 3.2 and, in particular, in Section 3.5 it was discussed that DAEs of higher index contain hidden constraints. As mentioned above, these are algebraic constraints which do not appear explicitly in the original form of the system. Regarding quasi-linear DAEs, in Procedure 3.5.11 it is shown that by differentiating the differential-algebraic equations l times with respect to t, with $l = 1, ..., \nu_c$, and by applying algebraic transformations one can determine the hidden constraints of level l. In the case of equations of motion of modeling level 4 we have found in Lemma 4.2.22 that they have maximal constraint level $\nu_c = 2$.
Let us consider the equations of motion of modeling level 4 (4.43) satisfying Assumption 4.2.4 with noncontradictory constraints (4.43f) and (4.43g).

Definition 4.4.1 (Drift function) *For the equations of motion of modeling level 4 (4.43), the* holonomic drift function $\gamma(t)$ *and the* nonholonomic drift function $\eta(t)$ *are defined as the residual depending on t of the holonomic constraints on position level (4.43g) and of the nonholonomic constraints on velocity level (4.43f), respectively, i.e.,*

$$\gamma(t) = g(p(t), s(t), u(t))$$

and

$$\eta(t) = H(p(t), s(t), u(t))Z(p(t))v(t) + h(p(t), s(t), u(t))$$

along a solution $x(t) = \begin{bmatrix} p^T(t) & v^T(t) & r^T(t) & w^T(t) & s^T(t) & \lambda^T(t) & \mu^T(t) \end{bmatrix}^T$ *and with given $u(t)$.*

In particular, from Definition 4.4.1 it follows that

$$\begin{aligned}
\gamma(t) &= g(p(t), s(t), u(t)), && \text{(4.94a)}\\
\dot{\gamma}(t) &= g^{I}(p(t), v(t), s(t), \mathfrak{u}^1(t)), && \text{(4.94b)}\\
\ddot{\gamma}(t) &= g^{II}(p(t), v(t), r(t), w(t), s(t), \lambda(t), \mu(t), \mathfrak{u}^2(t)), && \text{(4.94c)}\\
\dddot{\gamma}(t) &= g^{III}(p(t), v(t), r(t), w(t), s(t), \lambda(t), \mu(t), \dot{\lambda}(t), \dot{\mu}(t), \mathfrak{u}^3(t)), && \text{(4.94d)}\\
\eta(t) &= \breve{h}(p(t), v(t), s(t), u(t)), && \text{(4.94e)}\\
\dot{\eta}(t) &= h^{I}(p(t), v(t), r(t), w(t), s(t), \lambda(t), \mu(t), \mathfrak{u}^1(t)), && \text{(4.94f)}\\
\ddot{\eta}(t) &= h^{II}(p(t), v(t), r(t), w(t), s(t), \lambda(t), \mu(t), \dot{\lambda}(t), \dot{\mu}(t), \mathfrak{u}^2(t)), && \text{(4.94g)}
\end{aligned}$$

and with respect to the initial values we get

$$\begin{aligned}
\gamma(t_0) &= g(p_0, s_0, u_0), && \text{(4.95a)}\\
\dot{\gamma}(t_0) &= g^{I}(p_0, v_0, s_0, \mathfrak{u}_0^1), && \text{(4.95b)}\\
\ddot{\gamma}(t_0) &= g^{II}(p_0, v_0, r_0, w_0, s_0, \lambda_0, \mu_0, \mathfrak{u}_0^2), && \text{(4.95c)}\\
\eta(t_0) &= \breve{h}(p_0, v_0, s_0, u_0), && \text{(4.95d)}\\
\dot{\eta}(t_0) &= h^{I}(p_0, v_0, r_0, w_0, s_0, \lambda_0, \mu_0, \mathfrak{u}_0^1). && \text{(4.95e)}
\end{aligned}$$

Let us define the *drift differential equations* as follows.

Definition 4.4.2 (Drift differential equation) *The* holonomic drift differential equation *corresponding to the equations of motion of modeling level 4 (4.43) is defined by*

$$c_3\dddot{\gamma}(t) + c_2\ddot{\gamma}(t) + c_1\dot{\gamma}(t) + c_0\gamma(t) = \epsilon(t),$$

where $c_i \in \mathbb{C}$ *for* $i = 0, ..., 3$ *and the* nonholonomic drift differential equation *corresponding to the equations of motion of modeling level 4 (4.43) is defined by*

$$d_2\ddot{\eta}(t) + d_1\dot{\eta}(t) + d_0\eta(t) = \delta(t),$$

where $d_i \in \mathbb{C}$ *for* $i = 0, ..., 2$ *with (4.94).*

In the following, we consider different formulations of the equations of motion and analyze when the holonomic and nonholonomic constraints on position, velocity, and acceleration level of the original equations of motion, which describe the solution manifold (4.65), are satisfied, see also [170].

Let us start the considerations with the underlying differential equation (4.85). We get the drift differential equations which correspond to (4.52e) and (4.54e) in the form $\dddot{\gamma}(t) = 0$ and $\ddot{\eta}(t) = 0$, respectively. Since in the numerical integration the solutions are influenced by round-off errors or stopping errors denoted by $\epsilon(t)$ and $\delta(t)$ when applying an iterative method to solve the nonlinear equations, perturbations influence the equations of the underlying differential equations. Assume that the errors $\epsilon(t)$ and $\delta(t)$ do not vanish and there exist Taylor expansions

$$\epsilon(t) = \epsilon_0 + \epsilon_1\Delta t + \sum_{i=2}^{\infty}\frac{1}{i!}\epsilon_i\Delta t^i \quad \text{and} \quad \delta(t) = \delta_0 + \delta_1\Delta t + \sum_{i=2}^{\infty}\frac{1}{i!}\delta_i\Delta t^i$$

with $\Delta t = (t - t_0)$. This leads to perturbations of (4.52e) and (4.54e) in the form

$$\dddot{\gamma}(t) = \epsilon_0 + \epsilon_1\Delta t + \sum_{i=2}^{\infty}\frac{1}{i!}\epsilon_i\Delta t^i \quad \text{and} \quad \ddot{\eta}(t) = \delta_0 + \delta_1\Delta t + \sum_{i=2}^{\infty}\frac{1}{i!}\delta_i\Delta t^i. \qquad \text{(4.96)}$$

Then we get the solution of the perturbed drift differential equations (4.96) as

$$\begin{aligned}\gamma(t) &= \gamma(t_0) + \dot{\gamma}(t_0)\Delta t + \frac{1}{2}\ddot{\gamma}(t_0)\Delta t^2 + \frac{1}{6}\epsilon_0\Delta t^3 + \frac{1}{24}\epsilon_1\Delta t^4 + ...,\\ \eta(t) &= \eta(t_0) + \dot{\eta}(t_0)\Delta t + \frac{1}{2}\delta_0\Delta t^2 + \frac{1}{6}\delta_1\Delta t^3 + ...\end{aligned}$$

with (4.95) and $\Delta t = t - t_0$. Since $\gamma(t)$ and $\eta(t)$ represent the residuals of the right-hand sides $g(p(t), s(t), u(t))$ and $\breve{h}(p(t), v(t), s(t), u(t))$ of the holonomic constraints on position level and of the nonholonomic constraints on velocity level, respectively, the best what we can expect is that the residual of the holonomic constraints on position level is of order $\mathcal{O}(t^3)$ (if not worse) and of the nonholonomic constraints on velocity level is of order $\mathcal{O}(t^2)$ (if not worse) for $t \to \infty$. This behavior of the numerical solution is known as drift-off phenomenon (see also Section 3.4.1) and describes the behavior of the numerical solution which drifts away from the solution manifold defined be the holonomic and nonholonomic constraints on position, velocity, and acceleration level.

Remark 4.4.3 a) The underlying differential equations (4.85) have a solution for every set of initial values p_0, v_0, r_0, w_0, s_0, λ_0, and μ_0. Besides numerical aspects of the solution of ODEs, no problems arise from initial values or computed solutions at intermediate steps.
b) If the initial values are consistent with the equations of motion (4.43), see Lemma 4.2.26, then the analytical solution satisfies all constraints and, consequently, it lies on the solution manifold.
c) If the initial values are not consistent with the equations of motion (4.43), then even the analytical solution does not satisfy the constraints and therefore, it does not lie on the solution manifold. In particular, the analytical solution is deviating from the position manifold $\mathbb{M}_p$, such that the residual of the constraints on position level has quadratic behavior in t. Likewise, the analytical solution is deviating from the velocity manifold $\mathbb{M}_v$ with a residual of constraints on velocity level behaving linearly in t. The residual of the acceleration constraints is constant. □

If one uses the s-index-0 formulation EoM_0 (4.82) of the equations of motion as a basis for the numerical integration, we get the perturbed drift differential equations

$$\ddot{\gamma}(t) = \epsilon_0 + \epsilon_1\Delta t \ + \sum_{i=2}^{\infty}\frac{1}{i!}\epsilon_i\Delta t^i \quad \text{and} \quad \dot{\eta}(t) = \delta_0 + \delta_1\Delta t + \sum_{i=2}^{\infty}\frac{1}{i!}\delta_i\Delta t^i. \tag{4.97}$$

Therefore, we get the solutions as

$$\begin{aligned}\gamma(t) &= \gamma(t_0) + \dot{\gamma}(t_0)\Delta t + \frac{1}{2}\epsilon_0\Delta t^2 + \frac{1}{6}\epsilon_1\Delta t^3 + ...,\\ \eta(t) &= \eta(t_0) + \delta_0\Delta t + \frac{1}{2}\delta_0\Delta t^2 + ...\end{aligned}$$

with (4.95) and $\Delta t = t - t_0$. Therefore, the best what we can expect is that the residual of the holonomic constraints on position level is of order $\mathcal{O}(t^2)$ (if not worse) and of the nonholonomic constraints on velocity level is of order $\mathcal{O}(t^1)$ (if not worse) for $t \to \infty$.

Remark 4.4.4 a) If the initial values w_0, s_0, λ_0, and μ_0 are not consistent, then the strangeness-free form of the equations of motion EoM_0 (4.82) has no solution.
b) Furthermore, if the initial values are consistent with the equations of motion (4.43), see Lemma 4.2.26, then the holonomic constraints on acceleration level (4.51a) and the nonholonomic constraints on acceleration level (4.53a) are enforced

by using these constraints explicitly in EoM_0. Therefore, if the initial values are consistent, then the analytical solution according to EoM_0 satisfies all constraints and, therefore, lies on the solution manifold.
c) If the initial values w_0, s_0, λ_0 and μ_0 are consistent and p_0 and v_0 are not consistent with the equations of motion (4.43), then the analytical solution deviates from the position manifold $\mathbb{M}_p$ in a linear way with respect to the residuals and lies with a constant residual near the velocity manifold $\mathbb{M}_v$. The analytical solution itself lies on the acceleration manifold $\mathbb{M}_a$.
d) Since the strangeness-free formulation of the equations of motion has no solution if the initial values w_0, s_0, λ_0, or μ_0 are not consistent, one would expect numerical problems. Actually, if these initial values are inconsistent then no solution exists but it is easy to compute consistent initial values by solving the constraints (4.43d), (4.43e) (4.51a), and (4.53a) for w_0, s_0, λ_0, and μ_0. All necessary constraints are explicitly available in EoM_0 (4.82). Furthermore, in all subsequent integration steps the intermediate numerical solutions x_i at t_i satisfy all constraints. Therefore, the intermediate solutions w_i, s_i, λ_i, and μ_i are consistent in each intermediate step. The consistency of p_i and v_i is not important for numerical aspects. Therefore, neglecting numerical aspects of the solution of ODEs, no numerical problems arise from initial values or computed solutions at intermediate steps. □

In analogy to the considerations above with respect to the s-index-1 formulation EoM_1 (4.81) as a basis for the numerical integration we get the perturbed drift differential equations

$$\dot{\gamma}(t) = \epsilon_0 + \epsilon_1 \Delta t + \sum_{i=2}^{\infty} \frac{1}{i!} \epsilon_i \Delta t^i \quad \text{and} \quad \eta(t) = \delta_0 + \delta_1 \Delta t + \sum_{i=2}^{\infty} \frac{1}{i!} \delta_i \Delta t^i. \tag{4.98}$$

Therefore, we get the solutions as

$$\begin{aligned} \gamma(t) &= \gamma(t_0) + \epsilon_0 \Delta t + \frac{1}{2} \epsilon_1 \Delta t^2 + ..., \\ \eta(t) &= \delta_0 + \delta_1 \Delta t + ... \end{aligned}$$

with (4.95) and $\Delta t = t - t_0$. Since $\gamma(t)$ and $\eta(t)$ represent the residuals of the right-hand sides $g(p(t), s(t), u(t))$ and $\breve{h}(p(t), v(t), s(t), u(t))$ of the holonomic constraints on position level and of the nonholonomic constraints on velocity level, respectively, the best what we can expect is that the residual of the holonomic constraints on position level is of order $\mathcal{O}(t^1)$ (if not worse) and of the nonholonomic constraints on velocity level is of order $\mathcal{O}(t^0)$ (if not worse) for $t \to \infty$.

Remark 4.4.5 a) If the initial values v_0, w_0, s_0, λ_0 or μ_0 are inconsistent then the EoM_1 has no solution.
b) If the initial values p_0 are consistent with the equations of motion (4.43), see Lemma 4.2.26, then according to the analytical solution the holonomic constraints on velocity level (4.50a) and also their first derivative with respect to t, i.e., the holonomic constraints on acceleration level (4.51a), as well as the nonholonomic constraints on velocity level (4.43f) and also their first derivative with respect to t, i.e., the nonholonomic constraints on acceleration level (4.53a), are enforced by using the velocity constraints explicitly in (4.81). With consistent initial values, the analytical solution satisfies all constraints even in the form (4.81) and, Therefore, it lies on the solution manifold $\mathbb{M}$.
c) Since (4.81) is no longer strangeness-free, it contains hidden constraints on level one. These are $0 = h^I$ and $0 = g^{II}$.
d) If the initial values v_0, w_0, s_0, λ_0, and μ_0 are consistent and p_0 is not consistent with the equations of motion (4.43), then it follows that the holonomic constraints

on position level are not satisfied and have a constant residual $g(p(t), s(t), u(t)) = \gamma(t_0) \not\equiv 0$ for all $t \in \mathbb{I}$ with respect to an analytical solution. The constraints on velocity level (4.43f) and (4.50a) and also their first derivative with respect to t, i.e., in particular, the holonomic constraints on acceleration level (4.51a) are enforced by using the constraints on velocity level explicitly in (4.81). Therefore, by using inconsistent initial values p_0 the analytical solution lies near the position manifold $\mathbb{M}_p$ with a constant residual, but the solution lies on the velocity manifold $\mathbb{M}_v$ and on the acceleration manifold $\mathbb{M}_a$.
e) Since the s-index-1 formulation (4.81) of the equations of motion has no solution if the initial values v_0, w_0, s_0, λ_0, and μ_0 are not consistent, one would expect numerical problems. Actually, if these initial values are inconsistent then no solution exists. But it is possible to compute initial values w_0, s_0, and v_0 by solving the constraints (4.81d)-(4.81f). Furthermore, if one provides consistent initial values, after every numerical integration step the consistency of the solution components v_i, w_i, and s_i is guaranteed by the explicit appearance of the constraints on velocity level (4.81d)-(4.81f). But, in general, the solution components λ_i and μ_i are not consistent because of rounding errors. Therefore, in addition to the numerical aspects of the solution of ODEs, in general it is not possible to guarantee a convergence behavior for Lagrange multipliers. The consistency of p_i is not important for the numerical solvability by use of the EoM_1. □

Consider the original equations of motion of modeling level 4 (4.43). Since the holonomic constraints on position level (4.43g) and the nonholonomic constraints on velocity level (4.43f) are satisfied because of their explicit appearance in the original equations of motion, also their first and second derivative with respect to t, i.e., the constraints on velocity level (4.50a) and the constraints on acceleration level (4.51a) and (4.53a) are satisfied by an analytical solution.
However, with respect to a numerical integration we get the perturbed drift differential equations

$$\gamma(t) = \epsilon_0 + \epsilon_1 \Delta t + \sum_{i=2}^{\infty} \frac{1}{i!} \epsilon_i \Delta t^i \quad \text{and} \quad \eta(t) = \delta_0 + \delta_1 \Delta t + \sum_{i=2}^{\infty} \frac{1}{i!} \delta_i \Delta t^i. \tag{4.99}$$

Therefore, we get their solutions as

$$\begin{aligned} \gamma(t) &= \epsilon_0 + \epsilon_1 \Delta t + ..., \\ \eta(t) &= \delta_0 + \delta_1 \Delta t + ... \end{aligned}$$

with (4.95) and $\Delta t = t - t_0$. Since $\gamma(t)$ and $\eta(t)$ represent the residuals of the right-hand sides $g(p(t), s(t), u(t))$ and $\breve{h}(p(t), v(t), s(t), u(t))$ of the holonomic constraints on position level and of the nonholonomic constraints on velocity level, respectively, the best what we can expect is that the residual of the holonomic constraints on position level is of order $\mathcal{O}(t^0)$ (if not worse) and of the nonholonomic constraints on velocity level is of order $\mathcal{O}(t^0)$ (if not worse) for $t \to \infty$.

Remark 4.4.6 a) If all initial values are consistent, then the analytical solution lies on the solution manifold $\mathbb{M}$.
b) If the initial values are inconsistent, then the original equations of motion (4.43) with s-index two (and d-index three, when exist) have no solution.
c) Since the original equations of motion (4.43) have no solution if the initial values are not consistent, numerical problems have to be expected. Actually, if these initial values are inconsistent, then no solution can be found but it is possible to compute consistent initial values p_0, w_0, and s_0 by solving the constraints (4.43d), (4.43e), and (4.43g). Therefore, if one provides consistent initial values after each numerical integration step, the consistency of solution components p_i, w_i, and s_i is enforced

by the explicit appearance of the constraints (4.43d), (4.43e), and (4.43g). On the other hand, the solution components v_i, λ_i, and μ_i in general are not consistent, because of rounding errors. Therefore, in addition to the numerical aspects of the solution of ODEs, it is in general not possible to guarantee a convergence behavior for the velocity components and the Lagrange multipliers. □

In [5] the influence of perturbations on different formulations is discussed. It is demonstrated that the numerical integration of the original equations of motion of s-index two leads to (large) errors in the solution independent of the prescribed tolerance. But otherwise, the numerical integration of the EoM_1 of s-index 1 leads to appropriate errors in the solution depending on the prescribed tolerance. This effect arises from the fact that the solution of higher index DAEs does not only depend continuously on perturbations but, rather in addition on derivatives of the perturbations, too.
Table 4.2 summarizes all these results. Note, that the d-index only exists if the equations of motion of modeling level 4 (4.43) with noncontradictory constraints (4.43f) and (4.43g) satisfy the Assumptions (4.48a), (4.48b), and (4.49). From Table 4.2 it becomes clear that none of the formulations above is ideally suited as a base for numerical integrations. Either the analytical solution deviates from the solution manifold in a cubic way but without possible numerical oscillations when using the underlying differential equations. By using the original equations of motion the solution does not deviate from the solution manifold but the numerical integration may yield numerical oscillations with respect to λ, μ, and v. A trade-off between oscillations and deviation is the use of EoM_0 or EoM_1.
This shows that it is necessary to look for a formulation which does not contain any possible oscillating behavior, i.e, one has to look for a strangeness-free formulation. On the other hand, this formulation must contain all possible information about the solution manifold. Different ways out of this dilemma are regularization techniques are discussed in the following Section 4.6.
The index of a differential-algebraic system and the way how all information about the solution manifold is given have an essential influence on the quality and the success of numerical integration. Here, a small index, if possible d-index 0 or d-index 1, i.e., s-index 0, is preferred, see [69]. Furthermore, there should be no hidden constraints, i.e., all information of the solution manifold should be given in an explicit way in the system which has to be integrated.

Example 4.4.7 The drift-off phenomenon is demonstrated in [82] for the example of the mathematical pendulum, see also Figure 5.6 in Example 5.3.1. Furthermore, the drift-off phenomenon is considered in [167] for the nonlinear truck model, see Example 4.1.14. □

4.5 Two paradigms

As we have seen in Section 4.4, it is essential for the quality of a numerical solution that the constraints which restrict the solution are provided in an explicit way, i.e., the information about the constraints should appear as equations in the equations of motion. We have seen that in the case of hidden constraints contained in the used form of the equations of motion, the numerical solution is influenced by instabilities and in the case of replacing constraints by their derivatives, yielding the s-index-1 formulation of the equations of motion (4.81) and the s-index-0 formulation of the equations of motion (4.82), the numerical solution is influenced by the drift-off effect.
Furthermore, in Section 4.1.4 we have seen that in the case of existing solution invariants which are satisfied by an analytical solution, the numerical solution usually

Table 4.2: Deviation and numerical behavior of different forms of equations of motion

used formulation	**UDE (4.85)**	**EoM$_0$ (4.82)**	**EoM$_1$ (4.81)**	**original EoM (4.43)**
last equations in used form	$0 = d^I$, $0 = c^I$ $0 = h^{I\!I}$, $0 = g^{I\!I\!I}$	$0 = d$, $0 = c$ $0 = h^I$, $0 = g^{I\!I}$	$0 = \underset{\smile}{d}$, $0 = c$ $0 = \underset{\smile}{h}$, $0 = g^I$	$0 = \underset{\smile}{d}$, $0 = c$ $0 = \underset{\smile}{h}$, $0 = g$
hidden constraints	none	none	$0 = g^{I\!I}$, $0 = h^I$	$0 = g^I$, $0 = g^{I\!I}$, $0 = h^I$
s-index	0	0	1	2
d-index (if defined)	0	1	2	3
deviation caused by perturbations	deviation from ... $\mathbb{M}_p$ - cubic in t $\mathbb{M}_v$ - quadratic in t $\mathbb{M}_a$ - linear in t	deviation from ... $\mathbb{M}_p$ - quadratic in t $\mathbb{M}_v$ - linear in t	deviation from ... $\mathbb{M}_p$ - linear in t	no deviation
existence of solution with respect to consistency of initial values	no consistency conditions	w_0, s_0, λ_0, μ_0 must be consistent	v_0, w_0, s_0, λ_0, μ_0 must be consistent	p_0, v_0, w_0, s_0, λ_0, μ_0 must be consistent
dimension of solution manifold	$n = n_p + n_v + n_r + n_w + n_s + n_\lambda + n_\mu$	$n - n_w + n_s - r_G - r_H$	$n - n_w + n_s - 2r_G - 2r_H$	$n - n_w + n_s - 3r_G - 2r_H$

not satisfy these solution invariants, since the numerical algorithm cannot force the numerical solution into invariant manifolds without explicit information describing these invariant manifolds, see Figure 4.9. Therefore, it is essential for the accuracy and the stability of the numerical solution that the corresponding numerical algorithm is provided in an explicit way with all information which is available. We get the following Modeling Paradigm.

Paradigm 4.5.1 (Modeling Paradigm) *The model equations of a dynamical system should explicitly contain all available information about the dynamical system.*

Remark 4.5.2 In the modeling paradigm "explicitly contain" means that the information appears as equations among the model equations.
In particular, for the equations of motion of modeling level 4 (4.43) it follows from the nature of the equations, that the first and the second derivative of the holonomic constraints with respect to t and the first derivative of the nonholonomic constraints should be explicitly provided in the model equations.
Furthermore, information of possibly existing invariant manifolds are of importance and should explicitly appear in the model equations. □

On the other hand, it is clear that explicitly providing all information about a dynamical system that is available, does not make sense if the used numerical algorithm is not able to handle this information. Therefore, a numerical algorithm should be able to respect all possibly available information of a dynamical system which leads to the following Algorithm Paradigm.

Paradigm 4.5.3 (Algorithm Paradigm) *A numerical algorithm for the numerical solution of model equations of dynamical systems should be able to respect all possible information about the dynamical system that is available.*

Remark 4.5.4 Previously there did not exist numerical codes which do respect all information, in particular, the information about invariant manifolds. Some numerical integration methods exist which reflect invariant manifolds in the used discretization scheme, see [80], but nevertheless, round-off errors and truncation errors lead to a drift away from the invariant manifolds. □

4.6 Regularization of the equations of motion

In general, the numerical integration of the equations of motion (4.43) is substantially more difficult and prone to intensive numerical computation than that of ODEs, see [25, 73, 82, 133]. As mentioned above, the index of a DAE provides a measure of the difficulty of solving the treatment of the DAE. A lower index is to be preferred for the numerical simulation. However, simple differentiation of the constraints does lower the index, but it does increase the drift-off effect as shown in Section 4.4.
Several approaches have been introduced in order to stabilize the integration process. For an overview see [25, 51, 82, 164]. In principle, the regularization techniques can be classified into three different categories.
The first class are the *stabilization methods*, e.g., Baumgarte stabilization [19], the use of singularly perturbed problems [82, 97], or lowering the index by differentiation of the constraints [66]. The second class covers the *state space methods*, e.g., [127, 139]. The third class is concerned with *projection methods*, e.g., the Gear-Gupta-Leimkuhler formulation [68], overdetermined or ssf-formulation [60].
In the following we will review some important and widely used regularization techniques and if possible we will extend these techniques to the equations of motion of modeling level 3 and 4 (4.43). Furthermore, in Section 4.6.2.3 we will develop a new

regularization technique for the equations of motion of modeling level 3 and 4 (4.43) based on the general approach for nonlinear DAEs, see Section 3.4.2, which exploits the structure of the equations of motion such that this regularization technique is simple to realize and offers the possibility of combination with a numerical discretization method to create a robust and stable integrator for mechanical systems, see Chapter 5.

4.6.1 Stabilization methods

4.6.1.1 Regularization by differentiation the constraints

The regularization of DAEs via index reduction by use of differentiated constraints instead of the original constraints is already discussed in Section 3.4.1.
The application of this regularization technique to the equations of motion of modeling level 4 leads to the formulations (4.81), (4.82), and (4.85). In particular, we have seen that lowering the index by differentiation creates a more suitable formulation for the numerical integration but increases the drift-off phenomenon as discussed in the previous Section 4.4.
Nevertheless, this technique is widely used in numerical integration methods. An acceptable compromise using one of the formulations (4.43), (4.81), (4.82), or (4.85) as basis for the numerical integration is the s-index-1 formulation (4.81), which is the basis of several numerical integration methods for the equations of motion, e.g., `MEXAX` [118], `HEDOP5` [6], `HEM5` [22, 21], see Section 4.7.

4.6.1.2 Baumgarte stabilization

In [18] , Baumgarte proposed a stabilization technique for the equations of motion of multibody systems of modeling level 0 (4.34), see also [82, 147, 164]. He introduced stabilization terms into the associated s-index-0 formulation of modeling level 0 (4.39), which steers a perturbed nonconsistent solution back to the solution manifold $\mathbb{M}$. The dimension of the obtained system is preserved. The s-index of the Baumgarte-stabilized equations of motion is reduced to s-index 0 (the d-index is reduced from 3 to 1) but the solution manifold associated with the stabilized equations of motion has a larger dimension than the original solution manifold $\mathbb{M}$ and contains the solution manifold $\mathbb{M}$.
In particular, the holonomic constraints (4.34c) are replaced by a linear combination of the holonomic constraints on position level (4.34c), on velocity level (4.35), and on acceleration level (4.36), i.e., the constraints are replaced by

$$0 = g^{II}(p, v, \lambda) + 2\alpha g^{I}(p, v) + \beta^2 g(p).$$

By using $\gamma(t) = g(p(t))$ according to Definition 4.4.2, we get the holonomic drift differential equation

$$0 = \ddot{\gamma}(t) + 2\alpha\dot{\gamma}(t) + \beta^2\gamma(t). \tag{4.100}$$

The coefficients α and β are chosen such that the roots $\lambda_{1,2}$ of the corresponding polynomial

$$0 = \lambda^2 + 2\alpha\lambda + \beta^2$$

have negative real part. In this case, the solution $\gamma(t) \equiv 0$ is a stable solution of (4.100), i.e., solutions with initial residuals $\gamma(t_0) = g(p_0)$ and $\dot{\gamma}(t_0) = g^I(p_0, v_0)$ which are possibly nonzero tend to zero. Using this stabilization technique, the solution manifold (4.37) is attracting.
The main difficulty with the Baumgarte stabilization is the choice of the parameters

α and β. This question is discussed in [14, 76], while in [14] it is shown that there exists no universal choice of the parameters for all problems. Rather, the optimal choice of the parameters α and β depends on the problem, the discretization method, and the discretization step size.
An extension of the Baumgarte stabilization to equations of motion of modeling level 4 (4.43) is possible and has the form

$$\begin{aligned}
\dot{p} &= Z(p)v, && \text{(4.101a)}\\
M(p,u)\dot{v} &= f(p,v,r,w,s,\lambda,\mu,u) && \text{(4.101b)}\\
&\quad -Z^T(p)G^T(p,s,u)\lambda - Z^T(p)H^T(p,s,u)\mu,\\
\dot{r} &= b(p,v,r,w,s,\lambda,\mu,u), && \text{(4.101c)}\\
0 &= d(p,v,r,w,s,\lambda,\mu,u), && \text{(4.101d)}\\
0 &= c(p,s,u), && \text{(4.101e)}\\
0 &= h^I(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^1) + \delta\breve{h}(p,v,s,u), && \text{(4.101f)}\\
0 &= g^{I\!I}(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^2) + \alpha g^I(p,v,s,\mathfrak{u}^1) + \beta g(p,s,u), && \text{(4.101g)}
\end{aligned}$$

with α and β chosen as in the classical Baumgarte stabilization and $\delta > 0$. In the case of $\delta > 0$ the corresponding nonholonomic drift differential equation

$$0 = \dot{\eta}(t) + \delta\eta(t),$$

arising from (4.101f), has $\eta(t) \equiv 0$ as a stable solution and the solution manifold becomes attracting.

4.6.2 Projection methods

4.6.2.1 Gear-Gupta-Leimkuhler formulation

In [68], Gear, Gupta, and Leimkuhler presented a possibility to modify the equations of motion of modeling level 0 in a simple way such that the s-index is lowered from two to one (d-index is lowered from three to two), while the solution manifold is preserved, i.e., no drift-off effects occur. The idea is to add the constraints on velocity level (4.35) to the equations of motion of modeling level 0 (4.34) whose purpose is to ensure that the constraints on velocity level are satisfied. This leads to an overdetermined DAE. This system can be made determined by introducing new Lagrange multipliers ϑ yielding the so-called *Gear-Gupta-Leimkuhler formulation* of the equations of motion of modeling level 0 given by

$$\begin{aligned}
\dot{p} &= v - G^T(p)\vartheta, && \text{(4.102a)}\\
M(p)\dot{v} &= f(p,v,t) - G^T(p)\lambda, && \text{(4.102b)}\\
0 &= g(p), && \text{(4.102c)}\\
0 &= G(p)v. && \text{(4.102d)}
\end{aligned}$$

The dimension of the system is increased by n_λ.
In [68] the relation between the equations of motion of modeling level 0 (4.34) and the Gear-Gupta-Leimkuhler formulation (4.102) is discussed. It is shown, that if (p,v,λ) is a solution of (4.34) then (p,v,λ,ϑ) with $\vartheta = 0$ is a solution of (4.102). Conversely, if (p,v,λ,ϑ) is a solution of (4.102) then it follows that $\vartheta = 0$ and (p,v,λ) is a solution of (4.34).
An extension of this approach to equations of motion of modeling level 3 (4.43) is

possible and has the form

$$\begin{aligned}
\dot{p} &= Z(p)v - G^T(p,s,u)\vartheta, && (4.103a)\\
M(p,u)\dot{v} &= f(p,v,r,w,s,\lambda,\mu,u) && (4.103b)\\
&\quad -Z^T(p)G^T(p,s,u)\lambda - Z^T(p)H^T(p,s,u)\mu,\\
\dot{r} &= b(p,v,r,w,s,\lambda,\mu,u), && (4.103c)\\
0 &= d(p,v,r,w,s,\lambda,\mu,u), && (4.103d)\\
0 &= c(p,s,u), && (4.103e)\\
0 &= H(p,s,u)Z(p)v + h(p,s,u), && (4.103f)\\
0 &= g(p,s,u), && (4.103g)\\
0 &= g^I(p,v,s,\mathfrak{u}^1). && (4.103h)
\end{aligned}$$

Let us call this formulation the *Gear-Gupta-Leimkuhler formulation of modeling level 3*. In the following we will perform the Procedure 3.5.11 concerning the Gear-Gupta-Leimkuhler formulation of modeling level 3.

Procedure 4.6.1 Let the equations of motion of modeling level 3 (4.43) satisfy Assumptions (4.55a)-(4.55c) and (4.56). Following Procedure 3.5.11 we get, according to the Gear-Gupta-Leimkuhler formulation of modeling level 3 (4.103) the DAE

$$E^0\dot{x} = k^0$$

with $x = \begin{bmatrix} p^T & v^T & r^T & w^T & s^T & \lambda^T & \mu^T & \vartheta^T \end{bmatrix}^T$ and

$$E^0 = \begin{bmatrix}
I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & M & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}, \quad k^0 = \begin{bmatrix}
Zv - G^T\vartheta\\
f - Z^TG^T\lambda - Z^TH^T\mu\\
b\\
d\\
c\\
HZv + h\\
g\\
g^I
\end{bmatrix}.$$

This is equivalent to

$$\widetilde{E}^0\dot{x} = \widetilde{k}^0$$

with $\widetilde{E}^0 = E^0$ and $\widetilde{k}^0 = k^0$. Differentiation of the constraints leads to

$$E^1\dot{x} = k^1$$

with

$$E^1 = \begin{bmatrix}
I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & M & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 & 0\\
d_{,p} & d_{,v} & d_{,r} & d_{,w} & d_{,s} & d_{,\lambda} & d_{,\mu} & 0\\
c_{,p} & 0 & 0 & 0 & c_{,s} & 0 & 0 & 0\\
\breve{h}_{,p} & HZ & 0 & 0 & \breve{h}_{,s} & 0 & 0 & 0\\
g_{,p} & 0 & 0 & 0 & g_{,s} & 0 & 0 & 0\\
g^I_{,p} & g^I_{,v} & 0 & 0 & g^I_{,s} & 0 & 0 & 0
\end{bmatrix}, \quad k^1 = \begin{bmatrix}
Zv - G^T\vartheta\\
f - Z^TG^T\lambda - Z^TH^T\mu\\
b\\
-d_{,u}\dot{u}\\
-c_{,u}\dot{u}\\
-\breve{h}_{,u}\dot{u}\\
-g_{,u}\dot{u}\\
-g^I_{,\mathfrak{u}^1}\dot{\mathfrak{u}}^1
\end{bmatrix}.$$

Elimination leads to

$$\widetilde{E}^1\dot{x} = \widetilde{k}^1$$

with

$$\widetilde{E}^1 = \begin{bmatrix} I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & M & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & d_{,w} & d_{,s} & d_{,\lambda} & d_{,\mu} & 0 \\ 0 & 0 & 0 & 0 & c_{,s} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and

$$\widetilde{k}^1 = \begin{bmatrix} Zv - G^T\vartheta \\ f - Z^TG^T\lambda - Z^TH^T\mu \\ b \\ -d_{,u}\dot{u} - d_{,p}(Zv - G^T\vartheta) - d_{,v}M^{-1}(f - Z^TG^T\lambda - Z^TH^T\mu) - d_{,r}b \\ -c_{,u}\dot{u} - c_{,p}(Zv - G^T\vartheta) \\ -h^I + (\breve{h}_{,p} - \breve{h}_{,s}c_{,s}^{-1}c_{,p})G^T\vartheta \\ -g^I + GG^T\vartheta \\ -g^{I\!I} + (g^I_{,p} - g^I_{,s}c_{,s}^{-1}c_{,p})G^T\vartheta \end{bmatrix}.$$

Further differentiation of the algebraic equations leads to

$$E^2\dot{x} = k^2$$

with

$$E^2 = \begin{bmatrix} I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & M & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & d_{,w} & d_{,s} & d_{,\lambda} & d_{,\mu} & 0 \\ 0 & 0 & 0 & 0 & c_{,s} & 0 & 0 & 0 \\ \times & \times & \times & -HZM^{-1}f_{,w} & \times & E^2_{66} & E^2_{67} & (\breve{h}_{,p} - \breve{h}_{,s}c_{,s}^{-1}c_{,p})G^T \\ \times & \times & \times & 0 & \times & 0 & 0 & GG^T \\ \times & \times & \times & -GZM^{-1}f_{,w} & \times & E^2_{86} & E^2_{87} & (g^I_{,p} - g^I_{,s}c_{,s}^{-1}c_{,p})G^T \end{bmatrix},$$

$$\begin{aligned} E^2_{66} &= -HZM^{-1}(f_{,\lambda} - Z^TG^T), & E^2_{67} &= -HZM^{-1}(f_{,\mu} - Z^TH^T), \\ E^2_{86} &= -GZM^{-1}(f_{,\lambda} - Z^TG^T), & E^2_{87} &= -GZM^{-1}(f_{,\mu} - Z^TH^T). \end{aligned}$$

Elimination and row permutation leads to

$$\widetilde{E}^2\dot{x} = \widetilde{k}^2$$

with

$$\widetilde{E}^2 = \begin{bmatrix} I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & M & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & d_{,w} & d_{,s} & d_{,\lambda} & d_{,\mu} & 0 \\ 0 & 0 & 0 & 0 & c_{,s} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \widetilde{E}^2_{66} & \widetilde{E}^2_{67} & (\breve{h}_{,p} - \breve{h}_{,s}c_{,s}^{-1}c_{,p})G^T \\ 0 & 0 & 0 & 0 & 0 & \widetilde{E}^2_{76} & \widetilde{E}^2_{77} & (g^I_{,p} - g^I_{,s}c_{,s}^{-1}c_{,p})G^T \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & GG^T \end{bmatrix}$$

and

$$\begin{bmatrix} \widetilde{E}^2_{66} & \widetilde{E}^2_{67} \\ \widetilde{E}^2_{76} & \widetilde{E}^2_{77} \end{bmatrix} = \begin{bmatrix} HZM^{-1}G_\lambda & HZM^{-1}H_\mu \\ GZM^{-1}G_\lambda & GZM^{-1}H_\mu \end{bmatrix}.$$

From Assumption (4.55a)-(4.55c) and (4.56) and Lemma 4.2.10 we get the nonsingularity of E^2 for all $(p, v, r, w, s, \lambda, \mu, \mathfrak{u}^2) \in \mathbb{M}$. Hence, the algebraic constraints vanish, i.e., $m_2^2 = 0$. In accordance to (3.42) the Procedure 3.5.11 terminates with $\nu = 2$. □

Lemma 4.6.2 *Let the equations of motion of modeling level 3 (4.43) satisfy Assumptions (4.55a)-(4.55c) and (4.56). Then the Gear-Gupta-Leimkuhler formulation (4.103) of modeling level 3 has maximal constraint level $\nu_c = 1$ and d-index $\nu_d = 2$, any solution $(p, v, r, w, s, \lambda, \mu, \vartheta)$ of (4.103) has $\vartheta = 0$ for all $t \in \mathbb{I}$, and $(p, v, r, w, s, \lambda, \mu)$ is a solution of the original equations of motion (4.43) of modeling level 3. Conversely, if $(p, v, r, w, s, \lambda, \mu)$ is a solution of the original equations of motion (4.43) of modeling level 3, then $(p, v, r, w, s, \lambda, \mu, 0)$ is a solution of the Gear-Gupta-Leimkuhler formulation (4.103) of modeling level 3.*

Proof: From Procedure 4.6.1 we get that the Gear-Gupta-Leimkuhler formulation (4.103) has maximal constraint level $\nu_c = 1$. Furthermore, from Lemma 3.5.29 it follows that the Gear-Gupta-Leimkuhler formulation (4.103) has d-index $\nu_d = 2$.
In [68] the equivalence of the standard form of the equations of motion (4.34) of modeling level 0 and its associated Gear-Gupta-Leimkuhler formulation (4.102) is given. The proof for the equivalence of the equations of motion of modeling level 3 (4.43) and its Gear-Gupta-Leimkuhler formulation of modeling level 3 (4.103) is similar.
Obviously, every solution of (4.43) does satisfy the Gear-Gupta-Leimkuhler formulation (4.103) with $\vartheta = 0$. On the other hand, suppose that $(p, v, r, w, s, \lambda, \mu, \vartheta)$ is a solution of (4.103). Differentiating (4.103g) leads to (4.50b). Substituting (4.103a) yields

$$0 = G(Zv - G^T\vartheta) + (g_{,u} - g_{,s}c_{,s}^{-1}c_{,u})\dot{u}. \tag{4.104}$$

Subtracting (4.103h) in the form (4.50c) from (4.104) yields

$$0 = GG^T\vartheta.$$

From Lemma 4.2.10 we have the full rank of G and therefore, the nonsingularity of GG^T and we get $\vartheta = 0$. Hence, the solution of (4.103) also satisfies (4.43). □

Remark 4.6.3 From Lemma 4.6.2 we get that the Gear-Gupta-Leimkuhler formulation (4.103) of modeling level 3 is equivalent to the original equations of motion (4.43) of modeling level 3 in the sense that both solution sets are identical apart from the additional Lagrange multipliers ϑ. □

The advantage of the Gear-Gupta-Leimkuhler formulation (4.103) is the reduced maximal constraint level $\nu_c = 1$, the reduced d-index and s-index, that its solution manifold is equivalent to the solution manifold $\mathbb{M}$, and that in addition to the constraints on position level the constraints on velocity level appear explicitly. The disadvantages are the higher index, i.e., the s-index is still larger than 0, and that the Gear-Gupta-Leimkuhler formulation is not strangeness-free. Slight instabilities may occur and the constraints on acceleration level remain hidden.
An extension of the Gear-Gupta-Leimkuhler formulation (4.102) of the equations of motion of modeling level 0 is proposed by Führer and Leimkuhler in [60] which is still of s-index one, i.e., not strangeness-free, but involves all constraints (4.34c), (4.35), as well as (4.36) in an explicit way. Therefore, no drift-off effects for the state variables of the multibody system but slight instabilities for the additionally introduced variables occur. For more details see [60, 164].

4.6.2.2 Overdetermined formulation

While the regularization techniques that are discussed in the previous sections mostly lead to an equivalent regularized form of the equations of motion of the same number of unknowns and equations, in [52, 60] an approach based on the equations of motion of modeling level 0 (4.34) is proposed which adds all hidden constraints to the equations of motion. This approach leads to an overdetermined system consisting of (4.34), (4.35), and (4.36). Applying this approach to the equations of motion of modeling level 3 or 4 (4.43) we get the overdetermined DAE consisting of (4.43), (4.50), (4.51), and (4.53)

$$\begin{aligned}
\dot{p} &= Z(p)v, && \text{(4.105a)}\\
M(p,u)\dot{v} &= f(p,v,r,w,s,\lambda,\mu,u) && \text{(4.105b)}\\
&\quad -Z^T(p)G^T(p,s,u)\lambda - Z^T(p)H^T(p,s,u)\mu,\\
\dot{r} &= b(p,v,r,w,s,\lambda,\mu,u), && \text{(4.105c)}\\
0 &= d(p,v,r,w,s,\lambda,\mu,u), && \text{(4.105d)}\\
0 &= c(p,s,u), && \text{(4.105e)}\\
0 &= H(p,s,u)Z(p)v + h(p,s,u), && \text{(4.105f)}\\
0 &= g(p,s,u), && \text{(4.105g)}\\
0 &= h^I(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^1), && \text{(4.105h)}\\
0 &= g^I(p,v,s,\mathfrak{u}^1), && \text{(4.105i)}\\
0 &= g^{II}(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^2). && \text{(4.105j)}
\end{aligned}$$

By application the Procedure 3.5.11 to the overdetermined formulation (4.105) we get the maximal constraint level $\nu_c = 0$. Furthermore, from Hypothesis 3.2.7 it follows that the DAE (4.105) has s-index $\nu_s = 1$. In particular, this means that the overdetermined formulation (4.105) is not strangeness free but all necessary information is contained in the system in an explicit way and, therefore, there exist no hidden constraints. However, the d-index is not defined for (4.105), because of its overdeterminedness.
After discretization of the DAE (4.105), an overdetermined set of $l(n_p + n_v + n_r + n_w + n_s + 3n_\lambda + 2n_\mu)$ nonlinear equations in $l(n_p + n_v + n_r + n_w + n_s + n_\lambda + n_\mu)$ unknowns has to be solved several times at each integration step. The number $l \in \mathbb{N}$ is specified by the integration method, e.g., $l = 1$ for BDF methods or l denotes the number of stages for Runge-Kutta-Methods. Because of truncation errors, the discretized equations become contradictory and can only be solved in a generalized sense. The solution of the resulting overdetermined algebraic system (discretized DAE) as described in [60] can be interpreted as a numerical projection onto the constraint manifold. This approach is implemented in the code `ODASSL` (thanks to Claus Führer for providing this algorithm for usage). For more details see [52, 60].

4.6.2.3 Projected-strangeness-free formulation

For equations of motion of modeling level 0 (4.34), a regularization based on the strangeness concept, see Section 3.4.2, is investigated in [170]. Furthermore, in [10] the regularization of linear equations of motion of modeling level 2 (4.41) is discussed. In this section we will extend and generalize the results obtained in [10, 170] to the equations of motion of modeling level 4 (4.43) with noncontradictory constraints (4.43f) and (4.43g), i.e., they satisfy Assumptions 4.2.4, via index reduction according to the results of Section 3.5.3.
As shown in Section 4.4 and postulated in the two paradigms in Section 4.5, it is important to preserve and to determine all information about the solution manifold.

Therefore, in the regularized form, in addition to the explicitly given constraints (4.43d)-(4.43g), all hidden constraints, i.e., the holonomic constraints on velocity level (4.50a), the holonomic constraints on acceleration level (4.51a), and the non-holonomic constraints on acceleration level (4.53a) should appear in an explicit way in the regularized equations of motion.
In Section 4.2 we determined the maximal constraint level of the equations of motion as $\nu_c = 2$, see Lemma 4.2.22, and the complete minimal reduced derivative in the form (4.80), see Lemma 4.2.25. Because of the semi-implicit structure of the equations of motion, we may use the complete minimal reduced derivative array in the form (3.67) instead of (3.66). From (3.74) we get for $\nu_c = 2$ the quantities

$$\begin{aligned} r^{\nu_c} &= n_p + n_v + n_r + n_w + n_s + r_G + r_H, \\ a^{\nu_c} &= n_w + n_s + r_G + r_H + r_G + r_H + r_G, \\ d^{\nu_c} &= (n_p - r_G) + (n_v - r_G - r_H) + n_r = n_{f_p} + n_{f_v} + n_r. \end{aligned}$$

Therefore, from Definition 3.5.44 with respect to (3.67) we get that a dynamic selector S_D has to have the size $n_{f_p} + n_{f_v} + n_r \times n_p + n_v + n_r$ and with $S_D = \begin{bmatrix} S_{D1} & S_{D2} & S_{D3} \end{bmatrix}$ it has to satisfy the condition that

$$\begin{aligned} &\operatorname{rank}\left(\begin{bmatrix} S_{D1} & S_{D2}M & S_{D3} & 0 & 0 & 0 & 0 \\ d_{,p} & d_{,v} & d_{,r} & d_{,w} & d_{,s} & d_{,\lambda} & d_{,\mu} \\ c_{,p} & 0 & 0 & 0 & c_{,s} & 0 & 0 \\ \breve{h}_{,p} & HZ & 0 & 0 & \breve{h}_{,s} & 0 & 0 \\ g_{,p} & 0 & 0 & 0 & g_{,s} & 0 & 0 \\ h^I_{,p} & h^I_{,v} & h^I_{,r} & h^I_{,w} & h^I_{,s} & h^I_{,\lambda} & h^I_{,\mu} \\ g^I_{,p} & GZ & 0 & 0 & g^I_{,s} & 0 & 0 \\ g^{I\!I}_{,p} & g^{I\!I}_{,v} & g^{I\!I}_{,r} & g^{I\!I}_{,w} & g^{I\!I}_{,s} & g^{I\!I}_{,\lambda} & g^{I\!I}_{,\mu} \end{bmatrix} \right) \\ &= n_p + n_v + n_r + n_w + n_s + r_G + r_H \qquad (= n - (n_\lambda - r_G) - (n_\mu - r_H)) \end{aligned} \tag{4.106}$$

for all $(x, u) \in \mathbb{M}$. Using block Gauß elimination, we get that the rank condition of the matrix in (4.106) is equivalent to

$$\operatorname{rank}\left(\begin{bmatrix} S_{D1} & S_{D2}M & S_{D3} \\ \breve{h}_{,p} - \breve{h}_{,s} c_{,s}^{-1} c_{,p} & HZ & 0 \\ G & 0 & 0 \\ g^I_{,p} - g^1_{,s} c_{,s}^{-1} c_{,p} & GZ & 0 \end{bmatrix} \right) = n_p + n_v + n_r \tag{4.107}$$

and

$$\operatorname{rank}\left(\begin{bmatrix} d_{,w} & 0 & d_{,\lambda} & d_{,\mu} \\ 0 & c_{,s} & 0 & 0 \\ 0 & 0 & h^I_{,\lambda} - h^I_{,w} d_{,w}^{-1} d_{,\lambda} & h^I_{,\mu} - h^I_{,w} d_{,w}^{-1} d_{,\mu} \\ 0 & 0 & g^{I\!I}_{,\lambda} - g^{I\!I}_{,w} d_{,w}^{-1} d_{,\lambda} & g^{I\!I}_{,\mu} - g^{I\!I}_{,w} d_{,w}^{-1} d_{,\mu} \end{bmatrix} \right) = n_w + n_s + r_G + r_H. \tag{4.108}$$

While the matrix in (4.108) is a square matrix of size $n_w + n_s + n_\lambda + n_\mu \times n_w + n_s + n_\lambda + n_\mu$ and the satisfaction of its rank condition (4.108) follows directly from Assumption 4.2.4, the matrix in (4.107) is rectangular of size $(n_{f_p} + n_{f_v} + n_r) + n_\mu + n_\lambda + n_\lambda \times n_p + n_v + n_r$ with $n_{f_p} = n_p - r_G$ and $n_{f_v} = n_v - r_G - r_H$ and the rank condition (4.107), i.e., the matrix in (4.107) has to have full (column) rank, is satisfied for certain choices of the dynamic selector S_D. If we choose the dynamic selector in the form

$$S_D = \begin{bmatrix} S_{D1} & S_{D2} & S_{D3} \end{bmatrix} = \begin{bmatrix} S_p & 0 & 0 \\ 0 & S_v & 0 \\ 0 & 0 & S_r \end{bmatrix} \tag{4.109}$$

and if we choose $S_r = I_{n_r}$, then we get from (4.107) the (rank) conditions for choosing the remaining dynamic selectors S_p and S_v such that

$$\operatorname{rank}(\begin{bmatrix} S_p \\ G \end{bmatrix}) = n_p \quad \text{and} \quad \operatorname{rank}(\begin{bmatrix} S_v M \\ HZ \\ GZ \end{bmatrix}) = n_v. \tag{4.110}$$

In the case of nonredundant constraints, the condition (4.106) for the determination of the dynamic selector S_D of size $n_{f_p} + n_{f_v} + n_r \times n_p + n_v + n_r$, with $n_{f_p} = n_p - n_\lambda$ and $n_{f_v} = n_v - n_\lambda - n_\mu$, corresponds to the condition that the matrix in (4.106) has to be nonsingular for all $(x, p) \in \mathbb{M}$. Analogous to the considerations above, this condition is equivalent to the condition that the matrices in (4.107) and (4.108) are nonsingular for all $x \in \mathbb{M}$. The nonsingularity of the matrix in (4.108) follows from Assumption (4.55a)-(4.55c) and (4.56). Furthermore, if we choose the dynamic selector in the form (4.109) and if we choose $S_r = I_{n_r}$, then we get the conditions for choosing the remaining dynamic selectors S_p and S_v such that the matrices

$$\begin{bmatrix} S_p \\ G \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} S_v M \\ HZ \\ GZ \end{bmatrix}$$

have to be nonsingular.

Lemma 4.6.4 *Let the equations of motion of modeling level 4 (4.43) be quasi-regular, i.e., they satisfy Assumptions 4.2.4. Furthermore, let the constraints (4.43g) and the contact equations (4.43e) be continuously differentiable with respect to p and s and continuous in u, i.e., $g \in \mathcal{C}^{1,1,0}(\mathbb{M}_p, \mathbb{R}^{n_\lambda})$, $c \in \mathcal{C}^{1,1,0}(\mathbb{M}_p, \mathbb{R}^{n_s})$. Then, there exists a matrix function $K_p \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_p, n_{f_p}})$ depending on (p, u) with $n_{f_p} = n_p - r_G$ such that K_p has full (column) rank and*

$$G(p, s(p, u), u)K_p(p, u) = 0 \text{ for every } (p, s, u) \in \mathbb{M}_p.$$

Proof: From Assumptions 4.2.4 together with the Implicit Function Theorem 2.3.1, it follows that the contact variables s are uniquely determined by the position variables p and the control variables u such that we have $s \in \mathcal{C}^{1,0}(\mathbb{M}_p, \mathbb{R}^{n_s})$ as function of (p, u). Hence, we get $G \in \mathcal{C}^{1,0}(\mathbb{M}_p, \mathbb{R}^{n_\lambda, n_p})$ depending on (p, u), i.e., $G = G(p, s(p, u), u)$, with $\operatorname{rank}(G) = r_G$, see Lemma 4.2.11, and the assertion follows from Lemma 2.1.6. □

Lemma 4.6.5 *Let the equations of motion of modeling level 4 (4.43) be quasi-regular, i.e., they satisfy Assumptions 4.2.4. Furthermore, let the contact equations (4.43e) be continuously differentiable with respect to p and s and continuous in u, i.e., $c \in \mathcal{C}^{1,1,0}(\mathbb{M}_p, \mathbb{R}^{n_s})$. For the holonomic constraints (4.43g) and the nonholonomic constraints (4.43f), let the matrix functions $G(p, s, u)Z(p)$ and $H(p, s, u)Z(p)$ be continuous in p, s, and u, i.e., $GZ \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_\lambda, n_v})$ and $HZ \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_\mu, n_v})$, respectively. Then, there exists a matrix function $K_v \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_v, n_{f_v}})$ depending on (p, u) with $n_{f_v} = n_v - r_G - r_H$ such that K_v has full (column) rank and*

$$\begin{bmatrix} G(p, s, u)Z(p) \\ H(p, s, u)Z(p) \end{bmatrix} K_v(p, u) = 0 \text{ for every } (p, s, u) \in \mathbb{M}_p.$$

Proof: From Assumptions 4.2.4 together with the Implicit Function Theorem 2.3.1 and the smoothness assumption on $c(p, s, u)$, it follows that the contact variables

s are uniquely determined by the position variables p and the control variables u such that we have $s \in \mathcal{C}^{1,0}(\mathbb{M}_p, \mathbb{R}^{n_s})$ depending on (p,u). Hence, we get

$$\begin{bmatrix} GZ \\ HZ \end{bmatrix} \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_\lambda + n_\mu, n_v}),$$

depending on (p,u), i.e., $G = G(p, s(p,u), u)$ and $H = H(p, s(p,u), u)$, with a rank $r_G + r_H$, see Lemma 4.2.10, and the assertion follows directly from Lemma 2.1.6. □

Lemma 4.6.6 *Let $K \in \mathcal{C}^0(\mathbb{X} \times \mathbb{U}, \mathbb{R}^{m,n})$ have full (column) rank for every $(p,u) \in \mathbb{X} \times \mathbb{U}$. Then there exists a map $S \in \mathcal{C}^0(\mathbb{X} \times \mathbb{U}, \mathbb{R}^{n,m})$ such that*

$$S(p,u)K(p,u) \quad \textit{is nonsingular for every } (p,u) \in \mathbb{X} \times \mathbb{U}.$$

Proof: The proof follows directly from Lemma 2.1.12. □

In preparation for the regularization of the quasi-regular equations of motion of modeling level 4 (4.43) we also need the following two lemmata.

Lemma 4.6.7 *Let the equations of motion of modeling level 4 (4.43) be quasi-regular, i.e., they satisfy Assumptions 4.2.4. For every $(p,u) \in \mathbb{M}_p$, let the columns of a matrix function $K_p \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_p, n_{f_p}})$ depending on (p,u), with $n_{f_p} = n_p - r_G$, span $\ker(G(p, s(p,u), u))$ with $G \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_\lambda, n_p})$ and $\operatorname{rank}(G(p,s,u)) = r_G$ for all $(p,s,u) \in \mathbb{M}_p$. Furthermore, let a matrix function $S_p \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_{f_p}, n_p})$ depending on (p,u) be given such that $S_p(p,u)K_p(p,u)$ is nonsingular for all $(p,u) \in \mathbb{M}_p$. Then*

$$\operatorname{rank}\left(\begin{bmatrix} S_p(p,u) \\ G(p, s(p,u), u) \end{bmatrix} \right) = n_p, \tag{4.111}$$

i.e., the matrix in (4.111) has full (column) rank, for all $(p,u) \in \mathbb{M}_p$.

Proof: Let the matrix function $\bar{K}_p \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_p, r_G})$ depending on (p,u) be given such that the columns of $\bar{K}_p(p,u)$ span $\operatorname{coker}(G(p, s(p,u), u))$. Hence, we have that $\operatorname{rank}(G(p, s(p,u), u)\bar{K}_p(p,u)) = r_G$.
Because of the nonsingularity of the matrix $\begin{bmatrix} K_p & \bar{K}_p \end{bmatrix}$ we have

$$\operatorname{rank}\left(\begin{bmatrix} S_p \\ G \end{bmatrix} \right) = \operatorname{rank}\left(\begin{bmatrix} S_pK_p & S_p\bar{K}_p \\ GK_p & G\bar{K}_p \end{bmatrix} \right) = \operatorname{rank}\left(\begin{bmatrix} S_pK_p & S_p\bar{K}_p \\ 0 & G\bar{K}_p \end{bmatrix} \right).$$

With the assumed nonsingularity of the matrix function $S_pK_p \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_{f_p}, n_{f_p}})$ and $\operatorname{rank}(G(p, s(p,u), u)\bar{K}_p(p,u)) = r_G$ it follows that

$$\operatorname{rank}\left(\begin{bmatrix} S_p \\ G \end{bmatrix} \right) = \operatorname{rank}\left(\begin{bmatrix} S_pK_p & S_p\bar{K}_p \\ 0 & G\bar{K}_p \end{bmatrix} \right) = n_{f_p} + r_G = n_p$$

and we have the assertion. □

Lemma 4.6.8 *Let the equations of motion of modeling level 4 (4.43) be quasi-regular, i.e., they satisfy Assumptions 4.2.4. For every $(p,u) \in \mathbb{M}_p$, let the columns of a matrix function $K_v \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_v, n_{f_v}})$, with $n_{f_v} = n_v - r_G - r_H$, depending on (p,u) span*

$$\ker\left(\begin{bmatrix} G(p, s(p,u), u)Z(p) \\ H(p, s(p,u), u)Z(p) \end{bmatrix} \right)$$

with $GZ \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_\lambda, n_v})$ *and* $HZ \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_\mu, n_v})$. *Furthermore, let a matrix function* $S_v \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_{f_v}, n_v})$ *depending on* (p,u) *be given such that the matrix function* $S_v(p,u)M(p,u)K_v(p,u)$ *is nonsingular for all* $(p,u) \in \mathbb{M}_p$. *Then*

$$\operatorname{rank}\left(\begin{bmatrix} S_v(p,u)M(p,u) \\ H(p,s(p,u),u)Z(p) \\ G(p,s(p,u),u)Z(p) \end{bmatrix}\right) = n_v, \tag{4.112}$$

i.e., the matrix in (4.112) has full (column) rank for all $(p,u) \in \mathbb{M}_p$.

Proof: From Lemma 4.2.11 we have that

$$\operatorname{rank}\left(\begin{bmatrix} G(p,s(p,u),u)Z(p) \\ H(p,s(p,u),u)Z(p) \end{bmatrix}\right) = r_G + r_H$$

for all $(p,u) \in \mathbb{M}_p$. Let $\bar{K}_v \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_v, r_G + r_H})$ depending on (p,u) be given such that the columns of $\bar{K}_v(p,u)$ span

$$\operatorname{coker}\left(\begin{bmatrix} G(p,s(p,u),u)Z(p) \\ H(p,s(p,u),u)Z(p) \end{bmatrix}\right),$$

i.e.,

$$\operatorname{rank}\left(\begin{bmatrix} G(p,s(p,u),u)Z(p) \\ H(p,s(p,u),u)Z(p) \end{bmatrix} \bar{K}_v(p,u)\right) = r_G + r_H. \tag{4.113}$$

Because of the nonsingularity of the matrix $\begin{bmatrix} K_v & \bar{K}_v \end{bmatrix}$ we have

$$\operatorname{rank}\left(\begin{bmatrix} S_vM \\ HZ \\ GZ \end{bmatrix}\right) = \operatorname{rank}\left(\begin{bmatrix} S_vMK_v & S_vM\bar{K}_v \\ GZK_v & GZ\bar{K}_v \\ HZK_v & HZ\bar{K}_v \end{bmatrix}\right) = \operatorname{rank}\left(\begin{bmatrix} S_vMK_v & S_vM\bar{K}_v \\ 0 & GZ\bar{K}_v \\ 0 & HZ\bar{K}_v \end{bmatrix}\right).$$

With the assumed nonsingularity of $S_vMK_v \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_{f_v}, n_{f_v}})$ and (4.113) it follows that

$$\operatorname{rank}\left(\begin{bmatrix} S_vM \\ HZ \\ GZ \end{bmatrix}\right) = \operatorname{rank}\left(\begin{bmatrix} S_vMK_v & S_vM\bar{K}_v \\ 0 & GZ\bar{K}_v \\ 0 & HZ\bar{K}_v \end{bmatrix}\right) = n_{f_v} + r_G + r_H = n_v$$

and the assertion follows. □

Theorem 4.6.9 *Let the equations of motion of modeling level 4 (4.43) be quasi-regular, i.e., they satisfy Assumptions 4.2.4. Then there exists a selector* $S_p \in \mathcal{C}^0(\mathbb{M}, \mathbb{R}^{n_{f_p}, n_p})$ *with* $n_{f_p} = n_p - r_G$ *and there exists a selector* $S_v \in \mathcal{C}^0(\mathbb{M}, \mathbb{R}^{n_{f_v}, n_v})$ *with* $n_{f_v} = n_v - r_G - r_H$ *such that the differential-algebraic system*

$$\begin{aligned}
S_p(p,u)\dot{p} &= S_p(p,u)Z(p)v, && (4.114a)\\
S_v(p,u)M(p,u)\dot{v} &= S_v(p,u)f(p,v,r,w,s,\lambda,\mu,u) && (4.114b)\\
&\quad -S_v(p,u)Z^T(p)G^T(p,s,u)\lambda - S_v(p,u)Z^T(p)H^T(p,s,u)\mu,\\
\dot{r} &= b(p,v,r,w,s,\lambda,\mu,u), && (4.114c)\\
0 &= d(p,v,r,w,s,\lambda,\mu,u), && (4.114d)\\
0 &= c(p,s,u), && (4.114e)\\
0 &= H(p,s,u)Z(p)v + h(p,s,u), && (4.114f)\\
0 &= g(p,s,u), && (4.114g)\\
0 &= h^I(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^1), && (4.114h)\\
0 &= g^I(p,v,s,\mathfrak{u}^1), && (4.114i)\\
0 &= g^{II}(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^2) && (4.114j)
\end{aligned}$$

has the same solution set as the equations of motion of modeling level 4 (4.43). Furthermore, it has maximal constraint level $\nu_c = 0$ and it is strangeness-free.

Proof: If the constraints (4.43f) and (4.43g) are contradictory then the solution set of the differential-algebraic system (4.114) as well as the solution set of the original differential-algebraic system (4.43) are empty and therefore identical.
In the following, we will consider the case with noncontradictory constraints (4.43f) and (4.43g) and we will omit the dependencies on p, v, r, w, s, λ, μ, and u. The existence of the selectors $S_p \in \mathcal{C}^0(\mathbb{M}, \mathbb{R}^{n_{f_p}, n_p})$ and $S_v \in \mathcal{C}^0(\mathbb{M}, \mathbb{R}^{n_{f_v}, n_v})$ is proved in Lemmata 4.6.4-4.6.8. Furthermore, from the construction of S_p and S_v it follows that the system (4.114) has maximal constraint level $\nu_c = 0$.
According to Lemma 3.5.1 we get for the semi-implicit DAE (4.114) that $m_1 = n_{f_p} + n_{f_v} + n_r$ and $r_C = n_w + n_s + 3r_G + 2r_H$. Furthermore, the conditions in Lemma 3.5.1 are satisfied. In particular, the condition (4.106) for the determination of the dynamic selector corresponds to the condition (3.27). Therefore, it follows from Lemma 3.5.1 that the DAE (4.114) is strangeness-free.
It remains to show that the solution set of (4.114) is identical to the solution set of (4.43). It is obvious that a solution of (4.43) is also a solution of (4.114). The other direction will be proven in the following.
With the trivial equation $S_pZv = S_pZv$ and (4.114i) in the form of (4.50c) we get

$$\begin{bmatrix} S_p \\ G \end{bmatrix} Zv = \begin{bmatrix} S_pZv \\ -(g_{,u} - g_{,s}c_{,s}^{-1}c_{,u})\dot{u} \end{bmatrix}. \tag{4.115}$$

On the other hand, it follows from (4.114a) and from the derivative of (4.114g) with respect to t in the form (4.50b), that

$$\begin{bmatrix} S_p \\ G \end{bmatrix} \dot{p} = \begin{bmatrix} S_pZv \\ -(g_{,u} - g_{,s}c_{,s}^{-1}c_{,u})\dot{u} \end{bmatrix}. \tag{4.116}$$

Because of the full rank of the matrix $\begin{bmatrix} S_p^T & G^T \end{bmatrix}^T$, see Lemma 4.6.7, it follows from the General Implicit Function Theorem 2.3.2 that Zv as well as $\dot{p}$ are uniquely determined by (4.115) and (4.116), respectively. Therefore, from (4.115) and (4.116) we get the kinematical equations of motion (4.43a)

$$\dot{p} = Zv.$$

Furthermore, with the trivial equation $S_v(f - Z^TG^T\lambda - Z^TH^T\mu) = S_v(f - Z^TG^T\lambda - Z^TH^T\mu)$ and (4.114h) in the form of (4.53e) and (4.114j) in the form of (4.51d) we get

$$\begin{bmatrix} S_vM \\ GZ \\ HZ \end{bmatrix} M^{-1}(f - Z^TG^T\lambda - Z^TH^T\mu) \tag{4.117}$$

$$= \begin{bmatrix} S_v(f - Z^TG^T\lambda - Z^TH^T\mu) \\ -(g_{,p}^I - g_{,s}^I c_{,s}^{-1}c_{,p})Zv - g_{,\mathfrak{u}^1}^I \dot{\mathfrak{u}}^1 + g_{,s}^I c_{,s}^{-1}c_{,u}\dot{u} \\ -(\breve{h}_{,p} - \breve{h}_{,s}c_{,s}^{-1}c_{,p})Zv - (\breve{h}_{,u} - \breve{h}_{,s}c_{,s}^{-1}c_{,u})\dot{u} \end{bmatrix}. \tag{4.118}$$

On the other hand, it follows from (4.114b), from the derivative of (4.114i) with respect to t in the form (4.51c), and from the derivative of (4.114f) with respect to t in the form (4.53d) that

$$\begin{bmatrix} S_vM \\ GZ \\ HZ \end{bmatrix} \dot{v} = \begin{bmatrix} S_v(f - Z^TG^T\lambda - Z^TH^T\mu) \\ -(g_{,p}^I - g_{,s}^I c_{,s}^{-1}c_{,p})Zv - g_{,\mathfrak{u}^1}^I \dot{\mathfrak{u}}^1 + g_{,s}^I c_{,s}^{-1}c_{,u}\dot{u} \\ -(\breve{h}_{,p} - \breve{h}_{,s}c_{,s}^{-1}c_{,p})Zv - (\breve{h}_{,u} - \breve{h}_{,s}c_{,s}^{-1}c_{,u})\dot{u} \end{bmatrix}.$$

Because of the full rank of the matrix

$$\begin{bmatrix} S_v M \\ GZ \\ HZ \end{bmatrix},$$

see Lemma 4.6.8, it follows from the General Implicit Function Theorem 2.3.2 that $M^{-1}(f - Z^T G^T \lambda - Z^T H^T \mu)$ as well as $\dot{v}$ are uniquely determined by (4.117) and (4.119), respectively. Therefore, from (4.117) and (4.119) we get the dynamical equations of motion (4.43b)

$$M\dot{v} = f - Z^T G^T \lambda - Z^T H^T \mu.$$

In addition, the equations equations (4.43c)-(4.43g) are explicitly contained in both formulations and therefore, satisfied. Hence, a solution of (4.114) is also a solution of (4.43). □

Remark 4.6.10 In the case of regular equations of motion , i.e., with nonredundant constraints, it follows from Lemma 3.5.29 that the DAE (4.114) has d-index $\nu_d = \nu_c + 1 = 1$. □

Analogous to the projected-strangeness-free DAE developed in Section 3.5.3 we will call the DAE (4.114) the *projected-strangeness-free formulation of the equations of motion of modeling level 4*. Because of the same solution set and the reduced maximal constraint level and the reduced index, the projected-strangeness-free formulation (4.114) corresponds to a regularization of the equations of motion.
The projected strangeness-free formulation (4.114) is a quasi-linear DAE of $n_p + n_v + n_r + n_w + n_s + n_\lambda + n_\mu + 2(n_\lambda - r_G) + (n_\mu - r_H)$ equations and $n_p + n_v + n_r + n_w + n_s + n_\lambda + n_\mu$ unknown variables which corresponds to an overdetermined DAE if $n_\lambda > r_G$ or $n_\mu > r_H$, i.e., in the case of redundant constraints. Furthermore, in general, a selection of a set of nonredundant constraints is not recommended, since a selection of the constraints possibly does change the solution set of the projected-strangeness-free formulation (4.114) which leads to possibly wrong results in the analytical or numerical solution, see Example 2.3.15.
In the following we will discuss the special case of the equations of motion of modeling level 2 with $Z = I_{n_p}$ and a positive definite mass matrix M which offers the possibility for simplifications of the regularization to the projected-strangeness-free formulation.

Lemma 4.6.11 *Let the equations of motion of modeling level 2 (4.41) satisfy Assumptions 4.2.4. Furthermore, let $M \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_v, n_v})$ be positive definite and $Z(p) = I_{n_p}$ for all $(p, u) \in \mathbb{M}$.*
Then there exists a selector $S_p \in \mathcal{C}^0(\mathbb{M}, \mathbb{R}^{n_{f_p}, n_p})$ with $n_{f_p} = n_p - r_G$ such that the differential-algebraic system

$$\begin{aligned}
S_p(p,u)\dot{p} &= S_p(p,u)v, && \text{(4.119a)}\\
S_p(p,u)M(p,u)\dot{v} &= S_p(p,u)f(p,v,r,w,s,\lambda,u) && \text{(4.119b)}\\
&\quad -S_p(p,u)G^T(p,s,u)\lambda, &&\\
\dot{r} &= b(p,v,r,w,s,\lambda,u), && \text{(4.119c)}\\
0 &= d(p,v,r,w,s,\lambda,u), && \text{(4.119d)}\\
0 &= c(p,s,u), && \text{(4.119e)}\\
0 &= g(p,s,u), && \text{(4.119f)}\\
0 &= g^I(p,v,s,\mathfrak{u}^1), && \text{(4.119g)}\\
0 &= g^{II}(p,v,r,w,s,\lambda,\mathfrak{u}^2) && \text{(4.119h)}
\end{aligned}$$

has the same solution set as the equations of motion of modeling level 2 (4.41) and is strangeness-free.

Proof: From $Z(p) = I_{n_p}$ it follows that $n_v = n_p$. Choosing the selector S_p in such a way that $\operatorname{range}(S_p^T) = \ker(G)$, i.e., $GS_p^T = 0$. Then the selector S_p satisfies both conditions (4.110), i.e.,

$$\operatorname{rank}\left(\begin{bmatrix} S_p \\ G \end{bmatrix}\right) = n_p \quad \text{and} \quad \operatorname{rank}\left(\begin{bmatrix} S_p M \\ G \end{bmatrix}\right) = n_v = n_p$$

because of the positive definiteness of the mass matrix M, see Lemma A.2.18. The remaining part of the proof follows from the proof of Theorem 4.6.9. □

With these preparations we have presented all the tools to perform the solution manifold preserving index reduction of the quasi-regular nonlinear equations of motion of modeling level 4 (4.43) as follows.

Algorithm 4.6.12 (Solution manifold preserving strangeness deletion)
The equations of motion of modeling level 4 (4.43) are assumed to be quasi-regular, i.e., they satisfy Assumptions 4.2.4, with noncontradictory constraints (4.43f) and (4.43g) having constant rank. Furthermore, let $M \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_v, n_v})$ and $Z \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_p, n_v})$.
Then the regularization by index reduction is done by choosing a selector $S_p \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_{f_p}, n_p})$ and a selector $S_v \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_{f_v}, n_v})$ depending on (p, u) with $n_{f_p} = n_p - r_G$ and $n_{f_v} = n_v - r_G - r_H$, in the following way.

1. **Determination of selector S_p**
   (a) Determine $K_p \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_p, n_{f_p}})$ depending on (p, u) such that the columns of $K_p(p, u)$ span $\ker(G(p, s(p, u), u))$ for all $(p, u) \in \mathbb{M}_p$, see Lemma 4.6.4.
   (b) Determine the selector $S_p \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_{f_p}, n_p})$ depending on (p, u) such that $S_p(p, u)K_p(p, u)$ is nonsingular for all $(p, u) \in \mathbb{M}_p$, see Lemma 4.6.7.
2. **Determination of selector S_v**
   (a) Determine $K_v \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_v, n_{f_v}})$ depending on (p, u) such that the columns of $K_v(p, u)$ span
   $$\ker\left(\begin{bmatrix} G(p, s(p, u), u)Z(p) \\ H(p, s(p, u), u)Z(p) \end{bmatrix}\right)$$
   for all $(p, u) \in \mathbb{M}_p$, see Lemma 4.6.5.
   (b) Determine the selector $S_v \in \mathcal{C}^0(\mathbb{M}_p, \mathbb{R}^{n_{f_v}, n_v})$ depending on (p, u) such that $S_v(p, u)M(p, u)K_v(p, u)$ is nonsingular for all $(p, u) \in \mathbb{M}_p$, see Lemma 4.6.8.
3. **Projected strangeness-free form of the equations of motion**
   By appending the constraints on velocity level (4.50a) and the constraints on acceleration level (4.51a) and(4.53a), the projected strangeness-free form of the equations of motion is given by (4.114).

With this algorithm we are able to determine an equivalent strangeness-free form (4.114) of the equations of motion which contains all information about the solution manifold (4.2.14). The strangeness-free form created in this way is analytically equivalent to the original equations of motion in the sense that both have the same solution set. Furthermore, this form is suitable for numerical integration using stiff ODE solvers like implicit Runge-Kutta-Methods or BDF methods.

Remark 4.6.13 Note that the selectors S_p and S_v satisfying the nonsingularity conditions (4.111) and (4.112) are not uniquely determined. Rather it is possible to choose the selectors in a piecewise constant fashion. This fact is of great advantage and importance for the numerical integration because it offers the possibility to reduce the amount of work for the computation of the selectors. But on the other hand the choice of the selectors influences the conditioning of the projected-strangeness-free formulation. In particular, this means, that the condition number of the iteration matrix $\mathfrak{N}$ (see Section 3.5.4.2), e.g., for solving the nonlinear stage equations (3.113a), depends directly on the choice of the selectors. □

Let us perform Algorithm 4.6.12 for the example of the mathematical pendulum.

Example 4.6.14 The mathematical pendulum: Following Algorithm 4.6.12 we have to consider G which is given in (4.21) as

$$G = \begin{bmatrix} 2p_1 & 2p_2 \end{bmatrix}.$$

The matrix function K_p can be determined as

$$K_p = \begin{bmatrix} -p_2 \\ p_1 \end{bmatrix}$$

and, therefore, the selector S_p can be chosen as

$$S_p = \begin{bmatrix} -p_2 & p_1 \end{bmatrix}$$

such that

$$S_p K_p = \begin{bmatrix} -p_2 & p_1 \end{bmatrix} \begin{bmatrix} -p_2 \\ p_1 \end{bmatrix} = \begin{bmatrix} p_2^2 + p_1^2 \end{bmatrix} = \begin{bmatrix} L^2 \end{bmatrix},$$

see the constraints (4.20c). Since the mass matrix is given such that $M = mI$, the equations of motion satisfy the conditions of Lemma 4.6.11 and we can use $S_v = S_p$ and we get the projected strangeness-free formulation

$$\begin{aligned} -p_2\dot{p}_1 + p_1\dot{p}_2 &= -p_2 v_1 + p_1 v_2, && (4.120a)\\ -mp_2\dot{v}_1 + mp_1\dot{v}_2 &= -mgp_1, && (4.120b)\\ 0 &= p_1^2 + p_2^2 - L^2, && (4.120c)\\ 0 &= 2p_1 v_1 + 2p_2 v_2, && (4.120d)\\ 0 &= 2v_1^2 + 2v_2^2 - 2p_2 g - \frac{4}{m}(p_1^2 + p_2^2)\lambda_1. && (4.120e) \end{aligned}$$

As mentioned in Remark 4.6.13, the selectors S_p and S_v are not uniquely determined by the conditions (4.111) and (4.112) or the Algorithm 4.6.12. In particular, the selectors can be chosen to be piecewise constant.
Let us consider this fact for the pendulum with the initial state $p_1 = 0$ und $p_2 = -L$, i.e., the pendulum is hanging downwards. In this position the selectors can be determined as

$$S_p(p,u) = S_v(p,u) = \begin{bmatrix} L & 0 \end{bmatrix}. \tag{4.121}$$

Keeping these selectors constant, the leading matrix of the left-hand side of the underlying ordinary differential equations, (obtained by substituting the algebraic equations in (4.120) by their derivatives with respect to t) is

$$\begin{bmatrix} L & 0 & 0 & 0 & 0 \\ 0 & 0 & mL & 0 & 0 \\ 2p_1 & 2p_2 & 0 & 0 & 0 \\ \times & \times & 2p_1 & 2p_2 & 0 \\ \times & \times & \times & \times & \frac{4}{m}(p_1^2 + p_2^2) \end{bmatrix}. \tag{4.122}$$

Obviously, this matrix is nonsingular as long as p_2 does not become zero. In particular, this means that as long as that the pendulum does not reach one of the horizontal positions, i.e., $p_1 = \pm L$ and $p_2 = 0$, the selectors may be chosen constant as in (4.121). Otherwise, if the pendulum reaches or crosses the horizontal position, the matrix (4.122) becomes singular and the first and third as well as the second and fourth equations are redundant such that the solution is not uniquely defined. Furthermore, the condition number of matrix (4.122) goes to infinity as p_2 goes to zero.
For these reason, in the neighborhood of the horizontal position of the pendulum new selectors have to be determined. See also the Example 5.3.1 for numerical results. □

4.6.2.4 Projected-s-index-1 formulation

In addition to the projected-strangeness-free formulation of quasi-linear DAEs we discussed in Remark 3.5.55 an incomplete regularization which leads to a DAE with reduced maximal constraint level and reduced index that has the same solution set but which is not necessarily strangeness-free. In the following we will discuss an incomplete regularization of level 1 for the equations of motion analogously to Example 3.5.56.

Theorem 4.6.15 *Let the equations of motion of modeling level 4 (4.43) be quasi-regular, i.e., they satisfy Assumptions 4.2.4. Furthermore, let the constraints (4.43d)-(4.43g), (4.50a) be continuously differentiable. Then there exists a selector* $S_p \in \mathcal{C}^0(\mathbb{M}, \mathbb{R}^{n_{f_p}, n_p})$ *with* $n_{f_p} = n_p - r_G$ *such that the differential-algebraic system*

$$
\begin{aligned}
S_p(p,u)\dot{p} &= S_p(p,u)Z(p)v, && \text{(4.123a)}\\
M(p,u)\dot{v} &= f(p,v,r,w,s,\lambda,\mu,u) && \text{(4.123b)}\\
&\quad -Z^T(p)G^T(p,s,u)\lambda - Z^T(p)H^T(p,s,u)\mu,\\
\dot{r} &= b(p,v,r,w,s,\lambda,\mu,u), && \text{(4.123c)}\\
0 &= d(p,v,r,w,s,\lambda,\mu,u), && \text{(4.123d)}\\
0 &= c(p,s,u), && \text{(4.123e)}\\
0 &= H(p,s,u)Z(p)v + h(p,s,u), && \text{(4.123f)}\\
0 &= g(p,s,u), && \text{(4.123g)}\\
0 &= g^I(p,v,s,\mathfrak{u}^1) && \text{(4.123h)}
\end{aligned}
$$

has the same solution set as the equations of motion of modeling level 4 (4.43) and has maximal constraint level $\nu_c = 1$.

Proof: If the constraints (4.43f) and (4.43g) are contradictory then the solution set of the differential-algebraic system (4.123) as well as the solution set of the original differential-algebraic system (4.43) are empty and therefore identical.
In the following, we will consider the case with noncontradictory constraints (4.43f) and (4.43g) and we will omit the dependencies on p, v, r, w, s, λ, μ, and u. The existence of the selector $S_p \in \mathcal{C}^0(\mathbb{M}, \mathbb{R}^{n_{f_p}, n_p})$ is proved in Lemmata 4.6.4-4.6.7.
Following Procedure 3.5.11 we get according to (4.123)

$$E^0\dot{x} = k^0$$

with $x = \begin{bmatrix} p^T & v^T & r^T & w^T & s^T & \lambda^T & \mu^T \end{bmatrix}^T$ and

$$E^0 = \begin{bmatrix} S_p & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & M & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \; k^0 = \begin{bmatrix} S_p Z v \\ f - Z^T G^T \lambda - Z^T H^T \mu \\ b \\ d \\ c \\ HZv + h \\ g \\ g^I \end{bmatrix}.$$

This is equivalent to the intermediate DAE (3.40) for $i = 0$

$$\widetilde{E}^0 \dot{x} = \widetilde{k}^0$$

with $\widetilde{E}^0 = E^0$ and $\widetilde{k}^0 = k^0$. We get the constraint set of level 0

$$\mathbb{M}_0 = \{(x, \mathfrak{u}^1) \in \mathbb{X} \times \mathbb{U}^1 \; : \; 0 = d, \; 0 = c, \; 0 = HZv + h, \; 0 = g, \; 0 = g^I\}.$$

Differentiation of the constraints leads to

$$E^1 \dot{x} = k^1$$

with

$$E^1 = \begin{bmatrix} S_p & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & M & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 \\ d_{,p} & d_{,v} & d_{,r} & d_{,w} & d_{,s} & d_{,\lambda} & d_{,\mu} \\ c_{,p} & 0 & 0 & 0 & c_{,s} & 0 & 0 \\ \breve{h}_{,p} & HZ & 0 & 0 & \breve{h}_{,s} & 0 & 0 \\ g_{,p} & 0 & 0 & 0 & g_{,s} & 0 & 0 \\ g^I_{,p} & g^I_{,v} & 0 & 0 & g^I_{,s} & 0 & 0 \end{bmatrix}, \; k^1 = \begin{bmatrix} S_p Z v \\ f - Z^T H^T \mu - Z^T G^T \lambda \\ b \\ -d_{,u} \dot{u} \\ -c_{,u} \dot{u} \\ -\breve{h}_{,u} \dot{u} \\ -g_{,u} \dot{u} \\ -g^I_{,\mathfrak{u}^1} \dot{\mathfrak{u}}^1 \end{bmatrix}.$$

Elimination leads to

$$\widetilde{E}^1 \dot{x} = \widetilde{k}^1$$

with

$$\widetilde{E}^1 = \begin{bmatrix} \Gamma & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & M & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & d_{,w} & d_{,s} & d_{,\lambda} & d_{,\mu} \\ 0 & 0 & 0 & 0 & c_{,s} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\widetilde{k}^1 = \begin{bmatrix} \gamma \\ f - Z^T G^T \lambda - Z^T H^T \mu \\ b \\ -d_{,u} \dot{u} - d_{,v} M^{-1} (f - Z^T G^T \lambda - Z^T H^T \mu) - d_{,r} b - d_p \Gamma^{-1} \gamma \\ -c_{,u} \dot{u} - c_p \Gamma^{-1} \gamma \\ -h^I \\ -g^{I\!I} \\ \bar{S}_\lambda (-g_{,u} \dot{u} + g_{,s} c_{,s}^{-1} c_{,u} \dot{u}) \end{bmatrix}$$

where $\widetilde{G} = S_\lambda G$ has full rank, with

$$\Gamma(p,s,u) = \begin{bmatrix} S_p \\ \widetilde{G} \end{bmatrix} \quad \text{and} \quad \gamma(p,s,u) = \begin{bmatrix} S_p Z v \\ S_\lambda(-g_{,u}\dot{u} + g_{,s}c_{,s}^{-1}c_{,u}\dot{u}) \end{bmatrix}.$$

Note that Γ is nonsingular. From the fact that the holonomic constraints are noncontradictory, it follows that $\bar{S}_\lambda(-g_{,u}\dot{u} + g_{,s}c_{,s}^{-1}c_{,u}\dot{u}) = 0$ for all $(x, \mathfrak{u}^1) \in \mathbb{M}_0 \times \mathbb{U}^{(1)}$. According to Remark 3.5.12c, therefore, it is not necessary to take the last block equations into account for further investigations. Further differentiation of the algebraic equations (without the last block equations) leads to

$$E^2 \dot{x} = k^2$$

with

$$E^2 = \begin{bmatrix} \Gamma & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & M & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & d_{,w} & d_{,s} & d_{,\lambda} & d_{,\mu} \\ 0 & 0 & 0 & 0 & c_{,s} & 0 & 0 \\ \times & \times & \times & -HZM^{-1}f_{,w} & \times & E^2_{66} & E^2_{67} \\ \times & \times & \times & -GZM^{-1}f_{,w} & \times & E^2_{76} & E^2_{77} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

where

$$\begin{aligned} E^2_{66} &= -HZM^{-1}(f_{,\lambda} - Z^T G^T), & E^2_{67} &= -HZM^{-1}(f_{,\mu} - Z^T H^T), \\ E^2_{76} &= -GZM^{-1}(f_{,\lambda} - Z^T G^T), & E^2_{77} &= -GZM^{-1}(f_{,\mu} - Z^T H^T), \end{aligned}$$

and with

$$k^2 = \begin{bmatrix} \gamma \\ f - Z^T G^T \lambda - Z^T H^T \mu \\ b \\ -d_{,u}\dot{u} - d_{,v}M^{-1}(f - Z^T G^T \lambda - Z^T H^T \mu) - d_{,r}b - d_p \Gamma^{-1}\gamma \\ -c_{,u}\dot{u} - c_p \Gamma^{-1}\gamma \\ h^I_{,\mathfrak{u}^1}\dot{\mathfrak{u}}^1 \\ g^{II}_{,\mathfrak{u}^2}\dot{\mathfrak{u}}^2 \\ \bar{S}_\lambda(-g_{,u}\dot{u} + g_{,s}c_{,s}^{-1}c_{,u}\dot{u}) \end{bmatrix}$$

Elimination leads to

$$\widetilde{E}^2 \dot{x} = \widetilde{k}^2$$

with

$$\widetilde{E}^2 = \begin{bmatrix} \Gamma & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & M & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & d_{,w} & d_{,s} & d_{,\lambda} & d_{,\mu} \\ 0 & 0 & 0 & 0 & c_{,s} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & S_\mu HZM^{-1}G_\lambda & S_\mu HZM^{-1}H_\mu \\ 0 & 0 & 0 & 0 & 0 & S_\lambda GZM^{-1}G_\lambda & S_\lambda GZM^{-1}H_\mu \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and

$$\widetilde{k}^2 = \begin{bmatrix} \overset{\gamma}{\lambda} \\ f - Z^T G^T \lambda - Z^T H^T \mu \\ b \\ -d_{,u}\dot{u} - d_{,v} M^{-1}(f - Z^T G^T \lambda - Z^T H^T \mu) - d_{,r} b - d_p \Gamma^{-1} \gamma \\ -c_{,u}\dot{u} - c_p \Gamma^{-1} \gamma \\ S_\mu \widetilde{h}^{II} \\ S_\lambda \widetilde{g}^{III} \\ \bar{S}_\mu \widetilde{h}^{II} \\ \bar{S}_\lambda \widetilde{g}^{III} \\ \bar{S}_\lambda(-g_{,u}\dot{u} + g_{,s} c_{,s}^{-1} c_{,u}\dot{u}) \end{bmatrix}$$

by use of the the matrices $S_\lambda(p,u)$ and $\bar{S}_\lambda(p,u)$ and $S_\mu(p,u)$ and $\bar{S}_\mu(p,u)$ defined in (4.66). From the fact that the holonomic constraints are noncontradictory it follows that $\bar{S}_\lambda \widetilde{g}^{III} = 0$ and $\bar{S}_\mu \widetilde{h}^{II} = 0$ for all $(x, \mathfrak{u}^1) \in \mathbb{M}_1 \times \mathbb{U}^{(2)}$. Therefore, the last three block equations which represent the algebraic constraints are trivially satisfied for all $(x, \mathfrak{u}^1) \in \mathbb{M}_1 \times \mathbb{U}^{(2)}$ and the procedure terminates with $\nu = 2$ and we get the maximal constraint level $\nu_c = \nu - 1 = 1$.
It remains to show that the solution set of (4.123) is identical to the solution set of (4.43). It is obvious that a solution of (4.43) is also a solution of (4.123). The other direction will be discussed in the following.
With the trivial equation $S_p Z v = S_p Z v$ and (4.123h) in the form of (4.50c) we get

$$\begin{bmatrix} S_p \\ G \end{bmatrix} Z v \;=\; \begin{bmatrix} S_p Z v \\ -(g_{,u} - g_{,s} c_{,s}^{-1} c_{,u})\dot{u} \end{bmatrix}. \tag{4.124}$$

On the other hand, it follows from (4.123a) and from the derivative with respect to t of (4.123g) in the form (4.50b) that

$$\begin{bmatrix} S_p \\ G \end{bmatrix} \dot{p} \;=\; \begin{bmatrix} S_p Z v \\ -(g_{,u} - g_{,s} c_{,s}^{-1} c_{,u})\dot{u} \end{bmatrix}. \tag{4.125}$$

Because of the full rank of the matrix $\begin{bmatrix} S_p^T & G^T \end{bmatrix}^T$, see Lemma 4.6.7, it follows from the General Implicit Function Theorem 2.3.2 that Zv as well as $\dot{p}$ are uniquely defined by (4.124) and (4.125), respectively. Therefore, from (4.115) and (4.116) we get the kinematical equations of motion (4.43a)

$$\dot{p} \;=\; Z v.$$

In addition, the equations (4.43c)-(4.43g) are explicitly contained in both formulations and therefore, satisfied. Hence, a solution of (4.123) is also a solution of (4.43). □

Remark 4.6.16 By use of Hypothesis 3.2.7 it can be shown that the set of differential-algebraic equations (4.123) has s-index $\nu_s = 1$. □

Example 4.6.17 The mathematical pendulum: In Example 4.6.14 we have developed the projected-strangeness-free form of the equations of motion for the mathematical pendulum. In this example we will state the projected-s-index-1 formulation which has the form

$$\begin{aligned} -p_2\dot{p}_1 + p_1\dot{p}_2 &= -p_2 v_1 + p_1 v_2, \\ m\dot{v}_1 &= -2p_1\lambda_1, \\ m\dot{v}_2 &= -mg - 2p_2\lambda_1, \\ 0 &= p_1^2 + p_2^2 - L^2, \\ 0 &= 2p_1 v_1 + 2p_2 v_2. \end{aligned}$$

For numerical results based on this projected-s-index-1 formulation see Example 5.3.1. □

4.7 Numerical methods and software: an overview

As discussed in the previous sections, the numerical integration of the equation of motion of s-index two (or d-index three) is nontrivial. Problems which can appear are for instance convergence order reduction for the used discretization methods, convergence problems of the iterative method for solving the nonlinear system because of ill conditioned iteration matrices, and hidden constraints up to level two. Furthermore, the step size control has to be adapted and the numerical solution may not depend continuously on perturbations of the input data.
Therefore, numerical methods have to combine suitable regularization methods, discussed in Section 4.6, with appropriate discretization methods. Different forms of the equations of motion as basis for numerical integrators of multibody systems are investigated in [15, 23, 60]. In particular, these are the

- original equations of motion (4.43) with s-index 2, d-index 3 (see [136]),
- s-index-1 formulation (4.81) with s-index 1, d-index 2 (see [136]),
- s-index-0 formulation (4.82) with s-index 0, d-index 1 (see [136]),
- state space form (4.5) with s-index 0, d-index 0 (see [52]),
- *semi state space form* with s-index 0, d-index 0 (see [52]),
- underlying ODE (4.79) with s-index 0, d-index 0 (see [52]),
- Baumgarte stabilization (4.101) with s-index 0, d-index 1 (see [18, 19]),
- Gear-Gupta-Leimkuhler formulation (4.103) with s-index 1, d-index 2 (see [68]).

The numerical integration of the equations of motion arising in mechanical systems is considered in several articles like [5, 9, 23, 52, 86, 116, 118, 127, 152, 154, 164]. An overview of suitable numerical methods is provided in [144, 164]. Some numerical algorithms are collected in libraries like `MBSPACK`[5] [164] and `MBSSIM`[6] [3].
Currently widely used solvers for general DAEs are `DASSL`[7] [25, 135] and `RADAU5`[8] [79, 82], see Section 3.6. Both codes do not actually exploit the special structure of multibody systems and therefore, they do not follow the Algorithm Paradigm 4.5.3. The presence of hidden constraints, i.e., the constraints on velocity and acceleration level is not considered.
Therefore, there is a need for numerical integration methods which exploit the properties and the structure of the equations of motion to avoid arising problematic effects. In this context, a large number of numerical methods has been developed for the numerical integration of DAEs arising in multibody dynamics. A detailed overview over the numerical solution of ordinary differential equations and differential-algebraic equations for technical simulations is given in [144, 164].
An important feature of numerical integrators is the type of the interface which allows a classification into *algorithms based on structural evaluations* and *algorithms based on residual evaluations*, see [164]. In particular, while the exploitation of the

[5] `MBSPACK - http://www-m2.ma.tum.de/~simeon/Software/mbspack.tar.gz`
[6] `MBSSIM - http://www1.iwr.uni-heidelberg.de/~Michael.Winckler//Projects/MBSSIM/`
[7] `DASSL - http://www.engineering.ucsb.edu/~cse/software.html`
[8] `RADAU5 - http://www.unige.ch/~hairer/software.html`

dominating structure of the equations of motion is highly developed for algorithms based on structural evaluations, the flexibility with respect to modifications or extensions of the equations of motion is strongly restricted. On the other hand the algorithms based on residual evaluations present a high degree of flexibility but the exploitation of the special structure of the equations of motion is restricted in favor of the adaptability.

In [52, 60] an approach based on the equations of motion of modeling level 0 (4.34) is proposed which adds all hidden constraints to the the equations of motion. This approach leads to an overdetermined system consisting of (4.34), (4.35), and (4.36) consisting of $2n_p + 3n_\lambda$ equations in $2n_p + n_\lambda$ variables, see Section 4.6.2.2. This approach is implemented in the code `ODASSL`[9] and is based on residual evaluations. It solves the system of overdetermined differential-algebraic equations of the form (4.105) in such a way, that the holonomic constraints on position level (4.34c) and on velocity level (4.35) are taken to define solution invariants to the s-index-0 of modeling level 1 formulation containing (4.34a), (4.34b), and (4.36). The subroutine `ODASSL` uses the backward differentiation formulas of orders one through five. For more details see [52, 60].

The subroutine library `MBSPACK` [164] provides a collection of numerical integration methods based on (half) explicit Runge-Kutta methods for the equations of motion of modeling level 1 (4.40) with $u(t) = t$. The numerical methods are based on the s-index-1 formulation of the equations of motion of modeling level 1 (4.40), i.e.,

$$\begin{aligned} \dot{p} &= Z(p)v, && \text{(4.126a)} \\ M(p,t)\dot{v} &= f(p,v,r,\lambda,t) - Z^T(p)G^T(p,t)\lambda, && \text{(4.126b)} \\ \dot{r} &= b(p,v,r,\lambda,t), && \text{(4.126c)} \\ 0 &= G(p,t)Z(p)v + g_{,t}(p,t), && \text{(4.126d)} \end{aligned}$$

with $G(p,t) = g_{,p}(p,t)$. The integration process is stabilized by additional projections onto the manifold of position and velocity constraints. All codes are designed for nonstiff problems and rely on structured evaluations, i.e., the system matrices are provided separately, such that the special structure of equations of motion is well exploited, but otherwise the structure of the equations of motion is strongly restricted. A modification or extension to equations of motion of modeling level 2 or 3 is difficult. Also possible solution invariants are not considered. Regarding the Algorithm Paradigm 4.5.3 only the constraints and their derivative with respect to t are respected.

The subroutine library `MBSPACK` contains the subroutines `HEDOP5` based on the half-explicit 5th order Runge-Kutta method of Arnold [6], `MDOP5` based on the explicit 5th order Runge-Kutta method of Dormand and Prince [82], `MHERK3` and `MHERK5` based on the half-explicit 3rd and 5th order Runge-Kutta method of Brasey and Hairer [23], and `PMDOP5` based on the explicit 5th order Runge-Kutta method of Dormand and Prince [82].

A similar numerical integration method is the subroutine `HEM5` [21, 22]. The numerical method bases on the the s-index-1 formulation of the equations of motion of modeling level 1 (4.126) and the discretization method is a half-explicit Runge-Kutta method of order 5(3).

In [139] the subroutine `EULAG` is presented. The method is based on the reduction of the equations of motion to a second order ODE on the solution manifold which is discretized by use of the explicit Runge-Kutta scheme `DOPRI5` [81]. The algorithm guarantees that the constraints are satisfied.

The numerical integrator `MEXAX`[10] [118] is suitable for the equations of motion of

[9] `ODASSL - available from the author, see [59]`

[10] `MEXAX - http://www.zib.de/Numerik/numsoft/CodeLib/ivpode.en.html`

modeling level 1 (4.40) and exploits its structure to a high degree. The code is based on coordinate projection and uses relatively expensive but very accurate extrapolation methods for the integration. The code MEXAX bases on structured evaluations, i.e., it needs detailed information about the system matrices and vectors in a separated form. Note that MEXAX originally was called MEXX.
Besides the mentioned numerical algorithms (mostly written in Fortran77, provided as source code and public-domain) which perform only the numerical integration of the equations of motion, there are some (mostly commercial) software packages which combine the modeling as well as the numerical simulation and animations. Among others the multibody system tools ADAMS[11] [150, 154], DYMOLA[12] [33, 131], NEWEUL[13] [98, 153, 154, 156], and SIMPACK[14] [149, 154] are software packages for the dynamic analysis of mechanical systems with the multibody system method. They comprise the computation of the symbolic equations of motion or the evaluation of the residuals of the model equations and the simulation of the dynamical behavior.
Concluding, MODELICA[15] [122], should be mentioned. MODELICA is an object-oriented modeling language designed for the modeling of complex physical systems.

[11] ADAMS - http://www.mscsoftware.com/products/products_detail.cfm?PI=413
[12] DYMOLA - http://www.dynasim.com/dymola.htm
[13] NEWEUL - http://www.mechb.uni-stuttgart.de/research/neweul/neweul_de.php
[14] SIMPACK - http://www.simpack.de/websitep.html
[15] MODELICA - http://www.modelica.org/

Chapter 5

Two New Solvers for Equations of Motion

In Section 4.7 an overview over numerical methods and numerical solvers which are adapted to the special structure of the equations of motion is given. All algorithms presented in Section 4.7 are suitable for regular equations of motion only, i.e., equations of motion of modeling level at most 3. In particular, this means that none of these algorithms is suitable for equations with redundant constraints or is capable of handling additional information on solution invariants as discussed in Section 4.1.4.
Therefore, in this chapter we will present two new methods `GEOMS` and `GMKSSOL`. While the latter code is designed for the numerical integration of the special form (5.14) of the equations of motion of modeling level 4 with possibly redundant constraints, the first code is designed for the numerical integration of general equations of motion of modeling level 4 (4.43) with possibly redundant constraints and takes into account possibly existing information concerning solution invariants. The presented codes `GEOMS` and `GMKSSOL` combine the stabilization technique developed in Section 4.6.2.3 with an implicit Runge-Kutta scheme, see Section 3.5.4, as discretization of the projected-strangeness-free formulation (4.114) of the equations of motion. Both codes are based on residual evaluations, i.e., the system matrices need not be given in explicit form. It is sufficient, that the right-hand side of (4.43) and the mass matrix M are specified.
The algorithms implemented in `GEOMS` and `GMKSSOL` are based on the 3-stage implicit Runge-Kutta Method Radau IIa of order 5. The Runge-Kutta matrix, the weight vector, and the node vector are given in Table 5.1.

$\frac{4-\sqrt{6}}{10}$	$\frac{88-7\sqrt{6}}{360}$	$\frac{296-169\sqrt{6}}{1800}$	$\frac{-2+3\sqrt{6}}{225}$
$\frac{4+\sqrt{6}}{10}$	$\frac{296+169\sqrt{6}}{1800}$	$\frac{88+7\sqrt{6}}{360}$	$\frac{-2-3\sqrt{6}}{225}$
1	$\frac{16-\sqrt{6}}{36}$	$\frac{16+\sqrt{6}}{36}$	$\frac{1}{9}$
	$\frac{16-\sqrt{6}}{36}$	$\frac{16+\sqrt{6}}{36}$	$\frac{1}{9}$

Table 5.1: Butcher tableau for 3-stage implicit Runge-Kutta method Radau IIa of order 5

As discussed in Section 3.5.4.3 and illustrated in Figure 3.2, the discretization of the projected-strangeness-free form (4.114) of the equations of motion and solution of the selected linear systems (3.123) and (3.124) in every Newton iteration step are equivalent to the discretization of the complete minimal reduced derivative array

(4.80) in combination with the solution of the unselected linear system of type 1 (3.154) and of the unselected linear system of type 2 (3.155) by use of index reducing decompositions, see Definition 3.5.72. Apart from the additional specialization of GMKSSOL to a certain type of equations of motion, the codes GEOMS and GMKSSOL are based on these two technically different approaches for the combination of regularization and discretization of the equations of motion. While in GMKSSOL the equations of motions are regularized to the projected-strangeness-free form with subsequent discretization (left branch of Figure 3.2), the code GEOMS discretizes the reduced derivative array with subsequent regularization by solving the unselected linear systems (3.154) and (3.155) via index reducing decompositions (right branch of Figure 3.2).
In Section 5.1 we will introduce the code GEOMS and we will discuss its features and its applicability in detail. Then we will present the multibody system code GMKSSOL in Section 5.2. Concluding this chapter, we will present numerical results for several examples of mechanical systems in Section 5.3.
Here and in the following we will use the typewriter style for objects which are part of the source codes of the implemented numerical algorithms. In particular, this involves names of subroutines like GEOMS, GMKSSOL, GEERREST, and variables like T, X, NWTMAT, CALSEL.

5.1 GEOMS

In this section we will present the code GEOMS and we will discuss its features in detail. For the usage and implementation of GEOMS see the manual in Appendix B.1.
As discussed in Sections 3.5 and 4.2, the initial values are restricted in their choice. In particular, they are restricted by the (hidden) constraints. On the other hand consistent initial values, in particular, consistent initial Lagrange multipliers, are not automatically given by the modeling process and their determination by solving a system of nonlinear algebraic equations is difficult for complex multibody systems with a large number of constraints. Therefore, the algorithm GEOMS provides the possibility to determine consistent initial values discussed in Section 5.1.2.
In Section 3.5.4 the discretization of general quasi-linear DAEs by use of an arbitrary Runge-Kutta method has been discussed. As already mentioned, the code GEOMS bases on the index reducing decomposition (see Definition 3.5.71) of the discretization of the minimal reduced derivative array (4.80). In Section 5.1.3 we will discuss in detail the approach which is used in the algorithm GEOMS for the discretization of the equations of motion of modeling level 4 (4.43) by use of the Runge-Kutta method of type Radau IIa of order 5. Subsequently, in Section 5.1.4 we will discuss the efficient solution of the linear systems arising in the Newton iteration inside the algorithm GEOMS. We will describe how the structure of the equations of motion may be exploited.
Further important topics for the efficiency and the robustness of an algorithm are an efficient error estimation and an appropriate step size control mechanism. For GEOMS, this will be discussed in Section 5.1.5. Furthermore, since the algorithm GEOMS is based on the combination of discretization and regularization to the projected-strangeness-free formulation of the equations of motion which is influenced by the choice of the selectors, see Section 3.5.3, an efficient choice of these selectors is also important and will be discussed in Section 5.1.6.
As mentioned above, the algorithm GEOMS is designed to handle equations of motion of modeling level 4 (4.43) with possible redundant constraints as well as solution invariants which may be provided as additional equations. The consideration of redundant constraints and of solution invariants is a very sensitive topic and will

be discussed in Sections 5.1.7 and 5.1.8.

5.1.1 Outline and features of GEOMS

Before we will discuss the features of GEOMS in detail, let us present an outline of the algorithm.

Algorithm 5.1.1 (GEOMS)

1. Initializing of variables, constants, and problem dimensions. Set T= t_0, X= x_0 and H= h.
2. Checking the consistency and correction of the initial values X= x_0, see Section 5.1.2.
3. Basic integration step.
   (a) Simplified Newton iteration, see Algorithm 5.1.4.
   (b) Error estimation and computation of a new step size HNEW, see Section 5.1.5.
   (c) Check acceptance of the integration step.
      i. If the integration step is not accepted, use the new step size H = HNEW and repeat the integration step going to 3a (in particular, goto 6 in Algorithm 5.1.4).
      ii. If the integration step is accepted, continue.
   (d) If T+H= t_f, i.e., the end of the time domain $\mathbb{I}$ is reached return to the calling program. If not, i.e., T+H< t_f, ...
      i. ... and the step size is unchanged and the convergence rate of the Newton method was high, set T := T + H and goto 3a (in particular, goto 6 in Algorithm 5.1.4).
      ii. ... and the step size has been changed or the convergence rate of the Newton method was not high, set H := HNEW, T := T + H and goto 3a (in particular, goto 1 in Algorithm 5.1.4).

As mentioned above, the algorithm GEOMS is based on residual computations. The information of the equations of motion has to be provided in the following form. The *vector of the unknown variables* has to be in the form

$$\mathtt{X}^T = \left[\begin{array}{ccc|c|c|cc} w^T & \lambda^T & \mu^T & r^T & v^T & s^T & p^T \end{array}\right]$$

and the *right-hand side* has to be specified in a user-supplied subroutine which name is given from the user. The residuals of different parts have to be given in the following order if they occur.

$$\mathtt{RDA} = \left[\begin{array}{c} d(p,v,r,w,s,\lambda,\mu,u) \\ g^{II}(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^2) \\ h^I(p,v,r,w,s,\lambda,\mu,\mathfrak{u}^1) \\ \hline g^I(p,v,s,\mathfrak{u}^1) \\ H(p,s,u)Z(p)v + h(p,s,u) \\ e(p,v,s,u) \\ \hline c(p,s,u) \\ g(p,s,u) \\ \hline\hline b(p,v,r,w,s,\lambda,\mu,u) \\ \hline f(p,v,r,w,s,\lambda,\mu,u) - Z^T(p)G^T(p,s,u)\lambda - Z^T(p)H^T(p,s,u)\mu \\ \hline Z(p)v \end{array}\right] \tag{5.1}$$

In particular, the right-hand side has to be ordered such that the upper part contains the algebraic constraints ordered with respect to their dependencies, i.e., first the constraints which restrict the additional variables w as well as the Lagrange multipliers λ and μ, secondly, the constraints on velocity level and the information concerning solution invariants which restrict the velocities v and third, the constraints on position level, which restrict the position p and the contact variables s. The specified order of the algebraic part leads to a Jacobian of the form (4.58) without respecting solution invariants and the unrestricted variables r. In particular, the obtained Jacobian has already block upper triangular structure and this will be exploited in `GEOMS`.
The second part of the right-hand side contains the differential equations also ordered in the same way as the algebraic part. We have first the equations that describe the behavior of the dynamical force elements followed by the dynamical equations of motion and, finally, the kinematical equations of motion.

Option	Name	Feature	Section
		respecting invariant solutions	5.1.8
		respecting hidden constraints	
		respecting nonholonomic constraints	
		respecting redundant constraints	5.1.7
IOPT(2)	LUN	optional output for integration information	
IOPT(3)	NIT	maximal number of Newton iterations	5.1.3.4
IOPT(4)	STARTN	starting values for the internal stages in the Newton iteration	5.1.3.2
IOPT(5)	FORM	incomplete regularization	5.1.1
IOPT(6)	NMAX	maximal number of integration steps	
IOPT(8)	PRED	step size control	5.1.5
IOPT(9)	NWTMAT	approximation of the Newton matrix at x_0 or one of the extrapolated stages possible	5.1.3.3
IOPT(10)	NWTUPD	update of the Newton matrix	5.1.3.5
IOPT(11)	DECOMPC	LU, QR, or SV decomposition for the algebraic part	5.1.4
IOPT(12)	DECOMPD	LU or QR decomposition for the differential part	5.1.4
IOPT(13)	SELCOMP	selector control	5.1.6
IOPT(14)	AUTONOM	exploitation of autonomous equations of motion	
IOPT(15)	MASSTRCT	exploitation of the structure of the mass matrix	
IOPT(16)	COMPLMDA	avoiding the computation of Lagrange multipliers	
IOPT(17)	IVCNSST	check and correction of the initial values with respect to its consistency	5.1.2

Table 5.2: Options and features of `GEOMS`

In some cases the constraints of acceleration level, i.e., $0 = g^{II}$ and $0 = h^I$, are not available. In this case the user has the possibility to use the incomplete regularization of level 1 of the equations of motion as basis for the discretization, as described in Remark 3.5.55 and Example 3.5.56. The incomplete regularization of level 1 concerning the equations of motion of modeling level 4 (4.43) is discussed in Theorem 4.6.15 and given by (4.123). This fact has to be communicated by the user to the code `GEOMS` with help of the option `IOPT(5)=FORM`. If `IOPT(5)=0` then the projected-strangeness-free form (4.114) of the equations of motion will be expected

Subroutines contained in the code GEOMS	
GEBSUBST	backward substitution of the algebraic part
GECORE	core routine
GEDECCLU	decomposition of the algebraic part FX1, FX2 and FX3 with LU decomposition
GEDECCQR	decomposition of the algebraic part FX1, FX2 and FX3 with QR decomposition
GEDECCSV	decomposition of the algebraic part FX1, FX2 and FX3 with SV decomposition
GEDECDLU	LU decomposition of the differential part, i.e., of E1 and E2
GEELIMFXQ	elimination in the differential part, i.e., of FX4, FX5 and FX6, according to QR decomposition of the algebraic part
GEELIMFXS	elimination in the differential part, i.e., of FX4, FX5 and FX6, according to SV decomposition of the algebraic part
GEELIMMIQ	elimination in the mass matrix and the identity of the kinematical equations of motion according to QR decomposition of the algebraic part
GEELIMMIS	elimination in the mass matrix and the identity of the kinematical equations of motion according to SV decomposition of the algebraic part
GEERREST	error estimation, see Section 5.10
GEFXNUM	numerical approximation of the Jacobian of the right-hand side of the equations of motion
GEGREPEQ	picking of nonpivot columns of the differential part and storing in E1 and E2 according to QR decomposition
GEGREPES	picking of nonpivot columns of the differential part and storing in E1 and E2 according to LU and SV decomposition
GEINIVAL	determination of consistent initial values, see Section 5.1.2
GEOMS	main routine
GESOLDLU	solving the differential part by use of LU decomposition
GESOLDQR	solving the differential part by use of QR decomposition
GETRFRHSC	transformation of the right-hand side according to the algebraic part
User-supplied subroutines	
EOM	provides the reduced derivative array RDA (5.1)
IVCOND	provides additional initial conditions needed for the consistent initialization, see Section 5.1.2
JAC	provides the Jacobi matrix of the reduced derivative array
MAS	provides the mass matrix
SOLOUT	output of the numerical solution and additional information during integration

Table 5.3: Subroutines of GEOMS

as basis for the discretization. Thus, the user has to specify all information of the hidden constraints up to level 2, i.e., up to acceleration level. If IOPT(5)=1, then the discretization will be based on the incomplete regularization of level 1 (4.123), such that the constraints on acceleration level $0 = g^{II}$ and $0 = h^I$ do not have to be specified. In this case the used formulation of the equations of motion has maximal constraint level $\nu_c = 1$ and is of s-index $\nu_s = 1$, i.e., it is not strangeness-free. Because of the fact that the used formulation is not strangeness-free, the success of the numerical integration highly sensitively depends on the problem and on the consistency of the initial values, in particular, on the consistency of the Lagrange multipliers λ and μ.

An overview over the features of GEOMS is given in Table 5.2. Furthermore, in Table 5.3 the subroutines belonging to GEOMS and their task is listed.

5.1.2 Determination of consistent initial values

In Sections 4.2 we have discussed the regularity of the equations of motion of multibody systems. The initial values are of great importance for the existence and the uniqueness of the solution. For the existence of a solution the consistency of the initial values is necessary, see Definition 3.1.3 and Remark 4.2.26. But in complex multibody systems, in particular, with a large number of kinematical closed loops the consistency of the initial values is not natural. While the additional variables w and the contact variables s are completely determined by the solution manifold $\mathbb{M}$ (4.65) the position variables p and the velocity variables v are restricted to the solution manifold $\mathbb{M}$, see (4.65), but some of the position and velocity variables are freely choosable. Furthermore, the variables r describing the dynamical force elements are completely freely choosable. In case of nonredundant constraints the Lagrange multipliers λ and μ are uniquely defined by the holonomic constraints on acceleration level in combination with the nonholonomic constraints on acceleration level, which are not explicitly given in the equations of motion. This uniqueness arises from the nonsingularity of the matrix in (4.49) respectively (4.56) and the Implicit Function Theorem 2.3.1.
The code GEOMS overcomes these problems by offering the possibility to determine consistent initial values.
In addition to the algebraic equations determining the solution manifold (4.65), the user has to define in a subroutine IVCOND additional conditions to determine consistent initial values. Such conditions offer the possibility to determine some of the freely choosable variables or to give further relations to other variables which allows a unique determination of consistent initial values.

Example 5.1.2 The mathematical pendulum: In Example 4.1.12 we have introduced the mathematical pendulum. The position variables p are restricted to the circle with radius L, i.e., the constraint on position level is given by $0 = p_1^2 + p_2^2 - L^2$. If one of the position variables is given, the other one is uniquely determined up to the sign.
By defining some additional conditions via the subroutine IVCOND the user can force the pendulum into a deviation of $\pi/4$ by setting $p_1 = L/\sqrt{2}$ or by $0 = p_1 + p_2$, for instance. Furthermore, a certain angular velocity ω can be prescribed by setting $0 = \sqrt{v_1^2 + v_2^2}/L - \omega$. □

The determination of consistent initial values is done in the subroutine GEINIVAL and is based on the collection of all algebraic constraints (4.43d)-(4.43g) and (4.50), (4.51), and (4.53) in connection to the conditions defined in the subroutine IVCOND. The user has to decide if the given initial values are assumed to be consistent or not. By setting IOPT(17)=IVCNSST=1, the initial values are assumed to be consistent and no check of consistency or correction of the initial values is done during the run of GEOMS. Otherwise, by setting IOPT(17)=0, the initial values are considered to be possibly inconsistent. Thus, consistency will be checked and the initial values will be corrected during the run of GEOMS, if necessary. If the user does not provide sufficiently many additional conditions, only the consistency is checked. If the initial values are consistent, then the integration will be continued, otherwise the run of GEOMS will be stopped. If the user provides more additional conditions than necessary, then the correction (if necessary) is done regarding the overdetermined nonlinear system. If all conditions together are noncontradictory, then consistent initial values will be determined. Otherwise, the Newton iteration used in this process will diverge and the run of GEOMS will be stopped.

The solution of the nonlinear system of equations is obtained via a simplified Newton method with the possibility of a certain number of updates of the iteration matrix, as described in Section 5.1.3.5. The stopping criterion is the same as that for the simplified Newton method during the integration process described in Section 5.1.3.5.

Remark 5.1.3 Note the fact that the conditions provided to `IVCOND` by the user dominate the given initial guess, i.e., if the given initial guess is consistent but does not satisfy the (possibly wrong) conditions provided to `IVCOND`, the initial guess will be corrected in such a way that both, the constraints (4.43d)-(4.43g) and (4.50), (4.51), and (4.53) and the initial conditions provided to `IVCOND` are satisfied.
In case of an initial guess which is consistent to the constraints, the option `IOPT(17)` can be set to one to avoid such a correction. Otherwise, the conditions provided to `IVCOND` should be adapted. □

If there is only interest in the computation of consistent initial values, the user has to set `T=TEND` and `IOPT(17)=0`. Then the code `GEOMS` determines consistent initial values, will call the user-supplied subroutine `SOLOUT`, and finally will return to the calling subroutine.

5.1.3 Numerical solution of the nonlinear stage equations arising from discretization methods

The presented code `GEOMS` is designed for the numerical integration of equations of motion of modeling level 4 (4.43) with possibly redundant constraints and possibly known solution invariants (4.28), as discussed in Section 4.1.4. The algorithm combines the regularization technique developed in Section 4.6.2.3 with the discretization of the obtained projected-strangeness-free form of the equations of motion (4.114) by use of the implicit Runge-Kutta method of type Radau IIa of order 5. The projected-strangeness-free formulation of the equations of motion belongs to the class of quasi-linear DAEs discussed in Section 3.5. The discretization of quasi-linear DAEs by use of implicit Runge-Kutta methods has been discussed in Section 3.5.4 and requires the solution of the nonlinear stage equation (3.113a). Following Section 3.5.4.2, the nonlinear stage equation is solved by the simplified Newton method. Regarding the simplified Newton method the following four questions are important:

1. how to get suitable starting values which are close enough to the solution to guarantee the convergence and to decrease the effort for the whole Newton process;
2. how to choose the Newton iteration matrix $\mathfrak{N}$;
3. how to implement the simplified Newton method efficiently such that as much as possible of the given structure of the problem is exploited;
4. how to stop Newton iteration process such that the computed numerical solution is close enough to the analytical solution with respect to the given tolerances.

In the following we will discuss theses four questions in detail.

5.1.3.1 Discretization scheme

As mentioned above, the numerical integration of the equations of motion of modeling level 4 (4.43) is based on the discretization of the projected-strangeness-free

form (4.114) which corresponds to a semi-implicit DAE of the form (3.106) with

$$\widetilde{E}_D(x,\widetilde{u}) = \begin{bmatrix} I_{n_p} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & M(p,u) & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_{n_r} & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\widetilde{k}_D(x,\widetilde{u}) = \begin{bmatrix} Z(p)v \\ f(p,v,r,w,s,\lambda,\mu,u) - Z^T(p)G^T(p,s,u)\lambda - Z^T(p)H^T(p,s,u)\mu \\ b(p,v,r,w,s,\lambda,\mu,u) \end{bmatrix},$$

$$\widetilde{k}_C(x,\widetilde{u}) = \begin{bmatrix} d(p,v,r,w,s,\lambda,\mu,u) \\ c(p,s,u) \\ H(p,s,u)Z(p)v + h(p,s,u) \\ g(p,s,u) \\ h^I(p,v,r,w,s,\lambda,\mu,u) \\ g^I(p,v,s,u) \\ g^{I\!I}(p,v,r,w,s,\lambda,\mu,u) \\ e(p,v,s,u) \end{bmatrix},$$

$$S_D(x,\widetilde{u}) = \begin{bmatrix} S_p(p,u) & 0 & 0 \\ 0 & S_v(p,u) & 0 \\ 0 & 0 & I_{n_r} \end{bmatrix},$$

$$S_C(x,\widetilde{u}) = \begin{bmatrix} I_{n_w} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & I_{n_s} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & S_\mu(p,u) & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & S_\lambda(p,u) & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & S_\mu(p,u) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & S_\lambda(p,u) & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & S_\lambda(p,u) & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & I_{n_e} \end{bmatrix}.$$

Note that in addition to the constraints we take into account possible information of solution invariants (4.28).
The discretization of this DAE follows the investigations in Section 3.5.4.1 by exploiting the structure of the Jacobi matrix of the right-hand side which has convenient structure similar to (4.58) after row permutations.
The discretization leads to the nonlinear stage equation (3.113a) which has to be solved by use of the simplified Newton method described in Section 3.5.4.2. This leads to the linear systems (3.128) and (3.129) which are to be solved in an efficient way by exploiting the structure as will be described in Section 5.1.4 below.

5.1.3.2 Determination of starting values

Lemmata 2.3.19 and 2.3.21 show the importance of a good choice of starting values for the Newton process. In the code `GEOMS` two different possibilities for the determination of starting values for the integration step from t_1 to $t_2 = t_1 + h_2$ are implemented. The user has to define in advance which of both shall be chosen during the integration process.
By setting `IOPT(4)=STARTN`=1 the starting values for the internal stages are chosen by $X_i^0 = x_1$, $i = 1, ..., 3$, i.e., the shifted internal stages are set to zero $Y_i^0 = 0 \in \mathbb{R}^n$, $\mathfrak{Y}^0 = 0 \in \mathbb{R}^{3n}$.
On the other hand setting `IOPT(4)`=0 (which is the default) the starting values for the Newton iteration are obtained by evaluating the interpolation polynomial q of degree s over the already passed integration interval from t_0 to $t_1 = t_0 + h_1$ with

$$q(0) = 0,\ q(c_i) = Y_i,\ i = 1, ..., 3$$

at $1 + c_i h_2/h_1$ such that we obtain the starting values for the Newton iteration as

$$Y_i^0 = q(1 + c_i h_2/h_1) + x_0 - x_1, \ i = 1, ..., 3,$$

where x_0 and x_1 denote the numerical solutions at the points t_0 and t_1, respectively. In particular, this means that the new starting values in the integration step from t_1 to $t_2 = t_1 + h_2$ are obtained by extrapolation to the points $t_1 + c_i h_2$, $i = 1, ..., 3$ based on the internal stages of the earlier integration step from t_0 to $t_1 = t_0 + h_1$. Of course, this is not possible in the first step. Here we set $Y_i^0 = 0 \in \mathbb{R}^n$, $\mathfrak{Y}^0 = 0 \in \mathbb{R}^{3n}$. For more details see [82].

5.1.3.3 Determination of the Newton iteration matrix

In Section 3.5.4.2 the numerical solution of the nonlinear stage equation (3.113a) via the Newton method is discussed. In particular, it is mentioned that a constant Newton iteration matrix $\mathfrak{N}$ during the whole or several parts of the Newton iteration process inside the current integration step from t_i to $t_{i+1} = t_i + h_i$ leads to the simplified Newton method. Obviously, the choice of a constant Newton iteration matrix reduces the amount of computation because of the saved evaluation of Jacobians and saved decompositions of the Newton iteration matrix in every except the first iteration step. But the choice of the Newton iteration matrix influences the convergence of the iteration process. For this reason, the code `GEOMS` offers the possibility to choose between several reference points (X^*, U^*), see (3.115). The kind of choice has to be determined by the user by setting the option `IOPT(9)=NWTMAT`. The range of possible choices is related to the stages during the integration step. As discussed in the previous Section 5.1.3.2, there are two possibilities for the choice of initial values for the iteration process for the determination of the internal stages. In case of `IOPT(4)`=0 the initial values are obtained by extrapolation of the so far computed solution in the points $t_i + h_i c_j$, $j = 1, 2, 3$. This offers the possibility to approximate the Newton iteration matrix at four different reference points $(X^*, U^*) = (X_j^0, u(t_i + h_i c_j))$, where $c_0 = 0$ and c_j, $j = 1, 2, 3$ correspond to the node vector of the Runge-Kutta method, see Table 3.1 or, in particular, Table 5.1. Furthermore, $X_j^0 = Y_j^0 + x_i$ corresponds to the extrapolated initial values for the internal stages at the times $t_i + h_i c_j$, $j = 0, ..., 3$, and, in particular, $X_0 = x_0$ corresponds to the initial state of the current integration interval. Note that this possibility is only given if the initial values for the Newton iteration process are extrapolated. In the case of initial values chosen such that $X_j = x_0$ for all $j = 1, 2, 3$ this possibility is not given and the Newton iteration matrix will be approximated at the initial point with the initial state of the current integration step.
Several numerical experiments have shown that the convergence of the iteration process can be improved by use of extrapolated initial values, i.e., `IOPT(4)`=0 in connection with an approximation of the Newton iteration matrix at the second internal stage, i.e., $(X^*, U^*) = (X_2^0, u(t_i + h_i c_2))$ with `IOPT(9)`=2. But, if the Newton iteration detects convergence problems, and the integration step has to be repeated with a smaller step size, the Newton iteration matrix has to be recomputed such that the overall time consume may increase if the number of times a convergence problem is detected is large. This number is reflected in the counter `NCRJCT` which corresponds to the number of step rejections caused by convergence test failures.
Furthermore, the code `GEOMS` offers the possibility of a certain number of updates of the Newton iteration matrix during the iteration process inside of one integration step, see Section 5.1.3.5.

5.1.3.4 Simplified Newton method

The simplified Newton method is implemented in the following way.

Algorithm 5.1.4 (Simplified Newton method in GEOMS) Consider the numerical solution of the nonlinear stage equation (3.113a) in the integration step from t_k to $t_{k+1} = t_k + h_k$.

1. Determination of starting values $\mathfrak{Y}_k^0$, see Section 5.1.3.2.
2. Computation of $\mathfrak{Z}_k^0 = (T^{-1} \otimes I_n)\mathfrak{Y}_k^0$.
3. Initialization of $i = 0$.
4. Determination of the Newton iteration matrix, see Section 5.1.3.3.
   (a) Computation of the mass matrix and storage in FMAS and M0.
   (b) Initialization of IKIN corresponding to the leading matrix in the kinematical equations of motion.
   (c) Computation of the (negative and partially scaled) Jacobian of the right-hand side, storage in FX1, ..., FX6.
5. Predecomposition of the Jacobian and determination of the pivoting vector PIV, see Section 5.1.4.
   (a) Decomposition of the constraint part FX1, FX2 and FX3 and determination of the pivoting vector.
   (b) Elimination in the differential part, i.e., FX4, FX5, FX6, FMAS, IKIN.
6. Decomposition of the differential part, see Section 5.1.4.
   (a) Determination of E1 und E2.
   (b) Decomposition of the differential part, i.e., of E1 and E2.
7. Internal loop of the simplified Newton iteration, see Section 5.1.3.4.
   (a) Computation of $\mathfrak{X}_k^i = \mathfrak{Y}_k^i + (\mathbb{1} \otimes x_k)$.
   (b) Computation of the right-hand side.
   (c) Transformation of the right-hand side with respect to the decompositions of FX1, ..., FX6, FMAS, IKIN, E1, and E2.
   (d) Backward substitution of the differential part.
   (e) Backward substitution of the constraint part.
   (f) Transformation of the solution with the transformation matrix P.
   (g) Update of the Newton iterate $\mathfrak{Z}_k^{i+1} = \mathfrak{Z}_k^i + \Delta\mathfrak{Z}_k^i$.
   (h) Computation of $\mathfrak{Y}_k^{i+1} = (T \otimes I_n)\mathfrak{Z}_k^{i+1}$.
   (i) Check of the convergence of the Newton method.
      i. If the maximal number of iterations is reached, i.e., $i >$ NIT = IOPT(3), then stop the Newton iteration, half the step size H := H/2 and repeat the integration step going to 1 .
      ii. If a solution is not yet found and an update of the Newton matrix is necessary and allowed, goto 4.
      iii. If a solution is not yet found and an update of the Newton matrix is not necessary, increase i by one and goto 7a.

Note that during the whole Newton iteration process the shifted stages $\mathfrak{Y}_k^i$ as well as the transformed stages $\mathfrak{Z}_k^i$ are stored in order to avoid additional effort for the transformation $\mathfrak{Z}^i = (T^{-1} \otimes I_n)\mathfrak{Y}^i$ during the evaluation of the functions.

5.1.3.5 Convergence and termination criterion of the simplified Newton process

The convergence rate of the simplified Newton method is investigated in detail in [42]. For the special case of applying the simplified Newton method to the linear systems arising in the discretization of semi-explicit DAEs or of equations of motion we have convergence results from [79, 117].
One important question in the use of an iterative method for solving linear systems inside an integration process is when to stop the iteration such that the obtained accuracy of the computed solution of the nonlinear system is within the prescribed tolerance without performing too many Newton iteration steps.
The convergence estimation and the stopping criterion implemented in GEOMS is adopted from the code RADAU5 and described in [82]. The estimation of the convergence bases on the *weighted root square norm* $||\cdot||_{sc}$ which is defined for $\zeta \in \mathbb{R}^n$ by

$$||\zeta||_{sc} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{\zeta_i}{sc_i}\right)^2} \tag{5.2}$$

with $sc_i = \texttt{ATOL}(i) + \max(|x_{0i}|, |x_{1i}|)\texttt{RTOL}(i)$, see [82]. For the sake of completeness we will review the results from [82].
If we are in the Newton iteration step k and have determined the new iterate $\mathfrak{Y}^{k+1}$, the iteration will be stopped if

$$||\mathfrak{Y}^{k+1} - \mathfrak{Y}^*||_{sc} \leq \kappa\texttt{TOL} \tag{5.3}$$

with an appropriate choice of the parameter κ, where $\mathfrak{Y}^*$ is the exact solution of the nonlinear stage equation (3.113a) and TOL the prescribed tolerance for the integration process. The question now is how to estimate $||\mathfrak{Y}^{k+1} - \mathfrak{Y}^*||_{sc}$.
Since the convergence of the simplified Newton method is linear, we have the estimate

$$||\Delta\mathfrak{Y}^{k+1}||_{sc} \leq \Theta||\Delta\mathfrak{Y}^k||_{sc},$$

where the $\Delta\mathfrak{Y}^k$ are determined in (3.117) inside the simplified Newton iteration for the solution of the nonlinear stage equation (3.113a). By applying the triangular inequality we get from

$$\mathfrak{Y}^{k+1} - \mathfrak{Y}^* = (\mathfrak{Y}^{k+1} - \mathfrak{Y}^{k+2}) + (\mathfrak{Y}^{k+2} - \mathfrak{Y}^{k+3}) + ...$$

the estimate

$$||\mathfrak{Y}^{k+1} - \mathfrak{Y}^*||_{sc} \leq \frac{\Theta}{1-\Theta}||\Delta\mathfrak{Y}^k||_{sc}. \tag{5.4}$$

We estimate the convergence rate Θ by Θ_k defined by

$$\Theta_k = ||\mathfrak{Y}^k||_{sc}/||\mathfrak{Y}^{k-1}||_{sc} \quad \text{for } k \geq 1.$$

Therefore, in view of (5.3) and (5.4) we can decide to stop the Newton iteration and to accept $\mathfrak{Y}^{k+1}$ as approximation to the exact solution $\mathfrak{Y}^*$ if

$$\eta_k||\Delta\mathfrak{Y}^k||_{sc} \leq \kappa\texttt{TOL} \quad \text{with} \quad \eta_k = \frac{\Theta_k}{1-\Theta_k}. \tag{5.5}$$

In the first Newton iteration step we set

$$\eta_0 = (\max(\eta_{old}, Uround))^{0.8}$$

where η_{old} is last η_k from the preceding integration step and *Uround* denotes the rounding unit. The norm used for the convergence estimate should be the same as for the the local error estimate of the integration process, see Section 5.1.5.
In the case of a very slow convergence or, in particular, in the case of divergence, the number of Newton iteration steps has to be restricted by a maximal number $k_{max} = \texttt{NIT}$. Thus, the Newton iteration will be stopped if
a) for some k we have that

$$\frac{\Theta_k^{k_{max}-k}}{1-\Theta_k}||\Delta\mathfrak{Y}^k||_{sc} > \kappa\texttt{TOL},$$

in this case the criterion (5.5) will probably not be satisfied within the maximal number k_{max} of allowed Newton iteration steps, or
b) there is a k with $\Theta_k \geq 1$, i.e., the iteration diverges.
If case a) applies, i.e., the Newton iteration is not fast enough to converge within k_{max} Newton iteration steps, the user has to decide whether the whole integration step has to be rejected because of convergence failures and to be repeated with a reduced step size, or if the Newton iteration should be continued with an updated Newton iteration matrix. In `GEOMS` this decision is made by defining the maximal number of updates in the option `IOPT(10)=NWTUPD`. However, several numerical results suggest that the number of allowed updates should not exceed 1, because of the fact that the amount of computation for one integration step with one allowed update of the Newton iteration matrix is about as high as the amount of computation for two integration steps with half the step size but no allowed update of the Newton iteration matrix.
It should be noted that the possibility of an update of the Newton iteration matrix within the Newton iteration process is not available in the code `RADAU5`.

5.1.4 Numerical solution of the linear systems arising from discretization methods

As we have seen in Section 3.5.4.3 the sequence of regularization and discretization in the numerical treatment of quasi-linear DAEs can be reversed, i.e., we can either first regularize the differential-algebraic equations via the projected-strangeness-free DAE (3.77) and then discretize the projected-strangeness-free DAE, or first discretize the complete minimal reduced derivative array (3.66) and uses index reducing decompositions for the solution of the linear system of type 1 (3.154) and of the linear system of type 2 (3.155) in the Newton iteration. Both approaches are equivalent, see Theorem 3.5.75 and Figure 3.2. In `GEOMS` we will follow the second approach.
for this, we have to solve two different linear problems. First, the linear problem (3.154) with $n = n_p + n_v + n_r + n_w + n_s + n_\lambda + n_\mu$ unknowns ξ and $m = m_C + m_D$ equations has to be solved. In particular, this means that we have to satisfy the algebraic part $C\xi = b_C$ consisting of $m_C = n_w + n_s + 3n_\lambda + 2n_\mu + m_e$ equations within a prescribed tolerance, where m_e is the number of solution invariants, and with respect to the differential part $m_D = n_p + n_v + n_r$ equations $(\gamma D + hB)\xi = b_D$ have to solved in a generalized way with respect to the restrictions of the algebraic part. Secondly, the linear problem (3.155) has to be solved which has the double number of equations and unknown variables.
The numerical solution of these linear problems is discussed in general in Section 3.5.4.3 without a detailed specification of the transformation matrices $\mathring{Q}_R$, Q_R, Q_I, and $\mathring{P}$.
The code `GEOMS` offers the possibility to decompose the differential part and the algebraic part via different decomposition methods. The user has to specify with

the option IOPT(11)=DECOMPC if the algebraic part should be decomposed by use of the LU decomposition with full pivoting (IOPT(11)=1), by a QR decomposition with pivoting (IOPT(11)=2), or by a SV decomposition (IOPT(11)=3). It is known that the LU decomposition is the cheapest choice with respect to the amount of computation. But, with respect to numerical stability it is not the best choice, despite full pivoting. On the other hand, the SV decomposition offers excellent stability properties but is more expensive. Nevertheless, the default decomposition is the SV decomposition unless the user specifies another decomposition. Let us note that heuristically seen, the LU decomposition with (partial) pivoting is a good compromise concerned efficiency and stability such that it is sufficient for many computations. Furthermore, with the option IOPT(12)=DECOMPD, the user can specify how to decompose the differential part. By setting IOPT(12)=0 the LU decomposition with partial pivoting is used and by setting IOPT(12)=1 the QR decomposition is used. Concerning the decomposition of the differential part also note the following detailed considerations. For strangeness-free differential-algebraic systems in semi-implicit form the scaling of the algebraic constraints with $1/h$ is recommended in [136], see Remark 3.5.61. Since the numerical integration of the equations of motion in GEOMS is based on the projected strangeness-free formulation of the equations of motion, the constraints are scaled by $1/h$. Note that the matrix in (3.154) and (3.155) has the following structure

$$C = \begin{bmatrix} F_{11} & F_{12} & F_{13} & F_{14} \\ 0 & 0 & F_{23} & F_{24} \\ 0 & 0 & 0 & F_{34} \end{bmatrix}, \tag{5.6}$$

with

$$F_{11} = \begin{bmatrix} d_{,w} & d_{,\lambda} & d_{,\mu} \\ g^{II}_{,w} & g^{II}_{,\lambda} & g^{II}_{,\mu} \\ h^{I}_{,w} & h^{I}_{,\lambda} & h^{I}_{,\mu} \end{bmatrix}, \; F_{12} = \begin{bmatrix} d_{,r} \\ g^{II}_{,r} \\ h^{I}_{,r} \end{bmatrix}, \; F_{13} = \begin{bmatrix} d_{,v} \\ g^{II}_{,v} \\ h^{I}_{,v} \end{bmatrix}, \; F_{14} = \begin{bmatrix} d_{,s} & d_{,p} \\ g^{II}_{,s} & g^{II}_{,p} \\ h^{I}_{,s} & h^{I}_{,p} \end{bmatrix},$$

$$F_{23} = \begin{bmatrix} g^{I}_{,v} \\ HZ \\ e_{,v} \end{bmatrix}, \; F_{24} = \begin{bmatrix} g^{I}_{,s} & g^{I}_{,p} \\ \breve{h}_{,s} & \breve{h}_{,p} \\ e_{,s} & e_{,p} \end{bmatrix},$$

$$F_{34} = \begin{bmatrix} c_{,s} & c_{,p} \\ g_{,s} & g_{,p} \end{bmatrix},$$

where the matrix entries are evaluated at the particular state $(X^*, \hat{U}^*)$ given by $(X^*, \hat{U}^*) = (p^*, v^*, r^*, w^*, s^*, \lambda^*, \mu^*, \mathfrak{u}^{2*})$, see (3.115). Because of this block structure, it is more efficient only to store the nonzero blocks such that the storage is done with

$$\texttt{FX1} = \begin{bmatrix} -F_{11} & -F_{12} & -F_{13} & -F_{14} \end{bmatrix}, \; \texttt{FX2} = \begin{bmatrix} -F_{23} & -F_{24} \end{bmatrix}, \; \texttt{FX3} = \begin{bmatrix} -F_{34} \end{bmatrix}.$$

In the case of regular equations of motion, we have the full (row) rank of F_{11}, F_{23}, and F_{34} from Assumptions (4.55a)-(4.55c) and (4.56). In the case of quasi-regular equations of motion we have at least the constant rank of F_{11}, F_{23}, and F_{34} from Assumption 4.2.4.
Because of the upper block triangular structure, the transformation matrix $\mathring{Q}_C$ can

be chosen as

$$\overset{*}{Q}_C = \begin{bmatrix} \overset{*}{Q}^1_{C1} & 0 & 0 \\ 0 & \overset{*}{Q}^1_{C2} & 0 \\ 0 & 0 & \overset{*}{Q}^1_{C3} \\ \overset{*}{Q}^2_{C1} & 0 & 0 \\ 0 & \overset{*}{Q}^2_{C2} & 0 \\ 0 & 0 & \overset{*}{Q}^2_{C3} \end{bmatrix}$$

where the block dimensions of $\overset{*}{Q}_C$ corresponds to the block structure of C. The particular structure of C and $\overset{*}{Q}_C$ reduces the amount of computation with respect to the decomposition of the constraint part. Furthermore, the matrix $\overset{*}{P}$ is determined in the form

$$\overset{*}{P} = \begin{bmatrix} \overset{*}{P}_{11} & 0 & 0 & \overset{*}{P}_{12} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{n_r} & 0 & 0 \\ 0 & \overset{*}{P}_{21} & 0 & 0 & 0 & \overset{*}{P}_{22} & 0 \\ 0 & 0 & \overset{*}{P}_{31} & 0 & 0 & 0 & \overset{*}{P}_{32} \end{bmatrix}.$$

The matrices $\overset{*}{Q}_C$ and $\overset{*}{P}$ are chosen such that the relations

$$\begin{aligned} \begin{bmatrix} \overset{*}{Q}^1_{C1} \\ \overset{*}{Q}^2_{C1} \end{bmatrix} F_{11} \begin{bmatrix} \overset{*}{P}_{11} & \overset{*}{P}_{12} \end{bmatrix} &= \begin{bmatrix} \overset{*}{R}_{C1} & \overset{*}{S}_{C1} \\ 0 & 0 \end{bmatrix}, \\ \begin{bmatrix} \overset{*}{Q}^1_{C2} \\ \overset{*}{Q}^2_{C2} \end{bmatrix} F_{23} \begin{bmatrix} \overset{*}{P}_{21} & \overset{*}{P}_{22} \end{bmatrix} &= \begin{bmatrix} \overset{*}{R}_{C2} & \overset{*}{S}_{C2} \\ 0 & 0 \end{bmatrix}, \\ \begin{bmatrix} \overset{*}{Q}^1_{C3} \\ \overset{*}{Q}^2_{C3} \end{bmatrix} F_{34} \begin{bmatrix} \overset{*}{P}_{31} & \overset{*}{P}_{32} \end{bmatrix} &= \begin{bmatrix} \overset{*}{R}_{C3} & \overset{*}{S}_{C3} \\ 0 & 0 \end{bmatrix}, \end{aligned}$$

are satisfied with nonsingular upper triangular matrices $\overset{*}{R}_{Ci}$, $i = 1, 2, 3$ of appropriate size. We get (3.157) with

$$\begin{aligned} \overset{*}{R}_C &= \begin{bmatrix} \overset{*}{R}_{C1} & \overset{*}{Q}^1_{C1} F_{13} \overset{*}{P}_{21} & \overset{*}{Q}^1_{C1} F_{14} \overset{*}{P}_{31} \\ 0 & \overset{*}{R}_{C2} & \overset{*}{Q}^1_{C2} F_{24} \overset{*}{P}_{31} \\ 0 & 0 & \overset{*}{R}_{C3} \end{bmatrix}, \\ \overset{*}{V}_C \check{P}_2 &= \begin{bmatrix} \overset{*}{S}_{C1} & \overset{*}{Q}^1_{C1} F_{12} & \overset{*}{Q}^1_{C1} F_{13} \overset{*}{P}_{22} & \overset{*}{Q}^1_{C1} F_{14} \overset{*}{P}_{32} \\ 0 & 0 & \overset{*}{S}_{C2} & \overset{*}{Q}^1_{C2} F_{24} \overset{*}{P}_{32} \\ 0 & 0 & 0 & \overset{*}{S}_{C3} \end{bmatrix}. \end{aligned}$$

Note that the permutations are not actually performed in `GEOMS` and the zeros in the left lower parts of $\overset{*}{R}_C$ and $\overset{*}{V}_C \check{P}_2$ are respected. Furthermore, the transformation matrices are stored in the new zero entries in the arrays `FX1`, `FX2`, and `FX3`, i.e., in the lower left part of the matrices $\overset{*}{R}_{Ci}$ for $i = 1, 2, 3$. If the LU decomposition or QR decomposition are used in the algebraic part, the transformation matrix $\overset{*}{P}$ corresponds to the pivoting and is stored in a pivoting vector `PIV`. In the case of the SV decomposition $\overset{*}{P}$ corresponds to the orthogonal matrix that is applied from the right.

Subsequently, after detecting the full rank part of the constraints, the elimination of the corresponding columns in the differential part will be performed. This will analogously be done for the Jacobian of the right-hand side, i.e., for

$$B = \begin{bmatrix} F_{41} & F_{42} & F_{43} & F_{44} \\ F_{51} & F_{52} & F_{53} & F_{54} \\ 0 & 0 & F_{63} & F_{64} \end{bmatrix}, \tag{5.7}$$

with

$$\begin{aligned}
F_{41} &= \left[\begin{array}{ccc} b_{,w} & b_{,\lambda} & b_{,\mu} \end{array}\right], \\
F_{42} &= \left[\begin{array}{c} b_{,r} \end{array}\right], \\
F_{43} &= \left[\begin{array}{c} b_{,v} \end{array}\right], \\
F_{44} &= \left[\begin{array}{cc} b_{,s} & b_{,p} \end{array}\right], \\
F_{51} &= \left[\begin{array}{ccc} f_{,w} & f_{,\lambda} - Z^T G^T & f_{,\mu} - Z^T H^T \end{array}\right], \\
F_{52} &= \left[\begin{array}{c} f_{,r} \end{array}\right], \\
F_{53} &= \left[\begin{array}{c} f_{,v} \end{array}\right], \\
F_{54} &= \left[\begin{array}{cc} (f - Z^T G^T \lambda - Z^T H^T \mu)_{,s} & (f - Z^T G^T \lambda - Z^T H^T \mu)_{,p} \end{array}\right], \\
F_{63} &= \left[\begin{array}{c} Z \end{array}\right], \\
F_{64} &= \left[\begin{array}{cc} 0 & (Zv)_{,p} \end{array}\right],
\end{aligned}$$

and with respect to the corresponding part of the leading matrix, i.e., for

$$D = \begin{bmatrix} 0 & I_{n_r} & 0 & 0 \\ 0 & 0 & M & 0 \\ 0 & 0 & 0 & [0\ I_{n_p}] \end{bmatrix}, \tag{5.8}$$

where the matrix entries in B and D are evaluated at the particular state $(X^*, \hat{U}^*) = (p^*, v^*, r^*, w^*, s^*, \lambda^*, \mu^*, \mathfrak{u}^{2*})$, see (3.115). The matrices are stored as

$$\begin{aligned}
\texttt{FX4} &= \left[\begin{array}{cccc} -F_{41} & -F_{42} & -F_{43} & -F_{44} \end{array}\right], \quad \texttt{FMAS} = \left[\begin{array}{cc} M & 0 \end{array}\right], \\
\texttt{FX5} &= \left[\begin{array}{cccc} -F_{51} & -F_{52} & -F_{53} & -F_{54} \end{array}\right], \quad \texttt{IKIN} = \left[\begin{array}{cc} 0 & I_{n_p} \end{array}\right], \\
\texttt{FX6} &= \left[\begin{array}{cc} -F_{63} & -F_{64} \end{array}\right].
\end{aligned}$$

The allocation of the zero entries in `FMAS` and `IKIN` is necessary for subsequent matrix transformations. Furthermore, we determine two transformation matrices $\overset{*}{L}_D$ and $\overset{*}{L}_B$ according to (3.159) of the form

$$\overset{*}{L}_D = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \overset{*}{L}_{D52} & \overset{*}{L}_{D53} \\ 0 & 0 & \overset{*}{L}_{D63} \end{bmatrix}$$

and

$$\overset{*}{L}_B = \begin{bmatrix} \overset{*}{L}_{B41} & \overset{*}{L}_{B42} & \overset{*}{L}_{B43} \\ \overset{*}{L}_{B51} & \overset{*}{L}_{B52} & \overset{*}{L}_{B53} \\ 0 & \overset{*}{L}_{B62} & \overset{*}{L}_{B63} \end{bmatrix}.$$

These matrices will be stored in the new zero entries in the arrays `FX4`, `FX5`, `FX6`, `FMAS`, and `IKIN` which originate from the elimination process, see (3.160a).
As mentioned in Remark 3.5.82, note that the transformations done up to this point correspond to the predecomposition process and, therefore, no information on the step size h or the Runge-Kutta coefficients is necessary for these eliminations. These eliminations are done at a very early stage of an integration step. The transformation matrices $\overset{*}{Q}_C$, $\overset{*}{P}$, $\overset{*}{L}_B$, and $\overset{*}{L}_D$ as well as the obtained matrices $\overset{*}{R}_C$, $\overset{*}{V}_C\overset{*}{P}_2$, $\overset{*}{V}_B$, and $\overset{*}{V}_D$ allow an efficient solution of the linear problems (3.154) and (3.155).
After finishing the predecomposition process, we have a decomposition of the algebraic part and we have those parts of the solution inside the differential part eliminated which are already determined by the algebraic part.
It remains the investigation of the differential part, which now depends on the step

size h and the Runge-Kutta coefficients γ, α, and β. For the investigation of the differential part, we have to decompose the linear system of the form

$$\begin{bmatrix} 0 & (\gamma \overset{*}{V}_D + h\overset{*}{V}_B)\check{P}_2 \end{bmatrix} \begin{bmatrix} \zeta_1 \\ \zeta_2 \end{bmatrix} = (\gamma \overset{*}{L}_D + h\overset{*}{L}_B)Q_C^1 b_C + b_D, \tag{5.9}$$

see (3.160a) with ζ_2 having a dimension corresponding to $(\gamma \overset{*}{V}_D + h\overset{*}{V}_B)\check{P}_2$ and $\overset{*}{V}_D$ and $\overset{*}{V}_B$ are defined in (3.160b). We are only interested in the second part ζ_2 because the first part ζ_1 then can be determined from the algebraic part. In particular, we have to determine and to decompose the matrix

$$\texttt{E1} = (\gamma \overset{*}{V}_D + h\overset{*}{V}_B)\check{P}_2 = \gamma(\overset{*}{L}_D \overset{*}{Q}_C^1 C + D)\check{P}_2 + h(\overset{*}{L}_B \overset{*}{Q}_C^1 C + B)\check{P}_2.$$

Fortunately, because of the predecomposition process, we have already computed $\overset{*}{L}_D \overset{*}{Q}_C^1 C + D$ and $\overset{*}{L}_B \overset{*}{Q}_C^1 C + B$ and have stored them in the arrays `FMAS`, `IKIN` and `FX4`, `FX5`, `FX6`, respectively. Furthermore, also the matrix $\check{P}_2$ is known from the predecomposition process such that the determination of `E1` mainly needs storage effort.

Furthermore, we have to solve the second linear system of type 2 (3.155) that consists also of the blocks C, B, and D given in (5.6), (5.7), and (5.8), respectively. As discussed in Section 3.5.4.3, the transformation matrices $\overset{*}{Q}_C^1$, $\overset{*}{P}$, $\overset{*}{L}_D$, and $\overset{*}{L}_B$ may be reused for the linear system of type 2 (3.155) such that a large part of the necessary computations already has been done. Thus, in the decomposition process of the linear system (3.155) it remains to decompose the differential part of the linear system (3.155). For this, it remains to determine and to decompose the matrix

$$\texttt{E2} = \begin{bmatrix} (\alpha \overset{*}{V}_D + h\overset{*}{V}_B)\check{P}_2 & (-\beta \overset{*}{V}_D + h\overset{*}{V}_B)\check{P}_2 \\ (\beta \overset{*}{V}_D + h\overset{*}{V}_B)\check{P}_2 & (\alpha \overset{*}{V}_D + h\overset{*}{V}_B)\check{P}_2 \end{bmatrix},$$

compare with (3.169). From the predecomposition process we already have computed $\overset{*}{V}_D$ and $\overset{*}{V}_B$ and we have stored them in the arrays `FMAS`, `IKIN`, and `FX4`, `FX5`, `FX6`, respectively, such that their determination mainly needs storage effort. Hence, the decomposition of the remaining differential part contained in `E1` and `E2` may be carried out in a straight-forward way using the LU decomposition or QR decomposition (in general, the use of the SV decomposition is possible but it is not implemented).

After the transformation of the right-hand side $\begin{bmatrix} b_C^T & b_D^T \end{bmatrix}^T$ according to (3.167a) and $\begin{bmatrix} b_{1C}^T & b_{1D}^T & b_{2C}^T & b_{2D}^T \end{bmatrix}^T$ according to (3.170a) we get the solution of the linear systems (3.154) and (3.155) via backward substitution.

Note that we offer two alternatives for the decomposition of the differential part of the linear systems (3.154) and (3.155). First, the solution may obtained by use of the LU decomposition of `E1` and `E2` with `IOPT(12)=DECOMPD`=0 and on the other hand the solution may obtained by use of the QR decomposition of `E1` and `E2` with `IOPT(12)=DECOMPD`=1.

Remark 5.1.5 The described approach has the advantage that the predecomposition process can be done independently of the step size h or the parameters γ, α, or β, for both linear systems (3.154) and (3.155). Only the decomposition of the differential part has to be done separately using h, γ, α, and β. In particular, if the Newton iteration process has convergence problems and the algorithm interrupts the Newton process for another try with a reduced step size, then the information of the predecomposition process may be recycled which saves computational work. In the case of the use of the LU decomposition for the differential part, it is possible to consider the linear system (3.155) as a system of half the dimensions in $\mathbb{C}$ according to [82]. □

Remark 5.1.6 For the linear algebra computations like QR decompositions and SV decompositions we use BLAS[1] (Basic Linear Algebra Subprograms) [111] and LAPACK[2] (Linear Algebra PACKage) [2] subroutines. □

Remark 5.1.7 In the case of the numerical integration of regular equations of motion, see Definition 4.2.8, the linear systems (3.154) and (3.155) are uniquely solvable. □

5.1.5 Error estimation and step size control

The step size control of the integration process is a very sensitive topic in the implementation of numerical algorithms for the integration of ODEs as well as for DAEs. An overview over several step size control strategies is given in [169], see also [25, 39, 63, 82]. The code GEOMS works with two different step size control strategies as used in the code RADAU5 adapted to the structure of the equations of motion.
The base for a control mechanism of the step size is a local error estimation. According to [82] we use in GEOMS

$$(\gamma\hat{E}(X^*,\hat{U}^*) - h\hat{k}_{,x}(X^*,\hat{U}^*))err = h\hat{k}(x_0,\hat{u}(t_0)) + \hat{E}(x_0,\hat{u}(t_0))\sum_{i=1}^{3} e_i Y_i \quad (5.10a)$$

with

$$(e_1,e_2,e_3) = (-13-7\sqrt{6},-13+7\sqrt{6},-1)/(3\gamma) \quad (5.10b)$$

for the determination of the local error err. Following the investigations in Section 3.5.4.3 for the numerical integration of the projected-strangeness-free formulation (4.114) of the equations of motion (4.43) we get the unselected linear system in the form (3.154) with (5.6), (5.7), (5.8), and with $\xi = err$ and

$$\begin{aligned} b_C &= 0, \\ b_D &= h \begin{bmatrix} b \\ f - Z^T G^T \lambda - Z^T H^T \mu \\ Zp \end{bmatrix} \\ &\quad + \gamma \begin{bmatrix} 0 & I_{n_r} & 0 & 0 \\ 0 & 0 & M & 0 \\ 0 & 0 & 0 & \begin{bmatrix} 0 & I_{n_p} \end{bmatrix} \end{bmatrix} \sum_{i=1}^{3} e_i Y_i, \end{aligned}$$

where the entries are evaluated at the particular state $(X^*,\hat{U}^*)$ given by $(X^*,\hat{U}^*) = (p^*,v^*,r^*,w^*,s^*,\lambda^*,\mu^*,\mathfrak{u}^{2*})$, see (3.115) and which has to be solved in an index reducing sense as discussed in Sections 3.5.4.3 and 5.1.4. The right-hand side b_C of the algebraic part can be assumed to be zero, since it describes the algebraic restrictions and all of them are satisfied by all stages X_i because of the properties of Radau IIa methods applied to semi-implicit DAEs (3.24), see Section 5.1.3.1.
Since the solution of the linear system (5.10) is based on the same approach as discussed in Section 5.1.4, from the simplified Newton iteration we have already all information of the decomposition of the leading matrix in (5.10) which corresponds to the leading matrix in (3.123), see Section 5.1.4. Only the transformation of the right-hand side and the backward substitution has to be performed. The error estimation is implemented in the subroutine GEERREST.

[1] BLAS - http://www.netlib.org/blas/
[2] LAPACK - http://www.netlib.org/lapack/

For the choice of a new step size for the next integration step or a repeated integration step there are two possibilities implemented in GEOMS which have to be selected by use of the option IOPT(8)=PRED. First, with IOPT(8)=2 the *classical step size controller* developed in [63] uses the step size strategy

$$h_{new} = fac\ h_k\ ||err_{k+1}||_{sc}^{-1/4} \tag{5.11}$$

with err_{k+1} determined by (5.10) in the current integration step from t_k to $t_{k+1} = t_k + h_k$, and the weighted root square norm $||\cdot||_{sc}$ defined in (5.2) with $sc_i = \texttt{ATOL}(i) + \max(|x_{0i}|, |x_{1i}|)\texttt{RTOL}(i)$, see [82]. The safety factor fac is proposed to depend on NEWT, the number of Newton iterations of the current step and on the maximal allowed number of Newton iterations NIT. We use $fac = \texttt{SAFE}(2\texttt{NIT} + 1)/(2\texttt{NIT} + \texttt{NEWT})$, where SAFE is a safety factor with default value SAFE= 0.9, see [82]. Secondly, if IOPT(8)=1, then the step size selection is based on a *predictive step size controller*, developed by Gustafsson in [78]. There, it has been proposed to determine the new step size by

$$h_{new} = fac\ h_k \left(\frac{1}{||err_{k+1}||_{sc}}\right)^{1/4} \frac{h_k}{h_{k-1}} \left(\frac{||err_k||_{sc}}{||err_{k+1}||_{sc}}\right)^{1/4}, \tag{5.12}$$

see [82]. Obviously, this predictive step size control (5.12) is not possible in the first step. Therefore, the classical step size controller (5.11) will be used in the first integration step. The predictive controller (5.12) needs slightly more work and storage than the classical controller but is more flexible in the adaptation of the step size. By use of the controller (5.12) a faster reduction of the step size without step rejections is possible than by use of the classical controller (5.11). This leads to a possible reduction of the overall amount of computation. It is our experience that the predictive step size controller (5.12) seems to produce safer results for simple problems, on the other hand, the choice of the classical controller (5.11) produces often slightly faster runs, see also [82].
In GEOMS the predictive step size controller (5.12) will be used by default if the user does not specify anything else.

5.1.6 Selector control

In general, it is not necessary to newly compute selectors in every integration step. Rather, it is possible to keep the selectors constant for a certain number of integration steps.
In case of the QR decomposition for the differential part, the whole selector in matrix form has to be stored. Then in every integration step the differential part has to be premultiplied with the stored selectors and after that the decomposition has to be performed. At least, this procedure is inefficient for systems with a small number of constraints. If the LU decomposition is used, there is the possibility derived from the pivoting to store the selectors in a pivoting vector only. Let us illustrate this strategy for the decomposition of the differential part of the linear system (3.154), i.e., we have to solve the linear system (5.9) with $\texttt{E1} \in \mathbb{R}^{m_D, n-r_C}$, $m_D \geq n - r_C$. We have the LU decomposition with pivoting in the form

$$L_{\texttt{E1}} P_{\texttt{E1}} \texttt{E1} \begin{bmatrix} \overset{*}{P}_2 & \overset{*}{P}_3 \end{bmatrix} = \begin{bmatrix} \overset{*}{R}_R & \overset{*}{V}_R \\ 0 & 0 \end{bmatrix},$$

where $\overset{*}{P} = \begin{bmatrix} \overset{*}{P}_1 & \overset{*}{P}_2 & \overset{*}{P}_3 \end{bmatrix} \in \mathbb{R}^{n,n}$ is nonsingular and $\overset{*}{P}_1 \in \mathbb{R}^{n,r_C}$ is known from (3.156), $L_{\texttt{E1}} \in \mathbb{R}^{m_D, m_D}$ is a lower triangular matrix with ones on the diagonal, $\overset{*}{R}_R \in \mathbb{R}^{r_D, r_D}$ is a upper triangular matrix, and

$$P_{\texttt{E1}} = \begin{bmatrix} P^1_{\texttt{E1}} \\ P^2_{\texttt{E1}} \end{bmatrix} \in \mathbb{R}^{m_D, m_D}$$

with $P^1_{\texttt{E1}} \in \mathbb{R}^{r_D,m_D}$ and $P^2_{\texttt{E1}} \in \mathbb{R}^{m_D-r_D,m_D}$ is the pivoting matrix which performs the selection of the differential part according to the regularization. Here, the matrix $L_{\texttt{E1}}P_{\texttt{E1}}$ takes the place of Q_D in (3.161). In particular, we have

$$P_{\texttt{E1}}\texttt{E1} = \begin{bmatrix} \widetilde{E}_1 \\ \widetilde{E}_2 \end{bmatrix},$$

where $\widetilde{E}_1 \in \mathbb{R}^{r_D,n-r_C}$ has full rank, i.e., $\operatorname{rank}(\widetilde{E}_1) = r_D$. Assuming that all input functions are continuous in t then E1 as a function of t is continuous, too, as long as the pivoting matrix P is constant, see Sections 3.5.4.3 and 5.1.4. Therefore, we have that $P^1_{\texttt{E1}}\texttt{E1}(t+\Delta t)$ has full rank for all $\Delta t \in (-\epsilon,\epsilon)$ with sufficiently small ϵ, see Lemma 2.1.3, such that it is possible to replace the linear system (5.9) by the selected linear system

$$P^1_{\texttt{E1}} \begin{bmatrix} 0 & \overset{*}{V}_D\check{P}_2 \end{bmatrix} \zeta = P^1_{\texttt{E1}}((\gamma \overset{*}{\dot{L}}_D + h\overset{*}{L}_B)\overset{*}{Q}{}^1_C b_C + b_D), \tag{5.13}$$

as long as $P^1_{\texttt{E1}}\texttt{E1}(t+\Delta t)$ has full rank and the pivoting matrix P_1 of the constraints remains unchanged. In particular, this means that a selector recomputation in GEOMS will only be done if the column pivoting P_1 with respect to the algebraic constraints does change from one integration step to the next or if convergence problems are observed inside the Newton iteration process. This fact is demonstrated in Example 5.3.1 below in Tables 5.8 and 5.9.
If the LU decomposition is used for the decomposition of the differential part, with the option INFO(13)=SELCOMP, then it is possible to decide whether the determination of the selectors is done in each integration step (INFO(13)=1) or by following the strategy described above (INFO(13)=1), where the latter case is default if the user does not specify anything else.

5.1.7 Consideration for redundant constraints

The code GEOMS offers the possibility to integrate the equations of motion of modeling level 4 (4.43) with possibly redundant constraints. As discussed in Section 4.2 the solution is not unique in the case of redundant constraints, but under certain conditions the nonuniqueness is only restricted to the Lagrange multipliers λ, μ, and w, see Theorem 4.2.32.
Very important for the integration of equations of motion with redundant constraints is the detection of the degree of redundancy, i.e., the determination of the rank of the Jacobians of the constraints. Therefore, the code GEOMS offers the possibility to decompose the constraints via the LU decomposition, the QR decomposition or the SV decomposition using the option IOPT(11)=DECOMPC. The reliable numerical determination of the rank of a matrix is a delicate task and the SV decomposition is a commonly used tool to do this. Therefore, the numerical integration of equations of motion with redundant constraints is only allowed via the SV decomposition for the constraints.
The rank of the constraints will be determined in every integration step. If it is detected in the first step that the constraints are redundant, a reliable numerical integration requires the use of the SV decomposition at least for the decomposition of the constraints. If the LU decomposition or QR decomposition have been chosen for the decomposition of the constraints, the integration will be stopped with the indication that the SV decomposition should be used and the integration should be restarted.
Furthermore, if a change of the rank from one step to another is detected, then the integration has reached a singular point and the integration will be stopped with an error message. See for example the numerical results of the slider crank, Example 5.3.4, in particular, Figures 5.24 and 5.25.

5.1.8 Consideration for invariants

In Section 4.1.4 we have discussed the existence of solution invariants and the preservation of these invariants in the numerical solution. Furthermore, in Section 4.5 we have emphasized the necessity of the explicit occurrence of such solution invariants in the model equations, i.e., the solution invariants should be explicitly contained as equations in the equations of motion.
For this reason, `GEOMS` provides the possibility to introduce such equations (4.28) which define the solution invariants in the right-hand side (5.1) of the equations of motion.

5.2 GMKSSOL

In this section we will present the code `GMKSSOL` in detail. For the usage and implementation of `GMKSSOL` see the manual in Appendix B.2.
The code `GMKSSOL` has been developed as part of an industrial project and therefore, it is designed for a special type of equations of motion, namely

$$\begin{aligned} \dot{p} &= v, && \text{(5.14a)} \\ \dot{v} &= f_d(p, v, r, \lambda, t), && \text{(5.14b)} \\ \dot{r} &= b(p, v, r, \lambda, t), && \text{(5.14c)} \\ 0 &= g(p, t) && \text{(5.14d)} \end{aligned}$$

with $f_d(p, v, r, \lambda, t) = M^{-1}(p, t)(f(p, v, r, \lambda, t) - G^T(p, t)\lambda)$ and $G(p, t) = \partial g(p, t)/\partial p$.
The software package `GMKSSOL` has been proposed in [50] and is based on residual evaluations. The residual of the right-hand side of the equations of motion is provided from a multibody system formalism such that the mass matrix M and the constraint matrix G are not provided separately. `GMKSSOL` integrates the equations of motion (5.14) with nonredundant or uniformly redundant constraints (5.14d) such that it is not necessary that the constraint matrix has full rank, but the constraint matrix must have constant rank for the whole integration interval $\mathbb{I} = [t_0, t_f]$.
The discretization method implemented in `GMKSSOL` is based on the projected-strangeness-free form (4.114) adapted to (5.14) and uses a modified version of the subroutine `RADAU5` for the numerical integration of the regularized equations of motion.

5.2.1 Features of GMKSSOL

As mentioned above, the code `GMKSSOL` has been created in an industrial project. Therefore, it is a very specific solver which has been adapted to the precise requirements of the industrial partner. In particular, the degree of flexibility is reduced in favor of the efficiency. The features and options of the code `GMKSSOL` are listed in Table 5.4. Furthermore, the subroutines which are part of the code are listed in Table 5.5. It should be mentioned that the subroutines `DEC`, `DECC`, `SOL`, and `SOLC` are part of the code `RADAU5` [79, 82].
As mentioned above, the algorithm `GMKSSOL` is based on residual computations. The information of the equations of motion has to be provided in the following form.
The *vector of the unknown variables* has to be provided in the form

$$\mathtt{X}^T = \begin{bmatrix} p^T & v^T & r^T & \lambda^T \end{bmatrix}$$

and the *right-hand side* has to be specified in the user-supplied subroutine `EQUOFMOT`

Option	Feature	Section
	respecting hidden constraints	5.2.3
	respecting redundant constraints	5.2.3
	check of the consistency of the initial values	5.2.2
	correction of the initial Lagrange multipliers with respect to its consistency	5.2.2
	special treatment of forced mechanical systems	5.2.4
IOPT(2)	optional output for integration information	
IOPT(3)	maximal number of integration steps	
IOPT(9)	choice of selector	5.2.3
IOPT(14)	selector control	5.2.3

Table 5.4: Options and features of GMKSSOL

Subroutines contained in the code GMKSSOL	
GMKSCOIN	determination of consistent initial values, see Section 5.2.2
GMKSSOL	main routine
GMRHSIRK	providing the right-hand side of the projected-strangeness-free form of the equations of motion, see Section 5.2.3
GMSELECT	determination of the selectors, see Section 5.2.3
GMSOLIRK	core routine
GODESOLIM	numerical integration of stiff ODEs, see Section 5.2.4
DEC	decomposition of the leading matrix respecting the decoupled linear subsystem (3.123)
DECC	decomposition of the leading matrix respecting the decoupled linear subsystem (3.124)
SOL	backward substitution respecting the decoupled linear subsystem (3.123)
SOLC	backward substitution respecting the decoupled linear subsystem (3.124)
user-supplied subroutines	
EQUOFMOT	provides the reduced derivative array RDA (5.15)
SOLOUT	output of the numerical solution and additional information during integration

Table 5.5: Subroutines of GMKSSOL

in the following order.

$$\texttt{RDA} = \begin{bmatrix} v \\ f_d(p,v,r,\lambda,t) \\ b(p,v,r,\lambda,t) \\ g(p,t) \\ g^I(p,v,t) \\ g^{II}(p,v,r,\lambda,t) \end{bmatrix} \tag{5.15}$$

Depending on the internal option IOPT(31), it is either necessary to provide the whole right-hand side of the equations of motion or it is sufficient to provide only the holonomic constraints, see Section 5.2.3.
For the output of the numerical solution the user has to define the points where an output is expected in the array TOUT in an increasing sequence. If the code GMKSSOL passes an output point, say TOUT(i), then the code will interpolate the numerical solution at TOUT(i) and will store the numerical solution in the i-th column of

the array `XOUT`. Furthermore, after passing such output points the user-supplied subroutine `SOLOUT` which has to be provided from the user will be called.
Furthermore, the code will give some information about the success or the failure of the numerical integration. The user has to decide if this information shall be written to an output or not. By setting the option `IOPT(2)` to zero, the output of this information will be dropped. Otherwise, this information will be written to the output device defined by `IOPT(2)`, for more detail see the manual in Appendix B.2.

5.2.2 Determination of consistent initial values

In Section 4.2 we have discussed the regularity of the equations of motion of multibody systems. For the existence of a solution the consistency of the initial values, see Definition 3.1.3, is necessary. But in complex multibody systems, in particular, with a large number of kinematical closed loops, the consistency of the initial values is not natural. In particular, the position variables p and the velocity variables v are restricted to the solution manifold $\mathbb{M}$, see (4.65), but some components in the position and velocity variables are freely choosable. Furthermore, the variables r describing dynamical force elements are completely freely choosable and at least in the case of nonredundant constraints the Lagrange multipliers λ are fixed by the constraints on acceleration level which are not explicitly given in the equations of motion.
Before the integration will be started, the code `GMKSSOL` checks the consistency of the initial values. It is necessary that the user provides consistent initial values at least for the position variables $p(t_0)$ and for the velocity variables $v(t_0)$. If the initial positions or the initial velocities are not consistent the code will stop with the error message `IERR`=-1006. The initial values for the dynamical force elements $r(t_0)$ are not restricted in their choice and the user has to provide them. Furthermore, the code `GMKSSOL` offers the possibility to determine consistent initial values for the Lagrange multipliers $\lambda(t_0)$. If inconsistencies in the initial Lagrange multipliers are detected, the subroutine `GMKSCOIN` determines consistent initial Lagrange multipliers.

5.2.3 Regularization of the equations of motion and selector control

According to Theorem 4.6.9 and Lemma 4.6.11 the projected-strangeness-free form of the equations of motion (5.14) with selected constraints is given by

$$\begin{aligned}
S_p(p,t)\dot{p} &= S_p(p,t)v, && \text{(5.16a)}\\
S_p(p,t)\dot{v} &= S_p(p,t)f_d(p,v,r,\lambda,t), && \text{(5.16b)}\\
\dot{r} &= b(p,v,r,\lambda,t), && \text{(5.16c)}\\
0 &= S_\lambda(p,t)g(p,t), && \text{(5.16d)}\\
0 &= S_\lambda(p,t)g^I(p,v,t), && \text{(5.16e)}\\
0 &= S_\lambda(p,t)g^{II}(p,v,r,\lambda,t), && \text{(5.16f)}
\end{aligned}$$

with S_p of dimension $n_p - r_G \times n_p$ such that the matrix

$$\begin{bmatrix} S_p(p,t) \\ G(p,t) \end{bmatrix}$$

has full (column) rank n_p for all $(p,t) \in \mathbb{M}$. Note that the selector S_p equals the selector S_v, in accordance to the Lemma 4.6.11. According to Algorithm 4.6.12 the subroutine `GMSELECT` determines the selector $S_p(p,t)$ at the current state (p,t) by

use of the SV decomposition [72]. The SV decomposition of the constraint matrix function $G(p,t)$ yields

$$\begin{bmatrix} U_1^T \\ U_2^T \end{bmatrix} G \begin{bmatrix} V_1 & V_2 \end{bmatrix} = \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix}. \tag{5.17}$$

The columns of the matrix V_2 span $\ker(G)$ and satisfy the criterion for K_p in Algorithm 4.6.12 Therefore, we have $K_p = V_2$. Furthermore, the identity $I = V_2^T V_2$ $(= V_2^T K_p)$ holds and the selector S_p can be chosen as

$$S_p = V_2^T. \tag{5.18}$$

Moreover, from the singular value decomposition (5.17) we get the selector

$$S_\lambda = U_1^T \tag{5.19}$$

such that

$$S_\lambda G = U_1^T G = \begin{bmatrix} \Sigma & 0 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix} = \Sigma V_1^T \in \mathbb{R}^{r_G, n_{\mathbf{P}}}$$

has full rank. If the subroutine `GMSELECT` detects a change in the rank of the constraint matrix G the integration will be stopped with the error code `IERR`=-1303.

Certain motions of mechanical systems do not require a state or time depending selector, see Example 4.6.14. Therefore, by setting the option `IOPT( 9)`=1 the user has the possibility to tell the code that the selectors have to be determined at the initial point only and to keep them constant for the rest of the integration procedure. If this is possible, the amount of computation of the integration is reduced.

However, the code `GMKSSOL` is not able to decide how long the selectors can be kept constant or when a recomputation is necessary. Therefore, the user has to tell the code after how many successful steps the selectors have to be recomputed. This number has to be set in the option `IOPT(14)`. If `IOPT(14)`=0 the selectors are recomputed after every successful integration step. Several numerical experiments suggest that `IOPT(14)`=10 is a good choice, see [50].

Obviously, the determination of the selectors needs additional effort. For keeping this additional cost low, the internal option `IOPT(31)` tells the user-supplied routine if the whole reduced derivative array has to be provided (`IOPT(31)`=0) or if only the computation of the residual of the constraints is necessary (`IOPT(31)`=1) which is sufficient for the determination of the selectors.

During the integration process, the subroutine `GMRHSIRK` provides the right-hand side of the regularized equations of motion (5.16) obtained by scaling of the right-hand side of (5.14a) and (5.14b) with the selector S_p and by scaling of the holonomic constraints (5.14d) and its derivatives g^I and G^{II} with the selector S_λ. The right-hand side of (5.14) is provided from the user-supplied subroutine `EQUOFMOT` as part of the array `RDA` (5.15).

5.2.4 Discretization and numerical integration

Depending on the number and redundancies of the constraints, the numerical integration will be done in three different ways which also depend on the degrees of freedom of the mechanical system, see Definition 4.1.7 and Lemma 4.2.16. If the mechanical system is unconstrained, i.e., if we have for the degrees of freedom that $n_f = n_{f_p} = n_{f_v} = n_p$, then the equations of motion correspond to an ODE such that the integration will be done by use of the subroutine `GODESOLIM` which is suited for stiff ODEs and is an adaption of the code `RADAU5` [79, 82]. For more details on the

code GODESOLIM see [50]. If the mechanical system is a forced mechanical system, i.e., we have for the degrees of freedom $n_f = n_{f_p} = n_{f_v} = 0$, then the equations of motion correspond to a system of algebraic equations such that at every time the state is completely determined by the holonomic constraints on position level, on velocity level, and on acceleration level. Therefore, only a nonlinear system consisting of these constraints has to be solved at every output point TOUT(i). This will be done by the subroutine GMKSCOIN. In the other case, i.e., if we have for the degrees of freedom $0 < n_f = n_{f_p} = n_{f_v} < n_p$, the numerical integration of the equations of motion is based on the discretization of the projected-strangeness-free form of the equations of motion and will be done in the subroutine GMSOLIRK. The subroutine GMSOLIRK is an adaption of the code RADAU5 [79, 82].
The discretization and the solution of the arising nonlinear and linear algebraic systems in every integration step bases on the investigations of Section 3.5.4 respecting that the leading matrix of the considered DAE is constant for the integration step and has the form

$$\hat{E} = \begin{bmatrix} S_p & 0 & 0 & 0 \\ 0 & S_p & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

The right-hand side of the projected-strangeness-free form of the equations of motion is determined from the subroutine GMRHSIRK and provided for the numerical integration process in GMSOLIRK.
If the maximal number of successful integration steps for which the selectors are kept constant (defined by IOPT(14)) is reached then the subroutine GMSOLIRK interrupts the integration process and returns to the subroutine GMKSSOL for the recomputation of the selectors by use of GMSELECT. After the recomputation, the integration will be continued with the call of GMSOLIRK.

5.3 Numerical experiments

In the following we will demonstrate the applicability and the performance of the new solvers GEOMS and GMKSSOL in comparison to other well known and widely used solvers. The integration with GMKSSOL will be based on the projected-strangeness-free formulation (4.114) of the equations of motion. The corresponding numerical results are abbreviated by GMKSSOL(psfEoM). Furthermore, the integration with GEOMS will be performed on three different formulations. First, the numerical results obtained with GEOMS using the projected-strangeness-free form (4.114) of the equations of motion will be abbreviated by GEOMS(psfEoM). Secondly, the numerical results obtained with GEOMS using the projected-s-index-1 form (4.123) of the equations of motion will be abbreviated by GEOMS(pEoM1). Furthermore, if the solution of the considered example satisfies some solution invariants, e.g., the conservation of the total energy, we will use in addition to the two formulations above the projected-strangeness-free form of the equations of motion with explicite forcing of the solution invariants as described in Section 5.1.8. The obtained numerical results are denoted by GEOMS(psfEoM+I).
In Section 4.7 it is pointed out that the numerical algorithms suited for the integration of the equations of motion should be partitioned into two classes, the class of those algorithms based on residual evaluations and the class of algorithms based on structural evaluations. Therefore, the numerical results of GEOMS and GMKSSOL (which are based on residual evaluations) will be compared to the numerical solutions of both classes separately. First we compare the numerical results of GMKSSOL

numerical algorithm	used formulation	abbreviation of the numerical results
algorithms based on residual evaluations		
DASSL	s-index-0 form (4.82)	DASSL(EoM0)
GEOMS	projected-strangeness-free form (4.114)	GEOMS(psfEoM)
GEOMS	projected-strangeness-free form (4.114) with solution invariants	GEOMS(psfEoM+I)
GEOMS	projected-s-index-1 form (4.123)	GEOMS(pEoM1)
GMKSSOL	projected-strangeness-free form (4.114)	GMKSSOL(psfEoM)
RADAU5	original equations of motion (4.43)	RADAU5(EoM)
RADAU5	s-index-1 form (4.81)	RADAU5(EoM1)
RADAU5	s-index-0 form (4.82)	RADAU5(EoM0)
RADAU5	Gear-Gupta-Leimkuhler form (4.103)	RADAU5(GGL)
ODASSL	overdetermined form (4.105)	ODASSL(oEoM)
algorithms based on structural evaluations		
MEXAX	s-index-1 form (4.81)	MEXAX
HEDOP5	s-index-1 form (4.81)	HEDOP5
MHERK3	s-index-1 form (4.81)	MHERK3
MHERK5	s-index-1 form (4.81)	MHERK5

Table 5.6: Used numerical algorithms and used formulations of the equations of motion

and GEOMS with the numerical results obtained from numerical algorithms based on residual evaluations, i.e., RADAU5 (version from April 14, 2000) [79, 82], ODASSL (version from January 03, 1990) [59, 60], and by use of DASSL (version from June 24, 1991) [25, 135]. As representatives for the algorithms based on structural evaluations we will compare the numerical results obtained with GEOMS and GMKSSOL with the results obtained by use of MEXAX (version from July 18, 1996) [118], HEDOP5 (version from April 09, 1996) [6], MHERK3 (version from February 01, 1994), and MHERK5 (version from February 01, 1994). In Table 5.6 it is listed which numerical algorithm in combination with which formulation of the equations of motion are used and who they numerical results are abbreviated in the following. The numerical integrations are done on an AMD Athlon XP 1800+, 1533 MHz.
Let us note that we will abstain from the use of physical units like meters or seconds.

Example 5.3.1 The mathematical pendulum: In Example 4.1.12 we introduced the equations of motion of the mathematical pendulum and in Example 4.6.14 we have regularized the equations of motion which are used for the numerical integration via GEOMS and GMKSSOL.

mass	$m = 1.0$
length	$L = 1.0$
gravitational acceleration	$g = 13.75$

Table 5.7: Mathematical pendulum: Parameters

For the numerical simulations of the movement we used the parameter listed in Table 5.7. Let us note that we did modify the gravitational acceleration to approximately $g = 13.75$ such that the exact solution has a period of 2 which allows the comparison of the accuracy every period.
Let us first consider the conservation of the total energy. In Section 4.1.4 invari-

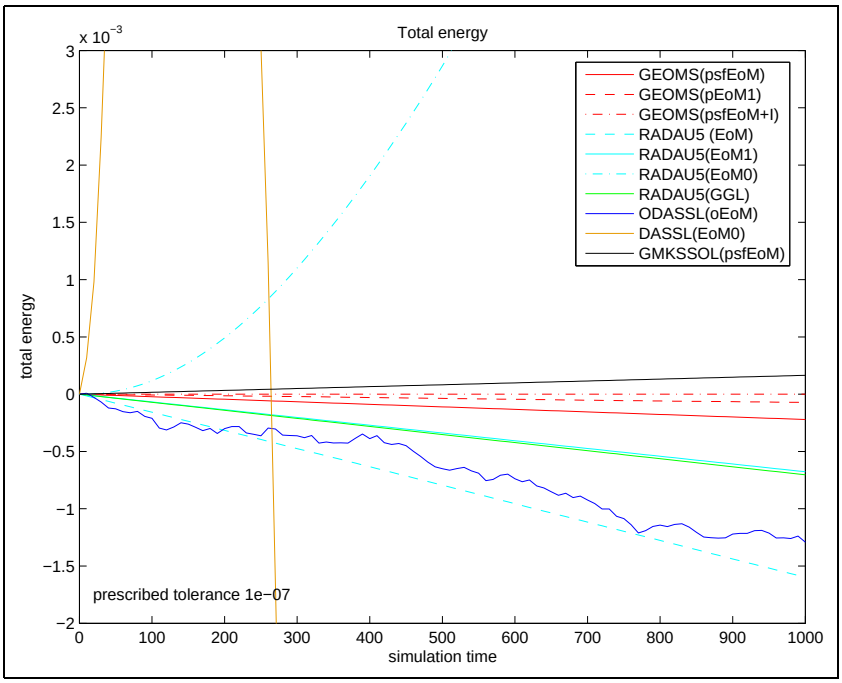

Figure 5.1: Mathematical Pendulum: Conservation of the total energy by the numerical solutions for prescribed RTOL=ATOL=10^{-7} on the time domain $\mathbb{I} = [0, 1000]$

ants of the motion of mechanical systems are considered. In particular, in Example 4.1.18 it is demonstrated that the conservation of the total energy is in general not satisfied by the numerical solution of the equations of motion, see Figure 4.9. Let us repeat the computations with the codes GEOMS and GMKSSOL. Figure 5.1 depicts again the total energy of the numerical solutions of the equations of motion computed with the algorithms based on residual evaluations. Obviously, only the numerical solution GEOMS(psfEoM+I) obtained with GEOMS conserves the total energy exactly. This is expected because the energy conservation is contained as an equation in the used formulation and is therefore explicitely forced during the numerical integration. Apart from the numerical results GEOMS(psfEoM+I) only the numerical results GEOMS(pEoM1), GEOMS(psfEoM), and GMKSSOL(psfEoM) satisfy the conservation of total energy very accurately.

In the Figures 5.2 and 5.3 the efficiency, i.e., the relation between the obtained accuracy and the consumed computation time of the different algorithms is depicted. Obviously, the integration with use of GEOMS based on the projected-strangeness-free formulation of the equations of motion plus solution invariants GEOMS(psfEoM+I) offers the best performance for this example. This fact substantiates the demand of the Paradigms 4.5.1 and 4.5.3. Furthermore, the numerical results RADAU5(EoM1), GEOMS(psfEoM), GMKSSOL(psfEoM), and RADAU5(GGL) are obtained in an efficient way. The accuracy of the numerical results obtained via ODASSL and DASSL is restricted to RTOL = ATOL = 10^{-3} and 10^{-1}, respectively. While the integration with ODASSL fails up to a prescribed tolerance of RTOL = ATOL = 10^{-10} an integration with DASSL over the time domain $\mathbb{I} = [0, 1000]$ was possible but the obtained accuracy is not acceptable. Note that the approximation of the Lagrange multipliers by GEOMS(psfEOM+I) is much better than of the other results.

A comparison of the numerical results obtained with GEOMS and GMKSSOL and the results obtained with algorithms based on structural evaluations is depicted in the

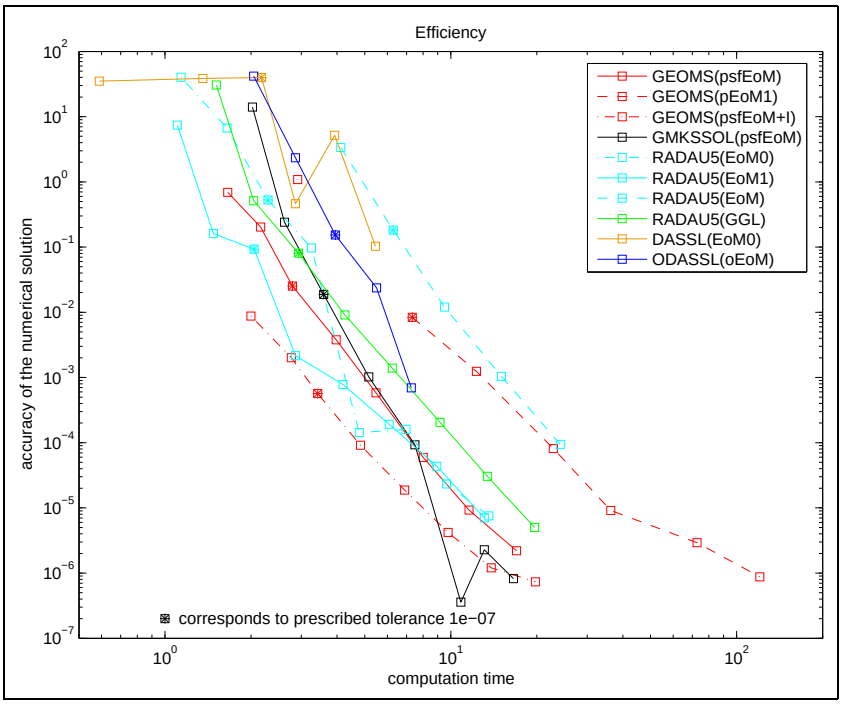

Figure 5.2: Mathematical Pendulum: Efficiency of the solvers based on residual evaluations. Simulations are done on the time domain $\mathbb{I} = [0, 1000]$.

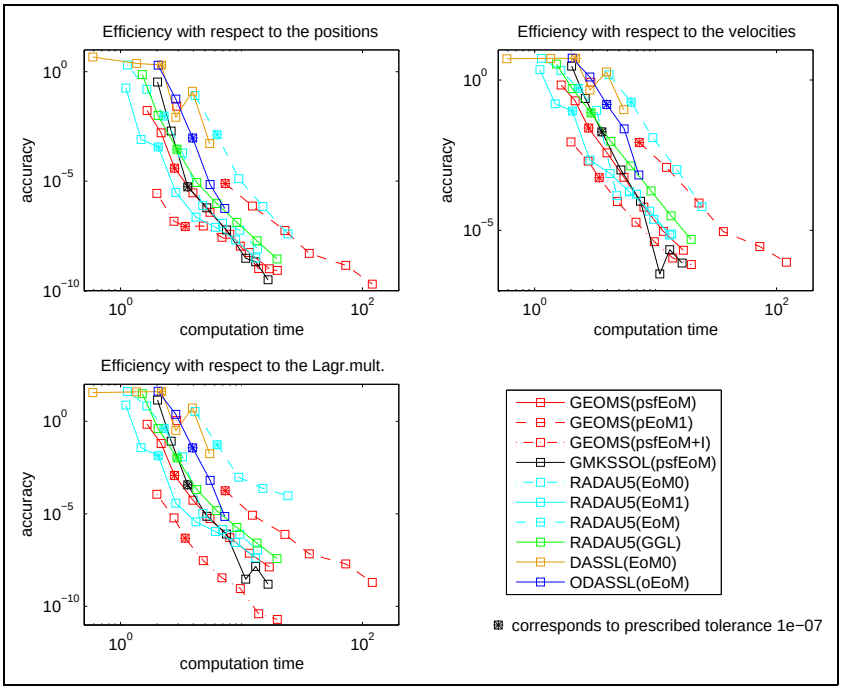

Figure 5.3: Mathematical Pendulum: Efficiency of the solvers based on residual evaluations. Simulations are done on the time domain $\mathbb{I} = [0, 1000]$.

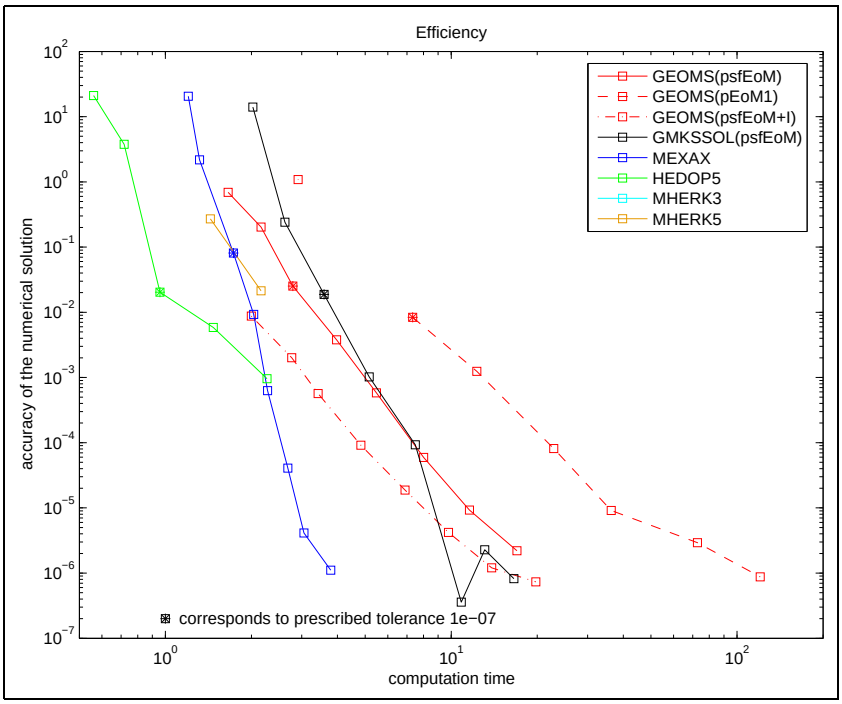

Figure 5.4: Mathematical Pendulum: Efficiency of the solvers based on structural evaluations. Simulations are done on the time domain $\mathbb{I} = [0, 1000]$.

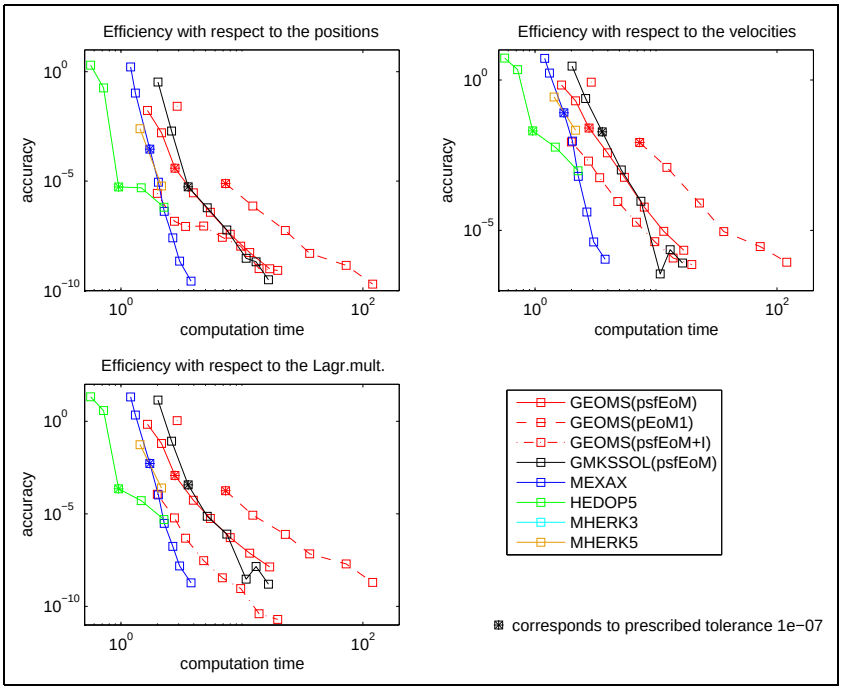

Figure 5.5: Mathematical Pendulum: Efficiency of the solvers based on structural evaluations. Simulations are done on the time domain $\mathbb{I} = [0, 1000]$.

Figures 5.4 and 5.5. Obviously the performance of the algorithms MEXAX and HEDOP5 is great and explained by the high level of exploitation of the structure of the equations of motion. Unfortunately, an integration with HEDOP5 was only possible for a prescribed tolerance RTOL=ATOL$\leq 10^{-9}$. A successful integration by use of MHERK3 was not possible at all and the integration with MHERK5 was only possible for a prescribed tolerance RTOL=ATOL$\leq 10^{-6}$.

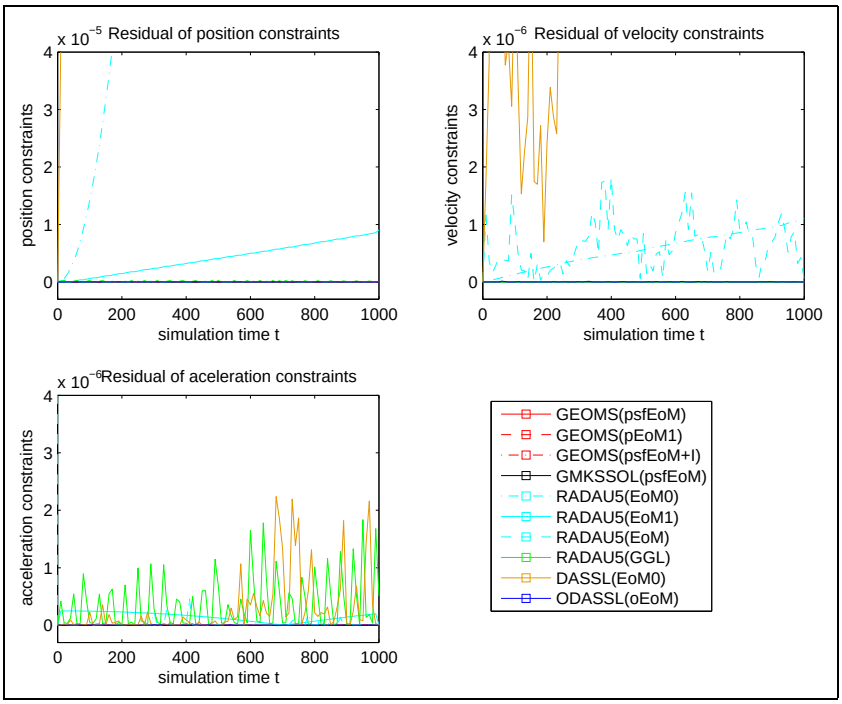

Figure 5.6: Mathematical Pendulum: Residual of the constraints depending on $t \in [0, 1000]$ for prescribed RTOL=ATOL=10^{-7}

A very important fact for the numerical integration and the stability of the numerical algorithms regarding the integration of DAEs is the satisfaction of the constraints, including the hidden constraints. In Figure 5.6 the residual of the constraints of position level, of velocity level, and of acceleration level depending on the simulation time is depicted. Obviously, the residual of the constraints on position as well as on velocity level grows very rapidly when using the s-index-0 formulation of the equations of motion, see the numerical results DASSL(EoM0) and RADAU5(EoM0) in Figure 5.6. The results RADAU5(EoM0) show a quadratic growth of the residual of the position constraints while the residual of the velocity constraints grows only linearly. Furthermore, the numerical results obtained by use of the s-index-1 formulation, i.e., the numerical results RADAU5(EoM1), show a linear growing behavior in the residuals of the constraints on position level. This illustrates the results of Section 4.4, see Table 4.2. In Figure 5.7 the maximal obtained residual of the constraints depending on the prescribed tolerances by use of the numerical algorithms based on residual evaluations is illustrated. As one can see, the satisfaction of every kind of constraint is only insured by use of GEOMS. Furthermore, in comparison the residuals of the constraints obtained by the integration by use of the numerical algorithms based on structural evaluations are depicted in Figure 5.8. While the constraints on velocity level are satisfied very accurately for the numerical solutions MEXAX and HEDOP5, the residual of the constraints on position level are accept-

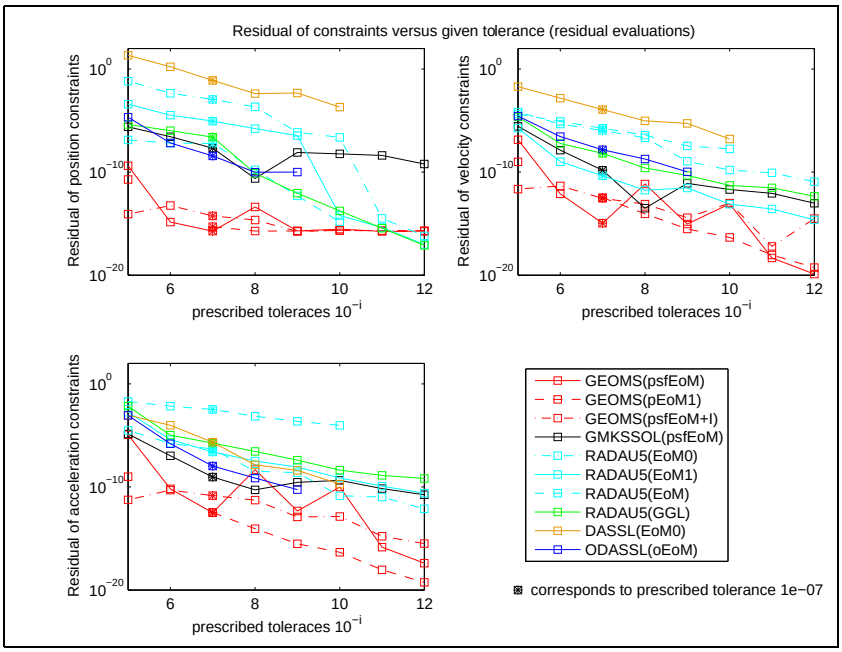

Figure 5.7: Mathematical Pendulum: Residual of the constraints depending on the prescribed tolerance. Simulations are done on the time domain $\mathbb{I} = [0, 1000]$ with solvers based on residual evaluations.

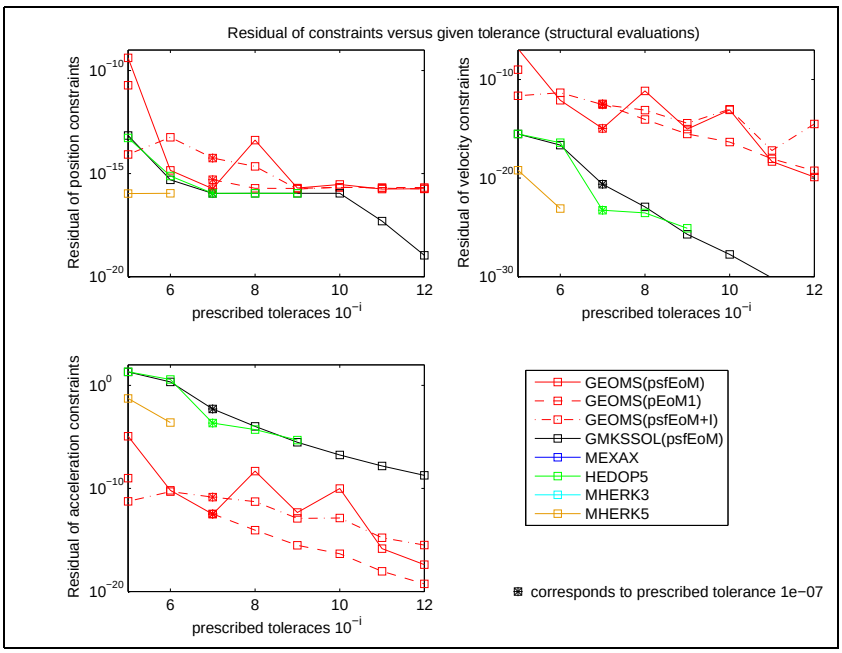

Figure 5.8: Mathematical Pendulum: Residual of the constraints depending on the prescribed tolerance. Simulations are done on the time domain $\mathbb{I} = [0, 1000]$ with solvers based on structural evaluations.

able but the satisfaction of the constraints on acceleration level is not given caused from the slight instabilities because of the used s-index-1 formulation which are not

strangeness-free. In comparison, every kind of constraints is satisfied according to the prescribed tolerances by use of GEOMS, since all constraints appear explicitly in the used formulation.

```
Example 01_SimpPend
 Integration with GEOMS(psfEoM)

 TSTART =  0.00     TEND   =  5.00     H0 = 0.100E-01
 TOLMIN =  1.0D- 7  TOLMAX =  1.0D- 9
 Initial velocity   2.80 rad

[SimpPend] GEOMS(psfEoM) starts   with IDID=    0 H= 0.100E-01 TOL = 0.100E-06
[SimpPend] GEOMS(psfEoM) finished with IDID=    0 H= 0.254E-01 at T= 0.500E+01
   NACCPT =    187    | NEOM=   2167 | NPDEC =    187
   NERJCT =     16    | NJAC=    187 | NEDEC =    204
   NCRJCT =      1    | NMAS=      1 | NBSUB =    660
   CPUTIME=   0.060s  |              | NSEL  =      2

[SimpPend] GEOMS(psfEoM) starts   with IDID=    0 H= 0.100E-01 TOL = 0.100E-07
[SimpPend] GEOMS(psfEoM) finished with IDID=    0 H= 0.176E-01 at T= 0.500E+01
   NACCPT =    270    | NEOM=   2961 | NPDEC =    270
   NERJCT =     13    | NJAC=    270 | NEDEC =    284
   NCRJCT =      1    | NMAS=      1 | NBSUB =    897
   CPUTIME=   0.060s  |              | NSEL  =      2

[SimpPend] GEOMS(psfEoM) starts   with IDID=    0 H= 0.100E-01 TOL = 0.100E-08
[SimpPend] GEOMS(psfEoM) finished with IDID=    0 H= 0.118E-01 at T= 0.500E+01
   NACCPT =    391    | NEOM=   4141 | NPDEC =    391
   NERJCT =      9    | NJAC=    391 | NEDEC =    401
   NCRJCT =      1    | NMAS=      1 | NBSUB =   1250
   CPUTIME=   0.080s  |              | NSEL  =      2
```

Table 5.8: Mathematical Pendulum: Statistical results for the numerical simulation with GEOMS using the psfEoM with initial velocity $v_{10} = 2.8$

In Section 5.1.6 the strategy for the determination of appropriate selectors has been discussed. Furthermore, the projected-strangeness-free formulation of the equations of motion of the pendulum have been developed in Example 4.6.14 and the choice of the selectors S_p and S_v has been considered. There, it is mentioned that in principle the selectors may be kept constant as long as the deviation of the pendulum does not reach 90 degrees with respect to the initial state. But with respect to the conditioning of the linear systems which have to be solved inside the Newton iteration process, the selectors should be recomputed early enough and not just before reaching a deviation of 90 degrees. This fact is treated in GEOMS by the recomputation of the selector if the column pivoting with respect to the algebraic constraints changes or convergence problems of the Newton iteration process occur. This is demonstrated in two simulation scenarios which are depicted in Tables 5.8 and 5.9.
Both scenarios simulate the motion of the pendulum starting with the downward hanging initial position $p_0 = \begin{bmatrix} 0 & -1 \end{bmatrix}^T$ and an initial velocity $v_0 = \begin{bmatrix} v_{10} & 0 \end{bmatrix}^T$ over the time domain $\mathbb{I} = [0, 5]$. In Table 5.8 the simulation starts with an initial velocity of $v_{10} = 2.8$. This initial velocity leads to the highest deviation of $p = \begin{bmatrix} \pm 0.699 & -0.715 \end{bmatrix}$ which does not reach the deviation of 45 degrees. Because of the constraint matrix $G = \begin{bmatrix} 2p_1 & 2p_2 \end{bmatrix}$ we have $|2p_1| < |2p_2|$ for all $t \in \mathbb{I}$ and a change of the pivoting is not necessary such that a (re-)computation of the selector is only necessary at the beginning of the integration process and after every detected convergence failure. Therefore, the number NSEL of (re-)computations of the selector equals the number NCRJCT of rejections because of convergence failures

```
Example 01_SimpPend
 Integration with GEOMS(psfEoM)

 TSTART =  0.00     TEND   =  5.00     H0 = 0.100E-01
 TOLMIN =  1.0D- 7  TOLMAX =  1.0D- 9
 Initial velocity   2.90 rad

[SimpPend] GEOMS(psfEoM) starts   with IDID=    0 H= 0.100E-01 TOL = 0.100E-06
[SimpPend] GEOMS(psfEoM) finished with IDID=    0 H= 0.236E-01 at T= 0.500E+01
   NACCPT =    207    | NEOM=   2730 | NPDEC =    207
   NERJCT =      7    | NJAC=    207 | NEDEC =    227
   NCRJCT =     13    | NMAS=      1 | NBSUB =    841
   CPUTIME=   0.060s  |              | NSEL  =     26

[SimpPend] GEOMS(psfEoM) starts   with IDID=    0 H= 0.100E-01 TOL = 0.100E-07
[SimpPend] GEOMS(psfEoM) finished with IDID=    0 H= 0.121E-02 at T= 0.500E+01
   NACCPT =    303    | NEOM=   3729 | NPDEC =    303
   NERJCT =     12    | NJAC=    303 | NEDEC =    321
   NCRJCT =      6    | NMAS=      1 | NBSUB =   1142
   CPUTIME=   0.080s  |              | NSEL  =     19

[SimpPend] GEOMS(psfEoM) starts   with IDID=    0 H= 0.100E-01 TOL = 0.100E-08
[SimpPend] GEOMS(psfEoM) finished with IDID=    0 H= 0.109E-01 at T= 0.500E+01
   NACCPT =    441    | NEOM=   5208 | NPDEC =    441
   NERJCT =     12    | NJAC=    441 | NEDEC =    459
   NCRJCT =      6    | NMAS=      1 | NBSUB =   1589
   CPUTIME=   0.090s  |              | NSEL  =     19
```

Table 5.9: Mathematical Pendulum: Statistical results for the numerical simulation with GEOMS using the psfEoM with initial velocity $v_{10} = 2.9$

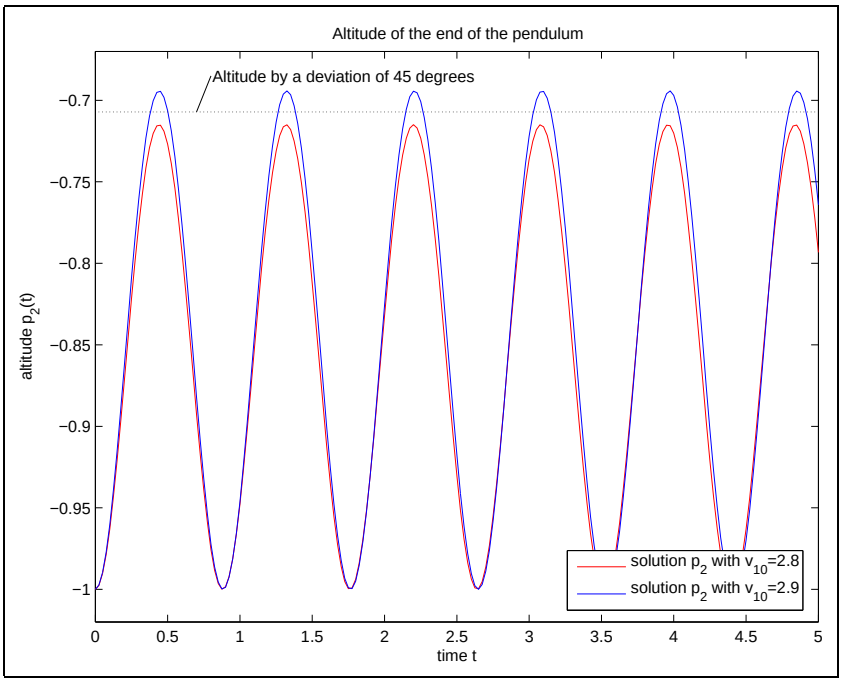

Figure 5.9: Mathematical Pendulum: Solution for p_2 for initial velocity $v_{10} = 2.8$ and $v_{10} = 2.9$ on the time domain $\mathbb{I} = [0, 5]$.

plus one initial computation. The situation changes completely if the pendulum passes the deviation of 45 degrees with respect to the initial state. This happens if the initial velocity is increased to $v_{10} = 2.9$. The numerical results are depicted in Table 5.9. Obviously, the (re-)computations of the selector `NSEL` happened 13 times more often than convergence problems `NCRJCT` are detected. In Figure 5.9 the motion of the pendulum is depicted. One can see that the altitude of the pendulum passes 12 times the altitude of a deviation of 45 degrees. Therefore, the number `NSEL` of (re-)computations of the selectors is 13, i.e., 12 times plus one initial time, more often than the number `NCRJCT` of convergence problems.

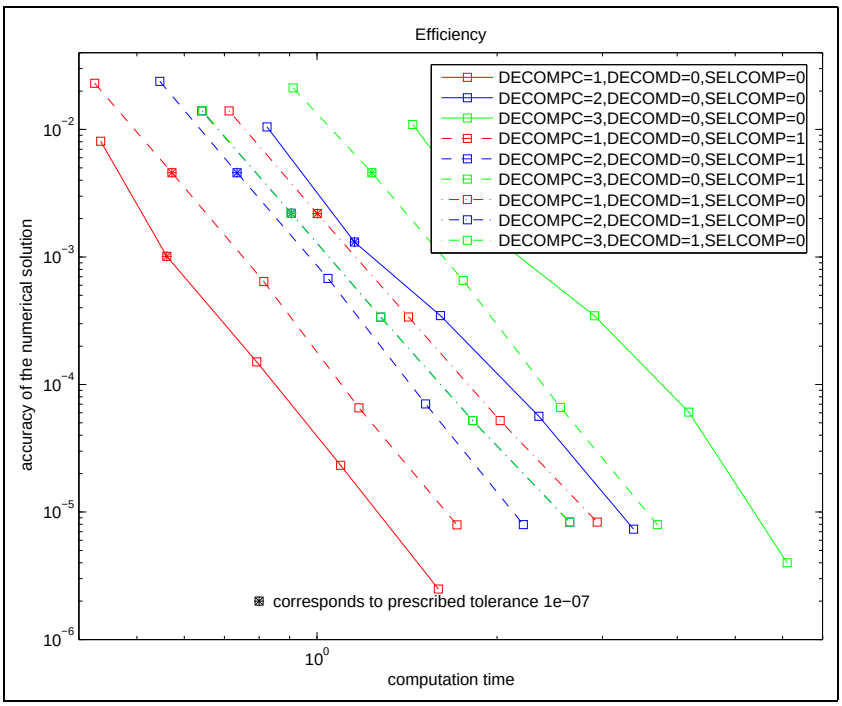

Figure 5.10: Mathematical Pendulum: Efficiency of the solver `GEOMS` by use of different decompositions. Simulations are done on the time domain $\mathbb{I} = [0, 200]$.

In Section 5.1.4 we have discussed the numerical solution of the linear systems (3.154) and (3.155) in every Newton iteration step. Several decompositions have been presented. By use of the option `IOPT(11)=DECOMPC` the user has to choose if the algebraic part shall be decomposed by the LU decomposition (`DECOMPC`=1) with full pivoting, by the QR decomposition (`DECOMPC`=2) with column pivoting, or by the SV decomposition (`DECOMPC`=3), and by use of the option `IOPT(12)=DECOMPD` the user has to choose if the differential part shall be decomposed by the LU decomposition (`DECOMPD`=0) with partial pivoting or by the QR decomposition (`DECOMPD`=1). It is clear that the amount of computation depends strongly on the choice of the decomposition and also the obtained accuracy depends on the decomposition. The efficiency of the several choices of decompositions is illustrated in Figure 5.10. Furthermore, the precision and the amount of computation depend on the number of (re-)computations of the selector which can be influenced by setting the option `IOPT(13)=SELCOMP`. Obviously, the use of the LU decomposition for the algebraic as well as for the differential part and adapted selector control strategy the integration is most efficient for this example. □

Example 5.3.2 The lolly: In Example 4.1.13 we introduced the equations of

motion of the horizontal pendulum on a surface.

mass	$m = 1.0$
gravitational acceleration	$g = 9.81$
surface parameter	β =variable

Table 5.10: Lolly: Parameters

For the numerical integrations we use the parameters as shown in Table 5.10.

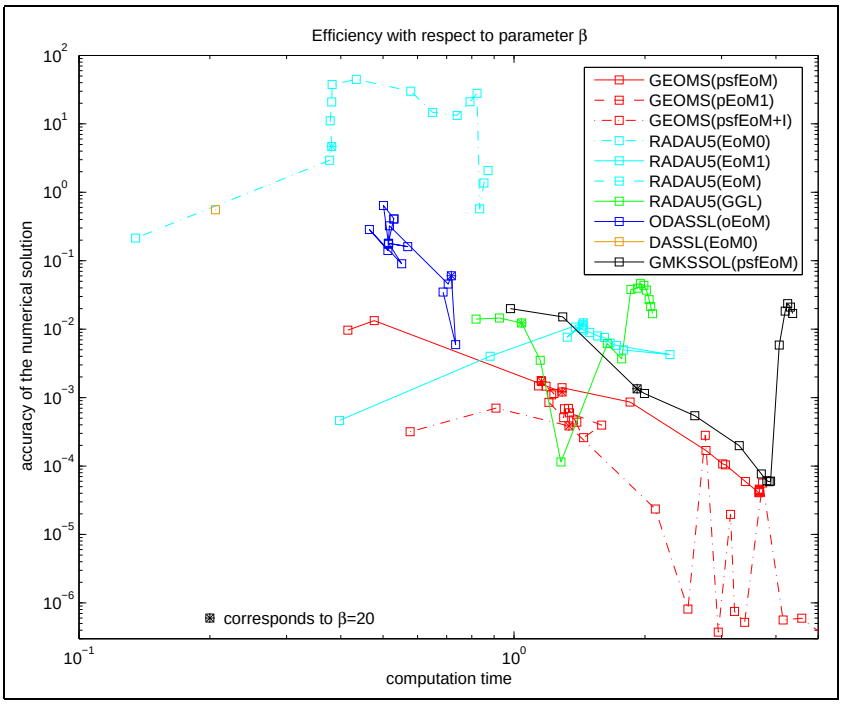

Figure 5.11: Lolly: Efficiency of the solvers based on residual evaluations. Simulations are done with varying $\beta \in [0, 140]$ and prescribed tolerance RTOL=ATOL=10^{-8} on the time domain $\mathbb{I} = [0, 100]$.

Note the influence of the parameter β in the second constraint equation (4.22c) which forces the end of the pendulum to remain on the surface. We made several numerical simulations with different parameters $\beta = 1, ..., 140$. Increasing the parameter β leads to stability problems for several formulations of the equations of motion. The s-index-0 formulation (4.82) which does not contain the explicit information about the surface is unsuitable for the numerical integration if the parameter β increases. This is demonstrated in Figures 5.11 and 5.12. There, we simulated the motion of the lolly for a time domain of $\mathbb{I} = [0, 10]$ for several parameters β. In Figure 5.11 the obtained accuracy of the numerical solution is depicted over the consumed computation time. Obviously, the efficiency of the numerical integration DASSL(EoM0) and RADAU5(EoM0) depends very strongly on the parameter β. While the integration for RADAU5(EoM0) are successful for all prescribed β but becomes inefficient with increasing β, the numerical integration DASSL(EoM0) fails already for small β. This indicates that the use of the s-index-0 formulation (4.82) of the equations of motion is not suitable as basis for the numerical integration. Furthermore, the numerical integration RADAU5(EoM) was not successful for any

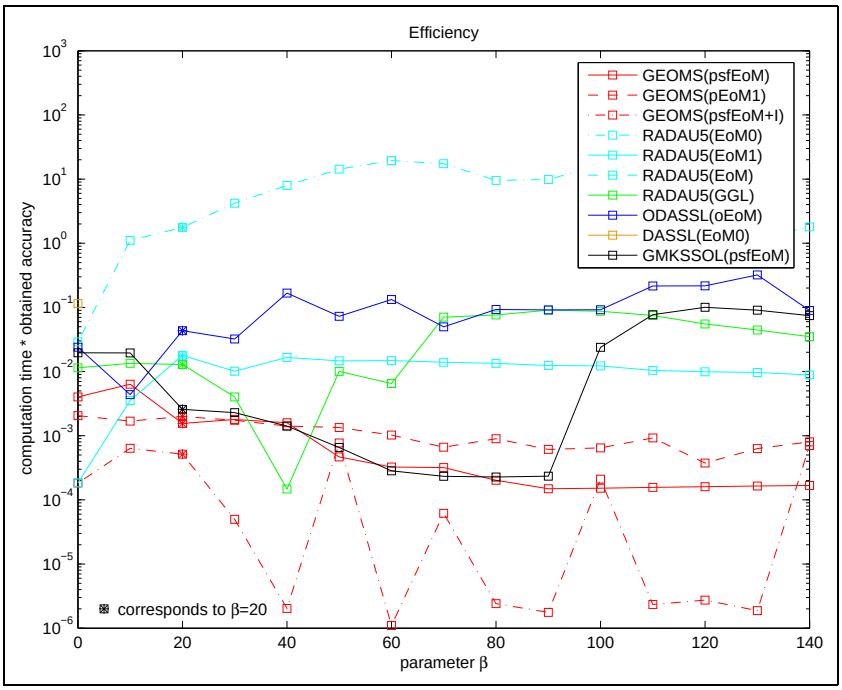

Figure 5.12: Lolly: Efficiency of the solvers based on residual evaluations. Simulations are done with varying $\beta \in [0, 140]$ and prescribed tolerance RTOL=ATOL=10^{-8} on the time domain $\mathbb{I} = [0, 100]$.

choice of the parameter β. Therefore, the corresponding results do not occur in Figures 5.11 and 5.12.

In Figures 5.13-5.15 the efficiency of the solvers based on residual evaluations for the numerical simulation with the parameter $\beta = 4, 6$, and 10 on the time domain $\mathbb{I} = [0, 10]$ is depicted. The numerical integrations are done with a prescribed tolerance RTOL=ATOL=10^i, $i = -4, ..., -12$. Obviously, for a parameter $\beta = 4$ the efficiency for obtaining the results DASSL(EoM0), GEOMS(psfEoM+I), GEOMS(psfEoM), and RADAU5(EoM1) is great, but the efficiency obtaining the results ODASSL(oEoM), RADAU5(GGL), GMKSSOL(psfEoM), and GEOMS(pEoM1) is not inferior, see Figure 5.13. The numerical results RADAU5(EoM0) and RADAU5(EoM) in principle are not acceptable since the best obtained accuracy is 10^{-4} and 10^{-2}, respectively.

For an increased parameter $\beta = 6$, see Figure 5.14, the situation of the efficiency by use of DASSL completely changes such that only GEOMS(psfEoM+I), GEOMS(psfEoM), and RADAU5(EoM1) offer an excellent performance in its computation, see Figure 5.14. The performance of the solvers obtaining the numerical results in principle is unchanged except that of DASSL(EoM0). Even if the integration is successful for all prescribed tolerances the obtained efficiency is unfavorable. Again increasing the parameter β to $\beta = 10$ yields a disastrous efficiency behavior of DASSL(EoM0), see Figure 5.15. Furthermore, while the efficiency for obtaining the other results is approximately unchanged the performance of the obtaining of RADAU5(EoM1) is slightly impaired.

In Figure 5.16 the efficiency of the solvers based on residual evaluations and based on structural evaluations for the numerical simulation with the parameter $\beta = 6$ on the time domain $\mathbb{I} = [0, 10]$ is depicted. Obviously, again the efficiency of HEDOP5 is excellent closely followed from the efficiency obtaining the results

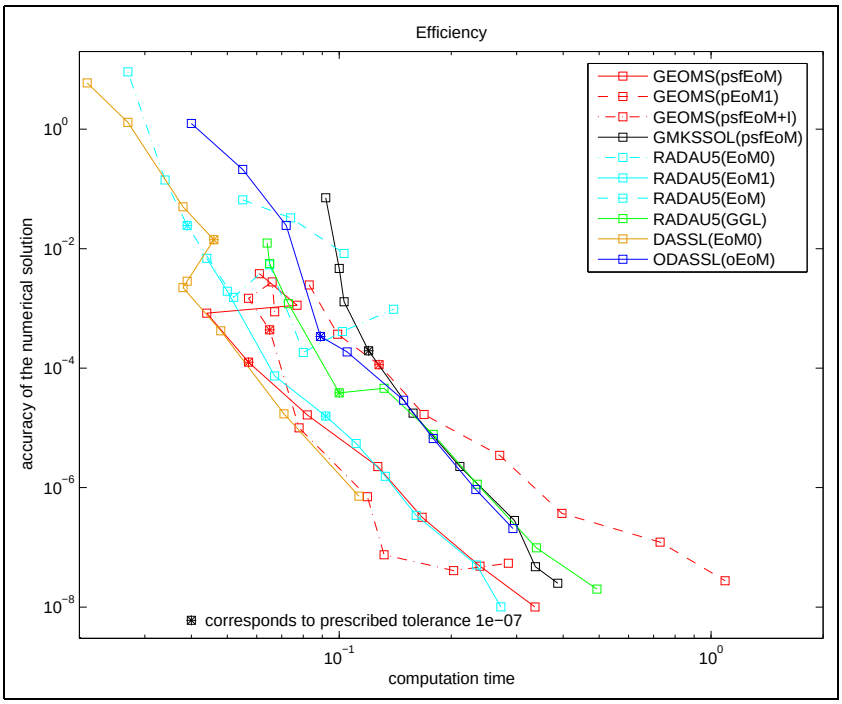

Figure 5.13: Lolly: Efficiency of the solvers based on residual evaluations. Simulations are done with $\beta = 4$ on the time domain $\mathbb{I} = [0, 10]$.

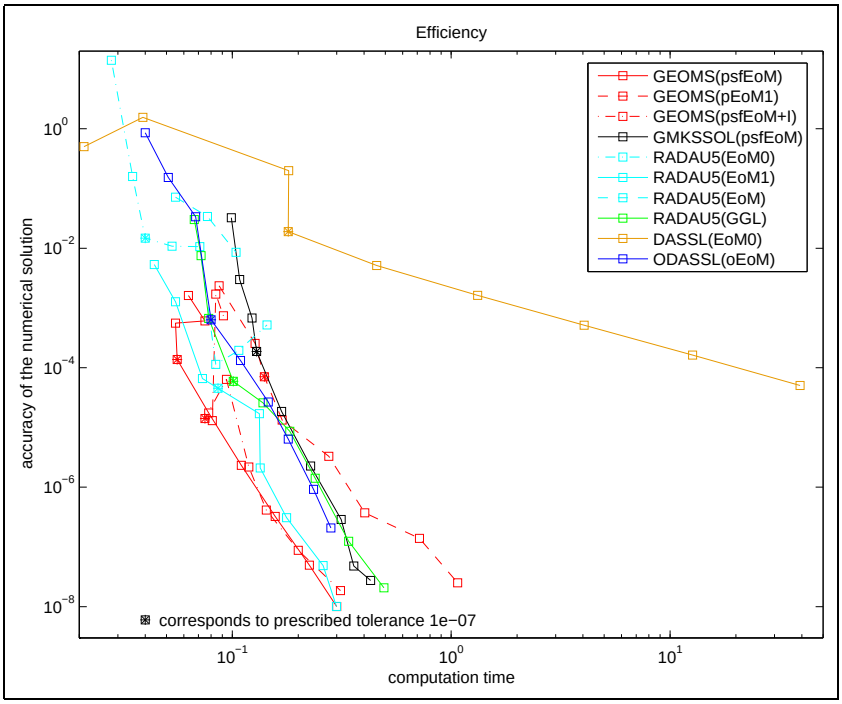

Figure 5.14: Lolly: Efficiency of the solvers based on residual evaluations. Simulations are done with $\beta = 6$ on the time domain $\mathbb{I} = [0, 10]$.

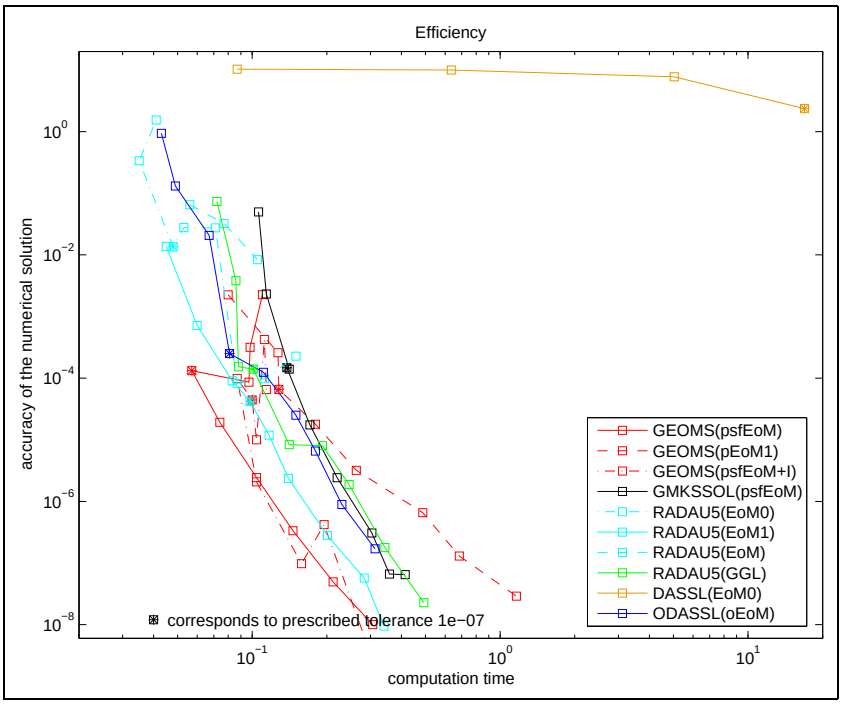

Figure 5.15: Lolly: Efficiency of the solvers based on residual evaluations. Simulations are done with $\beta = 10$ on the time domain $\mathbb{I} = [0, 10]$.

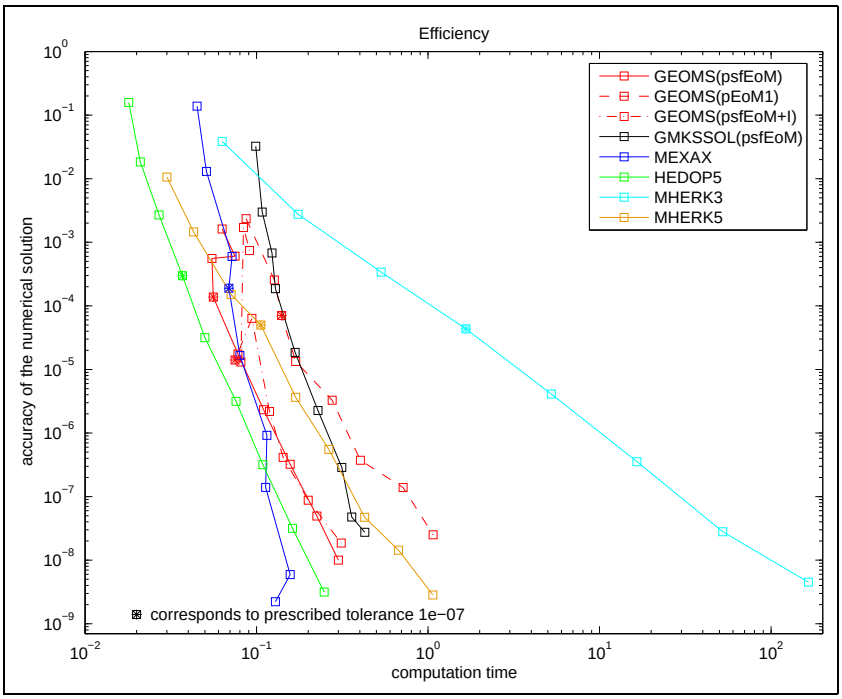

Figure 5.16: Lolly: Efficiency of the solvers based on structural evaluations. Simulations are done with $\beta = 6$ on the time domain $\mathbb{I} = [0, 10]$.

MEXAX, GEOMS(psfEoM), and GEOMS(psfEoM+I). The efficiency of MHERK5, GMKSSOL(psfEoM), and GEOMS(pEoM1) is also very good. Only the performance of the solver MHERK3 trailed. Furthermore, a significant influence of the parameter β on the efficiency is not recognized such that the efficiency remains approximately unchanged with respect to a further increased β.
Concluding this example, it should be noted that the success of the numerical integrations extremely depend on the consistency of the initial values. This results in the fact that by slightly inconsistent initial values, in particular, for the Lagrange multipliers, the integrations are only mastered for prescribed tolerances RTOL=ATOL=10^i, $i = -3, ..., -10$. Only GEOMS using the projected-strangeness-free formulation with or without additional information on solution invariants mastered the numerical simulations for all prescribed tolerances RTOL=ATOL=10^i, $i = -3, ..., -15$. □

Example 5.3.3 The truck model: In Example 4.1.14 we introduced the model of the two dimensional truck example. For a detailed description of the modeling and the equations of motion we refer to [167].

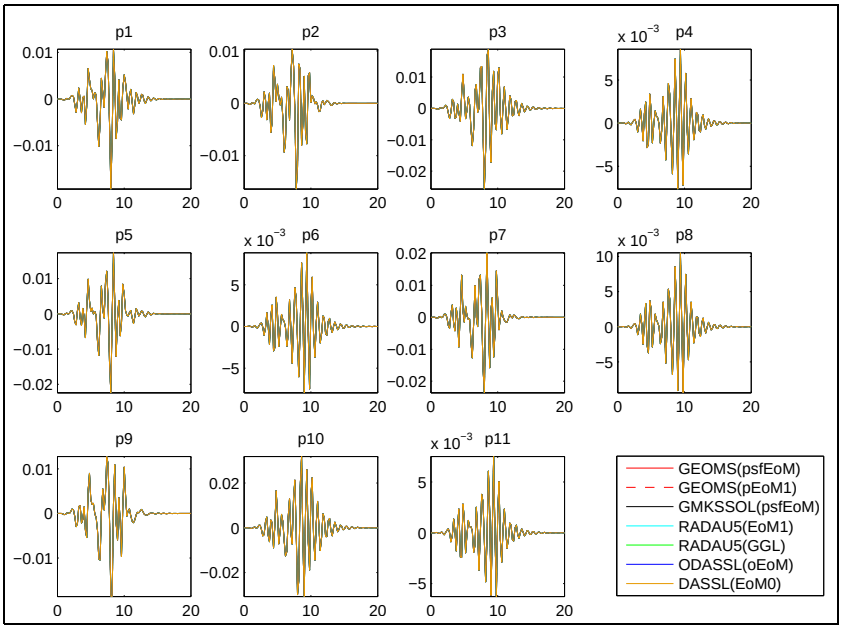

Figure 5.17: Truck: Numerical solutions. Simulation is done with RTOL=ATOL=10^{-7} on the time domain $\mathbb{I} = [0, 20]$.

In Figure 5.17 the numerical solutions of the position variables obtained with a prescribed tolerance RTOL = ATOL=10^{-7} are illustrated.
The accuracy of the numerical solutions is compared with the numerical solution RADAU5(GGL) obtained with a prescribed tolerance RTOL=ATOL=10^{-15}. The precision of all results obtained by a prescribed tolerance are of similar accuracy but the consumed computation time differs, as seen in the Figure 5.18. Obviously, the numerical results DASSL(EoM0) and ODASSL(oEoM) have been obtained in an very efficient way, but a successful integration of the equations of motion was only possibly for prescribed tolerances RTOL=ATOL$\geq 10^{-11}$ by use of DASSL and RTOL=ATOL$\geq 10^{-9}$ by use of ODASSL such that by respecting the accuracy of all solution components, i.e., including the Lagrange multipliers, the best obtained absolute accuracy was about 10^{-4}. On the other hand, by the use of the codes RADAU5, GEOMS, and GMKSSOL not any problem in the numerical integrations for any

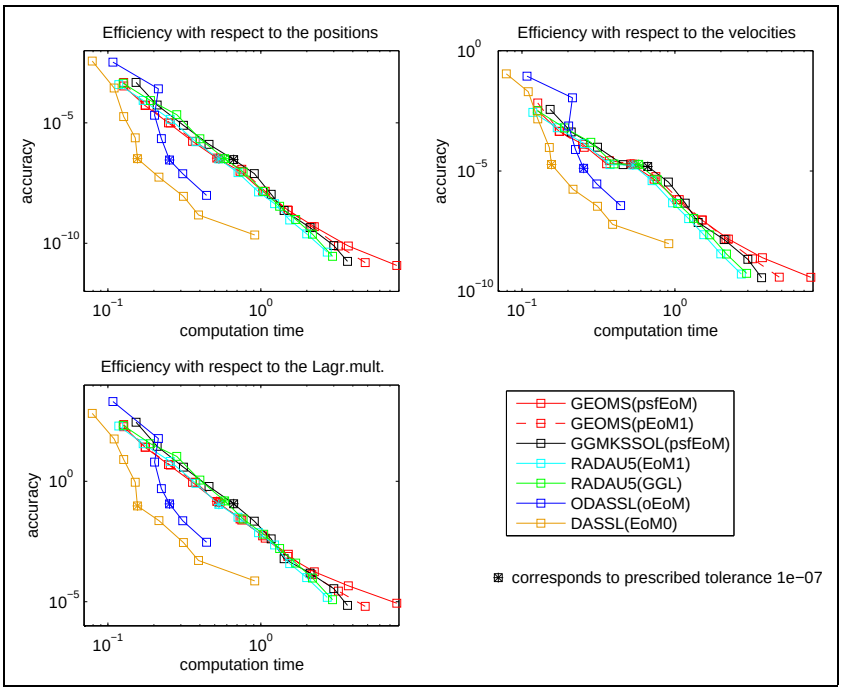

Figure 5.18: Truck: Efficiency of the solvers based on residual evaluations. Simulations are done on the time domain $\mathbb{I} = [0, 20]$.

prescribed tolerances RTOL=ATOL$\geq 10^{-15}$ occurred. □

Example 5.3.4 Slider crank: In Example 4.1.15 we have introduced the model of a slider crank. The topology of the model is depicted in Figure 4.6 and the equations of motion are given by (4.23).

masses	$m_1 = 1.0$	$m_2 = 1.8$
lengths	$l_1 = 1.0$	$l_2 = 3.0$
gravitational acceleration	$g = 9.81$	

Table 5.11: Slider Crank: Parameters (Set 1)

The integration is done on the time domain $t \in [0, 100]$. The initial state is the horizontal strunged-out position to the right such that the initial values are given by

$$p_1(0) = 0,\ p_2(0) = 0,\ v_1(0) = 0,\ v_2(0) = 0,\ \lambda(0) = -17.658.$$

The joint between both arms is falling down initiated by the gravitational acceleration.
The obtained accuracy of the numerical results is again compared with respect to the accuracy of the numerical solution RADAU5(GGL) obtained with a prescribed tolerance RTOL=ATOL=10^{-15}. Figures 5.19 and 5.20 illustrates the efficiency of the numerical solvers based on residual evaluations for the integration of the equations of motion on the time domain $\mathbb{I} = [0, 100]$.
Aside from the excellent efficiency of RADAU5 based on the s-index-0 formulation

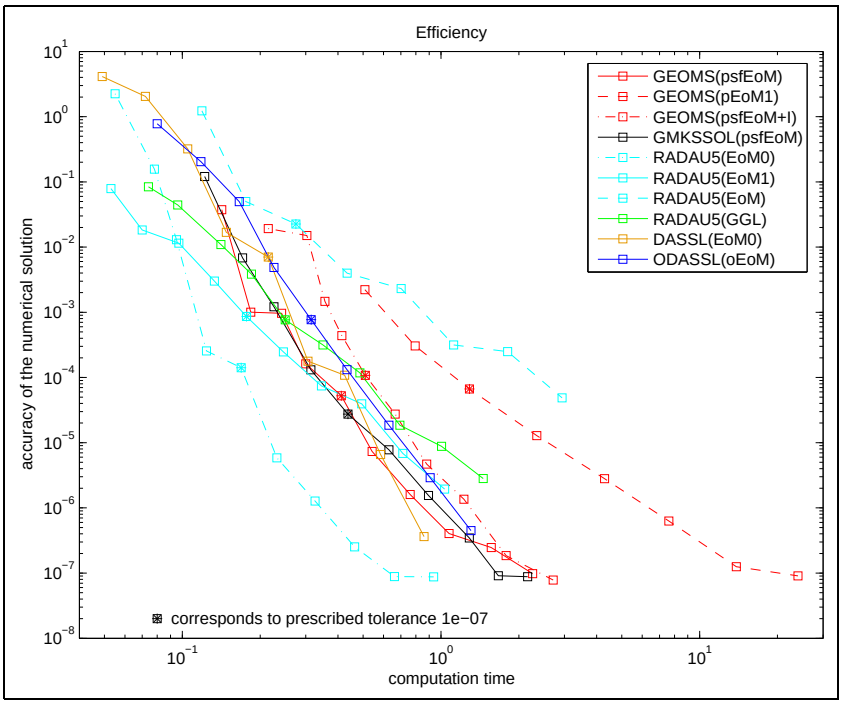

Figure 5.19: Slider Crank: Efficiency of the solvers based on residual evaluations. Simulations are done on the time domain $\mathbb{I} = [0, 100]$.

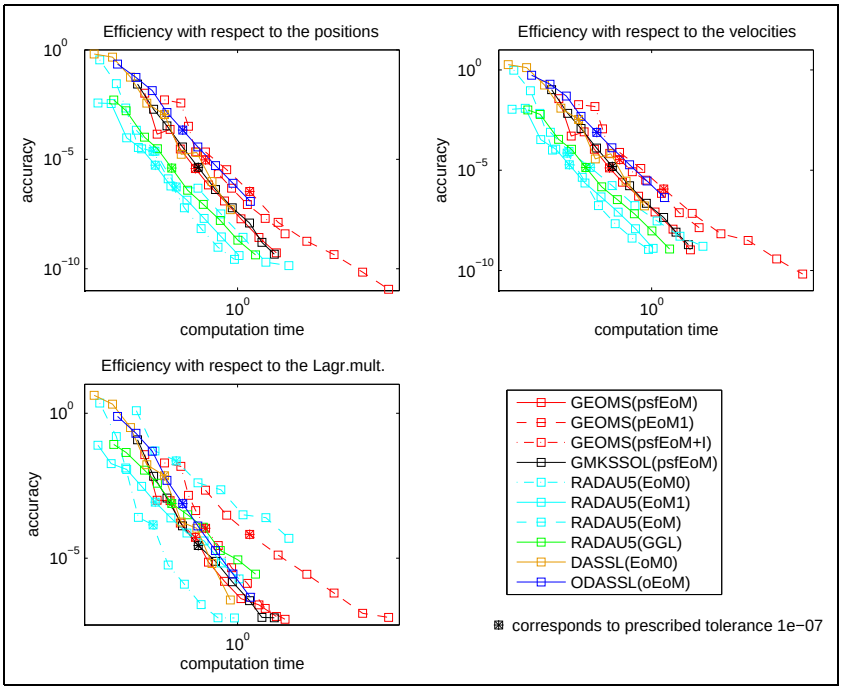

Figure 5.20: Slider Crank: Efficiency of the solvers based on residual evaluations. Simulations are done on the time domain $\mathbb{I} = [0, 100]$.

(4.82) obtaining the results RADAU5(EoM0) for this example (note that the strangeness-index-0 formulation is in general not suitable for the numerical integration because of the occurring drift) the efficiency in obtaining the numerical results RADAU5(EoM1), GEOMS(psfEoM), GMKSSOL(psfEoM), and DASSL(EoM0) show a good performance, while the obtained accuracy of the numerical results RADAU5(GGL), ODASSL(oEoM) is slightly reduced, i.e., the absolute error is slightly increased, which has impact on the efficiency. The efficiency in the obtaining of the numerical results RADAU5(EoM) and GEOMS(EoM1) are not very good, even though GEOMS(EoM1) show the best accuracy for the position variables and the velocity variables.

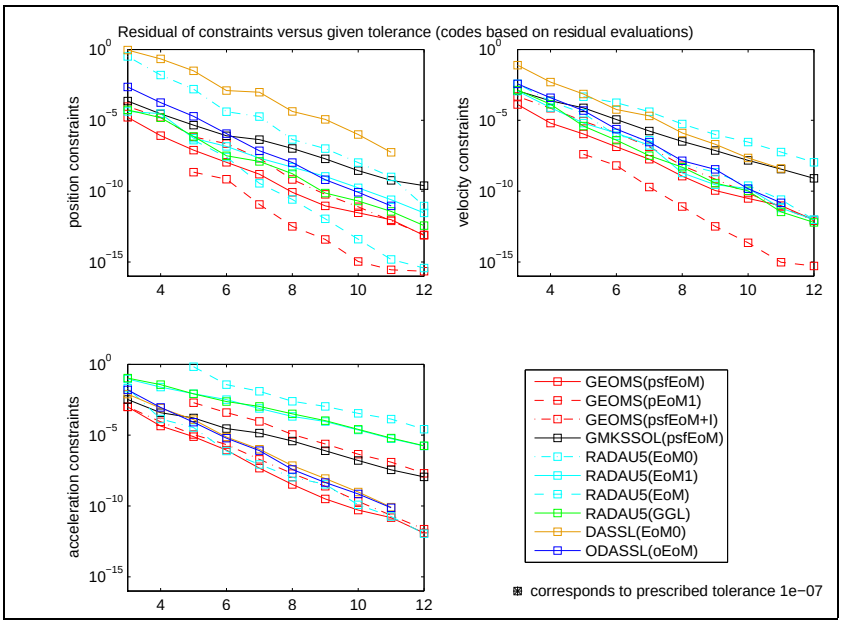

Figure 5.21: Slider Crank: Residual of constraints versus given tolerance (codes based on residual evaluations). Simulations are done on the time domain $\mathbb{I} = [0, 100]$.

As mentioned previously, the consistency of the numerical solution is a very sensitive property. In Figure 5.21 it is shown how accurately the constraints are satisfied in dependence on the prescribed tolerance. Obviously, this accuracy depends on the fact whether the constraints explicitly occur as equations in the used formulation or not. Therefore, the numerical results RADAU5(EoM0) and DASSL(EoM0) do not satisfy the position constraints and show a relatively large residuum. This is explained by the used s-index-0 formulation (4.82) of the equations of motion and the drift-off phenomenon, see Section 4.4. Obviously, the constraints on velocity level are more accurately satisfied but still not in a sufficient way, i.e., the residuals of the constraints on velocity level are smaller than the residuals of the constraints on position level but not small enough, and the constraints on acceleration level are very accurately satisfied. An opposite observation can be made by the use of the original form of the equations of motion (4.43). Here, the constraints on position level are accurately satisfied in contrast to the constraints on acceleration level. The analogous observations can be made for the other numerical solutions, in particular, the solutions GEOMS(pEoM1) satisfy the constraints on position and on velocity level very accurately while all constraints are satisfied from the solutions GEOMS(psfEoM).

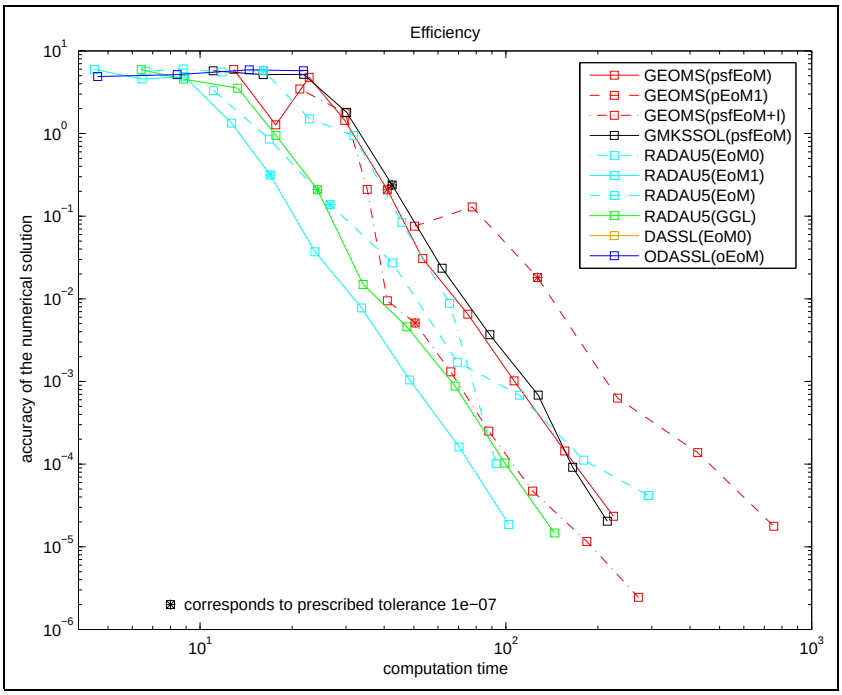

Figure 5.22: Slider Crank: Efficiency of the numerical simulation of the slider crank over the time domain $\mathbb{I} = [0, 10000]$ for codes based on residual evaluations.

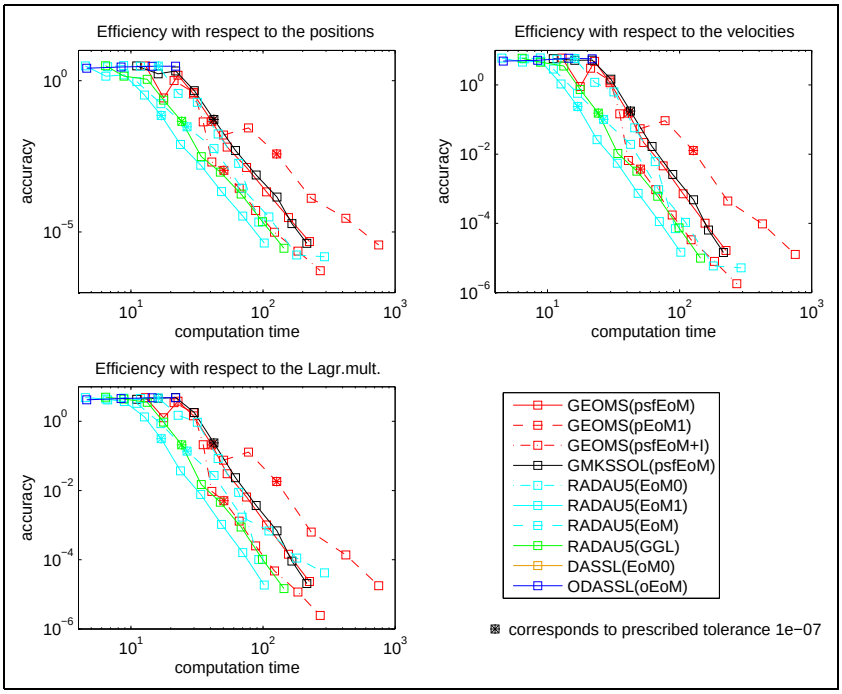

Figure 5.23: Slider Crank: Efficiency of the numerical simulation of the slider crank over the time domain $\mathbb{I} = [0, 10000]$ for codes based on structural evaluations.

It should be noted that the numerical integration on the domain $\mathbb{I} = [0, 10000]$ has only been mastered with `RADAU5` by use of the s-index-1 formulation and by use of the Gear-Gupta-Leimkuhler formulation (4.103), with `GEOMS` by use of the projected-strangeness-free formulation (4.114), and with `GMKSSOL` for all tolerances $\texttt{RTOL} = \texttt{ATOL} = 10^i$ with $i = -3, ..., -15$. The integration with `GEOMS` by use of the projected-s-index-1 formulation (4.123) was successful for tolerances $\texttt{RTOL} = \texttt{ATOL} = 10^i$ with $i = -5, ..., -10$, the integration with `ODASSL` was successful for tolerances $\texttt{RTOL} = \texttt{ATOL} = 10^i$ with $i = -3, ..., -6$, and the integration with `DASSL` by use of the s-index-0 formulation (4.82) was only successful for the tolerance $\texttt{RTOL} = \texttt{ATOL} = 10^{-5}$. The results for the simulation over the domain $\mathbb{I} = [0, 10000]$ with respect to the efficiency are depicted in Figures 5.22 and 5.23.

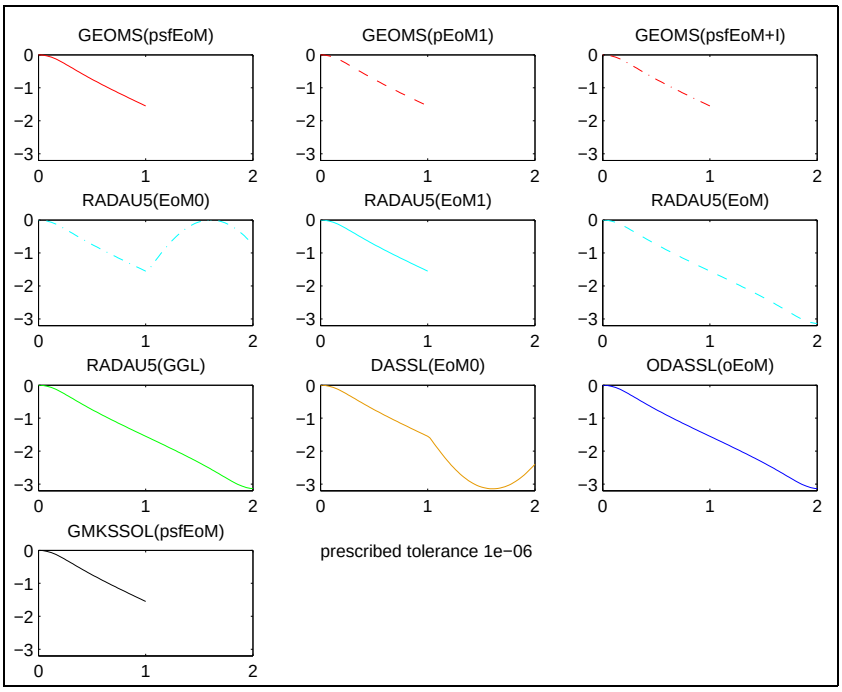

Figure 5.24: Slider crank: Numerical solutions for p_1 of the numerical simulation of the slider crank passing a singular state at $t = 1.01$. Simulation is done with $\texttt{RTOL}{=}\texttt{ATOL}{=}10^{-6}$ on the time domain $\mathbb{I} = [0, 2]$.

masses	$m_1 = 1.0$	$m_2 = 1.8$
lengths	$l_1 = 1.0$	$l_2 = 1.0$
gravitational acceleration	$g = 9.81$	

Table 5.12: Slider Crank: Parameters (Set 2)

In the following, let us consider the case that both arms of the slider crank have the same length, say $l_1 = l_2 = 1$, see Table 5.12. In this case the motion of the slider crank passes the state $p_1 = p_2 = -\pi/2$, where both arms of the slider crank are hanging down over each other and the end of the second rod is placed in the origin. In this situation the rank of the constraint matrix G jumps from 1 to 0 such that we have reached a singularity. Then the constraints are redundant and there is a singular point in the solution path such that the solution is not uniquely

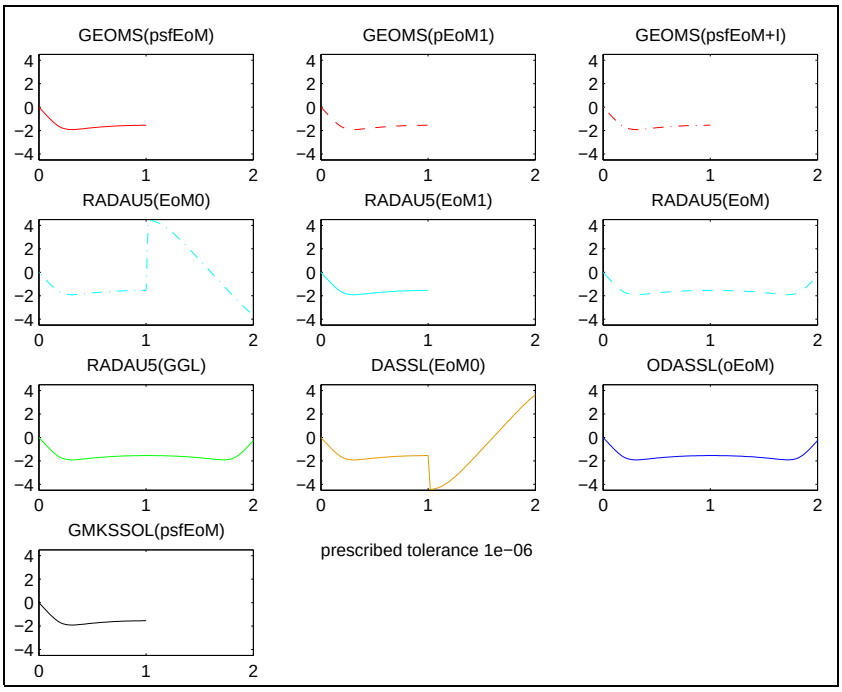

Figure 5.25: Slider crank: Numerical solutions for v_1 of the numerical simulation of the slider crank passing a singular state at $t = 1.01$. Simulation is done with RTOL=ATOL=10^{-6} on the time domain $\mathbb{I} = [0, 2]$.

determined. It is obvious that two motions are possible. First, the end of the slider crank remains in the point of origin and both rods are rotating around it, or secondly, the end of the slider crank moves to the left or to the right and the angle between the arms increases.
The obtained numerical solutions for the position p_1 and for the velocity v_1 for a prescribed tolerance RTOL=ATOL=10^{-6} over the time domain $\mathbb{I} = [0, 2]$ with several codes are given in Figures 5.24 and 5.25, respectively. From Figure 5.24 and Figure 5.25 it is obvious that the numerical solutions RADAU5(EoM0), RADAU5(EoM), as well as RADAU5(GGL) pass the singular state. Furthermore although the step size control of DASSL reduces the step size rapidly in the neighborhood of the singularity, the numerical solutions DASSL(EoM0) passes this point of the singularity with a following increasing step size and without any warning or error message. The same holds for the numerical solutions ODASSL(oEoM). Only in case of using the s-index-1 formulation (4.81) for RADAU5, i.e., RADAU5(EoM1), and by use of GEOMS and GMKSSOL, i.e., GEOMS(psfEoM), GEOMS(pEoM1), GEOMS(psfEoM+I), and GMKSSOL(psfEoM), the singular state is detected and the integration is stopped.
As mentioned in the Example 4.1.15 using automatic tools for the generation of the equations of motion it may happen that the constraints are modeled in a redundant way. Therefore, let us consider the equations of motion in form (4.24), i.e., with redundant constraints. The numerical integration is again done on the domain $\mathbb{I} = [0, 100]$ with the initial values

$$p_1(0) = 0,\ p_2(0) = 0,\ v_1(0) = 0,\ v_2(0) = 0,$$
$$\lambda_1(0) = -17.658,\ \lambda_2(0) = 0,\ \lambda_3(0) = 0.$$

The accuracy of the obtained solutions are compared with the numerical solution

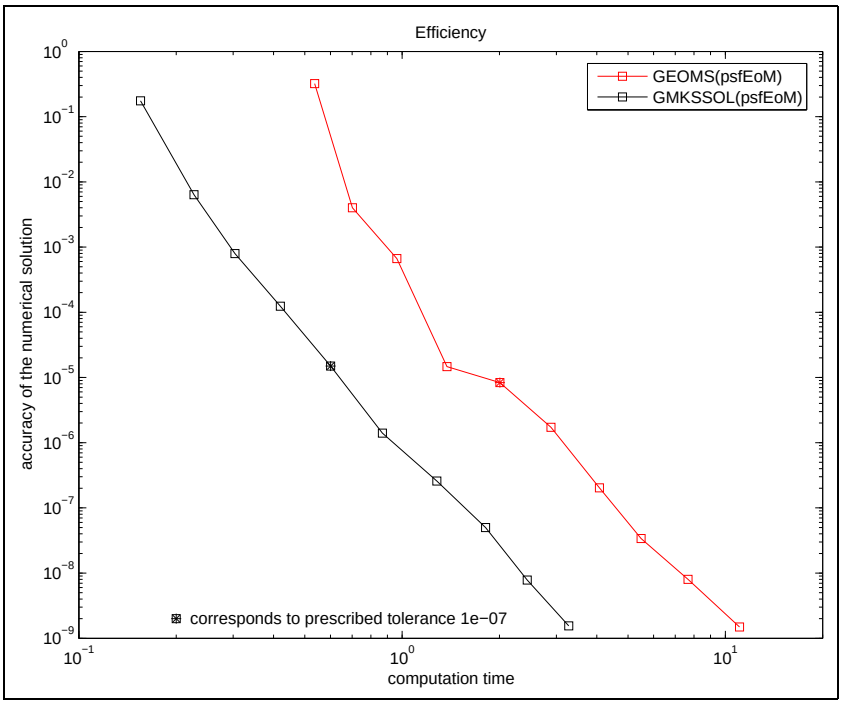

Figure 5.26: Slider Crank: Efficiency of the solvers based on residual evaluations. Simulations are done on the time domain $\mathbb{I} = [0, 100]$.

RADAU5(GGL) with a prescribed tolerance RTOL=ATOL=10^{-15} by use of nonredundant constraints as discussed before. Because of the redundancies in the constraints, the Lagrange multipliers are not unique such that the obtained accuracy is only measured for the position variables p and the velocity variables v.
The efficiency of the integration is illustrated in Figure 5.26. Note that a numerical integration by use of RADAU5, DASSL, or ODASSL is not possible because of the redundancies in the constraints. As seen in Figure 5.26 the numerical solutions via GEOMS needs more computation time as via GMKSSOL for the same obtained accuracy. For stability reasons it is only allowed in GEOMS to solve equations of motion with redundant constraints by use of the SV decomposition for the constraints, see Section 5.1.7. Therefore, the results GEOMS(psfEoM) are obtained by use of the SV decomposition for the algebraic part and by use of the LU decomposition for the remaining differential part, see Section 3.5.4.3. On the other hand, since the integration method in GMKSSOL is an adaption of the code RADAU5, the linear integration system is solved by use of the LU decomposition for both parts. Therefore, the time consumption by use of GMKSSOL is smaller than via GEOMS, but the determination of the redundancies in the constraints is safer via GEOMS. □

Example 5.3.5 The skateboard: In Example 4.1.17 we introduced the model of a skateboarder. The equations of motion are given in (4.27). Obviously, because of the nonholonomic constraints, the equations of motion are of modeling level 3 (4.43).
Since the codes GMKSSOL and the codes based on structural evaluations are not designed for the numerical integration of equations of motion including nonholonomic constraints, the numerical integration is only done by the codes RADAU5, DASSL, ODASSL, and GEOMS.
First, let us simulate the motion of the skater with the parameter as depicted in

masses	$m_1 = 1$	$m_2 = 1$	$m_3 = 80$
inertias	$J_1 = 0.001$	$J_2 = 0.001$	
lengths	$L_1 = 0.5$	$L_3 = 1.5$	
stiffness	$c = 0$		
dampings	$d_1 = d_2 = d_3 = d_\varphi = d_\theta = 0.01$		
gravitational acceleration	$g = 9.81$		
banking coefficient	$a = 11.0$		

Table 5.13: Skateboarder: Parameters (Set 1)

Table 5.13 and the initial values

$$
\begin{array}{rclrclrcl}
x_1 &=& 0.25, & \dot{x}_1 &=& 0.5, & \lambda_1 &=& 0, \\
y_1 &=& 0, & \dot{y}_1 &=& 0, & \lambda_2 &=& 0.44e-04, \\
x_2 &=& -0.25, & \dot{x}_2 &=& 0.5, & \lambda_3 &=& -0.9634e-03, \\
y_2 &=& 0, & \dot{y}_2 &=& 0, & \lambda_4 &=& 0.6667e-06, \\
\varphi &=& 0, & \dot{\varphi} &=& 0, & \lambda_5 &=& 784.8, \\
x_3 &=& 0, & \dot{x}_3 &=& 0.5, & \mu_1 &=& 0.01109, \\
y_3 &=& 0, & \dot{y}_3 &=& -0.0015, & \mu_2 &=& -0.01109, \\
z_3 &=& 1.5, & \dot{z}_3 &=& 0, & & & \\
\theta &=& 0, & \dot{\theta} &=& 0.001. & & &
\end{array}
$$

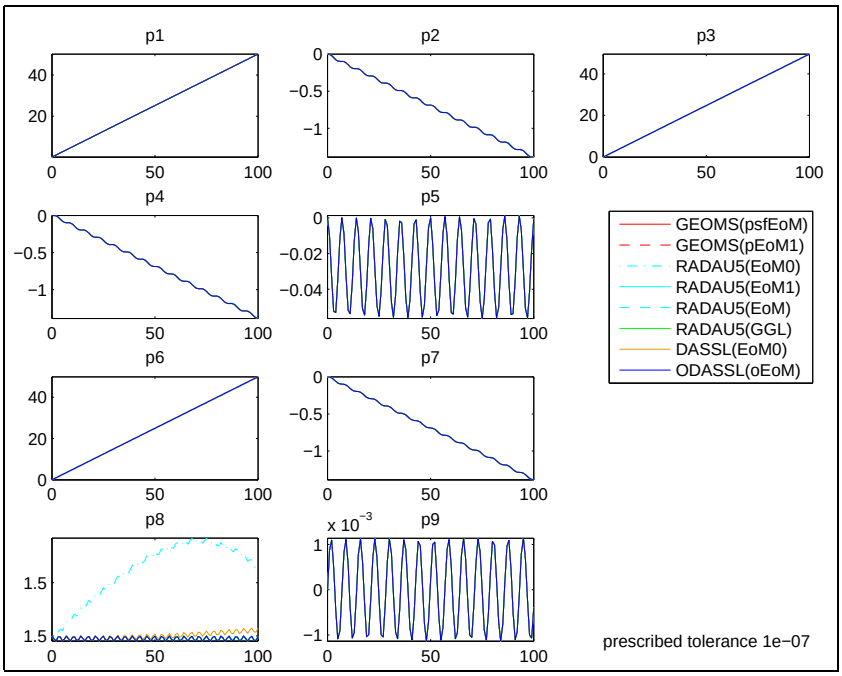

Figure 5.27: Skateboard: Numerical solutions of the solvers based on residual evaluations. Simulations are done with the tolerance RTOL=ATOL=10^{-7} on the time domain $\mathbb{I} = [0, 100]$.

The numerical solutions are shown in Figure 5.27. It seems that all solutions are of similar quality except the solutions based on the s-index-0 formulation, i.e., DASSL(EoM0) and RADAU5(EoM0), in the component p_8. But in Figure 5.28 the

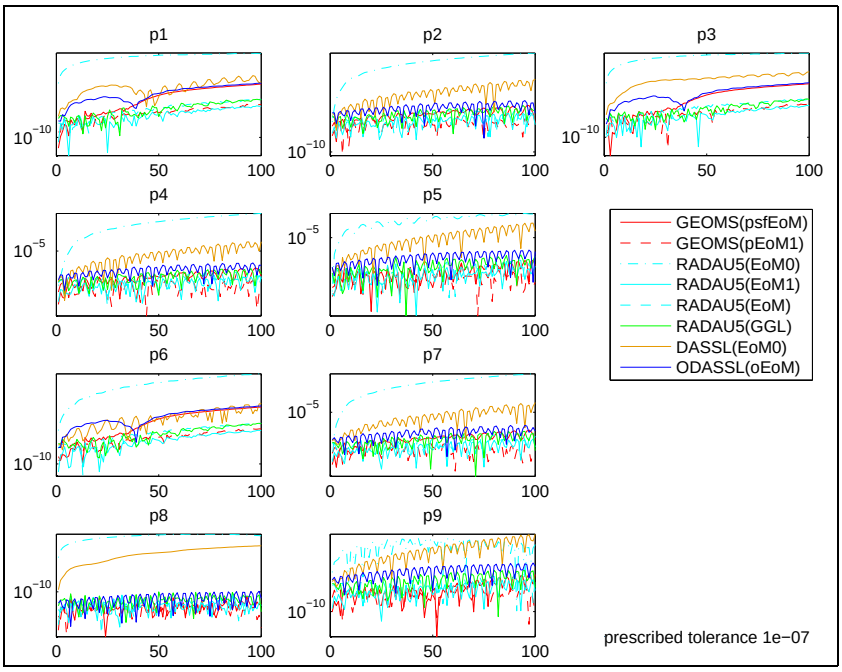

Figure 5.28: Skateboard: Numerical error of the solvers based on residual evaluations. Simulations are done with the tolerance RTOL=ATOL=10^{-7} on the time domain $\mathbb{I} = [0, 100]$.

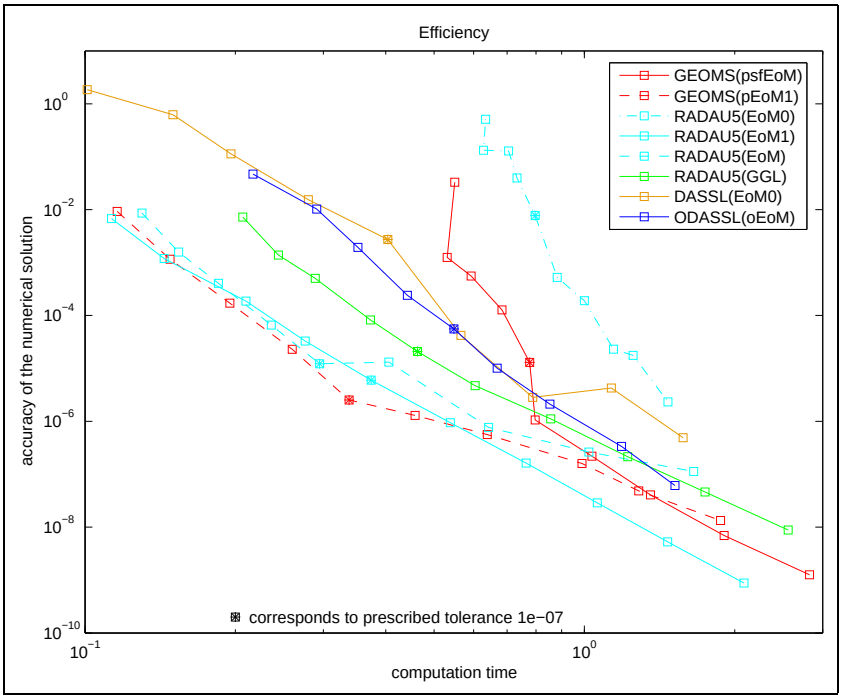

Figure 5.29: Skateboard: Efficiency of the solvers based on residual evaluations. Simulations are done on the time domain $\mathbb{I} = [0, 100]$.

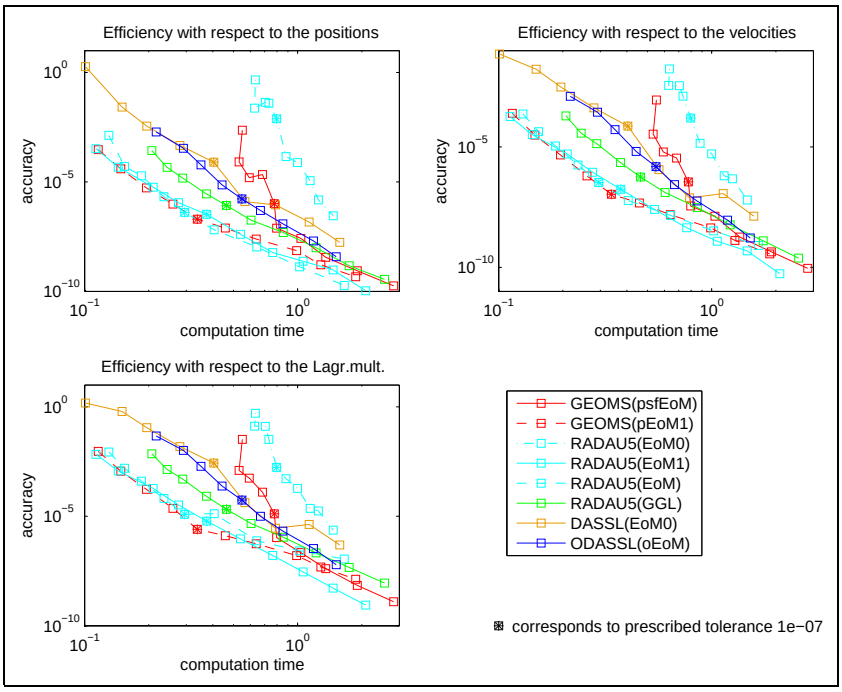

Figure 5.30: Skateboard: Efficiency of the solvers based on residual evaluations. Simulations are done on the time domain $\mathbb{I} = [0, 100]$.

obtained accuracy by a prescribed tolerance of RTOL=ATOL=10^{-7} is illustrated and it is obvious that numerical solutions DASSL(EoM0) and RADAU5(EoM0) are not very accurate. The accuracy of the numerical solutions is compared with the numerical solution RADAU5(GGL) obtained with a prescribed tolerance RTOL=ATOL=10^{-15}. In Figures 5.29 and 5.30 the efficiency of the solvers is illustrated. Obviously, the numerical solutions GEOMS(pEoM1) and RADAU5(EoM1) are obtained in a very efficient way for all prescribed tolerances RTOL=ATOL=10^i with $i = -3, ..., -12$. For large prescribed tolerances, i.e., RTOL = ATOL = 10^i with $i = -3, ..., -7$ the numerical results GEOMS(psfEoM) are not obtained very efficiently, but for smaller tolerances, the efficiency of GEOMS is growing and, in particular, it is more efficient than RADAU5 with use of the Gear-Gupta-Leimkuhler formulation. The efficiency of DASSL as well as of ODASSL is slightly reduced and the best obtained accuracy, approximately 10^{-6} and 10^{-7}, respectively, is not as good as the one of the other numerical solutions, except RADAU5(EoM0). The efficiency obtaining the results RADAU5(EoM0) is out of interest.

masses	$m_1 = 1$	$m_2 = 1$	$m_3 = 80$
inertias	$J_1 = 0.001$	$J_2 = 0.001$	
lengths	$L_1 = 0.5$	$L_3 = 1.5$	
stiffness	$c = 1$		
dampings	$d_1 = d_2 = d_3 = d_\varphi = d_\theta = 0.005$		
gravitational acceleration	$g = 9.81$		
banking coefficient	$a = 11.0$		

Table 5.14: Skateboarder: Parameters (Set 2)

In the following we will change the parameter of the model of the skateboard as shown in Table 5.14 by use of initial values

$$\begin{array}{rclrclrcl} x_1 &=& 0.25, & \dot{x}_1 &=& 5, & \lambda_1 &=& 0, \\ y_1 &=& 0, & \dot{y}_1 &=& 0, & \lambda_2 &=& 0.044, \\ x_2 &=& -0.25, & \dot{x}_2 &=& 5, & \lambda_3 &=& -0.04817, \\ y_2 &=& 0, & \dot{y}_2 &=& 0, & \lambda_4 &=& 0.0003333, \\ \varphi &=& 0, & \dot{\varphi} &=& 0, & \lambda_5 &=& 783.6, \\ x_3 &=& 0, & \dot{x}_3 &=& 5, & \mu_1 &=& 11.09, \\ y_3 &=& 0, & \dot{y}_3 &=& -0.15, & \mu_2 &=& -11.09, \\ z_3 &=& 1.5, & \dot{z}_3 &=& 0, & & & \\ \theta &=& 0, & \dot{\theta} &=& 0.1. & & & \end{array}$$

In particular, mainly we did introduce a stiffness c and reduced the damping d_1, d_2, d_3, d_φ, d_θ of the system. To compensate these changes, we increased the initial velocity of the skateboard.

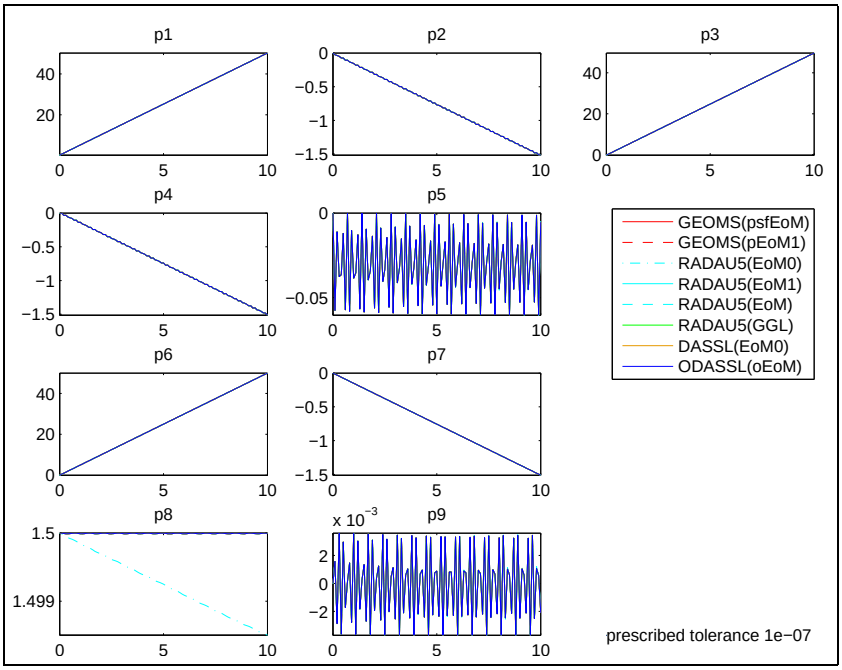

Figure 5.31: Skateboard: Numerical solutions of the solvers based on residual evaluations. Simulations are done with the tolerance RTOL=ATOL=10^{-7} on the time domain $\mathbb{I} = [0, 10]$.

In Figure 5.31 we have illustrated the solution of the numerical integration by use of several numerical algorithms with a prescribed tolerance RTOL=ATOL=10^{-7}. In comparison to the first scenario the oscillating motion is increased, i.e., the frequence as well as the amplitude, as we can see, in particular, in the solution component p_5 and p_9. Analogously to the first test scenario, the numerical solutions seem to be of comparable accuracy, except the numerical solutions RADAU5(EoM0) with a visible deviation. The absolute error of the position components is illustrated in Figure 5.32. Apart from the numerical results RADAU5(EoM0) and DASSL(EoM0), all numerical results satisfy the prescribed tolerance.
In Figures 5.33 and 5.34, the efficiency of the several used numerical integrations is shown. Again the efficiency behavior for obtaining the numerical results

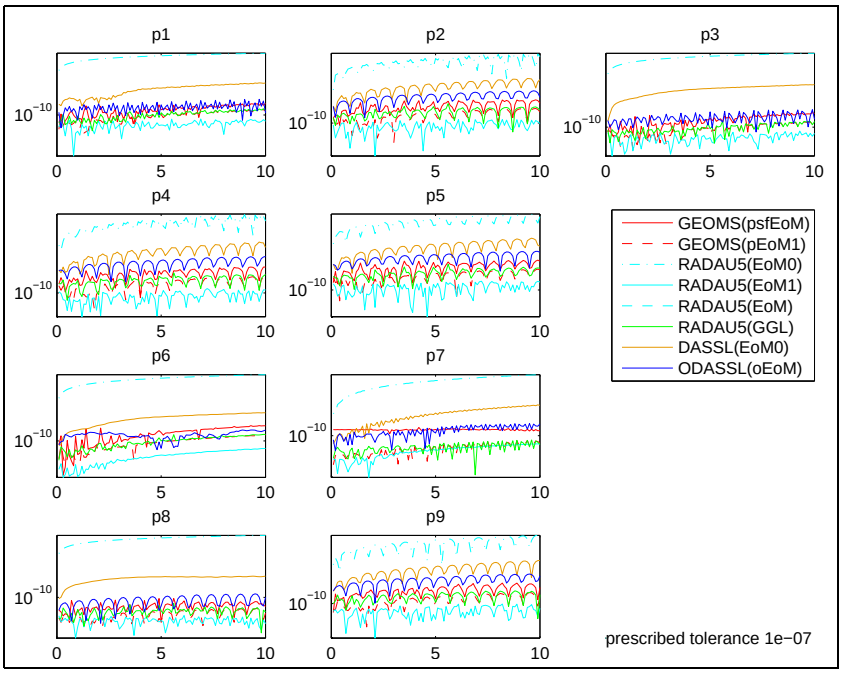

Figure 5.32: Skateboard: Numerical error of the solvers based on residual evaluations. Simulations are done with the tolerance `RTOL`=`ATOL`=10^{-7} on the time domain $\mathbb{I} = [0, 10]$.

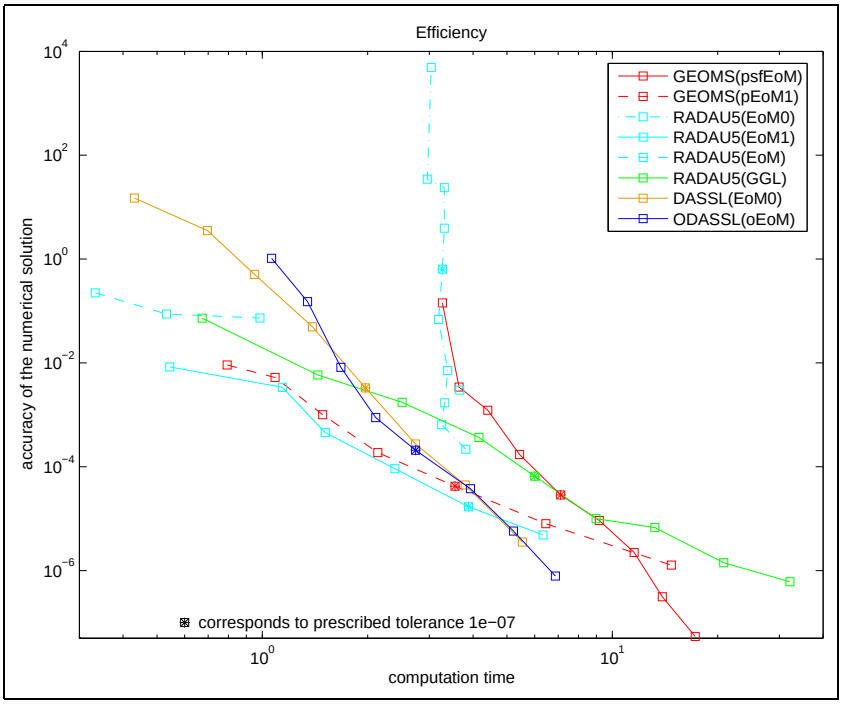

Figure 5.33: Skateboard: Efficiency of the solvers based on residual evaluations. Simulations are done on the time domain $\mathbb{I} = [0, 10]$.

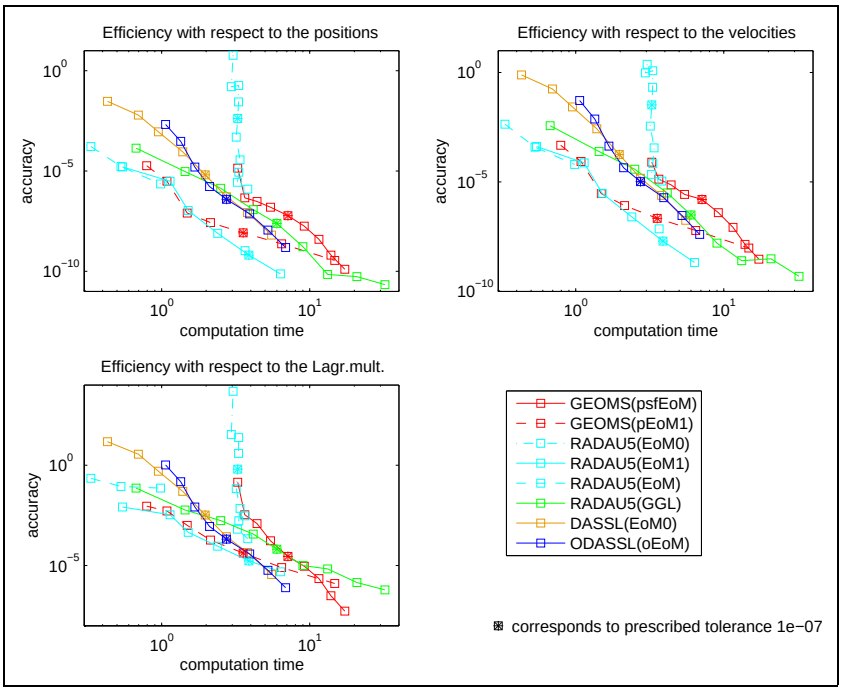

Figure 5.34: Skateboard: Efficiency of the solvers based on residual evaluations. Simulations are done on the time domain $\mathbb{I} = [0, 10]$.

RADAU5(EoM0) is not acceptable and even the numerical results RADAU5(EoM), too. Furthermore, the comparison of the efficiency of the several numerical integration methods has to be partitioned into two accuracy ranges. First, for an obtained accuracy in $[10^2, 10^{-4}]$ the results RADAU5(EoM1) and GEOMS(pEoM1) are obtained in a very efficient way closely followed by RADAU5(GGL), DASSL(EoM0), and ODASSL(oEoM). Secondly, for an obtained accuracy in $[10^{-4}, 10^{-8}]$ the results ODASSL(oEoM) are obtained in a very efficient way followed by GEOMS(pEoM1), GEOMS(psfEoM), and RADAU5(GGL) while only the results GEOMS(psfEoM) does reach a accuracy in the range of approximately $[10^{-6}, 10^{-8}]$. □

Example 5.3.6 An academical example: Let us conclude this section with an academical example. The equations of motion are defined by

$$
\begin{aligned}
\begin{bmatrix} \dot{p}_1 \\ \dot{p}_2 \\ \dot{p}_3 \end{bmatrix} &= \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}, \\
\begin{bmatrix} \dot{v}_1 \\ \dot{v}_2 \\ \dot{v}_3 \end{bmatrix} &= \begin{bmatrix} -2p_2 \\ 2v_1 + p_2 + w_1 \\ r_1 + r_2 p_3 \end{bmatrix} - \begin{bmatrix} -t + s_2/p_3 + s_1 & 0 \\ 0 & 2p_2 \\ -s_2 s_1/p_3 - s_1 & 2p_3 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix}, \\
\begin{bmatrix} \dot{r}_1 \\ \dot{r}_2 \end{bmatrix} &= \begin{bmatrix} 2p_2 \\ -2 - 2s_1 \end{bmatrix}, \\
0 &= \begin{bmatrix} w_1 + 2p_3 \end{bmatrix}, \\
0 &= \begin{bmatrix} s_1 p_3 - p_1 \\ p_1 - s_2 - p_3 \end{bmatrix}, \\
0 &= \begin{bmatrix} s_1 s_2 - t p_1 \\ p_2^2 + p_3^2 - 1 \end{bmatrix}
\end{aligned}
$$

This example is designed in such a way that the exact solution is known analytically and such that it includes most of the features of the equations of motion used in industrial applications. The exact solution of our interest is given by

$$\begin{array}{lllllllll} p_1(t) &=& \cos(t)(1+t), & p_2(t) &=& \sin(t), & p_3(t) &=& \cos(t), \\ v_1(t) &=& -\sin(t)(t+1)+\cos(t), & v_2(t) &=& \cos(t), & v_3(t) &=& -\sin(t), \\ r_1(t) &=& -2\cos(t), & r_2(t) &=& -4t-t^2, & & & \\ w_1(t) &=& -2\cos(t), & & & & & & \\ s_1(t) &=& 1+t, & s_2(t) &=& \cos(t)t, & & & \\ \lambda_1(t) &=& \cos(t), & \lambda_2(t) &=& -t. & & & \end{array}$$

The constraint matrix is defined in Lemma 4.1.19 and, therefore, it is given by

$$G = \begin{bmatrix} -t + s_2/p_3 + s_1 & 0 & -s_2 s_1/p_3 - s_1 \\ 0 & 2p_2 & 2p_3 \end{bmatrix}.$$

With respect to the solution it follows that

$$G = \begin{bmatrix} 1+t & 0 & -t^2-2t-1 \\ 0 & 2\sin(t) & 2\cos(t) \end{bmatrix}.$$

Consequently, the constraint matrix has full rank for all $t \neq -1$.
Let us integrate these equations of motion on the domain $\mathbb{I} = [0, 1.5]$. Note that, in particular, because of the additional variables w, the solver `GMKSSOL` and the solvers based on structural evaluations are not able to integrate these equations of motion. In Figure 5.35 the numerical solutions obtained by use of several solvers are depicted. Obviously, the integration by use of `ODASSL` breaks down when reaching $t = 0.623$. By use of smaller tolerances, i.e., `RTOL`=`ATOL`=10^i with $i < -7$ this break down happens earlier in the simulation. The other numerical solutions stay in the neighborhood of the exact solution for the whole domain $\mathbb{I}$ as shown in Figure 5.36. The efficiency of the numerical integration is illustrated in Figures 5.37 and 5.38. Obviously, a successful integration of this problem by use of `ODASSL` is impossible for any prescribed tolerances. Furthermore, the numerical integration by use of `DASSL` based on the s-index-0 formulation (4.82) is only successful for prescribed tolerances `RTOL`=`ATOL`=10^i, $i = -1, ..., -8$, while the numerical integration by use of `RADAU5` and `GEOMS` is successful for almost any prescribed tolerances, i.e., `RTOL`=`ATOL`=10^i, $i = -1, ..., -15$.
While the numerical results GEOMS(pEoM1), RADAU5(EoM1), and RADAU5(GGL) are of similar quality, the accuracy of the numerical solutions GEOMS(psfEoM) is much better. Figure 5.38 gives detailed information of the obtained efficiency regarding different types of unknowns. Here, it becomes obvious that the numerical approximation of the Lagrange multipliers of the results GEOMS(psfEoM) is much better than by use of any other solver while the numerical approximation of the position and the velocity variables are similar compared to the other numerical results.
The situation changes if we extend the domain to $\mathbb{I} = [0, 2]$. The numerical solutions are given in Figure 5.39. In Figure 5.40 the obtained accuracy is illustrated by a prescribed tolerance of `RTOL`=`ATOL`=10^{-7}. Obviously, in addition to the numerical solutions ODASSL(oEoM) also the numerical solutions DASSL(EoM0), RADAU5(EoM1), and GEOMS(pEoM1) do not reach the end of $\mathbb{I}$. They end by reaching $t = \pi/2$. At this moment, the position variable p_3 as well as the contact variable s_2 and the Lagrange multiplier λ_1 are zero.
In Figures 5.41 and 5.42 again the efficiency of the obtained numerical solutions is illustrated. Obviously, the numerical integration by use of `DASSL`, `ODASSL`, or `RADAU5` based on the s-index-1 formulation (4.81) does not lead to success for any

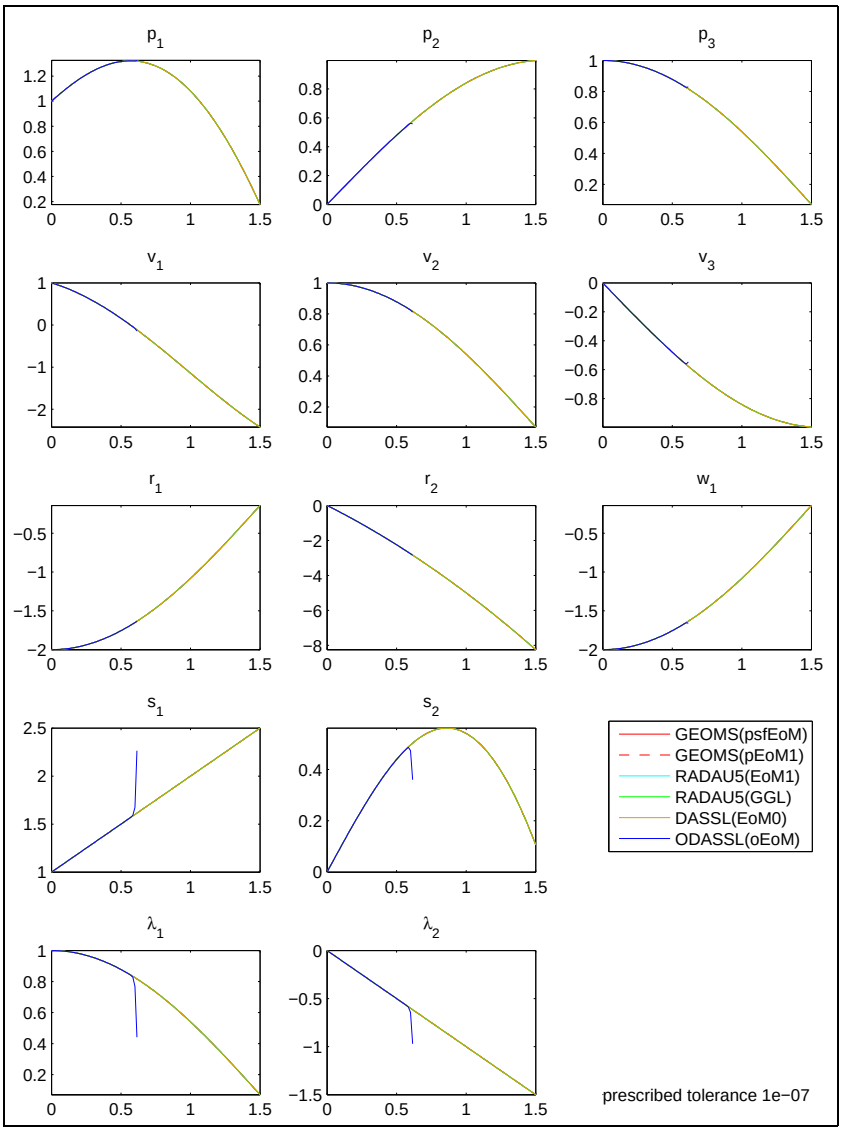

Figure 5.35: Academical example: Numerical solutions of the solvers based on residual evaluations. Simulations are done with the tolerance RTOL=ATOL=10^{-7} on the time domain $\mathbb{I} = [0, 1.5]$.

prescribed tolerances i.e., RTOL=ATOL=10^i, $i = -1, ..., -15$. The numerical integration by use of GEOMS based on the projected-s-index-1 formulation leads to success at least for RTOL=ATOL=10^{-6}. While the numerical integration by use of RADAU5 based on the Gear-Gupta-Leimkuhler formulation (4.103) is successful for the prescribed tolerances RTOL=ATOL=10^i, $i = -1, ..., -9, -11, -12, -14, -15$, the numerical integration by use of GEOMS based on the projected-strangeness-free formulation was successful for RTOL=ATOL=10^i, $i = -1, ..., -12$ and the most efficient. □

In summary, the examples show that the numerical algorithm RADAU5 in combination with the Gear-Gupta-Leimkuhler formulation (4.103) and in combination with the s-index-1 formulation (4.81) is well suited for the numerical integration of the equations of motion. Very seldomly an integration of the examples was not successful. Furthermore, the performance is mostly very good. The numerical results

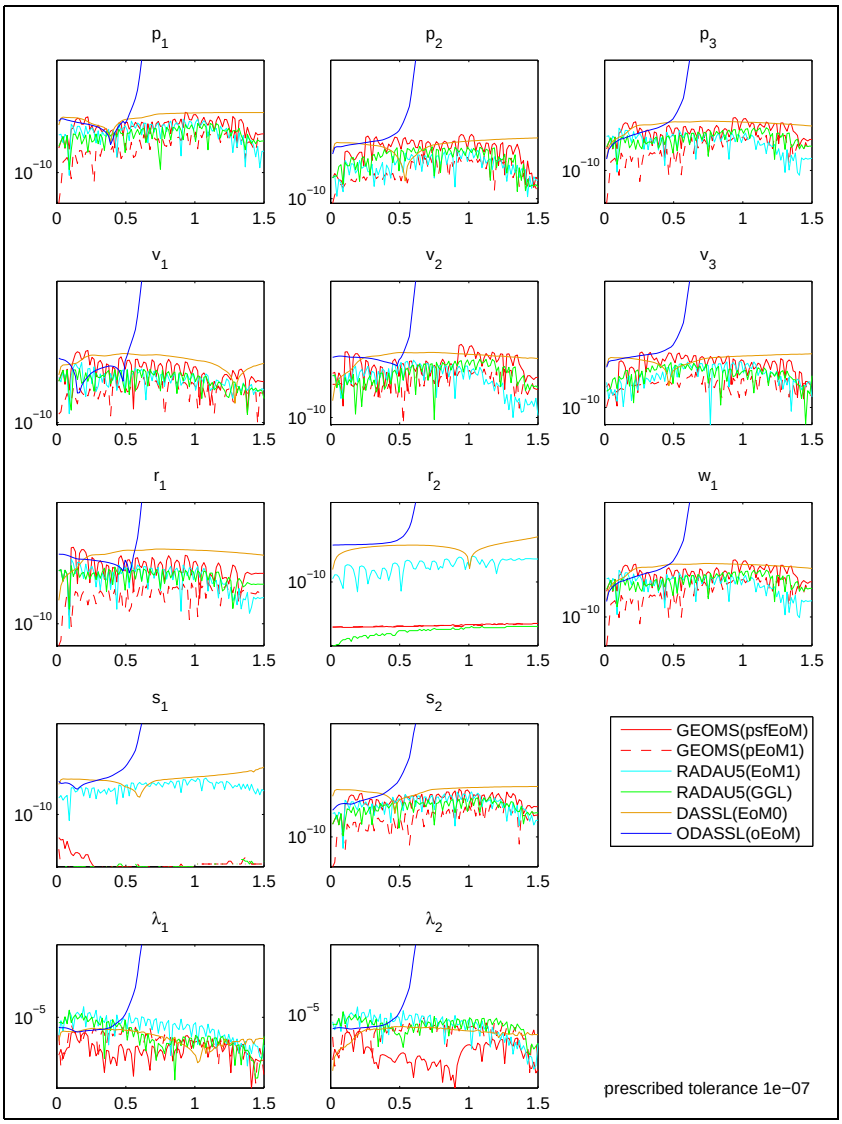

Figure 5.36: Academical example: Accuracy of the solvers based on residual evaluations. Simulations are done with the tolerance RTOL=ATOL=10^{-7} on the time domain $\mathbb{I} = [0, 1.5]$.

obtained by use of the s-index-0 formulation (4.82) as well as the original form of the equations of motion (4.43) substantiates the fact that these formulations are in general not suitable for numerical integrations.
Furthermore, the numerical results suggest that the numerical algorithm ODASSL is an adequate method for large tolerances, i.e., say RTOL=ATOL$\in [10^{-3}, 10^{-7}]$. The success of the numerical integration by use of ODASSL depends highly sensitively on the consistency of the initial values. In particular, an inconsistent choice of the Lagrange multipliers very often prevents the success of the numerical integration. A similar observation holds for the use of DASSL while the code ODASSL behaves a bit more robust and a bit more accurate.
The efficiency of the numerical algorithms HEDOP5 and MEXAX which base on structural evaluations is undisputed at the expense of the flexibility in its applicability.

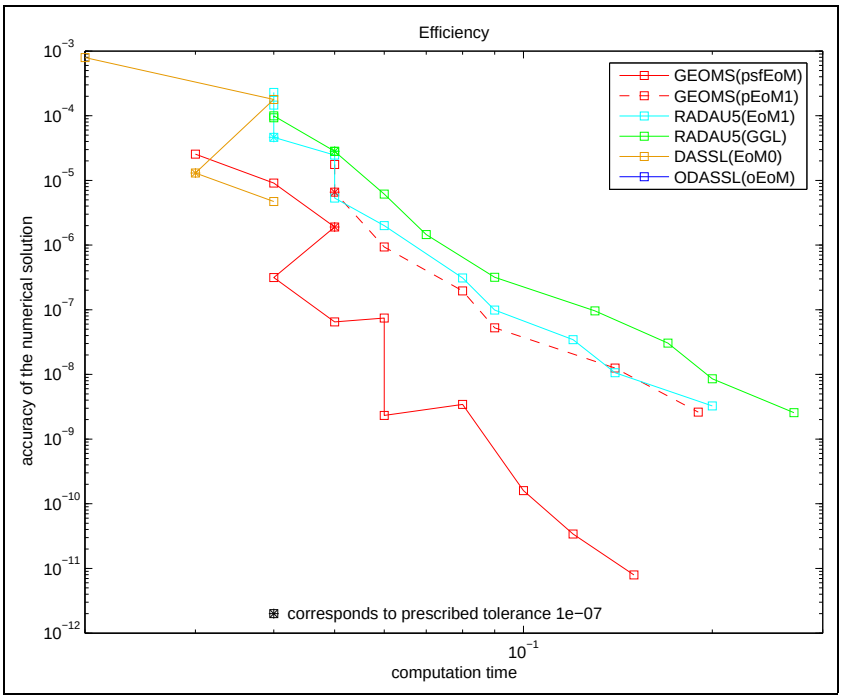

Figure 5.37: Academical example: Efficiency of the solvers with unstructured interface. Simulations are done on the time domain $\mathbb{I} = [0, 1.5]$.

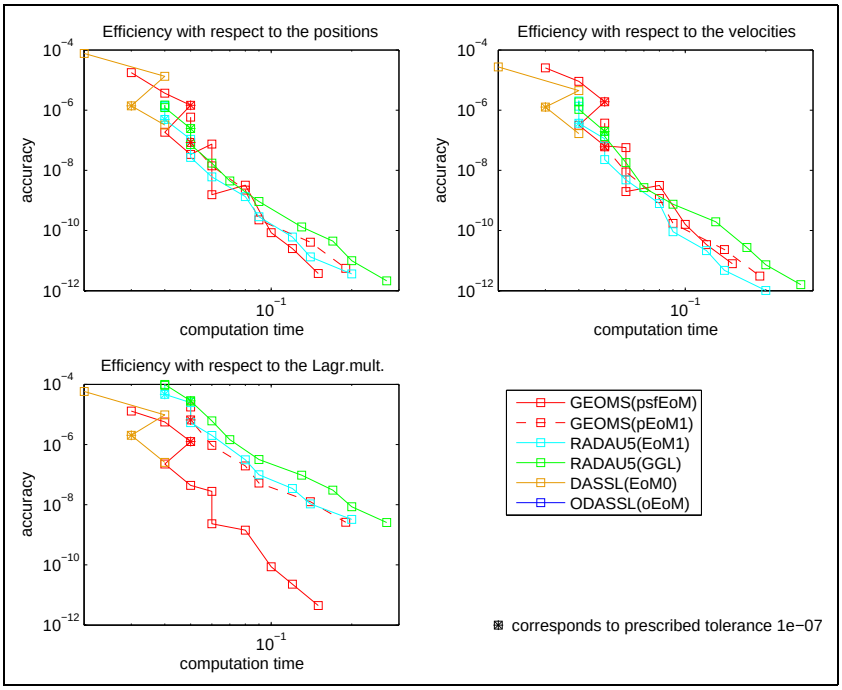

Figure 5.38: Academical example: Efficiency of the solvers with unstructured interface without respecting the Lagrange multipliers. Simulations are done on the time domain $\mathbb{I} = [0, 1.5]$.

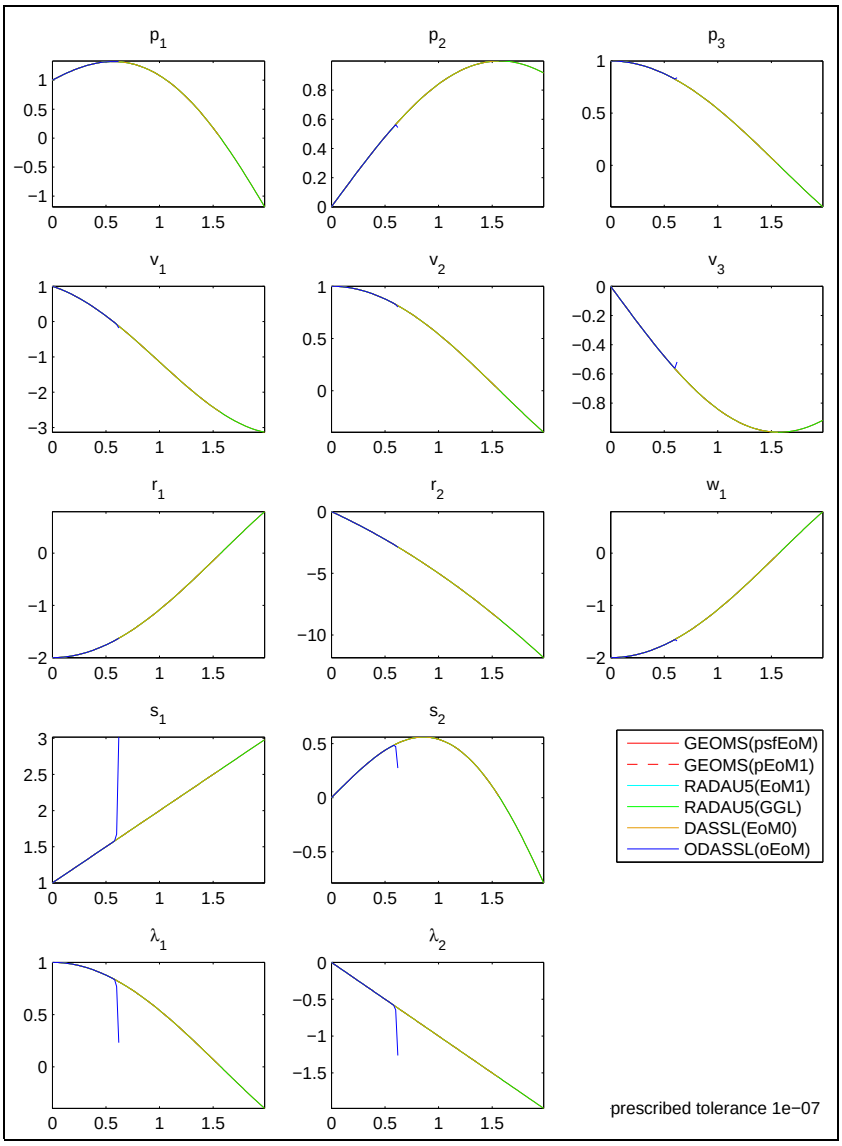

Figure 5.39: Academical example: Numerical solutions of the solvers based on residual evaluations. Simulations are done with the tolerance RTOL=ATOL=10^{-7} on the time domain $\mathbb{I} = [0, 2]$.

Note that the numerical algorithms based on structural evaluations used for the numerical experiments above are not applicable for nonholonomic systems. Furthermore, if the numerical integration by use of `MHERK5` was successful, the efficiency of the numerical algorithm `MHERK5` is comparable to the efficiency obtained with the numerical algorithms based on residual evaluations. However, the efficiency and the robustness of the numerical algorithm `MHERK3` was underwhelming.
According to the efficiency and the maximal reachable precision the new multibody system solvers `GEOMS` and `GMKSSOL` work very good in cases of long time integration (see Example 5.3.1 of the mathematical pendulum and the Example 5.3.4 of the slider crank) and stiff mechanical systems (see Example 5.3.2 of the lolly for large parameter β and the Example 5.3.4 of the slider crank with $l_1 \approx l_2$). Furthermore, the new multibody system solvers `GEOMS` and `GMKSSOL` are suitable for

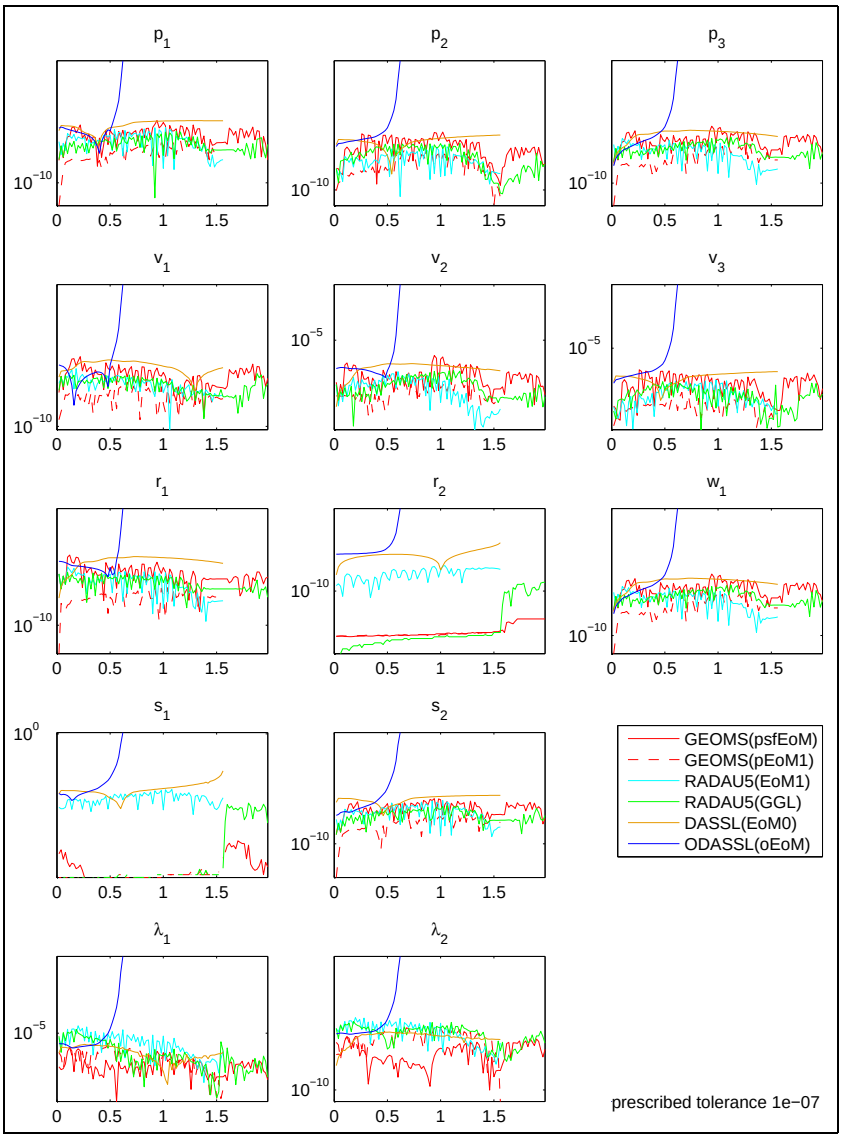

Figure 5.40: Academical example: Accuracy of the solvers based on residual evaluations. Simulations are done with the tolerance RTOL=ATOL=10^{-7} on the time domain $\mathbb{I} = [0, 2]$.

overdetermined systems, i.e., with redundant constraints (see Example 5.3.4 of the slider crank). GEOMS by use of the projected-strangeness-free formulation is a good choice for robust solution. Note that the integration of all numerical examples was successful for almost all prescribed tolerances RTOL=ATOL=10^i, $i = -3, ..., -16$. Furthermore, the numerical approximation of the Lagrange multipliers is very accurate and, in particular, more accurate than by use of any of the other solvers that are based on residual evaluations.

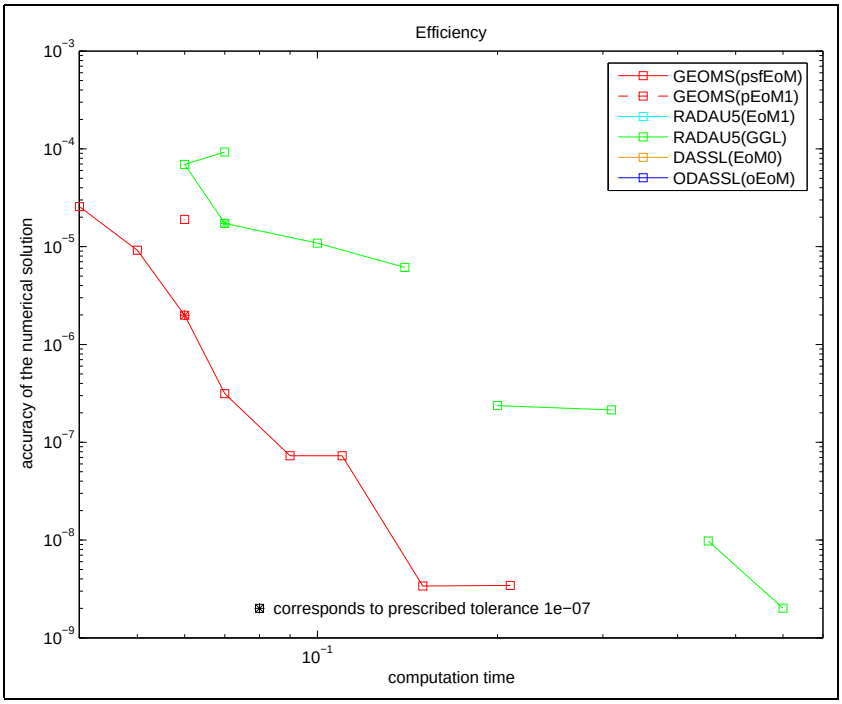

Figure 5.41: Academical example: Efficiency of the solvers with unstructured interface. Simulations are done on the time domain $\mathbb{I} = [0, 2]$.

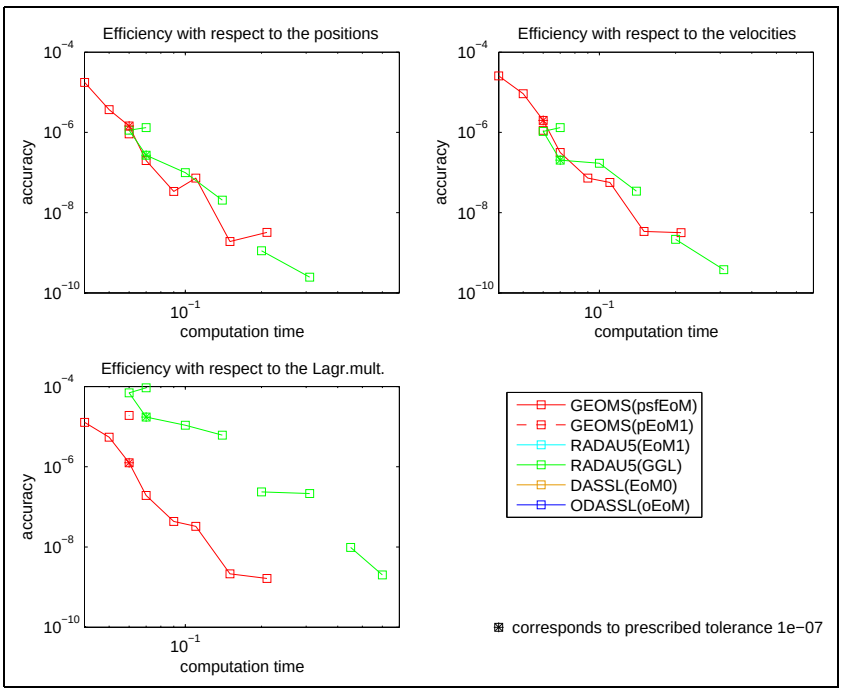

Figure 5.42: Academical example: Efficiency of the solvers with unstructured interface without respecting the Lagrange multipliers. Simulations are done on the time domain $\mathbb{I} = [0, 2]$.

Chapter 6

Summary

In this thesis we have focused on three topics. First, we have investigated quasi-linear differential-algebraic equations of arbitrary high index in view of their regularization and their numerical integration. Second, we have investigated the regularization and numerical integration of general nonlinear equations of motion as they arise in industrial applications. As a third topic we have developed and implemented two new numerical algorithms which perform the numerical integration of these equations of motion in an efficient and stable way.

In preparation of the treatment of quasi-linear differential-algebraic equations and of the equations of motion and, in particular, as a basis for the development of the new numerical algorithms, we have investigated smooth factorizations of matrix functions in Section 2.1. This is a very sensitive and important topic in the regularization of general nonlinear differential-algebraic equations. Therefore, in Theorem 2.1.6 we have presented the generalization of smooth decompositions of matrix functions which depend on several variables.
Furthermore, we have investigated systems of nonlinear equations in Section 2.3. We have taken into account redundant systems of nonlinear equations with respect to the numerical treatment of nonlinear differential-algebraic equations and, in particular, with respect to the numerical treatment of the equations of motion of mechanical systems.

We did not restrict our considerations of quasi-linear differential-algebraic equations to regular systems of differential-algebraic equations. Rather, we have investigated general over- and underdetermined quasi-linear differential-algebraic equations which are allowed to contain redundancies. In particular, we have presented in Procedure 3.5.11 a general tool for the analysis of quasi-linear differential-algebraic equations of arbitrarily high index. This procedure offers the possibility to determine all hidden constraints, the solution manifold, the (complete) minimal reduced derivative array, and the underlying differential equations in an efficient way. Based on the procedure we have defined a very important quantity of general quasi-linear differential-algebraic equations, the so-called maximal constraint level. This corresponds to the highest level of existing hidden constraints of the DAE and is identical with the minimal number of differentiations of the DAE which are restricting the solution.
In contrast to the strangeness-concept the developed procedure is restricted to quasi-linear differential-algebraic equations, but its application is less technical than the one of the strangeness-concept for quasi-linear DAEs because of its iterative nature and, in particular, because the procedure does only involve the derivatives of the necessary parts of the DAE, while the strangeness-concept is based on the

whole derivative array. Furthermore, the procedure also involves the treatment of quasi-linear differential-algebraic equations which do not satisfy constant-rank assumptions on the system matrices and the (hidden) constraints. In the case of constant rank of the system matrices the procedure is still less technical than in the case of nonconstant ranks in the system matrices.
The relation of the strangeness-concept and the procedure developed in Section 3.5 is investigated and it is shown if both concepts are applicable to the investigated DAE that the s-index is an upper bound of the maximal constraint level by different technical effort. Furthermore, based on the procedure we have presented a regularization technique in Section 3.5.3 for general quasi-linear differential-algebraic equations. This yields an equivalent semi-implicit DAE of minimum size with the same solution set, the so-called projected-strangeness-free formulation, see Theorem 3.5.52.
In Section 3.5.4 we have investigated the numerical treatment of quasi-linear differential-algebraic equations. We have developed a discretization technique based on a Runge-Kutta scheme in combination with the regularization technique developed in Section 3.5.3. We have shown that the sequence of regularization and discretization can be permuted under certain conditions which are defined in Theorem 3.5.75. The investigation of differential-algebraic equations is concluded with an overview over available and commonly used numerical algorithms designed for differential-algebraic equations.

The investigation of multibody systems ranges from modeling aspects in view of nonholonomic mechanical systems with possibly redundant constraints and regularity discussions up to the development of a simple regularization strategy based on the procedure developed in Section 3.5.2.
In this thesis we have investigated the modeling of mechanical systems in view of industrial applications on a very general level. In particular, we have extended the classification of the equations of motion given in [164] by two additional modeling levels including nonholonomic constraints and including possible redundancies in the constraints. We have investigated the equations of motion in their most general form including all relevant features appearing in industrial applications like hydraulic components, contact point condition, holonomic as well as nonholonomic constraints, redundancies in the constraints, and solution invariants.
We have stated assumptions on the equations of motion which insure that they are regular in some sense. Furthermore, we have investigated the existence and the uniqueness of solutions in view of these assumptions, in Theorems 4.2.32 and 4.2.33. In particular, the satisfaction of all (hidden) constraints is a very important topic in view of the numerical treatment of the equations of motion under investigation. In particular, we have compared several commonly used formulations of the equations of motion in view of the drift-off phenomenon. It is shown that, in principle, the explicit appearance of the hidden constraints is responsible for the fact that they are satisfied by the numerical solution.
Based on the results obtained so far we have stated in Section 4.5 two paradigms. The first one is the modeling paradigm concerning the modeling of mechanical systems which postulates that the modeling of dynamical systems has to be done in such a way that the model equations have to contain all important and necessary information as equations such that the numerical algorithms are able to handle them. The second paradigm, the algorithm paradigm, postulates that the numerical algorithms should be able to deal with all possible information which is important for the modeling of dynamical systems. Both paradigms together facilitate a robust and efficient numerical treatment of dynamical systems.
We have discussed commonly used regularization techniques for the equations of motion and we have discussed their extensions to the general form of the equations

of motion considered in this thesis. Furthermore, based on the investigations for quasi-linear differential-algebraic equations we have presented a simple regularization technique for general nonlinear equations of motion in Theorem 4.6.9. This leads to an equivalent formulation of DAEs which is strangeness-free and contains the same solution set as the original equations of motion. In particular, we carved out all hidden constraints and selected those differential equations from the differential part of the equations of motion which describe the dynamics in the solution manifold given by the hidden constraints. Furthermore, we have discussed in Theorem 4.6.15 an incomplete regularization of the equations of motion.

Based on the results obtained from the investigation of quasi-linear differential-algebraic equations and of the equations of motion, we have developed two numerical algorithms `GEOMS` and `GMKSSOL` for the numerical integration of general equations of motion. Based on the observation that the sequence of discretization and regularization can be permuted under certain conditions, the algorithms follow two different strategies. While `GEOMS` is based on the regularization of the discretized equations of motion, the algorithm `GMKSSOL` is based on the discretization of the regularized equations of motion.
In particular, the algorithm `GEOMS` is developed to perform the numerical integration of the most general form of the equations of motion, including nonholonomic constraints and possible redundancies in the constraints, as they may appear in industrial applications. Besides the numerical integration it offers some additional features like respecting invariant solutions, respecting hidden constraints, use of different decomposition strategies, use of an incomplete regularization, and also check and correction of the initial values with respect to their consistency. It should be mentioned that the algorithm `GEOMS` follows the demands stated in the algorithm paradigm such that `GEOMS` is able to respect all possible information like hidden constraints and solution invariants.
On the other hand, although the algorithm `GMKSSOL` is specialized to a certain type of equations of motion it still provides a certain measure of generality. This algorithm offers the possibility to check and to correct the initial values of the Lagrange multipliers with respect to its consistency. Furthermore, `GMKSSOL` respects hidden constraints and is able to deal with redundancies in the constraints.
Subsequently, we have demonstrated the performance and the applicability in comparison to commonly used numerical algorithms suitable for the numerical simulation of multibody systems at several mechanical examples of different degrees of complexity. The experience with the numerical examples in Section 5.3 and several other numerical tests suggest that the codes `RADAU5` by use of the s-index-1 formulation (4.81) and the code `GEOMS` by use the projected-strangeness-free formulation (4.114) are very efficient methods for the numerical integration of the equations of motion. This is justified by the fact that both codes have successfully finished almost all test scenarios with all prescribed tolerances `TOL`$=10^i$, $i = -3, ..., -15$. Furthermore, the numerical results suggest that the numerical algorithm `GEOMS` represents a general tool for the efficient and robust simulation of general multibody-systems as they appear in industrial applications.

In view of the treatment of differential-algebraic equations, an open problem is the precise specification of the relation of the maximal constraint level and the strangeness index. We conjecture that both quantities have the relations that in view of the Procedure 3.5.11

$$\nu_s = \begin{cases} \nu_c & \text{for } \operatorname{range}((\widetilde{E}^{\nu_c})^T) \cap \operatorname{range}((\widetilde{k}_{2,x}^{\nu_c})^T) = \{0\}, \\ \nu_c + 1 & \text{for } \operatorname{range}((\widetilde{E}^{\nu_c})^T) \cap \operatorname{range}((\widetilde{k}_{2,x}^{\nu_c})^T) \neq \{0\} \end{cases}$$

and that in view of the Hypothesis 3.2.7

$$\nu_c = \begin{cases} \nu_s & \text{for } Z_{2\nu_s}^T \neq 0, \\ \nu_s - 1 & \text{for } Z_{2\nu_s}^T = 0, \end{cases}$$

where $Z_{2\nu_s}^T$ of size $a \times m$ is the last block column of the matrix function

$$Z_2^T = \left[\begin{array}{cccc} Z_{20}^T & Z_{21}^T & \cdots & Z_{2\nu_s}^T \end{array} \right].$$

Furthermore, the developed procedure offers the extension of considerations to DAEs with singularities as they appear in DAEs for which the procedure terminates successfully but with $\operatorname{rank}(\widetilde{E}_1^\nu) \neq \text{const}$ or $\operatorname{rank}(\widetilde{k}_2^i) \neq \text{const}$ for $i = 0, ..., \nu$. Subsequently, the obtained results in this thesis can be used as basis for further investigations in view of the control of dynamical systems, in particular, of mechanical systems. For example this includes path control or optimal control of dynamical systems. Moreover, the code `GEOMS` can in principle be used for path control problems and can be generalized for the numerical treatment of general quasi-linear DAEs.

Appendix A

Basics

In this Chapter, we will review and discuss fundamentals associated with nonlinear functional analysis in Section A.1 and fundamentals associated with linear algebra in Section A.2. Subsequently, the manual for the use of the numerical algorithms `GEOMS` and `GMKSSOL` is stated in Sections B.1 and B.2, respectively.

A.1 Basic nonlinear functional analysis

The following notation is adapted from [47, 143, 183].

Definition A.1.1 (Function) *A set $\mathcal{R}_{\mathbb{X}\mathbb{Y}}$ of ordered pairs (x, y) with $x \in \mathbb{X} \subset \mathbb{R}^n$ and $y \in \mathbb{Y} \subset \mathbb{R}^m$ is called* function *if for every $x \in \mathbb{X}$ there exists one and only one y such that the pair $(x, y) \in \mathcal{R}_{\mathbb{X}\mathbb{Y}}$. Let us write $f : \mathbb{X} \to \mathbb{Y}$ and $y = f(x)$ if $(x, y) \in \mathcal{R}_{\mathbb{X}\mathbb{Y}}$.*

Definition A.1.2 (Surjective, injective, and bijective function) *A function $f : \mathbb{X} \to \mathbb{Y}$, $\mathbb{X} \subset \mathbb{R}^n$, $\mathbb{Y} \subset \mathbb{R}^m$ is called* surjective *or a* surjection *if $f(\mathbb{X}) = \mathbb{Y}$, i.e., if for every $y \in \mathbb{Y}$ there is (at least) one $x \in \mathbb{X}$ with $y = f(x)$. The function f is called* injective *or* injection *if the relation $f(x_1) = f(x_2)$ implies $x_1 = x_2$ for every $x_1, x_2 \in \mathbb{X}$. Furthermore, the function f is called* bijective *or* bijection *if it is both injective and surjective.*

Definition A.1.3 (Inverse function) *Let $f : \mathbb{X} \to \mathbb{Y}$ with $\mathbb{X} \subset \mathbb{R}^n$ and $\mathbb{Y} \subset \mathbb{R}^m$ be a bijective function then a function $g : \mathbb{Y} \to \mathbb{X}$ with $g(f(x)) = x$ is called the* inverse function *of f and is denoted by $g = f^{-1}$.*

An inverse function of f is not defined if the function f is not bijective. If f is bijective the equation $y = f(x)$ is equivalent to $f^{-1}(y) = x$. Furthermore, f^{-1} is bijective and we have $(f^{-1})^{-1} = f$.

Definition A.1.4 (Continuous function) *A function $f : \mathbb{X} \to \mathbb{R}^m$, $\mathbb{X} \subset \mathbb{R}^n$ is called* continuous *at a point $x_0 \in \mathbb{X}$ if, for every neighborhood V_2 of $f(x_0)$ in $\mathbb{R}^m$, there is a neighborhood V_1 of x_0 in $\mathbb{X}$ such that $f(V_1) \subset V_2$. The function f is called* continuous *(in $\mathbb{X}$) if it is continuous at every point of $\mathbb{X}$. Functions which are continuous are denoted by $f \in \mathcal{C}(\mathbb{X}, \mathbb{R}^m)$ or $f \in \mathcal{C}^0(\mathbb{X}, \mathbb{R}^m)$.*

Definition A.1.5 (Norm) *A function $||\cdot|| : \mathbb{X} \to \mathbb{R}$, $\mathbb{X} \in \mathbb{R}^n$ is called a* norm *if the conditions*

$$
\begin{aligned}
&1) \quad x \neq 0 \;\Rightarrow\; ||x|| > 0,\\
&2) \quad ||\alpha x|| = |\alpha|\ ||x||,\\
&3) \quad ||x + y|| \leq ||x|| + ||y||
\end{aligned}
$$

are satisfied for all $x, y \in \mathbb{X}$ *and* $\alpha \in \mathbb{R}$.

In the following we will base our investigations on the norm of matrices and vectors. We will restrict us to the so called *Hölder*[1] *norms* which are defined for $0 \leq p \leq \infty$ by

$$||x||_p = \Big(\sum_{i=1}^{n} |x_i|^p \Big)^{\frac{1}{p}}$$

for the vector $x = [x_i]_{i=1,\ldots,n} \in \mathbb{R}^n$. Of these the 1, 2, and ∞ norms are the most important, where $||x||_\infty = \max_{1 \leq i \leq n} |x_i|$. Furthermore, with respect to matrices we will use the associated norms defined for $0 \leq p \leq \infty$ by

$$||A||_p = \max_{x \in \mathbb{X}} \frac{||Ax||_p}{||x||_p}.$$

for the matrix $A \in \mathbb{R}^{m,n}$ and $\mathbb{X} \subset \mathbb{R}^n$. In our following investigations it is not necessary to distinguish between different p. Therefore, we will omit the suffix such that we will use $||\cdot||$ instead of $||\cdot||_p$.

Definition A.1.6 (Sphere, neighborhood) *Let* $x \in \mathbb{X} \subset \mathbb{R}^n$ *be given, then the set* $\mathbb{S}(x, \epsilon) = \{y \in \mathbb{X} \ : \ ||x - y|| < \epsilon\}$ *is called* open sphere *or* open neighborhood *of* x *with* $\epsilon > 0$. *Furthermore, the set* $\bar{\mathbb{S}}(x, \epsilon) = \{y \in \mathbb{X} \ : \ ||x - y|| \leq \epsilon\}$ *is called* closed sphere *or* closed neighborhood *of* x *with* $\epsilon \geq 0$.

Lemma A.1.7 *If* $f \ : \ \mathbb{X} \to \mathbb{R}$, $\mathbb{X} \subset \mathbb{R}^n$ *is continuous in* $x_0 \in \mathbb{X}$ *and* $f(x_0) > a$ $(< a)$. *Then there exists an* $\epsilon > 0$ *such that* $f(x) > a$ $(< a)$ *for all* $x \in \{x \in \mathbb{X} \ : \ ||x - x_0|| < \epsilon\}$.

Proof: See [87]. □

Definition A.1.8 (Lipschitz continuous) *A function* $f \ : \ \mathbb{X} \to \mathbb{R}^m$, $\mathbb{X} \subset \mathbb{R}^n$ *which satisfies the condition*

$$||f(x) - f(y)|| \leq L||x - y|| \tag{1.1}$$

for all $x \in \mathbb{X}$ *and* $y \in \mathbb{X}$, *where* L *is a constant independent of* x *and* y, *is called* Lipschitz continuous *or* Lipschitz function. *The condition (1.1) is called* Lipschitz condition *and the constant* L *is called the* Lipschitz constant.

Definition A.1.9 (Differentiable function) *A function* $f \ : \ \mathbb{X} \to \mathbb{R}^m$ *with* $\mathbb{X} \subset \mathbb{R}^n$ *is called* differentiable at the point $x_0 \in \mathbb{X}$ *if there exists a linear function* $u \ : \ h \mapsto A(x_0)h$ *with* $A(x_0) \in \mathbb{R}^{m,n}$ *such that*

$$\lim_{h \to 0} \frac{||f(x_0 + h) - f(x_0) - u(h)||}{||h||} = 0.$$

If f *is differentiable at every point* $x_0 \in \mathbb{X}$ *the function* f *is called* differentiable on $\mathbb{X}$ *or* differentiable *in short. Furthermore,* $A(x_0)$ *is called the* derivative of f at the point x_0 *and if* f *is differentiable at* $\mathbb{X}$ *the function* $A \ : \ \mathbb{X} \to \mathbb{R}^{m,n}$ *is called* derivative of f. *The derivative of the function* f *(with respect to* x*) is denoted by* $f_{,x}$.

[1] Otto Ludwig Hölder (1859 in Stuttgart, Germany - 1937 in Leipzig, Germany)

Definition A.1.10 (Continuously differentiable) *A differentiable function $f : \mathbb{X} \to \mathbb{R}^m$, $\mathbb{X} \subset \mathbb{R}^n$ is called a* continuously differentiable function *if the derivative $f_{,x}$ of f (see Definition A.1.9) is continuous in $\mathbb{X}$. The set of continuously differentiable functions from $\mathbb{X}$ into $\mathbb{R}^m$ is denoted by $\mathcal{C}^1(\mathbb{X}, \mathbb{R}^m)$.*
A function $f : \mathbb{X} \to \mathbb{R}^m$, $\mathbb{X} \subset \mathbb{R}^n$ is called l-times continuously differentiable or also $\mathcal{C}^l$-function if the derivative $f_{,x}$ of f is an $(l-1)$-times continuously differentiable function from $\mathbb{X}$ into $\mathbb{R}^m$. The set of l-times continuously differentiable functions is denoted by $\mathcal{C}^l(\mathbb{X}, \mathbb{R}^m)$. Furthermore, the set $\mathcal{C}^\infty(\mathbb{X}, \mathbb{R}^m) = \cap_{l=1}^{\infty} \mathcal{C}^l(\mathbb{X}, \mathbb{R}^m)$ is called the set of infinitely continuously differentiable functions.

In view of the asymptotic behavior of functions, we will use the so called Landau[2] symbolic as defined as follows.

Definition A.1.11 (Landau symbol) *Let $f : \mathbb{X} \to \mathbb{R}$ and $g : \mathbb{X} \to \mathbb{R}$. Then the relation*

$$\limsup_{x \to a} \left| \frac{f(x)}{g(x)} \right| < \infty$$

is denoted by the Landau symbol $\mathcal{O}$ *in the form*

$$f(x) = \mathcal{O}(g(x)).$$

Definition A.1.12 (Homeomorphism) *A function $f : \mathbb{X} \to \mathbb{R}^m$ with $\mathbb{X} \subset \mathbb{R}^n$ is called a* homeomorphism *if f is a bijection and both f and its inverse function f^{-1} are continuous.*

Definition A.1.13 (Diffeomorphism, diffeomorphic) *Let $0 \leq l \leq \infty$. A function $f : \mathbb{X} \to \mathbb{R}^m$ with $\mathbb{X} \subset \mathbb{R}^n$ is called a* $\mathcal{C}^l$-diffeomorphism *if f is bijective and both f and its inverse function f^{-1} are $\mathcal{C}^l$-functions. Two subsets $\mathbb{X}$ and $\mathbb{Y}$ are said to be* $\mathcal{C}^l$-diffeomorphic *if there exists a $\mathcal{C}^l$-diffeomorphism $f : \mathbb{X} \to \mathbb{Y}$ between them.*

Lemma A.1.14 (Heine[3]-Borel[4]-Lebesgue[5]Covering Theorem) *Let $\mathbb{X}$ be a compact subset of $\mathbb{R}^n$ and let $\mathcal{B}$ be a collection of open subsets of $\mathbb{R}^n$ such that every point of $\mathbb{X}$ belongs to at least one of the subsets of $\mathcal{B}$. Then $\mathbb{X}$ is covered by a finite number of open sets of $\mathcal{B}$.*

Proof: See [182]. □

A.2 Basic linear algebra

In this section we will briefly introduce some frequently used aspects and notation from linear algebra, numerical linear algebra, and tensor algebra. For more details the reader is referred to [72, 92, 95, 158, 171, 176].
We start with the introduction of important properties concerning (sub)spaces and matrices.

Definition A.2.1 (Orthogonal complement of subspaces) *The* orthogonal complement of a subspace $\mathbb{S} \in \mathbb{R}^n$ *is defined by* $\mathbb{S}^\perp = \{y \in \mathbb{R}^n : y^T x = 0 \text{ for all } x \in \mathbb{S}\}$.

[2] Edmund Landau (1877 Berlin, Germany - 1938 Berlin, Germany)
[3] Heinrich Eduard Heine (1821 in Berlin, Germany - 1881 in Halle, Germany)
[4] Félix Edouard Justin Émile Borel (1871 in Saint Affrique, France - 1956 in Paris, France)
[5] Henri Léon Lebesgue (1875 in Beauvais, France - 1941 in Paris, France)

Definition A.2.2 (Minor and adjugate) *Let $A=[A_{ij}] \in \mathbb{R}^{m,n}$. Let $I=[i_1,...,i_r]$ and $J=[j_1,...,j_r]$, $\min(m,n) \geq r \in \mathbb{N}_0$ be index vectors with $i_k \neq i_l$ and $j_k \neq j_l$ for $k \neq l$, $i_l \in \{1,...,m\}$ and $j_l \in \{1,...,n\}$ for $l=1,...,r$, then the r-th order* minor $[A]_{IJ} \in \mathbb{R}$ *defined by I and J is*

$$[A]_{IJ} = \det(A_{IJ})$$

with A_{IJ} introduced in Notation 2.0.3. If $m=n$ then $\mathrm{adj}(A)_{ij} = (-1)^{i+j}[A]_{IJ}$ *with $I=[1,...,i-1,i+1,...,n]$ and $J=[1,...,j-1,j+1,...,n]$ is called* adjugate.

Note that any minor $[A]_{IJ}$ as well as any adjugate $\mathrm{adj}(A)_{ij}$ depend smoothly on the entries a_{ij} of A.

Definition A.2.3 (Kernel, cokernel, range and corange) *Let $A \in \mathbb{R}^{m,n}$. The kernel, cokernel, range, and corange of A are defined by*

$$\begin{aligned}
\text{kernel } &\textit{of } A: \ \ker(A) = \{x \in \mathbb{R}^n \ : \ Ax = 0\},\\
\text{cokernel } &\textit{of } A: \ \mathrm{coker}(A) = (\ker(A))^{\perp},\\
\text{range } &\textit{of } A: \ \mathrm{range}(A) = \{y \in \mathbb{R}^m \ : \ \textit{there exists an } x \in \mathbb{R}^n \textit{ such that } y = Ax\},\\
\text{corange } &\textit{of } A: \ \mathrm{corange}(A) = (\mathrm{range}(A))^{\perp}.
\end{aligned}$$

Lemma A.2.4 *Let $A \in \mathbb{R}^{m,n}$, then*

$$\mathrm{coker}(A) = \mathrm{range}(A^T) \quad \textit{and} \quad \mathrm{corange}(A) = \ker(A^T).$$

Proof: See [72] □

Definition A.2.5 (Rank and corank) *Let $A \in \mathbb{R}^{m,n}$. The* rank *of matrix A is the integer*

$$\mathrm{rank}(A) = \dim(\mathrm{range}(A)),$$

where $\dim(\mathbb{S})$ *denotes the dimension of a subspace $\mathbb{S}$. The* corank *is the dimension of the corange of A, i.e.,*

$$\mathrm{corank}(A) = m - \mathrm{rank}(A).$$

Definition A.2.6 (Orthogonal matrix) *A matrix $A \in \mathbb{R}^{n,n}$ satisfying $A^TA = I_n$ is called an* orthogonal matrix.

Definition A.2.7 (Nonsingular matrix and inverse matrix) *Let the matrix $A \in \mathbb{R}^{n,n}$. If a uniquely determined matrix $X \in \mathbb{R}^{n,n}$ exists with $AX = XA = I_n$, where the matrix $I_n \in \mathbb{R}^{n,n}$ denotes the identity matrix, then the matrix A is called* nonsingular *or* invertible *and the matrix X is called* inverse matrix *of A or the* inverse *of A in short. The inverse of A is denoted by $A^{-1} = X$.*

Lemma A.2.8 *Let the matrix $A \in \mathbb{R}^{n,n}$ be nonsingular, then*

$$A^{-1} = \frac{1}{\det(A)} \begin{bmatrix} \mathrm{adj}(A)_{11} & \mathrm{adj}(A)_{21} & \cdots & \mathrm{adj}(A)_{n1} \\ \mathrm{adj}(A)_{12} & \mathrm{adj}(A)_{22} & \cdots & \mathrm{adj}(A)_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{adj}(A)_{1n} & \mathrm{adj}(A)_{2n} & \cdots & \mathrm{adj}(A)_{nn} \end{bmatrix}.$$

Proof: See [92]. □

Definition A.2.9 (Orthogonal projection) *Let $\mathbb{S} \subset \mathbb{R}^n$ be a subspace. A matrix $P_{\mathbb{S}} \in \mathbb{R}^{n,n}$ is called* orthogonal projection *onto $\mathbb{S}$ if* $\text{range}(P_{\mathbb{S}}) = \mathbb{S}$, $P_{\mathbb{S}}^2 = P_{\mathbb{S}}$, *and $P_{\mathbb{S}}^T = P_{\mathbb{S}}$.*

Remark A.2.10 From Definition A.2.9 it follows that $I - P_{\mathbb{S}}$ is the orthogonal projection onto $\mathbb{S}^{\perp}$. Furthermore, the orthogonal projection onto a subspace $\mathbb{S}$ is unique. See [72]. □

Lemma A.2.11 *Let $\mathbb{S} \subset \mathbb{R}^m$ be a subspace and let the columns of the matrix $A \in \mathbb{R}^{m,n}$ with full (column) rank span this subspace, i.e.,* $\text{range}(A) = \mathbb{S}$. *Then the orthogonal projection $P_{\mathbb{S}} \in \mathbb{R}^{m,m}$ onto $\mathbb{S}$ is given by*

$$P_{\mathbb{S}} = A(A^T A)^{-1} A^T. \tag{1.2}$$

Proof: See [95]. □

Important tools for the investigation of matrices are matrix decompositions, in particular, the *LU decomposition* and the *QR decomposition*. For more details we refer to [72, 92, 171]. Furthermore, a very important tool for investigations of rank decisions, redundancies of systems of nonlinear equations and others is the *singular value decomposition* (SV decomposition).

Lemma A.2.12 (Singular value decomposition) *Let $A \in \mathbb{R}^{m,n}$. Then there exist orthogonal matrices $U \in \mathbb{R}^{m,m}$ and $V \in \mathbb{R}^{n,n}$ such that*

$$U^T A V = \Sigma = \begin{bmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{bmatrix},$$

where $\Sigma_1 = \text{diag}(\sigma_1, ..., \sigma_r) \in \mathbb{R}^{r,r}$ and $\sigma_1 \geq ... \geq \sigma_r > 0$.

Proof: See [72]. □

Note that, in general, the orthogonal matrices U and V in the SV decomposition are not uniquely determined.
Another useful decomposition that follows immediately from the SV decomposition is the *polar decomposition*.

Lemma A.2.13 (Polar decomposition) *Let $A \in \mathbb{R}^{m,n}$ be given.*
a)If $m \leq n$, then $A = PY$, where $P \in \mathbb{R}^{m,m}$ is symmetric positive semi-definite, $P^2 = AA^T$, and $Y \in \mathbb{R}^{m,n}$ has orthonormal rows.
b)If $m \geq n$, then $A = XQ$, where $Q \in \mathbb{R}^{n,n}$ is symmetric positive semi-definite $Q^2 = A^T A$, and $X \in \mathbb{R}^{m,n}$ has orthonormal columns.
c)If $m = n$, then $A = PU = UQ$, where $P, Q \in \mathbb{R}^{n,n}$ are symmetric positive semi-definite $P^2 = AA^T$, $Q^2 = A^T A$, and $U \in \mathbb{R}^{n,n}$ is orthogonal.

Proof: See [92]. □

In all cases, the symmetric positive semidefinite factors P and Q are uniquely determined by A.
The SV decomposition provides a tool which allows the computation of the orthogonal projections onto the range, corange, kernel, and cokernel of a matrix A.

Lemma A.2.14 *Let $A \in \mathbb{R}^{m,n}$ with* $\text{rank}(A) = r$. *Suppose that $A = U\Sigma V^T \in \mathbb{R}^{m,n}$ is a SV decomposition of A. If we have the partitioning $U = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \in \mathbb{R}^{m,m}$ with $U_1 \in \mathbb{R}^{m,r}$, $U_2 \in \mathbb{R}^{m,m-r}$ and $V = \begin{bmatrix} V_1 & V_2 \end{bmatrix} \in \mathbb{R}^{n,n}$ with $V_1 \in \mathbb{R}^{n,r}$, $V_2 \in \mathbb{R}^{n,n-r}$ then*

$$\begin{aligned} P_{\text{range}(A)} &= U_1 U_1^T, & P_{\text{corange}(A)} &= U_2 U_2^T, \\ P_{\text{ker}(A)} &= V_2 V_2^T, & P_{\text{coker}(A)} &= V_1 V_1^T. \end{aligned} \tag{1.3}$$

Proof: See [72]. □

Note that the orthogonal projections are uniquely determined by U and V although the matrices U and V are in general not unique.
A generalization of the inverse of a matrix is given by the Moore[6]-Penrose pseudo-inverse, see [72].

Definition A.2.15 (Moore-Penrose pseudo-inverse) *Let $A \in \mathbb{R}^{m,n}$. The* Moore-Penrose pseudo-inverse *of A is defined to be the unique matrix $A^+ \in \mathbb{R}^{n,m}$ that satisfies the four Moore-Penrose conditions*

$$\begin{array}{llllll} (i) & AA^+A = A, & (iii) & (AA^+)^T = AA^+, \\ (ii) & A^+AA^+ = A^+, & (iv) & (A^+A)^T = A^+A. \end{array}$$

If $\operatorname{rank}(A) = n$, then $A^+ = (A^TA)^{-1}A^T$, while if $\operatorname{rank}(A) = n = m$, then $A^+ = A^{-1}$.

Lemma A.2.16 *Let $A \in \mathbb{R}^{m,n}$. Then*

$$\begin{array}{rclrcl} P_{\text{range}(A)} & = & AA^+, & P_{\text{corange}(A)} & = & I - AA^+, \\ P_{\ker(A)} & = & I - A^+A, & P_{\text{coker}(A)} & = & A^+A. \end{array} \tag{1.4}$$

Proof: See [72, 176]. □

Lemma A.2.17 *Let $E \in \mathbb{R}^{m,n}$, $A \in \mathbb{R}^{a,n}$, $m \geq n$, $a \leq n$. Let A have full (row) rank, i.e.,* $\operatorname{rank}(A) = a$. *Furthermore, let the matrix $\begin{bmatrix} E \\ A \end{bmatrix}$ have full (column) rank. Then, there exists a matrix $T \in \mathbb{R}^{n,d}$, with $d = n - a$ such that $AT = 0$ and* $\operatorname{rank}(ET) = d$. *Furthermore, there exists a matrix $S \in \mathbb{R}^{m,d}$ such that* $\operatorname{rank}(S^TET) = d$.

Proof: Because of the full (row) rank of A from the SV decomposition of A we get

$$A \begin{bmatrix} V_1 & V_2 \end{bmatrix} = \begin{bmatrix} \Sigma_A & 0 \end{bmatrix},$$

with $\Sigma_A \in \mathbb{R}^{a,a}$ nonsingular, $V_1 \in \mathbb{R}^{n,a}$, and $V_2 \in \mathbb{R}^{n,d}$. Because of the full (column) rank of $\begin{bmatrix} E \\ A \end{bmatrix}$ and because of $AV_2 = 0$ we get the full (column) rank of $EV_2 \in \mathbb{R}^{m,d}$, i.e., $\operatorname{rank}(EV_2) = d$, and from the SV decomposition of EV_2 we get

$$\begin{bmatrix} U_1^T \\ U_2^T \end{bmatrix} EV_2 = \begin{bmatrix} \Sigma_E \\ 0 \end{bmatrix},$$

with $\Sigma_E \in \mathbb{R}^{d,d}$ nonsingular, $U_1 \in \mathbb{R}^{m,d}$, and, $U_2 \in \mathbb{R}^{m,m-d}$. Choosing $T = V_2$ and $S = U_1$ yields the assertion. □

Lemma A.2.18 *Let the matrices $A \in \mathbb{R}^{m_1,n}$, $B \in \mathbb{R}^{m_2,n}$ be given such that the matrix $\begin{bmatrix} A^T & B^T \end{bmatrix}^T$ is nonsingular and $BA^T = 0$. Then for every positive or negative definite matrix $M \in \mathbb{R}^{n,n}$ the matrix*

$$\begin{bmatrix} AM \\ B \end{bmatrix}$$

is nonsingular.

[6] Eliakim Hastings Moore (1862 in Marietta, USA - 1932 in Chicago, USA)

Proof: We have

$$\begin{bmatrix} AM \\ B \end{bmatrix} \begin{bmatrix} A^T & B^T \end{bmatrix} = \begin{bmatrix} AMA^T & AMB^T \\ BA^T & BB^T \end{bmatrix} = \begin{bmatrix} AMA^T & AMB^T \\ 0 & BB^T \end{bmatrix}. \quad (1.5)$$

From the definiteness of M we get that AMA^T is also definite, i.e., in particular, AMA^T is nonsingular. Furthermore, BB^T is positive definite and therefore also nonsingular such that the matrix (1.5) is nonsingular. Then from the nonsingularity of $\begin{bmatrix} A^T & B^T \end{bmatrix}$ we get the assertion. □

Concluding this section, we will introduce the so called Kronecker product which will be turned out as a very practical notation, in particular, in the investigation of the numerical treatment of DAEs.

Definition A.2.19 (Kronecker product) *Let $A = [a_{ij}] \in \mathbb{R}^{m,n}$ and $B = [b_{kl}] \in \mathbb{R}^{p,q}$. The* Kronecker product *of A and B is defined by*

$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix} \in \mathbb{R}^{mp,nq}.$$

Lemma A.2.20 *Let $A \in \mathbb{R}^{m,n}$, $B \in \mathbb{R}^{l,p}$ $C \in \mathbb{R}^{n,q}$, and $D \in \mathbb{R}^{p,s}$. Then*

$$(A \otimes B)(C \otimes D) = (AC \otimes BD).$$

If, in addition, $m = n$ and A is nonsingular, then

$$(A \otimes I)^{-1} = (A^{-1} \otimes I).$$

Proof: The proof follows immediately from Definition A.2.19. □

Appendix B

Manuals

B.1 Manual of GEOMS

```
      SUBROUTINE GEOMS(
     #   NP,NV,NR,NW,NS,NL,NM,NI,M,N,NIVCOND,
     #   X,T,TEND,H,RTOL,ATOL,ITOL,IOPT,ROPT,
     #   IVCOND,EOM,MAS,JAC,IJAC,
     #   SOLOUT,IOUT,
     #   LIWORK,IWORK,LRWORK,RWORK,
     #   RPAR,IPAR,IERR,
     #   IDID)
C -----------------------------------------------------------------------
C
C  NAME      : (G)eneral (E)quations (O)f (M)otion (S)olver
C
C  PURPOSE   : This subroutine performs the numerical simulation
C              of a multibody system whose state is described by
C
C              p - position variables                  of dimension NP,
C              v - velocity variables                  of dimension NV,
C              r - dynamical force element variables   of dimension NR,
C              w - auxiliary variables                 of dimension NW,
C              s - contact point variables             of dimension NS,
C              l - holonomic Lagrange multipliers      of dimension NL,
C              m - nonholonomic Lagrange multipliers   of dimension NM
C
C              by numerical integration of the equations of motion
C              of the form
C
C                        p'= Z(p)*v,                              (1) (f_kin)
C                  M(p,t)*v'= f(p,v,r,w,s,l,m,t)-ZT(p)*GT(p,s,t)*l
C                                             -ZT(p)*HT(p,s,t)*m, (2) (f_dyn)
C                        r'= b(p,v,r,w,s,l,m,t),                  (3)
C                        0 = d(p,v,r,w,s,l,m,t),                  (4)
C                        0 = c(p,s,t),                            (5)
C                        0 = H(p,s,t)Z(p)v+h(p,s,t)               (6)
C                        0 = g(p,s,t),                            (7)
C                        0 = e(p,v,s,t)                           (8)
C
C              on the domain [t_0,t_f]=[T,TEND].
C
C              The prime denotes the time derivative, e.g., p'=dp/dt, and the
C              'T' following a matrix or vector denotes the transpose of this
C              matrix or vector, e.g.,GT is the  transpose of G and ZT is the
C              transpose of Z. Furthermore, the equations correspond to
C
C              (1) Kinematical equations of motion      of dimension NP,
C              (2) Dynamical equations of motion        of dimension NV,
C              (3) Dynamical force element equations    of dimension NR,
```

```
C             (4) Additional equations for variables w of dimension NW,
C             (5) Contact equations                    of dimension NS,
C             (6) Nonholonomic constraints             of dimension NM,
C                   Notation: h~(p,v,s,t)=H(p,s,t)Z(p)v+h(p,s,t)
C             (7) Holonomic constraints                of dimension NL,
C             (8) Solution invariants                  of dimension NI.
C
C             The System (1)-(8) has to satisfy the following.
C              a) G = dg/dp - dg/ds*(dc/ds)^{-1}*dc/dp.
C              b)      [ GZM^{-1}Gl   GZM^{-1}Hm]
C                 rank([                       ])=rank(G)+rank(H)=constant
C                      [ HZM^{-1}Gl   HZM^{-1}Hm]
C                 with Gl=ZT*GT-df/dl+df/dw*(dd/dw)^{-1}*dd/dl
C                 and  Hm=ZT*HT-df/dm+df/dw*(dd/dw)^{-1}*dd/dm
C                 for all t in [T,TEND].
C                 Alternatively,
C                      [ M  Gl Gm ]
C                 rank([ GZ 0  0  ])=NV+rank(G)+rank(H)
C                      [ HZ 0  0  ]
C                 has to be satisfied for all t in [T,TEND].
C              c) dc/ds has to be nonsingular for all times t in [T,TEND].
C              d) dd/dw has to be nonsingular for all times t in [T,TEND].
C              e) de/dv has to have full rank for all times t in [T,TEND].
C
C             The integration method used is the implicit Runge-Kutta method
C             (Radau IIa) of order 5 with step size control, continuous
C             output, and consistent initialization.
C
C  METHOD   :  The equations of motion are integrated by the implicit
C              Runge-Kutta method of type RADAU IIa of order 5 and using the
C              projected-strangeness-free formulation or the
C              projected-strangeness-index-1 formulation of the equations of
C              motion.
C
C  VERSION  : July 31, 2005
C
C  REVISIONS : -
C
C  AUTHORS  : Address: A. Steinbrecher
C                      Institut fuer Mathematik
C                      Technische Universitaet Berlin
C                      Strasse des 17. Juni
C                      10623 Berlin, Germany
C             e-mail:  steinbrecher@math.tu-berlin.de
C
C  REFERENCES: This code is part of the PhD thesis:
C              A.Steinbrecher. Numerical Solution of Quasi-Linear Differential-
C              Algebraic Equations and Industrial Simulation of Multibody
C              Systems. PhD thesis, TU Berlin, Institut fuer Mathematik, 2005
C
C  KEYWORDS  : numerical simulation of mechanical systems, equations of motion,
C              differential-algebraic equations, projected-strangeness-free
C              formulation, projected-strangeness-index-1 formulation
C
C  NOTE      : The (basic) linear algebra routines are provided by the
C              libraries BLAS and LAPACK
C
C -----------------------------------------------------------------------------
C
C  CALL
C  --------------------------
C
C      SUBROUTINE GEOMS(
C     #   NP,NV,NR,NW,NS,NL,NM,NI,M,N,NIVCOND,
C     #   X,T,TEND,H,RTOL,ATOL,ITOL,IOPT,ROPT,
C     #   IVCOND,EOM,MAS,JAC,IJAC,
C     #   SOLOUT,IOUT,
```

```
C     #   LIWORK,IWORK,LRWORK,RWORK,
C     #   RPAR,IPAR,IERR,
C     #   IDID)
C      IMPLICIT NONE
C      INTEGER          NP,NV,NR,NW,NS,NL,NM,NI,M,N,NIVCOND,
C     #                 ITOL,IJAC,IOUT,LIWORK,LRWORK,IERR,IDID,
C     #                 IOPT(40),IWORK(LIWORK),IPAR(*)
C      DOUBLE PRECISION T,TEND,H,
C     #                 X(N),RTOL(*),ATOL(*),ROPT(40),RWORK(LRWORK),
C                      RPAR(*)
C      EXTERNAL         IVCOND,EOM,MAS,JAC,SOLOUT
C
C  INPUT- AND OUTPUT-ARGUMENTS
C  ---------------------------
C
C     NP      Input  : integer
C             Number of position variables p.
C
C     NV      Input  : integer
C             Number of velocity variables v.
C
C     NR      Input  : integer
C             Number of dynamical force element variables r.
C
C     NW      Input  : integer
C             Number of auxiliary variables w.
C
C     NS      Input  : integer
C             Number of contact point variables s.
C
C     NL      Input  : integer
C             Number of Lagrange multipliers l=lambda for holonomic
C               constraints.
C
C     NM      Input  : integer
C             Number of Lagrange multipliers m=mu for nonholonomic
C               constraints.
C
C     NI      Input  : integer
C             Number of invariants, e.g., energy conservation.
C
C     M       Input  : integer
C             Total number of provided equations (M.GE.N), i.e., dimension of
C             RDA, see subroutine EOM. In the case of the use of the
C             * projected-strangeness-free formulation we have
C                 M=NP+NV+NR+NW+NS+3*NL+2*NM+NI,
C             * projected-strangeness-index-1 formulation we have
C                 M=NP+NV+NR+NW+NS+2*NL+NM+NI.
C
C     N       Input  : integer
C             Number of unknowns (M.GE.N), i.e., dimension of X. We have
C             N=NP+NV+NR+NW+NS+NL+NM.
C
C     NIVCOND Input  : integer
C             Number of initial  value conditions, which have to be satisfied
C             in addition to the constraints obtained from the provided equa-
C             tions of motion. See subroutine IVCOND.
C
C     X       Input  : double precision array X(N)
C             Initial values for X. The array X contains  the (initial) state
C             of the mechanical system in the following order
C
C                 X(1:NW)                                    =w
C                 X(NW+1:NW+NL)                              =l (=lambda)
C                 X(NW+NL+1:NW+NL+NM)                        =m (=mu)
C                 ---------------------------------------------------
C                 X(NL+NM+NW+1:NL+NM+NW+NR)                  =r
```

```
C                ---------------------------------------------------------
C                X(NL+NM+NW+NR+1:NL+NM+NW+NR+NV)             =v
C                ---------------------------------------------------------
C                X(NL+NM+NW+NR+NV+1:NL+NM+NW+NR+NV+NS)       =s
C                X(NL+NM+NW+NR+NV+NS+1:NL+NM+NW+NR+NV+NS+NP) =p
C
C            Output :
C            Numerical approximation of the solution at the last successfully
C            reached time T.
C
C     T      Input  : double precision
C            Initial time.
C            Output :
C            Last successfully reached time. If the whole integration was
C              successful then T=TEND.
C
C     TEND   Input  : double precision
C            Final time.
C
C     H      Input  : double precision
C            Initial step size.
C            Output :
C            Last used step size.
C
C     RTOL   Input  : double precision RTOL (or array RTOL(N))
C     ATOL   Input  : double precision ATOL (or array ATOL(N))
C            Relative and absolute error tolerances. They can be both
C              scalars or else both vectors of length N.
C            In the case of a scalar the prescribed relative and absolute
C              tolerances are valid for every component of the vector of
C              unknowns X. The code keeps, roughly, the local error of X(I)
C              below RTOL*ABS(X(I))+ATOL.
C            In the case of a vector of dimension N the prescribed relative
C              tolerances RTOL(I) and absolute tolerances ATOL(I) are valid
C              for the I-th component X(I) of the vector of unknowns X.
C              The code keeps, roughly, the local error of X(I) below
C              RTOL(I)*ABS(Y(I))+ATOL(I).
C
C     ITOL   Input  : integer
C            Switch for RTOL and ATOL:
C            ITOL=0  Both RTOL and ATOL are scalars.
C            ITOL=1  Both RTOL and ATOL are vectors.
C
C     IOPT   Input : integer array IOPT(40)
C            Serve as parameters for the code. For standard use of the code
C            IOPT(2),..,IOPT(17) must be set to zero before calling.
C            See below for a more sophisticated use.
C
C            IOPT( 2)=LUN output device
C                     0  - no output (default)
C                     6  - output to the screen
C                    >10 - other output devices (to define)
C                     In the case that the output of several messages is de-
C                     sired, the user has to define an  output device and to
C                     associate this device with IOPT(2), e.g.,
C                       IOPT(2)=13
C                       OPEN(UNIT=13,FILE='geoms.log')
C                     Finally, the output device has to be closed, e.g.,
C                       CLOSE(13)
C                     In the case of an unsuccessful  run of GEOMS it is re-
C                     commended  to set IOPT(2) > 0 such  that GEOMS is able
C                     to provide more detailed informations to the user.
C                     Furthermore, it is recommended to set
C                       IOPT(2)=0, 6, or >10.
C
C            IOPT( 3)=NIT maximum number of  Newton iterations for the solu-
C                     tion of the implicit system in each step.
```

```
C                     The default value (for IOPT(3)=0) is 10.
C
C            IOPT( 4)=STARTN defines the choice  of starting  values for the
C                     Newton method solving the nonlinear stage equations
C                     0 - The extrapolated collocation solution is  taken as
C                         starting value for Newton method. (default)
C                     1 - Zero  starting values are  used as  starting value
C                         forNewton method.
C                     IOPT(4)=1 is recommended if the Newton method has con-
C                     verging difficulties (this is the case  when IWORK(11)
C                     is very large  in comparison to  IWORK(1),  see output
C                     parameters).
C
C            IOPT( 5)=FORM    Used  formulation  as basis  of the  numerical
C                     integration
C                     0 - projected-strangeness-free  formulation, i.e., the
C                           user has to provide  the equations (1)-(7) toge-
C                           ther with  the first and  second time derivative
C                           of the holonomic constraints, i.e.,
C                           gI(p,v,t)           = d/dt g(p,t),
C                           gII(p,v,r,w,s,l,m,t)= d^2/dt^2 g(p,t),
C                           and the first time derivative of the nonholonomic
C                           constraints, i.e.,
C                           hI(p,v,r,w,s,l,m,t)= d/dt(H(p,s,t)Z(p)v+h(p,s,t)).
C                           If there exist some solution invariants (8) the
C                           user should also provide them and set NI equal
C                           to the number of the solution invariants. All
C                           provided equations have to be defined in the sub-
C                           routine EOM and the subroutine MAS in the correct
C                           order, see the subroutines EOM and MAS for more
C                           details.
C                     1 - projected-strangeness-index-1 formulation , i.e.,
C                           the user has to provide the equations  (1)-(7)
C                           together with the first time derivative of the
C                           holonomic constraints, i.e.,
C                           gI(p,v,t)            = d/dt g(p,t).
C                           If there exist some solution invariants (8) the
C                           user should also provide them and set NI equal
C                           to the number of the solution invariants. All
C                           provided equations have to be defined in the sub-
C                           routine EOM and the subroutine MAS in the correct
C                           order, see the subroutines EOM and MAS for more
C                           detail.
C
C            IOPT( 6)=NMAX    Maximal number of allowed steps.
C                     The default value (for IOPT(6)=0) is 100000.
C                     If the code stops with the error message IDID=-1117,
C                     IOPT(6) has to be increase or
C                     the integration can be continued by use of the obtained
C                     X and T as initial values for the continued integration.
C
C            IOPT( 8)=PRED    Step size strategy
C                      1 -  predictive controller (Gustafsson)
C                      2 -  classical step size control
C                     The default value (for IOPT(8)=0) is 1.
C                     The choice IOPT(8)=1 seems to produce safer results;
C                     for simple problems, the choice IOPT(8)=2 produces
C                     often slightly faster runs.
C
C            IOPT( 9)=NWTMAT   Approximation of the Newton iteration matrix
C                      0 - approximation at the initial point x_i of the
C                            current integration interval [t_{i},t_{i+1}]
C                            i.e., at (t_{i},x_{i}) (default)
C                      1 - approximation at the first extrapolated stage of
C                            the current integration interval [t_{i},t_{i+1}]
C                            i.e., at (t_{i}+c_{1}*h,X_{i1})
C                      2 - approximation at the second extrapolated stage of
```

```
C                        the current integration interval [t_{i},t_{i+1}]
C                        i.e., at (t_{i}+c_{2}*h,X_{i2})
C                    3 - approximation at the third extrapolated stage of
C                        the current integration interval [t_{i},t_{i+1}]
C                        i.e., at (t_{i}+c_{3}*h,X_{i3})
C                  Several numerical experiments turned out that the
C                  choice IOPT(9)=2 is the fastest while the
C                  choice IOPT(9)=0 is theoretically the safest.
C                  IOPT(9).NE.0 is only possible if IOPT(4)=STARTN=0,
C                  i.e., the extrapolated collocation solution is taken
C                  as starting value for Newton method.
C
C            IOPT(10)=NWTUPD   Update of the Newton iteration matrix
C                     0 - for the whole Newton iteration process in one
C                         integration step the same Newton iteration matrix
C                         is used, i.e., no update is allowed. (default)
C                  >0 - during the Newton iteration process in the current
C                         integration step IOPT(10) updates of the Newton
C                         iteration matrix are allowed.
C                  If convergence problems during the Newton iteration
C                  process occur, often the Newton matrix is not suitable.
C                  Therefore, in the case of IOPT(10)=0 the
C                  current integration step is rejected, the counter NCRJCT
C                  will be increased by one and the current integration
C                  step will be repeated with reduced step size.
C                  The option IOPT(10)>0 allows the update of the Newton
C                  iteration matrix IOPT(10) times. The Newton iteration
C                  matrix will be updated by use of the current iterates
C                  and the Newton iteration will be continued.
C                  Several numerical experiments have shown that IOPT(10)
C                  should not exceed 1.
C
C            IOPT(11)=DECOMPC  Decomposition of the algebraic part
C                     1 - LU decomposition with full pivoting
C                     2 - QR decomposition with pivoting
C                     3 - SV decomposition
C                  The default value (for IOPT(11)=0) is 3.
C                  By use of IOPT(11)=1 the integration becomes fastest
C                  but the stability of the decomposition can not be
C                  guaranteed. In situations with isolated singularities
C                  it may happen that the integrator does not detect
C                  the singularity if the tolerances RTOL or ATOL are
C                  too large.
C                  By use of IOPT(11)=2 or 3 the stability of the
C                  decomposition is guaranteed but the integration
C                  becomes slower.
C                  In case of redundant constraints only IOPT(11)=3 is
C                  possible.
C
C            IOPT(12)=DECOMPD  Decomposition of differential part
C                     0 - LU decomposition with partial pivoting (default)
C                     1 - QR decomposition
C                     By use of IOPT(12)=0 the integration becomes fastest.
C
C            IOPT(13)=SELCOMP  Recomputation strategy for the selectors
C                     0 - situation adapted
C                         the recomputation of the selector will be done
C                         only if the row pivoting of the constraints is
C                         changing or convergence problems occur during
C                         the Newton iteration process (default)
C                     1 - in every integration step
C                  In case of IOPT(13)=0 the amount of computations
C                  is reduced and the integration becomes faster.
C                  This speed-up is only possible if DECOMPD=0.
C
C            IOPT(14)=AUTONOM  Autonomy of the equations of motion
C                     0 - the equations of motion are not autonomous
```

```
C                          (default)
C                     1 - the equations of motion are autonomous
C                    If the equations of motion are autonomous the amount
C                    of computations can be reduced and the integration
C                    becomes faster.
C
C            IOPT(15)=MASSTRKT Structure of the mass matrix
C                     1 - full and time or/and state dependent
C                     2 - diagonal and time or/and state dependent
C                     3 - full and constant
C                     4 - diagonal and constant
C                    The default value (for IOPT(15)=0) is 1.
C
C            IOPT(17)=IVCNSST are the initial values consistent
C                    0 - No, the initial values are assumed to be not
C                          consistent. A check of consistency will be
C                          done and if necessary a correction will be
C                          computed. (default)
C                    1 - Yes, the initial values are assumed to be
C                          consistent. No check of consistency will be
C                          done.
C
C     ROPT    Input : double precision array ROPT(40)
C             Serve as parameters for the code. For standard use of the code
C             ROPT(1),..,ROPT(40) must be set to zero before calling.
C             See below for a more sophisticated use.
C
C             ROPT( 1)=UROUND   The rounding unit
C                      The default value (for ROPT(1)=0.0) is 1.D-16.
C
C             ROPT( 2)=SAFE     Safety factor in step size prediction
C                      The default value (for ROPT(2)=0.0) is 0.9.
C
C             ROPT( 3)=THET     Recomputation of the Jacobian
C                      Decides whether the Jacobian should be recomputed.
C                      Increase ROPT(3), to 0.1 say, when Jacobian evaluations
C                      are costly. for small systems ROPT(3) should be smaller
C                      (say 0.001D0). Negative ROPT(3) forces the code to
C                      compute the Jacobian after every accepted step.
C                      The default value (for ROPT(3)=0.0) is 0.001D0.
C
C             ROPT( 4)=FNEWT    Stopping criterion for Newton's method
C                      Smaller values of ROPT(4) make the code slower, but
C                      safer.
C                      The default value (for ROPT(4)=0.0) is
C                      MIN(0.03D0,RTOL(1)**0.5D0)
C
C
C             ROPT( 5)=QUOT1    Change of the step size
C                      See ROPT(6).
C                      The default value (for ROPT(5)=0.0) is 1.0D0
C
C             ROPT( 6)=QUOT2    Change of the step size
C                      If QUOT1 < HNEW/HOLD < QUOT2, then the step size is not
C                      changed. This saves, together with a large ROPT(3),
C                      decompositions and the amount of computations for
C                      large systems. For small systems one may have
C                      ROPT(5)=1.00D0, ROPT(6)=1.2D0, for large full systems
C                      ROPT(5)=0.99D0, ROPT(6)=2.0D0  might be good choices.
C                      The default value (for ROPT(6)=0.0) is 1.2D0
C
C             ROPT( 7)=HMAX     Maximal step size
C                      The default value (for ROPT(7)=0.0) is TEND-T
C
C             ROPT( 8)=FACL     PARAMETER FOR STEP SIZE SELECTION
C                      See ROPT(9).
C                      The default value (for ROPT(9)=0.0) is 8.0D0
```

```
C
C             ROPT( 9)=FACR     Step size selection
C                      The new step size is chosen subject to the restriction
C                      FACR <= HNEW/HOLD <= FACL
C                      The default value (for ROPT(8)=0.0) is 0.2D0
C
C     IVCOND   User supplied subroutine which provides initial conditions in
C                      addition to the constraints contained in the equations
C                      of motion (including hidden constraints)
C
C                   SUBROUTINE IVCOND(N,T,X,NCOND,COND,IPAR,RPAR,IERR)
C                   IMPLICIT NONE
C                   INTEGER          N,NCOND,IPAR(*),IERR
C                   DOUBLE PRECISION T,X(N),COND(NCOND),RPAR(*)
C
C                   N     Input  : integer
C                         Number of unknowns, i.e., dimension of X
C                         X has to remain unchanged.
C
C                   T     Input  : double precision
C                         Initial time t_0.
C                         T has to remain unchanged.
C
C                   X     Input  : double precision array X(N)
C                         Unknown variables, see above.
C                         X has to remain unchanged.
C
C                   NCOND Input  : integer
C                         Number of additional initial conditions provided in the
C                         subroutine IVCOND.
C                         NCOND has to remain unchanged.
C
C                   COND  Output : double precision array COND(NCOND)
C                         Residual of initial conditions, e.g. the condition
C                         COND(1)=X(4)-.5 forces the initial state of X(4) to
C                         be 0.5, i.e. X(4)=0.5D0.
C                         Note the fact, that the conditions given in IVCOND
C                         override the given initial values, i.e., if the given
C                         initial values are consistent but do not satisfy the
C                         (possibly wrong) conditions given in IVCOND the
C                         initial values will be corrected such that both,
C                         the constraints and the initial conditions are
C                         satisfied.
C                         In case of initial values which are consistent to
C                         the constraints the option IOPT(17) could be set to 1
C                         to avoid such a correction.
C
C                   IPAR  Input/Output: integer array IPAR(*)
C                         Integer parameters which are only used by the user.
C                         They are unused and unchanged by GEOMS.
C
C                   RPAR  Input/Output: double precision array RPAR(*)
C                         Double precision parameters which are only used by the
C                         user. RPAR is unused and unchanged by GEOMS.
C
C                   IERR  Output : integer
C                         Indicator of success. IERR is only used by
C                         user supplied subroutines. After every call of a user
C                         supplied  subroutine the status of IERR is checked. If
C                         IERR is negative  the run of GEOMS will be interrupted
C                         and GEOMS returns to the  calling program. IERR  is
C                         unchanged by GEOMS.
C
C     EOM      Name (EXTERNAL) of the user supplied subroutine which provides
C              the right-hand side (RHS) of EoM (1)-(8) together with the first
C              and second time derivative of the holonomic constraints, i.e.,
C                gI(p,v,t)            = d/dt g(p,t),
```

```
C               gII(p,v,r,w,s,l,m,t) = d^2/dt^2 g(p,t),
C             and the first time derivative of the nonholonomic constraints,
C             i.e.,
C               hI(p,v,r,w,s,l,m,t)  = d/dt (H(p,s,t)Z(p)v+h(p,s,t)).
C             The order and the number of the provided right-hand sides
C             depends on the used formulation, see IOPT(5) and above for more
C             detail.
C
C             SUBROUTINE EOM(M,N,T,X,RDA,IOPT,ROPT,IPAR,RPAR,IERR)
C             IMPLICIT NONE
C             INTEGER          M,N,IOPT(*),IPAR(*),IERR
C             DOUBLE PRECISION T,X(N),RDA(M),ROPT(*),RPAR(*)
C
C                M     Input  : integer
C                      Total umber of provided equations (M.GE.N),
C                      i.e., dimension of RDA, see below.
C                      In the case of use of
C                      * projected-strangeness-free formulation we have
C                        M=NP+NV+NR+NW+NS+3*NL+2*NM+NI,
C                      * projected-strangeness-index-1 formulation we have
C                        M=NP+NV+NR+NW+NS+2*NL+NM+NI.
C                      M has to remain unchanged.
C
C                N     Input  : integer
C                      Number of unknowns (M.GE.N), i.e., dimension of X.
C                      We have N=NP+NV+NR+NW+NS+NL+NM.
C                      N has to remain unchanged.
C
C                T     Input  : double precision
C                      Evaluation of the right-hand side of the provided
C                      equations at time T.
C                      T has to remain unchanged.
C
C                X     Input  : double precision array X(N)
C                      Vector of unknowns, see above.
C                      X has to remain unchanged.
C
C                RDA   Output : double precision array RDA(M)
C                      Right-hand side of the provided reduced derivative
C                      array. The order and the number of the provided
C                      right-hand sides depends on the used formulation, see
C                      IOPT(5).
C                      If IOPT( 5)=0 the numerical integration is based on the
C                        projected-strangeness-free formulation, i.e., the
C                        user  has to provide the equations (1)-(7) together
C                        with the first and second time derivative of the
C                        holonomic constraints, i.e.,
C                          gI(p,v,t)           =d/dt g(p,t),
C                          gII(p,v,r,w,s,l,m,t)=d^2/dt^2 g(p,t),
C                        and the first time derivative of the nonholonomic
C                        constraints, i.e.,
C                          hI(p,v,r,w,s,l,m,t) =d/dt(H(p,s,t)Z(p)v+h(p,s,t)).
C                        If there exist some solution invariants (8) the user
C                        should also provide them and set NI equal to the
C                        number of the solution invariants. The order is given
C                        by
C
C                        RDA(1:NW)                                      =d
C                        RDA(NW+1:NW+NL)                                =gII
C                        RDA(NW+NL+1:NW+NL+NM)                          =hI
C                        ------------------------------------------------
C                        RDA(NW+NL+NM+1:NW+NL+NM+NL)                    =gI
C                        RDA(NW+NL+NM+NL+1:NW+NL+NM+NL+NM)              =h
C                        RDA(NW+NL+NM+NL+NM+1:NW+NL+NM+NL+NM+NI)        =e
C                        ------------------------------------------------
C                        RDA(NW+NL+NM+NL+NM+NI+1:NW+NL+NM+NL+NM+NI+NS) =c
C                        RDA(NW+NL+NM+NL+NM+NI+NS+1:...
```

```
C                                        NW+NL+NM+NL+NM+NI+NS+NL) =g
C                       ------------------------------------------------------
C                       RDA(NW+NL+NM+NL+NM+NI+NS+NL+1:...
C                                          NW+NL+NM+NL+NM+NI+NS+NL+NR) =b
C                       ------------------------------------------------------
C                       RDA(NW+NL+NM+NL+NM+NI+NS+NL+NR+1:...
C                                        NW+NL+NM+NL+NM+NI+NS+NL+NR+NV) =f_dyn
C                       ------------------------------------------------------
C                       RDA(NW+NL+NM+NL+NM+NI+NS+NL+NR+NV+1:...
C                                    NW+NL+NM+NL+NM+NI+NS+NL+NR+NV+NP) =f_kin
C
C                     If IOPT( 5)=1 the numerical integration is based on the
C                       projected-strangeness-index-1 formulation , i.e., the
C                       user has to provide the equations (1)-(7) together
C                       with the first time derivative of the holonomic
C                       constraints, i.e.,
C                         gI(p,v,t)            = d/dt g(p,t).
C                       If there exist some solution invariants (8) the user
C                       should also provide them and set NI equal to the
C                       number of the solution invariants. The order is given
C                       by
C
C                       RDA(1:NW)                                          =d
C                       ------------------------------------------------------
C                       RDA(NW+1:NW+NL)                                    =gI
C                       RDA(NW+NL+1:NW+NL+NM)                              =h
C                       RDA(NW+NL+NM+1:NW+NL+NM+NI)                        =e
C                       ------------------------------------------------------
C                       RDA(NW+NL+NM+NI+1:NW+NL+NM+NI+NS)                  =c
C                       RDA(NW+NL+NM+NI+NS+1:NW+NL+NM+NI+NS+NL)            =g
C                       ------------------------------------------------------
C                       RDA(NW+NL+NM+NI+NS+NL+1:NW+NL+NM+NI+NS+NL+NR) =b
C                       ------------------------------------------------------
C                       RDA(NW+NL+NM+NI+NS+NL+NR+1:...
C                                          NW+NL+NM+NI+NS+NL+NR+NV) =f_dyn
C                       ------------------------------------------------------
C                       RDA(NW+NL+NM+NI+NS+NL+NR+NV+1:...
C                                       NW+NL+NM+NI+NS+NL+NR+NV+NP) =f_kin
C
C
C                  IOPT  Input  : integer array IOPT(40)
C                        Serve as parameters for the code.
C                        IOPT has to remain unchanged.
C
C                  ROPT  Input  : double precision array ROPT(40)
C                        Serve as parameters for the code.
C                        ROPT has to remain unchanged.
C
C                  IPAR  Input/Output: integer array IPAR(*)
C                        Integer parameters which are only used by the user.
C                        They are unused and unchanged by GEOMS.
C
C                  RPAR  Input/Output: double precision array RPAR(*)
C                        Double precision parameters which are only used by the
C                        user. They are unused and unchanged by GEOMS.
C
C                  IERR  Output : integer
C                        Indicator of success. IERR is only used by
C                        user supplied subroutines. After every call of a user
C                        supplied  subroutine the status of IERR is checked. If
C                        IERR is negative  the run of GEOMS will be interrupted
C                        and GEOMS returns to the  calling program. IERR  is
C                        unchanged by GEOMS.
C
C     MAS      Name (EXTERNAL) of the user supplied subroutine which provides
C              the mass matrix M(p,t) in equation (2) of the EoM
C
```

```
C              SUBROUTINE MAS(T,NX,X,M,N,MA,IOPT,ROPT,IPAR,RPAR,IERR)
C              IMPLICIT NONE
C              INTEGER          NX,M,N,IOPT(*),IPAR(*),IERR
C              DOUBLE PRECISION T,X(NX),MA(M,N),ROPT(*),RPAR(*)
C
C                 T     Input  : double precision
C                       Evaluation of the mass matrix MA at time T.
C                       T has to remain unchanged.
C
C                 NX    Input  : integer
C                       Number of unknowns, i.e., dimension of X. We have
C                       NX=NP+NV+NR+NW+NS+NL+NM.
C                       NX has to remain unchanged.
C
C                 M     Input  : integer
C                       Number of rows of the mass matrix MA. We have M=NV.
C                       M has to remain unchanged.
C
C                 N     Input  : integer
C                       Number of rows of the mass matrix MA. We have N=NV.
C                       N has to remain unchanged.
C
C                 X     Input  : double precision array X(NX)
C                       Vector of unknowns, see above.
C                       X has to remain unchanged.
C
C                 MA    Output : double precision array MA(M,N)
C                       Mass matrix of the equations of motion. The mass matrix
C                       has to be provided as a full M x N array,
C                       also in the case of diagonal structure. Because of the
C                       used regularization technique a sparse storage is not
C                       possible and does not save time or memory.
C
C                 IOPT  Input  : integer array IOPT(40)
C                       Serve as parameters for the code.
C                       IOPT has to remain unchanged.
C
C                 ROPT  Input  : double precision array ROPT(40)
C                       Serve as parameters for the code.
C                       IOPT has to remain unchanged.
C
C                 IPAR  Input/Output: integer array IPAR(*)
C                       Integer parameters which are only used by the user.
C                       They are unused and unchanged by GEOMS.
C
C                 RPAR  Input/Output: double precision array RPAR(*)
C                       Double precision parameters which are only used by the
C                       user. They are unused and unchanged by GEOMS.
C
C                 IERR  Output : integer
C                       Indicator of success. IERR is only used by
C                       user supplied subroutines. After every call of a user
C                       supplied  subroutine the status of IERR is checked. If
C                       IERR is negative  the run of GEOMS will be interrupted
C                       and GEOMS returns to the  calling program. IERR  is
C                       unchanged by GEOMS.
C
C     JAC       Name (EXTERNAL) of the user supplied subroutine which computes
C               the NEGATIVE partial derivatives of the right-hand side of the
C               equations of motion. (This routine is only called if IJAC=1.
C               Supply a dummy subroutine in the case IJAC=0).
C
C               SUBROUTINE JAC(M1,M2,M3,M4,M5,M6,N1,N2,N3,N4,M,N,
C              #               T,X,FX1,FX2,FX3,FX4,FX5,FX6,
C              #               IOPT,ROPT,RPAR,IPAR,IERR)
C               IMPLICIT NONE
C               INTEGER          M1,M2,M3,M4,M5,M6,N1,N2,N3,N4,M,N,
```

```
C             #                IOPT(*),IPAR(*),IERR
C              DOUBLE PRECISION T,X(N),FX1,FX2,FX3,FX4,FX5,FX6,ROPT(*),RPAR(*)
C
C                 M1    Input  : integer
C                       Number of constraints depending on all unknown
C                       variables and restricting the Lagrange multipliers l
C                       and m and the auxiliary variables w, i.e., 0=d,
C                       0=gII, 0=hI. IF IOPT(5)=0 we have M1=NW+NL+NM and if
C                       IOPT(5)=1 we have M1=NW (note that M1=0 is possible).
C                       Compare with the block row structure of RDA in the
C                       subroutine EOM.
C                       M1 has to remain unchanged.
C
C                 M2    Input  : integer
C                       Number of constraints only depending on the unknown
C                       variables p, v, and s and restricting the velocity
C                       variables v, i.e., 0=gI, 0=h, 0=e. We have M2=NL+NM+NI.
C                       Compare with the block row structure of RDA in the
C                       subroutine EOM.
C                       M2 has to remain unchanged.
C
C                 M3    Input  : integer
C                       Number of constraints only depending on the unknown
C                       variables p and s and restricting the position p and
C                       the contact variables s, i.e., 0=c, 0=g. We have
C                       M3=NS+NL.
C                       Compare with the block row structure of RDA in the
C                       subroutine EOM.
C                       M3 has to remain unchanged.
C
C                 M4    Input  : integer
C                       Number of dynamical force element equations (3), i.e.,
C                       we have M4=NR.
C                       Compare with the block row structure of RDA in the
C                       subroutine EOM.
C                       M4 has to remain unchanged.
C
C                 M5    Input  : integer
C                       Number of dynamical equations of motion (2), i.e.,
C                       we have M5=NV.
C                       Compare with the block row structure of RDA in the
C                       subroutine EOM.
C                       M5 has to remain unchanged.
C
C                 M6    Input  : integer
C                       Number of kinematical equations of motion (1), i.e.,
C                       we have M6=NP.
C                       Compare with the block row structure of RDA in the
C                       subroutine EOM.
C                       M6 has to remain unchanged.
C
C                 N1    Input  : integer
C                       Number of auxiliary variables plus the number of
C                       Lagrange multipliers, i.e., we have N1=NW+NL+NM.
C                       Compare with the block row structure of X above.
C                       N1 has to remain unchanged.
C
C                 N2    Input  : integer
C                       Number of dynamical force element variables, i.e.,
C                       we have N2=NR.
C                       Compare with the block row structure of X above.
C                       N2 has to remain unchanged.
C
C                 N3    Input  : integer
C                       Number of velocity variables, i.e., we have N3=NV.
C                       Compare with the block row structure of X above.
C                       N3 has to remain unchanged.
```

```
C
C                N4    Input  : integer
C                      Number of contact point variables plus the number of
C                      position variables, i.e., we have N1=NS+NP.
C                      Compare with the block row structure of X above.
C                      N4 has to remain unchanged.
C
C                M     Input  : integer
C                      Total number of provided equations,
C                      i.e., dimension of RDA, see subroutine EOM and the
C                      number of rows of the partial derivatives. We have
C                      M=M1+M2+M3+M4+M5+M6.
C                      M has to remain unchanged.
C
C                N     Input  : integer
C                      Number of unknowns, i.e., dimension of X.
C                      We have N=NP+NV+NR+NW+NS+NL+NM=N1+N2+N3+N4.
C                      N has to remain unchanged.
C
C                T     Input  : double precision
C                      Evaluation of the partial derivatives at time T.
C                      T has to remain unchanged.
C
C                X     Input  : double precision array X(NX)
C                      Vector of unknowns, see above.
C                      X has to remain unchanged.
C
C                FX1   Output : double precision array FX1(M1,N)
C                      NEGATIVE partial derivatives of d, gII, hI with
C                      respect to [w l m | r | v | s p ]. We have
C                            [ d d  /d[w l m | r | v | s p ] ]
C                        FX1=[ d gII/d[w l m | r | v | s p ] ] in R^(M1,N)
C                            [ d hI /d[w l m | r | v | s p ] ]
C                      Compare with the block row structure of RDA in the
C                      subroutine EOM and with the block row structure of X
C                      above.
C
C                FX2   Output : double precision array FX2(M2,N3+N4)
C                      NEGATIVE partial derivatives of gI, h~, e with
C                      respect to [v s p]. We have
C                            [ d gI/d[v s p] ]
C                        FX2=[ d h~/d[v s p] ] in R^(M2,N3+N4)
C                            [ d e /d[v s p] ]
C                      Compare with the block row structure of RDA in the
C                      subroutine EOM and with the block row structure of X
C                      above.
C
C                FX3   Output : double precision array FX3(M3,N4)
C                      NEGATIVE partial derivatives of c and g with
C                      respect to [s p]. We have
C                            [ d c/d[s p] ]
C                        FX3=[            ] in R^(M3,N4)
C                            [ d g/d[s p] ]
C                      Compare with the block row structure of RDA in the
C                      subroutine EOM and with the block row structure of X
C                      above.
C
C                FX4   Output : double precision array FX4(M4,N)
C                      NEGATIVE partial derivatives of the right-hand side of
C                      the dynamical force element equations, i.e., of b with
C                      respect to [w l m | r | v | s p ]. We have
C                        FX4=d b/d[w l m | r | v | s p ] in R^(M4,N)
C                      Compare with the block row structure of RDA in the
C                      subroutine EOM and with the block row structure of X
C                      above.
C
C                FX5   Output : double precision array FX5(M5,N)
```

```
C                     NEGATIVE partial derivatives of the right-hand side of
C                     the dynamical equations of motion, i.e., of f_dyn with
C                     respect to [w l m | r | v | s p ]. We have
C                       FX5=d f_dyn/d[w l m | r | v | s p ] in R^(M5,N)
C                     Compare with the block row structure of RDA in the
C                     subroutine EOM and with the block row structure of X
C                     above.C
C
C                FX6  Output : double precision array FX6(M6,N3+N4)
C                     NEGATIVE partial derivatives of the right-hand side of
C                     the kinematical equations of motion, i.e., of f_kin
C                     with respect to [v | s p ]. We have
C                       FX6=d f_kin/d[ v | s p ] in R^(M6,N3+N4)
C                     Compare with the block row structure of RDA in the
C                     subroutine EOM and with the block row structure of X
C                     above.
C
C                IOPT Input  : integer array IOPT(40)
C                     Serve as parameters for the code.
C                     IOPT has to remain unchanged.
C
C                ROPT Input  : double precision array ROPT(40)
C                     Serve as parameters for the code.
C                     IOPT has to remain unchanged.
C
C                IPAR Input/Output: integer array IPAR(*)
C                     Integer parameters which are only used by the user.
C                     They are unused and unchanged by GEOMS.
C
C                RPAR Input/Output: double precision array RPAR(*)
C                     Double precision parameters which are only used by the
C                     user. They are unused and unchanged by GEOMS.
C
C                IERR Output : integer
C                     Indicator of success. IERR is only used by
C                     user supplied subroutines. After every call of a user
C                     supplied  subroutine the status of IERR is checked. If
C                     IERR is negative  the run of GEOMS will be interrupted
C                     and GEOMS returns to the  calling program. IERR  is
C                     unchanged by GEOMS.
C
C    IJAC    Input  : integer
C            Switch for the computation of the partial derivatives of the
C            right-hand side of the equations of motion
C            IJAC=0 Partial derivatives are computed internally by finite
c                   differences, subroutine JAC is never called.
C            IJAC=1 Partial derivatives are supplied by subroutine JAC.
C
C    SOLOUT  Name (EXTERNAL) of subroutine providing the numerical solution
C            during integration.
C            If IOUT=1, it is called after every successful step. Supply a
C            dummy subroutine if IOUT=0.
C            SOLOUT furnishes the solution X at the nr-th grid-point T
C            (Thereby the initial value is the first grid-point).
C
C            SUBROUTINE SOLOUT(NACCPT,TOLD,T,X,N,NN2,NN3,NN4,CONTX,H,C1M1,
C           #                  C2M1,RPAR,IPAR,IERR)
C            IMPLICIT NONE
C            INTEGER          NACCPT,N,NN2,NN3,NN4,IPAR(*),IERR
C            DOUBLE PRECISION TOLD,T,H,X(N),CONTX(NN4),RPAR(*),C1M1,C2M1
C            DOUBLE PRECISION GEDENSOUT
C            EXTERNAL         GEDENSOUT
C
C                NACCPT Input  : integer
C                     Number of accepted steps so far.
C                     NACCPT has to remain unchanged.
C
```

```
C                TOLD  Input  : double precision
C                      The preceeding grid-point.
C                      TOLD has to remain unchanged.
C
C                T     Input  : double precision
C                      Current simulation time T.
C                      T has to remain unchanged.
C
C                X     Input  : double precision array X(NX)
C                      Vector of unknowns, see above.
C                      X has to remain unchanged.
C
C                N     Input  : integer
C                      Number of unknowns, i.e., dimension of X.
C                      We have N=NP+NV+NR+NW+NS+NL+NM=N1+N2+N3+N4.
C                      N has to remain unchanged.
C
C                NN2,NN3,NN4,CONTX,H,C1M1,C2M1 Input: integer/double precision
C                      Internal communication for the use by the subroutine
C                      GEDENSOUT for dense output.
C                      NN2,NN3,NN4,CONTX,H,C1M1,C2M1 have to remain unchanged.
C
C                IPAR  Input/Output: integer array IPAR(*)
C                      Integer parameters which are only used by the user.
C                      They are unused and unchanged by GEOMS.
C
C                RPAR  Input/Output: double precision array RPAR(*)
C                      Double precision parameters which are only used by the
C                      user. They are unused and unchanged by GEOMS.
C
C                IERR  Output : integer
C                      Indicator of success. IERR is only used by
C                      user supplied subroutines. After every call of a user
C                      supplied  subroutine the status of IERR is checked. If
C                      IERR is negative  the run of GEOMS will be interrupted
C                      and GEOMS returns to the  calling program. IERR  is
C                      unchanged by GEOMS.
C
C                -----  Continuous output -----
C                      During calls to "SOLOUT", a continuous solution
C                      for the interval [TOLD,T] is available through
C                      the function
C                          GEDENSOUT(I,TOUT,N,NN2,NN3,NN4,T,H,CONTX,C1M1,C2M1)
C                      which provides an approximation to the I-th
C                      component of the solution at the point TOUT, e.g.,
C                         DO I=1,N
C                         XOUT(I)=GEDENSOUT(I,TOUT,N,NN2,NN3,NN4,T,H,CONTX,
C                        #                  C1M1,C2M1)
C                         END DO
C                      The value TOUT should lie in the interval [TOLD,T].
C                      Do not change the entries of N, NN2, NN3, NN4, T, H,
C                      CONTX, C1M1, C2M1.
C                      The function GEDENSOUT is adopted from the code RADAU5,
C                      see the book:
C                      E. Hairer and G. Wanner, Solving Ordinary Differential
C                      Equations II. Stiff and Differential-Algebraic Problems
C                      Springer Series in Computational Mathematics 14,
C                      Springer-Verlag 1991, Second edition 1996.
C                      The former name was CONTR5.
C
C     IOUT    Input  : integer
C             Switch for the calling of subroutine SOLOUT.
C             IOUT=0 Subroutine is never called.
C             IOUT=1 Subroutine is available for output.
C
C     LIWORK  Input  : integer
C             Declares the length of the array IWORK. LIWORK has to be at least
```

```
C             20.
C
C     IWORK   Output: integer array IWORK(LIWORK)
C             Statistical information
C             IWORK( 1) NACCPT - Number of accepted integration steps
C             IWORK( 2) NEOM   - Number of evaluations of the right-hand side
C                                of the equations of motion
C             IWORK( 3) NMAS   - Number of evaluations of the mass matrix
C             IWORK( 4) NJAC   - Number of evaluations of the Jacobian of the
C                                right-hand side of the equations of motion
C             IWORK( 5) NSEL   - Number of determinations of suitable selectors
C             IWORK( 6) NPDEC  - Number of predecompositions, i.e., of FX, M,
C                                and IKIN
C             IWORK( 7) NEDEC  - Number of E-decompositions, i.e., of E1 and E2
C             IWORK( 8) NBSUB  - Number of backward substitutions
C             IWORK( 9) NSTEP  - Number of steps
C             IWORK(10) NERJCT - Number of rejections caused by error test
C                                failures
C             IWORK(11) NCRJCT - Number of rejections caused by convergence
C                                problems of the Newton process
C
C     LRWORK  Input  : integer
C             Declares the length of the array RWORK.
C             A safe choice for all possible setting in IOPT is
C               LRWORK at least 5*N
C             Depending on IOPT it is sufficient ...
C               If IOPT(17)=IVCNSST=0 then LRWORK has to be at least 5*N
C               If IOPT(11)=DECOMPC=3 then LRWORK has to be at least
C                 5*MAX(M1,M2,M3,N1,N3,N4), see comments to subroutine JAC.
C               If IOPT(11)=DECOMPC=2 then LRWORK has to be at least N
C               If IOPT(12)=DECOMPD=1 then LRWORK has to be at least 2*N
C             For good performance, LRWORK should generally be larger.
C
C     RWORK   Intern : integer array IWORK(LIWORK)
C
C     IPAR    Input/Output  : integer array IPAR(*)
C             Integer parameters which are only used by the user. They are
C             unused and unchanged by GEOMS.
C
C     RPAR    Input/Output: double precision array RPAR(*)
C             Double precision parameters which are only used by the user.
C             RPAR is unused and unchanged by GEOMS.
C
C     IERR    Input/Output : integer
C             Indicator of success. IERR is only used by user
C             supplied subroutines. After every call of a user supplied
C             subroutine the status of IERR is checked. If IERR is negative
C             the run of GEOMS will be interrupted and GEOMS returns to the
C             calling program. IERR  is unchanged by GEOMS.
C
C     IDID    Output : integer
C             Reports success upon return. The first two digits
C             indicate the subroutine which causes trouble.
C
C       IDID=-10.. An error occurred in the subroutine GEOMS
C            -1001 Option array IOPT or ROPT or tolerances RTOL or ATOL
C                  contains wrong data
C                  Check the output in UNIT=IOPT(2) for more information
C                  If the option IOPT(2) equals 0 turn on the output.
C            -1002 Initial IDID lower than 0
C
C       IDID=-11.. An error occurred in the subroutine GECOR
C            -1101 Stop initialized by SOLOUT
C            -1102 Stop initialized by EOM
C            -1103 Stop initialized by MAS
C            -1104 Stop initialized by JAC
C            -1105 Initial conditions not consistent
```

```
C            -1106 Final time TEND before initial time T
C            -1111 QR-Decomposition of FX1 not possible
C            -1112 QR-Decomposition of FX2 not possible
C            -1113 QR-Decomposition of FX3 not possible
C            -1114 QR-Decomposition of E1 or E2 not possible
C            -1115 Newton method repeatedly does not converge NSING.GE.5
C            -1116 Newton method repeatedly does not converge NSING.GE.5
C            -1117 More than NMAX steps are needed
C            -1118 Step size too small
C            -1128 An error occurred during use of DORMQR
C            -1129 An error occurred during use of DORMQR
C
C       IDID=-12.. An error occurred in the subroutine GEFXNUM
C            -1201 Stop initialized by EOM
C
C       IDID=-14.. An error occurred in the subroutine GEDECCQR
C            -1401 Constraints redundant or dd/dw singular
C                  (FX1 rank deficient).
C                  Try the integration again with IOPT(11)=3 (SVD).
C            -1402 Constraints or the invariant equations are redundant
C                  (FX2 rank deficient). Try the integration again with
C                  IOPT(11)=3 (SVD).
C            -1403 Constraints redundant or dc/ds singular
C                  (FX3 rank deficient).
C                  Try the integration again with IOPT(11)=3 (SVD).
C
C       IDID=-18.. An error occurred in the subroutine GETRFRHSC
C            -1801 Multiplication with Q1 not possible
C            -1802 Multiplication with Q2 not possible
C            -1803 Multiplication with Q3 not possible
C            -1804 Multiplication with Q4 not possible
C
C       IDID=-20.. An error occurred in the subroutine GEERREST
C            -2004 Multiplication with Q4 not possible
C
C       IDID=-21.. An error occurred in the subroutine GEINIVAL.
C            -2101 Stop initialized by EOM.
C            -2102 Stop initialized by IVCOND.
C            -2103 An error occurred during SVD.
C            -2104 Divergence during determination of consistent initial
C                  values. The given conditions in IVCOND together with
C                  all constraints of the EoM form an overdetermined system.
C                  Perhaps it is contradictory.
C                  => Check consistency of all constraints of the EoM in
C                     relation to the conditions given in IVCOND!
C                  => If you are sure that the initial values are consistent
C                     (at least variables P and V) you can set IOPT(17)=1.
C            -2105 No Convergence in the given limit of iterations.
C                  (See the source code of GEINIVAL and increase NIT or/and
C                  NNWTUPD.
C            -2106 Given conditions in IVCOND together with constraints in
C                  EoM are not sufficient to uniquely determine consistent
C                  initial values. Perhaps there are not enough conditions
C                  or they are redundant.
C                  => Provide more (nonredundant) conditions in IVCOND!
C                  => Check NIVCOND!
C                  => Check redundancy of all constraints of the EoM in
C                     relation to the conditions given in IVCOND!
C                  => If you are sure that the initial values are consistent
C                     (at least variables P and V) you can set IOPT(17)=1.
C
C       IDID=-24.. An error occurred in the subroutine GEDECCSV
C            -2401 Constraints are not uniformly redundant, i.e., rank of FX1
C                  was changing
C            -2402 Constraints are not uniformly redundant, i.e., rank
C                  deficiency of FX1 not identical to rank deficiency of FX2
C            -2403 Constraints are not uniformly redundant, i.e., rank
```

```
C                 deficiency of FX1 not identical to rank deficiency of FX3
C           -2404 An error occurred during SVD of FX1 or FX2 or FX3
C
C       IDID=-26.. An error occurred in the subroutine GEDECCLU
C           -2601 Constraints redundant or dd/dw singular
C                 (FX1 rank deficient).
C                 Try the integration again with IOPT(11)=3 (SVD).
C           -2602 Constraints or the invariant equations are redundant
C                 (FX2 rank deficient). Try the integration again with
C                 IOPT(11)=3 (SVD).
C           -2603 Constraints redundant or dc/ds singular
C                 (FX3 rank deficient).
C                 Try the integration again with IOPT(11)=3 (SVD).
C
C -------------------------------------------------------------------------
```

B.2 Manual of GMKSSOL

```
      SUBROUTINE GMKSSOL(T,NOUT,TOUT,
     #                   NP,NR,NL,X,XP,XOUT,
     #                   LIWORK,IWORK,LRWORK,RWORK,
     #                   IOPT,ROPT,IPAR,RPAR,IERR)
C -------------------------------------------------------------------------
C
C  NAME      : (G)eneralisierter (M)ehr(K)oerper(S)ystem (SOL)ver
C                   Multibody system solver
C
C  PURPOSE   : This subroutine performs the numerical simulation
C              of a multibody system whose state is described by
C
C              p - position variables                   of dimension NP,
C              v - velocity variables                   of dimension NP,
C              r - dynamical force element variables    of dimension NR,
C              l - holonomic Lagrange multipliers       of dimension NL,
C
C              by numerical integration of the equations of motion
C              in the form
C
C                p'=v                                                (1) (f_kin)
C                v'=f(p,v,r,t)-G^T*lambda =: fdyn(p,v,r,lambda,t)    (2) (f_dyn)
C                r'=b(p,v,r,t)                                      (3)
C                0 =g(p,t)                                          (4)
C
C              on the domain [t_0,t_f].
C
C              The prime denotes the time derivative, e.g., p'=dp/dt, and
C              the 'T' denotes the transpose of a matrix or vector, e.g.,
C              GT=transpose of G. Furthermore, the equations correspond to
C
C              (1) Kinematical equations of motion      of dimension NP,
C              (2) Dynamical equations of motion        of dimension NV,
C              (3) Dynamical force element equations    of dimension NR,
C              (4) Holonomic constraints                of dimension NL,
C
C              The System (1)-(4) has to satisfy the following.
C               a) G = dg/dp.
C               b) rank(GM^{-1}GT)=rank(G)=const for all t in [T,TEND].
C                  Alternatively
C                       [ M  GT ]
C                  rank([       ])=NP+rank(G)=constant
C                       [ G  0  ]
C                  has to be satisfied for all t in [t_0,t_f]=[T,TEND].
C
C              The integration method used is an implicit Runge-Kutta method
C              (Radau IIa) of order 5 with step size control, continuous
C              output, and consistent initialization for lambda.
```

```
C
C  METHOD   :  The equations of motion are integrated by the implicit
C             Runge-Kutta-Method of type RADAU IIa of order 5 and using the
C             projected-strangeness-free formulation of the equations of
C             motion.
C
C  VERSION   : May 31, 2005
C
C  REVISIONS : -
C
C  AUTHORS   : Address: F. Ebert
C                       Institut fuer Mathematik
C                       Technische Universitaet Berlin
C                       Strasse des 17. Juni
C                       10623 Berlin, Germany
C              e-mail:  ebert@math.tu-berlin.de
C
C              Address: A. Steinbrecher
C                       Institut fuer Mathematik
C                       Technische Universitaet Berlin
C                       Strasse des 17. Juni
C                       10623 Berlin, Germany
C              e-mail:  steinbrecher@math.tu-berlin.de
C
C  REFERENCES: This code is part of the PhD thesis:
C              A.Steinbrecher. Numerical Solution of Quasi-Linear Differential-
C              Algebraic Equations and Industrial Simulation of Multibody
C              Systems. PhD thesis, TU Berlin, Institut fuer Mathematik, 2005
C
C              F.Ebert,F.and A.Steinbrecher. Dokumentation verschiedener Loeser
C              zur numerischen Integration gewoehnlicher Differentialgleichun-
C              gen und differentiell-algebraischer Gleichungen in der Mehrkoer-
C              perdynamik. Technical report: Bosch Rexroth AG, BR / VES
C              (Simulationstechnik), Number IR-005-04-VE 12/2004. 2004.
C
C  KEYWORDS  : numerical simulation of mechanical systems, equations of motion,
C              differential-algebraic equations, projected-strangeness-free
C              formulation
C
C  NOTE      : The (basic) linear algebra routines are provided by the
C              libraries BLAS and LAPACK
C
C -----------------------------------------------------------------------------
C
C  CALL
C  ---------------------------
C
C      SUBROUTINE GMKSSOL(T,NOUT,TOUT,
C     #                   NP,NR,NL,X,XP,XOUT,
C     #                   LIWORK,IWORK,LRWORK,RWORK,
C     #                   IOPT,ROPT,IPAR,RPAR,IERR)
C       IMPLICIT NONE
C       INTEGER          NOUT,NP,NR,NL,LIWORK,LRWORK,IERR,
C     #                  IWORK(LIWORK),IOPT(40),IPAR(*)
C       DOUBLE PRECISION T,TOUT(NOUT),X(NP+NP+NR+NL),XP(NP+NP+NR+NL),
C     #                  XOUT(NP+NP+NR+NL,NOUT),RWORK(LRWORK),ROPT(40),
C     #                  RPAR(*)
C       EXTERNAL         EQUOFMOT,CONSMATR,SOLOUT
C
C  INPUT- AND OUTPUT-ARGUMENTS
C  ---------------------------
C
C     INPUT- AND OUTPUT-ARGUMENTS
C
C     T       Input  : double precision
C             Initial time t
C             Output :
```

```
C             Last successfully reached time. If whole integration was
C             successful T=TOUT(NOUT).
C
C     NOUT    Input : integer
C             Number of time steps where the solution shall be stored in XOUT.
C
C     TOUT    Input : double precision : array TOUT(NOUT)
C             Defines the times at which the solution should by stored in XOUT.
C             The times must be sorted chronologically such that
C             t_0 <= TOUT(I) < TOUT(J) if I<J
C
C     NP      Input  : integer
C             Number of position variables p.
C
C     NR      Input  : integer
C             Number of position variables r.
C
C     NL      Input  : integer
C             Number of position variables lambda.
C
C     X       Input  : double precision : array X(NP+NP+NR+NL)
C             Contains all variables of the MBS in the following order.
C             X( 1:NP                 )=p
C             X(NP+1:NP+NP            )=v
C             X(NP+NP+1:NP+NP+NR      )=r
C             X(NP+NP+NR+1:NP+NP+NR+NL)=lambda
C
C     XP      Input  : double precision : array XP(NP+NP+NR+NL)
C             Contains the time derivative of all variables of the MBS.
C               XP( 1:NP                )=p'     (=v)
C               XP(NP+1:NP+NP           )=v'
C               XP(NP+NP+1:NP+NP+NR     )=r'
C               XP(NP+NP+NR+1:NP+NP+NR+NL)=lambda'
C             It is not necessary to initialize XP.
C
C     XOUT    Output : double precision : array XOUT(NP+NP+NR+NL,NOUT)
C             Contains the numerical solution of all variables of the MBS at
C             intermediate times TOUT(I), I=1,...,NOUT
C               XOUT( 1:NP                )=p(TOUT(I))
C               XOUT(NP+1:NP+NP           )=v(TOUT(I))
C               XOUT(NP+NP+1:NP+NP+NR     )=r(TOUT(I))
C               XOUT(NP+NP+NR+1:NP+NP+NR+NL)=lambda(TOUT(I))
C
C     LIWORK  Input : integer
C             Length of integer work array
C               LIWORK .GE. 21+3*N          with N=NP+NP+NR+NL
C
C     IWORK   Input : integer : array IWORK(LIWORK)
C             Integer work array.
C
C     LRWORK  Input : integer
C             Length of double precision work array
C               LRWORK .GE. 40+12*N+5*N*N    with N=NP+NP+NR+NL
C
C     RWORK   Input : integer : array RWORK(LRWORK)
C             Double precision work array.
C
C     IOPT    Input : integer : array IOPT(40)
C             Serve as parameters for the code. For standard use of the code
C             IOPT(2),..,IOPT(14) must be set to zero before calling.
C             See below for a more sophisticated use.
C
C             IOPT( 2)=LUN output device
C                      0 - no output (default)
C                      6 - output to the screen
C                      10- output to gmkslog.log
C                      In the case that the output of detailed information
```

```
C                     is desired, the user has to define an output device
C                     and to associate this device with IOPT(2), i.e.,
C                       IOPT(2)=13
C                       OPEN(UNIT=13,FILE='gmkssol.log')
C                     Finally, the output device has to be closed, i.e.,
C                       CLOSE(13)
C                     In the case of an unsuccessful run of GMKSSOL it is
C                     recommended to set IOPT(2) > 0 such that the GMKSSOL is
C                     able to provide more detailed information to the user.
C                     Furthermore, it is recommended to set
C                       IOPT(2)=0, 6, or >10.
C
C            IOPT( 3)=MAXSTP maximal number of allowed steps
C                    <0 no restriction (be careful)
C                    =0 => 1000 (default)
C                    >0 maximal number of steps is given by IOPT(3)
C
C            IOPT( 9) Selectors are constant
C                    =0 -NO
C                    =1 -YES - therefore, only to be computed at initial time
C
C            IOPT(14) compute new selectors
C                    =0 -every successful step
C                    >0 -after IOPT(14) successful steps
C
C     ROPT    Input : integer : array ROPT(40)
C             ROPT( 1)=RTOL Relative tolerance
C                      The default value (for ROPT(1)=0.0) is 1.D-6.
C
C             ROPT( 2)=ATOL Absolute tolerance
C                      The default value (for ROPT(2)=0.0) is 1.D-6.
C
C             ROPT( 3)=INIPREC Precision of the initial values and accuracy
C                      for forced systems.
C                      The default value (for ROPT(3)=0.0) is 1.D-8.
C
C             ROPT( 4)=UROUND   The rounding unit
C                      The default value (for ROPT(4)=0.0) is 1.D-16.
C
C     IPAR    Input  : integer : array IPAR(*)
C             Integer parameter which are only used by the user. They are
C             unused and unchanged in GMKSSOL.
C
C     RPAR    Input  : double precision : array RPAR(*)
C             Double precision parameter which are only used by the user. They
C             are unused and unchanged in GMKSSOL.
C
C     IERR    Output : integer
C             Reports success upon return. The first two digits
C             indicate the subroutine which causes trouble.
C
C       IERR=-10.. An error occurred in the subroutine GMKSSOL
C            -1001 NP lower than or equal zero
C            -1002 NL lower than or equal zero
C            -1003 NR lower than zero
C            -1004 RG not valid
C            -1006 initial values are not consistent
C            -1007 more than NMAX =IOPT(3) steps are needed
C            -1008 insufficient storage for RWORK (adapt LRWORK)
C            -1009 insufficient storage for IWORK (adapt LIWORK)
C            -1010 Stop initiated by EQUOFMOT
C            -1011 Stop initiated by SOLOUT
C            -1012 Tolerances RTOL=ROPT(1) too small
C            -1013 Tolerances ATOL=ROPT(2) too small
C
C       IERR-13.. An error occurred in the subroutine GMSELECT
C            -1301 constraint matrix is not analytically computable
```

```
C            -1302 SVD of constraint matrix failed
C            -1303 rank of G is changing
C            -1304 Z not the Identity not yet implemented
C            -1305 error while computing Equ. of Mot.
C            -1306 selector determination via QR not
C                                        implemented
C
C       IERR=-15.. An error occurred in the subroutine GMRHSIRK
C            -1505 error while computing Equ. of Mot.
C
C       IERR=-16.. An error occurred in the subroutine GMSOLIRK
C            -1601 exit caused by SOLOUT
C            -1602 more than NMAX steps are needed
C            -1603 step size H too small
C            -1604 matrix is repeatedly singular
C
C       IERR=-17.. An error occurred in the subroutine GMKSCOIN
C            -1710 dynamic and kinematic components cannot be set to 0
C            -1720 Newton method did not converge
C             IERR<-1730 error in DGLSY -(IERR+1730)
C
C -------------------------------------------------------------------------
C
C     PROBLEM DESCRIPTION
C  --------------------------
C
C     The user has to supply two subroutines which describe the problem to
C     solve.
C
C     EQUOFMOT User supplied subroutine which provides the
C              right-hand-side (RHS) of EoM (1)-(4)
C
C                 SUBROUTINE EQUOFMOT(T,NP,P,V,A,NR,R,NL,L,NF,RDA,
C                #                    IOPT,ROPT,IPAR,RPAR,IERR)
C                  INTEGER          NP,NR,NL,NF,IERR,IOPT(*),IPAR(*)
C                  DOUBLE PRECISION T,P(NP),V(NP),A(NP),L(NL),R(NR),
C                 #                 RDA(NP+NP+NR+3*NL),ROPT(*),RPAR(*)
C
C                  T    Input  : double precision
C                       Evaluate the RHS at time t.
C                       T has to remain unchanged.
C
C                  NP   Input  : integer
C                       Number of position variables p.
C                       NP has to remain unchanged.
C
C                  P    Input  : double precision : array P(NP)
C                       Position variables p.
C                       P has to remain unchanged.
C
C                  V    Input  : double precision : array V(NP)
C                       Velocity variables v.
C                       V has to remain unchanged.
C
C                  A    Input  : double precision : array A(NP)
C                       Acceleration variables a(=v') (unused).
C                       A has to remain unchanged.
C
C                  NR   Input  : integer
C                       Number of position variables r.
C                       If r does not exist then NR=1.
C                       NR has to remain unchanged.
C
C                  R    Input  : double precision : array R(NR)
C                       Hydraulic variables r.
C                       R has to remain unchanged.
C
```

```
C                NL   Input  : integer
C                     Number of position variables lambda.
C                     NL has to remain unchanged.
C
C                L    Input  : double precision : array L(NL)
C                     Lagrange-Multipliers lambda.
C                     L has to remain unchanged.
C
C                NF   Input  : integer
C                     Degree of freedom.
C                     NF has to remain unchanged.
C
C                RDA  Output : double precision : array RDA(NP+NP+NR+3*NL)
C                     Right-hand-side of reduced derivative array
C                     RDA( 1:NP                              )=v
C                     RDA(NP+1:NP+NP                         )=f_dyn
C                     RDA(NP+NP+1:NP+NP+NR                   )=b
C                     RDA(NP+NP+NR+1:NP+NP+NR+NL             )=g
C                     RDA(NP+NP+NR+NL+1:NP+NP+NR+NL+NL       )=d/dt g
C                     RDA(NP+NP+NR+NL+NL+1:NP+NP+NR+NL+NL+NL)=d^2/dt^2
C
C                IOPT Input  : integer : array IOPT(40)
C                     Integer options (see above)
C                     IOPT(31)=1 then only evaluation of constraints g(p,t)
C                              is needed
C                           else evaluation of the whole RDA is expected
C                     IOPT has to remain unchanged.
C
C                ROPT Input  : double precision : array ROPT(40)
C                     Double precision options (see above)
C                     ROPT has to remain unchanged.
C
C                IPAR Input/Output: integer : array IPAR(*)
C                     Integer parameters which are only used by the user.
C                     They are unused and unchanged in GMKSSOL.
C
C                RPAR Input/Output: double precision : array RPAR(*)
C                     Double precision parameters which are only used by the
C                     user. They are unused and unchanged in GMKSSOL.
C
C                IERR Output : integer
C                     Error message. The user can stop the integration by
C                     setting IERR negative. (IERR will be changed after
C                     returning from EQUOFMOT))
C
C     CONSMATR User supplied subroutine which provides the
C              constraint matrix G (see (5)).
C              Not yet implemented please use dummy routine
C                 SUBROUTINE CONSMATR()
C                 END
C
C -------------------------------------------------------------------------
```

List of Figures

List of Tables

Bibliography

[1] F.M.L. Amirouche. *Computional Methods in Multibody Dynamics*. Prentice Hall, Englewood Cliffs, New Jersey 07632, Chicago, 1992.

[2] E. Anderson and et.al. *LAPACK Users' Guide*. SIAM, 3rd edition, 1999.

[3] T. Andrzejewski, H.G. Bock, E. Eich, and R. von Schwerin. Recent advances in the numerical integration of multibody systems. In W.O. Schiehlen, editor, *Advanced Multibody System Dynamics - Simulation and Software Tools*, pages 127–151. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1993.

[4] T. Andrzejewski, E. Eich, C. Führer, M. Otter, and G. Leister. Entwurf von Schnittstellen zur numerischen Integration von Mehrkörpersystemen. Technical Report TR R 30-90, DLR German Aerospace Center, Oberpfaffenhofen, Germany, 1990.

[5] M. Arnold. A perturbation analysis for the dynamical simulation of mechanical multibody systems. *Applied Numerical Mathematics. An IMACS Journal*, 18(1-3):37–56, 1995. Seventh Conference on the Numerical Treatment of Differential Equations (Halle, 1994).

[6] M. Arnold. Half-explicit Runge-Kutta methods with explicit stages for differential-algebraic systems of index 2. *BIT*, 38(3):415–438, 1998.

[7] M. Arnold. *Zur Theorie und zur numerischen Lösung von Anfangswertproblemen für differentiell-algebraische Systeme von höherem Index*. Fortschritt-Berichte VDI Reihe 20, Nr. 264. VDI–Verlag, Düsseldorf, 1998.

[8] M. Arnold and M. Günther. Preconditioned dynamic iteration for coupled differential-algebraic systems. *BIT*, 41(1):001–025, 2001.

[9] M. Arnold, V. Mehrmann, and A. Steinbrecher. Index reduction in industrial multibody system simulation. Technical Report IB 532–01–01, DLR German Aerospace Center, Institute of Aeroelasticity, Vehicle System Dynamics Group, 2001.

[10] M. Arnold, V. Mehrmann, and A. Steinbrecher. Index reduction of linear equations of motion in industrial multibody system simulation. Technical Report 146, DFG Research Center MATHEON, Technische Universität Berlin, Berlin, Germany, 2004.

[11] M. Arnold and H. Netter. The approximation of contact geometry in the dynamical simulation of wheel-rail systems. *Mathematical and Computer Modelling of Dynamical Systems*, 4:162–184, 1998.

[12] V.I. Arnold. *Mathematical Methods of Classical Mechanics*. Springer-Verlag, Berlin, Germany, 1978.

[13] V.I. Arnold. *Dynamical Systems III*, volume 3. Springer-Verlag, Berlin, Germany, 1993.

[14] U.M. Ascher, H. Chin, L.R. Petzold, and S. Reich. Stabilization of constrained mechanical systems with DAEs and invariant manifolds. *Mechanics of Structures and Machines*, 23:135–157, 1995.

[15] U.M. Ascher and L.R. Petzold. Stabilization of computional methods for constrained dynamics systems. *SIAM Journal on Scientific and Statistic Computing*, 14:95–120, 1993.

[16] L. Auslander and R.E. MacKenzie. *Introduction to differentiable manifolds*. Dover Publications Inc., New York, 1977. Corrected reprinting.

[17] S. Banach. *Mechanics*. Monografie Matematyczne. Polish Mathematical Society, Warszawa, Poland, 1951.

[18] J. Baumgarte. Stabilisation of constraints and integrals of motion in dynamical systems. *Computer Methods in Applied Mechanics and Engineering*, 1:1–16, 1972.

[19] J. Baumgarte. Asymptotische Stabilisierung von Integralen bei gewöhnlichen Differentialgleichungen 1. Ordnung. *Zeitschrift für Angewandte Mathematik und Mechanik*, 53:701–704, 1973.

[20] A. Bellen and M. Zennaro. The use of Runge-Kutta formulae in waveform relaxation methods. *Applied Numerical Mathematics*, 11:95–114, 1993.

[21] V. Brasey. *Half-Explicit Method for Semi-Explicit Differential-Algebraic Equations of Index 2*. PhD thesis, Département. de Mathématiques, Université de Genéve, 1994.

[22] V. Brasey. HEM5 user's guide. Technical report, Département de Mathématiques, Université de Genéve, 1994.

[23] V. Brasey and E. Hairer. Half-explicit Runge-Kutta-methods for differential-algebraic systems of index 2. *SIAM Journal on Numerical Analysis*, 30(2):538–552, 1993.

[24] H. Bremer. *Dynamik und Regelung mechanischer Systeme*. Teubner Studienbücherei, Stuttgart, 1988.

[25] K.E. Brenan, S.L. Campbell, and L.R. Petzold. *Numerical Solution of Initial-Value Problems in Differential Algebraic Equations*, volume 14 of *Classics in Applied Mathematics*. SIAM, Philadelphia, PA, 1996.

[26] P.N. Brown, A.C. Hindmarsh, and L.R. Petzold. Using Krylov methods in the solution of large-scale differential-algebraic systems. *SIAM Journal on Scientific Computing*, 15(6):1467–1488, 1994.

[27] J.C. Butcher. On the implementation of implicit Runge-Kutta methods. *BIT*, 16:237–240, 1976.

[28] J.C. Butcher. *Numerical methods for ordinary differential equations*. John Wiley & Sons Ltd., Chichester, 2003.

[29] S.L. Campbell. A general form for solvable linear time varying singular systems of differential equations. *SIAM Journal on Mathematical Analysis*, 18:1101–1115, 1987.

[30] S.L. Campbell. Linearization of DAE's along trajectories. *Zeitschrift für Angewandte Mathematik und Physik*, 46:70–84, 1995.

[31] S.L. Campbell and C.W. Gear. The index of general nonlinear DAEs. *Numerische Mathematik*, 72(2):173–196, 1995.

[32] S.L. Campbell and E. Griepentrog. Solvability of general differential algebraic equations. *SIAM Journal on Scientific Computing*, 16:257–270, 1995.

[33] F.E. Cellier. *Continuous system modeling*. Springer-Verlag, New York, 1991.

[34] L. Collatz. *The Numerical Treatment of Differential Equations*. Springer-Verlag, Berlin, 3rd edition, 1966.

[35] L. Collatz. *Differentialgleichungen*. Teubner Studienbücherei Mathematik, 2nd edition, Hamburg, 1973.

[36] R. Courant and D. Hilbert. *Methoden der Mathematischen Physik*. Springer-Verlag, Berlin, 4th edition, 1993.

[37] Curtiss.C.F. and J.O. Hirschfelder. Integration of stiff equations. *Proceedings of the National Academy of Sciences of the United States of America*, 38:235–243, 1952.

[38] G. Dahlquist. A special stability problem for linear multistep methods. *Nordisk Tidskr. Informations-Behandling*, 3:27–43, 1963.

[39] J.J.B. de Swart and G. Söderlind. On the construction of error estimators for implicit Runge-Kutta methods. *Journal of Computional and Applied Mathematics*, 86:347–358, 1998.

[40] K. Dekker and J.G. Verwer. *Stability of Runge-Kutta methods for stiff nonlinear differential equations*, volume 2 of *CWI Monographs*. North-Holland Publishing Co., Amsterdam, 1984.

[41] J. Denavit and R.S. Hartenberg. A kinematic notation for lower-pair mechanisms based on matrices. *ASME Journal of Applied Mechanics*, 22:215–221, 1955.

[42] P. Deuflhard. *Newton methods for nonlinear problems. Affine invariance and adaptive algorithms*, volume 35 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, 2004.

[43] P. Deuflhard and F. Bornemann. *Numerische Mathematik II. Integration gewöhnlicher Differentialgleichungen*. de Gruyter Lehrbuch. Walter de Gruyter & Co., Berlin, 1994.

[44] P. Deuflhard, E. Hairer, and J. Zugck. One-step and extrapolation methods for differential-algebraic systems. *Numerische Mathematik*, 51:501–516, 1987.

[45] P. Deuflhard and U. Nowak. Extrapolation integrators for quasilinear implicit ODEs. In P. Deuflhard and B. Engquist, editors, *Large-Scale Scientific Computing*. Birkhäuser, Boston, 1987.

[46] L. Dieci and T. Eriola. On smooth decompositions of matrices. *SIAM Journal on Matrix Analysis and Applications*, 20(20):800–819, 1999.

[47] J. Dieudonné. *Foundations of Modern Analysis*. Academic Press, New York, 1969.

[48] H. Dresig and I.I. Vul'fson. *Dynamik der Mechanismen*. Springer-Verlag, Vienna, 1989.

[49] F. Ebert. A control-theoretic approach to simulator coupling. Diplomarbeit, Institut für Mathematik, Technische Universität Berlin, Berlin, Germany, 2004.

[50] F. Ebert and A. Steinbrecher. Dokumentation verschiedener Löser zur numerischen Integration gewöhnlicher Differentialgleichungen und differentiell-algebraischer Gleichungen in der Mehrkörperdynamik. Technical Report IR-005-04-VE 12/2004, Bosch Rexroth AG, BR / VES (Simulationstechnik), 2004.

[51] E. Eich, C. Führer, B Leimkuhler, and S. Reich. Stabilization and projection methods for multibody dynamics. Technical Report A281, Helsinki Univ. of Technology, 1990.

[52] E. Eich-Soellner and C. Führer. *Numerical Methods in Multibody Dynamics*. B.G.Teubner, Stuttgart, 1998.

[53] A. Eichberger. *Simulation von Mehrkörpersystemen auf parallelen Rechnerarchitekturen*. Number 332 in Fortschritt-Berichte VDI, Reihe 8: Meß-, Steuerungs- und Regelungstechnik. VDI-Verlag Düsseldorf, 1993.

[54] D. Estévez Schwarz and R. Lamour. The computation of consistent initial values for nonlinear index-2 differential-algebraic equations. *Numerical Algorithms*, 26(1):49–75, 2001.

[55] H.O. Fattorini. *Infinite-dimensional optimization and control theory*, volume 62 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, 1999.

[56] G.M. Fichtenholz. *Differential- und Integralrechnung*, volume 1. VEB Deutscher Verlag der Wissenschaften, Berlin 1989.

[57] A. Föppl. *Vorlesung über Technische Mechanik - Die wichtigsten Lehren der höheren Dynamik*, volume 6. Carl Hanser Verlag München, 1909.

[58] R. Frank, J. Schneid, and C.W. Ueberhuber. Stability properties of implicit Runge-Kutta methods. *SIAM Journal on Numerical Analysis*, 22:497–514, 1985.

[59] C. Führer. *Differential-algebraische Gleichungssysteme in mechanischen Mehrkörpersystemen - Theorie, numerische Ansätze und Anwendungen*. PhD thesis, Technische Universität München, 1988.

[60] C. Führer and B.J. Leimkuhler. Numerical solution of differential-algebraic equations for constrained mechanical motion. *Numerische Mathematik*, 59:55–69, 1991.

[61] F.R. Gantmacher. *The Theory of Matrices*, volume 2. AMS Chelsea Publishing, New York, NY, 1998. Transl. from the Russian by K. A. Hirsch. Reprint of the 1959 translation.

[62] F.R. Gantmacher. *The Theory of Matrices*, volume 1. AMS Chelsea Publishing, New York, NY, 1998. Transl. from the Russian by K. A. Hirsch. Reprint of the 1959 translation.

[63] C.W. Gear. *Numerical Initial Value Problems in Ordinary Differential Equations*. Prentice-Hall Inc. Englewood Cliffs, New Jersey, 1971.

[64] C.W. Gear. Simultaneous numerical solution of differential-algebraic equations. *IEEE Transaction on Circuit Theory*, CT-18(1):89–95, 1971.

[65] C.W. Gear. Maintaining solution invariants in the numerical solution of ODEs. *SIAM Journal on Scientific and Statistic Computing*, 7(3):734–743, 1986.

[66] C.W. Gear. Differential-algebraic equation index transformations. *SIAM Journal on Scientific and Statistic Computing*, 9:39–47, 1988.

[67] C.W. Gear. Differential algebraic equations, indices, and integral algebraic equations. *SIAM Journal on Numerical Analysis*, 27(6):1527–1534, 1990.

[68] C.W. Gear, B. Leimkuhler, and G.K. Gupta. Automatic integration of Euler-Lagrange equations with constraints. *Journal of Computional and Applied Mathematics*, 12/13:77–90, 1985.

[69] C.W. Gear and L.R. Petzold. ODE methods for the solution of differential/algebraic systems. *SIAM Journal on Numerical Analysis*, 21:716–728, 1984.

[70] C.W. Gear and K.W. Tu. The effect of variable mesh size on the stability of multistep methods. *SIAM Journal on Numerical Analysis*, 11(5):1025–1043, 1974.

[71] C.W. Gear and D.S. Watanabe. Stability and convergence of variable order multistep methods. *SIAM Journal on Numerical Analysis*, 11:1044–1058, 1974.

[72] G.H. Golub and C.F. van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, MD, 3rd edition, 1996.

[73] E. Griepentrog and R. März. *Differential-Algebraic Equations and Their Numerical Treatment*, volume 88 of *Teubner-Texte zur Mathematik*. BSB B.G.Teubner Verlagsgesellschaft, Leipzig, 1986.

[74] R.D. Grigorieff. *Numerik gewöhnlicher Differentialgleichungen*, volume 1. Teubner Studienbücher Mathematik, Berlin, 1972.

[75] R.D. Grigorieff. *Numerik gewöhnlicher Differentialgleichungen*, volume 2. Teubner Studienbücher Mathematik, Berlin, 1977.

[76] W.v. Grünhagen. *Zur Stabilisierung der numerischen Integration von Bewegungsgleichungen*. PhD thesis, Universität Braunschweig, 1979.

[77] M. Günther and P. Rentrop. Suitable one-step methods for quasilinear-implicit ODE's. Technical Report TUM-M9405, Technische Universität München, 1994.

[78] K. Gustafsson. Control-theoretic techniques for step size selection in implicit Runge-Kutta methods. *Association for Computing Machinery. Transactions on Mathematical Software*, 20:496–517, 1994.

[79] E. Hairer, C. Lubich, and M. Roche. *The Numerical Solution of Differential-Algebraic Systems by Runge-Kutta Methods*. Springer-Verlag, Berlin, Germany, 1989.

[80] E. Hairer, C. Lubich, and G. Wanner. *Geometric Numerical Integration*. Springer-Verlag, Berlin, Germany, 2002.

[81] E. Hairer, S.P. Norsett, and G. Wanner. *Solving Ordinary Differential Equations I - Nonstiff Problems*. Springer-Verlag, Berlin, Germany, 2nd edition, 1993.

[82] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II - Stiff and Differential-Algebraic Problems*. Springer-Verlag, Berlin, Germany, 2nd edition, 1996.

[83] G. Hamel. *Theoretische Mechanik*. Springer-Verlag, Berlin, 1967.

[84] M. Hanke and R. Lamour. Consistent initialization for nonlinear index-2 differential-algebraic equation: large sparse systems in MATLAB. *Numer. Algorithms*, 32(1):67–85, 2003.

[85] E.J. Haug, editor. *Computer Aided Analysis and Optimization of Mechanical System Dynamics*, volume 9 of *NATO Advanced Science Institutes Series F: Computer and Systems Sciences*, Berlin, Germany, 1984. Springer-Verlag.

[86] E.J. Haug. *Computer Aided Kinematics and Dynamics of Mechanical Systems*, volume 1: Basic Methods. Allyn & Bacon, Boston, 1989.

[87] H. Heuser. *Lehrbuch der Analysis*, volume 1. B.G.Teubner Stuttgart, 8th edition, 1990.

[88] H. Heuser. *Gewöhnliche Differentialgleichungen*. B.G. Teubner Stuttgart, 1991.

[89] H. Heuser. *Lehrbuch der Analysis*, volume 2. B.G.Teubner Stuttgart, Karlsruhe, 7th edition, 1992.

[90] A.C. Hindmarsh. LSODE and LSODI, two new initial value ordinary differential equation solvers. *ACM-SIGNUM Newsletter*, 15:10–11, 1980.

[91] W.W. Hooker and G. Margulies. The dynamical attitude equations for n-body satellite. *Journal on Astronomical Science*, 12:123–128, 1965.

[92] R.A. Horn and C.R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

[93] C.T. Kelley. *Iterative methods for linear and nonlinear equations*, volume 16 of *Frontiers in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1995.

[94] C.T. Kelley. *Solving nonlinear equations with Newton's method*. Fundamentals of Algorithms. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2003.

[95] A. Kiełbasiński and H. Schwetlick. *Numerische lineare Algebra*. Verlag Harri Deutsch, Warschau/Halle 1986.

[96] E. Klingbeil. *Variationsrechnung*. Wissenschaftsverlag Mannheim–Wien–Zürich, 1988.

[97] M. Knorrenschild. *Regularisierung von Differentiell-Algebraischen Systemen - theoretische und numerische Aspekte*. PhD thesis, Rheinisch-Westfälische Technische Hochschule Aachen, 1988.

[98] E. Kreuzer. *Symbolische Berechnung der Bewegungsgleichungen von Mehrkörpersystemen*. Number 32 in Fortschritt-Berichte VDI Reihe 11. VDI Verlag, Düsseldorf, Germany, 1979.

[99] P. Kunkel and V. Mehrmann. Smooth factorization of matrix valued functions and their derivatives. *Numerische Mathematik*, 60:115–132, 1991.

[100] P. Kunkel and V. Mehrmann. Canonical forms for linear differential-algebraic equations with variable coefficients. *Journal of Computional and Applied Mathematics*, 56:225–251, 1994.

[101] P. Kunkel and V. Mehrmann. Local and global invariants of linear differential-algebraic equations and their relation. *Electronic Transactions on Numerical Analysis*, 4:138–157, 1996.

[102] P. Kunkel and V. Mehrmann. A new class of discretization methods for the solution of linear differential-algebraic equations with variable coefficients. *SIAM Journal on Numerical Analysis*, 33(5):1941–1961, 1996.

[103] P. Kunkel and V. Mehrmann. Regular solutions of nonlinear differential-algebraic equations and their numerical determination. *Numerische Mathematik*, 79:581–600, 1998.

[104] P. Kunkel and V. Mehrmann. Analysis of over- and underdetermined nonlinear differential-algebraic systems with application to nonlinear control problems. *Mathematics of Control, Signals and Systems*, 14:233–256, 2001.

[105] P. Kunkel and V. Mehrmann. *Differential-Algebraic Equations. Analysis and Numerical Solution*. EMS Publishing House, Zürich, Switzerland, 2006. To appear.

[106] P. Kunkel, V. Mehrmann, and W. Rath. Analysis and numerical solution of control problems in descriptor form. *Mathematics of Control, Signals, and Systems*, 14(1):29–61, 2001.

[107] P. Kunkel, V. Mehrmann, W. Rath, and J. Weickert. A new software package for linear differential-algebraic equations. *SIAM Journal on Scientific Computing*, 18(1):115–138, 1997.

[108] P. Kunkel, V. Mehrmann, and S. Seidel. A MATLAB toolbox for the numerical solution of differential-algebraic equations. Technical Report 16-2005, Institut für Mathematik, Technische Universität Berlin, Berlin, Germany, 2005.

[109] P. Kunkel, V. Mehrmann, and I. Seufer. GENDA: A software package for the solution of general nonlinear differential-algebraic equations. Technical Report 730-02, Institut für Mathematik, Technische Universität Berlin, Berlin, Germany, 2002.

[110] J.L. Lagrange. *Méchanique analytique*. Paris, 1788.

[111] C.L. Lawson, R.J. Hanson, D.R. Kincaid, and F.T. Krogh. Basic Linear Algebra Subprograms for Fortran usage. *ACM Transactions on Mathematical Software*, 5(3):308–323, September 1979.

[112] M. Lehner. Entwicklung eines Simulationsmoduls zur Analyse von Mehrkörpersystemen in einem übergeordneten CAE-System. Diplomarbeit, Lehrstuhl für Mechanik der Universität Erlangen-Nürnberg, 2002.

[113] P. Lötstedt. Mechanical systems of rigid bodies subject to unilateral constraints. *SIAM Journal on Applied Mathematics*, 42:281–296, 1982.

[114] P. Lötstedt. On the relation between singular perturbation problems and differential-algebraic equations. Technical Report 100, Dept. of Computer Science, Uppsala University, 1985.

[115] P. Lötstedt and L.R. Petzold. Numerical solution of nonlinear differential equations with algebraic constraints. Technical Report SAND83-8877, Sandia National Laboratories, Livermore,CA, 1983.

[116] C. Lubich. Extrapolation integrators for constrained multibody systems. *Impact of Computing in Science and Engineering*, 3:213–234, 1991.

[117] C. Lubich. Integration of stiff mechanical systems by Runge-Kutta methods. *Zeitschrift für Angewandte Mathematik und Physik*, 44:1022–1053, 1993.

[118] C. Lubich, U. Nowak, U. Pöhle, and C. Engstler. MEXX – numerical software for the integration of constrained mechanical multibody systems. Preprint SC 92-12, Konrad-Zuse-Zentrum für Informationstechnik Berlin, dec 1992.

[119] R. März. Criteria for the trivial solution of differential algebraic equations with small nonlinearities to be asymptotically stable. *Journal of Mathematical Analysis and Applications*, 225:587–607, 1998.

[120] R. März. The index of linear differential algebraic equations with properly stated leading terms. *Results in Mathematics*, 42:308–338, 2002.

[121] R. März. Characterizing differential algebraic equations without the use of derivative arrays. *Computers and mathematics with applications*, 2005.

[122] S.E. Mattsson, H. Elmqvist, and J.F. Broenink. Modelica: An international effort to design the next generation modelling language. *Journal A, Benelux Quarterly Journal on Automatic Control*, 38(3):16–19, 1997.

[123] U. Miekkala and O. Nevanlinna. Convergence of dynamic iteration methods for initial value problems. *SIAM Journal on Scientific and Statistic Computing*, 8:459–482, 1987.

[124] U. Miekkala and O. Nevanlinna. Sets of convergence and stability regions. *BIT*, 27:557–584, 1987.

[125] A.R. Mitchell and J.W. Craggs. Stability of difference relations in the solution of ordinary differential equations. *Mathematical tables and other aids to computation*, 7:127–129, 1953.

[126] D. Morgenstern and I. Szabó. *Vorlesungen über theoretische Mechanik*. Die Grundlehren der mathematischen Wissenschaften, Bd. 112. Springer-Verlag, Berlin, 1961.

[127] D. Negrut, E.J. Haug, and H.C. German. An implicit Runge-Kutta method for integration of differential algebraic equations of multibody dynamics. *Multibody System Dynamics*, 9:121–142, 2003.

[128] S.I. Newton. *Philosophiae naturalis principia mathematica*. 1687.

[129] J. Nielsen. *Vorlesung über elementare Mechanik*. Springer-Verlag, Berlin, Germany, 1935.

[130] J.M. Ortega and W.C. Rheinboldt. *Iterative solution of nonlinear equations in several variables*, volume 30 of *Classics in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000. Reprint of the 1970 original.

[131] M. Otter. *Objektorientierte Modellierung mechatronischer Systeme am Beispiel geregelter Roboter*. Fortschritt-Berichte VDI Reihe 20, Nr. 147. VDI–Verlag, Düsseldorf, 1995.

[132] C.C. Pantelides. The consistent initialization of differential-algebraic systems. *SIAM Journal on Scientific and Statistic Computing*, 9(2):213–231, 1988.

[133] L.R. Petzold. Differential/algebraic equations are not ODEs. *SIAM Journal on Scientific and Statistic Computing*, 3:367–384, 1982.

[134] L.R. Petzold. Order results for implicit Runge-Kutta methods applied to differential/algebraic systems. *SIAM Journal on Numerical Analysis*, pages 837–852, 1986.

[135] L.R. Petzold. A description of DASSL: A differential/algebraic system solver. In R.S. Stepleman and et al., editors, *Scientific Computing*, pages 65–68. North Holland, Amsterdam, 1983.

[136] L.R. Petzold and P. Lötstedt. Numerical solution of nonlinear differential equations with algebraic constraints ii: Practical implications. *SIAM Journal on Scientific and Statistic Computing*, 7(3):720–733, 1986.

[137] A. Prothero and A. Robinson. On the stability and accuracy of one-step methods for solving stiff systems of ordinary differential equations. *Math. of Comput.*, 28:145–162, 1974.

[138] P.J. Rabier and W.C. Rheinboldt. A general existence and uniqueness theory for implicit differential algebraic equations. *Differential and Integral Equations*, 4:563–582, 1991.

[139] P.J. Rabier and W.C. Rheinboldt. On the numerical solution of the Euler-Lagrange equations. *SIAM Journal on Numerical Analysis*, 32(1):318–329, 1995.

[140] P.J. Rabier and W.C. Rheinboldt. *Nonholonomic Motion of Rigid Mechanical Systems from a DAE Viewpoint*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000.

[141] P. Rabinowitz. *Numerical Methods for Nonlinear Algebraic Equations*. Gordon and Breach Science Publishers, London, 1970.

[142] G. Reißig, W.S. Martinson, and P.I. Barton. Differential-algebraic equations of index 1 may have an arbitrarily high structural index. *SIAM Journal on Scientific Computing*, 21(6):1987–1990, 2000.

[143] R. Remmert. *Funktionentheorie 1*. Springer, Berlin, 4th edition, 1995.

[144] P. Rentrop, K. Strehmel, and R. Weiner. Ein Überblick über Einschrittverfahren zur numerischen Integration in der technischen Simulation. *GAMM-Mitteilungen*, 1:9–43, 1996.

[145] W.C. Rheinboldt. Differential-algebraic systems as differential equations on manifolds. *Mathematics of Computation*, 43(168):473–482, 1984.

[146] W.C. Rheinboldt. On the computation of multi-dimensional solution manifolds of parametrized equations. *Numerische Mathematik*, 53:165–181, 1988.

[147] R.E. Roberson and R. Schwertassek. *Dynamics of multibody systems.* Springer-Verlag, Berlin, Germany, 1988.

[148] D. Rüdiger and A. Kneschke. *Technische Mechanik - Kinematik, Kinetik*, volume 3. Teubner Verlag Leipzig, 1964.

[149] W. Rulka. Effiziente Simulation der Dynamik mechatronischer Systeme für industrielle Anwendungen. Technical Report IB 532–01–06, DLR German Aerospace Center, Institute of Aeroelasticity, Vehicle System Dynamics Group, 2001.

[150] R.R. Ryan. Adams-multibody system analysis software. In *Schiehlen W.O. (ed.): Multibody system handbook*, pages 361–402. Springer-Verlag, Berlin, Germany, 1990.

[151] M. Schaub. *Numerische Integration steifer mechanischer Systeme mit impliziten Runge-Kutta-Verfahren.* Fortschritt-Berichte VDI Reihe 20, Nr. 384. VDI–Verlag, Düsseldorf, 2004.

[152] M. Schaub and B. Simeon. Blended Lobatto methods in multibody dynamics. *Zeitschrift für Angewandte Mathematik und Mechanik*, 83:720–728, 2003.

[153] W.O. Schiehlen. Computer generation of equations of motion. In E.J. Haug, editor, *Computer Aided Design and Optimization of Mechanical System Dynamics*, pages 183–215. Springer-Verlag, 1984.

[154] W.O. Schiehlen. *Multibody System Handbook.* Springer-Verlag, Berlin, Germany, 1990.

[155] W.O. Schiehlen. *Advanced Multibody System Dynamics.* Kluwer Academic Publishers, Dordrecht, The Netherlands, 1993.

[156] W.O. Schiehlen. Multibody system dynamics: Roots and perspectives. *Multibody System Dynamics*, 1:149–188, 1997.

[157] S. Schlauch. Modeling of stirred liquid-liquid dispersions. Technical Report 41-2004, Institut für Mathematik, Technische Universität Berlin, Berlin, Germany, 2004.

[158] W. Schultz-Piszachich. *Tensoralgebra und -analysis*, volume 11 of *Mathematik für Ingenieure, Naturwissenschaftler, Ökonomen und Landwirte.* Teubner Verlag Leipzig, 4th edition, 1988.

[159] S. Schulz. General linear methods for nonlinear DAEs in circuit simulation. Technical Report 20-2004, Institut für Mathematik, Humboldt-Universität zu Berlin, 2004.

[160] R. Schwertassek and R.E. Roberson. A perspective on computer-oriented multibody dynamical. In G. Bianchi and W.O. Schiehlen, editors, *Dynamics of Multibody Systems*, pages 263–273. Springer-Verlag, Berlin, 1986.

[161] L.F. Shampine and M.K. Gordon. *Computer Solution of Ordinary Differential Equations, The Initial Value Problem.* Freeman and Company, San Francisco, 1975.

[162] L.M. Silverman. Inversion of multivariable linear systems. *IEEE Transactions on Automatic Control*, AC-14:270–276, 1969.

[163] B. Simeon. *Numerische Integration mechanischer Mehrkörpersysteme: Projizierende Deskriptorformen, Algorithmen und Rechenprogramme*. Fortschritt-Berichte VDI Reihe 20, Nr. 130. VDI Verlag, Düsseldorf, Germany, 1994.

[164] B. Simeon. MBSPACK — Numerical integration software for constrained mechanical motion. *Surveys on Mathematics for Industry*, 5(3):169–202, 1995.

[165] B. Simeon. *Numerische Simulation gekoppelter Systeme von partiellen und differential-algebraischen Gleichungen in der Mehrkörperdynamik*. Fortschritt-Berichte VDI Reihe 20, Nr. 325. VDI–Verlag, Düsseldorf, 2000.

[166] B. Simeon, C. Führer, and P. Rentrop. Diffrential-algebraic equations in vehicle system dynamics. *Surveys on Mathematics for Industry*, 1:1–37, 1991.

[167] B. Simeon, F. Grupp, C. Führer, and P. Rentrop. A nonlinear truck model and its treatment as a multibody system. *Journal of Computional and Applied Mathematics*, 50:523–532, 1994.

[168] S.N. Singh. A modified algorithm for invertibility in nonlinear systems. *IEEE Transactions on Automatic Control*, 26(2):595–598, 1981.

[169] G. Söderlind. The automatic control of numerical integration. *CWI Quarterly*, 11(1):55–74, 1998.

[170] A. Steinbrecher. Regularization of nonlinear equations of motion of multibody systems by index reduction with preserving the solution manifold. Technical Report 742-02, Institut für Mathematik, Technische Universität Berlin, Berlin, Germany, 2002.

[171] G.W. Stewart. *Matrix algorithms. Vol. I. Basic decompositions*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1998.

[172] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*. Texts in Applied Mathematics, Vol.12. Springer-Verlag, 3rd edition, 2002.

[173] K. Strehmel and R. Weiner. *Numerik gewöhnlicher Differentialgleichungen*. Teubner Studienbücher Mathematik, Stuttgart 1995.

[174] I. Szabó. *Höhere Technische Mechanik*. Berlin-Heidelberg-New York: Springer-Verlag, 5th edition, 1972.

[175] C. Tischendorf. On the stability of solutions of autonomous index-1 tractable and quasilinear index-2 tractable DAEs. *Circuits Systems Signal Process*, 13(2-3):139–154, 1994.

[176] L.N. Trefethen and D. Bau, III. *Numerical linear algebra*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1997.

[177] P.E. Van Keken, D.A. Yuen, and L.R. Petzold. DASPK: a new high order and adaptive time-integration technique with applications to mantle convection with strongly temperature- and pressure-dependent rheology. *Geophysical and Astrophysical Fluid Dynamics*, 80(1-2):57–74, 1995.

[178] M. van Veldhuizen. D-stability. *SIAM Journal on Numerical Analysis*, 18(1):45–64, 1981.

[179] S. Voigtman. General linear methods for linear DAEs. Technical Report 10(2003), Institut für Mathematik, Humboldt-Universität zu Berlin, Berlin, Germany, 2003.

[180] E.T. Whittaker. *A Treatise on the Analytical Dynamics of Particles and Rigid Bodies.* Cambridge University Press, Cambridge, 1959.

[181] J. Wittenburg. *Dynamics of Systems of Rigid Bodies.* B.G. Teubner, Stuttgart, 1977.

[182] A.C. Zaanen. *Linear analysis. Measure and integral, Banach and Hilbert space, linear integral equations.* Interscience Publishers Inc., New York, 1953.

[183] E. Zeidler. *Nonlinear functional analysis and its applications I. Fixed-point theorems.* Springer-Verlag, New York, 1986.

Index