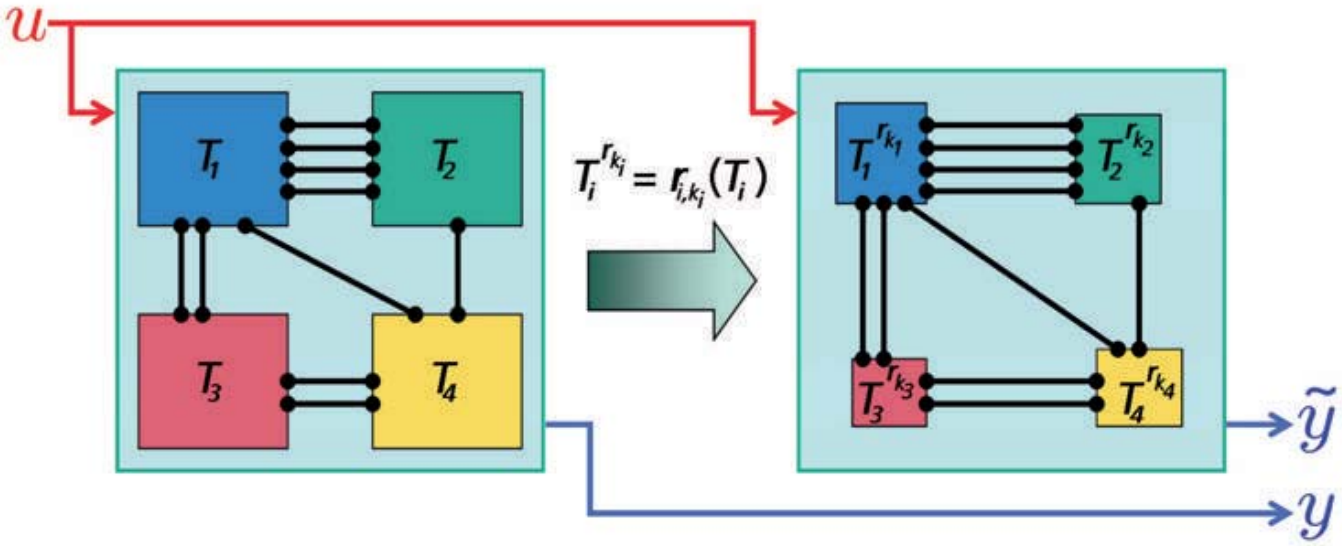
Structure-Exploiting Coupled Symbolic-Numerical Model Reduction For Electrical Networks

Oliver Schmidt



Cuvillier Verlag Göttingen
Internationaler wissenschaftlicher Fachverlag

Structure-Exploiting Coupled Symbolic-Numerical Model Reduction For Electrical Networks

Vom Fachbereich Mathematik
der Technischen Universität Kaiserslautern
zur Verleihung des akademischen Grades
Doktor der Naturwissenschaften
(Doctor rerum naturalium, Dr. rer. nat.)
genehmigte

Dissertation

von

Dipl.-Math. Oliver Schmidt

Kaiserslautern 2010

D 386

1. Gutachter:	Prof. Dr. Dr. h.c. Gert-Martin Greuel
2. Gutachter:	Prof. Dr. Michael Günther
Datum der Disputation:	8. Juli 2010

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.
1. Aufl. - Göttingen : Cuvillier, 2010
Zugl.: Kaiserslautern (TU), Diss., 2010

978-3-86955-502-7

© CUVILLIER VERLAG, Göttingen 2010
Nonnenstieg 8, 37075 Göttingen
Telefon: 0551-54724-0
Telefax: 0551-54724-21
www.cuvillier.de

Alle Rechte vorbehalten. Ohne ausdrückliche Genehmigung des Verlages ist es nicht gestattet, das Buch oder Teile daraus auf fotomechanischem Weg (Fotokopie, Mikrokopie) zu vervielfältigen.
1. Auflage, 2010
Gedruckt auf säurefreiem Papier

978-3-86955-502-7

To my family.

Vorwort

Das Schreiben einer Dissertation ohne verschiedene Arten von Unterstützung ist nahezu unmöglich. Zum Gelingen der vorliegenden Arbeit haben zahlreiche Personen und Institutionen beigetragen, denen ich im Folgenden meinen Dank und meine Anerkennung aussprechen möchte.

Zunächst gilt mein Dank meinem Doktorvater, Herrn Prof. Dr. Dr. h.c. Gert-Martin Greuel, für das Bereitstellen des Themas, die Betreuung der Arbeit und für das Erstellen des Gutachtens. Außerdem danke ich ihm für die moralische Unterstützung insbesondere in der Endphase des Schreibens der Dissertation. Sein Interesse und Engagement sorgten stets für einen neuen Motivationsschub.

Desweiteren danke ich Herrn Prof. Dr. Michael Günther (Bergische Universität Wuppertal) für das Erstellen des Zweitgutachtens und für das Zusenden eines Exemplars seiner Habilitationsschrift, die mir bei der Anfertigung meiner Dissertation hilfreich gewesen ist.

Ermöglicht wurde meine Forschungstätigkeit von Oktober 2006 bis September 2009 zunächst durch ein Stipendium des DFG-Graduiertenkollegs "Mathematik und Praxis" an der Technischen Universität Kaiserslautern. Seit Oktober 2009 stellte die Abteilung *Systemanalyse, Prognose und Regelung* des Fraunhofer Instituts für Techno- und Wirtschaftsmathematik (ITWM) in Kaiserslautern finanzielle Mittel ebenfalls in Form eines Stipendiums dafür bereit. Daher möchte ich beiden Institutionen meinen besonderen Dank aussprechen.

Ein weiterer großer Dank gilt den Mitarbeitern der o.g. Abteilung des ITWM, wo ich die meiste Zeit während meiner Promotion verbrachte. Besonders hervorheben möchte ich dabei zunächst Herrn Dipl.-Ing. Thomas Halfmann und Herrn Dr. Jochen Broz, sowie Herrn Dr. Alexander Dreyer, in deren Team ich am ITWM mitarbeiten durfte, und die stets Zeit und ein offenes Ohr für fachliche Gespräche und Diskussionen hatten. Außerdem danke ich Thomas und Alexander für das Korrekturlesen eines sehr großen Teils dieser Arbeit, das sicherlich einige Zeit in Anspruch nahm. Ich fühlte mich in ihrem Team stets wohl und gut aufgehoben.

Mein weiterer besonderer Dank gilt Herrn Dr. Patrick Lang, Leiter der Abteilung *Systemanalyse, Prognose und Regelung*, für fachliche Diskussionen und Inspirationen, sowie für das Ermöglichen der Teilnahme am BMBF-Verbundprojekt *SyreNe* und der Übernahme der Kosten für damit verbundene Reisen aus dienstlichem Anlaß.

Weiterhin möchte ich in besonderem Maße den Mitarbeitern und Doktoranden Frau Dipl.-Math. techn. Annette Krengel, Herrn Dipl.-Math. techn. Matthias Hauser, Herrn Dipl.-Math. techn. Christian Salzig, Herrn Eng. Alberto Venturi und Herrn Eng. Carmelo

Vicari, sowie Herrn Dr. Jan Hauth und Herrn Dipl.-Math. Dominik Stahl für zahlreiche fachliche Diskussionen, unsere Zusammenarbeit, das tolle Arbeitsklima in der Abteilung, das Korrekturlesen größerer Teile dieser Dissertation, gemeinsame Freizeitgestaltung und ihre Freundschaft herzlich danken. Insbesondere für die Abschnitte über die Funktionsweise einer Diode und des Operationsverstärkers durfte ich mehrmals von Albertos Wissen profitieren, wofür ich ihm herzlich danken möchte.

Ebenso möchte ich es nicht versäumen, meinen Mitdoktoranden und ehemaligen Mitdoktoranden an der Technischen Universität Kaiserslautern, Herrn Dr. Oleksandr Iena, Herrn M.Sc. Oleksandr Motsak, Herrn Dr. Stanislav Bulygin und Herrn Dr. Oleksandr Manzyuk, für ihre Freundschaft und gemeinsame Unternehmungen zu danken. Desweiteren danke ich Frau Petra Bäsell, Sekretärin in der Arbeitsgruppe *Algebra, Geometrie und Computeralgebra* der Technischen Universität Kaiserslautern, für ihre stets hilfsbereite und herzliche Art.

Mein weiterer Dank gilt den Mitarbeitern des BMBF-Projekts *SyreNe*, insbesondere Frau Dr. Tatjana Stykel für Diskussionen über das input/output-Konzept in der Systemtheorie und Herrn M.Sc. Juan Pablo Amorocho D. für Diskussionen zum *TPWL Approach*, sowie für das Korrekturlesen des entsprechenden Abschnitts in dieser Arbeit. Ebenso möchte ich Herrn Dr. Tim Wichmann für das Zusenden einiger Exemplare seiner Dissertation und sein Interesse an meiner Arbeit, sowie Frau Dipl.-Math. Barbara Jung für ihr ausdauerndes Korrekturlesen von nahezu der gesamten vorliegenden Arbeit meinen herzlichen Dank aussprechen.

Ein nachhaltiger Dank geht an meine Eltern Karin und Gerhard Schmidt, die mich nun schon seit meiner Geburt in jedweder Hinsicht unterstützt haben. Im Grunde genommen wurden mein Studium, meine Promotion und somit auch diese Arbeit erst durch sie ermöglicht.

Schließlich möchte ich es nicht versäumen, meiner Susanne für ihre stete persönliche Unterstützung in den letzten Jahren zu danken, die zusammen mit ihrer Geduld besonders während der Endphase in hohem Maße zur Fertigstellung dieser Arbeit beigetragen haben.

Preface

Simulator tools play an indispensable role in the design of electrical networks already for decades. They allow a beforehand verification of the circuit's performance without having it realized as hardware. Hence, redesigns causing immense time and financial effort are avoided and modifications in design can easily be carried out and tested by subsequent simulation runs.

The development in fabrication technology of integrated circuits (IC) during the last years led to an unprecedented increase of functionality of systems on a single chip. Nowadays, ICs have hundreds of millions of semiconductor devices which are placed in several layers and whose feature sizes reach the nanometer range. Besides the integration of analog and digital circuitry, today's *systems-on-a-chip* sometimes also include mechanical parts. The corresponding mixed-technology design causes multi-physical descriptions of the resulting systems.

At the same time, growing miniaturization and integration density force the modelling of low-level physical effects such as thermal interaction, crosstalk, or electromagnetic radiation, to guarantee that the signal propagation is not defective. However, mathematical model descriptions based on *differential-algebraic equations* (DAE) have almost reached their limit and cannot model the occuring effects on transistor level accurately enough.

As a consequence, one uses *distributed elements* such as transistors and transmission lines which yield supplementary model descriptions based on *partial differential equations* (PDE). These do not only incorporate the dependence on time, but also the spatial dependence. The coupling with DAEs, which model the remaining parts of the circuit, then leads to systems of *partial differential-algebraic equations* (PDAE). A semidiscretization with respect to space finally results in systems of differential-algebraic equations of very large dimension, thus rendering analysis and simulation tasks unacceptably expensive and time consuming.

For these reasons, *model order reduction* (MOR) becomes inevitable. Dedicated techniques in various areas of research have been developed among which the most popular ones are *numerical methods* taylored for linear systems. Besides these, there also exist methods for nonlinear systems and, in particular, *symbolic methods* which allow deeper insights into functional dependences of the circuit's behavior on parameters of the system. *Symbolic* in this sense means that besides the system's variables also its parameters are given as symbols instead of numerical values. Model order reduction in the symbolic case is a rather new and still small area of research [**Hen, SomHenEA, SomKraEA, Wic04**]. In this thesis, we develop a framework and tools for coupled symbolic-numerical model reduction of electrical networks with a given hierarchical structure.

To make this presentation as far as possible self-contained, we include a description of a large part of the modelling of electrical networks given as netlist or circuit diagram. We assume that the *topological hierarchy* of the system is given. This means that, together with the circuit, we are given a segmentation into suitable subcircuits and a connecting network describing the hierarchical structure of the system. Hence, the modelling is closely related to the physical system.

In order to avoid infeasibility of analysis and reduction of systems of ever-growing complexity, new approaches have to be invented for any kind of model reduction. So far model order reduction did not incorporate the hierarchical structure available on circuit level. Therefore, the focus of this thesis is the exploitation of the given hierarchy of highly complex nanoelectronical systems for the use with coupled symbolic-numerical reduction techniques.

A segmentation of the entire circuit according to its subcircuit structure allows a faster processing of smaller subproblems. It further offers possibilities for the coupling of both symbolic and numerical model order reduction techniques, since distinct parts of the circuit can be reduced by different reduction methods. Moreover, for those subcircuits that are symbolically reduced, one can take advantage of their parameterization to gain insights in the circuit's behavior.

By standard graph theoretical methods such as *modified nodal analysis* for transforming a circuit into a describing system of equations, its structural information is lost. In this thesis, we present a new workflow that allows separate reductions of the single subcircuits. After segmenting the entire circuit into its set of subcircuits, information obtained from a previous simulation run is used to proceed with the single subcircuits in the reduction process. Depending on the complexity and the degree of the aspired insights, various symbolic and numerical techniques can be applied to the different subsystems corresponding to the separated subcircuits. Hence, our approach has two new ingredients: first, taking advantage of the circuit's hierarchical subsystem structure, and second, the possibility to couple different reduction techniques of both symbolic and numerical type.

Further, a new concept of *sensitivity analysis* of the entire circuit with respect to its subcircuits is introduced. It measures the influence of the single subcircuits on the behavior of the entire circuit by keeping track of the error on its output. Since the *error functions* used for calculating this error crucially influence the entire reduction process, a number of different such functions have been created resp. investigated in this thesis.

Based on the concept of sensitivities, a new *ranking of subsystem reductions* has been invented. This ranking computes an optimized order of reductions of the circuit's subsystems with the aim of achieving a high degree of reduction for the entire system. Using the ranking to control the entire hierarchical reduction process, appropriate reductions of single subcircuits are carried out in order to meet the user-specified accuracy for the overall reduction. The original subcircuits in the connecting structure finally are replaced by these reduced models. Thus, one obtains an entirely reduced behavioral model for the original circuit. This approach clearly affords considerable complexity reductions of the entire system and facilitates its analysis.

The algorithms are implemented in *Analog Insydes* [**AI**], a software package developed by the Fraunhofer ITWM[1] in Kaiserslautern, Germany. It is based on the computer algebra system *Mathematica* [**MMA**]. In this thesis, we are interested not only in academic algorithms, but also in their application to circuits of industrial size. In order to show their aptitude within this context, the algorithms are successfully applied, with significant savings in computation time, to an operational amplifier typically used in industry. A further application of non-hierarchical reduction techniques to the obtained *interim models* finally yields reduced systems that are not only up to about 20 times faster w.r.t. simulation time. They also prove to be very robust with regard to different inputs such as highly non-smooth pulse excitations. We also compare our results to the usual *non-hierarchical* reduction approach. With a similar level of accuracy, the non-hierarchical approach – depending on the used transistor models – is infeasible or needs significantly higher computational effort than our new *hierarchical* approach[2].

The work is arranged as follows: In Chapter 1, we give an introduction to the topics treated in this thesis. After a brief review of different kinds of system analysis, we motivate the need for model reduction techniques by considering their computational complexity. Finally, this chapter addresses the aims of this thesis in more detail.

In order to carefully connect mathematics with the modelling of electrical circuits, we provide extensive fundamental material about these topics in Chapter 2. To make this work self-contained, we start with a presentation of network analysis techniques for setting up a describing system of equations. Next, a review of the resulting systems is given. We mainly deal with *differential-algebraic equations (DAE)*, i.e. sets of differential equations with additional algebraic constraints, but also *partial-differential equations (PDEs)* and *partial differential-algebraic equations (PDAEs)* are considered. After describing three different numerical analysis methods, basic terms and notions from systems and control theory needed throughout this work are given. Finally, some of the most popular model order reduction techniques for both the numerical and the symbolic case are surveyed in more detail.

Chapter 3 illuminates the structure of electrical circuits. We briefly discuss component-based, coupled, and interconnected systems and further provide short introductions to the behavioral approach as well as a macromodelling concept.

Chapter 4 contains the main new results of this thesis. In the first place, a new reduction algorithm is presented which exploits the system's hierarchical structure by adapting the macromodelling idea. We start with a motivating example and show that reductions exploiting a circuit's hierarchy are very promising and have great potential for large savings in computational complexity. One of the advantages is a faster processing of smaller subcircuits. Furthermore, such an approach allows for the coupling of symbolic and numerical reduction methods for separate subsystems in order to derive a reduced model of the entire system. Following this, a new workflow for separate subsystem

[1] Fraunhofer Institute for Industrial Mathematics (Fraunhofer Institut für Techno- und Wirtschaftsmathematik), Fraunhofer-Platz 1, 67663 Kaiserslautern, Germany (`http://www.itwm.fraunhofer.de`).

[2] All computations in this example are performed symbolically on a nonlinear system of DAEs in the time domain.

reduction is presented. Then, our concept of sensitivity analysis of the entire system with respect to its subsystems is explained. This finally results in the development of the reduction algorithm exploiting the subsystem structure of the entire system. The chapter concludes with an application of this algorithm to the example of an operational amplifier typically used in industrial circuit design.

Chapter 5 provides an overview of the implementations that have been made throughout this thesis. The corresponding algorithms have been implemented in *Analog Insydes* [**AI**], a symbolic analysis tool based on the computer algebra system *Mathematica* [**MMA**]. *Analog Insydes* is capable to carry out complex symbolic analysis and reduction calculations automatically and in correspondence to a user-specified accuracy. It is being developed by the department *System Analysis, Prognosis and Control* at the Fraunhofer ITWM and is the implementation platform for the algorithms developed in this thesis. We implemented prototypical versions of large parts of the new hierarchical reduction approach. Furthermore, new data structures and procedures for their manipulation as well as suitable error functions have been implemented within this thesis. They are successfully used for the hierarchical reduction of the operational amplifier in Section 4.7. Moreover, the corresponding development environment needed for tests and motivating examples of the new approach had to be newly implemented.

Chapter 6 finally offers a summary of this thesis and provides an outlook to this field of research.

Contents

Vorwort v

Preface vii

List of Tables xv

List of Figures xvii

Chapter 1. Introduction 1
1.1. System Analysis 1
1.2. Analysis Methods 2
1.3. Complexity Considerations 3
1.4. Model Order Reduction 4
1.5. Motivations for Symbolic and Numerical Methods 5
1.6. Aim of this Thesis 8

Chapter 2. Foundations 11
2.1. Network Analysis 11
2.1.1. Connection Graphs and Incidence Matrices 12
2.1.2. The Kirchhoff Laws 15
2.1.3. Current-Voltage Relations of Circuit Components 17
2.1.4. Sparse Tableau Analysis 19
2.1.5. Modified Nodal Analysis 20
2.1.6. Further Analysis Methods 22
2.2. Systems of Equations 23
2.2.1. Systems of DAEs 23
2.2.2. Index Concepts 27
2.2.3. Systems of PDAEs 28
2.2.4. Examples 30
2.3. Numerical Analysis Methods 36
2.3.1. Transient Analysis 37
2.3.2. DC Analysis 37
2.3.3. AC Analysis 38
2.4. Systems and Control Theory 41
2.4.1. Reachability 42
2.4.2. Observability 43
2.4.3. Passivity 45
2.4.4. Stability 48
2.5. Model Order Reduction 51

2.5.1. Numerical MOR 51
2.5.2. Symbolic MOR 68

Chapter 3. Hierarchical Systems 77
3.1. Component-based Systems 78
3.2. Coupled and Interconnected Systems 79
3.3. The Behavioral Approach 81
3.4. The Macromodel Concept 84

Chapter 4. Hierarchical Model Order Reduction 87
4.1. Motivating Example 88
4.1.1. Symbolic Reduction of the Entire Circuit 88
4.1.2. Exploitation of the Hierarchical Structure 89
4.1.3. Application of Different Inputs 91
4.1.4. Comparison of the two Approaches 93
4.2. A Workflow for Subcircuit Reductions 93
4.3. Subsystem Sensitivities 95
4.3.1. Term Ranking 96
4.3.2. Clustering 97
4.3.3. Subsystem Sensitivity Analysis 98
4.4. Optimized Order of Subsystem Reductions 100
4.5. Reduction Algorithm Exploiting Circuit Hierarchy 102
4.6. Error Functions 102
4.7. Analysis of the Operational Amplifier op741 106
4.7.1. Reduction using the $\mathcal{L}^2$-Norm Error Function 108
4.7.2. Reduction using the New Error Function 112
4.7.3. Improvement 116
4.7.4. A Hybrid Reduction Approach 121
4.7.5. Model Quality Check 123
4.7.6. Conclusion 125

Chapter 5. Implementations 129
5.1. GetStateSpace and GetDAE 129
5.2. Arnoldi Iteration 130
5.3. Models for Transmission Lines and LTI State Space Systems 131
5.4. Subcircuit Reductions 134
5.5. Development Environments and Error Functions 138

Chapter 6. Summary and Outlook 139
6.1. Summary 139
6.2. Outlook 142

Appendix A. Network Theory 145

Appendix B. Semiconductor Device Modelling – The Drift-Diffusion Model for the Diode 147
B.1. Physics of a Diode 147

B.2. Current Density and Continuity Equations 149
B.3. The Poisson Equation 151
B.4. Initial and Boundary Conditions 151
B.5. Coupling Conditions 153

Appendix C. The Operational Amplifier op741 155

Bibliography 159

Index 167

Curriculum Vitae 171

Lebenslauf 173

List of Tables

3.1 Development of Intel processors. 77

4.1 Results of the two different reduction approaches for the differential-amplifier circuit. 93

4.2 Separate reduction of the seven subcircuits of the op741 using the $\mathcal{L}^2$-norm error function. 109

4.3 Hierarchical reduction using subsystem sensitivities, the $\mathcal{L}^2$-norm error function, and Algorithm 4.5. 111

4.4 Separate reduction of the seven subcircuits of the op741 using the new error function. 114

4.5 Hierarchical reduction using subsystem sensitivities and the new error function. 115

4.6 Hierarchical reduction by taking into account the number of equations and terms of the reduced subsystems. 120

4.7 Hybrid reduced entire systems of the operational amplifier op741. 122

4.8 Comparison of time costs of simulations of the original and different reduced op741 models. 125

List of Figures

1.1 Folded-cascode operational amplifier. 6

1.2 Bode diagram of the amplifier's frequency response. 7

1.3 Approximated formula for the dominant pole pair of the amplifier's transfer function. 7

1.4 Stable behavior after redesign. 7

2.1 An electrical circuit as a network of subsystems. 12

2.2 A circuit graph, tree and link branches, and fundamental loops. 13

2.3 Branch magnitudes of various circuit components. 17

2.4 Equivalent circuit for a small part of length Δx of an electrically long homogeneous transmission line. 32

2.5 The transmission line model in the interconnecting network. 33

2.6 The pn-diode circuit component and its one-dimensional geometry. 34

2.7 The drift-diffusion model for the diode in the interconnecting network. 36

2.8 Linearization of the parabola. 39

2.9 Diagram showing the projections of E and the states x via $\Pi : \mathbb{R}^n \to \mathbb{R}^k \to \mathbb{R}^n$, $x \mapsto W^*x \mapsto VW^*x$. 54

3.1 Moore's Law compared to the development of Intel CPUs. 78

3.2 Acceleration sensor. 80

3.3 Interconnected differential-amplifier circuit. 82

3.4 Graphical description of the differential amplifier on circuit level and in the input-output framework. 83

3.5 Segmentation of a circuit Σ into two subcircuits (macros) P and Q. 85

3.6 Separation of P and Q by introducing current sources that generate appropriate currents. 85

4.1 Differential-amplifier circuit. 88

4.2 Solution and error plots of the original and the reduced model of the amplifier (2% maximum error). 89

4.3 Solution and error plots of the original and the reduced model of the amplifier (10% maximum error). 89

4.4 Differential-amplifier circuit with its "intuitive" hierarchy. 90

4.5 Solution and error plots using the hierarchical reduction approach and reducing (L1) with 3 Arnoldi iterations. 91
4.6 Solution and error plots using the hierarchical reduction approach and reducing (L1) with 5 Arnoldi iterations. 91
4.7 Three other inputs u_1, u_2, and u_3 for V1. 92
4.8 Model quality check applying three different inputs. 92
4.9 Schematic illustration of the workflow for subcircuit reductions. 94
4.10 Schematic illustration of the algorithm for hierarchical model order reduction using sensitivities of subsystems. 103
4.11 Operational amplifier op741 including seven subcircuits. 106
4.12 Input and output data of the operational amplifier. 107
4.13 Comparison of the original and the hierarchically reduced op741 model in the $\mathcal{L}^2$-norm error function case. 110
4.14 Comparison of the original and the non-hierarchically reduced model in the $\mathcal{L}^2$-norm error function case. 112
4.15 Comparison of the original and the hierarchically reduced op741 model obtained by using the new error function. 113
4.16 Comparison of the original and the non-hierarchically reduced op741 model in the new error function case. 116
4.17 Sensitivity diagrams for the seven subsystems of the operational amplifier op741. 119
4.18 Outputs of the original and the reduced system obtained by using further subsystem information. 121
4.19 Outputs of the original and the reduced system obtained by using the hybrid approach. 122
4.20 Reduced model quality check by applying different inputs. 124
4.21 Summary of the reduced models obtained in this section via different reduction approaches. 126

5.1 Spatial discretization grids for the voltages and currents in a transmission line of length l. 132

B.1 A pn-diode and a pnp-transistor with their 2-dimensional geometries. 147
B.2 Ranges of electron energy in a semiconductor. 148
B.3 A 2-dimensional diode model with contacts Γ_1 and Γ_2. 151
B.4 The electrostatic field E between the diode's contacts. 153
B.5 Connecting the drift-diffusion model equations for the diode to the remaining network equations. 154

C.1 Operational amplifier op741, segmented into seven subcircuits. 155

CHAPTER 1

Introduction

This chapter provides an introduction to the topics treated in this thesis. We first motivate the general need for system analysis and later on focus on electrical networks. Different methods for system analysis are surveyed and some historical background information is given especially for symbolic analysis. Using complexity considerations of problems from electrical engineering, the need for model order reduction techniques is motivated, for both the symbolic and the numerical case. The research area of reduction methods is then briefly traced in the fourth section and general ideas for symbolic and the most popular numerical techniques are outlined. Two application examples from weather prediction and electrical circuit design are given to further motivate the need for both types of system analysis in combination with reduction techniques. The chapter finally concludes with a description of the aims of this work.

1.1. System Analysis

In our today's technological world, physical processes are mainly described by mathematical models, i.e. dynamical systems whose future behavior depends on their past evolution. The resulting systems of equations in general consist of *differential-algebraic equations* (DAE), since the differential equations coming from the dynamical part of the *real* system mostly are restricted by certain physical laws contributing additional algebraic constraints. Equations containing PDE parts usually are semidiscretized w.r.t. space. System analysis investigates the behavior of a *real* system by making use of such mathematical models. The input-output behavior of the *real* system then corresponds to solving the mathematical system of equations.

Such mathematical models are employed in versatile areas. Their main application, however, is for simulation and control in order to predict or modify the behavior of a *real* system. Especially for electrical networks, simulator tools play a key role in the design process already for decades. In the 50's and 60's of the last century, electrical circuits were designed by using discrete components for a prototype realization and a subsequent adjustment (*redesign*) by taking measurements. A little later, when integrated circuits (ICs) with increased integration density were introduced, these techniques had reached their limits.

On the other hand, using a simulator permits an easy verification of the circuit performance without having it realized as hardware. Advantages such as the possibility for fast modifications and an easy testing of ideas for design are obvious. Particularly redesigns of an accidentally deficiently designed electrical circuit, accompanied with immense time and financial costs in the industrial production of modern ICs, can be made

on the computer screen and be verified by a subsequent simulation run. Parasitic effects caused by measurement setups such as inductances due to cables are avoided and analysis under very different operating conditions is possible. Therefore, simulator tools are indispensable in modern industrial circuit design.

1.2. Analysis Methods

For the analysis of a system, there are different methods which will be described and compared to each other in this section. While the most widely used method, *numerical analysis*, can be applied to systems of large size, *symbolic analysis* allows deeper insights into functional dependences among system parameters.

Numerical Analysis. The most commonly used method for system analysis is numerical analysis. Here, all parameters in the describing system of equations have to be given by their numerical values. Then the system is solved and the obtained solution data can be analyzed and displayed graphically.

In the context of analog circuits, starting from a netlist description of the circuit topology, numerical simulators are able to automatically generate and subsequently solve a system of describing equations for the circuit's behavior. For the numerical calculation, all parameters of the system have to be given by suitable numerical values, i.e. a complete dimensioning of the circuit's components has to be provided.

Using numerical methods for circuit analysis, lots of important characteristics such as amplification factors of amplifiers are computable. Moreover, numerical analysis is applicable to systems of large size. On the other hand, particularly for its *analog* parts which are usually much smaller than the entire circuit, an accurate prediction of the fully sized system's behavior is opposed by no qualitative insights into functional dependences among system parameters and their effects on the system's behavior. This is due to the output solely consisting of "tables of numbers" which is a severe drawback especially for early stages in the design process. For investigations aiming at a deeper understanding of the circuit functionality, one has to carry out further simulations with different numerical parameter values, but this still does not guarantee any success.

Hand Analysis. If one is interested in an analytical description, e.g. in order to gain insights into the system's behavior, a direct analysis of the system equations is necessary.

Usually the access to "internal" equations of a simulator is not given, so up to around 1970, analytic computations had to be carried out by hand. In order to keep the level of complexity low, simplified models for the system components were used which omit all but the most relevant effects, whereas numerical analysis usually uses very precise models taking almost all possible physical effects into account. The validity of the use of simplified models was mostly proven afterwards, if ever.

By intuition, it is clear that *hand computations* are very tedious and error-prone and, particularly in the context of electrical circuits, need a lot of experience and knowledge in circuit design. Furthermore, there is no possibility for an error check during the computations.

Symbolic Analysis. With the development of computer algebra systems, the approaches of the symbolic *hand analysis* could be automated. Around 1970, the first computer programs for *symbolic circuit analysis* were introduced in order to capture the circuit's behavior analytically and in dependence on its parameters. In this context, *symbolic* means that not only the variables, but also the *parameters* are given as symbols in the corresponding system of equations. Solving the *symbolic* equations then yields visible dependences of the system's behavior on its parameters.

Nevertheless, the limits in efficiency and disc space were reached soon, thus preventing symbolic methods for the use with problems of practical size. Therefore, interest was lost for several years. Just at the end of the 1980's, this area of research became revitalized by the enormous increase of computer performance and the development of *symbolic reduction methods*[1] that allowed an application to circuit problems of larger size. Since then it was possible to use symbolic analysis for *linear* circuits even of industrial size. Due to this, in the 1990's one tried to transfer these approaches to nonlinear circuits [**Bor97, Hen**].

Particularly for early stages in the design process of analog circuits, symbolic analysis tools proved to be useful. Especially when the symbolic equations can be solved explicitly for its output variables, they provide an automatic generation of mathematical formulas that express performance characteristics in terms of circuit parameters. Unlike waveforms produced by numerical simulators, symbolic expressions allow to read off influences of components on circuit characteristics and, hence, to identify those parameters that have to be altered in order to meet certain design specifications.

In combination with *symbolic reduction methods*, symbolic analysis is further used to *automatically* derive *behavioral models*, i.e. parameterized systems of equations describing the approximated circuit behavior, which can be employed for accelerated simulation and optimization of analog circuits. Therefore, symbolic analysis is an indispensable tool that simplifies design, dimensioning, and optimization of analog circuits or, more generally, nonlinear systems.

1.3. Complexity Considerations

In order to get an impression of the high computational complexity, consider the development of ICs or, more precisely, *VLSI circuits* [**Bec, FelParFar, Reis06, Tis**] up to today; while the Intel 4004 released in 1971 incorporated 2300 components with feature sizes of $\sim$10 μm and an operating frequency of 64 kHz, the Intel Pentium 4 released in 2001 already had about 42 million components with dimensions around 180 nm arranged in seven layers, an interconnecting structure of 2 km length, and an operating frequency around 2 GHz (see [**wikipedia**], Figure 3.1, and Table 3.1 in Chapter 3). The *passive* parts modelling the interconnecting structure finally yield systems with $n \approx 10^5$ to 10^6 equations. Consequently, simulations of the full systems cannot be handled anymore.

[1] Sometimes one can find the term *symbolic approximation methods* in literature. It has to be mentioned that these approximations are not meant in the sense of *Taylor approximations* or *interpolation polynomials* etc.

The situation is worse yet in the symbolic case. Even for small circuits, the limits of symbolic circuit analysis, beyond which mathematically exact computations are feasible, are reached quickly. Considering the number of terms in a linear system's *transfer function* as a measure for its computational complexity, an estimate for this number shows that symbolic circuit analysis even for *linear* circuits has an asymptotical order of complexity in between $\mathcal{O}(a^n)$ (exponential) and $\mathcal{O}(n^n)$ (superexponential), where n denotes the number of nodes in the circuit [**Hen, Moo**]. This shows that *exact* symbolic circuit analysis becomes infeasible very quickly. Furthermore, lengthy expressions of more than one line do not allow qualitative insights and exact results turn out to be useless.

The above considerations motivate the need for complexity reduction techniques. They are inevitable not only to avoid the enormous costs of very large systems. Following the maxim that high precision is less important than physical interpretability, particularly *symbolic* reduction methods are designed to neglect all insignificant information from the describing equations. In this context, from now on we will consider *system analysis with complexity reduction* as follows:

> ***System analysis with complexity reduction*** *is the description of a real physical process by using a suitable system of equations together with its* ***complexity reduction*** *with the aim of* ***analyzing the system's behavior*** *and the* ***generation of behavioral models****.*

The following section introduces model order reduction, reviews some of its application areas, and provides some background of its history.

1.4. Model Order Reduction

The general task of model order reduction techniques is the derivation of approximate models from given large-scale systems which have a significantly lower level of complexity, but still capture the dominant input-output behavior of the original system. The approximate model should satisfy certain user-specified accuracy requirements and preserve important system properties such as stability and passivity.

Lots of reduction approaches have been developed in various areas of research such as electrical and mechanical engineering, control design, computational fluid dynamics, or biological and chemical engineering, see, e.g., [**Ant, BenMehSor, ObiAnd, Rew, SchVorRom**] and references therein. The most popular *numerical* techniques are taylored for *linear* systems and rely on projections onto lower dimensional subspaces of the original system's state space. The corresponding projection bases can be constructed by methods based on Krylov subspaces or singular value decompositions (SVD) of appropriate system matrices.

Although Krylov methods have certain drawbacks, they are well-suited for applications with large-scale systems and, therefore, are very popular particularly in electrical engineering. In contrast to this, SVD methods provide error bounds and preserve certain system properties. But since they cause high numerical costs, their applicability is limited to systems of medium size.

Symbolic model order reduction techniques aim at approximating symbolic systems by models with reduced complexity and increased interpretability. They are indeed costly to compute, but particularly for nonlinear systems of DAEs they additionally allow deeper insights into functional relations among the system parameters given symbolically [**Hen, Wic04**]. Symbolic methods further allow the generation of parameterized behavioral models for various uses.

The origin of symbolic analysis of nonlinear systems using symbolic reduction methods is found in applications of analog circuit design. There, it is mainly used in addition to numerical simulations as a tool for design, analysis, dimensioning, and optimization of nonlinear systems. Symbolic reduction methods are hybrid numerical and symbolic algorithms which are able to *automatically* reduce the complexity of a given symbolic system of equations according to a user-specified accuracy. Starting at a netlist-description level, the analog circuit is mapped on a symbolic system of DAEs by means of graph-theoretical methods such as the *modified nodal analysis* (cf. Section 2.1.5). Subsequently, comparisons to numerical reference simulations are used to detect the dominant terms of the system. Neglecting the insignificant ones guarantees the preservation of the system's dominant behavior.

At the Fraunhofer ITWM[2], both numerical and particularly symbolic model reduction techniques for complex systems are being developed and applied. Symbolic reduction techniques are the core of the software package *Analog Insydes* [**AI**] developed by the ITWM which is an add-on for the computer algebra system *Mathematica* [**MMA**]. *Analog Insydes* is the implementation platform for the algorithms developed in this thesis.

1.5. Motivations for Symbolic and Numerical Methods

This section motivates both symbolic and numerical analysis for dynamical systems by describing two applications in which such systems of high complexity arise. While the first example concerns *weather prediction* and *data assimilation*, the second is taken from *electrical engineering*.

North Sea Wave Surge Forecast. For this paragraph, we follow the notes of [**Ant**, Section 2.2.1]. The problem has originally been studied in [**HeeVerSeg, Ver**].

Since parts of The Netherlands are below sea level, the monitoring of wave surges at river mouths is important. In case of need, water barriers can be closed in order to prevent a flooding of the landscape. However, for certain reasons a respective warning has to be given six hours in advance.

To be able to forecast such wave surges, one uses the *shallow water equations*[3] as a PDE model predicting their evolution. In addition to this, the water level and the movement of sea currents are measured at various locations. The resulting data assimilation problem affords a prediction of wave surges based on the model and the measurements.

[2]Fraunhofer Institute for Industrial Mathematics (Fraunhofer Institut für Techno- und Wirtschaftsmathematik), Kaiserslautern, Germany.

[3]The average depth of the North Sea is not more than 100 m.

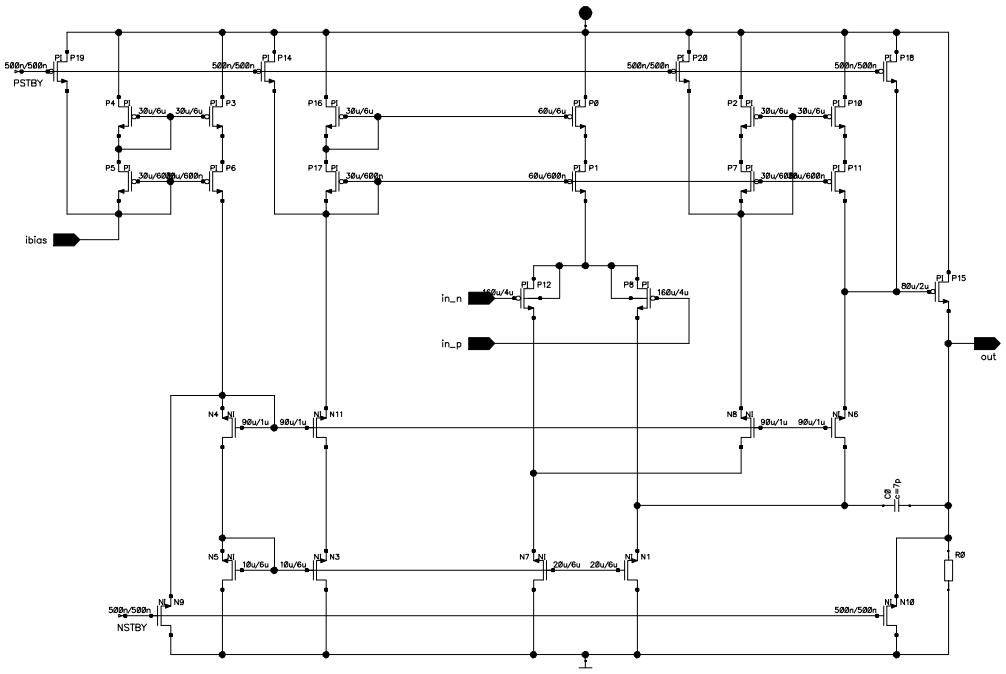


FIGURE 1.1. Folded-cascode operational amplifier.

A space discretization finally yields about $60,000$ equations, whose full computational costs are a multiple of the six-hour limit. Therefore, reduced-order modelling is inevitable. While the complexity of numerical computations is polynomial, for the symbolic case it is at least exponential (cf. Section 1.3). Therefore, it is quite clear that attempts for symbolic analysis in applications of *weather prediction* – at least in this case – are hopeless.

Analysis of a Folded-Cascode Operational Amplifier. The example in this paragraph is partly taken from [**HalWic03**] and describes a problem from industrial circuit design that really occured in 2002. In this case, symbolic model order reduction was used to improve the original design of the circuit.

Consider the folded-cascode operational amplifier depicted in Figure 1.1. The original design showed an instability in the amplifier's small-signal behavior. This was visible as a resonance peak in the amplifier's frequency response at a frequency of approximately 10 MHz (cf. Figure 1.2, solid curves) which was caused by a pair of complex conjugate parasitic poles.

As a classic approach, one would have performed parameter variations and numerical simulations in a trial-and-error fashion to solve the underlying instability problem. But due to a fairly high number of parameters, this was not feasible here.

An estimation showed that the exact symbolic transfer function contains more than $5 \cdot 10^{19}$ terms. However, *symbolic analysis and reduction* methods allowed the computation of an approximate transfer function using a relative error bound of 10%. From this, a symbolic formula for the dominant pole pair shown in Figure 1.3 could be derived.

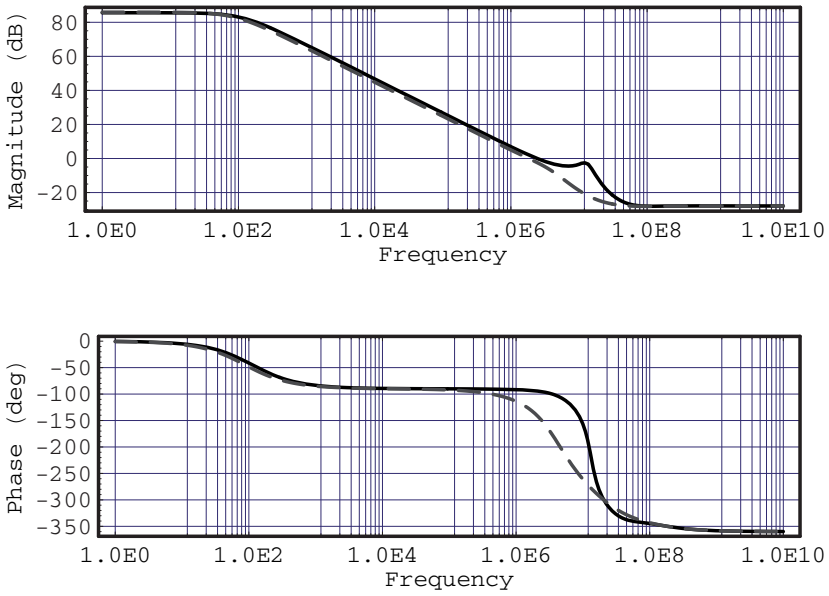


FIGURE 1.2. Bode diagram of the amplifier's frequency response, the original design (solid) and with an additional capacitor (dashed).

$$s_p^{1,2} == -\frac{(CC0 + CL)\ gm\$MN6}{2\ CC0\ CL} \pm \frac{\sqrt{Cgs\$MP15\ gm\$MN6\ (Cgs\$MP15\ (CC0 + CL)^2\ gm\$MN6 - 4\ CC0^2\ CL\ gm\$MP15)}}{2\ CC0\ Cgs\$MP15\ CL}$$

FIGURE 1.3. Approximated formula for the dominant pole pair of the amplifier's transfer function.

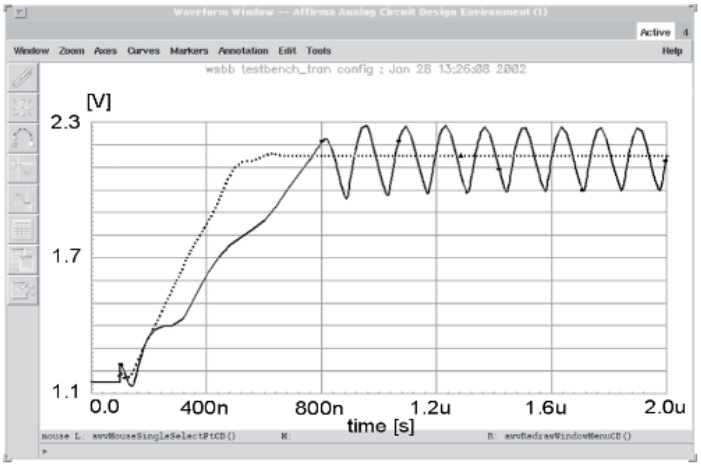


FIGURE 1.4. Stable behavior (dotted) after redesign of the originally unstable behavior (solid).

By interpretation of this formula, it was detected that one of the circuit's components had dominant influence with respect to the unstable behavior. Increasing the value of the gate-source capacitance `Cgs$MP15` of the transistor MP15 allowed for decreasing the

imaginary parts of the pole pair. Hence, by adding an additional capacitor between the gate and source terminals of this transistor, the original value could be modified and the resonance could be damped (cf. frequency analysis in Figure 1.2, dashed curves). Thus, a stable behavior was obtained after a redesign of the amplifier (cf. time-domain analysis in Figure 1.4, dotted curve).

1.6. Aim of this Thesis

In order to enlarge feasibility of analysis and reduction tasks to electrical circuits of ever-growing complexity, the focus of this thesis is the exploitation of the hierarchical subsystem structure of highly complex nanoelectronical systems. A segmentation of the entire circuit according to its structure then yields a set of interacting subcircuits and a coupling network. This allows an accelerated processing of separated subcircuits of smaller size in order to finally recombine a significantly reduced approximate model of the entire circuit.

Additionally, possibilities for the coupling of symbolic and numerical techniques are aspired. This allows for treating different parts of the circuit with suitable methods corresponding to their level of complexity. For example, those parts in a system of PDAEs that correspond to distributed elements such as transmission lines are modelled using PDEs. A spatial semidiscretization of these parts usually results in very large systems of DAEs which then can be processed using appropriate numerical approximation techniques. On the other hand, the parts yielding smaller systems can be treated by using symbolic methods. This allows for gaining deeper insight into functional relations and influences on the circuit's behavior by taking advantage of their parameterization.

So far model order reduction techniques, particularly in the symbolic case, did not incorporate the subsystem structure available on circuit level. Using methods such as the *modified nodal analysis* to set up describing equations for the circuit's behavior, the structural information is lost. We present a new workflow that allows for reducing subcircuits separately. After a segmentation of the entire circuit into a suitable set of subcircuits, their terminals are connected to voltage sources supplying voltage potentials obtained from a previous simulation run. In a second step, describing sets of equations are set up for the "closed circuits" formed by the separated subcircuits and the voltage sources. Depending on the complexity and the degree of the aspired insights, various symbolic and numerical techniques can be applied to these systems of equations. This affords an innovative coupling of different symbolic and numerical reduction methods for the derivation of an entirely reduced behavioral model.

We further introduce a new concept of sensitivity analysis of the entire circuit with respect to its subcircuits. A given subcircuit in the interconnecting structure is sequentially replaced by appropriate models of reduced complexity. Each time the error on the output of the entire system is observed in order to measure the subcircuit's influence on the overall circuit's performance. For keeping track of this error, a number of *error functions* accounting for different system characteristics have been implemented. Then, the subcircuit's sensitivity is defined as a vector of tuples containing reduction information

and the resulting errors on the output. The same procedure is repeated for each of the remaining subcircuits to finally obtain all their sensitivities.

Ordering the entries in the sensitivity vectors increasingly w.r.t. the resulting error on the output leads to a new *ranking of subsystem reductions*. In each step of the entire reduction process, the reduced model that causes the least error according to the ranking is used to replace the corresponding subcircuit in the overall circuit. If the accumulated error does not exceed the user-given error bound, the corresponding entry is deleted from the ranking. Then the procedure is repeated. Thus, the subsystems are reduced in an optimized order. This guarantees the user-specified accuracy and a high degree of reduction for the reduced entire system. This new approach yields considerably reduced entire systems and, hence, clearly facilitates their analysis.

In order to show its applicability to *real* systems of current industrial size, the above algorithms are implemented in *Mathematica* for the use in *Analog Insydes* [**AI**]. As an endurance test, they are successfully and with significant savings in computation time applied to an operational amplifier typically used in industry. The obtained models have very low levels of complexity and proved to be very robust w.r.t. different input excitations. On the other hand, the computational effort of a usual *non-hierarchical* reduction of this amplifier is significantly larger than the costs of the new *hierarchical* approach. Note that all computations in this example are performed symbolically on the corresponding nonlinear systems of DAEs in time domain.

CHAPTER 2

Foundations

This chapter deals with foundations from mathematics and physics, graph theory, electrical engineering, and systems and control theory, which are needed throughout this thesis. Its sections also serve to provide terms and notions necessary for subsequent sections and chapters.

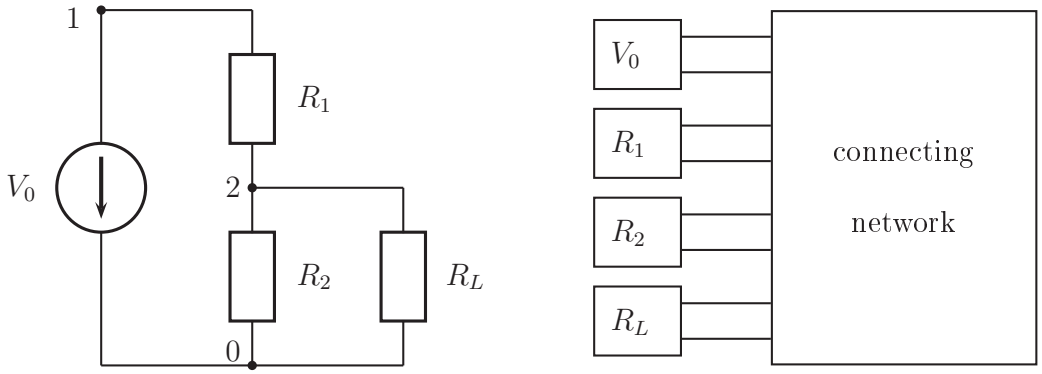
In order to make this work self-contained, we start with an introduction to network analysis including connection graphs, incidence matrices, and the Kirchhoff laws that finally allow the setting up of a system of equations that describes an electrical circuit's behavior mathematically. Next, we review the different categories of systems arising from this, i.e. DAEs and PDAEs, and take a closer look to some application examples from the field of electrical engineering. After a brief presentation of three widely used numerical analysis methods, we install some terms and notions for basic system properties commonly used in systems and control theory. In the last section, the most popular numerical model order reduction techniques based on projections are surveyed in some detail. Moreover, also symbolic reduction methods for both the linear and the nonlinear case in time domain are reviewed.

2.1. Network Analysis

Network analysis in the context of analog circuits in general means the computation of voltages and currents in an analog circuit either as a number value or as a parameterized formula in known variables and magnitudes. In this section, we describe how to obtain a system of equations from a circuit netlist that serves as a model for the mathematical description of the circuit behavior. These systems of equations will then be the topic of Section 2.2.

Abstractly, an electrical circuit is considered as a network of subsystems which are coupled by a certain connecting structure. Each circuit component is modelled by a network subsystem and its behavior is mathematically modelled by a system of equations describing the component's physical effects.

The situation is shown schematically for a voltage-divider circuit in Figure 2.1. In (a), all the used components V_0, R_1, R_2, R_L have two *interfaces*, i.e. "ends", with which they can be interconnected to the remaining circuit components. The corresponding subsystems on the more abstract "network of subsystems"-level in (b) have the same number of interfaces to the connecting structure, of course. Hence, all the subsystems in Figure 2.1 (b) have two connections to the block marked "connecting network". A bipolar junction transistor (BJT), for example, usually has three interfaces called *base*, *collector*, and *emitter*. From now on we will use the term *terminal* rather than *interface* to refer to



(a) Voltage-divider circuit. (b) Network model.

FIGURE 2.1. Example for an electrical circuit as a network of subsystems coupled by a connecting structure.

a component's or a subsystem's interconnections to the remaining network components or the connecting network.

In addition to the equations coming from the mathematical modelling of the circuit components' physics, the connecting structure yields further equations that are necessary to completely describe the circuit behavior mathematically. The following subsection deals with the derivation of such equations from a circuit that is given by its netlist description.

2.1.1. Connection Graphs and Incidence Matrices. As already mentioned, we will consider an analog circuit as a network of subsystems which are coupled by a connecting structure or topology. In particular, circuit components such as resistors, capacitors, diodes, transistors etc., but also sources and amplifiers are considered as subsystems of the circuit.

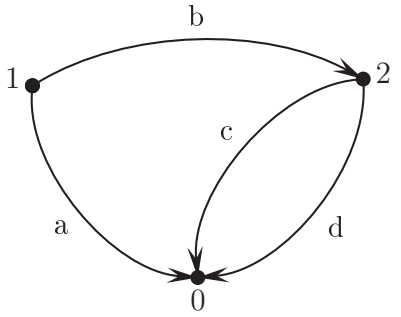
Since the topology of a circuit can be described well by graphs, we start with some basic definitions from graph theory [**ClaHol**].

DEFINITION 2.1. *A (finite) graph $G = (V, E)$ is a tuple of two finite sets, the nonempty* ***set of vertices V*** *and the* ***set of edges E****. An* ***unordered*** *pair of vertices $(u, v) \in V \times V$ is assigned to each edge $e \in E$, denoted by $e = (u, v)$.*

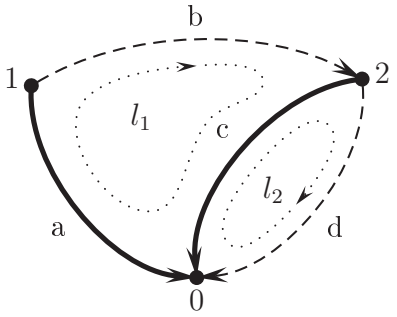
- *A* ***path in G*** *is a finite sequence $w = v_0, e_1, v_1, \ldots, e_k, v_k$ for an appropriate $k \in \mathbb{N}_0$, whose terms are alternately vertices v_i and edges e_j with $e_i = (v_{i-1}, v_i)$. w is also called a* ***path from v_0 to v_k****, where v_0 is the* ***origin of w*** *and v_k its* ***terminus***[1]*.*
- *A* ***loop in G*** *is a path $w = v_0, e_1, v_1, \ldots, e_k, v_k$ in G with $k \geq 1$, where $v_0 = v_k$ and where the edges e_i as well as the vertices v_i, $i = 1, \ldots, k$, are distinct.*
- *The graph G is* ***connected*** *if for each pair of vertices (u, v) in G there exists a path from u to v using edges in E.*[2]

[1]Note that v_0 and v_k need not be distinct.

[2]Equivalently, one can say that there is a path through all edges of E that uses all vertices in V and where some or all of the edges may be used more than once.



(a) Connection graph of the voltage-divider circuit.



(b) A tree, tree branches and link branches, and fundamental loops.

FIGURE 2.2. The voltage-divider circuit graph in (a) with a choice of a tree (bold) in (b), the corresponding link branches (dashed), and the corresponding fundamental loops l_1, l_2 (dotted).

- *G is **directed** if all pairs $(u, v) \in V \times V$ assigned to the edges in E are **ordered**, i.e. the graph's edges are directed*[3]*. Then, for a (directed) edge $e = (u, v)$, u is called the **origin of e** and v is its **terminus**.*

Graphically, the vertices of a graph are represented by dots, while edges are connection lines between two of these dots. In case of a directed graph, the connection lines are replaced by arrows. For example, for an edge $e = (u, v)$ the arrow starts in u and ends in v. Furthermore, from now on we will use the terms *nodes* and *branches* instead of *vertices* and *edges*, and we will consider only connected and directed graphs.

Let the connecting structure of a circuit be given by a finite, directed, and connected graph $G = (V, E)$. Note that this graph contains information only about the circuit *topology*, the information about its components is dumped. Let $n := |V|$ and $b := |E|$ denote the numbers of nodes and branches, respectively. Since G is connected, one then has at least $n - 1$ branches in G, which is the minimum number of branches needed to connect n nodes with each other. Hence, $b \geq n - 1$. To all $e_k \in E$ a branch current i_k and a branch voltage u_k is assigned, where their positive reference direction is given by the direction of e_k. For the circuit in Figure 2.1 (a), a suitable *circuit graph* for the description of its connecting structure is shown in Figure 2.2 (a).

For reasons to appear we will divide the set of branches of a graph into two topological groups, which are *tree branches* and *link branches*. Definitions needed for that are given as follows [**SomHen, SomHenEA**]:

DEFINITION 2.2. *Let the graph $G = (V, E)$ be connected.*

- *A **spanning tree** or **complete tree** or just a **tree** of G is a subgraph $T = (V, E_T)$ of G consisting of the same set of nodes, where $E_T \subseteq E$, $|E_T| = |V| - 1$,*[4] *is a subset of the branches of G connecting all nodes in V. The branches in E_T are called **tree branches**.*

[3] Usually the notation is $e = \{u, v\} = \{v, u\}$ for edges in undirected graphs and $e = (u, v) \neq (v, u)$ for directed graphs.

[4] $|E| \geq |V| - 1$, since G is connected.

- *A **fundamental loop** in a tree $T = (V, E_T)$ of G is a loop which is defined by adding a single branch of $E \setminus E_T$ to T. The branches contained in $E \setminus E_T$ are called **link branches**. Hence, by definition, fundamental loops and link branches are in one-to-one correspondence.*
- *Starting from G and a choice of a tree T the **fundamental loop system** is the set of all $|E| - |V| + 1$ fundamental loops corresponding to the same number of link branches.*

Remark 2.3. *Note that by the above definitions a tree of a graph is connected and does not have any loops*[5]*. Moreover, in general a graph contains many different trees, the number of tree branches, however, is always $|V| - 1$.*

For the voltage-divider circuit in Figure 2.1 (a), a choice of a suitable tree and the corresponding tree and link branches are shown in Figure 2.2 (b). The resulting fundamental loops then are formed by branches $\{b, c, a\}$ and $\{d, c\}$, respectively.

A proof of the following theorem can be found in the appendix, see Section A, Theorem A.1. It will be important later in this paragraph.

Theorem 2.4. *A fundamental loop system of a graph $G = (V, E)$ is linearly independent and spans the space of all its loops.* □

In fact, the loops of a graph G including the zero loop and the union of all its branch-disjoint loops form a vector space. In the proof, the loops are identified by their involved branches. While the addition of "loop vectors" l_1 and l_2 in the graph is given by the symmetric difference of their branches

$$l_1 \Delta l_2 := (l_1 \cup l_2) \setminus (l_1 \cap l_2),$$

the scalar multiplication is defined using scalars from the field $\mathbb{Z}_2 \cong \{0, 1\}$. According to the theorem, the set of fundamental loops of G can be chosen as a basis. In Figure 2.2 (b), for example, the only other loop in the graph is built by branches $\{b, d, a\}$ and simply given by the "sum" $l_1 \Delta l_2$.

The structure of a graph can also be described easily using matrices and linear algebra. For that purpose in what follows recall that we assume the graph G to be finite, connected, and directed.

Definition 2.5. *Let $G = (V, E)$ be a graph with $V = \{v_1, \ldots, v_n\}$ and $E = \{e_1, \ldots, e_b\}$. The **augmented nodal incidence matrix** of G is the $n \times b$-matrix $A_a = (a_{ij})$, where*

$$a_{ij} = \begin{cases} 1, & e_j = (v_i, y), \\ 0, & e_j = (x, y), \\ -1, & e_j = (x, v_i) \end{cases}$$

for some $x, y \in V \setminus \{v_i\}$.

Thus, each row of A_a corresponds to a node of G and each column to a branch. Further, each column of A_a contains exactly one entry 1 and one entry -1, since each branch of

[5] In directed graphs, loops are defined as in Definition 2.1, but without regarding the directions of the involved branches.

G has exactly one origin and one terminus. The rank of A_a is $n-1$, a proof of this is given in the appendix, see Theorem A.2 in Section A. This means that one can omit a (suitable) row of A_a without losing information, since this row is a linear combination of the remaining ones. One mostly leaves out the row which corresponds to the ground node in the circuit for which the graph was originally set up. We call the resulting matrix *reduced nodal incidence matrix* and simply denote it by A:

Definition 2.6. *Let $G = (V, E)$ be a graph with nodes $V = \{v_1, \ldots, v_n\}$ and edges $E = \{e_1, \ldots, e_b\}$. The **reduced nodal incidence matrix** of G is the $(n-1) \times b$-matrix A obtained from the augmented nodal incidence matrix A_a by omitting the row that corresponds to the ground node of the corresponding circuit.*

A similar representation is also possible for the loops of a graph, as the following definition shows.

Definition 2.7. *Let $G = (V, E)$ be a graph with $E = \{e_1, \ldots, e_b\}$ and m loops $l_1, \ldots, l_m$. The **loop incidence matrix** of G is the $m \times b$-matrix $B = (b_{ij})$, where*

$$b_{ij} = \begin{cases} 1, & e_j \text{ positively incident with } l_i, \\ 0, & e_j \text{ not incident with } l_i, \\ -1, & e_j \text{ negatively incident with } l_i. \end{cases}$$

*In this definition, a branch e_j is called **positively (negatively) incident** with the loop l_i if it is directed in the same (opposite) direction as l_i. Otherwise l_i and e_j are **not incident**.*

For example, in Figure 2.2 (b) the loop l_1 is positively incident with branches b and c, negatively incident with branch a, and not incident with branch d.

According to Theorem 2.4, a basis of the loop space of G is given by its $b - n + 1$ fundamental loops. Hence, the number of independent rows in B is $b - n + 1$. From now on we will consider only *reduced loop incidence matrices* that are built by $b - n + 1$ independent loops (the fundamental loops, for the sake of simplicity) and denote them by B as well.

Definition 2.8. *Let $G = (V, E)$ be a graph with $E = \{e_1, \ldots, e_b\}$ and $|V| = n$. Let T be a tree of G and further $l_1, \ldots, l_{b-n+1}$ be the fundamental loops of G corresponding to T. The **reduced loop incidence matrix** of G w.r.t. T is the $(b - n + 1) \times b$-matrix $B = (b_{ij})$, where*

$$b_{ij} = \begin{cases} 1, & e_j \text{ positively incident with } l_i, \\ 0, & e_j \text{ not incident with } l_i, \\ -1, & e_j \text{ negatively incident with } l_i. \end{cases}$$

The next section explains how the incidence matrices of a circuit graph are used to obtain a system of equations describing the circuit's connecting structure.

2.1.2. The Kirchhoff Laws. The $2b$ branch currents and voltages in a circuit cannot be chosen freely, since they must respect the constraints given by the circuit's connecting structure. These constraints are known as the *Kirchhoff laws*. According to

the *Kirchhoff current law (KCL)*, the sum of all branch currents i_k in a node v is zero, whereas the *Kirchhoff voltage law (KVL)* states that the sum of all branch voltages u_k in a loop l is zero. Using Definitions 2.6 and 2.8 from above, these propositions can be reformulated in terms of the reduced incidence matrices A and B:[6]

$$\text{KCL:} \quad A \cdot i = 0, \tag{2.1a}$$

$$\text{KVL:} \quad B \cdot u = 0. \tag{2.1b}$$

Here, i is the vector $i = (i_1, \ldots, i_b)$ of the b branch currents and u is the vector of the b branch voltages $u = (u_1, \ldots, u_b)$. Thus, one obtains a system of equations of size $(b \times 2b)$:

$$\begin{bmatrix} A & 0 \\ 0 & B \end{bmatrix} \cdot \begin{bmatrix} i \\ u \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{2.2}$$

Its solutions satisfy the Kirchhoff laws, i.e. the topological constraints of the connecting structure, and – as the kernel of the left-hand matrix which has full rank b – form a b-dimensional subspace of the space of all branch currents and voltages, the so-called *Kirchhoff space*.

For the determination of the Kirchhoff space it is helpful to compute a basis of that space. One property of the (reduced) incidence matrices A and B allows a direct indication of an appropriate one:

Theorem and Definition 2.9. *The (reduced) nodal incidence matrix* $\boldsymbol{A}$ *and the transpose* $\boldsymbol{B^T}$ *of the (reduced) loop incidence matrix form an* ***exact pair of matrices****, where two real matrices* $M_1 \in \mathbb{R}^{n \times k}$ *and* $M_2 \in \mathbb{R}^{k \times m}$ *are called* ***exact*** *if*

$$M_1 \cdot M_2 = 0 \qquad \textit{and} \qquad \operatorname{rank}(M_1) + \operatorname{rank}(M_2) = k.$$

□

A proof sketch of the theorem can be found in the appendix in Section A. Note that the exactness of B and A^T follows easily, since $BA^T = (AB^T)^T$. Thus, one has

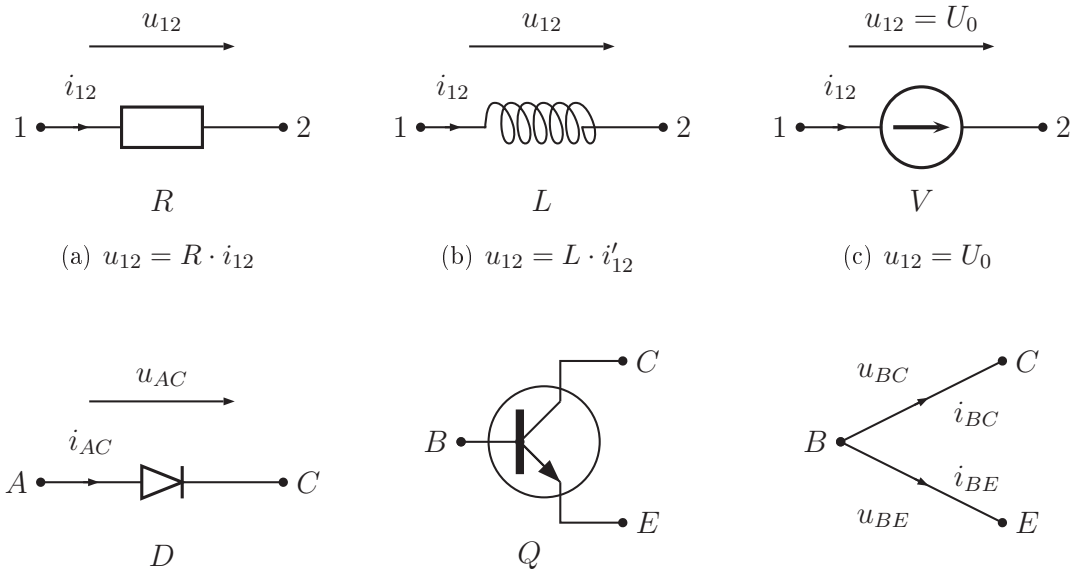
$$\begin{bmatrix} A & 0 \\ 0 & B \end{bmatrix} \cdot \begin{bmatrix} B^T & 0 \\ 0 & A^T \end{bmatrix} = 0 \tag{2.3}$$

and the columns of the right-hand matrix form a basis of the Kirchhoff space. Hence, the solutions (i, u) of the Kirchhoff laws in (2.1) are linear combinations of the columns of the matrices B^T and A^T, respectively:

$$i = B^T j \qquad \text{and} \qquad u = A^T v. \tag{2.4}$$

Note that j and v are unique. The coordinates of the vectors j and v are the coefficients of these linear combinations. For dimensioning reasons the components of j have to be currents, while the components of v are voltages. Since j is premultiplied by B^T and the columns of this matrix correspond to (fundamental) loops in the circuit graph, the components j_k of j are called *loop currents*. Similarly, since the columns of A^T correspond to nodes of the circuit graph, the components v_k of v are called node voltages or rather *node potentials*. Thus, an independent loop current is assigned to each loop in the circuit

[6] Recall that the rows in A correspond to the nodes in the circuit graph, while the rows in B correspond to its (fundamental) loops.



(a) $u_{12} = R \cdot i_{12}$

(b) $u_{12} = L \cdot i'_{12}$

(c) $u_{12} = U_0$

(d) $i_{AC} = I_s \cdot \left(e^{\frac{1}{V_T}\, u_{AC}} - 1\right)$

(e) A bipolar junction transistor in npn-configuration.

(f) Branch magnitudes of an npn-transistor (BJT).

FIGURE 2.3. Branch magnitudes of various circuit components.

graph. Then a branch current, as indicated by (2.4), can be computed by superposing all "its" loop currents, i.e. all the currents of those loops that it is part of. Similarly, a branch voltage simply is the difference between the two node potentials of its origin and terminus.

2.1.3. Current-Voltage Relations of Circuit Components. The so far presented equations only model the *connecting structure* of the analog circuit. The relations between the branch currents and voltages of its components have not yet been treated. They lead to further restrictions of the Kirchhoff space of solutions to the circuit's topological constraints.

For example, for a simple resistor R with two terminals marked 1 and 2 one has

$$f_R(i_{12}, u_{12}) = u_{12} - R \cdot i_{12} = 0, \tag{2.5}$$

where u_{12} and i_{12} denote the branch voltage above and the branch current through the resistor directed from terminal 1 to terminal 2 (cf. Figure 2.3 (a)). Similarly, for a linear inductor L, an independent voltage source V, and a nonlinear diode D (with terminals A and C, **a**node and **c**athode) one has

$$0 = f_L(i'_{12}, u_{12}) \quad = u_{12} - L \cdot i'_{12}, \tag{2.6a}$$

$$0 = f_V(i_{12}, u_{12}) \quad = 0 \cdot i_{12} + 1 \cdot u_{12} - U_0, \tag{2.6b}$$

$$0 = f_D(i_{AC}, u_{AC}) = i_{AC} - I_s \cdot \left(e^{\frac{1}{V_T}\, u_{AC}} - 1\right). \tag{2.6c}$$

Within these equations, i'_{12} denotes the derivative of i_{12} with respect to time. Thus, (2.6a) already gives a hint that we will also have to deal with *differential equations* for the mathematical modelling of analog circuits.

A circuit component is called ***linear*** if it acts as a *linear operator* that maps branch currents and voltages to other branch currents and voltages. Usually, resistors, inductors, and capacitors are examples for linear circuit components. The corresponding linear operators are given by $R\cdot$, $L \cdot \frac{d}{dt}$, and $C \cdot \frac{d}{dt}$, respectively, and map the component's branch current to its own branch voltage (R, L) or vice versa (C). Other examples for linear components are *controlled sources* which generate branch currents or voltages that are steered by other branch currents or voltages. *Independent sources* such as V in Figure 2.3 (c) that are given by *inhomogeneous* linear equations in the involved branch currents and voltages are also linear components.

Diodes and transistors, for example, are *nonlinear* components, since the relation between their involved branch currents and voltages cannot be described by linear operators or (inhomogeneous) linear equations. Equation (2.6c) for the diode with terminals A (anode) and C (cathode) is the so-called *Shockley equation* [**Sho**], which models the diode's behavior in lower frequencies. While $I_s = -\lim_{u_{AC}\to-\infty} i_{AC}(u_{AC}) \approx 10^{-9\ldots-15}\,\mathrm{A}$ is a (local) instance-dependent parameter[7] called the *reverse leakage current* or *saturation current*, the *thermal voltage* V_T is a global parameter with $V_T \approx 0.026\,\mathrm{V}$ at room temperature. There is a variety of models available for the diode's behavior with different levels of complexity and taking into account different kinds of physical effects of the *real* (physical) diode.

The bipolar junction transistor (BJT) Q with three terminals B (base), C (collector), and E (emitter), in Figure 2.3 (e) in npn-configuration, is a more complicated nonlinear circuit component. As for the diode, there is a variety of distinct behavioral models for a BJT with different levels of complexity. A very basic one is given by the *Ebers-Moll equations* [**GraMey, VlaSin**], it is valid only for low frequencies, since it neglects certain physical effects of the semiconductor material in a real BJT. Using the branch magnitudes from Figure 2.3 (f)[8], it is given by

$$0 = \begin{pmatrix} i_{BC} + I_s\big(e^{\frac{1}{V_T}\,u_{BE}} - 1\big) - I_s\frac{1+\beta_b}{\beta_b}\big(e^{\frac{1}{V_T}\,u_{BC}} - 1\big) \\ i_{BE} + I_s\big(e^{\frac{1}{V_T}\,u_{BC}} - 1\big) - I_s\frac{1+\beta_f}{\beta_f}\big(e^{\frac{1}{V_T}\,u_{BE}} - 1\big) \end{pmatrix} \tag{2.7}$$

for a BJT in npn-configuration. An appropriate model for a pnp-configuration is obtained by changing appropriate signs in equation (2.7). The parameters β_b and β_f denote the *backward* and *forward amplification*, respectively. I_s again is the *reverse leakage current* and V_T the *thermal voltage*. Normally, β_b takes values between 0.5 and 20, typical values for β_f are in a range from 50 to 500. The magnitude of I_s is approximately $10^{-14}\,\mathrm{A}$, while V_T at a temperature $T = 300.15\,\mathrm{K}$ is approximately $26\,\mathrm{mV}$.

[7]Note that this limit consideration is idealized. In a real physical device, there is a *reverse breakdown* if the applied voltage in *reverse direction*, the so-called *reverse bias*, is too large, i.e. beyond the *peak inverse voltage (PIV)*. This causes a large increase of the current in reverse direction and usually the semiconductor device is damaged permanently.

[8]Note that these are sufficient to compute the remaining branch magnitudes using the Kirchhoff laws. For instance, one has $u_{CE} = u_{BE} - u_{BC}$.

If, in addition, higher frequencies are to be taken into account, more precise models such as the *Gummel-Poon equations* [**GraMey**] have to be used.

Finally, the current-voltage relations of all the circuit's components are collected and incorporated in its mathematical model. Those branch voltages and currents which satisfy the set of current-voltage relations span an affine space, the so-called *Ohm space*. The *state space* obtained by intersecting the Kirchhoff and Ohm spaces consists of all those branch voltages and currents that satisfy all the constraints on the circuit simultaneously. For linear, resistive, non-dynamical networks or non-degenerate ones, the state space dimension is 0.

2.1.4. Sparse Tableau Analysis. [**HacBraGus**] The most obvious way to derive describing equations for an analog circuit is collecting all the equations coming from the current-voltage relations of the circuit's components and those due to the Kirchhoff laws, i.e. the circuit's topological constraints. The resulting equations are called *sparse tableau equations*, they are not minimal w.r.t. their number and the number of variables, but can be automatically set up. Moreover, the sparse tableau equations are suitable for both numerical and symbolic analysis of the given network.

If one deals with a *linear, time-invariant (LTI) network* (see Definition 2.15), the current-voltage relations of the b branches may be written in a form using matrices $P, Q \in \mathbb{R}^{b\times b}$,

$$P \cdot i + Q \cdot u = s, \tag{2.8}$$

where $s \in \mathbb{R}^b$ is a vector of independent source voltages and currents. In that case, the state space can be computed by solving the $2b \times 2b$-system

$$\begin{bmatrix} A & 0 \\ 0 & B \\ P & Q \end{bmatrix} \cdot \begin{bmatrix} i \\ u \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ s \end{bmatrix}. \tag{2.9}$$

This motivates the name *sparse tableau analysis (STA)* for this method, since (2.9) is a sparse tableau or sparse matrix[9] equation.

In general, however, one does not have only linear components in an electrical circuit. Further, the system parameters' independence of time usually is not given due to certain physical effects. So the current-voltage relations of a circuit usually are a collection of nonlinear vector-valued functions

$$f_j(i, i', u, u', x, x', t) = 0 \tag{2.10}$$

involving derivatives of branch currents and voltages that describe these relations for the single circuit components[10]. Examples of linear components such as resistors, inductors, and voltage sources, but also for a nonlinear diode and a BJT of lower complexity have yet been considered in the previous subsections, see equations (2.5) – (2.7) and Figure 2.3. The variables x in equation (2.10) are internal or auxiliary variables which do not

[9] A *sparse* matrix is a matrix with only few of its entries being non-zero. A common definition is: an $n \times n$ matrix is sparse if the number of its non-zero entries is $\mathcal{O}(n)$.

[10] Note that a first order representation like the above one is always possible, e.g. by introducing additional variables.

correspond to branch magnitudes of the circuit, but sometimes are necessary or useful to model the behavior of certain components.

As already mentioned, the most obvious method for generating a system of equations that models the behavior of the circuit is the usage of all branch currents and voltages as variables of the system. The corresponding equations are set up using the Kirchhoff laws KCL and KVL and the current-voltage relations

$$f(i, i', u, u', x, x', t) = \begin{pmatrix} f_{c_1}(i, i', u, u', x, x', t) \\ \vdots \\ f_{c_k}(i, i', u, u', x, x', t) \end{pmatrix} \tag{2.11}$$

of the k circuit components $c_1, \ldots, c_k$. Thus, one obtains a system of $2b+m$ equations in $2b+m$ variables, the ***sparse tableau equations***, where m is the number of all internal variables used in the k circuit components:

$$\begin{aligned} A \cdot i &= 0, \\ B \cdot u &= 0, \\ f(i, i', u, u', x, x', t) &= 0. \end{aligned} \tag{2.12}$$

Obvious advantages of this method for setting up the equations are its simplicity and the fact that there are no limitations on the circuit components; it is applicable to any circuit independent of its elements. However, the system of equations obtained by this method usually is very large even for smaller circuits, since at least all the $2b$ branch magnitudes are used as variables in the system, thus yielding $\geq 2b$ equations. Although these are sparse systems, they are costly to solve because of their size. The following section provides an alternative that uses the n node potentials of the circuit and only a few of its branch currents as variables. Consequently, one has to cope with much smaller systems.

2.1.5. Modified Nodal Analysis. [**HoRueBre**] The most serious disadvantage of the sparse tableau formulation is its large size. In order to circumvent this problem and to obtain more compact systems, the *modified nodal analysis (MNA)* uses only the *node potentials* introduced in equation (2.4) together with certain branch currents as variables of the system. This guarantees that *all* branch magnitudes of the system can be computed from this set of variables. Since the MNA is easy to implement and applicable to any analog circuit without restrictions of the used components, this method is most widely used in simulation tools for analog circuits.

As the name already implies, the MNA is a modification of the standard nodal analysis (cf., e.g., [**SomHen, SomHenEA**]). The modification rests on the removal of the restrictions on the circuit components in the latter analysis method, where components that do not have an *admittance formulation* are not permitted.

Definition 2.10. *If for a circuit component c the current-voltage relation*

$$f_c(i, i', u, u', x, x', t) = 0$$

can be expressed as

$$i_c = g_c(u, u', x, x', t), \tag{2.13}$$

where i_c are the branch currents of c, the latter formulation is called the ***admittance formulation*** *of c.*

Thus, for circuit components offering an admittance formulation, the branch currents can be computed from the branch voltages. Since the branch voltages themselves are computable from the node potentials by (2.4), it is intuitively clear that there is a more compact describing set of equations available.

Unfortunately, not all circuit components have an admittance formulation. For example, the current-voltage relations of an inductance L or an independent voltage source V are described by $f_L(i, i', u, u', x, x', t) = u_L(t) - L \cdot i'_L(t)$ and $f_V(i, i', u, u', x, x', t) = u_V(t) - U_0(t)$, respectively, and do not have an admittance formulation. Hence, by restricting to circuit components that have an admittance formulation, the entire circuit can be described mathematically by equations that use only the node potentials as variables. The corresponding method is called *(standard) nodal analysis*, see Section 2.1.6.

For a general analog circuit, however, the set of circuit components is divided into two parts. While the components in the first part *do* have an admittance formulation, those in the second part do *not*. According to this partitioning, the vector of branch currents i is also divided, w.l.o.g. $i = (i_1, i_2)$, such that the circuit's current-voltage relations are given by (cf. [**Wic04**])

$$f(i, i', u, u', x, x', t) = \begin{pmatrix} i_1 + g(i_2, i'_2, u, u', x, x', t) \\ h(i_2, i'_2, u, u', x, x', t) \end{pmatrix} = 0. \tag{2.14}$$

Let further the nodal incidence matrix A be divided according to $i = (i_1, i_2)$. Then due to the Kirchhoff current law (2.1a) one has

$$0 = A \cdot i = A_1 \cdot i_1 + A_2 \cdot i_2. \tag{2.15}$$

From this, one obtains the ***modified nodal equations*** by premultiplying the first block in (2.14) with A_1 to eliminate i_1 and by using (2.4) to replace the branch voltages u by the node potentials v:

$$\begin{aligned} A_2 \cdot i_2 &= A_1 \cdot g(i_2, i'_2, A^T v, A^T v', x, x', t) \\ 0 &= h(i_2, i'_2, A^T v, A^T v', x, x', t). \end{aligned} \tag{2.16}$$

Let m_1 be the number of auxiliary variables x and m_2 be the number of entries in i_2, i.e. $i_2 \in \mathbb{R}^{m_2}$. Thus, system (2.16) consists of $n-1+m_1+m_2$ equations in $n-1+m_1+m_2$ variables, namely, $n-1$ node potentials v, m_2 branch currents i_2, and m_1 auxiliary variables x. Note that these equations implicitly contain both the Kirchhoff current and voltage laws.

Remark 2.11. *Converting part of the current-voltage relations into admittance formulation may lead to very complex expressions in g occuring multiple times in the system of equations because of the multiplication by A_1.*

Nevertheless, this method has no principal limitations on the circuit's components and leads to compact, but usually dense systems of equations. As a further advantage, no

loop incidence matrix has to be computed. For these reasons, the modified nodal analysis is used most often to set up the describing system of equations of a circuit.

2.1.6. Further Analysis Methods. There are several other methods to set up appropriate systems of equations as mathematical models for the circuit's behavior of whom we would like to mention the *(standard) nodal analysis (NA)* and the *(standard) loop analysis (LA)*. But since – in contrast to the above presented approaches STA and MNA – there are certain restrictions on the circuit components, they are used rarely and, therefore, described rather briefly here.

The *nodal analysis*, from which the modified nodal analysis is derived, aims at obtaining a more compact set of equations by using the $n-1$ node potentials instead of the $2b$ branch voltages and currents. Further, the Kirchhoff laws are used *implicitly* instead of adding them explicitly to the current-voltage relations.

As an example, consider a linear time-invariant (LTI) system (cf. Definition 2.15) with current-voltage relations given by

$$P \cdot i + Q \cdot u = s, \tag{2.17}$$

where $P, Q \in \mathbb{R}^{b\times b}$ and $s \in \mathbb{R}^b$ is a vector of independent source voltages and currents. If each of the circuit's components has an *admittance formulation*, then P is invertible, hence

$$I_b \cdot i + P^{-1}Q \cdot u = P^{-1} \cdot s. \tag{2.18}$$

For dimension reasons, $P^{-1}Q$ contains only conductances, while $P^{-1} \cdot s$ only consists of currents. Premultiplication by the nodal incidence matrix A to eliminate the branch currents i and substitution of the b branch voltages u by the node potentials v using (2.4) yields

$$AP^{-1}QA^T \cdot v = AP^{-1} \cdot s. \tag{2.19}$$

Thus, one has $n-1$ equations in $n-1$ variables $v = (v_1, \ldots, v_{n-1})^T$ which implicitly contain the two Kirchhoff laws. The described approach is the ***(standard) nodal analysis (NA)***, and the matrix $AP^{-1}QA^T \in \mathbb{R}^{n\times n}$ on the left is often referred to as the *nodal admittance matrix* of the system. It looks complicated, but can easily be set up using fill-in patterns for respective circuit components.

Analogous to the above approach, one can also use the $b-n+1$ loop currents instead of the $2b$ branch magnitudes as independent variables in the equations. As an example, consider again an LTI system with current-voltage relations given by (2.17). This time, suppose that Q is invertible, i.e. the current-voltage relations of all circuit components can be solved for the respective branch voltages u. This yields

$$Q^{-1}P \cdot i + I_b \cdot u = Q^{-1} \cdot s, \tag{2.20}$$

where $Q^{-1}P$ solely contains resistances and $Q^{-1} \cdot s$ only contains voltages. Premultiplication by the loop incidence matrix B to eliminate u and using (2.4) to substitute the b branch currents i by the $b-n+1$ loop currents j leads to

$$BQ^{-1}PB^T \cdot j = BQ^{-1} \cdot s. \tag{2.21}$$

This system consists of $b-n+1$ equations in $b-n+1$ variables $j = (j_1, \ldots, j_{b-n+1})^T$ and implicitly contains the two Kirchhoff laws. The described method for its derivation is called the ***(standard) loop analysis (LA)***. The *loop impedance matrix* $BQ^{-1}PB^T \in \mathbb{R}^{b-n+1 \times b-n+1}$ can also be set up using fill-in patterns for the corresponding circuit components, but these patterns are more complicated than those for the nodal analysis and usually lead to rather dense coefficient matrices. Hence, this method is used only rarely for computer-aided circuit analysis.

2.2. Systems of Equations

The dynamical behavior of real physical systems such as electrical circuits considered in this thesis usually is modeled by the use of differential equations. If these equations can be written in the form

$$\dot{x} = f(x, t), \tag{2.22}$$

the rich and well known theory of *ordinary differential equations (ODE)* is available with a variety of good numerical solvability concepts.

However, physical systems such as electrical circuits usually are subject to certain constraints, thus yielding mathematical equations that are not of the ODE type. This section provides the necessary theory about systems of equations that arise in electrical network analysis, namely, systems of *differential-algebraic equations (DAE)* and systems of *partial differential-algebraic equations (PDAE)*. For DAEs, we mainly follow [**KunMeh**] and [**Wic04**]. Since the research area of PDAEs is quite young and we could not find any textbook concerning this topic, we collected some papers [**AliBarGueTis, BodTis, Gue00, Gue01, LamMarTis, MarBar, Reis05, Reis06**] on which the theory of PDAEs provided here is based. For further information, we refer to these and the references therein.

The most widely used numerical technique for the numerical solving of systems of PDEs is the *finite element method (FEM)*, further the *finite difference method (FDM)*, and the *finite volume method (FVM)*. All these methods basically rely on a semidiscretization w.r.t. space yielding systems of DAEs that depend on time only. Therefore, in the subsequent sections and chapters we are focussing on DAEs rather than PDEs or PDAEs.

The section concludes with some examples for systems of DAEs and PDAEs generated from analog circuits. In the latter case, mainly the PDE part of two circuit devices, a transmission line and a diode, is discussed, while in the first case describing equations for the voltage divider from Figure 2.1 (a) and a slightly varied version are set up.

2.2.1. Systems of DAEs. As already mentioned, real physical systems usually are in some way constrained by, e.g., conservation laws such as the Kirchhoff laws in electrical networks, thus restricting their solution to a certain subdomain. Therefore, the describing mathematical systems also contain algebraic equations to model these – explicit and also *implicit* – limitations (see also Example 2.18 at the end of this subsection). The corresponding equations are systems of *differential-algebraic equations*, they are special types of (implicit) differential equations and sometimes also called *algebro-differential* or

singular systems [**KunMeh**]. The most general form of a system of DAEs is given by the following definition.

Definition 2.12. *Let $F : \mathbb{D}_x \times \mathbb{D}_{\dot{x}} \times \mathcal{T} \to \mathbb{C}^m$ be differentiable with $\mathcal{T} \subseteq \mathbb{R}$ a time interval and $\mathbb{D}_x, \mathbb{D}_{\dot{x}} \subseteq \mathbb{C}^n$ open. If the Jacobian $D_{\dot{x}}F$ is singular on $\mathbb{D}_x \times \mathbb{D}_{\dot{x}} \times \mathcal{T}$, the system of implicit differential equations*

$$F(x, \dot{x}, t) = 0 \tag{2.23}$$

is called a system of ***differential-algebraic equations (DAE).***

In this definition, one usually has $\mathcal{T} = \mathbb{R}_{\geq 0}, \mathbb{R}$. It further motivates the term *singular system*, since for a system of ODEs F the Jacobian $D_{\dot{x}}F$ is (locally) *regular*. According to this definition, a system F of purely algebraic equations also is a system of DAEs, since then $D_{\dot{x}}F \equiv 0$. Due to the singularity of the Jacobian for a system of DAEs, certain effects are present that are not known for ODEs (cf. Example 2.18 and Section 2.2.2).

Note that the meaning of $\dot{x}$ is ambiguous in (2.23), since on the one hand it denotes an independent variable in F, and on the other hand it denotes the time derivative of x considered as a function $x : \mathcal{T} \to \mathbb{C}^n$. The reason for that is our ambition to find a differentiable function $x : \mathcal{T} \to \mathbb{C}^n$ that solves $F(x(t), \dot{x}(t), t) = 0$ for all $t \in \mathcal{T}$.

Definition 2.13. *Let $F : \mathbb{D}_x \times \mathbb{D}_{\dot{x}} \times \mathcal{T} \to \mathbb{C}^m$ be a system of DAEs in the sense of Definition 2.12. Let further $x : \mathcal{T} \to \mathbb{C}^n$ be differentiable with $(x(t), \dot{x}(t)) \in D_x \times D_{\dot{x}}$ for all $t \in \mathcal{T}$. If*

$$F(x(t), \dot{x}(t), t) = 0 \quad \text{for all } t \in \mathcal{T},$$

then x is called a ***solution of*** *F.*

Note that due to the *algebraic* constraints not all of the components of x have to be differentiable.

Similar to systems of ODEs, one also considers ***initial value problems*** for DAEs. But in contrast to the ODE case, the initial value x_0 cannot be chosen freely because of the algebraic constraints. Hence, the term *consistent initial value* has been formed.

Definition 2.14. *Let F be a system of DAEs in the sense of Definition 2.12 and $x : \mathcal{T} \to \mathbb{C}^n$ be a solution of F. Let further $t_0 \in \mathcal{T}$, and $x_0 \in \mathbb{R}^n$. If*

$$\begin{aligned} F(x(t), \dot{x}(t), t) &= 0 \quad \text{for } t_0 \leq t \in \mathcal{T}, \\ x(t_0) &= x_0, \end{aligned} \tag{2.24}$$

then x is called a ***solution of the initial value problem*** (2.24) *and x_0 is called* ***initial value*** *of the initial value problem. More precisely, it is called* ***consistent initial value*** *if $x_0 \in \mathbb{C}^n$ is chosen such that the initial value problem* (2.24) *has a solution.*

Note that often for analog circuits not the value of x is prescribed at $t = t_0$ in an initial value problem, but rather the value of $\dot{x}$.

Definition 2.12 holds for general *rectangular* systems with arbitrary n, m. In case of electrical circuits, one actually deals with quadratic systems ($n = m$), where the numbers

of variables and equations coincide. This is due to the methods from Section 2.1 for setting up the equations which yield only this kind of systems. Hence, from now on we mostly consider quadratic systems.

The by far best studied and at the same time most basic substructure of systems of DAEs are linear ones and, in particular, the *linear time-invariant (LTI)* cases:

Definition 2.15. *Let F be a system of DAEs in the sense of Definition 2.12. If there exist continuous $f : \mathcal{T} \to \mathbb{C}^n$ and $E, A : \mathcal{T} \to \mathbb{C}^{n\times n}$ such that F can be written in the form*

$$E(t)\dot{x} = A(t)x + f(t) \quad \text{for } t \in \mathcal{T}, \tag{2.25}$$

then it is called a ***linear time-variant (LTV)*** *system of DAEs. If E, A are constant on $\mathcal{T}$, it is called a* ***linear time-invariant (LTI)*** *system of DAEs.*

LTI systems are the simplest kind of DAEs. Nevertheless, many of the special features of the general nonlinear case are observable by considering linear time-*variant* systems of DAEs [**BreCamPet, Wic04**].

Linear DAEs arise, for example, in so-called *RCL circuits* that consist solely of resistors, capacitors, or inductors. Especially LTI systems, which may be treated using purely algebraic methods, are well studied and understood for more than one century. Their basic theory has been established by the fundamental works of Weierstraß [**Wei58, Wei67**] and Kronecker [**Kro**] on matrix pencils. However, it took until the pioneering work of Gear [**Gea**] to show the great importance of the theory of DAEs for modelling dynamical systems. Before that, implicit systems as in Definition 2.12 usually were transformed into ordinary differential equations (2.22) either by differentiation of the algebraic constraints or by solving these constraints analytically in order to eliminate variables in the dynamical part of the system. But since these approaches are limited to systems of small size or subject to problems such as numerical solutions drifting away from the constraint manifold after only a few iteration steps, it is in general preferable to (develop methods that) directly operate on the given system of DAEs.

Further systems with a special structure are the *semi-explicit* ones whose variables can be divided into a static and an explicit dynamical part:

Definition 2.16. *Let $\mathcal{T} \subseteq \mathbb{R}$ be a time interval and for $D_x \subseteq \mathbb{C}^{n_1}$ and $D_y, D_{\dot{y}} \subseteq \mathbb{C}^{n_2}$ open, $n = n_1 + n_2$, let $F : D_x \times D_y \times D_{\dot{y}} \times \mathcal{T} \to \mathbb{C}^n$ be a (quadratic) system of DAEs such that it can be divided into a differentiable $F_1 : D_x \times D_y \times D_{\dot{y}} \times \mathcal{T} \to \mathbb{C}^{n_2}$ and a continuous $F_2 : D_x \times D_y \times \mathcal{T} \to \mathbb{C}^{n_1}$ satisfying*

$$F(x, y, \dot{y}, t) = \begin{pmatrix} F_1(x, y, \dot{y}, t) \\ F_2(x, y, t) \end{pmatrix} = 0. \tag{2.26}$$

If $D_{\dot{y}}F_1$ is regular, then F is called a ***semi-explicit system****.*

Thus, semi-explicit systems can be transformed into the form

$$\begin{aligned} \dot{y} &= \widetilde{F}_1(x, y, t) \\ 0 &= F_2(x, y, t). \end{aligned}$$

Such systems are of some significance particularly for theoretical investigations because it is already known which of the variables occur with explicit derivatives.

For the purpose of circuit analysis, the so-called *quasi-linear systems* are of maybe the highest importance, as this kind of systems usually is obtained from the circuit netlist by applying the methods from Section 2.1.

Definition 2.17. *Let $\mathcal{T} \subseteq \mathbb{R}$ be a time interval, $D_x, D_{\dot{x}} \subseteq \mathbb{C}^n$, and $f : D_x \times \mathcal{T} \to \mathbb{C}^n$ continuous. Furthermore, let $A : D_x \times \mathcal{T} \to \mathbb{C}^{n\times n}$ be continuous and $A(x,t)$ singular on $D_x \times \mathcal{T}$. The system of DAEs F defined by*

$$F(x, \dot{x}, t) = A(x,t)\dot{x} + f(x,t) = 0 \quad \text{for } t \in \mathcal{T} \tag{2.27}$$

is called a ***quasi-linear system****.*

Specially structured systems that will be used in later sections of this thesis are ***control problems*** of the form

$$\begin{aligned} E\dot{x}(t) &= F(x(t), u(t)), \\ y(t) &= G(x(t), u(t)), \end{aligned} \tag{2.28}$$

where F and G are vector-valued maps of appropriate dimensions. These systems are also often referred to by the term ***state space systems***. While the variables u are considered as ***inputs*** to the system, the variables y and x are its ***outputs*** and ***states***, respectively. The linear case with constant coefficients, often referred to by the term ***(linear) descriptor system*** or ***linear state space system***, is well understood and can be written in the form

$$\begin{aligned} E\dot{x}(t) &= Ax(t) + Bu(t) + f(t), \\ y(t) &= Cx(t) + Du(t) + g(t) \end{aligned} \tag{2.29}$$

with $E, A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $D \in \mathbb{R}^{p\times m}$, $f : \mathcal{T} \to \mathbb{R}^n$ continuous, and $g : \mathcal{T} \to \mathbb{R}^p$ continuous, where $x : \mathcal{T} \to \mathbb{R}^n$ represents the *states*, $u : \mathcal{T} \to \mathbb{R}^m$ the *inputs* or *control*, and $y : \mathcal{T} \to \mathbb{R}^p$ the *outputs* of the system. It describes a so-called ***multi-input/multi-output (MIMO) system*** with m inputs and p outputs. If the system matrices A, B, C, D are constant, i.e. independent of t, such systems are also called ***linear time-invariant (LTI) state space systems*** (see also Definition 2.15).

We conclude this subsection with an example that shows that DAEs feature certain phenomena which do not occur with pure differential or algebraic equations.

Example 2.18. [Wic04]
Consider the system of DAEs given by

$$0 = \dot{x}_1 + x_1, \tag{2.30a}$$

$$0 = x_2\dot{x}_2 - x_3, \tag{2.30b}$$

$$0 = x_1^2 + x_2^2 - 1, \tag{2.30c}$$

where $x_1, x_3 \in \mathbb{R}$ and $x_2 > 0$ on a time interval $\mathcal{T} \subseteq \mathbb{R}$. While a system of ODEs $\dot{x} = f(x,t)$ with $x \in \mathbb{R}^n$ can be considered as a vector field V living on a manifold M (usually $\mathbb{R}^n$) and being tangential to it in any of its points, this situation is more complicated for general systems of DAEs $F(x, \dot{x}, t) = 0$: M is restricted to some subdomain by

the algebraic constraints, and since not all of the derivatives $\dot{x}_i$ are explicitly given, one has to cope with a *family* of vector fields. A *solution* of the system, however, certainly lies in the restricted submanifold and, moreover, is an integral curve for a single vector field in the family.

In (2.30), M obviously is restricted by (2.30c) yielding a new manifold M_1, while V is described by the family $V_a(x_1, x_2, x_3) = (-x_1, x_3/x_2, a)$ with an arbitrary parameter a. Due to V_a having to be tangential to M_1, i.e. $\operatorname{grad}(M_1) \perp V_a$, independently of the parameter a one obtains

$$0 = -x_1^2 + x_3. \tag{2.31}$$

Thus, system (2.30) contains additional ***hidden*** *implicit constraints* further restricting M_1 which is now narrowed down to M_2 given by equations (2.30c), (2.31).

But also V_a becomes limited, since those family members that are not tangential to the *new* manifold M_2 can be deleted. So $\operatorname{grad}(x_1^2 - x_3) \perp V_a$ further yields

$$0 = -2x_1^2 - a \tag{2.32}$$

which restricts the family of vector fields V_a to a single one given by

$$V(x_1, x_2, x_3) = (-x_1, x_3/x_2, -2x_1^2).$$

Finally, a parameterization yields a *consistent initial value*

$$x_1(t_0) = \sqrt{c}, \quad x_2(t_0) = \sqrt{1-c}, \quad x_3(t_0) = c$$

for a parameter $0 \leq c < 1$. □

In the above example, the challenges arising with systems of DAEs are observable. While on the one hand a general system of DAEs underlies *hidden constraints* which are not explicitly given, on the other hand the subsequent generation of new algebraic or differential equations may lead to derivatives of input magnitudes, thus making particularly the numerical solving of such systems a badly conditioned problem.

For this reason, the *index* of a system of DAEs was introduced. Although there is a variety of different index concepts available, they have a main purpose in common, namely, the classification of difficulties that one has to deal with while working with a given system of DAEs.

2.2.2. Index Concepts. The introduction of an *index* is motivated by the classification of the difficulties that one has to cope with while solving a system of differential-algebraic equations. In general, the numerical solving of systems with an index higher than 1 is an ill-posed problem.

A multitude of different index concepts have been invented for general systems of DAEs. Generally speaking, an *index* of a system of DAEs measures the integer-valued *distance to a system of ODEs*. Currently the so-called *differentiation* and *perturbation indices* are the most widely used concepts in literature. The differentiation index is the minimum number of differentiations with respect to t that have to be performed on

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

in order to solve for $\dot{x}$ as a continuous function in t and x. The motivation for this is historically based on the approach for solving implicit systems by using transformations to systems of ODEs, for which the theory is well established.

Since the theory of purely algebraic equations also is well investigated, the *strangeness index* measures the "distance" of a system of DAEs to a decoupled system of ODEs and purely algebraic equations. Thus, the index of a system of ODEs as well as the one of a system of purely algebraic equations are equal which is clearly not the case for the differentiation index. *Analog Insydes* uses the concepts of the *strangeness* and the *tractability index* to monitor the index of a symbolic system of DAEs during its reduction process.

However, at this point we abstain from describing further index concepts. Instead, we refer to [**GueFel, HalWic03, KunMeh, Wic04**] and references given therein, where detailed information and overviews can be found. There further exist some index concepts for systems of *partial differential-algebraic equations* (PDAE), see [**CamMar, MarBar**] for instance.

2.2.3. Systems of PDAEs. Similar to the concept of differential-algebraic equations, systems of *partial differential equations (PDE)* that are constrained by certain conditions contain additional algebraic equations to model these restrictions. The systems arising from that are called *partial differential-algebraic equations* or *PDAEs*.

In the context of electrical networks, systems of PDAEs came up in a natural way. During the last years, the design of integrated circuits (IC) and particularly the modelling of nonlinear semiconductor elements became more and more cumbersome. Increasing integration densities combined with decreasing spatial scales have led to nanoelectronic structures, i.e. feature sizes of less than 100 nm. Physical effects such as thermal interaction among the circuit components, electromagnetic radiation, substrate noise, or crosstalk cannot be neglected anymore. Ideas to incorporate these effects by using a higher number of modelling parameters showed that this number seems to increase exponentially. A more promising approach for taking them into account is the use of model descriptions based on PDEs, particularly for the precise modelling of semiconductor components. This obviously leads to systems combining DAE and PDE parts, a special form of PDAEs as will be shown in this subsection (cf. Proposition 2.21).

Since *partial differential-algebraic equations* are a special type of *implicit partial differential equations*, we firstly provide some theory about PDEs. A very general definition of a PDE is as follows [**Jos**]:

> *A* ***partial differential equation*** *is an equation for an unknown function $z : \Omega \to \mathbb{R}$, $\Omega \subseteq \mathbb{R}^d$ open, $d \geq 2$, which contains partial derivatives of z. Further, each occurrence of z and its partial derivatives is evaluated at the same point $x \in \Omega$.*

In general, Ω is an open subset of a differentiable manifold of dimension $d \geq 2$. A more precise definition is as follows:

Definition 2.19. *A system of* ***implicit partial differential equations*** *or* ***implicit PDE*** *is defined by*

$$F(\omega, z(\omega), Dz(\omega), D^2z(\omega), \dots, D^kz(\omega), \dots) = 0, \tag{2.33}$$

where for $\Omega \subseteq \mathbb{R}^d$ open, $d \geq 2$, $\omega \in \Omega$ are independent variables, $z : \Omega \to \mathbb{R}^N$ are sufficiently differentiable unknown functions, D^kz are the partial derivatives of z of degree k, and F is an arbitrary function mapping to $\mathbb{R}^m$.

An example for a physical process where time derivatives are not sufficient for an adequate modelling is the propagation of a wave that is caused by a drop of water falling on a water surface. While the time derivative corresponds to the speed of the wave propagation, the spatial derivatives describe the wave's shape. The corresponding equation is known as the *wave equation*. Another example of such a process is the propagation of heat yielding the *heat equation*. These and further examples can be found in [**Jos**].

Definition 2.19 is very general, but this does not mean that arbitrary systems are treated in the theory of PDEs. One rather studies those kinds of equations that occur in various applications in physics, technique, or other natural sciences. For various criteria that allow a classification of the corresponding PDEs we again refer to [**Jos**].

In the context of electrical networks, one deals with PDEs that involve independent variables (x, t), where t denotes the time variable and x is a spatial variable for the position in $\mathbb{R}$, $\mathbb{R}^2$, or $\mathbb{R}^3$. For simplicity, in the following we denote the partial derivative with respect to time by

$$\dot{z} := z_t = \frac{\partial z}{\partial t}.$$

The spatial derivatives are denoted by z_{x_i}, in case x is one-dimensional we write

$$z' := z_x = \frac{\partial z}{\partial x}.$$

As each system of PDEs may be transformed into first order formulation [**Jef, MarBar**], we from now on restrict ourselves to PDEs of the form

$$F(z, \dot{z}, z_{x_1}, \dots, z_{x_d}, x, t) = 0, \tag{2.34}$$

where $d \in \{1, 2, 3\}$. Moreover, systems arising from electrical networks have a number of equations that equals the number of dependent variables z, so we only consider "square systems", i.e. z and F both map to $\mathbb{R}^n$.

Definition 2.20. *Let $F : \Omega \times \mathcal{T} \times \mathbb{D}_z \times \mathbb{D}_{\dot{z}} \times \mathbb{D}_{z_{x_1}} \times \ldots \times \mathbb{D}_{z_{x_d}} \to \mathbb{R}^n$ be differentiable w.r.t. $\dot{z}$, where $z : \Omega \times \mathcal{T} \to \mathbb{R}^n$ is differentiable, $\Omega \subseteq \mathbb{R}^d$ an open subset with $d \in \{1, 2, 3\}$, $\mathcal{T} \subseteq \mathbb{R}$ a time interval, and $\mathbb{D}_z, \mathbb{D}_{\dot{z}}, \mathbb{D}_{z_{x_1}}, \dots, \mathbb{D}_{z_{x_d}} \subseteq \mathbb{R}^n$ open. Let further $z(x, t) \in \mathbb{D}_z$, $\dot{z}(x, t) \in \mathbb{D}_{\dot{z}}$, and $z_{x_i}(x, t) \in \mathbb{D}_{z_{x_i}}$ for all $(x, t) \in \Omega \times \mathcal{T}$ and $i = 1, \dots, d$.*

If the Jacobian $D_{\dot{z}}F$ is singular on $\Omega \times \mathcal{T} \times \mathbb{D}_z \times \mathbb{D}_{\dot{z}} \times \mathbb{D}_{z_{x_1}} \times \ldots \times \mathbb{D}_{z_{x_d}}$, the system of first order implicit partial differential equations

$$F\big((x, t), z, \dot{z}, z_{x_1}, \dots, z_{x_d}\big) = 0 \tag{2.35}$$

is called a ***partial differential-algebraic equation (PDAE)*** *.*

Systems of PDAEs arising in the analysis of electrical networks usually are coupled systems of DAEs and PDEs. The following proposition shows that, in fact, such systems are PDAEs in the sense of Definition 2.20.

Proposition 2.21. *Systems combined of a DAE part F and a PDE part G are PDAEs in the sense of Definition 2.20.*

Proof. A system

$$\left(\begin{array}{l} 0 = F(w, \dot{w}, t) \\ 0 = G((x,t), z, \dot{z}, z_{x_1}, \ldots, z_{x_d}) \end{array} \right)$$

with $x \in \Omega \subseteq \mathbb{R}^d$ and $t \in \mathcal{T} \subseteq \mathbb{R}$ can be written as

$$\widetilde{F}\big((w,z), \dot{(w,z)}, (w,z)_{x_1}, \ldots, (w,z)_{x_d}, x, t\big) = 0$$

and, therefore, with $\dot{(w,z)} = \frac{\partial}{\partial t}(w,z)$, one has

$$D_{\dot{(w,z)}}\widetilde{F} = \begin{bmatrix} D_{\dot{w}}F & 0 \\ 0 & D_{\dot{z}}G \end{bmatrix}.$$

F being a system of DAEs yields $D_{\dot{w}}F$ singular on $\mathbb{D}_w \times \mathbb{D}_{\dot{w}} \times \mathcal{T}$. The determinant of $D_{\dot{(w,z)}}\widetilde{F}$ is $\det(D_{\dot{w}}F) \cdot \det(D_{\dot{z}}G)$ which proves the proposition. □

2.2.4. Examples. This subsection is dedicated to the presentation of some small examples for systems of DAEs and PDAEs set up from electrical circuits. The PDAEs treated here are coupled systems of DAEs and PDEs, where the PDE part is used to model certain circuit components linked to the remaining ones at their physical *boundaries*. Therefore, these systems can be considered as a PDE *boundary value problem* whose boundaries have to satisfy a DAE constraint [**Reis05**].

Note that in the following, the dependent variables x and z in a system of DAEs or PDAEs $F(x, \dot{x}, t) = 0$ or $F\big((x,t), z, \dot{z}, z_{x_1}, \ldots, z_{x_d}\big) = 0$ have to be considered as current and voltage variables, since we are in the context of electrical circuits. Further, they also consist of auxiliary ones for (internal) modelling purposes (compare the above notation to the one of systems of equations in the section about network analysis).

2.2.4.1. *A Voltage-Divider Circuit.* As a small example for a system of DAEs arising in analog circuit analysis we consider the voltage-divider circuit shown in Figure 2.1 (a) on page 12 with four network elements R_1, R_2, R_L, and V_0. The vectors of the corresponding branch currents and voltages are denoted by

$$i = (i_{R_1}, i_{R_2}, i_{R_L}, i_{V_0})^T \quad \text{and} \quad u = (u_{R_1}, u_{R_2}, u_{R_L}, u_{V_0})^T,$$

respectively, and the node voltages at nodes 1 and 2 are denoted by v_1, v_2. The reduced nodal incidence matrix A computes to

$$A = \begin{bmatrix} 1 & 0 & 0 & 1 \\ -1 & 1 & 1 & 0 \end{bmatrix}.$$

The three resistors' current-voltage characteristics can be expressed in admittance formulation, but not the one of the independent voltage source given by $u_{V_0} = U_0$. Hence,

$i = (i_1, i_2)$ is separated such that i_1 consists of the first three entries. Applying the modified nodal analysis as described in Section 2.1.5 yields the system of DAEs

$$\begin{aligned} i_{V_0} &= -\frac{1}{R_1}(v_1 - v_2), \\ 0 &= \frac{1}{R_1}(v_1 - v_2) - \frac{1}{R_2}v_2 - \frac{1}{R_L}v_2, \\ 0 &= v_1 - U_0 \end{aligned}$$

in three variables i_{V_0}, v_1, v_2 as a mathematical model for the voltage divider. R_1, R_2, R_L, and U_0 are *parameters* of the system – which can also work as *inputs*, e.g. by applying a certain *input voltage* U_0.

If the sparse tableau analysis instead of the modified nodal approach is used, one ends up with eight instead of three equations

$$\begin{aligned} 0 &= u_{R_1} + u_{R_2} - u_{V_0}, \\ 0 &= u_{R_1} + u_{R_L} - u_{V_0}, \\ 0 &= i_{R_1} + i_{V_0}, \\ 0 &= -i_{R_1} + i_{R_2} + i_{R_L}, \\ U_0 &= u_{V_0}, \\ 0 &= R_1 \cdot i_{R_1} - u_{R_1}, \\ 0 &= R_2 \cdot i_{R_2} - u_{R_2}, \\ 0 &= R_L \cdot i_{R_L} - u_{R_L} \end{aligned}$$

in eight variables $(i, u)^T = (i_{R_1}, i_{R_2}, i_{R_L}, i_{V_0}, u_{R_1}, u_{R_2}, u_{R_L}, u_{V_0})^T$.

For a second example, consider again the voltage-divider circuit, but this time with the resistor R_1 being replaced by a nonlinear diode D with current-voltage characteristic given by the Shockley equation [**Sho**],

$$i_D = I_s \cdot \left(e^{\frac{1}{V_T} u_D} - 1 \right).$$

Its current and voltage i_D, u_D are assumed to be directed from node 1 to node 2 (cf. notations in Figure 2.1 (a)). The further procedure is the same as before; the reduced nodal incidence matrix A did not change, of course. Since D obviously has an admittance formulation, $i = (i_D, i_{R_2}, i_{R_L}, i_{V_0})$ is divided in $i = (i_1, i_2)$, where i_1 again consists of the first three entries of i. Thus, proceeding with the MNA, one finally obtains

$$\begin{aligned} i_{V_0} &= -I_s \cdot \left(e^{\frac{1}{V_T}(v_1 - v_2)} - 1 \right), \\ 0 &= I_s \cdot \left(e^{\frac{1}{V_T}(v_1 - v_2)} - 1 \right) - \frac{1}{R_2}v_2 - \frac{1}{R_L}v_2, \\ 0 &= v_1 - U_0. \end{aligned}$$

2.2.4.2. ***Transmission Line Equations in Electrical Circuits.*** As an example for a system of PDAEs arising in the analysis of an analog circuit, we consider the

differential-amplifier circuit shown in Figure 4.1 on page 88. It contains three *transmission lines* connecting voltage sources with the remaining circuit components. Transmission line effects in electrical circuits appear when interconnections between circuit elements cannot be modelled as short circuits due to effects like *energy loss* and *time* or *phase delay*. For further information, we refer to **[Gra, Gue00, Gue01, Miri, Reis06, Ung]**.

In this example, we are actually dealing with a coupled system of DAEs and (hyperbolic) PDEs, the latter one modelling the transmission lines. The *supply voltage sources* `VCC` and `VEE` of the amplifier and its *input voltage source* `V1` are connected to the remaining circuit components via the transmission lines. To model their physical effects in the interconnection of the circuit, the remaining network equations are coupled with the so-called ***telegrapher's equations***

$$\begin{aligned} -\tfrac{\partial u(x,t)}{\partial x} &= R' \cdot i(x,t) + L' \cdot \tfrac{\partial i(x,t)}{\partial t} \\ -\tfrac{\partial i(x,t)}{\partial x} &= G' \cdot u(x,t) + C' \cdot \tfrac{\partial u(x,t)}{\partial t}. \end{aligned} \tag{2.36}$$

In this system, $i(x,t)$ and $u(x,t)$ denote the current and the voltage at time t and position x in a single transmission line of length l. The magnitudes $R' = \frac{R}{l}$, $L' = \frac{L}{l}$, $C' = \frac{C}{l}$, and $G' = \frac{G}{l}$ are normalized parameters to model the following physical effects of the – considered homogeneous – transmission line; the resistance of such a line is denoted by R. Further, its self-inductance corresponding to a magnetic field is taken into account by an inductance L. Finally, the capacitance C simulates the electric field between the transmission line and the ground, and G corresponds to the corona and leakage currents along the insulator surfaces **[Miri]**.

Equations (2.36) can be derived from the *Maxwell equations* (cf. **[Gue01]** for details). However, here we will use an equivalent circuit for a small part of length Δx of the transmission line instead which is shown in Figure 2.4. This length is small enough to consider the currents and voltages at its ends as independent of the position and dependent only on the *concentrated elements* $R'\Delta x$, $L'\Delta x$, $C'\Delta x$, and $G'\Delta x$.

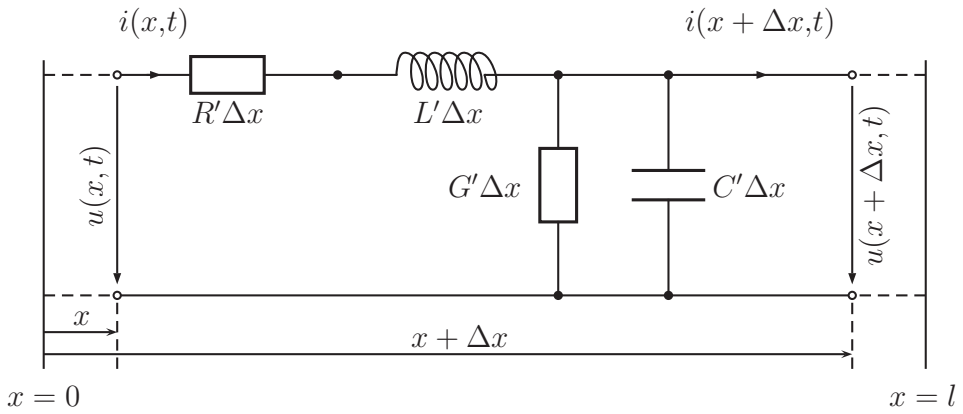


FIGURE 2.4. Equivalent circuit for a small part of length Δx of an electrically long homogeneous transmission line.

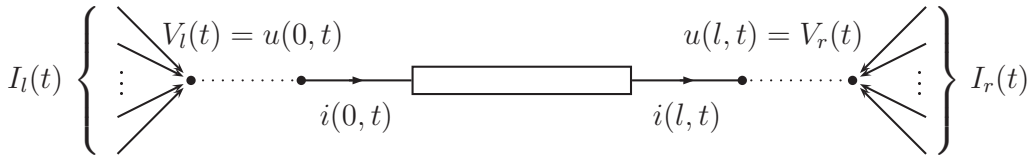


FIGURE 2.5. Connecting the system of PDEs of the telegrapher's equations to the remaining network equations by using boundary conditions (2.38).

Using the Kirchhoff laws (2.1) adapted to this circuit, one easily obtains

$$0 = i(x,t) - i(x+\Delta x,t) - G'\Delta x \cdot u(x+\Delta x,t) - C'\Delta x \cdot \frac{\partial u(x+\Delta x,t)}{\partial t}, \tag{2.37a}$$

$$0 = R'\Delta x \cdot i(x,t) + L'\Delta x \cdot \frac{\partial i(x,t)}{\partial t} + u(x+\Delta x,t) - u(x,t) = 0. \tag{2.37b}$$

Dividing by Δx and a subsequent limit transition $\Delta x \to 0$ finally yields the two coupled partial differential equations (2.36).

Now the telegrapher's equations can be coupled with the remaining network equations by the values at the boundary of the transmission line:

$$\begin{aligned} u(0,t) &= V_l(t), & i(0,t) &= I_l(t), \\ u(l,t) &= V_r(t), & i(l,t) &= -I_r(t). \end{aligned} \tag{2.38}$$

Here, $V_l(t)$ and $V_r(t)$ denote the voltage potentials at time t of the nodes to which the left respectively right end of the transmission line is connected, whereas $I_l(t)$ and $I_r(t)$ are the sums of all the remaining *incoming* currents of these nodes (see Figure 2.5).

Thus, one obtains an *initial-boundary value problem* for a mixed system of DAEs and (hyperbolic) PDEs.

2.2.4.3. *Drift-Diffusion Model for Semiconductor Devices.* The last subsection is dedicated to a second example for a system of PDAEs in the context of analog circuit analysis. It arises from circuit netlists that involve semiconductor devices such as diodes (Figure 2.6) or transistors. The behavior of such devices can be modelled by the *drift-diffusion equations*. They are treated rather shortly in this subsection, where we focus on diodes. For more detailed information we refer to Section B in the appendix and [**GerKneVog, Gue01, MarRinSch, Sel, Tis**]. There is also an explanation about how a diode works provided in the appendix.

The simplest model for the current-voltage relation of a diode is given by the *Shockley equation* [**Sho**]

$$i_D = I_s \cdot \left(e^{\frac{1}{V_T} u_D} - 1 \right),$$

where i_D and u_D are the current through and the voltage above the diode D, I_s is the *reverse saturation current*, and V_T is the *thermal voltage* which at room temperature is approximately 0.026 V. This equation is obtained by some simplifying assumptions on the one-dimensional pn-diode (Figure 2.6, right).

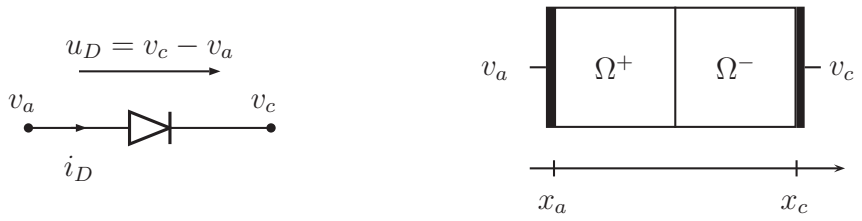


FIGURE 2.6. The pn-diode; on the left the symbol for the network component and on the right a sketch of its one-dimensional geometry including the two positively (Ω^+) and negatively (Ω^-) doped regions.

Without these simplifying assumptions one obtains a system of partial differential equations, the ***drift-diffusion equations***. They are given by

$$\operatorname{div}(\varepsilon \operatorname{grad} V) = q(n - p - N), \tag{2.39a}$$

$$\operatorname{div} J_n = q(\partial_t n + R), \tag{2.39b}$$

$$\operatorname{div} J_p = q(-\partial_t p - R), \tag{2.39c}$$

$$J_n = q(D_n \operatorname{grad} n - \mu_n n \operatorname{grad} V), \tag{2.39d}$$

$$J_p = q(-D_p \operatorname{grad} p - \mu_p p \operatorname{grad} V) \tag{2.39e}$$

in the variables n, p, V, J_n, and J_p which depend on both the time t and the spatial position $x \in \Omega^+ \cup \Omega^- =: \Omega \subseteq \mathbb{R}^d$, $d = 1, 2, 3$ (cf. Figures 2.6 and B.3 on page 151 in the appendix). The mathematical operators for the divergence

$$\operatorname{div}(\omega_1, \ldots, \omega_d) = \partial_{x_1}\omega_1 + \ldots + \partial_{x_d}\omega_d$$

and the gradient

$$\operatorname{grad}(f) = (\partial_{x_1} f, \ldots, \partial_{x_d} f)$$

are applied only with respect to the spatial variable x, the time variable t is excluded. The magnitude V denotes the *electrostatic potential*, so $-\operatorname{grad} V$ is the *electric field* caused by free positive and negative charge carriers, i.e. *electrons* and *holes*, in the interior of the semiconductor device. n and p denote the *concentrations of electrons and holes*, respectively. Their *current densities* are given by J_n and J_p, while D_n, D_p, μ_n, and μ_p are their *diffusion coefficients* and *mobilities*. ε is the so-called *permittivity constant* with an approximate value in silicon of $10^{-12}\frac{\mathrm{A\,s}}{\mathrm{V\,cm}}$. The *elementary charge* q has a value of approximately $1e := 1.60218 \cdot 10^{-19}\mathrm{A\,s}$. R is the so-called *generation-recombination rate* which describes the rate of generation of electron-hole pairs and their disappearance in case of a neutralization or recombination. $N = N(x)$ on the right side of (2.39a) is the *doping concentration* which describes the *preconcentration* of fixed *impurity atoms* in the silicon crystal. $N(x) = N_D^+(x) - N_A^-(x)$ is composed of the doping concentrations N_D^+ and N_A^- of *donor* and *acceptor atoms*, respectively. This preconcentration determines the positively and negatively doped regions of the silicon crystal, thus making the drift-diffusion equations available for the modelling of any semiconductor device (cf. Figure B.1 on page 147 in the appendix).

To set up the *boundary conditions*, two different kinds of bounding parts of the semiconductor are distinguished. While the metal (*Ohmic*) contacts are denoted by Γ_D, the insulating remaining part of the boundary is denoted by Γ_N (see also Figure B.4 on page 153 in the appendix). Therefore, one has $\Gamma_D \cup \Gamma_N = \partial\Omega$, $\Gamma_D \cap \Gamma_N = \emptyset$ [11]. At the contacts Γ_D, the *space charge* given by the right-hand side of (2.39a) vanishes, so $n - p - N = 0$ for $x \in \Gamma_D$. Considering the contacts to be ideal yields $n \cdot p = n_i^2$ for $x \in \Gamma_D$, where n_i is the *intrinsic density* ($\approx 10^{11}\,\mathrm{cm}^{-3}$ in silicon at room temperature). Combining these two relations, one has

$$n(x,t) \;=\; n(x) \;=\; \frac{1}{2}\left(\sqrt{N^2(x) + 4n_i^2(x)} + N(x)\right), \tag{2.40a}$$

$$p(x,t) \;=\; p(x) \;=\; \frac{1}{2}\left(\sqrt{N^2(x) + 4n_i^2(x)} - N(x)\right) \tag{2.40b}$$

for $x \in \Gamma_D$. For the electrostatic potential V, one has

$$V(x,t) = V_{\mathrm{ap}}(t) + V_{\mathrm{bi}}(x) \quad \text{for } x \in \Gamma_D, \tag{2.41}$$

where $V_{\mathrm{bi}}(x) = V_T \cdot \ln\left(\frac{n(x)}{n_i(x)}\right) = -V_T \cdot \ln\left(\frac{p(x)}{n_i(x)}\right)$, $x \in \Gamma_D$, is the so-called *built-in voltage* and V_{ap} is the applied voltage potential at the diode's contacts. Since there is no current flow at the insulating parts Γ_N and a zero electric field in its normal direction, one additionally has

$$J_n(x,t) \cdot \nu = 0, \tag{2.42a}$$

$$J_p(x,t) \cdot \nu = 0, \tag{2.42b}$$

$$\mathrm{grad}(V) \cdot \nu = 0 \tag{2.42c}$$

for $x \in \Gamma_N$, where ν is the unit outward normal vector on Γ_N. Finally, the initial values of the charge carrier concentrations n and p are prescribed by

$$n(x,0) = n_{\mathrm{init}}(x), \tag{2.43a}$$

$$p(x,0) = p_{\mathrm{init}}(x), \tag{2.43b}$$

where $x \in \Omega$ is the position in the silicon crystal.

Thus, by equations (2.39) through (2.43) one obtains an initial-boundary value problem which can be linked to the remaining network equations by

$$V_{\mathrm{ap},1}(t) = V_l(t), \tag{2.44a}$$

$$V_{\mathrm{ap},2}(t) = V_r(t), \tag{2.44b}$$

$$i_1(t) = -I_l(t), \tag{2.44c}$$

$$i_2(t) = -I_r(t). \tag{2.44d}$$

$V_{\mathrm{ap},1}$ and $V_{\mathrm{ap},2}$ are the voltage potentials applied to the diode's *left* and *right* contacts Γ_1 and Γ_2 whose disjoint union is $\Gamma_1 \cup \Gamma_2 = \Gamma_D$ (cf. Figure 2.7). V_l and V_r are the voltage potentials of the corresponding left and right connecting nodes n_l and n_r in the entire network. The currents I_l and I_r are the sums of all the remaining *incoming* currents

[11] The indices are due to the kind of boundary conditions. For the metal contacts Γ_D, the values of the corresponding *variables* are prescribed, so one has *Dirichlet boundary conditions*. For the insulating part Γ_N, one has *Neumann boundary conditions* because the values of *partial derivatives* are given.

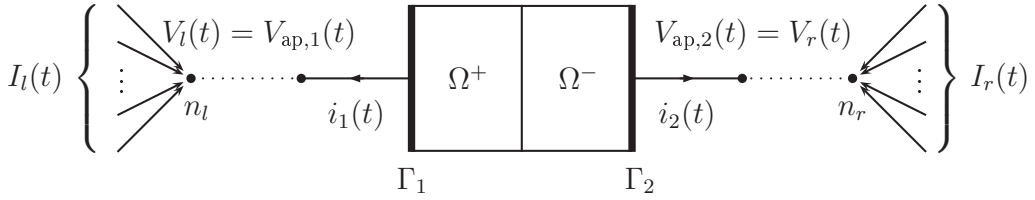


FIGURE 2.7. Connecting the drift-diffusion model equations for the diode to the remaining network equations.

of n_l and n_r, and finally the currents i_j, $j = 1, 2$, are the currents *leaving* the diode and flowing through its metal contacts Γ_1 and Γ_2 towards the nodes n_l and n_r. These currents are given by

$$i_j = \int_{\Gamma_j} (J_n + J_p - \varepsilon \partial_t \operatorname{grad} V) \cdot \nu_j \, d\gamma, \tag{2.45}$$

where ν_j is the unit outward normal vector on Γ_j.

As in the previous subsection about transmission lines, the drift-diffusion model for semiconductor devices interconnected to the remaining circuit components yields a coupled system of PDEs and DAEs. It can be transformed into a system of DAEs by applying a semidiscretization with respect to the position variable x. Depending on the refinement, this can increase the system's dimension drastically.

2.3. Numerical Analysis Methods

In this thesis, for the largest part we are dealing with *symbolic* analysis including the generation of *symbolic* systems of equations with parameters p given as symbols instead of numerical values. For a simulation, however, one of course needs numerical data to solve such a system. Hence, one needs to define a *design point* π, i.e. a set of numerical values for the symbolic parameters occuring in the system. To stress the system's dependence on the symbolic parameters p, we will denote it by $F(x, \dot{x}, t; p)$ instead of $F(x, \dot{x}, t)$ in the following definition. Further, since the most widely used numerical methods for solving systems of PDEs are based on semidiscretizations w.r.t. the spatial variable x, as mentioned earlier we focus on systems of DAEs in subsequent sections and chapters.

DEFINITION 2.22. *Let $F(x, \dot{x}, t; p) = 0$ denote a (parametrized) system of DAEs with symbolic parameter vector $p = (p_1, \ldots, p_N)$ and $\mathcal{T} \subseteq \mathbb{R}$ a time interval. A numerical realization $\pi = (\pi_1, \ldots, \pi_N)$ of p with $\pi_i \in \mathbb{R}$ or $\pi_i : \mathcal{T} \to \mathbb{R}$ is called a* ***design point*** *of F.*

In the following subsections, we review three important numerical analysis methods, i.e. *DC analysis*, *AC analysis*, and *transient analysis*[12]. For any of these methods it is necessary to consider the system F as an *input-output system*. For this, some of the symbolic parameters p will be tagged to be *inputs* $u = (u_1, \ldots, u_m)$, while the

[12]Further numerical analysis methods such as noise, pole/zero (cf., e.g., [**Hen**]), or parametric analysis will not be treated in this thesis.

output is a linear combination of (some of) the internal variables x and denoted by $y = (y_1, \ldots, y_k) = Cx$. Therefore, we will use the notation

$$\widetilde{F}(x, \dot{x}, y, t; u) := \widetilde{F}(x, \dot{x}, y, t; u, \widetilde{p}) = \begin{pmatrix} F(x, \dot{x}, t; p) \\ y - Cx \end{pmatrix} = 0$$

rather than $F(x, \dot{x}, t) = 0$ to denote the corresponding system throughout this section, where $\widetilde{p}$ are the remaining parameters in p without the tagged ones in u.

The most important numerical analysis method in this thesis is the transient analysis. However, later on we will denote any of the analysis methods in this section simply by $\mathcal{A}$. With this notation, $y = \mathcal{A}(F, u)$ denotes the output that is obtained by using the numerical analysis method $\mathcal{A}$ for the simulation of a system of equations $F = 0$, where the input and output are u and y, respectively. Note that a suitable design point π is necessary for any of the analysis methods $\mathcal{A}$, however, we will not explicitly mention it in the following subsections.

2.3.1. Transient Analysis. The *transient analysis* $\mathcal{A}_{\text{tran}}$ of a dynamical system $F = 0$ aspires the computation of the time-dependent output $y(t) = \mathcal{A}_{\text{tran}}(F, u(t))$ corresponding to given time-dependent input signals $u(t)$, $t \in \mathcal{T} \subseteq \mathbb{R}$. Mathematically, the system F is numerically integrated.

Definition 2.23. *Let $F(x, \dot{x}, y, t; u) : \mathbb{D}_x \times \mathbb{D}_{\dot{x}} \times \mathbb{D}_y \times \mathcal{T} \to \mathbb{R}^n$ be a system of DAEs, where $\mathbb{D}_x, \mathbb{D}_{\dot{x}} \subseteq \mathbb{R}^n$ open, $\mathbb{D}_y \subseteq \mathbb{R}^k$ open, $\mathcal{T} = [t_0, t_1] \subseteq \mathbb{R}$ is a time interval, and $u : \mathcal{T} \to \mathbb{R}^m$ is piecewise continuous. Let further $\dot{x}_0 \in \mathbb{R}^n$ and $(x, y) \in C^1(\mathcal{T}, \mathbb{R}^{n+k})$ be a solution of*

$$\begin{aligned} 0 &= F(x(t_0), \dot{x}_0, y(t_0), t_0; u(t_0)) \\ 0 &= F(x(t), \dot{x}(t), y(t), t; u(t)) \quad \text{for } t \in \mathcal{T}. \end{aligned} \tag{2.46}$$

*Then $(\boldsymbol{x}, \boldsymbol{y})$ is a **transient solution of $\boldsymbol{F}$ for the input $\boldsymbol{u}$**. The **transient analysis** $\mathcal{A}_{tran}$ of F is defined via $\boldsymbol{\mathcal{A}_{tran}(F, u, \mathcal{T}) := (x, y)}$.*

As stated earlier in Section 2.2.1, not the value of x is prescribed at $t = t_0$, but the value of the derivative $\dot{x}(t_0)$.

2.3.2. DC Analysis. The *DC analysis*[13] $\mathcal{A}_{\text{dc}}$ investigates the steady-state system's behavior without any dynamical influences, i.e. the system is analysed for $t \to \infty$ applying a constant excitation. Therefore, the dynamical components of F, i.e. the occuring derivatives of the system variables, are set to zero which physically means the replacement of, e.g., capacitances by open circuits and inductances by short circuits, since

$$\begin{aligned} i_C(t) &= C \cdot \partial_t u_C(t), \qquad \partial_t u_C \equiv 0, \\ u_L(t) &= L \cdot \partial_t i_L(t), \qquad \partial_t i_L \equiv 0. \end{aligned}$$

This leads to a *static system* F_{dc}. Then, for constant input signals u the static system F_{dc} is solved.

[13] DC means **d**irect **c**urrent.

Definition 2.24. *Let $F(x, \dot{x}, y, t; u) : \mathbb{D}_x \times \mathbb{D}_{\dot{x}} \times \mathbb{D}_y \times \mathcal{T} \to \mathbb{R}^n$ be a system of DAEs, where $\mathbb{D}_x, \mathbb{D}_{\dot{x}} \subseteq \mathbb{R}^n$ open, $\mathbb{D}_y \subseteq \mathbb{R}^k$ open, $\mathcal{T} \subseteq \mathbb{R}$ is a time interval, and $t_0 \in \mathcal{T}$. Let $F_{dc}(x, y; u) : \mathbb{D}_x \times \mathbb{D}_y \to \mathbb{R}^n$ be defined by*

$$F_{dc}(x, y; u) := F(x, 0, y, t_0; u) \quad \text{for} \quad (x, 0, y, t_0) \in \mathbb{D}_x \times \mathbb{D}_{\dot{x}} \times \mathbb{D}_y \times \mathcal{T}\,. \tag{2.47}$$

*Then F_{dc} is the **DC system** or **static system of F at t_0**. If $u_0 \in \mathbb{R}^m$ is given and $(x_0, y_0) \in \mathbb{R}^{n+k}$ is a solution of*

$$F_{dc}(x_0, y_0; u_0) = 0, \tag{2.48}$$

*then (x_0, y_0) is a **DC point** or **operating point of F**. The **DC analysis** $\mathcal{A}_{dc}$ of F is defined by $\boldsymbol{\mathcal{A}_{dc}(F, u_0) := (x_0, y_0)}$.*

Often one is not only interested in a single operating point, but rather in the behavior of the static system w.r.t. a certain parameter. This leads to the so-called *DC transfer analysis* or *DT analysis*.

Definition 2.25. *Let $F(x, \dot{x}, y, t; u) : \mathbb{D}_x \times \mathbb{D}_{\dot{x}} \times \mathbb{D}_y \times \mathcal{T} \to \mathbb{R}^n$ be a system of DAEs, where $\mathbb{D}_x, \mathbb{D}_{\dot{x}} \subseteq \mathbb{R}^n$ open, $\mathbb{D}_y \subseteq \mathbb{R}^k$ open, $\mathcal{T} \subseteq \mathbb{R}$ is a time interval, and $t_0 \in \mathcal{T}$. Let further $U \subseteq \mathbb{R}^m$ and for $u \in U$ let $(x_u, y_u) \in \mathbb{R}^{n+k}$ be a solution of*

$$F_{dc}(x_u, v_u; u) = 0.$$

*If $(x, y) : U \to \mathbb{R}^{n+k}$ is defined by $u \mapsto (x_u, y_u)$, the **DC transfer solution** or **DT solution** of F and u is given by (x, y). The **DC transfer analysis** or **DT analysis** $\mathcal{A}_{dt}$ of F is defined via $\boldsymbol{\mathcal{A}_{dt}(F, U) := (x, y)}$.*

2.3.3. AC Analysis. The *small-signal* or *AC analysis* investigates the linearized system behavior, where the input is a small sinusoidal signal with a constant frequency. For this, the system is linearized in a certain linearization point, mostly a beforehand computed DC operating point. From a mixed electrical engineering and computational point of view, this corresponds to

- "bringing the system to the linearization point" by using appropriate constant (DC) sources instead of time-dependent (AC) sources (The "small signals" are not yet taken into account.)
- and linearizing the system by its tangent (space) in this linearization point.

If, for example, a current-voltage relation $i = f(u)$ and a linearization point u_0 are given, one will have a linearization

$$i = f(u_0) + (u - u_0) \cdot \left(\frac{\partial f(u)}{\partial u}\bigg|_{u=u_0} \right) + \text{ (higher order terms)}$$

via a Taylor expansion. The first summand on the right side corresponds to the DC part, i.e. the linearization point. The second summand is the product of the *signal voltage*[14] $u - u_0$ and the *differential conductance* at u_0. The *higher order terms* are neglected, since one is interested only in a linearization.

[14] In general, the signal voltage $u - u_0$ is small in comparison to the voltage u_0 of the linearization point which motivates the term *small signal analysis*.

Usually, during linearization the system is transferred to frequency domain by a Fourier or Laplace transformation. The following definition explains how a general nonlinear system F can be linearized to obtain a corresponding *AC system* F_{ac}.

Definition 2.26. (Linearization)
Let $F(x, \dot{x}, y, t; u) : \mathbb{D}_x \times \mathbb{D}_{\dot{x}} \times \mathbb{D}_y \times \mathcal{T} \to \mathbb{R}^n$ *be a system of DAEs, where* $\mathbb{D}_x, \mathbb{D}_{\dot{x}} \subseteq \mathbb{R}^n$ *open,* $\mathbb{D}_y \subseteq \mathbb{R}^k$ *open,* $u : \mathcal{T} \to \mathbb{R}^m$*, and* $\mathcal{T} \subseteq \mathbb{R}$ *is a time interval. Further, let* $(x_0, \dot{x}_0, y_0, t_0, u_0) \in \mathbb{R}^{2n+k+1+m}$ *be a linearization point such that*

$$F(x_0, \dot{x}_0, y_0, t_0; u_0) = 0. \tag{2.49}$$

Define the Jacobians A_1*,* A_2*,* A_3*, and* B *by*

$$\begin{aligned} A_1 &:= D_x F(x_0, \dot{x}_0, y_0, t_0; u_0) \in \mathbb{R}^{(n+k)\times n}, \\ A_2 &:= D_{\dot{x}} F(x_0, \dot{x}_0, y_0, t_0; u_0) \in \mathbb{R}^{(n+k)\times n}, \\ A_3 &:= D_y F(x_0, \dot{x}_0, y_0, t_0; u_0) \in \mathbb{R}^{(n+k)\times k}, \\ B &:= D_u F(x_0, \dot{x}_0, y_0, t_0; u_0) \in \mathbb{R}^{(n+k)\times m}. \end{aligned} \tag{2.50}$$

Let s *denote the* ***Laplace variable*** *and* $U(s)$ *the* ***Laplace transform*** *of the input* $u(t)$*. Then for* $s \in \mathbb{C}$ *let*

$$\begin{aligned} A(s) &:= (A_1 + s \cdot A_2, A_3) \in \mathbb{C}^{(n+k)\times(n+k)}, \\ b(s) &:= -B \cdot U(s) \qquad\quad \in \mathbb{C}^{n+k}. \end{aligned} \tag{2.51}$$

Finally, the linear system $F_{ac}(x, y, s)$ *defined by*

$$F_{ac}(X, Y, s) := A(s) \cdot \begin{pmatrix} X \\ Y \end{pmatrix} - b(s) \tag{2.52}$$

is the ***AC system*** *of* F *w.r.t. the input* $u(t)$ *and the linearization point* $(x_0, \dot{x}_0, y_0, t_0, u_0)$*.*

The procedure in the above definition linearizes a general system of DAEs in the given linearization point. The following example shows that this procedure accomplishes the intuitive idea of a linearization.

Example 2.27. Consider the system $F : x - u^2 = 0$ in $\mathbb{R}^2$. This parabola obviously has the tangent line

$$x = 2u - 1$$

in $\rho = (u_0, x_0) = (1, 1)$.

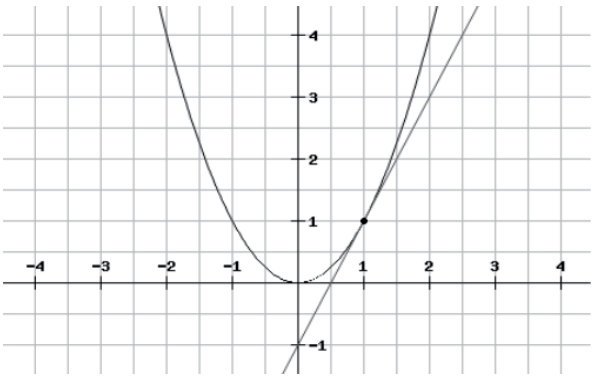


Figure 2.8. Linearization of the parabola in $(1, 1)$.

Algebraically, one can compute the tangent space $T_\rho F$ in ρ using

$$T_\rho F \perp \operatorname{grad}(F)(\rho),$$

where $\operatorname{grad}(F) = (-2u, 1)$. This leads to the tangent space $T_\rho F$ being the span of the vector $(1, 2)$ and, hence, $T_\rho F : x - 2u = 0$. Finally, the linearization of system F in ρ is given by

$$\begin{aligned} \rho + T_\rho F &= (1,1) + \lambda \cdot (1,2), \quad \lambda \in \mathbb{R} \\ &= (\underbrace{1+\lambda}_{u}, \underbrace{1+2\lambda}_{x}), \quad \lambda \in \mathbb{R} \end{aligned}$$

or, equivalently, by $x = 2u - 1$.

Now consider the given system as an input-output system with the input u and the output $y = x$. According to our notation in this section, this yields

$$0 = F(x, \dot{x}, y, t; u) = \begin{pmatrix} x - u^2 \\ y - x \end{pmatrix}.$$

Let $\rho = (x_0, \dot{x}_0, y_0, t_0, u_0) = (1, \dot{x}_0, 1, t_0, 1)$ denote the linearization point. With the notation in Definition 2.26, one obtains

$$A_1 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad A_3 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad B = \begin{bmatrix} -2 \\ 0 \end{bmatrix},$$

hence

$$A(s) = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}, \quad b(s) = \begin{bmatrix} 2 \\ 0 \end{bmatrix} \cdot U(s),$$

and finally

$$F_{ac}(X, Y, s) = A(s) \cdot \begin{pmatrix} X(s) \\ Y(s) \end{pmatrix} - b(s) = \begin{pmatrix} X(s) - 2U(s) \\ -X(s) + Y(s) \end{pmatrix}.$$

Equating the last expression to zero leads to $(U, X) = \lambda \cdot (1, 2)$, the same as $T_\rho F$ above. □

The following defines the *AC analysis* of a system of DAEs F w.r.t. a linearization point ρ and an input u. It is the solution of the AC system of F along the imaginary axis.

Definition 2.28. *Let $F_{ac}(X, Y, s) = 0$ be an AC system in the sense of Definition 2.26 with respect to a linearization point $\rho = (x_0, \dot{x}_0, y_0, t_0, u_0)$ and an input $u : \mathcal{T} \to \mathbb{R}^m$. If $(X, Y) : W \to \mathbb{C}^{n+k}$, $W \subseteq \mathbb{R}$, satisfies*

$$F_{ac}(X(\omega), Y(\omega), i\omega) = A(i\omega) \cdot \begin{pmatrix} X(\omega) \\ Y(\omega) \end{pmatrix} - b(i\omega) = 0 \quad \text{for } \omega \in W,$$

*then (X, Y) is the **AC solution** of F in W. The **AC analysis** $\mathcal{A}_{ac}$ of F is defined by $\boldsymbol{\mathcal{A}_{ac}(F, u, \rho, W) = (X, Y)}$.*

2.4. Systems and Control Theory

In this section, we give some basic definitions and properties from systems and control theory which will be needed throughout the following sections. It mainly serves to explain the necessary terms and notions. In general, we follow the notes of [**Zer**], however, in some cases we switch to those of [**Ant, HinPri, JosKel, RasNicLoz, Rew, Wil72**].

Some of the definitions and properties in the following subsections are extendable or valid for general dynamical systems. Nevertheless, we mostly restrict ourselves to LTI first order ODEs in state space formulation

$$\begin{aligned} \dot{x} &= Ax + Bu, \\ y &= Cx + Du \end{aligned} \tag{2.53}$$

with $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, and $D \in \mathbb{R}^{p\times m}$, outputs $y : \mathcal{T} \to \mathbb{R}^p$, inputs $u : \mathcal{T} \to \mathbb{R}^m$, and states $x : \mathcal{T} \to \mathbb{R}^n$, where $\mathcal{T}$ usually is $\mathbb{R}$ or $\mathbb{R}_{\geq 0}$.

For the space of *admissible input functions* $\mathcal{U} = \{\, u : \mathcal{T} \to \mathbb{R}^m \,\}$ we further restrict ourselves to *piecewise continuous input functions* $u(\cdot)$ such that the state functions $x(\cdot)$ are *piecewise* $\mathcal{C}^1$. The states x represent the "system's memory" because if $x_0 = x(t_0)$ is known for some $t_0 \in \mathcal{T}$ and u is known on some interval $[t_0, t_1] \subseteq \mathcal{T}$, then x and, hence, y are uniquely determined everywhere in $[t_0, t_1] \subseteq \mathcal{T}$ by

$$x(t) = e^{(t-t_0)A}x_0 + \int_{t_0}^{t} e^{(t-\tau)A}Bu(\tau)\,d\tau. \tag{2.54}$$

Thus, provided the future input is given, $x_0 = x(t_0)$ contains all the information about the "past" to compute the "future".

The ***state transition map***

$$\varphi : \{(t, t_0) \in \mathcal{T}^2 \mid t \geq t_0\} \times \mathcal{X} \times \mathcal{U} \to \mathcal{X}, \quad (t, t_0, x_0, u) \mapsto \varphi(t, t_0, x_0, u), \tag{2.55}$$

where $\mathcal{X} = \{\, x : \mathcal{T} \to \mathbb{R}^n \,\}$ is the *state space* of system (2.53), computes the state $x(t) = \varphi(t, t_0, x_0, u)$ at time t, where x_0 is the state at time t_0 and the applied input is u. More precisely, the value of $\varphi(t, t_0, x_0, u)$ is given by equation (2.54). From this, some properties such as ***time-invariance*** can be derived[15]:

$$\varphi(t, t_0, x_0, u) = \varphi(t - \tau, t_0 - \tau, x_0, \sigma^\tau u), \tag{2.56}$$

where $\sigma^\tau u(t) = u(t + \tau)$. The ***state-to-output map***

$$\eta : \{(t, t_0) \in \mathcal{T}^2 \mid t \geq t_0\} \times \mathcal{X} \times \mathcal{U} \to \mathcal{Y}, \quad (t, t_0, x_0, u) \mapsto \eta(t, t_0, x_0, u), \tag{2.57}$$

where $\mathcal{Y} = \{\, y : \mathcal{T} \to \mathbb{R}^p \,\}$ is the *output space*, computes the output of (2.53) at time t, assumed that x_0 is the state at time t_0 and the applied input is u. Hence, one has

$$\eta(t, t_0, x_0, u) = C \cdot \varphi(t, t_0, x_0, u) + D \cdot u(t)$$

or, more precisely and in correspondence to equation (2.54),

$$\eta(t, t_0, x_0, u) = Ce^{A(t-t_0)}x_0 + \int_{t_0}^{t} Ce^{A(t-\tau)}Bu(\tau)\,d\tau + Du(t). \tag{2.58}$$

[15]Note that system (2.53) is *time-invariant*, i.e. the system matrices A, B, C, D are independent of $t \in \mathcal{T}$. For further properties of φ we refer to [**Zer**].

2.4.1. Reachability. In this subsection, the leading question is the *reachability* of a certain state x_1 starting from another particular state x_0 by applying a suitable input function u, i.e. steering system (2.53) from x_0 to x_1.

Definition 2.29. *Let $t_0 \in \mathcal{T}$ be fixed and $x_1 \in \mathcal{X}$.*

(1) ***x_1 can be reached from $x_0 \in \mathcal{X}$ in time $\tau \geq 0$*** *if there exists $u \in \mathcal{U}$ such that*
$$\varphi(t_0 + \tau, t_0, x_0, u) = x_1.$$
(2) ***x_1 can be reached from $x_0 \in \mathcal{X}$*** *if (1) holds for at least one $\tau \geq 0$.*
(3) *System (2.53) is* ***completely reachable from $x_0 \in \mathcal{X}$*** *if any $x_1 \in \mathcal{X}$ is reachable from x_0.*
(4) *It is* ***completely reachable*** *if x_1 can be reached from x_0 for all $x_0, x_1 \in \mathcal{X}$.*

Note that in this definition the starting time t_0 is not important because of the time-invariance (2.56) of the state transition map:
$$\varphi(t_0 + \tau, t_0, x_0, u) = \varphi(\tau, 0, x_0, \sigma^{t_0} u).$$
Thus, for simplicity, one often can choose $t_0 = 0$. By
$$\mathcal{R}(\tau, x_0) := \{\, x \in \mathcal{X} \mid x \text{ is reachable from } x_0 \text{ in time } \tau \,\},$$
the set of states that are reachable from x_0 in time τ is denoted, further
$$\mathcal{R} := \bigcup_{\tau \geq 0} \mathcal{R}(\tau)$$
with $\mathcal{R}(\tau) := \mathcal{R}(\tau, x_0)$ denotes the set of states that are reachable from the initial state. System (2.53) is completely reachable from the initial state x_0 if and only if $\mathcal{R} = \mathcal{X}$.

One can show that $\mathcal{R}, \mathcal{R}(\tau)$ with $0 \leq \tau$ are subspaces of $\mathcal{X} = \{\, x : \mathcal{T} \to \mathbb{R}^n \,\}$. Moreover, $\mathcal{R}(\varepsilon) = \mathcal{R}$ for every $\varepsilon > 0$. Using this, one can further show that

completely reachable $\quad \Leftrightarrow \quad$ completely reachable from x_0.

In order to make a statement about reachability in terms of the system matrices A, B, C, D of (2.53), some further definitions are necessary:

Definition 2.30. *The* ***finite reachability Gramian*** *of (2.53) is defined by*
$$\mathcal{P}(t) = \int_{t_0}^{t_0+t} e^{A(\tau - t_0)} B B^* e^{A^*(\tau - t_0)}\, d\tau \;\in \mathbb{R}^{n \times n} \tag{2.59}$$
using system matrices A, B. For asymptotically stable systems (see Section 2.4.4), it is also defined for $t = \infty$ and
$$\mathcal{P} = \int_{t_0}^{\infty} e^{A(\tau - t_0)} B B^* e^{A^*(\tau - t_0)}\, d\tau \;\in \mathbb{R}^{n \times n} \tag{2.60}$$
is called the ***(infinite) reachability Gramian****. Further,*
$$\mathcal{R}(A, B) = [\; B \quad AB \quad A^2B \quad \dots \quad A^{n-1}B \;] \in \mathbb{R}^{n \times nm} \tag{2.61}$$
is the ***(Kalman) reachability matrix*** *of system (2.53).*

See also Theorem 2.42 and Remark 2.43 on page 50, and (2.78).

Sometimes, the reachability matrix $\mathcal{R}(A,B)$ is given by the infinite matrix

$$\mathcal{R}(A,B) = [\, B \quad AB \quad A^2B \quad \dots \quad A^nB \quad A^{n+1}B \quad \dots\,].$$

However, by the Cayley-Hamilton theorem, i.e.

$$\chi_A(A) = A^n + c_{n-1}A^{n-1} + \ldots + c_1A + c_0I = 0$$

with χ_A the characteristic polynomial of A, the span of the columns of $\mathcal{R}(A,B)$ is determined by the first n terms. We will not distinguish between $\mathcal{R}(A,B)$ as a matrix operator and as a space spanned by its columns.

The following theorem gives a relation between the reachability space $\mathcal{R}$, the reachability Gramian $\mathcal{P}(\cdot)$, and the reachability matrix $\mathcal{R}(A,B)$.

Theorem 2.31. [**Zer**, Thm. 4.9]
Consider system (2.53) *and* $\varepsilon > 0$ *arbitrary. One has*

$$\mathcal{R} = \mathcal{R}(\varepsilon) = \operatorname{im}\mathcal{P}(\varepsilon) = \operatorname{im}\mathcal{R}(A,B).$$

Hence, the following are equivalent:

(1) *System* (2.53) *is reachable.*
(2) $\mathcal{P}(\varepsilon)$ *is non-singular.*
(3) $\mathcal{R}(A,B)$ *has full row rank.*

□

By this theorem, one has

completely reachable $\quad\Leftrightarrow\quad$ rank $\mathcal{R}(A,B) = n$.

Further, in that case, a (smooth) input function steering the system from the initial state x_0 to x in time ε is given by

$$u(t) = B^*e^{A^*(\varepsilon+t_0-t)}\mathcal{P}(\varepsilon)^{-1}\left(x - e^{A\varepsilon}x_0\right)$$

because

$$\begin{aligned}
&\varphi(t_0+\varepsilon, t_0, x_0, u)\\
&\quad = e^{A\varepsilon}x_0 + \textstyle\int_{t_0}^{t_0+\varepsilon} e^{A(t_0+\varepsilon-\tau)}BB^*e^{A^*(t_0+\varepsilon-\tau)}P(\varepsilon)^{-1}\left(x - e^{A\varepsilon}x_0\right)\,d\tau\\
&\quad = e^{A\varepsilon}x_0 + \textstyle\int_0^{\varepsilon} e^{A(\varepsilon-s)}BB^*e^{A^*(\varepsilon-s)}\,ds\Big(\int_0^{\varepsilon} e^{As}BB^*e^{A^*s}\,ds\Big)^{-1}\left(x - e^{A\varepsilon}x_0\right)\\
&\quad = x.
\end{aligned}$$

Since $\mathcal{R}(A,B)$ is built by the system matrices A, B, one also simply says that the ***matrix pencil*** $(\boldsymbol{A,B})$ ***is reachable***.

2.4.2. Observability. Consider the dynamical system (2.53), whose ***state trajectory*** $t \mapsto x(t) = \varphi(t, t_0, x_0, u)$ using the state transition map φ describes its evolution. The main problem to discuss in this section is as follows: In a state space system, the so-called ***manifest variables*** are the input and output variables, while the ***latent variables*** are the states x. In general, the latent variables are auxiliary or artificially introduced, e.g. in order to transform a system to first order. As a consequence, the

physical meaning of latent variables may be obscure. Since they normally cannot be measured directly, the question that arises is about the conclusions that can be drawn w.r.t. the latent variables of a system where we know the manifest ones. This finally leads to the term of an *observable* system that allows the reconstruction of the latent variables from the manifest ones.

Definition 2.32. *Let $t_0 \in \mathcal{T}$ be fixed.*

(1) ***$x_1 \in \mathcal{X}$ can be distinguished from $x_2 \in \mathcal{X}$ in time $\tau \geq 0$*** *if there exists $u \in \mathcal{U}$ and $t_0 \leq t \leq t_0 + \tau$ such that*

$$\eta(t, t_0, x_1, u) \neq \eta(t, t_0, x_2, u).$$

*Then u **distinguishes** between x_1 and x_2 in time τ.*

(2) ***x_1 can be distinguished from x_2*** *if (1) holds for at least one $\tau \geq 0$.*

(3) *System (2.53) is **observable** if for any $x_1, x_2 \in \mathcal{X}$, $x_1 \neq x_2$, the state x_1 can be distinguished from x_2.*

Also in this definition, the starting time t_0 is not important because of the time-invariance (2.56) of the state transition map. Therefore, one often chooses $t_0 = 0$.

Define the set

$$\mathcal{J}(\tau, x) := \{\, x' \in X \mid x' \text{ is indistinguishable from } x \text{ in time } \tau \,\}$$

and let further

$$\mathcal{J} := \bigcap_{\tau \geq 0} \mathcal{J}(\tau)$$

with $\mathcal{J}(\tau) := \mathcal{J}(\tau, x_0)$ denote the set of states that are indistinguishable from the initial state.

One can show that if x can be distinguished from x_0 at all, then it can also be distinguished from x_0 by the zero input function. Therefore, w.l.o.g. we put $\boldsymbol{u \equiv 0}$ for the rest of this subsection. Furthermore, $\mathcal{J}, \mathcal{J}(\tau)$ with $\tau \geq 0$ are subspaces of $\mathcal{X} = \{\, x : \mathcal{T} \to \mathbb{R}^n \,\}$ and $\mathcal{J}(\varepsilon) = \mathcal{J}$ for every $\varepsilon > 0$. Using the above, it can further be shown that

$$\boxed{\text{System (2.53) is observable} \quad \Leftrightarrow \quad \mathcal{J} = \{\, x_0 \,\}.}$$

In order to derive a statement about observability in terms of the system matrices A, B, C, D, two more definitions are necessary:

Definition 2.33. *The **finite observability Gramian** of system (2.53) is defined by*

$$\mathcal{Q}(t) = \int_{t_0}^{t_0+t} e^{A^*(\tau - t_0)} C^* C e^{A(\tau - t_0)} \, d\tau \; \in \mathbb{R}^{n \times n}. \tag{2.62}$$

*Furthermore, for asymptotically stable systems (see Section 2.4.4) the **(infinite) observability Gramian** is defined by*

$$\mathcal{Q} = \int_{t_0}^{\infty} e^{A^*(\tau - t_0)} C^* C e^{A(\tau - t_0)} \, d\tau \; \in \mathbb{R}^{n \times n} \tag{2.63}$$

and further

$$\mathcal{O}(C,A) = \begin{bmatrix} C \\ CA \\ CA^2 \\ \cdots \\ CA^{n-1} \end{bmatrix} \in \mathbb{R}^{np\times n} \tag{2.64}$$

denotes the ***(Kalman) observability matrix*** *of* (2.53).

See also Theorem 2.42 and Remark 2.43 on page 50, and (2.78).

The following theorem describes the relation among the observability space $\mathcal{J}$, the observability matrix $\mathcal{O}(C,A)$ [16], and the observability Gramian $\mathcal{Q}(\cdot)$.

Theorem 2.34. [**Zer**, Thm. 6.5]
Consider system (2.53) *with the zero input function* $u \equiv 0$ *and* $\varepsilon > 0$ *arbitrary. Then*

$$\mathcal{J} = \mathcal{J}(\varepsilon) = \ker \mathcal{Q}(\varepsilon) = \ker \mathcal{O}(C,A).$$

Hence, the following are equivalent:

(1) *The system* $\dot{x} = Ax$, $y = Cx$ *is observable,*
(2) $\mathcal{Q}(\varepsilon)$ *is non-singular,*
(3) $\mathcal{O}(C,A)$ *has full column rank.*

□

By this theorem, one has

$$\boxed{\text{observable} \quad \Leftrightarrow \quad \operatorname{rank} \mathcal{O}(C,A) = n.}$$

Further, in that case, the *latent* state variables x can be reconstructed via (2.54) and the initial state $x_0 = x(t_0)$ given by

$$x_0 = \mathcal{Q}(\varepsilon)^{-1} \int_{t_0}^{t_0+\varepsilon} e^{A^*(\tau - t_0)} C^* y(\tau)\, d\tau,$$

where $y(t) = \eta(t, t_0, x_0, u)$ is given by (2.58).

Since the observability matrix $\mathcal{O}(C,A)$ is built by the system matrices A and C, one also simply says that the ***matrix pencil*** (A, C) ***is observable***, if $\mathcal{O}(C,A)$ is so, i.e. if $\operatorname{rank} \mathcal{O}(C,A) = n$.

2.4.3. Passivity. *Passivity* is a crucial property of dynamical systems which first was introduced in the network theory literature. Generally speaking, passivity means that the system internally does not generate ***"energy"***. A *passive component* is one that either dissipates energy or is incapable of power gain such as resistors, diodes, capacitors etc. Otherwise it is *active* like, for instance, voltage and current sources. Obviously, an electrical circuit composed entirely of passive components is passive.

[16] As in the case of $\mathcal{R}(A,B)$, the observability matrix $\mathcal{O}(C,A)$ sometimes is denoted as an infinite matrix, but due to the Cayley-Hamilton theorem the first n terms are sufficient for the determination of its kernel.

In the following, let $\mathcal{T}_+ = [t_0, \infty)$ for $t_0 \in \mathcal{T}$ and, for $q \in [1, \infty]$, let

$$\mathcal{L}^q := \mathcal{L}^q(\mathcal{T}_+, \mathbb{R}^m) = \left\{ f : \mathcal{T}_+ \to \mathbb{R}^m \mid \int_{\mathcal{T}_+} \|f(t)\|^q \, dt < \infty \right\}$$

denote the Lebesgue space of real-valued functions of time t that are *q-integrable* on $[t_0, \infty)$. In this notation, $\|\cdot\|$ is some norm on $\mathbb{R}^m$. An *extension* of $\mathcal{L}^q$ is defined by

$$\mathcal{L}^q_{\text{ext}} := \mathcal{L}^q_{\text{ext}}(\mathcal{T}_+, \mathbb{R}^m) := \left\{ v \mid v_\tau \in \mathcal{L}^q \; \forall \tau \geq t_0 \right\},$$

where

$$v_\tau(t) = \begin{cases} v(t), & t_0 \leq t \leq \tau, \\ 0, & t > \tau, \end{cases}$$

is a *truncation* of v. Passivity can be defined in the *internal sense* or in the more general *input-output sense* for systems with *equal numbers of inputs and outputs*:

Definition 2.35. [**JosKel**]
Let Σ be a dynamical system in state space representation described by

$$\Sigma : \quad \begin{aligned} \dot{x} &= F(x, u), \\ y &= G(x, u) \end{aligned} \tag{2.65}$$

with F, G smooth, states $x \in \mathcal{X}_+ := \{ x : \mathcal{T}_+ \to \mathbb{R}^n \}$, and the same number of inputs and outputs $u, y \in \mathcal{L}^q_{ext}$.

- *Σ is called **passive in the input-output sense** if there exists a constant β such that*
$$\langle u, y \rangle_\tau + \beta \geq 0 \; \forall u \in \mathcal{L}^q_{ext}, \tau \geq t_0,$$
where $\langle u, y \rangle_\tau = \int_{t_0}^{\tau} u^*(t) y(t) \, dt$.
- *Σ is called **internally passive** if there exists a nonnegative **storage function** $\boldsymbol{S : \mathcal{X}_+ \to \mathbb{R}_{\geq 0}}$ such that*
$$\langle u, y \rangle_\tau \geq S(x(\tau)) - S(x(t_0)) \; \forall u \in \mathcal{L}^q_{ext}, \tau \geq t_0.$$

Input-output passivity is the more general concept, since it only uses inputs and outputs, but no states. However, for (finite-dimensional) reachable and observable state space systems, the above two definitions are equivalent [**JosKel**]. If one thinks of an electrical network with m ports, where m voltages are prescribed and serve as the input to the system, while the corresponding currents are the m outputs of the system, the inner product $\langle u, y \rangle_\tau$ is exactly the integral over the instantaneous electrical power (energy) $u^*(t)y(t)$ at time t. In Definition 2.36, *(internal) passivity* is generalized to the concept of *dissipativity* and the restrictions on the numbers of inputs and outputs are lifted.

In the case of LTI dynamical systems in state space formulation

$$\begin{aligned} E\dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t) + Du(t) \end{aligned} \tag{2.66}$$

with $E, A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{p \times n}$, and $D \in \mathbb{R}^{p \times m}$, a classical result yields conditions for the passivity of (2.66) in terms of its ***transfer function***; assume that the matrix pencil $\lambda E - A$ is regular, i.e. $\det(\lambda E - A) \not\equiv 0$. Then the *transfer function*

of (2.66) is given by the $p \times m$ matrix-valued function H in the independent complex frequency variable s

$$H(s) = C\big(sE - A\big)^{-1}B + D, \tag{2.67}$$

where each of the entries in H is a rational function in $s \in \mathbb{C}$. In general, a system's transfer function is defined by the Laplace transform of its $p \times m$ matrix-valued ***impulse response***[17] $\boldsymbol{h}$, i.e.

$$H(s) = (\mathcal{L}h)(s) = \int_0^\infty h(t)e^{-st}\,dt, \qquad s \in \mathbb{C},$$

and describes its input-output relation $Y(s)/U(s)$ in the frequency domain, i.e.

$$Y(s) = H(s) \cdot U(s).$$

In this notation, $U(s) = (\mathcal{L}u)(s)$ and $Y(s) = (\mathcal{L}y)(s)$ are the Laplace transforms of the input and output, respectively. The formula in (2.67) is obtained by eliminating the Laplace transform $X(s) = (\mathcal{L}x)(s)$ of the state trajectory $x(\cdot)$ from the Laplace transformed system.

It is well known in network theory that a linear time-invariant system of DAEs of the form (2.66) is ***passive*** if and only if its transfer function H is ***positive real***, i.e. H is analytic in $\mathbb{C}_+ = \{\, z \in \mathbb{C} \mid \operatorname{Re}(z) > 0 \,\}$ and $H(s) + H(s)^* \geq 0$ for all $s \in \mathbb{C}_+$.

Dissipativity is a generalization of the passivity concept that uses *storage functions* as a generalization of *stored "energy"* in a system. More precisely, dissipativeness means that the system absorbs supplied energy. Typical examples of dissipative systems are electrical networks where parts of the electrical energy are dissipated, e.g., in resistors in the form of heat.

For the following definition, assume that the dynamical system Σ in (2.65) is given together with a real-valued function w, the ***supply rate*** or ***supply function*** defined on $\mathcal{U} \times \mathcal{Y}$. It represents something like the *power* or *energy* delivered to the system by its external or manifest variables, i.e. input and output. Assume further that

$$\int_{t_1}^{t_2} |w(t)|\,dt < \infty$$

for any $t_1 \leq t_2 \in \mathcal{T}$, $u \in \mathcal{U}$, and $y \in \mathcal{Y}$, where $w(t) := w(u(t), y(t))$, i.e. w is locally integrable.

Definition 2.36. [Ant, Wil72]
A dynamical system Σ with supply function w is ***dissipative*** *if there exists a nonnegative function $S : \mathcal{X} \to \mathbb{R}_{\geq 0}$, the* ***storage function****, such that*

$$S(x_2) - S(x_1) \leq \int_{t_1}^{t_2} w(t)\,dt \tag{2.68}$$

[17] Consider an LTI system with m input channels $u \in \mathcal{U} = \{\, u : \mathcal{T} \to \mathbb{R}^m \,\}$ and p output channels $y \in \mathcal{Y} = \{\, y : \mathcal{T} \to \mathbb{R}^p \,\}$ as a linear operator $\mathcal{S} : \mathcal{U} \to \mathcal{Y}$ mapping the input space to the output space by $y(t) = \int_{\mathcal{T}} h(t-\tau)u(\tau)\,d\tau$. The $p \times m$ matrix-valued function $h : \mathbb{R} \to \mathbb{R}^{p \times m}$ is called the *impulse response* of the system, see, e.g., [**Ant**].

for all $t_2 \geq t_1$, $x_1 \in \mathcal{X}$, and $u \in \mathcal{U}$, where $x_2 = \varphi(t_2, t_1, x_1, u) \in \mathcal{X}$ and $y \in \mathcal{Y}$ is the output of Σ corresponding to u.

The storage function is something like a *generalized energy function* for the dissipative system in question. Thus, the above definition means that the *change in internal storage*, i.e. $S(x_2) - S(x_1)$, cannot exceed the energy that is supplied to the system. The inequality (2.68) is called ***dissipation inequality***. If equality holds for all $t_1 \leq t_2 \in \mathcal{T}$, $u \in \mathcal{U}$, and $x_1 \in \mathcal{X}$, then Σ is called *lossless*.

If one compares the above definition of dissipativity to the definition of internal passivity in Definition 2.35, the similarities are obvious.

2.4.4. Stability. There exists a variety of *stability* definitions for dynamical systems. In general, a *stable* system has the property that its solution trajectories do not change too much under small perturbations. Hence, the most general definition for stability of a dynamical system is in terms of the state trajectories $t \mapsto x(t) = \varphi(t, t_0, x_0, u)$ so that it is applicable to a wide class of systems. Here, u is assumed to be fixed and x_0 is a distinguished initial state at initial time t_0 of the given dynamical system.

In order to prove that a trajectory $y(\cdot)$ is stable, one has to show that if the initial values $y_0 = y(t_0)$, $\overline{y}_0 = \overline{y}(t_0)$ of two solutions $y(\cdot)$, $\overline{y}(\cdot)$ are close to each other, then they remain close to each other for all $t \geq t_0$, $t_0 \in \mathcal{T}$.

Definition 2.37. (Lyapunov stability)
Let $t \mapsto y(t)$ be the solution trajectory of a given dynamical system Σ with initial time $t_0 \in \mathcal{T}$, initial value $y_0 = y(t_0)$, and (fixed) input $u \in \mathcal{U}$. Σ is called ***Lyapunov stable at time $t_0 \in \mathcal{T}$*** *if for all $\varepsilon > 0$ there exists $\delta = \delta(\varepsilon, t_0) > 0$ such that*

$$\|y_0 - \overline{y}_0\| < \delta \qquad \Rightarrow \quad \|y(t) - \overline{y}(t)\| < \varepsilon \quad \text{for all } t \geq t_0$$

for any other solution trajectory $\overline{y}(\cdot)$ of Σ with initial value $\overline{y}_0$.

In the linear case, for the time being, consider the LTI state space equations of an ***ODE***

$$\begin{aligned} \dot{x} &= Ax + Bu, \\ y &= Cx + Du. \end{aligned} \tag{2.69}$$

The following defines stability in terms of the state trajectory $x(\cdot)$.

Definition 2.38. (Stability of state space systems)
Let $t \mapsto x(t) = \varphi(t, t_0, x_0, u)$ be the state trajectory of a given dynamical system Σ in the shape of equation (2.69) with initial time $t_0 \in \mathcal{T}$, initial state $x_0 = x(t_0)$, and (fixed) input $u \in \mathcal{U}$.

- *Σ is called* ***stable*** *if two solution trajectories $x_1(\cdot), x_2(\cdot)$ belonging to the same input u satisfy*

  $$\|x_1(t) - x_2(t)\| < M \qquad \text{for all } t \in \mathcal{T}_+ = [t_0, \infty)$$

  for some constant M.
- *It is called* ***asymptotically stable*** *if in addition*

  $$\lim_{t \to \infty} \|x_1(t) - x_2(t)\| = 0.$$

Then, for the corresponding two outputs y_1, y_2 one has

$$\|y_1(t) - y_2(t)\| = \|C(x_1(t) - x_2(t))\| < M_1 \qquad \text{for all } t \in \mathcal{T}_+$$

in the stable case and, in addition,

$$\lim_{t\to\infty} \|y_1(t) - y_2(t)\| = 0$$

in the asymptotically stable case.

In all the above, $\|\cdot\|$ denotes norms in the corresponding spaces, e.g. the respective Euclidean ones. The above definitions for linear systems mean that the *autonomous* or *zero-input* dynamical part $\dot{x} = Ax$ of (2.69) in the stable case implies the boundedness of $x(\cdot)$ on $\mathcal{T}_+$. In the asymptotically stable case, one additionally has $\|x(t)\|$ tending to zero as t tends to infinity.

The definitions of (asymptotic) stability can also be expressed in terms of the eigenvalues of the system matrices. Considering the autonomous (or homogeneous) part $\dot{x} = Ax$ of system (2.69), note that its solution is given by

$$x(t) = e^{A(t-t_0)}x_0, \qquad x_0 = x(t_0).$$

Proposition 2.39. [Ant, Zer]
The dynamical system (2.69) *is*

- ***asymptotically stable*** $\Leftrightarrow$ $\mathrm{Re}\,(\lambda) < 0$ *for all eigenvalues of A, i.e. for all λ with* $\det(\lambda I - A) = 0$, *and*
- ***stable*** $\Leftrightarrow$ $\mathrm{Re}\,(\lambda) \leq 0$ *for all eigenvalues of A and, moreover, each purely imaginary eigenvalue λ is semi-simple, i.e. its geometric multiplicity equals its algebraic multiplicity.* □

In this case, one also says that ***matrix A is (asymptotically) stable***. Further, this proposition can be extended to linear time-invariant *DAEs*, i.e. systems with dynamical part $E\dot{x} = Ax + Bu$. In that case, one does not consider the eigenvalues of the ***pencil (I, A)***, i.e. the zeroes of $\det(\lambda I - A)$, but the ***generalized eigenvalues of the pencil (E, A)***, i.e. the zeroes of $\det(\lambda E - A)$.[18]

Proposition 2.40. *Let an LTI system of DAEs be given by*

$$\begin{aligned} E\dot{x} &= Ax + Bu, \\ y &= Cx + Du. \end{aligned} \tag{2.70}$$

*This system is **(asymptotically) stable** if the conditions in Proposition 2.39 are satisfied for the matrix pencil (E, A) instead of (I, A).* □

While the above definitions and propositions rely on systems with fixed inputs u and are given in the mathematical background of differential equations, there are also some stability definitions known in engineering and control theory that allow different kinds of inputs u.

[18]Note that $\dot{x} = Ax$ can be expressed by $P(\frac{d}{dt})x = 0$, where $P(s) = sI - A$, and $E\dot{x} = Ax$ by $Q(\frac{d}{dt})x = 0$, where $Q(s) = sE - A$.

Definition 2.41. [Ant, HinPri, Rew]
A (general) dynamical state space system Σ is called ***internally stable*** *if and only if it is* ***zero-input asymptotically stable****, i.e.*

$$u(t) = 0,\ t > t_0 \qquad \Rightarrow \quad x(t) \to 0 \ \text{ for } \ t \to \infty.$$

It is called ***bounded-input, bounded-output (BIBO) stable*** *if any bounded input u results in a bounded output y:*

$$u = (u_1, \ldots, u_m) \in \mathcal{L}^\infty(\mathcal{T}_+, \mathbb{R}^m) \qquad \Rightarrow \quad y = (y_1, \ldots, y_p) \in \mathcal{L}^\infty(\mathcal{T}_+, \mathbb{R}^p).$$

Furthermore, Σ is ***input-output stable*** *or* ***$\mathcal{L}^q$-stable*** *for some $q \in [1, \infty]$ if there exist nonnegative constants β and γ such that*

$$\|y_\tau\|_{\mathcal{L}^q} \le \gamma \|u_\tau\|_{\mathcal{L}^q} + \beta$$

for all $t_0 \le \tau < \infty$ and all input signals $u \in \mathcal{L}^q_{ext} = \mathcal{L}^q_{ext}(\mathcal{T}_+, \mathbb{R}^m)$, where

$$\mathcal{L}^q_{ext} = \left\{ v \,|\, v_\tau \in \mathcal{L}^q \ \forall \tau \ge t_0 \right\} \qquad \text{and} \qquad v_\tau(t) = \begin{cases} v(t), & t_0 \le t \le \tau, \\ 0, & t > \tau \end{cases}$$

is an extension of $\mathcal{L}^q$ and a truncation of v, respectively, and where

$$\|f\|_{\mathcal{L}^q} = \left(\int_{\mathcal{T}_+} \|f(t)\|^q \, dt \right)^{1/q}.$$

Thus, internal stability of an LTI system is exactly asymptotic stability of its autonomous part. The definition of input-output stability means that any input signal $u \in \mathcal{L}^q(\mathcal{T}_+, \mathbb{R}^m)$ is transformed by Σ into an output signal $y \in \mathcal{L}^q(\mathcal{T}_+, \mathbb{R}^p)$, i.e. also the $\mathcal{L}^q$-norm of y is finite which normally does not have to be the case [**HinPri**].

To close this section, we finally cite some properties and connections between asymptotic stability of LTI systems (2.69) and *Lyapunov equations* (see also Subsection 2.5.1.3):

Theorem 2.42. *The following are equivalent:*

(1) *The matrix A is asymptotically stable.*
(2) *For every positive semi-definite Q (written $Q \ge 0$), there exists $P \ge 0$ such that $AP + PA^* + Q = 0$. This equation is called the (continuous-time)* ***Lyapunov equation****.*
(3) *There exists $P \ge 0$ such that $AP + PA^* + I = 0$.* □

Here, A is the system matrix which describes the dynamics or the internal evolution of (the autonomous part of) system (2.69).

Remark 2.43. *If A is asymptotically stable, then the solution P of the Lyapunov equation is uniquely determined by*

$$P = \int_{t_0}^{\infty} e^{A(\tau - t_0)} Q e^{A^*(\tau - t_0)} \, d\tau.$$

Hence, if $Q = BB^*$, then P is the infinite reachability Gramian $\mathcal{P}$ from (2.60) *in Definition 2.30. See also* (2.78).

2.5. Model Order Reduction

The general task of model order reduction (MOR) is to obtain an approximate model of a given large-scale system, which is a lot smaller in size and has a noticeably lower complexity. The model shall still describe the original system's dominant behavior up to a certain accuracy, i.e. the input-output behavior of the original system must be maintained. Further, relevant system properties such as passivity and stability should be preserved. Therefore, a good reduction methodology must be accurate, efficient concerning computational and time costs, numerically robust, and it must generate useful models with respect to stability and passivity.

There have been developed lots of model reduction approaches in a variety of research areas such as electrical and mechanical engineering, control design, or computational fluid dynamics (cf. [**Ant, SchVorRom**], for instance). The by far largest group of MOR algorithms is taylored for linear and, in particular, LTI systems. The most popular classes within this group are methods based on Hankel norm approximants and truncated balancing realizations, on Krylov subspaces, or on sampling methods such as the proper orthogonal decomposition.

Besides these purely numerical MOR techniques, there also exist symbolic ones [**Hen, Wic04**] which are indeed costly to compute, but particularly for nonlinear DAEs arising in electrical circuit design, they additionally allow a deeper insight into the functional relations between the circuit's components. "Symbolic" in this sense means that the system parameters are kept as symbols instead of merely numerical values.

The following subsections provide an overview of the most popular algorithms for model order reduction in their basic versions. Also symbolic methods for both linear and nonlinear systems are reviewed.

2.5.1. Numerical MOR. Numerical MOR techniques can be applied in many different settings. Linear or, more precisely, LTI systems are best investigated and, in a consequence, the largest group of MOR algorithms applies to that kind of dynamical systems. Often they are Laplace transformed and investigated in frequency domain by making use of their transfer function.

Within the last years, there were mainly two categories of approaches. The first one relies on approximating a matrix with one of lower rank by singular value decompositions which leads to so-called *Hankel norm approximants* and *truncated balancing realizations*. These approaches usually lead to very accurate and very small reduced order systems[19]. Furthermore, one has global error bounds and the guarantee of stability and, under certain conditions, passivity preservation. However, they have very high numerical cost, since there are Lyapunov or Riccati equations involved whose solving need $\mathcal{O}(n^3)$ operations, where n is the dimension of the state space. Recent advances on these topics have been made though in order to overcome this problem [**Li, LiWhi, RabPed, SidGri, SorAnt, ZhoLiCaiGuo**].

[19]The order of a system is the dimension of its state space.

The second category of MOR algorithms is based on Krylov subspaces and constructs approximations to the original system's transfer function around specified frequency points. The widest known methods within this category are the algorithms of *Lanczos* and *Arnoldi*. Since the original system matrices are needed only for matrix-vector multiplications, this class of numerically robust algorithms is suitable for MOR of very large-scale dynamical systems and works efficiently at low numerical cost. These facts together with their ease of implementing are mainly responsible for the popularity of Krylov methods, particularly in electrical engineering. Unfortunately, there is also a number of drawbacks: they provide only locally good approximations and lack of global error bounds. Further, there is no guarantee of stability and passivity preservation which makes post-processing necessary. Recent developments, however, showed that under certain circumstances both can be guaranteed [**Ant05, Bai, BaiSloSmiYe, FreFel, Fre04, Gug03, OdaCelPil, Sor**]. There are also some efforts w.r.t. error estimation [**BaiSloSmiYe, Slo**].

There exist further interpolation methods such as vector fitting and tangential interpolation, hamiltonian based concepts, and some hybrid methods. However, these methods will not be treated here. In literature, one can also find structure-preserving MOR methods for linear systems of DAEs.

The above techniques are well-established approaches to generate reduced systems in the linear case. The development of effective and efficient stategies for the reduction of *nonlinear* systems, however, remains challenging and relatively open. Attempts to perform (bi)linearizations or Taylor expansions on a system's nonlinearities generate reduced models of only local validity and, due to computational cost, mostly are limited to quadratic expansions. Therefore, their application is limited to weakly nonlinear systems. *Proper orthogonal decomposition* (POD) and the *trajectory piecewise-linear approach* (TPWL) are MOR techniques that can be applied also to strongly nonlinear systems. While the latter one is based on weighted sums of linearizations along a *training* state-trajectory of the original system, POD – which is popular also for LTI systems – is a sampling method that derives a reduced model from a given set of data, i.e. time snapshots of the original system's state trajectory.

In all of the above methods the state space of the original system is projected onto a lower-dimensional one in order to obtain a more compact system. The bottleneck of these methods is finding the matrices for the projections. It should be noted that the change of states leads to system variables which in general are not physically interpretable anymore. Further, during the reduction process the structure of the system matrices usually is lost by performing the projections. The system structure will be focussed in Chapter 4 when we exploit the hierarchy of the system in order to derive reduced models of coupled systems in the context of electrical circuits.

The following subsections present some of the most popular numerical model order reduction techniques. For further information, we refer to the references given there. For ODEs in state space formulation, a brief overview of some different MOR methods is given, e.g., in [**AntSor**].

2.5.1.1. ***Projection-based Methods.*** Model order reduction based on projection derives simplified models for a given system by projecting the state space of the original system onto a lower dimensional subspace. This concept is the basis for the most widely used approaches for model order reduction of dynamical systems in state space representation. In a certain way, it is similar to a simple truncation in an appropriate basis.

The general concept is not limited to linear systems. Therefore, assume that a dynamical system Σ is given in state space form by

$$\Sigma: \quad \begin{aligned} E\dot{x}(t) &= F(x(t), u(t)), \\ y(t) &= G(x(t), u(t)) \end{aligned} \tag{2.71}$$

with states $x : \mathcal{T} \to \mathbb{R}^n$, inputs $u : \mathcal{T} \to \mathbb{R}^m$, and outputs $y : \mathcal{T} \to \mathbb{R}^p$. The corresponding spaces are denoted by $\mathcal{X}, \mathcal{U}$, and $\mathcal{Y}$. $F : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ and $G : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^p$ in general are smooth nonlinear functions and $E \in \mathbb{R}^{n \times n}$ may be singular. Consider a coordinate transformation

$$\overline{x}(t) = Tx(t) \tag{2.72}$$

with $T \in \mathbb{R}^{n \times n}$ regular. Partitioning

$$\overline{x} = \begin{pmatrix} \widehat{x} \\ \widetilde{x} \end{pmatrix}, \quad T = \begin{bmatrix} W^* \\ T_2^* \end{bmatrix}, \quad T^{-1} = [\, V \; T_1 \,]$$

with $\widehat{x}(t) \in \mathbb{R}^k$, $\widetilde{x}(t) \in \mathbb{R}^{n-k}$, $V, W \in \mathbb{R}^{n \times k}$, and the superscript $*$ denoting the transpose[20], one has $W^*V = I_k$ and, thus, $\Pi = VW^* \in \mathbb{R}^{n \times n}$ in general is an *oblique* projection onto the k-dimensional subspace spanned by the columns of V along the kernel of W^*. If $V = W$, i.e. the columns of V form an orthonormal set of vectors, Π is orthogonal and called a ***Galerkin projection***. Otherwise, if $V \neq W$, Π is called a ***Petrov-Galerkin projection***.

Substituting for x in (2.71) by using the change of coordinates (2.72) and premultiplication of the dynamical part by W^* yields

$$\begin{aligned} W^*E(V\dot{\widehat{x}}(t) + T_1\dot{\widetilde{x}}(t)) &= W^*F(V\widehat{x}(t) + T_1\widetilde{x}(t), u(t)), \\ y(t) &= G(V\widehat{x}(t) + T_1\widetilde{x}(t), u(t)). \end{aligned} \tag{2.73}$$

Note that these equations are still exact. Assuming that $T_1\widetilde{x}$ in (2.73) is "small", its neglect performs the actual approximation. This corresponds to restricting or "truncating" the states x of the original system Σ by performing the projection Π:

$$x = T^{-1}\overline{x} = V\widehat{x} + T_1\widetilde{x} \approx V\widehat{x} = VW^*x = \Pi(x).$$

Thus, the resulting reduced order system $\widehat{\Sigma}$ given by

$$\widehat{\Sigma}: \quad \begin{aligned} \widehat{E}\dot{\widehat{x}}(t) &= W^*F(V\widehat{x}(t), u(t)), \\ \widehat{y}(t) &= G(V\widehat{x}(t), u(t)) \end{aligned} \tag{2.74}$$

with $\widehat{E} = W^*EV$ is a dynamical system that evolves in a subspace of dimension k which normally is chosen a lot smaller than n. Obviously, the approximant $\widehat{\Sigma}$ of Σ is

[20]For complex-valued matrices and vectors, the superscript $*$ denotes the *conjugate* transpose.

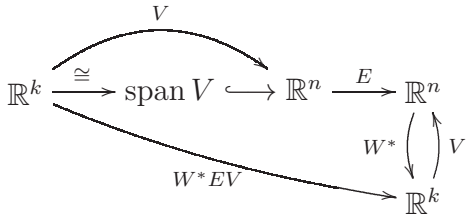


FIGURE 2.9. Diagram showing the projections of E and the states x via $\Pi : \mathbb{R}^n \to \mathbb{R}^k \to \mathbb{R}^n$, $x \mapsto W^*x \mapsto VW^*x$.

"good" only if the influence of the neglected term $T_1\widetilde{x}$ is "small" in an appropriate sense. $\Pi(E) := \widehat{E}$ is called the *projected matrix* of E. The situation is shown in Figure 2.9.

The MOR approaches in the following subsections make use of the projection concept. The determination of the projection matrices V, W is made using Krylov spaces or singular value decompositions of certain matrices.

2.5.1.2. *Singular Value Decomposition (SVD).* The *singular value decomposition* (SVD) is one of the most useful tools in applied linear algebra which can efficiently be computed. First, we will give a definition of the SVD of a matrix.

DEFINITION 2.44. (Singular value decomposition)
For a matrix $M \in \mathbb{C}^{n\times m}$, $n \leq m$, let $\sigma_1 \geq \ldots \geq \sigma_n \geq 0$ be the nonnegative square roots of the eigenvalues of MM^*, where M^* is the conjugate transpose of M. A decomposition

$$M = U\Sigma V^* \tag{2.75}$$

*with $\Sigma \in \mathbb{R}^{n\times m}$, $\Sigma_{ii} = \sigma_i$, $\Sigma_{ij} = 0$ for $i \neq j$, and unitary matrices $U \in \mathbb{C}^{n\times n}$, $V \in \mathbb{C}^{m\times m}$ is called a **singular value decomposition of M**, and $\sigma_1, \ldots, \sigma_n$ are the **singular values of M**.*

Note that the singular values $\sigma_1, \ldots, \sigma_n$ of M are unique. Every matrix with entries in $\mathbb{C}$ has a singular value decomposition, it is unique up to simultaneous multiplication of corresponding columns of U and V by -1, if all σ_i have multiplicity 1.

REMARK 2.45. *The largest singular value of a matrix M is equal to its induced 2-norm:*

$$\|M\|_2 = \sup_{x\neq 0} \frac{\|Mx\|_2}{\|x\|_2} = \sqrt{\lambda_{max}(MM^*)} = \sigma_1(M). \tag{2.76}$$

Hankel norm approximation, which is not treated here, uses the SVD concept to obtain approximate systems of order $k \ll n$ that are optimal w.r.t. the *Hankel norm*. This method uses a direct generalization of the *Schmidt-Mirsky/Eckart-Young* theorem (cf. p. 58 and **[Ant, AntSor]** for further information) to integral operators resulting from LTI systems, namely, the theorem of *Adamjan-Arov-Krein* **[AdaAroKre71, AdaAroKre78]**. However, *balanced truncation* and *proper orthogonal decomposition* are two very popular MOR techniques using SVD that will be explained in a basic version in the following two subsections.

2.5.1.3. ***Balanced Truncation (BT).*** Balanced truncation refers to a whole class of algorithms for SVD-based model order reduction of LTI dynamical systems. For simplicity, we treat here a very basic version and consider a first order ODE given in state space representation by

$$\Sigma: \quad \begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t) + Du(t). \end{aligned} \tag{2.77}$$

In addition, we assume Σ to be ***asymptotically stable***, ***reachable***, and ***observable***, i.e. all the eigenvalues of A have a negative real part and $\operatorname{rank} \mathcal{R}(A, B) = \operatorname{rank} \mathcal{O}(C, A) = n$, where n is the dimension of the state space (cf. the corresponding subsections in Section 2.4).

The reduction of Σ is performed by applying a so-called *balancing transformation* and subsequently truncating the state vector x in an appropriate way. For the reduced system, stability is preserved and an a priori computable error bound is available. In the following, we explain the details.

The concept of *balancing* is strongly connected to the amount of *energy* (cf. Section 2.4.3) that is needed to steer a stable system from the initial state x_0 to a state x and the energy that is produced by observing the output where the initial state is x. Since certain states of the system have to be removed in order to obtain a reduced system, the idea is to remove those states that are *difficult* to reach and those that are *difficult* to observe, i.e. those states that need a lot of *energy* to be reached and yield only little *observing energy*.

However, these concepts depend on the chosen basis. Therefore, states that are difficult to reach in general may not be difficult to observe and vice versa. For that purpose, a basis for the state space is needed in which states that are difficult to reach *simultaneously* are difficult to observe. A transformation satisfying these requirements is called a *balancing transformation*.

States that are difficult to reach have a significant component in the span of the eigenvectors of the *reachability Gramian* $\mathcal{P}$ corresponding to small eigenvalues. The same holds for states which are difficult to observe and the *observability Gramian* $\mathcal{Q}$, where $\mathcal{P}$ and $\mathcal{Q}$ are given by the infinite integrals in (2.60), (2.63). They are the unique solutions of the ***Lyapunov equations*** (see Theorem 2.42 and Remark 2.43)

$$0 = A\mathcal{P} + \mathcal{P}A^* + BB^*, \tag{2.78a}$$

$$0 = A^*\mathcal{Q} + \mathcal{Q}A + C^*C. \tag{2.78b}$$

Note that from these equations it follows immediately that $\mathcal{P}$ and $\mathcal{Q}$ are symmetric (hermitian). Moreover, since Σ is (asymptotically) stable, reachable, and observable, $\mathcal{P}$ and $\mathcal{Q}$ are positive definite (cf. Theorem 2.42 and [**Ant**]). By applying appropriate equivalence transformations

$$\widetilde{\mathcal{P}} = T\mathcal{P}T^*, \qquad \widetilde{\mathcal{Q}} = (T^*)^{-1}\mathcal{Q}T^{-1}$$

to $\mathcal{P}$ and $\mathcal{Q}$, it is achieved that

$$\widetilde{\mathcal{P}} = \widetilde{\mathcal{Q}} = \operatorname{diag}(\sigma_1, \ldots, \sigma_n) \tag{2.79}$$

with $\sigma_i := \sigma_i(\Sigma)$ being the ***Hankel singular values of*** $\boldsymbol{\Sigma}$, i.e. the square roots $\sigma_i = \sqrt{\lambda_i(\widetilde{\mathcal{P}}\widetilde{\mathcal{Q}})} = \sqrt{\lambda_i(\mathcal{P}\mathcal{Q})}$ of the eigenvalues of the product $\mathcal{P}\mathcal{Q}$ of the reachability and observability Gramians[21]. They are input-output invariants of Σ. By the transformations above, it is ensured that states that are difficult to reach *simultaneously* are difficult to observe, since $\widetilde{\mathcal{P}}$ and $\widetilde{\mathcal{Q}}$ are equal. To obtain a suitable transformation matrix T, Cholesky factorizations $\mathcal{P} = UU^*$ and $\mathcal{Q} = LL^*$ and the SVD of $U^*L = ZSY^*$ are computed, where $S = \operatorname{diag}(\sigma_1, \ldots, \sigma_n)$. Then, S is used to define the transformation matrix T by

$$T = S^{\frac{1}{2}} Z^* U^{-1} = S^{-\frac{1}{2}} Y^* L^*.$$

Due to the computation of the SVD of U^*L in order to obtain the Hankel singular values of the system, balanced truncation is called an *SVD-based* model order reduction technique.

Applying the same transformation T to the system matrices in (2.77), one obtains a ***balanced*** version of Σ

$$\Sigma_{bal}: \quad \begin{aligned} \dot{x}(t) &= TAT^{-1}x(t) + TBu(t), \\ y(t) &= CT^{-1}x(t) + Du(t). \end{aligned} \tag{2.80}$$

Clearly, the transformed Gramians $\widetilde{\mathcal{P}}$ and $\widetilde{\mathcal{Q}}$ solve the Lyapunov equations (2.78), where the system matrices A, B, and C have to be replaced by their balanced equivalents $\widehat{A} := TAT^{-1}, \widehat{B} := TB$, and $\widehat{C} := CT^{-1}$. Since the gramians $\widetilde{\mathcal{P}}$ and $\widetilde{\mathcal{Q}}$ are equal and given by $S = \operatorname{diag}(\sigma_1, \ldots, \sigma_n)$, they can be partitioned,

$$\widetilde{\mathcal{P}} = \widetilde{\mathcal{Q}} = S = \begin{bmatrix} S_1 & 0 \\ 0 & S_2 \end{bmatrix},$$

such that, w.l.o.g., S_1 is a $k \times k$ matrix and contains the k largest Hankel singular values σ_1 through σ_k, while S_2 contains the $n-k$ smallest ones. According to this partitioning, the system matrices $\widehat{A}, \widehat{B}, \widehat{C}$ are partitioned as well:

$$\widehat{A} = \begin{bmatrix} \widehat{A}_{11} & \widehat{A}_{12} \\ \widehat{A}_{21} & \widehat{A}_{22} \end{bmatrix}, \quad \widehat{B} = \begin{bmatrix} \widehat{B}_1 \\ \widehat{B}_2 \end{bmatrix}, \quad \widehat{C} = \begin{bmatrix} \widehat{C}_1 \, \widehat{C}_2 \end{bmatrix}.$$

Finally, the system defined by

$$\widehat{\Sigma}: \quad \begin{aligned} \dot{z}(t) &= \widehat{A}_{11} z(t) + \widehat{B}_1 u(t), \\ \widehat{y}(t) &= \widehat{C}_1 z(t) + Du(t) \end{aligned} \tag{2.81}$$

is the ***reduced system of order k*** obtained from Σ by balanced truncation. As well as Σ itself, $\widehat{\Sigma}$ is balanced. Moreover, if the diagonal elements of S_1 are distinct from those of S_2, ***asymptotic stability, reachability, and observability*** of Σ are ***preserved in*** $\widehat{\boldsymbol{\Sigma}}$. Furthermore, in this case the *error system* $\Sigma - \widehat{\Sigma}$ is bounded by twice the sum of the neglected Hankel singular values σ_i [**Ant, AntSor**, and references therein],

$$\sigma_k \leq \|\Sigma - \widehat{\Sigma}\|_{\mathbb{H}_\infty} \leq 2(\sigma_{k+1} + \ldots + \sigma_n). \tag{2.82}$$

The $\mathbb{H}_\infty$-norm in this bound is given by

$$\|\Sigma\|_{\mathbb{H}_\infty} := \|G_\Sigma\|_{\mathbb{H}_\infty} := \sup_{\operatorname{Re}(s)>0} \|G_\Sigma(s)\|_2 = \sup_{\omega \in \mathbb{R}} \|G_\Sigma(i\omega)\|_2,$$

[21] Here, $\lambda_k(M)$ denotes the k-th largest (in magnitude) eigenvalue of M.

where $G_\Sigma := C(sI - A)^{-1}B + D$ is the transfer function of Σ given by system matrices A, B, C, and D as in (2.77). In other words, the $\mathbb{H}_\infty$-norm describes the maximum of the highest peak of the system's frequency response, i.e. the largest singular value of G_Σ evaluated on the imaginary axis.

Remark 2.46. *Usually, the Hankel singular values decay rapidly such that the upper bound in* (2.82) *guarantees good approximations even for very small systems* $\widehat{\Sigma}$ *of order* $k \ll n$. *A study on the decay rate of Hankel singular values can be found in* [**Ant**, Chapter 9].

2.5.1.4. ***Proper Orthogonal Decomposition (POD).*** The model order reduction method of proper orthogonal decomposition (POD) is an application of the SVD in order to approximate *general dynamical systems*. This application flexibility relies on the fact that the reduced system is derived from a given *set of data*, i.e. measurements of the state trajectory at certain points of time. For nonlinear systems, the generation of these *time snapshots* might be costly due to the required simulation of the original nonlinear system. Furthermore, in both cases of linear and nonlinear systems the choice of suitable inputs becomes involved, since the time snapshots clearly depend on the input-dependent simulation data.

In this subsection, we will consider state space systems of the form given by (2.71). The data set usually is a collection of time snapshots of the resulting state trajectory $x(\cdot)$ w.r.t. a certain input function. The snapshots are then arranged in a matrix in order to compute an SVD. A low-dimensional subspace for the approximation of a (usually large) data set then is provided by taking into account only those contributions that correspond to the largest singular values.

Assume that the states $x(t)$ of the dynamical system (2.71) live in $\mathbb{R}^n$. The goal is to approximate this state trajectory by one that lives in a lower-dimensional space $\mathbb{R}^k$ with $k \ll n$. Assume further that for a *fixed* input $u(t)$ applied to the system N *snapshots* of the trajectory $x(\cdot)$ are given, i.e. *samples* or *measurements* of $x(\cdot)$ at N distinct instances $t_1, \ldots, t_N$ of time. Arranging them in a matrix X yields a ***snapshot matrix***

$$X = [x_1 \ \ldots \ x_N] := [x(t_1) \ \ldots \ x(t_N)] \in \mathbb{R}^{n \times N}. \tag{2.83}$$

Note that usually $N \gg n$. In a next step, a singular value decomposition $X = USV^*$, $S = \operatorname{diag}(\sigma_1, \ldots, \sigma_n) \in \mathbb{R}^{n \times N}$ with $\sigma_1 \geq \ldots \geq \sigma_n \geq 0$, is computed. From this a low-order approximation of the snapshot matrix X can be derived, if the singular values decay rapidly – which usually is the case – and only the k largest ones contribute significantly. This becomes more obvious if one writes out the *dyadic decomposition*, i.e. a sum of n outer products of rank 1

$$\begin{aligned} X \ = USV^* \ &= \sigma_1 u_1 v_1^* + \ldots + \sigma_k u_k v_k^* + \ldots + \sigma_n u_n v_n^* \\ &\approx \sigma_1 u_1 v_1^* + \ldots + \sigma_k u_k v_k^* \qquad\qquad = U_k S_k V_k^*, \end{aligned}$$

where $U = (u_1, \ldots, u_n) \in \mathbb{R}^{n \times n}$ and $V = (v_1, \ldots, v_N) \in \mathbb{R}^{N \times N}$ are unitary as in Definition 2.44, where U_k and V_k consist of the first k columns of U and V, respectively, and where $S_k = \operatorname{diag}(\sigma_1, \ldots, \sigma_k) \in \mathbb{R}^{k \times k}$. Depending on the decay of the singular values, usually $k \ll n$.

By defining $U_k S_k V_k^* =: \widetilde{X} = [\widetilde{x}_1 \ \dots \ \widetilde{x}_N] \in \mathbb{R}^{n\times N}$, the matrix $\widetilde{X}$ of the truncated elements $\widetilde{x}_i$ approximates the original data set X optimally in the sense that the 2-induced norm $\|X - M\|_2$ for matrices M with $\operatorname{rank} M \leq k$ is minimized by $\widetilde{X}$. This is exactly the theorem of ***Schmidt-Mirsky/Eckart-Young*** **[EckYou, Mir, SchE, SteSun]** which states that

$$\min_{\operatorname{rank} M \leq k} \|X - M\|_2 = \sigma_{k+1}(X)$$

if $\sigma_k(X) > \sigma_{k+1}(X)$ and with the notation given here. Therefore, the original data $x_i = \sum_{j=1}^n \gamma_{ji} u_j$ with $(\gamma_{ji}) := \Gamma = SV^* \in \mathbb{R}^{n\times N}$ are approximated *optimally* in this sense by the truncated elements $\widetilde{x}_i = \sum_{j=1}^k \gamma_{ji} u_j$.

The final step to derive a reduced system is that of a projection. Since $U_k^* U_k = I_k$, a Galerkin projection $\Pi = U_k U_k^*$ is defined and, thus, one obtains a simplification of the original system (2.71)

$$\widehat{\Sigma}: \quad \begin{aligned} \widehat{E}\dot{z}(t) &= U_k^* F(U_k z(t), u(t)), \\ \widehat{y}(t) &= G(U_k z(t), u(t)), \end{aligned} \tag{2.84}$$

where $\widehat{E} = U_k^* E U_k$ and where the trajectory $z = U_k^* x : \mathcal{T} \to \mathbb{R}^k$ evolves in a low-dimensional subspace of $\mathcal{X}$ spanned by the k leading columns of U.

To close this section, we add some

Remarks 2.47. [Ant, Her]

- *In general, for a matrix $A \in \mathbb{R}^{n\times m}$, $n \leq m$, its* ***proper orthogonal decomposition*** *is given by a decomposition*

$$A = [\, a_1 \ \dots \ a_m \,] = [\, u_1 \ \dots \ u_n \,] \cdot \begin{bmatrix} \gamma_{11} & \cdots & \gamma_{1m} \\ \vdots & \ddots & \vdots \\ \gamma_{n1} & \cdots & \gamma_{nm} \end{bmatrix},$$

  where $U = [u_1 \ \dots \ u_n] \in \mathbb{R}^{n\times n}$ is a set of orthonormal vectors u_j, the ***principal directions****, and $\Gamma = (\gamma_{ij}) \in \mathbb{R}^{n\times m}$ is a coefficient matrix.*
- *Since the resulting simplified model $\widehat{\Sigma}$ depends on the input function applied to the system and the time instances at which the snapshots are taken, the resulting singular values are no system invariants. Hence, one does not obtain an approximation of the original system, but an approximation of the original system together with its applied external input. Further phenomena of the system which are not captured by the snapshots cannot be represented by the reduced system. Hence, especially at time instances where the dynamics of the system change rapidly, the sampling rate for the snapshots should be increased.*
- *A general problem of POD methods for MOR is the determination of the reduced model's quality, i.e. the question how well the reduced model approximates trajectories other than the measured one. However, empirically the reduced system yields good approximations also for other inputs.*

2.5.1.5. ***Krylov Subspace Methods.*** In SVD-based approaches such as the balanced truncation methods or Hankel norm approximations the Gramians $\mathcal{P}, \mathcal{Q}$ are involved which are solutions to the Lyapunov equations (2.78). Therefore, these methods

require dense computations of order n^3 (cf. Section 2.5.1.6), where n is the state space dimension of the original system. Hence, in their *basic* versions these methods are applicable to systems of only moderate complexity.

However, alternatives are given by MOR approaches using concepts based on Krylov subspaces. These *Krylov methods* provide iterative algorithms for the computation of appropriate projection matrices in order to project a system's state space onto a lower-dimensional subspace. Although system properties such as passivity and stability in general are not preserved by Krylov methods, those approaches are most widely used for the reduction of LTI systems. This is mainly due to significant savings in computational effort and their applicability to large-scale systems.

A ***Krylov space*** of order i is defined by an $n \times n$-matrix M and an n-dimensional vector v as follows:

$$\mathcal{K}_i(M, v) = \operatorname{span}\operatorname{col}[\, v \quad Mv \quad M^2v \quad \dots \quad M^{i-1}v \,]. \tag{2.85}$$

In the following, we will not distinguish between the space $\mathcal{K}_i(M, v)$ and the matrix on the right hand side. If M is sparse, it can quickly be multiplied to v, hence a basis for $\mathcal{K}_i(M, v)$ can quickly be computed. A *generalized* form uses an $n \times m$-matrix V instead of v, where its m columns are interpreted as vectors. Hence, the ***generalized Krylov space*** is given by

$$\mathcal{K}_i(M, V) = \operatorname{span}\operatorname{col}[\, v_1 \quad Mv_1 \ \dots \ M^{i-1}v_1 \quad \dots \quad v_m \quad Mv_m \ \dots \ M^{i-1}v_m \,], \tag{2.86}$$

where v_j are the columns of V. Krylov spaces are well known in the numerical linear algebra community. In the control systems community, however, $\mathcal{K}_n(A, B)$ and $\mathcal{K}_n(A^*, C^*)^*$ are known as *reachability* and *observability subspaces* $\mathcal{R}(A, B)$ and $\mathcal{O}(C, A)$ (cf. Sections 2.4.1 and 2.4.2). The origin of Krylov methods is found in iterative computations of eigenvectors and eigenvalues of matrices, but there are also applications to the iterative solving of matrix equations and to iterative approximation of linear dynamical systems by matching *moments* of their transfer functions. In the following, we will focus on the latter topic.

Assume a linear time-invariant system Σ is given in state space form by

$$\Sigma : \begin{aligned} E\dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t) + Du(t), \end{aligned} \tag{2.87}$$

where $E \in \mathbb{R}^{n\times n}$ may be singular. The remaining system matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, and $D \in \mathbb{R}^{p\times m}$ are as usual. Since Σ is uniquely determined by its *impulse response* h or, equivalently, by its transfer function H which is the Laplace transform

$$H(s) = (\mathcal{L}h)(s) = \int_0^\infty h(t)e^{-st}dt, \qquad s \in \mathbb{C},$$

of the impulse response, a possibility to approximate system (2.87) is the approximation of its transfer function. This can be achieved by matching a number of *moments*, i.e. coefficients, of the Laurent series expansion of H at some points in the complex plane. Thus, an *interpolation* of the original system's transfer function is aspired.

Recall that if the matrix pencil $\lambda E - A$ is regular, i.e. $\det(\lambda E - A) \not\equiv 0$, the transfer function of Σ is given by the $p \times m$ matrix-valued function H in the independent complex

frequency variable s

$$H(s) = D + C\big(sE - A\big)^{-1}B,$$

where each of the entries in H is a rational function in $s \in \mathbb{C}$. If E is regular, i.e. (2.87) is an *ODE*, expanding H around infinity by applying the *Neumann expansion*, one obtains

$$\begin{aligned} H(s) \;=\; & D + C(E^{-1}B)\cdot s^{-1} + C(E^{-1}A)(E^{-1}B)\cdot s^{-2} \\ & + C(E^{-1}A)^2(E^{-1}B)\cdot s^{-3} + \ldots . \end{aligned} \tag{2.88}$$

The moments $m_0 = D$, $m_j = C(E^{-1}A)^{j-1}(E^{-1}B)$, $j > 0$, are called *Markov parameters*. Then an approximation $\widetilde{H}(s) = \sum_{j\geq 0} \widetilde{m}_j \; s^{-j}$ of H is aspired such that its first k moments $\widetilde{m}_j$, $j = 0, \ldots, k-1$, match the first k ones of H. This problem is known as the *partial realization problem* [**Ant, Kal**].

If H is expanded in $s_0 \in \mathbb{C}$, $\det(s_0E - A) \neq 0$, one obtains

$$\begin{aligned} H(s) = D & - C(A - s_0E)^{-1}B \\ & - C(A - s_0E)^{-1}E(A - s_0E)^{-1}B\cdot(s - s_0) \\ & - C\big((A - s_0E)^{-1}E\big)^2(A - s_0E)^{-1}B\cdot(s - s_0)^2 + \ldots \end{aligned} \tag{2.89}$$

with *shifted moments* at the expansion point s_0 given by

$$\begin{aligned} m_0 &= D - C(A - s_0E)^{-1}B, \\ m_j &= -C\big((A - s_0E)^{-1}E\big)^j(A - s_0E)^{-1}B, \quad j > 0. \end{aligned}$$

Again an approximation is searched for whose first k moments match those ones of (2.89). If $s_0 = 0$, this is known as the *Padé approximation problem* [**BakGra**], while for a general value of s_0 this is the problem of *rational interpolation* [**AndAnt**].

In general, one can also construct reduced-order models whose transfer functions match the original one at multiple interpolation points $s_0, s_1, \ldots, s_l$. The corresponding approximation is called *multi-point rational interpolant* [**GalGriDoo, Gri, LasWil**].

In our case here, the described problems can be solved numerically stable and efficient by applying ***Arnoldi*** or ***Lanczos methods*** which will be explained subsequently. The big advantage of these methods is that it is guaranteed that the resulting reduced system's transfer function has the same moments as the one of the original system up to a certain number without having to compute these moments *explicitly*. Hence, the matrix inversions and factorizations involved in explicit moment computations are avoided[22]. To be more precise, if one considers the span of the first k moments of the original system's transfer function H expanded around $s_0 \neq \infty$ and premultiplied by C^{-1} (D assumed to be zero), i.e. the Krylov space

$$\mathcal{K}_k\big(A_0^{-1}E, A_0^{-1}B\big) = \big[\, A_0^{-1}B \quad A_0^{-1}EA_0^{-1}B \quad \ldots \quad (A_0^{-1}E)^{k-1}A_0^{-1}B \,\big]$$

with $A_0 := A - s_0E$, then the orthonormalization of $\mathcal{K}_k(A_0^{-1}E, A_0^{-1}B)$ yields a matrix $V \in \mathbb{R}^{n\times mk}$ such that for $mk \leq n$ a Galerkin projection $\Pi = VV^*$, $V^*V = I_{mk}$, in $\mathbb{R}^n$ is

[22] Note that in general the computation of the moments is numerically problematic, since the powers of eigenvalues of A grow exponentially fast.

Input: matrix $M \in \mathbb{R}^{n\times n}$, vector $b \in \mathbb{R}^n$, and order k of a Krylov space $\mathcal{K}_k\big(M, b\big)$
Output: $V \in \mathbb{R}^{n\times k}$, $f \in \mathbb{R}^n$, and $H \in \mathbb{R}^{k\times k}$ such that

$$MV = VH + fe_k^*,$$
$$\text{with} \quad H = V^*MV, \quad V^*V = I_k, \quad V^*f = 0,$$

where e_k is the k-th unit vector and where H is in upper Hessenberg form.

$v_1 = \frac{b}{\|b\|}$
$w = Mv_1$
$\alpha_1 = v_1^* w$
$f_1 = w - \alpha_1 v_1$
$V_1 = [v_1]$
$H_1 = [\alpha_1]$
for $j = 1, \ldots, k-1$ **do**
  $\beta_j = \|f_j\|$
  $v_{j+1} = \frac{f_j}{\beta_j}$
  $V_{j+1} = [V_j, v_{j+1}]$
  $\widehat{H}_j = \begin{bmatrix} H_j \\ \beta_j e_j^* \end{bmatrix}$
  $w = Mv_{j+1}$
  $h = V_{j+1}^* w$
  $f_{j+1} = w - V_{j+1}h$
  $H_{j+1} = [\widehat{H}_j, h]$
end

Algorithm 2.48: The Arnoldi algorithm.

given. This projection then is used to derive a reduced system $\widehat{\Sigma}$ defined by

$$\widehat{\Sigma}: \quad \begin{aligned} \widehat{E}\dot{z}(t) &= \widehat{A}z(t) + \widehat{B}u(t), \\ \widehat{y}(t) &= \widehat{C}z(t) + Du(t), \end{aligned} \tag{2.90}$$

where $\widehat{E} = V^*EV$, $\widehat{A} = V^*AV$, $\widehat{B} = V^*B$, $\widehat{C} = CV$, and where the states $z = V^*x$ evolve in $\mathbb{R}^{mk}$. The *crucial fact* now is that the first k moments $\widehat{m}_j$ of the reduced system's transfer function $\widehat{H} = \widehat{C}(s\widehat{E} - \widehat{A})^{-1}\widehat{B} + D$ match those of H [**Gri**]:

$$\widehat{m}_j = m_j, \qquad 0 \le j \le k-1.$$

The computation of V can be carried out efficiently and in an iterative way by the ***Arnoldi algorithm*** (cf. Algorithm 2.48) or *Arnoldi-type methods*, depending on whether one deals with single-input single-output (SISO) or multi-input multi-output (MIMO) systems. Similarly, considering the two Krylov spaces

$$\mathcal{K}_k\big(A_0^{-1}E, A_0^{-1}B\big) = [\, A_0^{-1}B \quad A_0^{-1}EA_0^{-1}B \quad \ldots \quad (A_0^{-1}E)^{k-1}A_0^{-1}B \,],$$
$$\mathcal{K}_k\big(A_0^{-*}E^*, A_0^{-*}C^*\big) = [\, A_0^{-*}C^* \quad A_0^{-*}E^*A_0^{-*}C^* \quad \ldots \quad (A_0^{-*}E^*)^{k-1}A_0^{-*}C^* \,]$$

with $A_0 = A - s_0 E$ and the superscript $-*$ denoting the transposed inverse, the ***two-sided Lanczos iteration*** or *Lanczos-type methods* yield two *biorthogonal* matrices $V, W \in \mathbb{R}^{n \times mk}$, $W^*V = I_{mk}$, such that a Petrov-Galerkin projection $\Pi = VW^*$ is given. Applying this projection to (2.87), one obtains a reduced system

$$\widetilde{\Sigma}: \quad \begin{aligned} W^*EV\dot{z}(t) &= W^*AVz(t) + W^*Bu(t), \\ \widetilde{y}(t) &= CVz(t) + Du(t), \end{aligned} \tag{2.91}$$

whose transfer function has the same moments as H not only up to the first k, but even up to the first $2k$ ones:

$$\widetilde{m}_j = m_j, \qquad 0 \leq j \leq 2k-1.$$

The same numbers of matched moments are achieved if one considers the *Markov parameters* of the corresponding transfer functions, i.e. the coefficients of the Laurent series expansion at $s_0 = \infty$. In that case, recall that E has to be regular. Then one has to consider the Krylov spaces

$$K_k(E^{-1}A, E^{-1}B) = [\, E^{-1}B \quad E^{-1}AE^{-1}B \quad \ldots \quad (E^{-1}A)^{k-1}E^{-1}B \,]$$

for Arnoldi-type methods or

$$\begin{aligned} K_k(E^{-1}A, E^{-1}B) &= [\, E^{-1}B \quad E^{-1}AE^{-1}B \quad \ldots \quad (E^{-1}A)^{k-1}E^{-1}B \,] \\ K_k(E^{-*}A^*, E^{-*}C^*) &= [\, E^{-*}C^* \quad E^{-*}A^*E^{-*}C^* \quad \ldots \quad (E^{-*}A^*)^{k-1}E^{-*}C^* \,] \end{aligned}$$

for Lanczos-type iterations. More information about Lanczos- and Arnoldi-type methods and algorithms can be found in [**Bai, FelFre, Fre03, FreFel, GalGriDoo, Gri, OdaCelPil**].

For reasons of simplicity, the *Arnoldi* and *two-sided Lanczos algorithms* are described here in Algorithms 2.48 and 2.50 only for SISO systems, i.e. for $m = p = 1$. Since in that case the system matrices B, C actually are *vectors* in $\mathbb{R}^n$, the above algorithms are formulated involving *vectors* b, c^*.

In the Arnoldi iteration, which is mainly a modified Gram-Schmidt orthonormalization, one takes advantage of the fact that from step to step only the last column and the entry $(k, k-1)$ of the projected matrix $H_k = V_k^* M V_k$ change. Note that for the two-sided Lanczos Algorithm 2.50 one has

$$MV_k = V_k T_k + \beta_{k+1} v_{k+1} e_k^*, \qquad M^*W_k = W_k T_k^* + \gamma_{k+1} w_{k+1} e_k^*,$$

where $V_k = (v_1 \ \ldots \ v_k)$, $W_k = (w_1 \ \ldots \ w_k)$, and

$$T_k = \begin{bmatrix} \alpha_1 & \gamma_2 & & \\ \beta_2 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \gamma_k \\ & & \beta_k & \alpha_k \end{bmatrix}.$$

Remark 2.49. *The two-sided Lanczos algorithm is a modification of the original Lanczos procedure which is designed only for symmetric matrices M. Thus, the two-sided one is more general.*

Input: matrix $M \in \mathbb{R}^{n\times n}$, vectors $b, c^* \in \mathbb{R}^n$, and order k of Krylov spaces $\mathcal{K}_k\big(M,b\big)$ and $\mathcal{K}_k\big(M,c\big)$

Output: $V_k, W_k \in \mathbb{R}^{n\times k}$, $f_k, g_k \in \mathbb{R}^n$, and $T_k \in \mathbb{R}^{k\times k}$ such that

$$MV_k = V_kT_k + f_ke_k^*, \quad M^*W_k = W_kT_k^* + g_ke_k^* \quad \text{with}$$
$$T_k = W_k^*MV_k, \quad V_k^*W_k = I_k, \quad W_k^*f_k = 0, \quad V_k^*g_k = 0.$$

$\beta_1 = \sqrt{\|b^*c^*\|}$
$\gamma_1 = \operatorname{sign}(b^*c^*)\beta_1$
$v_1 = b/\beta_1$
$w_1 = c^*/\gamma_1$
for $j = 1, \ldots, k$ **do**
  $\alpha_j = w_j^*Mv_j$
  $r_j = Mv_j - \alpha_jv_j - \gamma_jv_{j-1}$
  $q_j = M^*w_j - \alpha_jw_j - \beta_jw_{j-1}$
  $\beta_{j+1} = \sqrt{\|r_j^*q_j\|}$
  $\gamma_{j+1} = \operatorname{sign}(r_j^*q_j)\beta_{j+1}$
  $v_{j+1} = r_j/\beta_{j+1}$
  $w_{j+1} = q_j/\gamma_{j+1}$
end

Algorithm 2.50: The two-sided Lanczos algorithm.

Furthermore, for symmetric matrices M the matrices H_j in the Arnoldi iteration are tridiagonal and the Arnoldi algorithm coincides with the Lanczos one [**Ant**, Remark 10.4.2].

There exists a variety of improved, modified, or generalized methods based on Krylov spaces such as *rational Krylov methods*, *implicitly restarted versions* of the Arnoldi or Lanczos procedures, algorithms for *multiple interpolation points*, or *rational interpolation* [**Ant, GalGriDoo, Gri, LasWil**]. Krylov-based moment-matching methods can efficiently be applied to large-scale problems, the dimension of the state space can be up to millions. On the other hand, the reduced system yields a good approximation only locally around the expansion point(s) s_0 respectively $s_0, \ldots, s_l$, whose optimal choice is an open problem. Furthermore, no global error estimate is known and system properties such as stability and passivity in general are not preserved, which usually makes post-processing necessary to realize these properties. However, recent advances on these topics have been made for standard state space systems and also for structured generalized ones arising in circuit simulation [**Ant05, Bai, BaiSloSmiYe, FreFel, Fre04, Gug03, OdaCelPil, Sor**].

2.5.1.6. ***SVD-Krylov Methods.*** SVD-Krylov methods aim at the development of approximation methods that combine the advantages of both the Krylov space and SVD-based concepts.

While the SVD-based model order reduction methods in the previous subsections preserve system properties such as stability and – under certain conditions – passivity and

provide global error bounds for the error system $\Sigma - \widehat{\Sigma}$, reduced systems obtained by Krylov methods might be unstable[23]. Global error bounds are in general also not available for the latter ones and Lanczos or Arnoldi methods might break down under certain conditions on the reachability and observability matrices of the corresponding system.

Nevertheless, Krylov methods mostly are preferred which is mainly due to the significant savings in computation effort; while solving the Lyapunov equations involved in SVD-based reduction methods needs $\mathcal{O}(n^3)$ operations and, therefore, the corresponding algorithms are applicable to systems with only a few hundred states[24], the computational cost for Lanczos- or Arnoldi-type iterations amounts to $\mathcal{O}(k^2n)$ respectively $\mathcal{O}(kn^2)$ operations, depending on whether one deals with ***sparse***[25] or *dense* systems[26]. Thus, Krylov methods can be applied also to large-scale systems, where n is up to millions. Moreover, matrix factorizations and inversions necessary in SVD-based approaches are avoided, since Krylov-based methods involve only matrix-vector multiplications. In addition, there is no need to compute the transformed n-th order balanced model in order to subsequently perform a truncation of the state vector.

For the reasons explained above, there are approaches available – *model reduction by least squares* [**Ant**], for example – that combine (some of) the advantages of both the Krylov- and SVD-based methods. This is achieved mainly by exploiting certain connections between the two approximation concepts. For example, the (generalized) reachability and observability Gramians can be obtained by solving the *Sylvester equations*, a more general form of the Lyapunov equations (2.78). Another example is the use of iterative methods to solve the Lyapunov equations in an approximate way, which yields *approximately balancing transformations*. Hence, one obtains reduced systems by *approximately balanced truncation*.

However, we abstain from further describing these methods and refer to [**Ant**, Chapter 12] instead.

2.5.1.7. ***The Trajectory Piecewise-Linear (TPWL) Approach.*** The trajectory piecewise-linear (TPWL) approach [**Rew**] is a model order reduction method which is preferentially applicable to *nonlinear* dynamical systems. It was firstly developed in [**RewWhi01**] by Rewieński and White (see also [**RewWhi02, RewWhi03**]) and mainly targets at MOR of systems arising from electrical circuits. The main idea of this technique relies on the linearization of a nonlinear dynamical system of first order at multiple points along the state trajectory that corresponds to a *training input* u. The linearized models then are reduced by appropriate MOR methods for linear systems to finally obtain a reduced approximation of the original nonlinear system by a weighted

[23] A remedy to this are so-called *implicitly restarted* Krylov methods. See, e.g., [**Ant**].

[24] However, recently there have been made advances in the development of methods to overcome this problem by using approximate Gramian computations [**Li, LiWhi, RabPed, SidGri, SorAnt, ZhoLiCaiGuo**].

[25] A system is called ***sparse*** if the system matrices A, E are sparse. A matrix is said to be sparse if *many* of its entries are zero. A common definition is: an $n \times n$ matrix M is sparse if the number of its non-zero entries is $\mathcal{O}(n)$.

[26] Memory costs in both cases are of the order $\mathcal{O}(kn)$ [**Ant**].

sum of all the reduced linearized models. In the following, the procedure is explained more detailed.

Since there exists a variety of reduction methods for linear and, in particular, LTI systems (see the preceding sections), the basic idea behind the TPWL approach is to take advantage of these well known approaches in order to reduce a given nonlinear system. Since linearizations yield good approximations only locally around the linearization point, multiple linearizations about suitably selected states are necessary. Hence, a first step in the TPWL approach is the choice of appropriate points on the state trajectory $x(\cdot)$ and the derivation of the corresponding linearized systems. In a next step, the projection bases V, W for the reduction of these linearized systems have to be constructed and applied. The *TPWL model* of the original system finally is a convex combination of the linearized reduced systems. Subsequently, this subsection follows the notes of [**Rew**].

Consider a nonlinear system given by

$$\Sigma: \quad \begin{aligned} \tfrac{d}{dt} g(x) &= f(x) + B(x)u, \\ y &= Cx, \end{aligned} \tag{2.92}$$

where $x \in \mathcal{X} = \{\, x : \mathcal{T} \to \mathbb{R}^n \,\}$ are the states, $f, g : \mathbb{R}^n \to \mathbb{R}^n$ are continuous nonlinear functions to describe the system's dynamics, $B = B(x) \in \mathbb{R}^{n \times m}$ is a state-dependent input mapping, $u \in \mathcal{U} = \{\, u : \mathcal{T} \to \mathbb{R}^m \,\}$ the *fixed* input, $C \in \mathbb{R}^{p \times n}$ the output mapping, and $y \in \mathcal{Y} = \{\, y : \mathcal{T} \to \mathbb{R}^p \,\}$ the output of Σ. The output equation, i.e. the second equation in (2.92), does not change during the following procedures – except for replacing x by a subspace approximation – so we only consider the dynamical part of Σ.

For the time being, assume that s linearization points $x_i := x(t_i)$ are chosen, where $i = 0, \ldots, s-1$. With the setup as above, the first order approximations of f and g, i.e. their linearizations, in states x_i are then given by

$$\begin{aligned} \widetilde{f}_i(x) &= f_i + F_i(x - x_i), \\ \widetilde{g}_i(x) &= g_i + G_i(x - x_i), \end{aligned} \tag{2.93}$$

where $f_i = f(x_i)$, $g_i = g(x_i)$, further $F_i = D_x f|_{x_i}$ and $G_i = D_x g|_{x_i}$ denote the Jacobians of f and g evaluated at $x = x_i$, and where $x_0 = x(t_0)$ is the initial state of Σ. Substituting the linearizations in the dynamical part of Σ for each i, one obtains a set of linearized systems given by

$$\Sigma_i: \quad \tfrac{d}{dt}\big(g_i + G_i(x - x_i)\big) = f_i + F_i(x - x_i) + B_i u \tag{2.94}$$

with $B_i = B(x_i)$. Then, an approximation of the original system Σ is constructed from these linearizations by a weighted combination

$$\frac{d}{dt}\Bigg(\sum_{i=0}^{s-1} \overline{w}_i(x)\big(g_i + G_i(x - x_i)\big)\Bigg) = \sum_{i=0}^{s-1} \overline{w}_i(x)\big(f_i + F_i(x - x_i) + B_i u\big), \tag{2.95}$$

where the weights $\overline{w}_i$ are used to *switch* the linearized models Σ_i on and off depending on the linearization point x_i to which the current state x is closest. The weights are considered real-valued nonnegative and satisfying the condition $\sum_{i=0}^{s-1} \overline{w}_i(x(t)) = 1$ for all $t \in \mathcal{T}$. Therefore, this combination is *convex*.

The next step is the reduction of the systems Σ_i for all i by applying a suitable MOR method for linear systems[27]. Assuming that $n \times k$ projection matrices V, W have already been computed, a reduced version of (2.95) is given by

$$\frac{d}{dt}\Big(\sum_{i=0}^{s-1} \overline{w}_i(Vz)\big(W^* g_i + W^* G_i(Vz - x_i)\big)\Big) = \sum_{i=0}^{s-1} \overline{w}_i(Vz)\big(W^* f_i + W^* F_i(Vz - x_i) + W^* B_i u\big). \tag{2.96}$$

In order to further reduce the computational cost of evaluating weight functions that still depend on n-dimensional states, one may take a new set of weight functions w_i that depend solely on the reduced states $z = W^* x$ instead. This yields the reduced order model

$$\begin{aligned} \frac{d}{dt}\Big(\Big(\sum_{i=0}^{s-1} w_i(z)\widehat{G}_i\Big)z + \gamma\Big) &= \Big(\sum_{i=0}^{s-1} w_i(z)\widehat{F}_i\Big)z + \varphi + \Big(\sum_{i=0}^{s-1} w_i(z)\widehat{B}_i\Big)u, \\ \widehat{y} &= \widehat{C}z, \end{aligned} \tag{2.97}$$

where

$$\begin{aligned} \gamma &= W^*(g_0 - G_0 x_0, \ldots, g_{s-1} - G_{s-1} x_{s-1})(w_0(z), \ldots, w_{s-1}(z))^*, \\ \varphi &= W^*(f_0 - F_0 x_0, \ldots, f_{s-1} - F_{s-1} x_{s-1})(w_0(z), \ldots, w_{s-1}(z))^*, \\ \widehat{G}_i &= W^* G_i V, \quad \widehat{F}_i = W^* F_i V, \quad \widehat{B}_i = W^* B_i, \quad \widehat{C} = CV, \end{aligned}$$

and such that

$$\sum_{i=0}^{s-1} w_i(z(t)) = 1, \qquad w_i(z(t)) \geq 0 \qquad \forall t \in \mathcal{T}\,.$$

System (2.97) is called the ***TPWL model*** of Σ, it is subsequently denoted by $\widehat{\Sigma}$. Note that this system is still *nonlinear*. The nonlinearities are introduced by the state-dependent weight functions w_i. It is of great importance to have the evaluation of w_i in z computationally efficient and, simultaneously, the reduced system (2.97) providing a good approximation of Σ. Hence, typically, only one or two of the weigths w_i should be non-zero at a time, depending on the dominant linearized reduced system $\widehat{\Sigma}_j$ corresponding to *that* state $z_j = W^* x_j$ which is nearest to the *current* state $z = W^* x$. Weight functions that proved to work effectively in [**Rew**] are given by

$$w_i(z) = \widetilde{w}_i(z)/S(z), \qquad i = 0, \ldots, s-1,$$

where $\widetilde{w}_i = e^{-\beta d_i / m_d}$, $S(z) = \sum_{j=0}^{s-1} \widetilde{w}_j(z)$, $d_i = \|z - z_i\|_2$, and m_d is the minimum of all the d_i. The larger the (positive) constant β, the faster the weights w_i change from ≈ 0 to ≈ 1 as z_i becomes the state closest to z, and back to ≈ 0, if another state z_k, $k \neq i$, is closer to z. This motivates the term *piecewise-linear* in the TPWL approach, although, in a strict sense, $\widehat{\Sigma}$ is *not* piecewise-linear due to the weights w_i still being smooth functions. For smaller β, the slopes of the weights are rather moderate[28].

[27] In [**Rew**], the use of Krylov methods and, in particular, the Arnoldi algorithm is proposed, but also SVD-based methods such as balanced truncation are in consideration.

[28] A value of $\beta = 25$ proved to be adequate in [**Rew**].

It is neither answered yet how the linearization points x_i are selected, nor how projection bases V, W are constructed. For this, we first add some remarks: Note that each model Σ_i in (2.94) and the reduced equivalents $\widehat{\Sigma}$ provide an adequate approximation of Σ only in a certain neighborhood around x_i, i.e. for $\|x - x_i\| < \varepsilon = \varepsilon(x_i)$. To avoid the very inefficient or infeasible covering of the entire state space by such small balls[29], it is instead "covered with models" only along the ***training trajectory*** $x(\cdot)$ corresponding to the fixed ***training input*** $u(\cdot)$. To achieve this, a single simulation of the original system Σ is performed in which the linearization points x_i as well as the corresponding linearized models Σ_i are extracted as follows:

(1) linearize Σ in the initial state $x_0 = x(t_0)$, initialize $i = 0$
(2) simulate Σ while x is close enough to the already extracted states x_i, i.e. while
$$\min_{0 \le j \le i} \left(\frac{\|x - x_j\|}{\|x_j\|} \right) < \delta$$
for some appropriate $\delta > 0$, $x_j \neq 0$
(3) construct a new linearized model about $x_{i+1} := x$ and set $i := i + 1$
(4) if $i < s - 1$ return to step 2

For the construction of the projection matrices V, W there is a number of possibilities. They can be generated by taking into account only the linearized model in the *initial state* x_0, or, alternatively, a union of different bases corresponding to subsequent linearized models at the linearization points x_i can be set up. Clearly, the latter yields more accurate results at an additional cost of generating multiple projection bases. Since the simpler case reduces to the application of a reduction method for linear systems to Σ_0, we focus on the second one. Further, in the first case, which is very low-cost, TPWL models of only low quality are expected for general nonlinear systems Σ. For the second case, however, assuming that Krylov methods are used to generate V, W, the first few moments of the corresponding transfer functions are matched *for all* i, which clearly yields a more promising TPWL model for Σ. Exemplarily applying the *Arnoldi method*, V is constructed as follows:

(1) initialize $i = 0$, $V_{\text{agg}} = [\,]$
(2) repeat until the *training simulation* is completed:
- compute the linearized system Σ_i corresponding to x_i and by using the Arnoldi process with a suitable k_i construct the two projection matrices V_1, V_2 such that they span the Krylov spaces
$$\mathcal{K}_{k_i}(F_i^{-1}G_i, F_i^{-1}B_i), \qquad \mathcal{K}_{k_i}(F_i^{-1}G_i, F_i^{-1}(f_i - F_i x_i))$$
- set $V_{\text{agg}} = [\,V_{\text{agg}}, V_1, V_2, x_i\,]$

(3) orthogonalize V_{agg} using, e.g., SVD and delete all columns that correspond to singular values $\sigma < \varepsilon$ for some appropriate $\varepsilon > 0$.

The remaining columns in V_{agg} are set to be the projection basis V. Note also that the term $f_i - F_i x_i$ in Σ_i (cf. equation (2.94)) is treated as an additional input, hence there are *two* Krylov spaces computed internally.

[29] The number of such balls grows exponentially with the dimension of the state space.

The above algorithms require a simulation of the original nonlinear system Σ that might be costly. A more efficient technique that avoids the computation is also mentioned in [**Rew**]. It uses the reduced linearized systems instead of the full order nonlinear one for an approximate simulation and extracts all the necessary models and linearization points *on the fly* during this process. For further information we refer to [**Rew**].

With the above TPWL approach, only the parts of the state space along the *training trajectory* are "covered with models". It is assumed that for systems whose state trajectories corresponding to certain inputs u lie within a certain region around the *training* trajectory, the constructed TPWL model $\widehat{\Sigma}$ in (2.97) will adequately approximate their input-output behavior. This means that although these trajectories stay close to the linearization points $x_i \approx Vz_i$, $i = 0, \ldots, s-1$, the corresponding input signals may be very different from the training input with respect to dynamics or frequency.

In the general case, *stability* of the original nonlinear system is not preserved for the reduced TPWL model. However, under certain conditions, there are *stability-preserving weights* computable for the reduced TPWL model. For details see [**Rew**, Chapter 5]. Moreover, under some further conditions, *passivity preservation* of the reduced TPWL model can be derived from the stability analysis therein.

Since in the general nonlinear case there is no frequency domain description in terms of transfer functions available, the time-domain responses of the original and reduced order systems are compared to each other by computing $\|x(t) - Vz(t)\|_2$. Under certain conditions, *a posteriori error bounds* at discrete time steps are available [**Rew**, Chapter 4]. Of course, it is assumed thereby that the involved trajectories do not behave pathologically between two such time steps. The former algorithms for deriving the TPWL model are further extended by computing the error bounds *on the fly* and taking them into account for the choice of subsequent linearization points. Under certain conditions, even a global *a priori error bound* is available that holds for all times $t \geq t_0$, where t_0 is the initial time. However, the practical importance of this bound is rather limited, since it will typically be very conservative.

After providing this overview of popular numerical MOR techniques, in the next section we discuss the symbolic case.

2.5.2. Symbolic MOR. By *symbolic model order reduction*, a whole family of hybrid symbolic and numerical algorithms for simplifying a system of equations involving symbolic parameters is referred to. The main task of these methods is the derivation of an approximate model for the original one that is more compact and has a decreased level of complexity. Due to the system parameters given as symbols instead of numerical values – such as "R_L" for a load resistor value or a transistor parameter "β_F" – in the context of electrical circuits the approximate model allows insights into the functional dependences among the circuit components and an interpretation of the obtained formulas in a physical sense.

Note that insights and interpretability of the original symbolic system typically are not given, since computer-aided analysis involving all exact physical effects usually yields very large expressions and formulas. Hence, symbolic approximation techniques have

to detect which ones of the parameters and terms in these expressions are dominant and neglect the remaining ones. This is achieved by manipulations on the system and subsequent comparisons to a *reference solution* of the original system which is computed via numerical reference values assigned to the symbolic parameters.

The general workflow in both the linear and nonlinear case is as follows [**HalWic03, SomHalBro, Wic04**]: Usually one has a circuit netlist description that is translated into a symbolic system of equations F via standard graph theoretical methods like MNA or STA from Section 2.1. Thus, F describes the circuit behavior mathematically. According to one or several inputs u, design points π, and a numerical analysis $\mathcal{A}$ such as AC, DC, or transient analysis, *reference solutions* $y_F = \mathcal{A}(F, u, \pi)$ for the output of F are computed[30]. Depending on the analysis task, the reference solution could be given as numerical transfer function, its poles and zeros, or as solution in time domain. Further, the user provides an error bound ε and an *error function*[31] E (usually some kind of seminorm) to guarantee a certain accuracy of the simplified system. The complexity of F then is reduced by iteratively applying symbolic reduction techniques such as removing terms from the equations and comparing the numerical solution $y_G = \mathcal{A}(G, u, \pi)$ of the so far reduced system G to the reference solutions y_F. As long as the error $E(y_F, y_G)$ is within the given bound ε, the performed reduction step is accepted, otherwise it is rejected.

Thus, the original system F finally is reduced to a *simpler* form with less equations, less terms, less derivatives, and so on [**Wic04**]. The above algorithm assures that the simplified system is a good approximation in the sense that its numerical behavior coincides with that one of the original system within the user-given error bound for a given numerical analysis task. The simpler form still contains the dominant parameters in symbolic form such that one can read off the influences of the dominant system parameters on certain circuit characteristics. This further allows the identification of those parameters that have to be altered in order to meet certain design specifications. Moreover, the simplified system represents a behavioral model for F which can also be translated into a hardware description language (HDL) and then simulated using a circuit simulator.

In order to obtain a simplified system with minimal complexity, a *ranking*, i.e. an optimized order of terms to be simplified, is computed (see Section 4.3.1). Since the number of terms that potentially can be simplified is large, the ranking is a trade-off between accuracy and computational efficiency that predicts the influence of a term simplification on the output of the system. Thus, those terms with the least influence on the output behavior are simplified first, while those term simplifications with a large influence are avoided.

Symbolic reduction methods are found to be very effective particularly for component-based systems such as electrical circuits. Usually these systems provide very accurate

[30] The reference solutions are given by pairs of sampling points and interpolation values, hence the accuracy in comparison to the *exact* mathematical solution y depends on the numerical solver's choice of the step size.

[31] See Definition 4.6 in Section 4.6.

equations for all their components, e.g. modelling all possible operating domains of semiconductor devices such as transistors or diodes. However, in general not all those domains have to be taken into account due to the topology and the input signals of the circuit at hand which keep a certain instance of the semiconductor device only in a limited number of its operating domains. Therefore, the corresponding terms can be simplified.

Symbolic reduction methods are able to detect the significant terms of the describing equations F *automatically*. Since the symbolic system parameters are replaced by appropriate numerical values corresponding to π in order to compute the reference solutions, they are a hybrid combination of numerical and symbolic algorithms. The following subsections present symbolic reduction methods for both linear and nonlinear systems. They are implemented in *Analog Insydes* [**AI**], an add-on for the computer-algebra system *Mathematica* [**MMA**]. *Analog Insydes* is developed by the Fraunhofer ITWM in Kaiserslautern, Germany. We mainly follow the approach of [**Hen, Wic04**], a brief overview of symbolic methods in industrial analog circuit design can be found in [**HalWic03**].

2.5.2.1. ***Symbolic Approximation Methods for Linear Systems.*** In linear symbolic analysis, the linear dynamical system usually is Laplace transformed into a system in frequency domain with the independent complex Laplace variable $s \in \mathbb{C}$. The main object of interest then is the system's transfer function. By evaluating it on the imaginary axis $s = i\omega$, the system's *frequency response* to a sinusoidal input signal with constant frequency is investigated. Using a *Bode diagram* (see, e.g., Figure 1.2 on page 7), the frequency response can be depicted in terms of magnitude and phase, where the frequency f, $\omega = 2\pi f$, is swept over the frequency domain.

Since even for small circuits the exact computation of a system's *symbolic* transfer function quickly leads to exhaustive expressions, in order to obtain meaningful and compact symbolic expressions by computer, symbolic analysis programs must be able to automatically perform certain approximations. Depending on whether they are applied before, during, or after the symbolic transfer function is computed, they are categorized in *simplification before, during,* and *after generation* methods, SBG, SDG, and SAG, respectively [**ChaMcKayWie, DroSomHor, FerRodHue, GieSan, HenTweSom, Kol, RodEA, SedDegFic, SomHenDroHor, WalGieSan, Wam**]. While SBG methods simplify the matrix equations describing the system's behavior *before* its transfer function is computed, SDG methods perform simplifications *during* its computation and SAG methods directly simplify the transfer function itself. In the following, we restrict ourselves to describing only SAG and SBG methods.

The aim of applying SBG methods is the elimination of as much insignificant information from a circuit analysis problem as possible before any symbolic operations are performed. The first fully automated SBG procedure for matrix-based symbolic analyzers is due to Dröge [**DroSomHor, SomHenDroHor**]. Its basic idea is the identification and elimination of insignificant terms in the sum-of-products[32] form of each system matrix entry to achieve complexity reductions. Note that these simplifications describe only

[32]In MNA contributions of components to the system typically are of the form $A^T \cdot f(Bx)$, where A and B are incidence matrices describing the circuit topology and f is a function describing the behavior of the corresponding component.

partial deletions, since no circuit components are actually removed from the circuit netlist. In the following, assume that the linear system is Laplace transformed into frequency domain and given by the linear system of equations (cf. Definition 2.26, page 39)

$$\Sigma: \; A(s,p)x(s) = b(u(s),s,p), \qquad s \in \mathbb{C}, \tag{2.98}$$

in the complex Laplace frequency variable s. Thereby, p is the vector of symbolic parameters and, by abuse of notation, $x(s)$ and $u(s)$ denote the Laplace transforms of the system variables $x(t)$, $x \in \mathcal{X} = \{\, x : \mathcal{T} \to \mathbb{R}^n \,\}$, and the input $u(t)$, $u \in \mathcal{U} = \{\, u : \mathcal{T} \to \mathbb{R}^m \,\}$. The output y as usual is a linear combination of the system variables x and is not important for the following considerations.

In [**Hen**], Dröge's algorithm is stepwise improved. In a first step, the term in question to be deleted from a matrix entry is set *temporarily* to zero. The solution $\widetilde{y}$ of the perturbed system is computed and compared to the reference solution y at certain frequency points. If the *nominal errors* in phase and magnitude,

$$\varepsilon_{N,\mathrm{abs}} = \left| \frac{\|\widetilde{y}\| - \|y\|}{\|y\|} \right|, \qquad \varepsilon_{N,\arg} = |\arg \widetilde{y} - \arg y|,$$

are small enough, the corresponding term is deleted from the system matrix. Otherwise or if the system becomes singular, the term is left unchanged. Then the procedure is repeated with the next term candidate.

In order to cancel as many terms as possible and to terminate as soon as the provided error bound ε is violated, those terms that have the least contribution should be removed first. Therefore, in a second step, the order of term cancellations is optimized by computing a *term ranking* (cf. Section 4.3.1). This is done by using a low-cost procedure to *estimate* the solution $\widehat{y}$ of the system which is perturbed by the removal of exactly one of all possible terms ϑ from the original system and an afterward ordering of the resulting list of terms by their nominal errors, i.e. their influence on the reference solution y.

In a third step, the matrix is *compressed* in order to remove redundant rows and columns that might have been created by the cancellation of appropriate matrix entries. Moreover, the system of circuit equations typically contains a large amount of unnecessary information, since lots of variables usually are not needed for the computation of the output y. This dispensable information can be detected by a graph search algorithm and then be deleted, which leads to typical reductions in matrix size between 50% and 90% [**Hen**, Section 2.6].

From such a reduced system, an approximation of the original symbolic transfer function can be computed which is reduced w.r.t. complexity and its polynomial order in s. This expression can further be reduced by SAG methods [**HalWic03**]. Techniques developed for computer-aided symbolic analysis due to Walscharts, Gielen, and Sansen [**GieSan, WalGieSan**] manipulate the transfer function by removing terms that cause only negligible deviations. More precisely, assume that the symbolic transfer function in the single-input single-output (SISO) case is given as a rational expression

$$H(s,p) = \frac{\sum a_i(p)s^i}{\sum b_i(p)s^i}, \tag{2.99}$$

where $a_i(p) = \sum_j a_{ij}(p)$ and $b_i(p) = \sum_j b_{ij}(p)$ are functions in the symbolic parameters p in sum-of-products form. Then those coefficients a_{ij}, b_{ij} are removed from (2.99) that cause only a small error not exceeding a given error bound. Thus, the complexity of the symbolic transfer function can further be reduced drastically, however, without any reductions on the degree w.r.t. s.

2.5.2.2. ***Symbolic Approximation Methods for Nonlinear Systems.*** As we will see in this subsection, the basic ideas of symbolic simplification methods for linear systems can be transferred to the nonlinear case. However, one cannot generally expect to obtain explicit and interpretable formulas for the output as in the linear case, but the presented methods allow an automated generation of behavioral models of reduced complexity and with a user-specified accuracy for the use in large circuits, e.g. to speed up their numerical simulation. The research in this area is relatively young and still in progress [**Bor98, PopHarHedBar, WicEA, Wic01**].

Starting from a system of DAEs F, symbolic simplification techniques are used to obtain a new system G of lower complexity whose input-output behavior is approximately the same as the one of F. Since the interpretation of *complexity* depends on the particular application example, we abstain from giving an explicit definition. Intuitive criteria for the complexity of a system are its number of equations and variables, the number of derivatives or nonlinear terms, the number of summands in the equations, or combinations thereof. Nevertheless, the reduction of a system F only makes sense if the obtained system G has lower complexity.

In order to guarantee that the input-output behavior of G is a good approximation to the one of F, the error $E(y_F, y_G)$ is checked after each reduction step, where G is the so far reduced system, y_F is a reference solution, and y_G is the corresponding numerical solution of G. As long as the user-given error bound ε is not exceeded, the corresponding reductions are considered to be valid.

In the following, four symbolic model order reduction techniques for nonlinear systems are reviewed [**Wic04**]. These methods are taylored for the complexity reduction of component-based systems such as electrical circuits, gas pipelines, and mechatronical systems[33].

Algebraic Manipulations. The first technique is a purely algebraic manipulation and, hence, an *exact* reduction method. Certain variables are eliminated and substituted in the remaining set of equations. Additionally, independent blocks of equations are removed. As an example, consider the symbolic system of DAEs in time domain $F(x, \dot{x}, y, t; u, p) = 0$ with symbolic parameters p, input u, output y, and internal variables $x = (x_1, \ldots, x_n)$. Assume that one of the equations in F can be solved for x_1, then F can be simplified by replacing x_1 in the corresponding terms in the remaining

[33]The analogies to currents, voltages, and the Kirchhoff laws in electrical circuits are given by forces, displacement, and the law of d'Alembert in mechanics, respectively. In a gas pipeline network the appropriate magnitudes are mass flows and pressures, while the Kirchhoff current law is substituted by the law of mass conservation at each node.

equations and by dropping the equation that defines x_1:

$$F: \begin{pmatrix} \cdots \\ 0 = x_1 - f(\overline{x}_1; p) \\ 0 = g(x; p) \\ \cdots \end{pmatrix} \quad \Rightarrow \quad G: \begin{pmatrix} \cdots \\ 0 = g(f(\overline{x}_1), \overline{x}_1; p) \\ \cdots \end{pmatrix},$$

where $\overline{x}_1 = (x_2, \ldots, x_n)$. Obviously, this method is mathematically exact. Nevertheless, one has to be careful, since its application may lead to equations including a huge number of terms. If for instance in the above example $f(\overline{x}_1; p)$ is a sum of a large number of terms and x_1 occurs in g with a high exponent, then an even higher number of terms is produced by substituting x_1 with $f(\overline{x}_1; p)$ in g.

Branch Reductions. The second mathematically exact reduction method detects and subsequently removes unused branches of piecewise-defined functions which occur, e.g., in the device model equations of transistors in order to model their different operating regions. Depending on the input signals, usually these components work in only a single operating region. Therefore, the branches modelling unused operating regions may be neglected:

$$f(x_i; p) = \begin{cases} f_1(x_i; p), & x_i < a \\ f_2(x_i; p), & a \leq x_i \leq b \\ f_3(x_i; p), & x_i > b \end{cases} \quad \Rightarrow \quad f(x_i; p) = f_2(x_i; p) \text{ for all } x_i.$$

In this case, f is to be considered as a function contained in at least one term in at least one of the equations of the system of DAEs F.

Term Reductions. Written in sum-of-products formulation, the system of DAEs F consists of a set of equations each containing a large number of terms. But the number of those that significantly contribute to the sum of a single equation often is small and the remaining terms, therefore, may be neglected, i.e. deleted from the equation. The following example, where the j-th equation F_j in F is a sum of terms ϑ_i and where the k-th term ϑ_k is cancelled, shows the principle of this technique:

$$F_j: \sum_{i=1}^{N_j} \vartheta_i(x; p) = 0 \quad \Rightarrow \quad G_j: \sum_{i=1, i \neq k}^{N_j} \vartheta_i(\widetilde{x}; p) = 0.$$

Applying term cancellations thus alters the solution x of the original system F to the solution $\widetilde{x}$ of the so far reduced system G. Therefore, term reductions are *approximating reduction methods*. As long as the accumulated error caused by a cancellation does not exceed the given error bound ε, the reduction is considered to be valid.

Note that this technique can also be applied to different *levels* in F, i.e. when a term contains a function whose arguments themselves are sums-of-products, term reductions can be applied to the terms of the arguments and so on.

Term Substitutions. Similar to the last reduction technique, *term substitutions* replace appropriate terms of the equations in F by adequate constants, often certain average values. Hence, *term reductions* can be considered as a special case of *term substitutions*, since cancelling a term from the equations is the same as replacing it by

0. Like *term reductions*, this technique can be applied to different levels in the system of equations as well.

In the following example, the k-th term ϑ_k in the j-th equation F_j of F is replaced by the constant κ:

$$F_j : \sum_{i=1}^{N_j} \vartheta_i(x;p) = 0 \qquad \Rightarrow \quad G_j : \sum_{i=1,\, i\neq k}^{N_j} \vartheta_i(\widetilde{x};p) + \kappa = 0.$$

Obviously, this method also is an *approximating reduction technique*, since it alters the solution x of the original set of equations.

The general reduction workflow has to be considered on two levels. On the first level, a certain symbolic reduction technique is chosen from the four methods above[34]. Once this has been done, one has a variety of possible *internal* reduction possibilities. For example, if *term reductions* are chosen, one usually has a large set of terms in the equations that are candidates to be deleted. Hence, on this second level, by the choice of a certain term, the corresponding *internal reduction* is performed.

Remark 2.51. *It turned out that the order of the applied symbolic reduction techniques influences the degree of reduction in dependence on the system that is to be investigated. Further influencing factors, of course, are the applied input u used during the reduction, the error function E, and the user-specified maximum error ε. Therefore, the general reduction workflow could not be completely automated in a satisfying way. While the internal reductions in the second level are completely automated, the choice of the reduction technique in the first level is made by the user.*

The order of term cancellations and replacements plays a crucial role: If a term is cancelled or replaced that causes a rather big error w.r.t. the reference solution y_F, the tolerance for further reductions is rather small. In order to cancel or replace such terms first that cause only a small error, a *term ranking* estimates the influences of the single term reductions and substitutions and orders them accordingly. Then in each *internal reduction step* the term with currently the smallest ranking value in the list is reduced.

Further, in order to avoid the time-consuming numerical analyses associated with error checks after each reduction step, a *term clustering* has been invented in [**Wic04**]. By clustering the terms according to their ranking values, the set of all terms is sub-divided into bundles of terms with more or less the same influence[35] on the reference solution y_F. Then all the terms in an entire cluster are simplified simultaneously and the costly numerical error check is performed only once afterwards. If the accumulated error caused by this violates the error bound ε, the term simplifications in the entire cluster are rejected and the cluster is sub-divided. Then the procedure is repeated with the new clustering.

[34] It turned out that one can obtain a high degree of reduction by using solely algebraic manipulations and term reductions.

[35] For example, one could take the ranking values' logarithms as a criterion for the clustering.

The above procedures finally yield a complexity reduced symbolic behavioral model for the real physical system at hand. In order to guarantee good numerical solvability, the *index* of the corresponding system of equations is monitored during the reduction process. Although there is a variety of different index concepts available (cf. Section 2.2.2), a common property is that the numerical solving of systems with an index higher than 1 is an ill-posed problem. Since symbolic simplifications of the equations may increase the index, the monitoring observes possible changes.

CHAPTER 3

Hierarchical Systems

An increasing problem particularly for symbolic model reduction is the continuously growing size of systems to be analysed. Together with their sizes, also the *complexity* of these systems increases very quickly[1]. According to *Moore's Law* (1975), the number of transistors on a chip – which he took as a measure for complexity – doubles every two years. While in 1971 the Intel 4004 processor incorporated 2,300 transistors with dimensions of $\sim 10\,\mu$m, nowadays one has feature sizes in the Intel Core 2 Extreme (quad-core) of ~ 45 nm with 820 million transistors arranged in 9 layers (cf. Figure 3.1 and Table 3.1).

Promising approaches to overcome the complexity problem for analysis and model order reduction of large systems and, in particular, electrical circuits are their segmentation in smaller subsystems (*macros*) and a coupling part. Hence, the smaller parts of lower complexity can be processed separately taking advantage of reduced resources and increased efficiency in computation time. Finally, after their processing, the resulting subsystems are "re-coupled" using the same coupling structure.

This approach seems just natural, since an electrical circuit has a *hierarchical structure* of components and devices that are coupled by an interconnecting network (see also Figure 2.1, page 12). As they are composed of various circuit components themselves, devices such as amplifiers can be considered as *subcircuits* and can be further subdivided, thus yielding a hierarchical structure. If standard graph theoretical methods like MNA or STA are used to set up the describing system of equations, the hierarchy available on

processor	technology	release	number of transistors
Intel 4004	$10\,\mu$m	1971	2,300
Intel 8086	$3\,\mu$m	1978	29,000
Intel 386	$1\,\mu$m	1985	275,000
Intel 486	800 nm	1989	1,200,000
Intel Pentium P4	180 nm	2000	42,000,000
Intel Core 2 Extreme (quad-core)	45 nm	2007	820,000,000 arranged in 9 layers

TABLE 3.1. Development of Intel processors from 1971 up to today [**wikipedia**].

[1]The complexity of a system shall not be further specified here, one should think of a system's complexity rather intuitively.

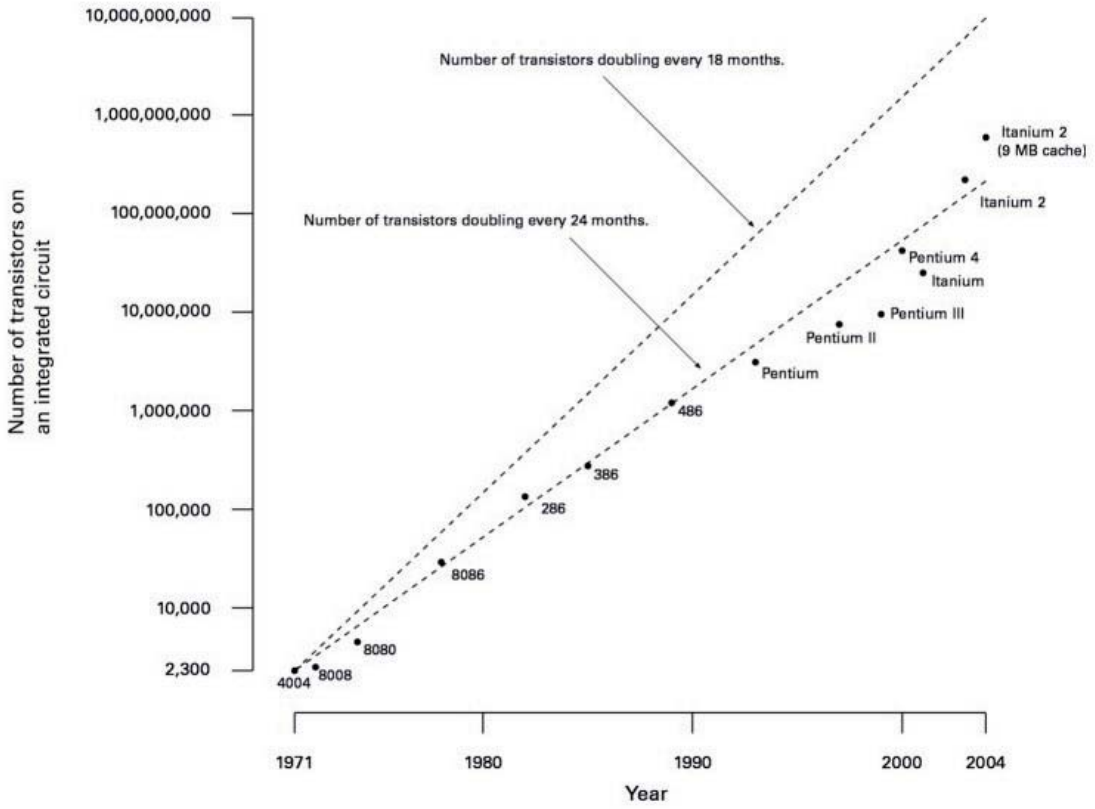


FIGURE 3.1. Moore's Law and the development of Intel CPUs (source: `http://upload.wikimedia.org/wikipedia/commons/a/a9/Moore_Law_diagram_(2004).jpg`)

circuit level is lost. Consequently, we aim at the exploitation of the circuit structure, i.e. its hierarchy.

The first section explains the modelling of component-based systems such as electrical circuits as a coupling network of subsystems. In the second section, interconnected and coupled systems are defined in a physical sense. Some examples and some references for structure-preserving MOR of interconnected LTI systems in an input-output framework are reviewed. The third section presents the behavioral approach that uses the concept of shared variables to model mutual interactions among subsystems. It is further compared to the systems-theoretical framework using inputs, outputs, and output-to-input assignments to model subsystem interactions. In order to show the differences between the two concepts, a differential amplifier is briefly investigated. Section four finally deals with the macromodel concept and provides an idea for MOR of large electrical circuits exploiting their hierarchical structure on circuit level.

3.1. Component-based Systems

A *component-based system* is a real physical system which consists of subsystems (components) that are coupled by a connecting network. As already mentioned in Section

2.1, an electrical circuit is considered as a network of subsystems coupled by a certain connecting structure. While the subsystems correspond to the circuit's components, the connecting structure is given by the topology of the circuit graph.

The modelling of such systems is done on two different levels, i.e. the modelling of the single subsystems and the modelling of the entire system. The single subsystems are modelled very precisely including all possible physical effects and operating domains, since their respective importance for the behavior of the entire system is not yet known. Therefore, parameterized systems of equations are used that describe the subsystems' exact input-output behavior for all possible applications and uses.

The actually relevant effects and operating domains are defined by the excitation and the coupling topology of the entire system. This is done in the second modelling level, the modelling of the entire system. In the context of electrical circuits, the resulting systems of equations consist of the model equations for the single subsystems and the Kirchhoff equations (2.1). Together with the applied inputs, this finally determines the subsystems' actual operating domains and physical effects of importance for the behavior of the entire system. Hence, the dynamical behavior of the entire system is determined only locally by the modelling of the single subsystems, the Kirchhoff equations only model the connecting structure.

3.2. Coupled and Interconnected Systems

A general definition of coupled or interconnected systems could be given as follows:

> *Two or more systems are* ***coupled*** *or* ***interconnected****, if they are interacting with each other. This does not necessarily mean that they have* ***physical*** *connections to each other, but one can consider them to be* ***abstractly connected*** *thus causing them to* ***operate as a unit****. Hence, some "information" from one of them is provided to and influences the behavior of the others. The "information" is passed among the systems by their* ***interconnections*** **[Wil97, Wil07, wikipedia]**.

The modelling of complex technical or physical systems such as electrical circuits yields coupled systems in a natural way as they usually are composed of different *building blocks* such as current mirrors, amplifying stages, polarization circuits, transmission lines, etc. physically connected to each other. In Section 3.3 and in the next chapter, we consider a *differential amplifier* composed of several building blocks which can be considered as systems themselves. The mathematical models corresponding to the building blocks or *subsystems* usually consist of ordinary differential equations, differential-algebraic equations, or partial differential equations. By interconnecting the *real* system's substructures among each other, these sets of equations also are coupled in a corresponding way.

Further examples for interconnected systems can be found in the design of *micro electromechanical systems (MEMS)* or *very large system integrated (VLSI) chips*, for instance **[Bec, FelParFar, Reis06, Tis]**. The acceleration sensor **[SomHalBro]** in Figure 3.2 is a MEMS which combines components of an electrical circuit with mechanical parts. The sensor includes three metal plates that form a series connection of two capacitors, where

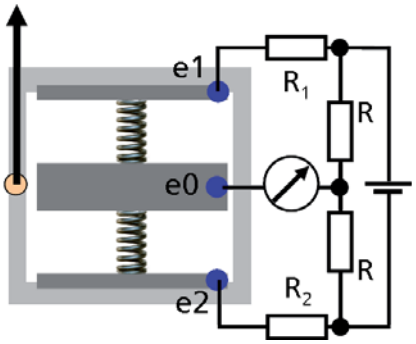


FIGURE 3.2.
The acceleration sensor combines components of electrical circuits with mechanical parts.

the middle plate is locomotive. Therefore, in case of acceleration the plate is moved due to inertia thus changing the capacitances of the two capacitors. This results in a measurable voltage drop from which the acceleration can be read off.

In [**VanVDoo**], model order reduction of interconnected linear systems has been studied. The systems considered there are composed of k subsystems given by linear MIMO transfer functions, their interconnections are captured by feedback loops and output-to-input assignments. More precisely, the *internal input* u_j to the j-th subsystem is a "linear combination" with *matrix coefficients* K_{ij}, H_j of the subsystems' *internal outputs* y_k and the *external input* u_{ext}, while the *external output* y_{ext} of the entire system is a "linear combination" with *matrix coefficients* R_j of the subsystems' *internal outputs*:

$$\begin{aligned} u_j &= K_{j1}y_1 + \ldots + K_{js}y_s + H_j u_{ext}, \qquad j = 1, \ldots, s, \\ y_{ext} &= R_1 y_1 + \ldots + R_s y_s, \end{aligned} \tag{3.1}$$

where s is the number of subsystems. Then, structure-preserving reduction methods based on SVD and moment-matching are presented. Structure preservation in that sense means the preservation of the block structure of the system matrices. This corresponds to separate reductions of the involved subsystems and their subsequent coupling in accordance with the original interconnecting structure given by (3.1).[2]

In [**ReiSty08**], coupled LTI control systems are considered which are coupled by the same mechanism as in [**VanVDoo**] above. Moment-matching approximation and balanced truncation methods then are used to reduce the order of the *closed-loop systems*. The results are compared to structure-preserving versions of the same reduction methods and some error bounds based on the $\mathbb{H}_\infty$-norm of the corresponding transfer functions are given.

Although the linear case is best investigated for dynamical systems, structure-preserving reduced-order modelling of coupled systems has received attention only recently. In [**Fre08, LiBai, ReiSty07, VilSchSil**] we found some further structure-preserving approaches which are based on Krylov space approximations with a certain shape of the projecting matrices in order to preserve the structure of the original system in its reduced equivalent. However, these techniques can only be applied to *linear* systems.

[2]However, note that the task is the approximation of the global mapping from u_{ext} to y_{ext} and not the approximation of the internal mappings from u_j to y_j.

Nevertheless, in general we are dealing with *nonlinear hierarchical* systems which are much less studied, but of great importance. Of course, there exist methods for model order reduction of *nonlinear* systems, however, we could not find any nonlinear approaches that explicitly exploit the *hierarchy*, i.e. the structure of the overall system.

3.3. The Behavioral Approach

Classic systems and control theoretic approaches use inputs, outputs, and signal-flow graphs for a mathematical description of a system's behavior. While the inputs are used to model the influences of the environment on the system, the outputs serve to model the system's influence on its environment. Output-to-input assignments such as feedback loops are used to model interconnections and interactions between its subsystems.

But the physics of the *real* system merely relate its variables to each other, interconnections between substructures in the *real* system merely mean the "sharing" of the corresponding involved variables. Hence, in general it does not make sense to define which of the variables should be viewed as inputs and which as outputs. When the subsystems become part of an interconnecting network, it is the interconnecting structure that determines which of the variables at the interconnections act as inputs and which as outputs. As an example, consider the port behavior of an RLC circuit. Since this network can be viewed both as current- or voltage-controlled, it is not endowed with a natural signal flow. Modelling it via input-output formulations, however, introduces a signal flow structure which is not present in the real physical system.

In contrast to this, by taking the *behavioral approach* [**Wil97, Wil07**], i.e. "variable sharing" and relations on system variables, as a mathematical model for the *real* physical system, all system variables are treated equally and none are specially tagged as inputs or outputs of certain subsystems. Consequently, the corresponding *behavioral models* can be employed in higher generality. As an example, consider the hydraulic interconnected system that consists of two barrels of water interconnected by a pipe. The physical interconnection just means that the pressures at the "link terminal" of the two barrels are set equal, while the corresponding flow variables add up to zero[3]. Hence, the two subsystems, i.e. the barrels, share the pressure and flow variables at their "link terminals" as soon as they are interconnected to each other by the pipe. One also could consider the whole setup as an interconnection of *three* subsystems, where the third one is the pipe itself. In this case, one has two interconnections, the interconnection between the first barrel and the pipe, and the one between the second barrel and the pipe. The corresponding pressure and flow variables in both cases then are shared among the participating subsystems. Physically, there is no signal flow from one of the subsystems to the other one by declaring some pressure or flow variables of one of the subsystems as its output and, simultaneously, the input to another one.

Nevertheless, input-output considerations are not ill-founded in general. For example, the reactions of humans or animals to stimuli from their environment, the contraction of an eye's iris blinded by light, or the response to any kind of external command can

[3]The respective flows are assumed to be directed towards the interior of the barrels.

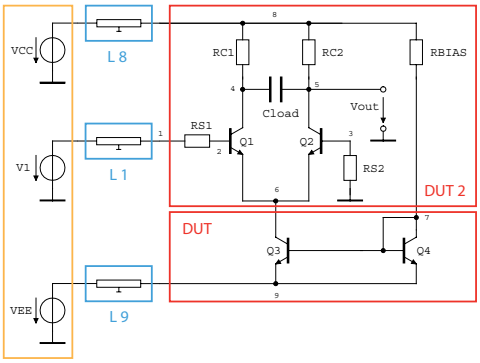


FIGURE 3.3.
Differential amplifier segmented into five coupled subcircuits (L1), (L8), (L9), (DUT), and (DUT2).

well be modelled by using inputs and outputs. Also in the context of electrical circuits, sometimes a classification into inputs and outputs is inevitable. As an example, consider an operational amplifier. In an electrical circuit such as the RLC one mentioned above, there exists a current[4] and a voltage potential at each of its ports. While one of them is imposed, the other one is computed. However, this will not be the case for a *logic* device such as the operational amplifier, since imposing the amplified *output* voltage will not lead to a suitable *input* voltage. Therefore, as a *general* methodology to cope with coupled and interconnected physical systems, the concept of inputs, outputs, and output-to-input assignments may lead to a deficient way of describing the actual physical phenomena of the real system.

The behavioral approach, however, does not seem to have met with a lot of acceptance in control theory. This might be caused by the fact that it is very natural to consider controlled variables as inputs and measured ones as outputs. Further, one can show that LTI systems always allow an interpretation as a coupled system of input-output based subsystems. This might have led to the impression that the input-output framework is always suitable.

The two different concepts reviewed above lead to different graphical descriptions in the contexts of circuit and systems theory. As an example, consider the differential amplifier shown in Figure 3.3 as an interconnected system composed of five subcircuits (the sources `VCC`, `VEE` included in (L8), (L9), respectively). The controlled variable, i.e. the *input* to the entire system, is given by the voltage V_1 of the voltage source `V1`, while the measured variable, i.e. the *output*, is the voltage potential of node 5 in the circuit, which is denoted by $V\$5$. A graphical description in a circuit-theoretic context on circuit level differs from that in a systems-theoretic context using the input-output framework as shown in Figure 3.4. While the graphic in the circuit-theoretic context is obvious and can be derived immediately from the diagram in Figure 3.3, the graphic for the latter concept can be explained as follows; at each terminal of one of the five subcircuits, there exists a voltage potential and a current which is assumed to be directed inwards. While one of them is imposed, the other one is computed. One thus has one input and one output, i.e. one *incoming* and one *outgoing* arrow for each terminal of the subcircuit. (DUT2), for example, has four terminals, hence four input arrows and four output arrows. The

[4]Assumed to be directed *inwards*.

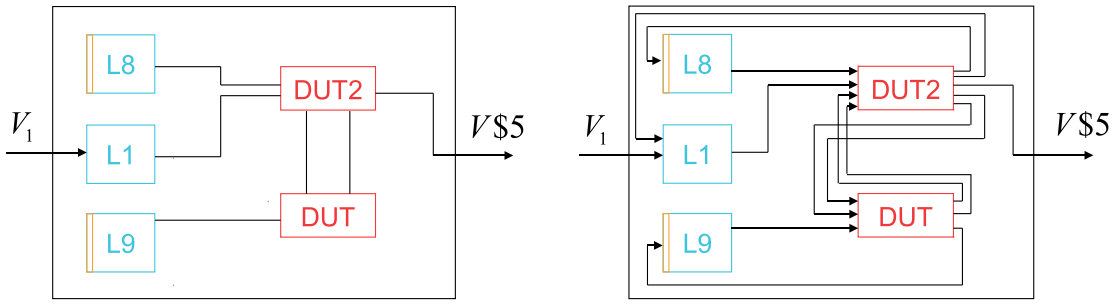


FIGURE 3.4. Graphical descriptions of the differential amplifier as a coupled system of *interacting* subcircuits on circuit level (*left*) and in the framework of inputs and outputs (*right*).

fifth output arrow denoted by $V\$5$ corresponds to the *external output*, i.e. the output of the entire system.

The three subcircuits (L1), (L8), (L9) are transmission lines and can be described by a linear system of DAEs. Therefore, the equations for describing the five subcircuits in the input-output framework could be shaped as follows for suitable states $x_{(\cdot)}$, matrices $E_{(\cdot)}$, $A_{(\cdot)}$, $B_{(\cdot)}$, $C_{(\cdot)}$, and functions $F_{(\cdot)}$:

$$\begin{aligned} E_{\mathsf{L1}}\dot{x}_{\mathsf{L1}} &= A_{\mathsf{L1}}x_{\mathsf{L1}} + B_{\mathsf{L1}}(v_1, u_{ext} = V_1)^T, \\ i_1 &= C_{\mathsf{L1}}x_{\mathsf{L1}}, \end{aligned} \tag{3.2}$$

$$\begin{aligned} E_{\mathsf{L8}}\dot{x}_{\mathsf{L8}} &= A_{\mathsf{L8}}x_{\mathsf{L8}} + B_{\mathsf{L8}}v_8, \\ i_8 &= C_{\mathsf{L8}}x_{\mathsf{L8}}, \end{aligned} \tag{3.3}$$

$$\begin{aligned} E_{\mathsf{L9}}\dot{x}_{\mathsf{L9}} &= A_{\mathsf{L9}}x_{\mathsf{L9}} + B_{\mathsf{L9}}v_9, \\ i_9 &= C_{\mathsf{L9}}x_{\mathsf{L9}}, \end{aligned} \tag{3.4}$$

and further, with the currents i_6, i_7 directed towards the interior of (DUT),

$$0 = F_{\mathsf{DUT}}(x_{\mathsf{DUT}}, \dot{x}_{\mathsf{DUT}}, y_{\mathsf{DUT}}, t; u_{\mathsf{DUT}}, p_{\mathsf{DUT}}), \tag{3.5}$$

$$0 = F_{\mathsf{DUT2}}(x_{\mathsf{DUT2}}, \dot{x}_{\mathsf{DUT2}}, y_{\mathsf{DUT2}}, t; u_{\mathsf{DUT2}}, p_{\mathsf{DUT2}}), \tag{3.6}$$

where $u_{\mathsf{DUT}} = (i_6, i_7, i_9)^T$ and $y_{\mathsf{DUT}} = (v_6, v_7, v_9)^T$ and where $u_{\mathsf{DUT2}} = (i_1, i_8, v_6, v_7)^T$ and $y_{\mathsf{DUT2}} = (v_1, v_8, -i_6, -i_7, y_{ext} = V\$5)^T$. Note that the choice of the internal inputs and outputs of the five subcircuits is not completely arbitrary, but there are various different choices possible. For example, i_6 and v_6 are interchangeable in the input and output vectors of (DUT) if they are interchanged in the input and output vectors of (DUT2) as well.

Let the *input* and *output vectors* u and y contain the *internal inputs* and *outputs* u_j and y_j as components. Arranging them in accordance with the ordering

$$(\mathsf{L1}, \mathsf{L8}, \mathsf{L9}, \mathsf{DUT}, \mathsf{DUT2})$$

of the subcircuits, i.e.

$$\begin{aligned} u &= \big((v_1, V_1)^T, v_8, v_9, (i_6, i_7, i_9)^T, (i_1, i_8, v_6, v_7)^T\big)^T, \\ y &= \big(i_1, i_8, i_9, (v_6, v_7, v_9)^T, (v_1, v_8, -i_6, -i_7, V\$5)^T\big)^T, \end{aligned}$$

the coupling coefficient matrices in (3.1) are given by the blocks of

$$K = \left[\begin{array}{ccc|ccc|cccc|c} 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{-1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{-1} & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right],$$

$$H = \left[\begin{array}{c|c|c|ccc|cccc} 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right]^T,$$

$$R = \left[\begin{array}{ccc|ccc|cccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} \end{array}\right],$$

i.e.

$$u = Ky + Hu_{ext}, \qquad y_{ext} = Ry.$$

In the aforementioned references for structure-preserving model order reduction of LTI systems, the block matrices K, H, and R play an important role.

3.4. The Macromodel Concept

As already mentioned in the introduction of this chapter, large systems and, in particular, large circuits can be segmented in smaller subsystems (***macros***) and an interconnecting network in order to overcome the complexity problem. Thus, the smaller subsystems of lower complexity can be processed separately with an increased level of efficiency. In the following, we will focus on the case of electrical circuits only.

The approach using segmentations of large circuits seems just natural because almost all of them are composed of several *building blocks* that are coupled by a suitable interconnecting structure. Thus, the segmentation of a large electrical circuit is given in a more or less natural way. And even the subcircuits or macros themselves may be composed of several components and subcircuits such that a further decomposition might be useful. Also in view of parallelization aspects these approaches seem to be valuable.

Ideas in that direction for the use in accelerated simulation are not new, see [**HsiRab80, HsiRab82, HsiRabRue, LiNorHsi, RabSanHsi, Spi83, Spi90, YanHajTri**]. Even earlier and more than 30 years ago concepts for *macromodelling* have been published [**ChuChe, Bra, RabHsi**]. In Chapter 4, we want to adapt the macromodelling approach to MOR of large electrical circuits.

For simplicity, consider an electrical circuit Σ which is divided only into two subcircuits P and Q (Figure 3.5). P and Q *share* the nodes with voltage potentials $v_1, \ldots, v_m$, where m is the number of wires connecting the two macros of Σ. According to the Kirchhoff

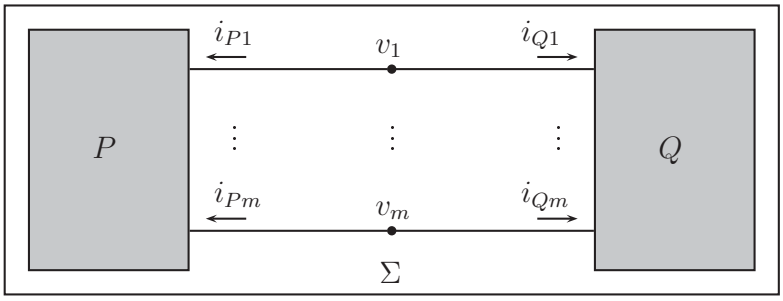


FIGURE 3.5. Segmentation of a circuit Σ into two subcircuits (macros) P and Q.

laws (2.1), one obviously has

$$i_{Pj} + i_{Qj} = 0, \qquad j = 1, \ldots, m. \tag{3.7}$$

Notional, one then separates Σ at the shared nodes $v_1, \ldots, v_m$ and connects the resulting "open wires" either to voltage sources generating the voltage potentials $v_1, \ldots, v_m$ or to current sources generating the currents $i_{P1}, \ldots, i_{Pm}$ and $i_{Q1}, \ldots, i_{Qm}$. This is done for the current source case in Figure 3.6. Thus, one obtains "closed subcircuits" with defined input-output behavior for P, Q and separate systems of equations can be formulated. These can be used to simulate or reduce the subsystems corresponding to the macros P, Q separately by applying different symbolic or numerical reduction methods. A reduced model for the original circuit Σ then is obtained by using the same interconnecting structure as for Σ to couple P and Q with each other. This means that the same current relations (3.7) are used. In this process, the artificially introduced current sources are deleted.

It is obvious that the described procedure yields big advantages w.r.t. computational efficiency if the entire circuit is divided into more than two macros. Moreover, if the *macromodel concept* is applied in subsequent *levels*, i.e. repeated recursively for appropriate subcircuits, one obviously achieves further improvement. Particularly those subcircuits that occur multiple times offer additional potential for further improvements. If they are operating in similar domains, it may be sufficient to process only some of them

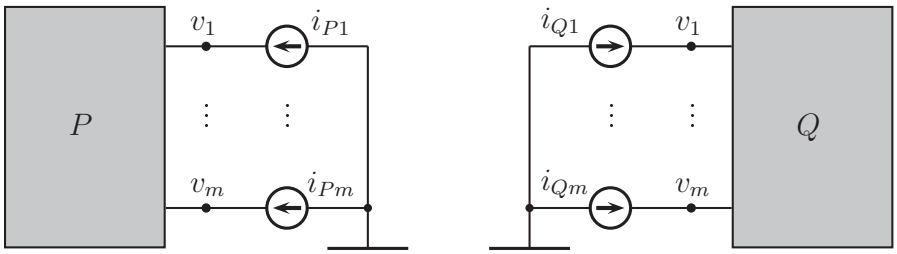


FIGURE 3.6. Separation of P and Q by introducing current sources that generate appropriate currents.

to derive reduced models for the use in each occuring instance. Finally, note that the entire algorithm can be applied to both linear and nonlinear circuits.

In the general setting of systems and subsystems, a great advantage especially with a view towards efficiency in computation time for simulations is that only those subsystems with slow convergence rates need a high number of iteration steps. The well converging ones only need few steps. According to experience, in large systems there are only few parts that yield highly flexuous solution curves or have badly converging nonlinearities. Therefore, using the macromodelling approach, one has to cope with small step sizes and high numbers of integration steps only for a small number of subsystems. In contrast to this, in a "regular" simulation of the entire system, the integration step size and the number of integration steps are steered by the "worst" nonlinearity or the most flexuous solution of the entire system.

An open problem using the macromodelling approach, however, is the choice of a good segmentation of the entire system, i.e. the question in how many macros and how many levels the entire system should be separated. Normally, an *optimal* or just *good* segmentation is anything but clear. A segmentation of the entire circuit into too many subsystems will cause high cost of administration of the interconnecting network. On the other hand, in the context of electrical circuits the design even nowadays is still made in a modular way. Hence, a – more or less – *good* segmentation seems to be given in a natural way by considering the building blocks used to design the circuit as macros of the entire circuit.

CHAPTER 4

Hierarchical Model Order Reduction

The model order reduction techniques presented in Section 2.5 are applied to the *entire* system and do not make use of the system's structure. In contrast to this, in this chapter we want to exploit its topological, hierarchical structure during the reduction process. One key question is to find methods to obtain an estimate about the "importance" of a single subsystem in the entire connecting structure of the system.

In the first section, we give a motivating example that compares the usual *non-hierarchical* reduction approach to a technique that exploits the structure on circuit level. This shows that the latter method is very promising. Compared to the non-hierarchical approach, we only need a fraction of the computation time to derive a reduced system with only about half the number of equations, but the same level of accuracy.

In the second section, we present a new workflow for the reduction of subcircuits T coming from a hierarchical segmentation on circuit level. By using information obtained from a previous simulation run of the entire electrical circuit Σ, appropriate voltage sources are connected to T and the obtained *closed circuit* is reduced.

The next section introduces the new concept of *subsystem sensitivities* that measure the influence of a subcircuit T on Σ similar to the ideas of a term ranking. Therefore, we subsequently give some detailed information about the idea behind *term ranking*. Also a brief review of the *clustering* strategy is provided in order to give ideas for improvement of the finally derived hierarchical reduction algorithm in a later section. The subsystem sensitivity analysis measuring the influence of T on the output of Σ can be exploited to obtain a *ranking of subsystem reductions*, i.e. an optimized order of these reductions.

Section 4.4 then deals with an algorithm for the computation of a hierarchically reduced model $\widetilde{\Sigma}$ of Σ by making use of subsystem sensitivities. It uses their ordering w.r.t. the error caused on the output of Σ to derive a *ranking of subsystem reductions*. Then in each step the subsystem with currently the least ranking value is reduced.

All the considerations from the previous sections finally are combined in the fifth section. There, a new reduction algorithm exploiting the hierarchy of the circuit and the sensitivities of the corresponding subsystems is introduced.

Error functions have crucial influence on a system's degree of reduction. Depending on the example under investigation, suitable error functions have to be employed for controlling the approximation error during the reduction process. Therefore, Section 4.6 provides a general definition as well as a choice of different functions that have been developed and used in this thesis.

Finally, a sample application is considered in the last section. We apply the new hierarchical reduction approach to an operational amplifier typically used in industry and

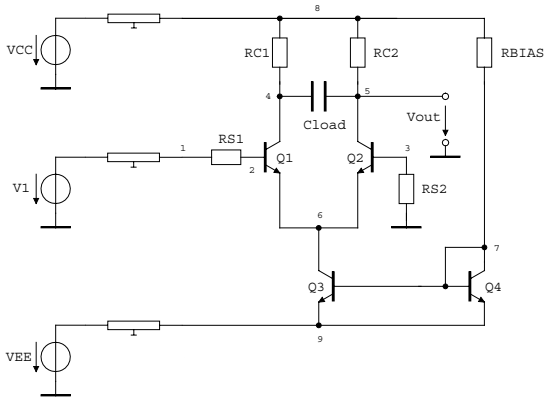


FIGURE 4.1.
Differential-amplifier circuit.

compare the results to the usual non-hierarchical reduction. Thereby, symbolic reduction methods for nonlinear systems and two different error functions come to operation. The corresponding results are also compared to each other. We then take additional information about the subsystems' number of equations and terms into account and derive reduced models of even less complexity. Then the obtained *interim models* are further reduced by applying non-hierarchical techniques. Compared to the original system, this yields *hybrid reduced models* of very low complexity. We show that some of the resulting models do not only have a high level of accuracy. They moreover are very robust w.r.t. certain input excitations and can be simulated quite a lot faster than the original system.

4.1. Motivating Example

Let us start with a motivating example and consider the differential-amplifier circuit introduced in the previous chapter, which is shown in Figure 4.1. The voltage sources `VCC` and `VEE` denote the voltage supply for the amplifier circuit, whereas the voltage generated by the source `V1` defines the input. The output is given by the voltage-potential of node 5, denoted by `Vout`. The three sources are connected to the remaining circuit components via three transmission lines, for which a discretized PDE model of the *telegrapher's equations* (2.36) from page 32 with 20 line segments each is used.

In the following subsections, we perform *structure exploiting* symbolic techniques using ideas from the previous chapter to reduce the differential amplifier and compare the resulting simplified system to the usual approach of reducing the entire circuit without taking any structure into account. We will see that, compared to the regular approach, the structure exploiting one offers a large potential to derive simplified models of lower complexity and at only a fraction of the computational cost.

4.1.1. Symbolic Reduction of the Entire Circuit. Using MNA in node voltage formulation to set up the describing system of equations for the transient circuit behavior, we obtain 167 equations with a total number of 645 terms. For the input `V1`, we choose a sine-wave voltage excitation u_0 with a magnitude of 2 V and a frequency of 10^5 Hz. The supply voltages `VCC` and `VEE` are 12 V and -12 V, respectively.

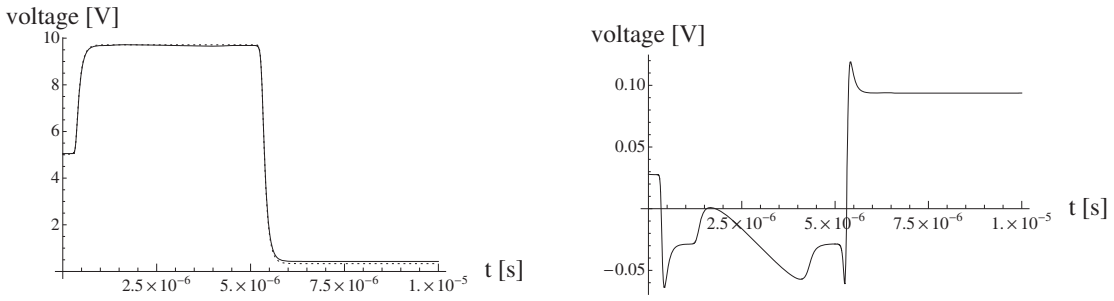


FIGURE 4.2. *Left:* Solutions of the original (solid) and the reduced system (dotted) allowing 2% maximum error. *Right:* The corresponding error plot.

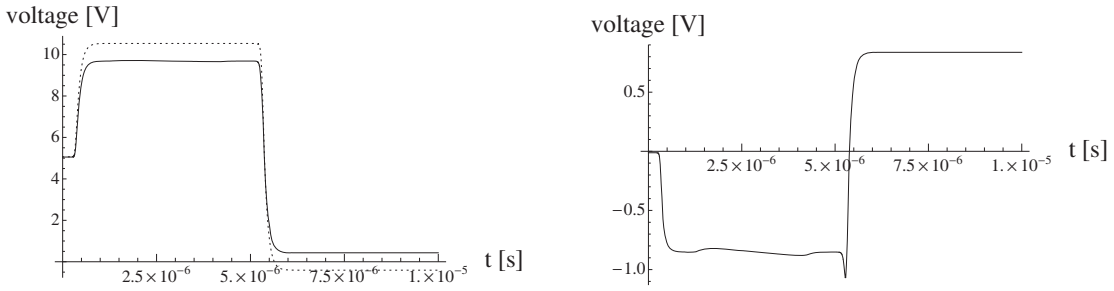


FIGURE 4.3. *Left:* Solutions of the original (solid) and the reduced system (dotted) allowing 10% maximum error. *Right:* The corresponding error plot.

By using the symbolic reduction techniques for nonlinear systems presented in Section 2.5.2.2 and permitting an error of 2%[1] for the output `Vout` to reduce the entire system, we obtain a system consisting of 124 equations with 425 terms in total. For the reduction, a few hours are needed[2], in which more than 95% of that time is spent for the computation of the transient term ranking[3]. See Section 4.3.1 for more information about the ranking of terms. Figure 4.2 shows the solution and the error of this reduced system in comparison to the solution of the original one, i.e. the reference solution. According to these figures, the reduced system with only about 1% error is a very accurate approximation.

If we allow an error of 10% for the reduction instead of only 2%, we obtain 44 equations with a total of 284 terms. The time needed is almost the same because of the ranking computation mentioned above. However, the reduced complexity of the equations has been achieved at the expense of accuracy (cf. Figure 4.3).

4.1.2. Exploitation of the Hierarchical Structure. What happens if we take an "intuitive" reconstruction of the hierarchy into account? Therefore, consider certain

[1] $x\%$ error here means $\frac{x}{100}$ of the maximum amplitude of the reference solution, considered on the entire time interval $\mathcal{T} = [t_0, t_1]$ of the analysis: $\frac{x}{100} \cdot \sup_{t\in[t_0,t_1]} \|y_F(t)\|$. This error is measured using the error function given by (4.9) in Section 4.6.

[2] The computations were performed on a Dual Quad Xeon E5420 with 2.5 MHz and 16 GB RAM.

[3] Note that we use full simulations for the ranking of the terms, i.e. the *perfect ranking*.

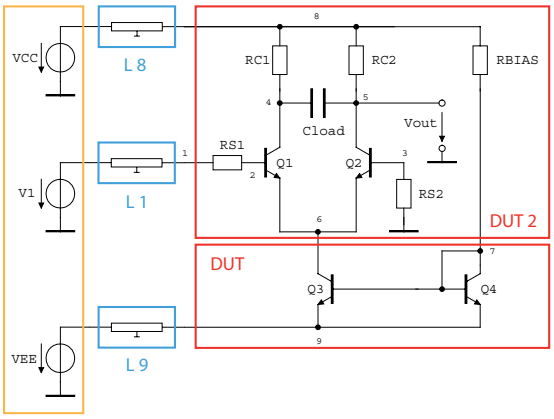


FIGURE 4.4.
Differential-amplifier circuit with its "intuitive" hierarchy.

structural patterns on circuit level in order to define subcircuits within the differential amplifier: the *differential pair* of transistors $Q1$, $Q2$ together with some resistors in the upper box on the right side of Figure 4.4 form a subcircuit (DUT2), the *current mirror* in the lower right defines (DUT), the three transmission lines in the middle form three subcircuits (L1), (L8), and (L9), and finally there are the sources on the left.

As mentioned before, by using e.g. MNA to set up the describing equations for the *entire* circuit, the hierarchy information is lost, since the system contains equations with mixed parts from different subcircuits. But if the five subcircuits are considered one by one, the hierarchy information on circuit level can be transmitted into the describing set of equations and the subcircuits can be reduced separately. This is done following the ideas from Section 3.4, an exact description is given in Section 4.2.

For the reduction of the three transmission lines, we translate the describing linear system of equations into state space formulation and apply Arnoldi's algorithm (see Section 2.5.1.5). For both the transmission lines (L8–9) we iterate only one step. This provides sufficient accuracy here, since `VCC` and `VEE` are DC sources. For the transmission line (L1) which is connected to the input voltage `V1`, we perform three steps in the Arnoldi iteration. Some experiments with a higher number of iteration steps show that the gain of additional accuracy in comparison to the size of the larger system is rather marginal for a number of iterations bigger than five. Due to two additional equations and variables for internal port modelling, we thus can reduce the transmission line subsystems from 50 down to 8 resp. 4 equations.

The subsystems corresponding to (DUT) and (DUT2) are reduced symbolically using the nonlinear techniques from Section 2.5.2.2. For both these reductions a 2% error bound is permitted, but we observe almost no further reductions, if we allow 10% error instead. Note that these error bounds are not limiting the error of the entire circuit's output `Vout` to the given values, since only the single subsystems are reduced (see Section 4.2 and Algorithm 4.1 for further details). In case of the current mirror subsystem, the symbolic simplification yields 9 instead of 16 equations with a total number of 20 instead of 59 terms. For the subsystem with the differential pair, we obtain a reduced system with 13 instead of 22 equations and 50 instead of 91 terms in total. Note that all these reductions are computed *within seconds*.

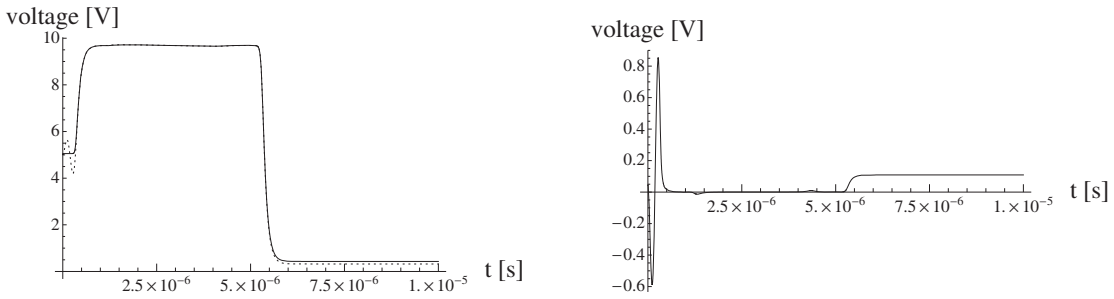


FIGURE 4.5. *Left:* Solutions of the hierarchically reduced system (dotted) and the original one (solid) by applying the 3-step Arnoldi iteration to (L1). *Right:* The corresponding error plot.

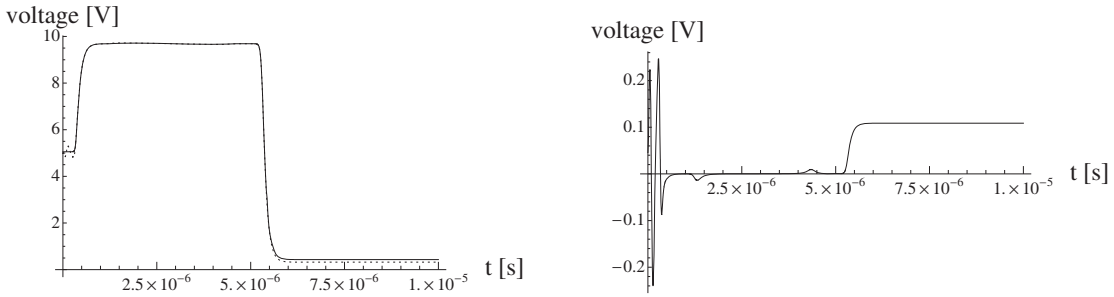


FIGURE 4.6. *Left:* Solutions of the hierarchically reduced system (dotted) and the original one (solid) by applying the 5-step Arnoldi iteration to (L1). *Right:* The corresponding error plot.

Plugging together all the reduced subsystems in order to obtain a simplified model for the entire differential-amplifier circuit, the results are very accurate as one can see in Figure 4.5. Instead of 167 equations and a total number of 645 terms, we only have 62 equations with 252 terms altogether. The error of the reduced entire system compared to the reference solution of the original one is approximately 8%.

A higher dimensional projection space obtained by the Arnoldi iteration provides more accurate solutions, but of course also leads to larger systems. If, for example, we perform five Arnoldi iteration steps to reduce the subsystem corresponding to (L1) instead of only three, we can further improve our result. We then obtain a maximum error of the entire reduced system of approximately 2% (Figure 4.6). However, in that case the reduced entire system consists of 66 equations and a total number of 396 terms.

4.1.3. Application of Different Inputs. In order to check whether the simplified model obtained by the hierarchical reduction approach works fine also for other inputs, we apply a pulse wave u_1 with a magnitude of 2 V, a sum u_2 of three pulses with magnitudes of 1 V, 2 V, and 3 V, respectively, and a sum of three sine functions

$$u_3(t) := 2 \cdot sin(2\pi \cdot 10^5 \cdot t) + 2 \cdot sin(2\pi \cdot 2 \cdot 10^5 \cdot t) + 1 \cdot sin(2\pi \cdot 5 \cdot 10^5 \cdot t)$$

to V1. The graphs of these functions are shown in Figure 4.7 on the time interval $[\,0\,s, 10^{-5}\,s\,]$.

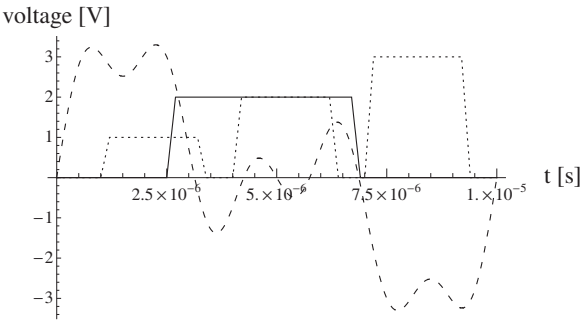


FIGURE 4.7. Three other inputs u_1, u_2, and u_3 for `V1` to test the reduced model of the differential-amplifier circuit.

With these inputs, both the original differential amplifier and the hierarchically reduced model[4] are simulated. The graphs of the simulations as well as the corresponding error plots are shown in Figure 4.8.

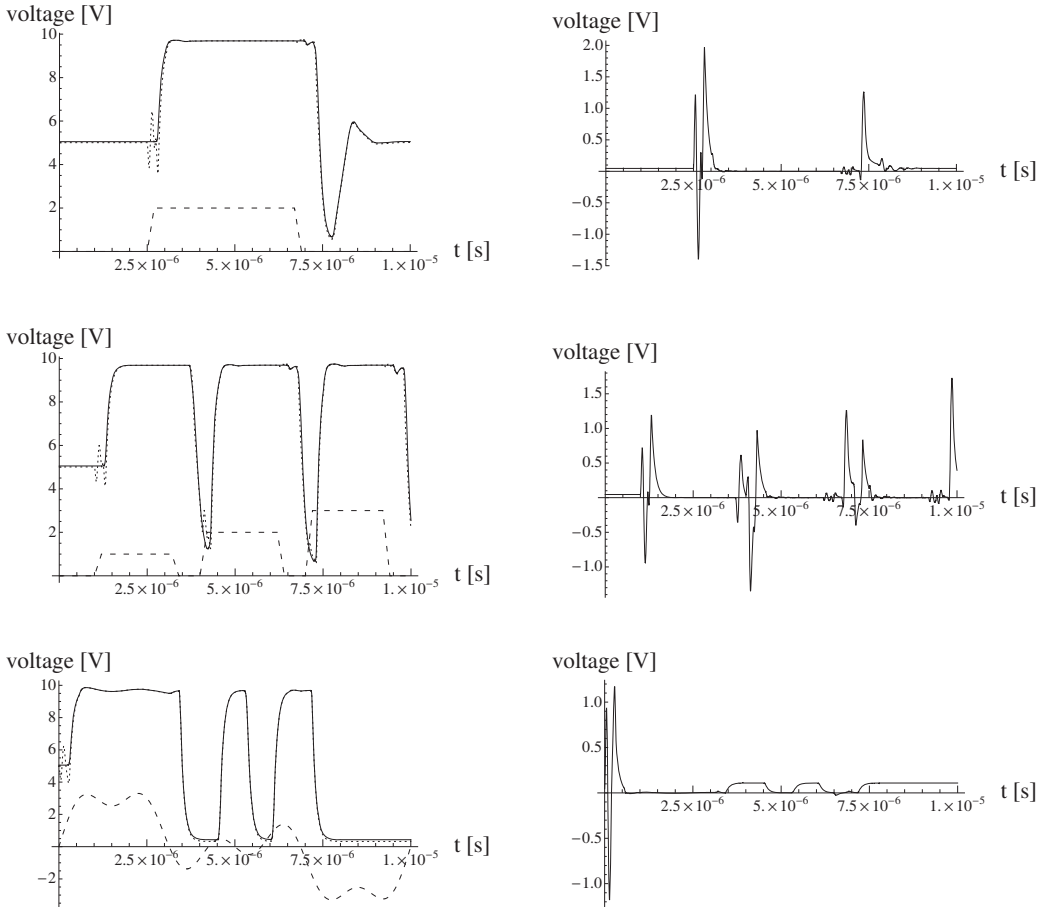


FIGURE 4.8. *Left:* Simulation results of the original (solid) and the hierarchically reduced model (dotted) of the differential amplifier, together with the corresponding input `V1` (dashed). *Right:* The corresponding error plots.

[4]We choose the more accurate model with 66 equations, see Figure 4.6.

system	orig.	non-hier. reduced		hier. reduced	
equations	167	124	44	66	62
terms	645	425	284	396	252
(perm.) error		2%	10%	$\sim 2\%$	$\sim 8\%$
Arnoldi steps				(5/1)	(3/1)
time costs		few hours		within seconds	
simulation performance					
input u_0	8.2 s	3.2 s	0.8 s	1.5 s	1.2 s
input u_1	9.4 s			1.9 s	
input u_2	13.2 s			3.1 s	
input u_3	12.1 s			2.1 s	

TABLE 4.1. Results of the two different reduction approaches for the differential-amplifier circuit.

According to these figures, the obtained model works quite well in all the test cases and, therefore, can be used as a *behavioral model* of the original differential-amplifier circuit in a certain domain.

4.1.4. Comparison of the two Approaches. The results of the two different reduction approaches for the differential-amplifier circuit are listed in Table 4.1. Comparing the two approaches, we conclude that the one exploiting the hierarchical structure is the better choice, since it delivers much better results in less computation time; we obtained reduced systems with almost the same accuracy in the 2% error case, but only about half the number of equations for the structure-exploiting approach. Performing only three steps in the Arnoldi iteration to reduce the transmission line connected to the input voltage `V1`, we still obtained a reduced overall system that fits the original *reference* solution quite well. On the other hand, the non-hierarchical reduction of the entire circuit permitting an error of 10% led to a system that fully exploits the error bound.

As it is further shown in Table 4.1, the speed-up for simulations of the hierarchically reduced model compared to the original full model is approximately by a factor of 5.

4.2. A Workflow for Subcircuit Reductions

In this section, we present an algorithm for reducing the single subcircuits coming from the hierarchical segmentation of the entire circuit.

Assume an electrical circuit Σ that already is hierarchically segmented into a set of subcircuits T_i and an interconnecting structure S:

$$\Sigma = (\{ T_i \,|\, i = 1, \ldots, m \}, S)\,. \tag{4.1}$$

In general, each T_i itself could possibly be segmented recursively into a set of subcircuits and a coupling structure. However, we do not further treat this case here and consider the segmentation only "on level 0".

In order to maintain the hierarchical structure available on circuit level, one cannot simply apply methods like MNA to set up describing equations for Σ, since this yields equations involving terms that are mixed from different subcircuits. Instead, we use a subcircuit reduction workflow that uses the segmentation on circuit level in a similar way as described in the macromodel concept in Section 3.4. Since we always need a reference solution to symbolically reduce a system and to keep track of the error, we need a "closed circuit" with defined input-output behavior. If we simply cut out a subcircuit from its connecting structure, we do not have a defined input-output behavior at its "open" terminals, i.e. we have no information about its current-voltage relations at its terminals. Also numerical reduction techniques rely on an input-output concept acting on the maxim that either the voltage potential or the (inward) current at a terminal is prescribed, while the other one is computed. Therefore, we simulate the subcircuit in question to be reduced, say T, in a *test bench*, i.e. a simulation test environment. For example, one could simulate the entire circuit Σ itself such that the remaining subcircuits together with the interconnecting structure S form a test bench for T. During the simulation, the voltage potentials at the terminals of T are recorded. Then T is connected to voltage sources generating exactly the recorded voltages. Thus, we have a "closed circuit" C_T with a defined input-output behavior at its terminals. A method such as the modified nodal analysis is then used to transform C_T into a describing system of equations[5] F_T which can be reduced using symbolic or numerical methods from Section 2.5.

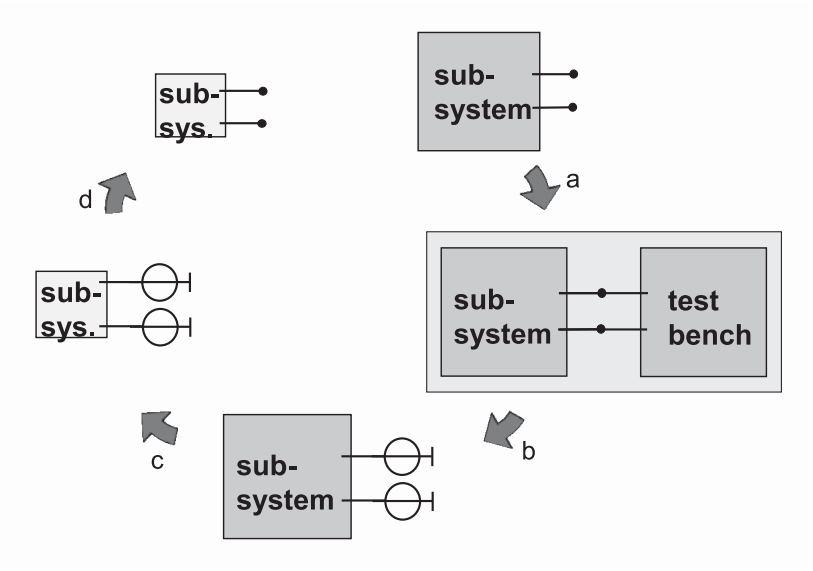


FIGURE 4.9. Schematic illustration of the workflow for subcircuit reductions.

[5]Assume we are dealing with systems of DAEs. In case PDEs are involved, apply a semidiscretization w.r.t. the spatial coordinates.

The whole procedure is summarized in Algorithm 4.1 and schematically shown in Figure 4.9. Of course, we could as well record the currents and connect current sources to the terminals of T. However, the approach using voltage sources seems to be easier, since no directions of the flowing currents have to be taken into account.

Algorithm 4.1. (Reduction of subcircuits)

Let T be a subcircuit in an electrical circuit $\Sigma = (\{ T_i \mid i = 1, \ldots, m \}, S)$.

a. *Connect T to a test bench and record the voltage potentials at its terminals during a simulation run applying a suitable input.*

b. *Remove the test bench and connect grounded voltage sources to the terminals of T that generate exactly the recorded voltage potentials, thus having T isolated as a "closed circuit" C_T; further, set up a describing system of equations F_T for C_T.*

c. *Reduce F_T by using appropriate symbolic or numerical reduction techniques, where the voltages at all terminals of C_T are the inputs and the currents (flowing inwards) are the outputs.*

d. *Remove all voltage sources after the reduction and finally obtain a reduced subsystem $\widetilde{F}_T$ that serves as a behavioral model of T.*

Proceeding in this manner, the terminals of T are preserved during the reduction process and the original subcircuit T in the entire circuit Σ can easily be replaced by the reduced version $\widetilde{F}_T$ of F_T with removed sources and using the same interconnecting structure S. Repeating the entire procedure for each of the subcircuits T_i in Σ, we obtain a reduced version of the overall circuit Σ.

This workflow has been applied to the five subcircuits of the differential-amplifier example in Section 4.1 and delivered the results described there.

It should be mentioned here that this approach only controls the errors at the terminals of the single subcircuits. A priori, one cannot guarantee a certain *global error*, i.e. the error on the output of the entire circuit Σ when replacing the original subcircuits T_i by reduced models $\widetilde{F}_{T_i}$. This could be a point of future investigation (see also Chapter 6).

4.3. Subsystem Sensitivities

In this section, we investigate the influences of the single subcircuits T_i on the behavior of the entire circuit $\Sigma = (\{ T_i \mid i = 1, \ldots, m \}, S)$. Therefore, we want to have an estimate of a subcircuit's *sensitivity*, i.e. the sensitivity of Σ with respect to changes in the corresponding subcircuit's behavior. This sensitivity is measured by the influences of subcircuit reductions on the output of Σ and leads to a *ranking of subcircuit reductions*, i.e. a heuristically optimized order of subcircuit reductions.

Before we explain the approach in more detail, we illuminate and review the concepts of *term ranking* and *clustering* used for an optimized order of efficient reductions in the symbolic case as briefly described in Section 2.5.2. We follow the notes of [**Wic04**].

4.3.1. Term Ranking. A ranking is a method to estimate the influence of certain magnitudes on others. In general, it is a trade-off between accuracy and efficiency in computation time. In the case of a term ranking mentioned in Section 2.5.2 it is supposed to estimate the influence of a symbolic term ϑ in a system of equations F on its solution y_F.

Definition 4.2. *Let $\mathcal{K}$ be the index set of all terms ϑ in a symbolic system of DAEs F and let $\mathcal{R} \in \{$term reduction, term substitution$\}$ be an approximating reduction method. For $k \in \mathcal{K}$, let $\mathcal{R}(F,k)$ denote the system of DAEs obtained by applying $\mathcal{R}$ on F, thus reducing the k-th term ϑ_k in F. For an input u and a numerical analysis $\mathcal{A}$, let $y_k := \mathcal{A}(\mathcal{R}(F,k),u)$ and $y_F := \mathcal{A}(F,u)$ denote the corresponding solutions of the obtained reduced system and the original system F, respectively. Finally, let E be an error function, $\varepsilon > 0$, and $\mathcal{K}_r \subseteq \mathcal{K}$. A function r with $r(F,k) \in \mathbb{R}_{\geq 0}$ and*

$$|\, r(F,k) - E(y_F, y_k)\,| \leq \varepsilon \quad \text{for all } k \in \mathcal{K}_r \tag{4.2}$$

*is called an **absolute ranking**.*

In this definition, $\mathcal{K}_r$ denotes the set of terms for which the ranking r computes a good prediction of their influence on y_F.

Obviously, the larger $\mathcal{K}_r$ is and the smaller ε can be chosen, the better is the absolute ranking r. The best estimate clearly is achieved if a full numerical analysis is performed for each term ϑ in F. Hence, the corresponding most accurate ranking computing the *exact* term influences on y_F, the *perfect ranking*, is obtained by computing the values

$$r^*(F,k) := E(y_F, y_k)$$

for all possible terms ϑ_k occuring in F. With the notation from above one then has $\mathcal{K}_{r^*} = \mathcal{K}$ and $\varepsilon = 0$.

The computed ranking values for all terms are put in an increasing order thus yielding a *ranking list*. This list is an optimized order in which the term reductions on F should be executed, since terms that cause rather small deviations from y_F are simplified first, while those ones causing large deviations are processed later or not at all. This approach leads to a high degree of reduction for the entire system.

However, a general term ranking only *estimates* the influence of a term ϑ on y_F and sometimes fails in predicting the correct order of magnitude. More precisely, for terms ϑ_k with $k \in \mathcal{K} \setminus \mathcal{K}_r$, the prediction is wrong. Consequently, the order obtained in the ranking list might be a little disarranged, since some term influences are over- or underestimated[6]. For $r(F,k) \gg E(y_F, y_k)$, a so-called *type* 1 *ranking error* occured and the influence of ϑ_k is *overestimated*. Hence, the reduction corresponding to a type 1 ranking error might be executed too late. In case $r(F,k) \ll E(y_F, y_k)$, a *type* 2 *ranking error* maybe leads to a reduction which is performed too early, since the influence of ϑ_k is *underestimated*.

[6]Note that this only influences the *order* of performed reductions, the correctness of the approximating reduction method is not affected. However, the more accurate the ranking method estimates the influences, the higher is the degree of reduction.

As described above, the choice of the next term to be reduced is steered by the current minimum of the ranking list. Thus, only the *relative relations* among the terms are important for this choice, the absolute ranking values, however, do not play an important role. The constraint (4.2) in Definition 4.2 therefore can be weakened as follows:

Definition 4.3. *Using the notations of Definition 4.2,* (4.2) *is replaced by*

$$r(F,k_1) < r(F,k_2) \;\Leftrightarrow\; E(y_F,y_{k_1}) < E(y_F,y_{k_2}) \;\; \textit{for all } k_1,k_2 \in \mathcal{K}_r. \tag{4.3}$$

The resulting r is then called a ***relative ranking****.*

As before, the quality of the relative ranking r grows with the cardinality of $\mathcal{K}_r$. If $k_1 \in \mathcal{K} \setminus \mathcal{K}_r$, a so-called *relative ranking error* occured, i.e. either

$$r(F,k_1) < r(F,k_2) \quad \text{and} \quad E(y_F,y_{k_1}) \geq E(y_F,y_{k_2})$$

or

$$r(F,k_1) \geq r(F,k_2) \quad \text{and} \quad E(y_F,y_{k_1}) < E(y_F,y_{k_2})$$

for some $k_2 \in \mathcal{K}$.

Unfortunately, the perfect ranking r^* is far too costly for realistic applications, since with $r^*(F,k) = E(y_F,y_k)$ and $y_k = \mathcal{A}(\mathcal{R}(F,k),u)$ it requires a full numerical simulation for each symbolic term ϑ occuring in F. Moreover, after the first term is reduced and a new system $G = \mathcal{R}(F,k)$ with one term less is created, the ordering of the terms' influences might have changed compared to the current ranking list. Therefore, the remaining $|\mathcal{K}|-1$ terms in G should be processed in a new ranking computation. After the second reduction step is executed, the entire procedure should be repeated for the remaining $|\mathcal{K}|-2$ terms and so on.

However, in order to avoid the very time-consuming complete numerical analyses $y_k = \mathcal{A}(\mathcal{R}(F,k),u)$ necessary for computing the perfect ranking r^*, a variety of techniques exists. Since in general numerical analyses $\mathcal{A}(G,u)$ are obtained by approaches such as Newton's method, one could for example stop the Newton iteration after a few – or even after one – steps to obtain an approximate solution $\widetilde{y_k} \approx y_k$. Then $r(F,k) := E(y_F,\widetilde{y_k})$ defines a ranking r for the terms in F. For an overview of some other methods we refer to **[Wic04]**. All of them are trade-offs between accuracy and efficiency in computation time, such that a faster ranking computation in general coincides with less accuracy of the resulting ranking list.

Furthermore, it was observed in numerous examples that the ranking list based on the original system of DAEs F yields sufficiently accurate estimates for the terms' influences on y_F during the entire reduction, such that repeated ranking computations during the entire reduction process are not necessary. Therefore, the ranking computation is performed only once at the very beginning.

4.3.2. Clustering. Normally, the accumulated error on y_F has to be checked after each single reduction step. But this means each time a costly analysis of the so far reduced system G. To avoid these time-consuming numerical analyses after each single

reduction step, the terms in F are *clustered* according to their ranking values, i.e. the ranking list R is segmented into a disjoint union of l subsets or *clusters* C_i:

$$R = \bigcup_{i=1}^{l} C_i.$$

Then, whole clusters of terms are reduced simultaneously without checking the validity of each single term reduction. Just after the reduction of a whole cluster of terms a numerical error analysis is carried out. If the error bound is *not* violated, the reduction of the next cluster is executed. Otherwise, all term reductions within the current cluster have to be rejected, a new cluster subdivision has to be made, and the procedure is repeated with the first *new* subcluster of terms. In this process, those clusters that cause a rather small error, i.e. which contain terms with the smallest ranking values, are reduced first. Otherwise, if a cluster of terms that cause a large deviation from y_F is reduced first, the tolerance for further reductions would be rather small. Proceeding in that manner heavily accelerates the entire reduction process.

For more information about the described techniques we refer to [**Wic04**]. They are integrated in a reduction algorithm for nonlinear symbolic systems of DAEs that further include an index-monitor to guarantee that the *index* of the original system of DAEs F is not increased during the reduction process.

4.3.3. Subsystem Sensitivity Analysis. Similar to the ideas of the previous subsections, we would like to know about the *importance* of a subcircuit in the connecting structure of the entire circuit Σ. In the *term ranking concept*, the influence of a *term* in the system of equations F on its solution y_F using a certain input u is estimated. Similarly, we would like to have an estimate of a subcircuit's *sensitivity* or influence on the behavior of the entire circuit, i.e. the *sensitivity* of the entire circuit Σ w.r.t. changes in the subcircuit's behavior. In the case of *term ranking*, this led to an optimized order of term simplifications to obtain a high degree of reduction. Here, we obtain a ***ranking of subsystem reductions***, i.e. an optimized order of subsystem reductions that offers a high degree of reduction for the entire system Σ by replacing its subcircuits T_i by corresponding and suitably reduced models $\widetilde{F}_{T_i}$ (see also Section 4.2 for the notation).

If one speaks about sensitivity analysis in the background of electrical circuits, one normally thinks of the influences of single components or system parameters on certain circuit or network variables. The sensitivity of such a variable z with respect to changes in a certain network parameter p is computed by using partial derivatives:

$$s_a(z,p) = \left.\frac{\partial z}{\partial p}\right|_{p=p_0} \tag{4.4}$$

is called the *absolute sensitivity* [**Spi90, Zer**] and

$$s_r(z,p) = p\left.\frac{\partial z}{\partial p}\right|_{p=p_0} = p \cdot s_a(z,p) \tag{4.5}$$

the *normalized* or *relative sensitivity* of z with respect to p. In these equations, p_0 is a nominal value of p. In an approximate way, one can compute s_a via

$$s_a(z,p) \approx \left.\frac{\Delta z}{\Delta p}\right|_{p=p_0} = \frac{z-\widetilde{z}}{p_0-\widetilde{p}} \tag{4.6}$$

using perturbed values $\widetilde{z} = z(\widetilde{p})$ and $\widetilde{p}$ of $z = z(p)$ and $p = p_0$. While $z = z(p_0)$ corresponds to a simulation of the describing equations using parameter $p = p_0$, $\widetilde{z}$ is obtained by using the perturbed parameter $\widetilde{p}$ of p_0 during the simulation.

One clearly cannot form the derivative of a circuit's or a system's output y with respect to one of its subcircuits or subsystems. Instead, we imitate the "meaning" of equation (4.6) and replace a single subcircuit T of Σ by a "perturbed version $\widetilde{T}$", i.e. by a reduced model $\widetilde{F}_T$ of the describing system of equations for T. Then we simulate the obtained entire system and compare the *original* output y to the perturbed entire system's output $\widetilde{y}$.

In the following, we define the sensitivity of a subcircuit T in the entire circuit Σ as a vector of tuples containing reduction information and the resulting error on the perturbed entire system. For simplicity, we will not distinguish between sub*circuits* and the corresponding describing sub*systems* based on equations and denote both of them simply by T.

Definition 4.4. *Let $\Sigma = (\{T_i \,|\, i = 1,\dots,m\}, S)$ be an electrical circuit of interconnected subcircuits T_i connected by a structure S. Let further T be one of the subcircuits in Σ. The **sensitivity of T in Σ** is the vector*

$$s_T = \big((r_1, E(y, y_{T;r_1})), \dots, (r_{m_T}, E(y, y_{T;r_{m_T}}))\big) \tag{4.7}$$

*that contains tuples of **reduction information r_j** for the subcircuit T and the resulting error $E(y, y_{T;r_j})$ on the output y of Σ. In this notation, $y_{T;r_j}$ is the output of the corresponding reduced entire system*

$$\Sigma_{T;r_j} = (\{T^{r_j}\} \cup \{T_i \,|\, i = 1,\dots,m\} \setminus \{T\}, S),$$

where T in comparison to the original circuit Σ is replaced by a reduced model $\widetilde{F}_T = T^{r_j}$ obtained by reducing T according to r_j.

In this definition, T^{r_j} denotes the subsystem that is obtained by performing a reduction on T which is defined by r_j. For example, r_j could contain information such as "nonlinear symbolic reduction" and "accepted error 10%" or "Arnoldi method" and "k iteration steps, projection matrix V". The subsystem reductions are executed using the workflow from Section 4.2. Note that the sensitivity of T involves systems $\Sigma_{T;r_j}$ which are the same as Σ itself except for ***exactly one subsystem replaced*** by a reduced version. In the following, the determination of a subsystem's sensitivity is explained in more detail by using an example.

Let the entire system Σ be given by (4.1) and further $T_j \in \{T_i, i = 1,\dots,m\}$ be the subsystem of which we want to investigate the sensitivity in Σ. We want to apply "*nonlinear symbolic approximating techniques*", e.g. term reductions, to T_j and therefore

provide a *sweep of error bounds*

$$sw = (1, 2, 5, 10, 25, 50, 80, 100),$$

where the numbers in sw correspond to percentage values of the accepted error[7]. The entries in this sweep are part of the reduction information in r_k, $k = 1, \ldots, m_{T_j}$, $m_{T_j} = 8$.

In the first step, the *original* subsystem T_j is reduced with the first error bound from the error sweep sw thus allowing a maximum error of 1%. For the reduction, the workflow from Section 4.2 is used. This yields a reduced system $T_j^{r_1}$. Then T_j in the original system Σ is replaced by $T_j^{r_1}$. The resulting entire system is denoted by $\Sigma_{T_j;r_1}$. Finally, a full numerical analysis is carried out on $\Sigma_{T_j;r_1}$ in order to obtain $y_{T_j;r_1} = \mathcal{A}(\Sigma_{T_j;r_1}, u)$ with the same input u as used for the computation of $y = \mathcal{A}(\Sigma, u)$.

In the second step, again the *original* subsystem T_j is reduced, but now permitting the second error bound 2% from sw. Hence, we obtain a reduced system $T_j^{r_2}$ which again replaces T_j in the *original* system Σ. We thus obtain the partially reduced entire system $\Sigma_{T_j;r_2}$, on which a full numerical analysis is carried out. This yields $y_{T_j;r_2} = \mathcal{A}(\Sigma_{T_j;r_2}, u)$.

Proceeding in this manner for the remaining error bounds b_i in sw, we obtain a set of partially reduced entire systems $\Sigma_{T_j;r_i}$, $i = 1, \ldots, 8$, where the bounds $b_i \in sw$ are "hidden" in the reduction information r_i. The corresponding numerical analyses $y_{T_j;r_i} = \mathcal{A}(\Sigma_{T_j;r_i}, u)$ might become less accurate due to the increasing error bounds b_i, while the complexities of the systems $\Sigma_{T_j;r_i}$ decrease for growing b_i. Consequently, via the error on the entire system's output, we can observe the influence of the original subsystem T_j on the original circuit $\Sigma = (\{T_i \mid i = 1, \ldots, m\}, S)$.

The procedure described up to here is repeated for all subcircuits T_k in Σ to obtain their sensitivities. The next section then describes how to use these sensitivities in order to obtain an optimized order of subsystem reductions for the derivation of a hierarchically reduced entire system.

4.4. Optimized Order of Subsystem Reductions

In this section, we present a strategy that allows an appropriate reduction of the subsystems of Σ in an optimized order. We call the list of reductions corresponding to that order a ***ranking of subsystem reductions***. The algorithm presented here uses this ranking for deriving a hierarchically reduced model of the entire system Σ.

For each subsystem T_i in Σ, we define L_i as the list containing the entries of s_{T_i} that is increasingly ordered with respect to $E(y, y_{T_i;r_j})$. In each step of the hierarchical reduction of Σ, we then take the subsystem T and the reduction information r that corresponds to the minimum of the currently first entries in the lists L_i.

If the accumulated error on the output y of Σ exceeds the user-specified error bound ε, the corresponding latest reduced subsystem T_{i_0} is reset. We assume that no further reductions on T_{i_0} are possible and, therefore, completely delete the remaining list L_{i_0}.

[7]Of course, the error depends on the choice of the error function E, see Section 4.6 for further details.

Input: segmented electrical circuit $\Sigma = (\{T_i \mid i = 1, \ldots, m\}, S)$, input u, error bound ε
Output: reduced system $\widetilde{\Sigma} = (\{T_i^{r_{i^*}} \mid i = 1, \ldots, m\}, S)$, where $T_i^{r_{i^*}}$ are suitable reduced subsystems and with $E(y, y_{\widetilde{\Sigma}}) \leq \varepsilon$

forall subsystems T_i **do**
  $L_i :=$ order(s_{T_i}) w.r.t. $E(y, y_{T_i;r_j})$
  $T_i^{r_0} := T_i$
end
$L := \{L_1, \ldots, L_n\}$
$y := \mathcal{A}(\Sigma, u)$
$\widetilde{\Sigma} := \Sigma$
repeat
  compute $(r_k, E(y, y_{T_i;r_k})) := \min_{i, L_i \in L}(\min(L_i))$ w.r.t. $E(y, y_{T_i;r_j})$
  replace $T_{i_0}^{r_{k_0}}$ by $T_{i_0}^{r_k}$
  update$(\widetilde{\Sigma})$
  $y_{\widetilde{\Sigma}} := \mathcal{A}(\widetilde{\Sigma}, u)$
  $\varepsilon_{\text{out}} := E(y, y_{\widetilde{\Sigma}})$
  if $\varepsilon_{\text{out}} \leq \varepsilon$ **then**
    $L_{i_0} := L_{i_0} \setminus \{\min(L_{i_0})\}$
    if $L_{i_0} = \emptyset$ **then** $L := L \setminus \{L_{i_0}\}$
  else
    reset $T_{i_0}^{r_k}$ to $T_{i_0}^{r_{k_0}}$
    update$(\widetilde{\Sigma})$
    $L := L \setminus \{L_{i_0}\}$
  end
until $L = \emptyset$

Algorithm 4.5: Algorithm for the computation of hierarchically reduced systems $\widetilde{\Sigma}$ using an optimized order of subsystem reductions.

Otherwise only the first entry is deleted from L_{i_0}. Then the procedure is repeated with the minimum of the first entries of all lists L_i until all these lists are empty.

Algorithm 4.5 shows the entire process in *pseudocode*. Note that this algorithm can further be improved, e.g. by an idea similar to the *clustering concept* from Section 4.3.2. If reductions of subsystems that cause a similar error on y are bundled in a cluster, costly multiple analyses $\widetilde{y} = \mathcal{A}(\widetilde{\Sigma}, u)$ of the so far reduced system $\widetilde{\Sigma}$ are avoided. They are performed only once after a whole cluster of subsystem reductions is executed. In case the error bound is still not violated, we can continue with the next cluster of subsystem reductions. Otherwise, however, all reductions in the current cluster have to be rejected and it has to be subdivided for further processing.

Another idea for further improvements on Algorithm 4.5 is the use of *approximate simulations* following the idea of the *one-step solver* briefly mentioned in Section 4.3.1. Thus,

we obtain an approximate solution $\widetilde{\widetilde{y}} \approx \widetilde{y}$ for the output of the so far reduced system $\widetilde{\Sigma}$ that can be used for the error check $E(y, \widetilde{y}) \leq \varepsilon$ instead of $\widetilde{y}$.

4.5. Reduction Algorithm Exploiting Circuit Hierarchy

This section combines all our thoughts and considerations from the previous sections in an algorithm for the reduction of the entire electrical circuit exploiting its structure on circuit level.

Since electrical circuits even nowadays are designed in a modular way using building blocks of network devices and substructures such as current mirrors and amplifying stages, assume that an electrical circuit Σ is given by a hierarchically segmented netlist description of interacting subsystems T_i as in (4.1):

$$\Sigma = (\{ T_i \,|\, i = 1, \ldots, m \}, S) .$$

Otherwise, the segmentation has to be made manually or by using *pattern matching* approaches to detect substructures in the entire circuit.

In a next step, for each subsystem T of Σ we choose an appropriate symbolic or numerical reduction technique together with a set of *reduction information* $\{ r_i \,|\, i = 1, \ldots, m_T \}$ and compute the corresponding sensitivity

$$s_T = \big((r_1, E(y, y_{T;r_1})), \ldots, (r_{m_T}, E(y, y_{T;r_{m_T}}))\big).$$

Based on these sensitivities, an optimized order of subsystem reductions, i.e. a *ranking of subsystem reductions* is set up. According to this order, the subsystems in Σ are reduced and replace the corresponding original ones in the entire circuit thus yielding a *hierarchically reduced* entire system $\widetilde{\Sigma}$.

This process is performed by the algorithm in the previous section. During the computations, numerical analyses have to be performed to check the error of the so far reduced entire system on the output of the original system Σ. If the user-specified error bound is exceeded, the current subsystem reduction has to be rejected. Otherwise, we can attempt to further reduce suitable subsystems in order to obtain a higher degree of reduction.

The entire algorithm for the hierarchical reduction of Σ is schematically shown in Figure 4.10.

Although we only worked on "level 0", i.e. we segmented Σ into subcircuits T_i without a further segmentation of a suitable subset thereof, the above algorithm can be recursively adapted to the more general case of subcircuits and subsystems in different levels of the resulting circuit hierarchy.

It turns out that error functions have a crucial influence on subsystem sensitivities and subsequent rankings of subsystem reductions (cf. remarks in Section 4.7). Therefore, the next section deals with different error functions used in this thesis.

4.6. Error Functions

In the reduction process of a system, it has to be checked several times whether the performed reduction steps are valid according to the user-given error bound specification.

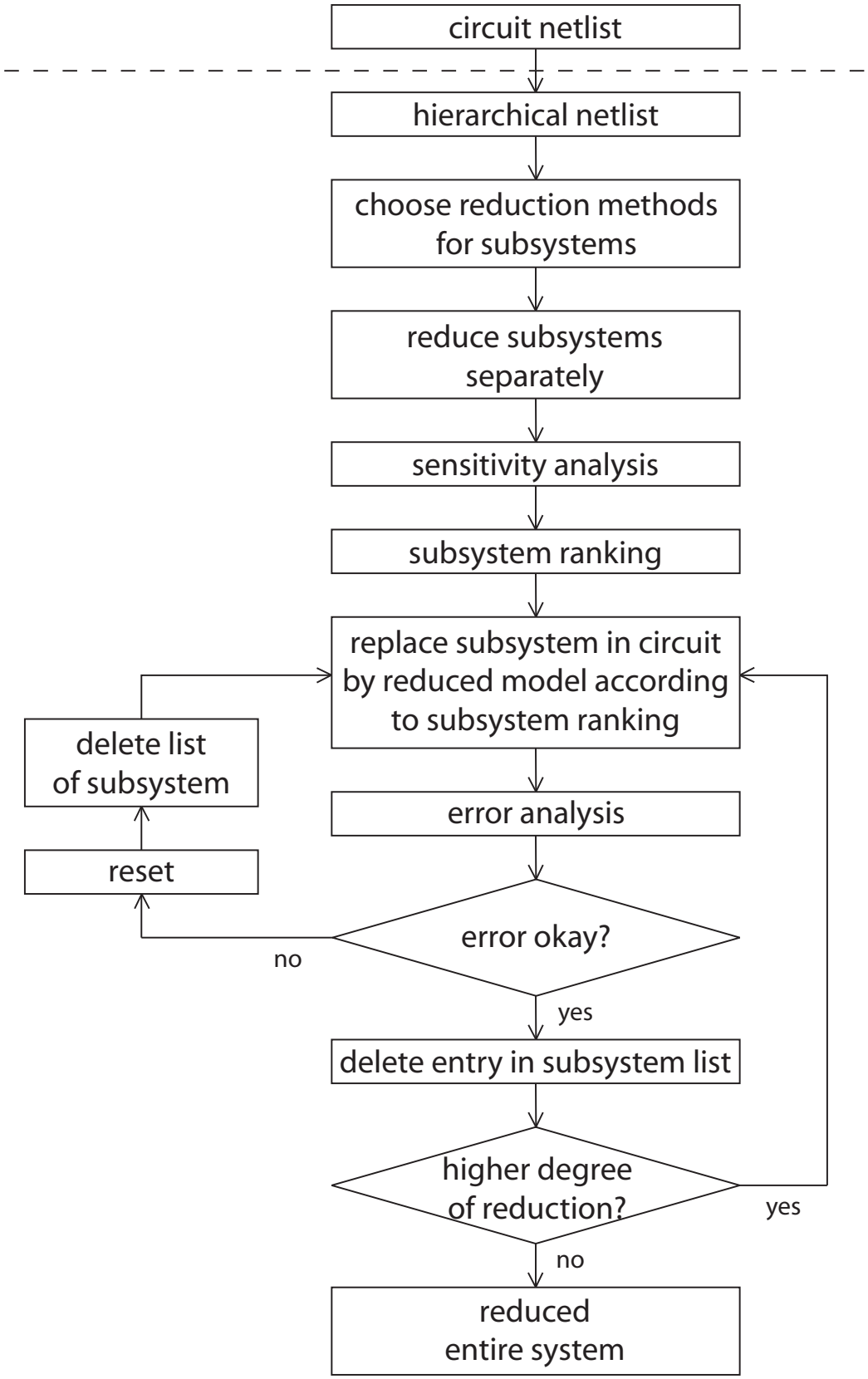


FIGURE 4.10. Schematic illustration of the algorithm for hierarchical model order reduction using sensitivities of subsystems.

Therefore, *error functions* are necessary that compare solutions of numerical analyses of different systems of DAEs to each other.

The choice of a suitable error function is of crucial importance for a system analysis and reduction problem. As a measure for the error, a variety of different mathematical criteria – also from an electrical-engineering point of view – can be taken into account. Since we never need the properties of a norm or a metric, we define *generalized error functions* simply as follows:

Definition 4.6. *Let F and G be systems of DAEs and $\mathcal{A}$ a numerical analysis method from Section 2.3. Let further $y_F = \mathcal{A}(F,u)$ and $y_G = \mathcal{A}(G,u)$ denote the solutions of F and G according to $\mathcal{A}$ and an input u. A function E that satisfies $E(y_F, y_G) \in \mathbb{R}_{\geq 0}$ is called a* ***generalized error function*** *for $\mathcal{A}$.*

Remark 4.7. *In* [**Hen**, Section 2.3]*, error functions ("Bewertungsfunktionen") measuring absolute errors are symmetric in y_F, y_G by definition. However, we here often measure relative errors scaled by a maximum value of y_F and, therefore, also have to deal with non-symmetric error functions. Nevertheless, by slight modifications these functions often can be turned into symmetric versions.*

The result of a system's reduction heavily depends on the chosen error function. For example, the result we would have obtained for the transient reduction of the op741 amplifier in Section 4.7 would not have been satisfying if we had taken the same error function as for the transient reduction of the differential amplifier in Section 4.1. In the latter case, the error function was simply given by

$$E(y_F, y_G) = \sup_{t \in \mathcal{T}} \|y_F(t) - y_G(t)\|, \tag{4.8}$$

where $\mathcal{T}$ is the considered time interval. This error function takes the pointwise largest difference between the two solutions $y_F = \mathcal{A}_{\text{tran}}(F,u)$ and $y_G = \mathcal{A}_{\text{tran}}(G,u)$ of the original and the so far reduced systems F and G on the entire time interval as a measure for the error. If y_F, y_G are smooth enough, this error function provides a good measure for the *absolute* error.

However, in the op741 amplifier example, y_F and y_G suddenly jump from one level to another one similar to step functions (cf. Figure 4.12 (b)). If in the reduced system G one of these jumps is only a little bit delayed (or too early), let's say at time t_d, the value of the error function (4.8) can be very large; while $y_G(t_d)$ is still on one level, $y_F(t_d)$ is already on the next one (or vice versa). Therefore, we have $E(y_F, y_G) \geq \|y_F(t_d) - y_G(t_d)\|$ which is a rather large value. Consequently, this error function is not well designed for the use in the op741 amplifier example in Section 4.7. In the following, we define some more suitable error functions which were used for diverse reductions in this thesis.

One of the most obvious error functions in the transient case is given by (4.8) and measures the maximum difference between $y_F = \mathcal{A}_{\text{tran}}(F,u)$ and $y_G = \mathcal{A}_{\text{tran}}(G,u)$ in $\mathcal{T}$. In order to obtain a measure for the *relative* error, we further divide the right side of (4.8) by the (non-zero) maximum amplitude

$$y_{F,\max} := \sup_{t \in \mathcal{T}} \|y_F(t)\|$$

on the considered time interval, thus yielding

$$E(y_F, y_G) = \sup_{t\in\mathcal{T}} \frac{\|y_F(t) - y_G(t)\|}{y_{F,\max}}. \tag{4.9}$$

However, in *Analog Insydes* as well as in other MOR and simulation software, y_F and y_G are given as data-based functions, i.e. pairs of sampling points and interpolation values. We therefore also introduce a discrete variant of (4.9) that chooses $y_{F,\max}$ and the maximum difference between y_F and y_G from a finite time grid $t_0 < \ldots < t_n$ instead of the entire (closed) interval $\mathcal{T} = [t_0, t_n]$:

$$\widetilde{E}(y_F, y_G) = \max_{i=0,\ldots,n} \frac{\|y_F(t_i) - y_G(t_i)\|}{\widetilde{y}_{F,\max}}, \qquad \widetilde{y}_{F,\max} := \max_{i=0,\ldots,n} \|y_F(t)\|. \tag{4.10}$$

Note that due to the scaling the error functions in (4.9) and (4.10) are *not* symmetric in y_F, y_G. As mentioned in Remark 4.7, a slight modification leads to symmetric versions, e.g. by replacing $y_{F,\max}$ by $\frac{1}{2}(y_{F,\max}+y_{G,\max})$. However, we abstain from doing so in order to scale only by the *original* maximum amplitude of the reference solution y_F.

In order to neglect *single peaks* of $\|y_F(t) - y_G(t)\|$, we define another transient error function. It averages the scaled differences of y_F, y_G over a time grid $t_0 < t_1 < \ldots < t_n$ contained in $\mathcal{T}$:

$$E(y_F, y_G) = \frac{1}{n+1} \cdot \left(\frac{\|y_F(t_0) - y_G(t_0)\|}{y_{F,\max}} + \ldots + \frac{\|y_F(t_n) - y_G(t_n)\|}{y_{F,\max}} \right). \tag{4.11}$$

Consequently, single peaks of $\|y_F(t) - y_G(t)\|$ causing large values of the previous error functions carry almost no weight in this definition. However, a drawback is as follows; if the difference $y_F(t) - y_G(t)$ is around zero for large parts of $\mathcal{T}$ and rather large for smaller *subintervals*, the *actual* error is "hidden" by the averaging error function and does not coincide with an *intuitive* impression.

In order to resolve this problem, we present an error function which was finally used for the reduction of the operational amplifier in Section 4.7. Similar to the one in (4.11), a time grid $t_0 < t_1 < \ldots < t_n$ with $[t_0, t_n] \subseteq \mathcal{T}$ is used. For each grid point t_i we then provide a small subinterval $\mathcal{T}_i \subseteq \mathcal{T}$ around t_i, e.g. $\mathcal{T}_i = (t_i - \delta, t_i + \delta)$ for $0 < \delta \ll 1$, and take $t \in \mathcal{T}_i$ such that the difference $\|y_F(t) - y_G(t_i)\|$ is minimized. Finally, we scale all these differences by $y_{F,\max}$ and take the maximum over all i as an estimate for the error between y_F and y_G:

$$E(y_F, y_G) = \max_i \left(\frac{\min_{t\in\mathcal{T}_i} \|y_F(t) - y_G(t_i)\|}{y_{F,\max}} \right). \tag{4.12}$$

Note that y_G is evaluated at the grid point t_i, while y_F is evaluated at an appropriately chosen time point $t \in \mathcal{T}_i$. For an implementation, we actually take a discrete set of points $\{t_{i,1}, \ldots, t_{i,m}\} \subseteq \mathcal{T}_i$ with a (user-)given m for the choice of t and obtain a discrete version of (4.12):

$$\widetilde{E}(y_F, y_G) = \max_i \left(\frac{\min_{j=1,\ldots,m} \|y_F(t_{i,j}) - y_G(t_i)\|}{\widetilde{y}_{F,\max}} \right). \tag{4.13}$$

In this definition, $\widetilde{y}_{F,\max}$ is defined as in (4.10). The big advantage of (4.12) and (4.13) for the use in the reduction of the operational amplifier in Section 4.7 is their acceptance

of *mutually delayed jumps* of y_F and y_G if the delays are not too big. Clearly, the smaller δ is chosen, the better y_G nestles to y_F. Using this error function, single peaks of $\|y_F(t) - y_G(t)\|$ with a duration of less than 2δ carry no weight. The side effects of (4.11), however, are avoided.

Further error functions that proved to be suitable for the reduction in Section 4.7 make use of $\mathcal{L}^p$-norms:

$$E(y_F, y_G) = \frac{\|y_F - y_G\|_{\mathcal{L}^p}}{\|y_F\|_{\mathcal{L}^p}} = \frac{\left(\int_{\mathcal{T}} \|y_F(t) - y_G(t)\|^p \, dt\right)^{1/p}}{\left(\int_{\mathcal{T}} \|y_F(t)\|^p \, dt\right)^{1/p}}. \tag{4.14}$$

Using a discrete version of this error function with $p = 2$ instead of (4.12) resp. (4.13) in Section 4.7, we obtain almost the same results for the reduction of an operational amplifier.

4.7. Analysis of the Operational Amplifier op741

As an example application of the algorithms in the previous sections of this chapter, we consider the operational amplifier op741 depicted in Figure 4.11. This type of amplifier is manufactured at low cost and, therefore, typically used in industry. We briefly describe the amplifier's hierarchical structure using interconnected subcircuits and shortly explain their functionality. Then we specify the settings for the following reductions of the entire amplifier circuit. We apply our new hierarchical approach as well as the usual non-hierarchical one that does *not* exploit the structure of the circuit. Furthermore, we perform these reductions for two different error functions from Section 4.6 and compare the corresponding results to each other. Finally, we improve the new approach by additionally taking the number of terms and equations of the respective reduced subcircuit

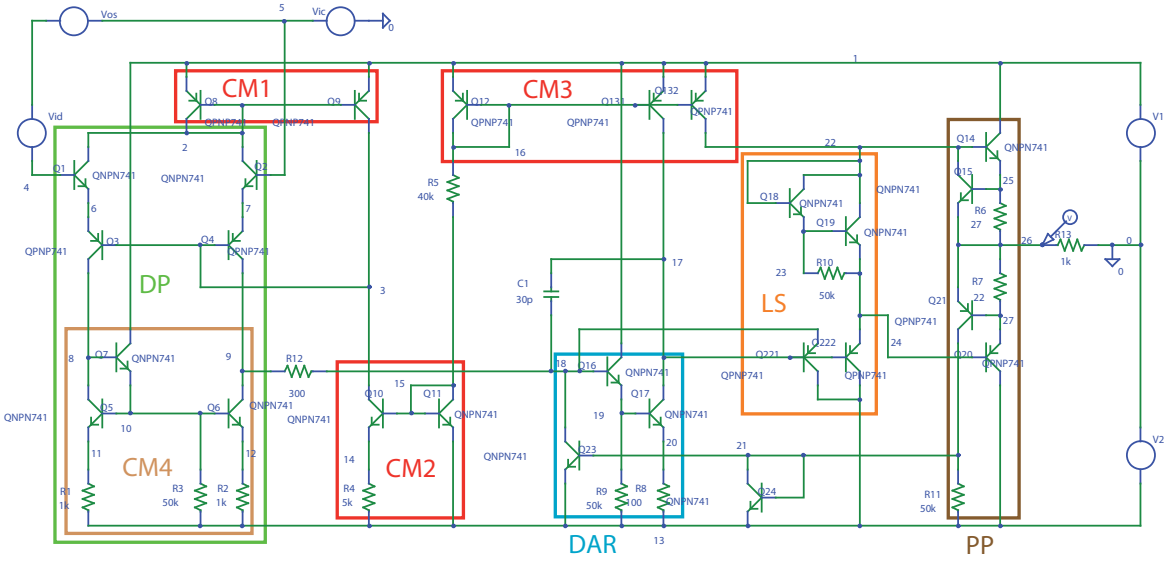


FIGURE 4.11. Operational amplifier op741 with seven subcircuits: a differential pair (DP) including a current mirror (CM4), three more current mirrors (CM1–3), a Darlington pair (DAR), a Level-Shift pair (LS), and a Push-Pull pair (PP) of transistors.

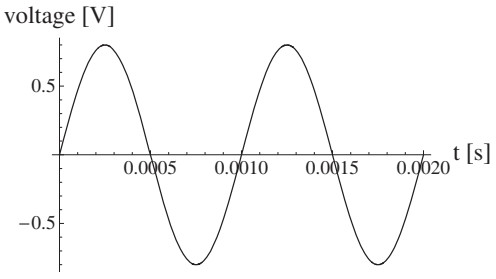
models into account. This then leads to *hybrid reduced* entire systems. By applying different input excitations to the reduced models, we conclude that the new approach can be employed with significant savings in time and, at the same time, yields reduced models of high accuracy. Furthermore, particularly the hybrid reduced models turn out to be very robust w.r.t. different inputs.

The amplifier can be divided into seven subcircuits according to their functionality, including some additional linear elements each. A more detailed explanation of the functionality of certain transistors or parts of the circuit is given in the appendix in Section C, here it is explained only briefly.

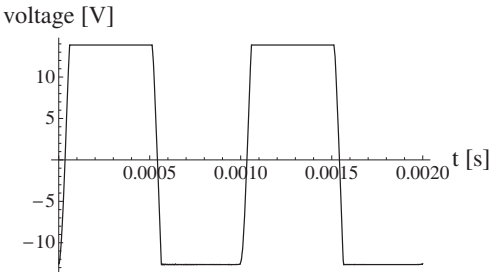
There are three current mirrors (CM1–3) which mirror the current from one side to the other, as the name already induces. Further, there is a first amplifying stage with the so-called differential pair (DP) of transistors, in which a fourth current mirror (CM4) is included. A second amplifying stage is given by a Darlington pair (DAR) of transistors. The amplifier's output stage is built by a Level-Shift pair (LS) and a Push-Pull pair (PP) of transistors. The output of the circuit is defined by the voltage potential `V$26` at node 26 on the right of Figure 4.11. The transient input is defined via the voltage source `Vid` on the left.

For any of the following transient reductions, we use a sine wave voltage excitation with 0.8 V amplitude and a frequency of 1 kHz on the time interval $\mathcal{T} = [0\,\mathrm{s}, 0.002\,\mathrm{s}]$ as input. It is shown in Figure 4.12 together with the corresponding output of the original amplifier circuit. Furthermore, we use full simulations to compute the term ranking of the corresponding systems, where the step sizes of the numerical solver are chosen automatically. As a measure for the error on the output of the amplifier, the $\mathcal{L}^p$-norm error function (4.14) for $p = 2$ and the newly created error functions (4.12) and (4.13), respectively, from Section 4.6 come to operation. The latter ones have been designed such that they accept a small delay in jumps as they occur in the output of the op741 amplifier. For both cases, we will accept an error of 10% on the amplifier's output `V$26`. Finally, the systems of equations in this section are set up by using *Gummel-Poon models* for the *bipolar junction transistors* in the amplifier circuit.

We apply modified nodal analysis in node voltage formulation to set up a system of DAEs that mathematically describes the behavior of the operational amplifier. This



(a) Input used for all reductions of the op741 amplifier in this section.



(b) Output of the original op741 system corresponding to the input on the left.

FIGURE 4.12. Input and output data of the operational amplifier.

leads to a system that consists of 215 equations and 1050 terms on level 0. A simulation with the above input yields the output shown in Figure 4.12 (b). The time costs for the simulation of the original system are ∼26 s.

4.7.1. Reduction using the $\mathcal{L}^2$-Norm Error Function. First, we apply the newly developed hierarchical reduction approach to the op741 amplifier. Then, we compare the obtained results to the usual non-hierarchical reduction approach that does not exploit the structure of the system on circuit level. In Section 4.7.3, we finally use this example to describe how the algorithm can further be improved.

4.7.1.1. ***Hierarchical Reduction.*** For the hierarchical reduction of the amplifier, we apply the algorithms from the previous sections. Therefore, we firstly simulate the entire system and record the voltage potentials at the nodes to which the terminals of the seven subcircuits (CM1–3), (DP), (DAR), (LS), and (PP) are connected. This means that for each subcircuit the remaining circuit serves as a test bench. Thus, we are able to apply the workflow from Section 4.2 for separate reductions of the seven subcircuits.

For each subcircuit of the operational amplifier we perform the same symbolic reduction techniques presented in Section 2.5.2.2, namely, *term reductions* and *algebraic manipulations*. In order to compute the *subsystem sensitivities* of the seven subcircuits, we provide a *sweep of error bounds*

$$sw = \{1, 2, 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100\},$$

whose elements have to be considered as percentage values. Thus, the *reduction information* r for each subsystem[8] is a vector of tuples

$$r = (\textit{"term reductions, algebraic manipulations"}, b),$$

where b is one of the error bounds in sw.

The time needed to compute the $91 = 7 \cdot 13$ reduced models of the seven subcircuits is 1 h and 21 min[9]. The additional computation time for the sensitivity analysis of the seven subcircuits is ∼38 min. We summarize the results in Table 4.2. While the first entry is the number of equations of the corresponding reduced subsystem, the second is its number of terms. The third entry is the error on the output `V$26` of the interconnected system, where exactly one subcircuit is replaced by the corresponding reduced subsystem. Larger errors sometimes are due to numerical solutions which miss the first "jump" of the original output. An entry "∞" means that the corresponding system could not be solved after several attempts[10]. In order to avoid very small numbers and since the error bounds in sw are given as percentage values, the same holds for the errors in Table 4.2. This allows for reading off the *subsystem sensitivities*

$$s_T = \big((r_1, E(y, y_{T;r_1})), \ldots, (r_{m_T}, E(y, y_{T;r_{m_T}}))\big)$$

as given in (4.7), since the errors $E(y, y_{T;r_j})$ are given by the corresponding third entries in the table. Thereby, T is one of the seven subsystems (CM1–3), (DAR), (DP), (LS),

[8] We do not further distinguish between subcircuits and subsystems in this section.

[9] The computations are performed on a Dual Quad Xeon E5420 with 2.5 MHz and 16 GB RAM.

[10] This might be due to an increased index of the corresponding system of DAEs.

error bound	CM1	CM2	CM3	DAR	DP	LS	PP
orig.	20	21	30	33	72	39	42
	77	81	124	143	320	167	190
1%	5	6	10	19	33	18	17
	14	18	32	65	182	65	46
	0.3075	23.6718	0.1317	0.8516	1.4563	1.6781	0.9580
2%	5	6	10	19	33	16	17
	13	18	30	65	178	58	46
	2.3093	23.6701	0.1955	0.8585	1.4487	1.6918	0.6310
5%	5	6	9	19	22	15	17
	12	18	28	64	75	42	44
	1.6930	0.3543	0.9478	0.8460	∞	0.8098	1.1075
10%	5	6	8	19	20	13	17
	12	18	28	61	68	33	44
	1.6930	0.9414	1.0130	0.7565	0.8579	∞	1.1075
20%	5	6	8	17	20	11	17
	12	18	26	46	66	28	43
	1.6930	0.3706	0.9958	0.8238	0.8072	1.0104	0.9619
30%	5	6	8	15	21	10	17
	12	18	24	37	66	27	43
	1.6930	0.3560	1.1204	0.3485	23.6771	1.0077	0.9619
40%	5	6	8	17	18	10	17
	12	18	24	43	43	27	43
	1.6930	0.9073	1.1204	0.8245	141.8790	1.0077	0.9619
50%	5	6	8	17	19	10	17
	12	18	19	43	43	27	43
	1.6930	0.9474	6.4630	0.8245	141.9140	1.0077	0.9619
60%	5	6	7	17	19	10	17
	10	18	16	37	43	27	43
	1.6930	0.8608	6.4670	0.2645	141.9150	1.0077	0.9619
70%	5	6	7	15	18	10	17
	10	18	16	36	41	25	43
	1.6930	0.8586	6.4670	0.8593	141.7710	0.9658	0.9619
80%	5	6	7	15	18	10	17
	10	18	16	34	39	25	43
	1.6930	0.9088	6.4670	0.8959	141.7710	1.0082	0.9619
90%	5	6	7	13	18	10	17
	10	18	16	29	39	25	43
	1.6930	0.3764	6.4670	0.2229	141.7710	1.0372	0.9619
100%	3	3	6	10	18	5	9
	0	0	2	6	24	0	11
	43.3565	100.0000	100.0000	143.3040	∞	143.3220	100.0000

TABLE 4.2. Separate reduction of the seven subcircuits of the op741. The table shows the number of equations (1st entry), the number of terms on level 0 (2nd entry), and the error measured by the $\mathcal{L}^2$-norm error function (3rd entry) as a percentage value.

and (PP), and r_j contains the reduction information (*"term reductions, algebraic manipulations"*, b_j), where b_j is the j-th error bound in the sweep sw.

Numbering the 13 error bounds in sw by \$1 = 1% up to \$13 = 100% and ordering the sensitivities of the subsystems w.r.t. the error on `V$26` increasingly yields the following seven lists:

CM1: (\$1, \$3–12, \$2, \$13)
CM2: (\$3, \$6, \$5, \$12, \$10, \$9, \$7, \$11, \$4, \$8, \$2, \$1, \$13)
CM3: (\$1, \$2, \$3, \$5, \$4, \$6–7, \$8, \$9–12, \$13)
DAR: (\$12, \$9, \$6, \$4, \$5, \$7–8, \$3, \$1, \$2, \$10, \$11, \$13)
DP: (\$5, \$4, \$2, \$1, \$6, \$10 and \$12, \$7, \$8, \$9)
LS: (\$3, \$10, \$6–9, \$11, \$5, \$12, \$1, \$2, \$13)
PP: (\$2, \$1, \$5–12, \$3–4, \$13)

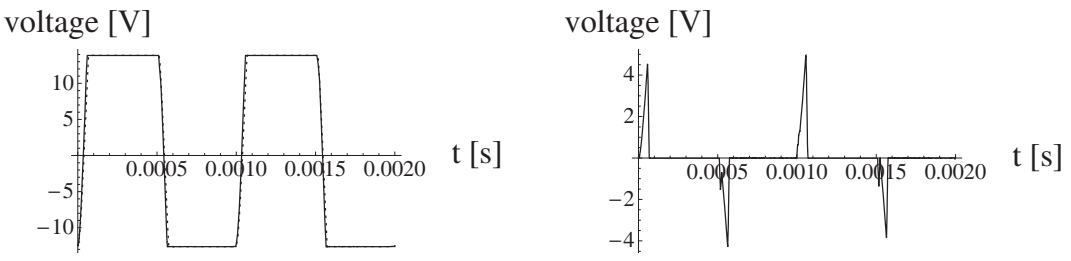
With these ordered lists we can proceed with the reduction process by using Algorithm 4.5. The computation time needed therefor is ∼25 min. The resulting order of reductions and the corresponding accumulated errors are listed in Table 4.3. We further included the development of the number of equations and terms of the so far reduced entire system in this list. Note that the error bound for the reduction of the entire system is 10%.

Thus, we finally end up with a configuration of reduced subsystems

$$(\mathsf{CM1}, \mathsf{CM2}, \mathsf{CM3}, \mathsf{DAR}, \mathsf{DP}, \mathsf{LS}, \mathsf{PP}) = (\$2, \text{orig.}, \$8, \$11, \$1, \$2, \$3). \tag{4.15}$$

where "orig." means that the *original* subsystem is used for the corresponding subcircuit. The entire system composed of the above models consists of 166 equations and 565 terms on level 0. The time costs of its simulation are ∼20.3 s and the error in comparison to the original output `V$26` is 6.8601% (cf. Figure 4.13).

By taking a closer look at Table 4.3, we see that some of the reduction steps do not seem to be reasonable, since they "re-increase" the number of equations and terms of the so far reduced entire system. For example, this is the case for the subsystem reductions from step 3 to step 4 or from step 9 to step 10. Although we actually come down to a



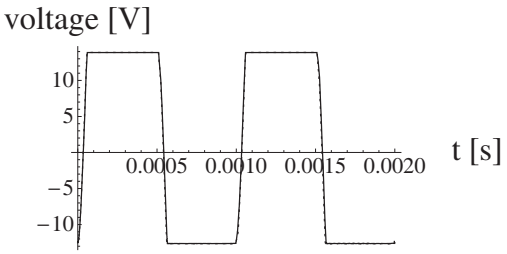
(a) Outputs of the original (solid) and the hierarchically reduced model (dotted).

(b) The corresponding error plot.

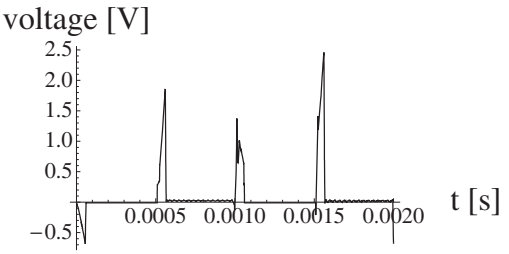
FIGURE 4.13. Comparison of the original and the hierarchically reduced op741 model in the $\mathcal{L}^2$-norm error function case.

	reduction step	eqns. / terms	accumulated error [%]	action
1	CM3 → \$1	207 / 982	0.1206	
2	CM3 → \$2	207 / 980	0.2003	
3	DAR → \$12	202 / 896	0.7258	
4	DAR → \$9	206 / 904	0.3998	
5	CM1 → \$1	200 / 859	0.8735	
6	DAR → \$6	198 / 859	0.9680	
7	CM2 → \$3	192 / 814	23.6715	> 10.0
8	CM2 → orig			reset CM2, delete list
9	PP → \$2	191 / 751	0.6255	
10	DAR → \$4	195 / 775	0.9459	
11	DP → \$5	167 / 569	1.0940	
12	LS → \$3	158 / 474	1.7760	
13	DAR → \$5	156 / 459	1.2950	
14	DAR → \$7	156 / 456	1.2532	
15	DAR → \$3	158 / 477	1.2991	
16	DAR → \$1	158 / 478	1.3137	
17	DP → \$4	158 / 480	1.3370	
18	DAR → \$2	158 / 480	1.3370	
19	DAR → \$10	154 / 451	1.9359	
20	DAR → \$11	154 / 449	1.9274	
21	CM3 → \$3	153 / 447	1.9200	
22	PP → \$1	153 / 447	1.9346	
23	PP → \$5	153 / 444	1.9258	
24	LS → \$10	148 / 427	2.3233	
25	CM3 → \$5	147 / 425	2.2778	
26	LS → \$6	147 / 427	1.9288	
27	LS → \$11	147 / 425	2.2377	
28	LS → \$5	148 / 428	1.9327	
29	CM3 → \$4	148 / 430	1.9230	
30	LS → \$12	147 / 427	2.2909	
31	PP → \$3	147 / 428	2.3725	
32	CM3 → \$6	147 / 424	2.3798	
33	DP → \$2	160 / 534	0.9919	
34	DP → \$1	160 / 538	0.9926	
35	LS → \$1	168 / 578	1.4670	
36	LS → \$2	166 / 571	1.4142	
37	CM1 → \$3	166 / 569	3.1130	
38	CM1 → \$2	166 / 570	2.2330	
39	CM3 → \$8	166 / 565	6.8601	
40	CM3 → \$9	165 / 562	12.0618	> 10.0
41	CM3 → \$8			reset CM3, delete list
42	DP → \$6	154 / 449	24.5622	> 10.0
43	DP → \$1			reset DP, delete list
44	CM1 → \$13	164 / 552	∞	> 10.0
45	CM1 → \$2			reset CM1, delete list
46	PP → \$13	158 / 532	100.0000	> 10.0
47	PP → \$3			reset PP, delete list
48	DAR → \$13	161 / 537	103.7530	> 10.0
49	DAR → \$11			reset DAR, delete list
50	LS → \$13	155 / 507	∞	> 10.0
51	LS → \$2			reset LS, delete list

TABLE 4.3. Hierarchical reduction using subsystem sensitivities, the $\mathcal{L}^2$-norm error function, and Algorithm 4.5.



(a) Outputs of the original (solid) and the non-hierarchically reduced model (dotted).



(b) The corresponding error plot.

FIGURE 4.14. Comparison of the original and the non-hierarchically reduced op741 model in the $\mathcal{L}^2$-norm error function case.

system containing 147 equations and 424 terms in step 32, we finally end up in step 39 with 166 equations and 565 terms.

This topic is further discussed in Section 4.7.3. The next section investigates the usual non-hierarchical reduction of the entire system without taking into account its hierarchical structure on circuit level.

4.7.1.2. ***Non-Hierarchical Reduction.*** In order to compare our results obtained by the hierarchical reduction approach to the usual *non-hierarchical* one, we symbolically reduce the original entire system containing 215 equations and 1050 terms on level 0. The reduction settings are as before, i.e. we apply *term reductions* and *algebraic manipulations* allowing an error of 10%.

Performing these two reduction techniques in the given order, we obtain a reduced entire system containing 97 equations and 593 terms. For the reduction, we need 10 h and 25 min[11]. The time costs for the simulation of this system are ∼16.0 s instead of ∼26.0 s for the original one. Furthermore, the error on `V$26` is 2.5124%. In Figure 4.14, the output of both the original and the non-hierarchically reduced system as well as the corresponding error plot are shown.

REMARK 4.8. *Note that the* ***direct*** *application of the methods presented in Section 2.5.2.2 to the entire system is limited due to complexity reasons. Hence, in general one might not succeed in applying these techniques to circuits of large size.*

4.7.2. Reduction using the New Error Function. We repeat both the hierarchical and the non-hierarchical reduction of the operational amplifier with exactly the same settings and specifications as before except for the choice of the error function. This time, we use the error function in (4.12) or (4.13), respectively, instead of the $\mathcal{L}^2$-norm error function given by (4.14) for $p = 2$.

4.7.2.1. ***Hierarchical Reduction.*** First, we compute the results for the reduction approach that exploits the hierarchical structure on circuit level. We proceed in exactly the same way as before using the algorithms in the previous sections of this chapter.

[11]The computations are performed on a Dual Quad Xeon E5420 with 2.5 MHz and 16 GB RAM.

After a simulation of the original system in order to obtain the voltage potentials of the nodes to which the subcircuits terminals are connected, we compute the $91 = 7 \cdot 13$ reduced models of the seven subcircuits. Thereby, we use the same symbolic reduction techniques and the same *sweep of error bounds sw* as before. The computational effort for this calculation is 1 h and 48 min. We need additional ∼26 min to compute the seven subsystem sensitivities. The results of the corresponding error analysis are summarized in Table 4.4. As in Table 4.2, the first entry is the number of equations of the corresponding reduced subsystem, the second is its number of terms. The third entry is the error on the output `V$26` of the interconnected system, where exactly one subcircuit is replaced by the corresponding reduced subsystem. It is given as a percentage value.

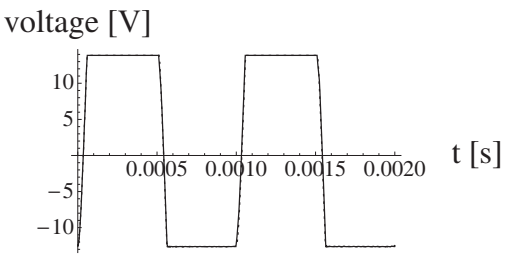
From the errors given in Table 4.4 one can read off the subsystem sensitivities. Numbering the 13 error bounds in *sw* by \$1 = 1% up to \$13 = 100% and ordering the sensitivities of the subsystems w.r.t. the error increasingly yields the following lists:

CM1: (\$3–12, \$1–2, \$13)
CM2: (\$1–12, \$13)
CM3: (\$8–9, \$1, \$6–7, \$3–4, \$2, \$5, \$10–12, \$13)
DAR: (\$1–3, \$5–12, \$4, \$13)
DP: (\$7, \$1–2, \$3, \$9–10, \$8, \$4–6, \$11–12, \$13)
LS: (\$1, \$4–9, \$3, \$2, \$10–12, \$13)
PP: (\$1–6, \$7–12, \$13)

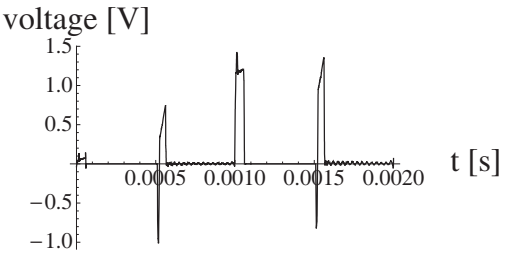
With these ordered lists, we use Algorithm 4.5 and proceed with the reduction process. The computation time needed therefor is ∼13 min. The reduction progress, the accumulated errors, and the development of the number of equations and terms of the so far reduced entire system are listed in Table 4.5. Also in this case the error bound for the reduction of the entire system is 10%.

Hence, we finally obtain a configuration of reduced subsystems

$$(\mathsf{CM1}, \mathsf{CM2}, \mathsf{CM3}, \mathsf{DAR}, \mathsf{DP}, \mathsf{LS}, \mathsf{PP}) = (\$3, \$1, \$1, \text{orig.}, \$4, \text{orig.}, \$7). \tag{4.16}$$



(a) Outputs of the original (solid) and the hierarchically reduced model (dotted).



(b) The corresponding error plot.

FIGURE 4.15. Comparison of the original and the hierarchically reduced op741 model obtained by using the new error function.

error bound	CM1	CM2	CM3	DAR	DP	LS	PP
orig.	20	21	30	33	72	39	42
	77	81	124	143	320	167	190
1%	5	6	8	18	37	20	23
	14	18	30	114	292	130	154
	0.0110	0.1516	0.0916	0.0528	0.0454	0.0109	0.0907
2%	5	6	8	18	37	20	23
	13	18	29	114	292	130	154
	0.0110	0.1516	0.1014	0.0528	0.0454	0.0497	0.0907
5%	5	6	8	18	23	20	23
	12	18	26	114	82	130	154
	0.0042	0.1516	0.0999	0.0528	0.0539	0.0232	0.0907
10%	5	6	8	10	22	10	23
	12	18	26	23	77	27	154
	0.0042	0.1516	0.0999	0.1038	78.2851	0.0121	0.0907
20%	5	6	8	8	23	10	23
	12	18	26	17	76	27	154
	0.0042	0.1516	0.1153	0.0547	78.2851	0.0121	0.0907
30%	5	6	8	8	23	10	23
	12	18	24	19	74	27	154
	0.0042	0.1516	0.0975	0.0547	78.2851	0.0121	0.0907
40%	5	6	8	8	22	10	13
	12	18	24	19	57	27	34
	0.0042	0.1516	0.0975	0.0547	0.0050	0.0121	0.0965
50%	5	6	8	8	22	10	13
	12	18	19	19	55	27	34
	0.0042	0.1516	0.0855	0.0547	14.2948	0.0121	0.0965
60%	5	6	8	8	22	10	13
	10	18	19	17	53	27	34
	0.0042	0.1516	0.0855	0.0547	12.0276	0.0121	0.0965
70%	5	6	7	8	22	9	13
	10	18	16	17	53	21	34
	0.0042	0.1516	4.5805	0.0547	12.0276	3.5716	0.0965
80%	5	6	7	8	22	9	13
	10	18	16	17	51	21	34
	0.0042	0.1516	4.5805	0.0547	190.2050	3.5716	0.0965
90%	5	6	7	8	22	9	13
	10	18	16	17	49	21	34
	0.0042	0.1516	4.5805	0.0547	190.2050	3.5716	0.0965
100%	3	3	6	7	18	5	12
	0	0	2	2	24	0	14
	105.0410	189.8860	189.0700	190.9600	∞	190.9830	99.9920

TABLE 4.4. Separate reduction of the seven subcircuits of the op741 using the new error function. Contained in this table are the number of equations (1st entry), the number of terms on level 0 of the reduced subsystem (2nd entry), and the error on `V$26` (3rd entry).

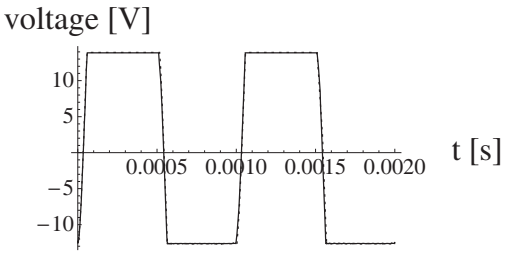
	reduction step	eqns. / terms	accumulated error [%]	action
1	CM1 → \$3	209 / 1003	0.0042	
2	DP → \$7	183 / 788	0.0051	
3	LS → \$1	179 / 781	78.2851	> 10.0
4	LS → orig			reset LS, delete list
5	CM1 → \$1	183 / 790	0.0256	
6	DP → \$1	198 / 1025	0.0515	
7	DAR → \$1	198 / 1026	0.0202	
8	DP → \$3	184 / 816	0.0583	
9	DAR → \$5	174 / 719	0.0970	
10	CM3 → \$8	164 / 638	78.3694	> 10.0
11	CM3 → orig			reset CM3, delete list
12	PP → \$1	173 / 719	0.0813	
13	PP → \$7	163 / 599	0.0871	
14	DAR → \$4	165 / 605	0.0717	
15	CM2 → \$1	159 / 560	0.0352	
16	DP → \$9	158 / 531	11.3933	> 10.0
17	DP → \$3			reset DP, delete list
18	PP → \$13	158 / 540	99.9920	> 10.0
19	PP → \$7			reset PP, delete list
20	CM1 → \$13	157 / 546	∞	> 10.0
21	CM1 → \$1			reset CM1, delete list
22	CM2 → \$13	156 / 542	∞	> 10.0
23	CM2 → \$1			reset CM2, delete list
24	DAR → \$13	156 / 539	190.9590	> 10.0
25	DAR → \$4			reset DAR, delete list

TABLE 4.5. Hierarchical reduction using subsystem sensitivities and the new error function (4.12) resp. (4.13).

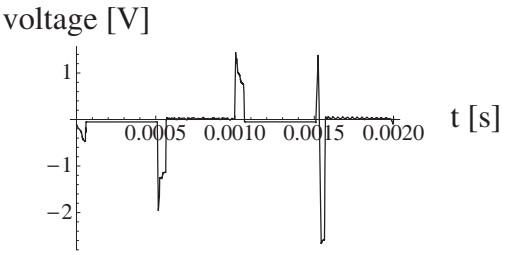
The reduced entire system composed of these models consists of 159 equations and 560 terms on level 0. The time costs of its simulation are ∼23.7 s and the error measured by the new error function and compared to the original output V\$26 is 0.0335% (cf. Figure 4.15).

Also in Table 4.5, some of the reduction steps do not seem to be reasonable. For example, avoiding the steps from 5 to 6 or from 13 to 14 might lead to a reduced entire system with less equations and a smaller number of terms. We further discuss this in Section 4.7.3.

4.7.2.2. ***Non-Hierarchical Reduction.*** In contrast to the above hierarchical reduction, we reduce the entire op741 amplifier without taking its hierarchy into account. In spite of the large size of the operational amplifier (see also Remark 4.8) we succeed in applying *term reductions* and *algebraic manipulations*. Allowing a 10% error bound, we come down to 80 equations and 405 terms on level 0. The result is a good approximation of the amplifier's output V\$26, cf. Figure 4.16. This system can be simulated in ∼9.5 s



(a) Outputs of the original (solid) and the non-hierarchically reduced model (dotted).



(b) The corresponding error plot.

FIGURE 4.16. Comparison of the original and the non-hierarchically reduced model of the op741 amplifier obtained by using the new error function defined in (4.12) resp. (4.13).

instead of ∼26.0 s for the original one. Further, the error on the original output `V$26` is 0.3718%.

However, the time effort for the reduction is more than 12 h[12]. This confirms that the application of "non-hierarchical symbolic methods" is limited to circuits with not too many nonlinear components.

4.7.3. Improvement. As already mentioned in the previous subsections, some steps in the two applications of Algorithm 4.5 do not seem to make sense. Our first attempt for deriving smaller hierarchically reduced systems is the deletion of inappropriate steps in the reduction process. In a second approach, we perform hierarchical reductions of the entire system based on structural information about the reduced models of the seven subcircuits. Finally, we conclude with a hybrid reduction approach by a further non-hierarchical reduction of the hierarchically reduced *interim* system.

4.7.3.1. *Avoiding Backwards Steps.* Table 4.3 shows that, e.g., from step 3 to 4 or from step 9 to 10 the number of both equations and terms increases. While in the first case the accumulated error decreases, in the latter it grows. In step 32, we have the smallest interim system that contains only 147 equations and 424 terms. Further subsystem replacements beyond step 32, however, increase again the size of the system.

The same holds for the hierarchical reduction process using the new error function. In Table 4.5, e.g. the steps from 5 to 6 and from 13 to 14 increase the number of terms and equations and, therefore, should be avoided. Nevertheless, the overall decay rate of the number of terms and equations in the latter case seems to be more even than in the case of the $\mathcal{L}^2$-norm error function.

Therefore, we might obtain smaller systems by avoiding "*backwards steps*", i.e. reduction steps where a subsystem is replaced by one that contains a higher number of equations or terms. This requires the additional monitoring of the number of equations and terms in the reduced models.

[12]The computation is performed on a machine with 8 Quad-core AMD Opteron 8384 "Shanghai" (32 cores in total) with 2.7 GHz and 512 GB RAM on a SuSE Linux 10.1 system.

By avoiding *backwards steps* during the reduction processes shown in Tables 4.3 and 4.5, we obtain smaller systems for both choices of the error function. For the $\mathcal{L}^2$-norm error function the resulting system is composed of the subsystems

$$(\mathsf{CM1}, \mathsf{CM2}, \mathsf{CM3}, \mathsf{DAR}, \mathsf{DP}, \mathsf{LS}, \mathsf{PP}) = (\$9, \text{orig.}, \$8, \$12, \$5, \$12, \$5). \tag{4.17}$$

and contains 145 equations and 407 terms. The computational cost for a simulation is ∼15.8 s and the error on `V$26` is 6.9893%. For the new error function, we obtain a subsystem configuration

$$(\mathsf{CM1}, \mathsf{CM2}, \mathsf{CM3}, \mathsf{DAR}, \mathsf{DP}, \mathsf{LS}, \mathsf{PP}) = (\$7, \$3, \$1, \text{orig.}, \$5, \text{orig.}, \$7). \tag{4.18}$$

The corresponding entire system consists of 156 equations and 527 terms on level 0, so the improvement in this case is rather small. The simulation of the system needs ∼20.0 s and the error on `V$26` is 0.0335%. The graphs of the output solutions of any of these systems are similar to the yet shown ones. We therefore relinquish further figures for the systems above.

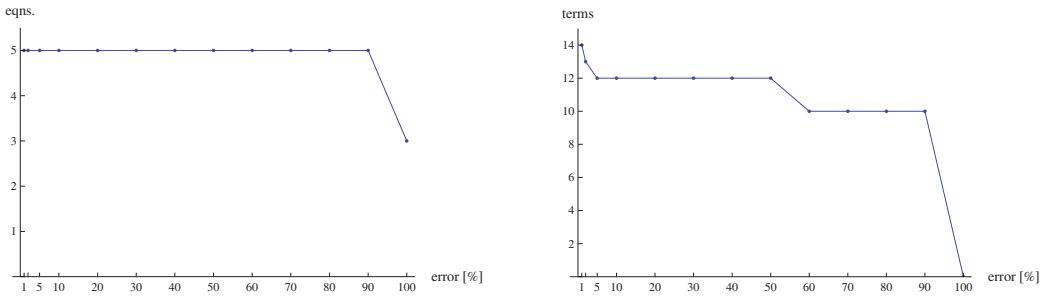
Note that the computational cost for deriving the above subsystem configurations is almost the same as before, since they only rely on an additional check of the number of equations and terms of the subsystem candidates. Another possibility to include this idea into the entire reduction process is the consideration of the number of equations and terms during the setup of the ordered lists from the subsystem sensitivities. If a reduced subsystem candidate causes a higher number of terms or equations than their predecessor in the list, it is simply left out. This also reduces the lists' length such that the entire reduction process can be accelerated.

However, this approach may also lead to actually feasible subsystem replacements that do not occur in the lists and, therefore, will never be executed. For example, in (4.16) and (4.18) we can actually replace (LS) by \$4–9, although the check of the accumulated error in step 3 of Table 4.5 fails, where the original (LS) subcircuit is tried to be replaced by \$4. In this case, this is due to the solution which misses the first "jump" of the amplifier's output. This means that the output solution of the so far reduced system did not start in −13 V and directly jumped to +14 V. Instead, it directly started on a level of +14 V, thus causing a rather large error.

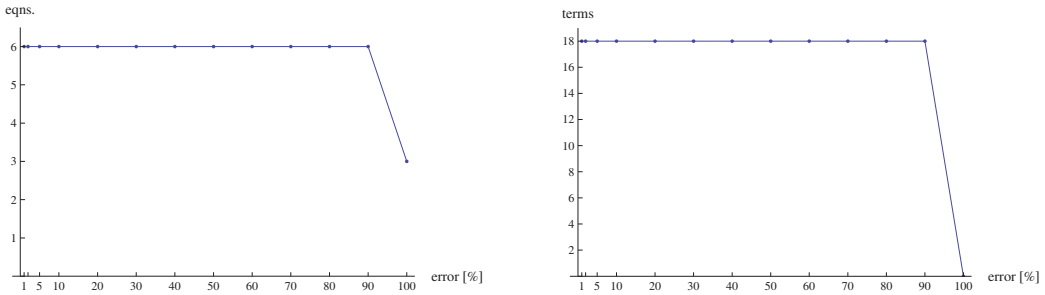
In this small example we see that settings such as step sizes of the numerical solver also play an important role during the entire reduction process.

4.7.3.2. ***Including Structural Subsystem Information.*** According to Algorithm 4.5, the ordered list of components of a subsystem sensitivity vector is deleted as soon as the check of the accumulated error caused by a replacement of the corresponding subsystem fails. This seems to be fatal particularly in those cases, where a subcircuit is not replaced by any of its reduced models at all.

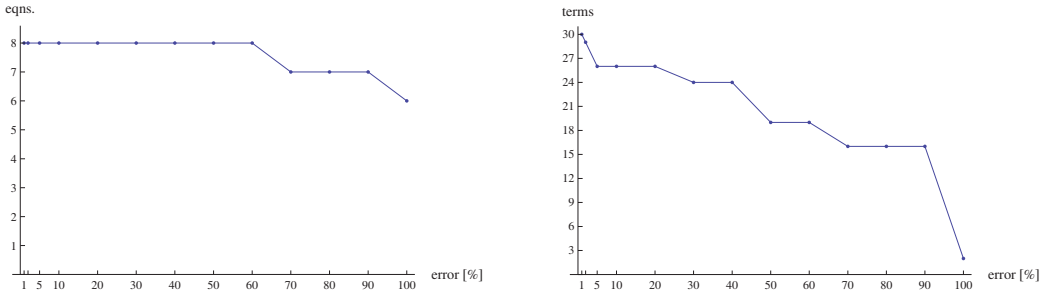
In order to replace (CM3) and (LS) in the subsystem configurations in (4.16) and (4.18), respectively, by a suitable reduced subsystem, we consider their *structure* in terms of the number of equations and terms. Using this *structural information*, we are able to derive systems of smaller size than before. The exact approach is described in the following by using the new error function.



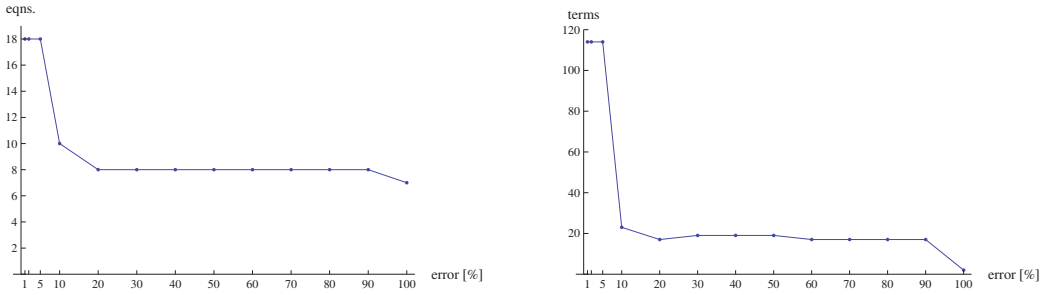
(a) Number of equations and terms for (CM1).



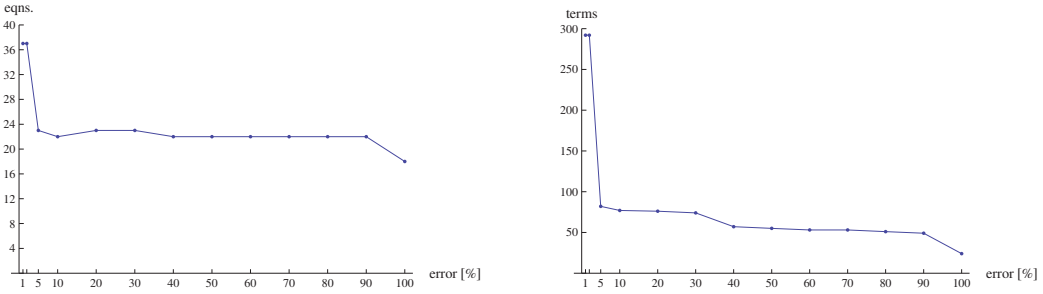
(b) Number of equations and terms for (CM2).



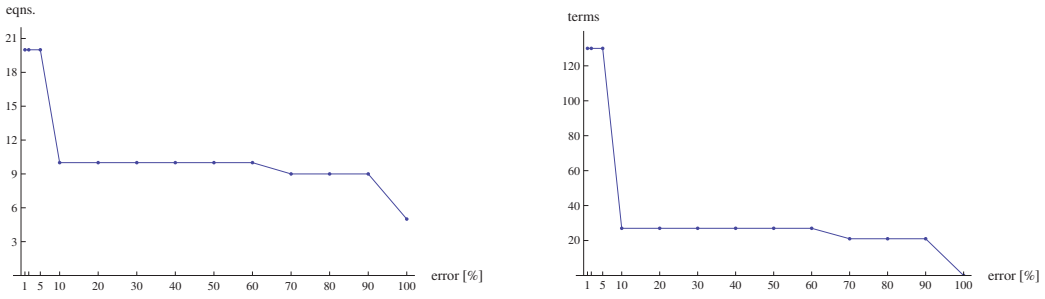
(c) Number of equations and terms for (CM3).



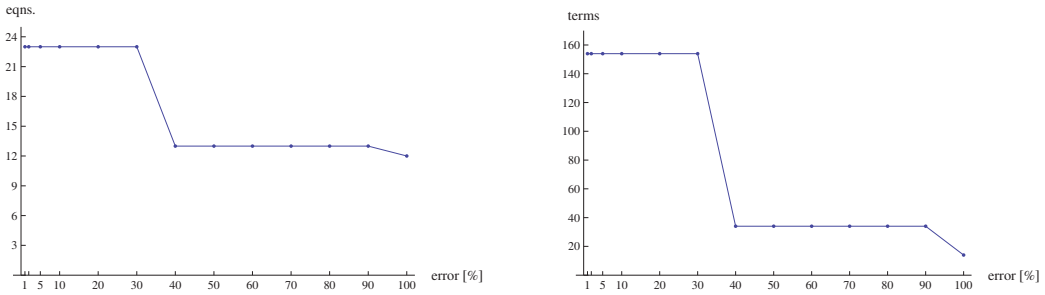
(d) Number of equations and terms for (DAR).



(e) Number of equations and terms for (DP).



(f) Number of equations and terms for (LS).



(g) Number of equations and terms for (PP).

FIGURE 4.17. Sensitivity diagrams for the seven subsystems of the operational amplifier op741.

Figure 4.17 shows diagrams of the number of terms and equations of all reduced models of the seven subsystems which, in general, decrease for growing bounds of the permitted errors. According to these diagrams, for the second current mirror (CM2) one obviously obtains always the same reduced model no matter which error bound $\varepsilon_{\mathsf{CM2}} \leq 90\%$ in sw is specified. Hence, in the original describing equations of subcircuit (CM2), there are only terms contained that either have an influence less than 1% or greater than 90% on the entire circuit's output V$26. Thus, we can try to replace (CM2) by its reduced model obtained by allowing any error bound $\varepsilon_{\mathsf{CM2}} \leq 90\%$ from the sweep sw.

For (CM1), the situation is almost the same, although the number of terms drops for a permitted error between 50% and 60%. Actually, during our manual sensitivity analysis, we observe that an error bound $\varepsilon_{\mathsf{CM1}} \geq 60\%$ yields a hierarchically reduced entire system, where the accumulated error is larger than 10%. Consequently, we try to replace (CM1) by a reduced model corresponding to an error bound $5\% \leq \varepsilon_{\mathsf{CM1}} \leq 50\%$.

For (CM3), the number of terms drops for almost every other increase within the values of the error sweep sw. Nevertheless, the number of equations is the same up to $\varepsilon_{\mathsf{CM3}} = 60\%$. Our analysis shows that an allowed error $\varepsilon_{\mathsf{CM3}} \geq 70\%$ yields bad results for the hierarchically reduced entire system. Therefore, we replace (CM3) by the model obtained by allowing an error of 60%.

According to the diagrams for the Darlington subsystem (DAR), the largest drops in the number of terms and equations occur for error bounds $\varepsilon_{\mathsf{DAR}} \leq 20\%$. If the reduced model resulting from $\varepsilon_{\mathsf{DAR}} = 20\%$ does not fail the error check on V$26, nor the one resulting from $\varepsilon_{\mathsf{DAR}} \leq 90\%$ probably will do so, since there are almost no changes in the number

subsystem	permitted error	equations
CM1	5 – 50%	20 → 5
CM2	1 – 90%	21 → 6
CM3	50 – 60%	30 → 8
DAR	20 – 90%	33 → 8
DP	10 – 40%	72 → 22
LS	10 – 60%	39 → 10
PP	40 – 90%	42 → 13

(23)

TABLE 4.6. Permitted errors for the hierarchical reduction of the operational amplifier op741 coming from an analysis of the number of terms and equations of its reduced subsystems. Entire systems composed of reduced subsystems with an error within the respective regions have an error of less than 10% compared to the original output `V$26`.

of equations and terms for higher error bounds. In fact, $\varepsilon_{\mathsf{DAR}} = 90\%$ turns out to be a suitable choice for a reduced model of (DAR).

For (DP), the biggest drops in the number of equations and terms also occur for error bounds $\varepsilon_{\mathsf{DP}} \leq 5\%$. Nevertheless, our manual sensitivity analysis shows that even with $\varepsilon_{\mathsf{DP}} = 70\%$ one still obtains a satisfying reduced model for (DP). However, the jumps of the hierarchically reduced system's output are a little delayed in comparison to the original circuit's output. Better results with the same number of equations are obtained for $\varepsilon_{\mathsf{DP}} \in \{10\%, 40\%\}$, so we restrict ourselves to reduced models of (DP) obtained by permitting these error bounds.

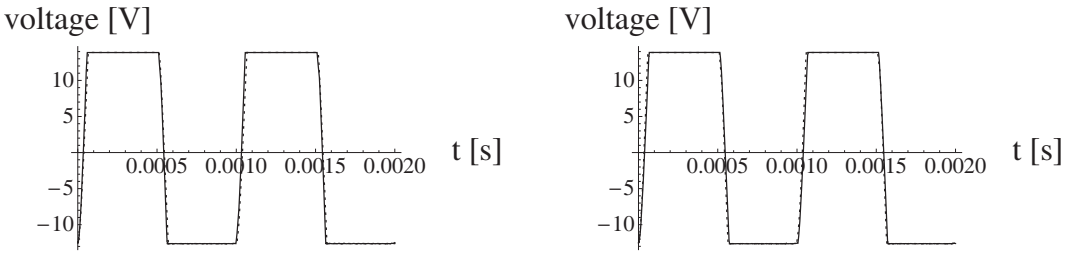
Finally, for the last two subsystems (LS) and (PP), the intervals $10\% \leq \varepsilon_{\mathsf{LS}} \leq 60\%$ and $40\% \leq \varepsilon_{\mathsf{PP}} \leq 90\%$ between two respective drops of the numbers of equations and terms turn out to be the regions for appropriate error bounds to derive reduced models.

Table 4.6 shows a summary of the above discussion and provides the resulting subsystem configurations of suitable hierarchically reduced entire systems. If for any of the amplifier's subsystems a respective higher bound from *sw* is chosen, the resulting hierarchically reduced system's output has a large error of up to 200% compared to the original system's output `V$26`.

Choosing the largest possible error bound for all seven subsystems corresponds to the subsystem configuration

$$(\mathsf{CM1}, \mathsf{CM2}, \mathsf{CM3}, \mathsf{DAR}, \mathsf{DP}, \mathsf{LS}, \mathsf{PP}) = (\$7, \$8, \$12, \$9, \$12, \$9, \$12), \tag{4.19}$$

thus yielding a system that contains 132 equations and 336 terms only. The error on the original system's output `V$26` is 0.0828% and the time costs for a simulation are ∼13.1 s.



(a) Using the new error function.

(b) Using the $\mathcal{L}^2$-norm error function.

FIGURE 4.18. Outputs of the original (solid) and the hierarchically reduced system (dotted) obtained by using information about the subsystems' number of equations and terms.

Proceeding in a similar way using the $\mathcal{L}^2$-norm error function, we obtain a subsystem configuration

$$(\mathsf{CM1}, \mathsf{CM2}, \mathsf{CM3}, \mathsf{DAR}, \mathsf{DP}, \mathsf{LS}, \mathsf{PP}) = (\$9, \$3, \$8, \$12, \$5, \$12, \$5). \tag{4.20}$$

Note that this is the same as the configuration ($12, $12, $8, $12, $5, $12, $12) (cf. Table 4.2). The corresponding entire system consists of 139 equations and 362 terms on level 0, the error on `V$26` is 7.1568%, and the time costs for a simulation are ∼11.4 s. Figure 4.18 shows that the outputs of the original and the reduced systems still fit quite well in both cases. Since we computed the two systems completely manually, we relinquish the indication of time effort.

4.7.4. A Hybrid Reduction Approach. In order to further reduce the complexity of the obtained systems, i.e. their number of equations and terms, we apply *term reductions* and *algebraic manipulations* in the usual way. Thus, we obtain hybrid hierarchically and non-hierarchically reduced entire systems of lower complexity and with less time effort than using the usual non-hierarchical reduction for simplifying the original system of equations with 215 equations and 1050 terms.

We already applied the hierarchical reduction to the original system using an error bound of 10%. Therefore, a further reduction of the interim systems using the same error bound finally might lead to systems whose error on the original output `V$26` is larger than 10%. In fact, a further reduction by using the above methods and an error bound of 10% in the case of the system containing 166 equations from Section 4.7.1.1 yields a system with an error of 11.2945% on the original output. Furthermore, this error is not due to delayed jumps, but rather to two different top levels of the pulse-shaped output. While the original output's top level is around 14 V, the simplified system's output has top levels of approximately 15.8 V.

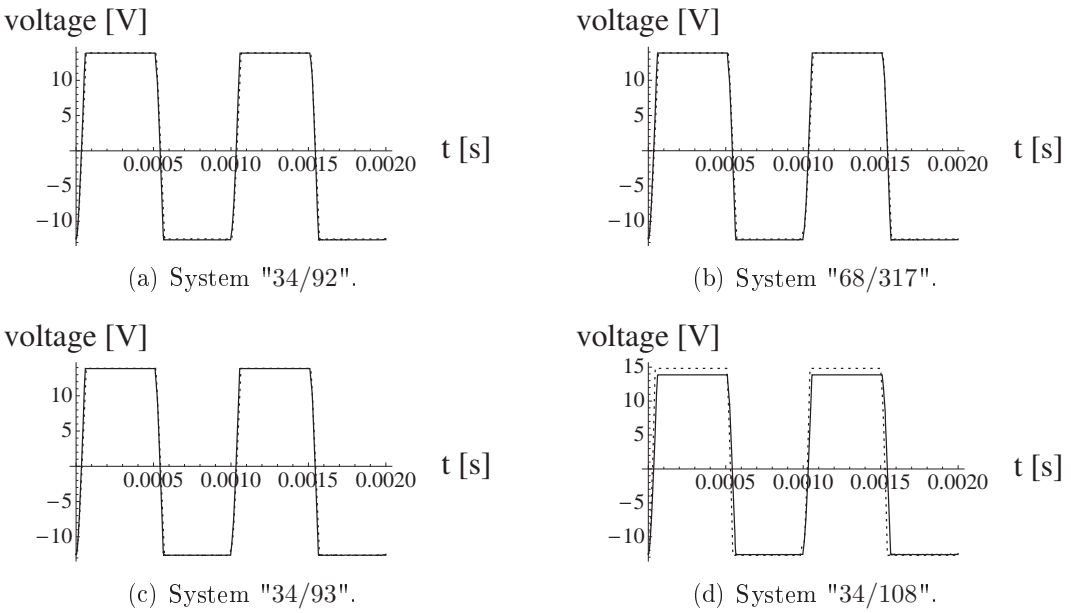
However, by using an error bound of 9% instead of 10%, the system mentioned above can be further reduced and the error on the output even decreases from 6.8601% to 5.2629%. Nevertheless, for one of the four systems that we consider in this section, this bound is still not suitable and the overall error bound of 10% is exceeded.

$\mathcal{L}^2$-norm error function			new error function		
139/362		34/92	132/336		34/93
∼11.4 s	⟶	∼2.2 s	∼13.1 s	⟶	∼2.0 s
7.1568%		5.6771%	0.0828%		5.3191%
166/565		68/317	159/560		34/108
∼20.3 s	⟶	∼2.5 s	∼23.7 s	⟶	∼0.8 s
6.8601%		5.2629%	0.0335%		15.6523%

TABLE 4.7. The hierarchically reduced systems are further reduced non-hierarchically allowing an error of 9%. Thus, one finally obtains hybrid reduced models of the original operational amplifier op741. The table contains the corresponding systems' number of equations and terms (cf. Notation 4.9), the time costs of a simulation, and the corresponding error on the entire system's output `V$26`.

Table 4.7 provides an overview of the four systems to which we apply hybrid reductions. Thereby, the systems are identified by their number of equations and terms.

NOTATION 4.9. *In the following, we identify the original system and its different reduced counterparts by their number of equations and terms n_e and n_t, respectively. The corresponding system then is denoted by "n_e/n_t".*



(a) System "34/92". (b) System "68/317".

(c) System "34/93". (d) System "34/108".

FIGURE 4.19. Outputs of the original (solid) and the reduced system (dotted) obtained by using the hybrid approach.

Hence, the original system with 215 equations and 1050 terms is denoted by "215/1050". System "166/565" is the system obtained in Section 4.7.1.1 by using the hierarchical reduction approach and the $\mathcal{L}^2$-norm error function, while system "159/560" is the one from Section 4.7.2.1 using the newly designed error function. Systems "139/362" and "132/336" correspond to the systems from Section 4.7.3.2, where we used information about the subsystems' number of equations and terms to derive hierarchically reduced entire systems. The table contains entries including the systems' number of equations and terms, the time costs for simulations, and the error on `V$26` of the original system given as a percentage value. We did not include the computational effort that is necessary for the additional non-hierarchical reductions. It depends on the exact settings of the numerical solver and ranges from several minutes to 1.5 h. Usually, more than 90% of this time is needed for the computation of the transient term ranking.

According to Figure 4.19, the hybrid reduced systems' outputs fit the original one quite well except for system "34/108".

4.7.5. Model Quality Check. In order to check the quality of the different reduced entire systems and, hence, the aptitude for employing them in different application areas, we apply some other input excitations. Then, we compare the corresponding outputs of the reduced systems and the original one to each other.

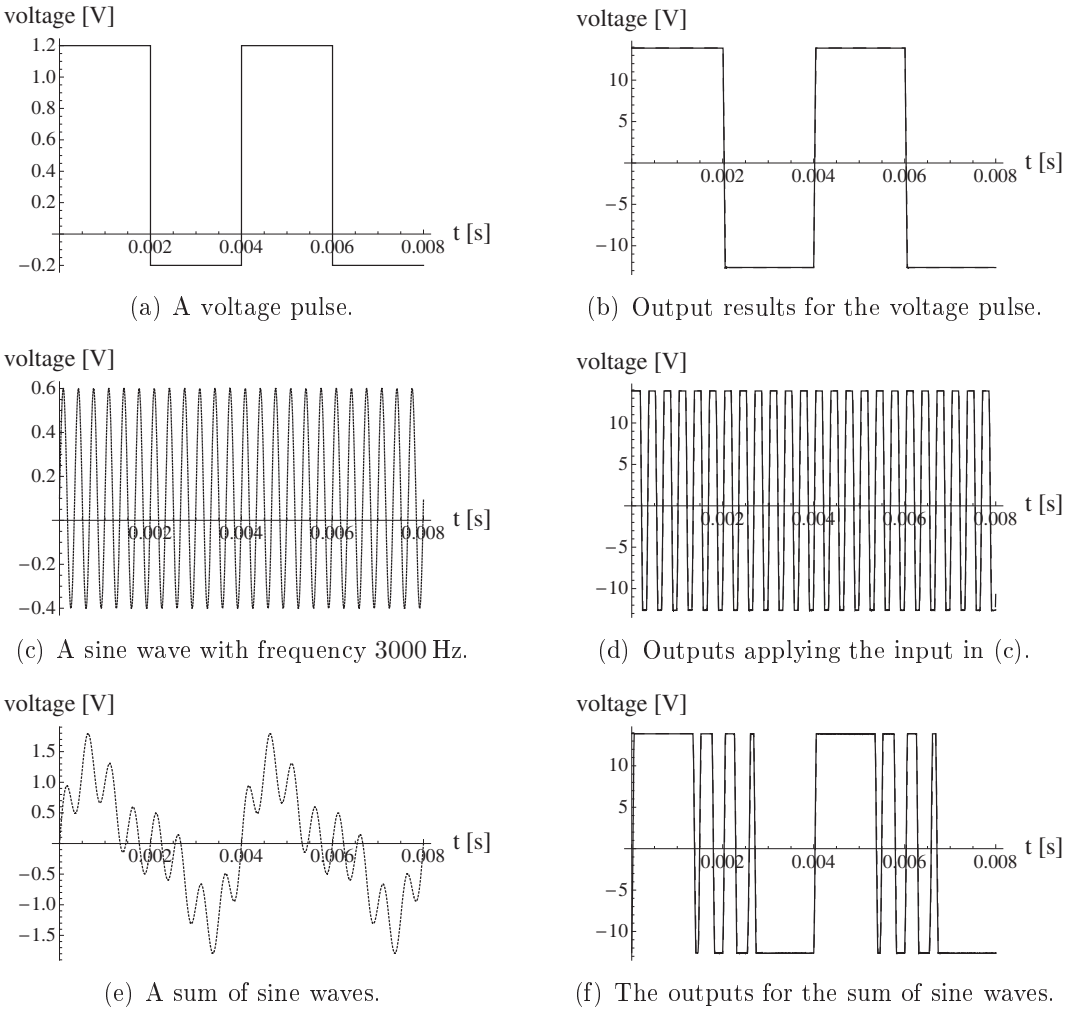
Figure 4.20 shows three different input functions on an extended time interval $\mathcal{T}_{\text{ext}} = [\,0\,s, 0.008\,s\,]$ which will be used for this purpose. The respective output results for both the original and the reduced systems are also shown. The first input function in Subfigure (a) is a pulse excitation. The function in Subfigure (c) is a sine wave with a frequency of 3,000 Hz and a maximum amplitude of 0.5 V that is slightly shifted in direction of the positive voltage axis. The function in Subfigure (e) is a sum of three sine waves, namely

$$\sin(2\pi \cdot 250\,t) + 0.5\sin(2\pi \cdot 500\,t) + 0.5\sin(2\pi \cdot 2000\,t).$$

The plots in Figure 4.20 correspond to simulations of the hybrid reduced system "34/93". Since the respective plots of the remaining reduced systems are very similar, we relinquish the inclusion of further figures.

However, we have not been able to solve all the remaining systems for all of the three input excitations above (see also Remark 4.11). For example, the non-hierarchically reduced systems "97/593" and "80/405" as well as the hierarchically reduced system "166/565" could not be solved if the voltage pulse in Figure 4.20 (a) was applied. Moreover, the hybrid reduced system "34/108" was not solvable for any of the three inputs. Besides this, for certain input excitations from above, some of the reduced systems need more time to be solved than the original one. For example, while the original system "215/1050" needs ∼1 min 44 s to be solved if the sum of sine waves in Figure 4.20 (e) is applied, systems "166/565" and "80/405" need ∼1 min 47 s and ∼2 min 5 s, respectively.

Nevertheless, even with a view towards reduced simulation time, the hybrid reduced systems "68/317", "34/92", and "34/93" yield very good results for all three inputs above. System "34/92" for example can be simulated up to 19 times faster than the original one by using the three input functions above (cf. Table 4.8). Also from an



(a) A voltage pulse.

(b) Output results for the voltage pulse.

(c) A sine wave with frequency 3000 Hz.

(d) Outputs applying the input in (c).

(e) A sum of sine waves.

(f) The outputs for the sum of sine waves.

FIGURE 4.20. *Left:* Three different input functions for testing the quality of the reduced models. *Right:* The corresponding output results of the original (solid) and the reduced systems (dashed).

accuracy point of view, according to Figure 4.20, the hierarchical and hybrid approaches yield good models for the op741 amplifier circuit with a much lower level of complexity than the original circuit.

REMARK 4.10. *The obtained hierarchically and hybrid reduced models are not only better than the non-hierarchically reduced ones w.r.t. simulation time, they are also more robust w.r.t. other input excitations; although the symbolic reduction was performed by applying the smooth sine wave excitation from Figure 4.12 (a), these models yield good approximations also for the highly non-smooth voltage pulse from Figure 4.20 (a).*

REMARK 4.11. (Observation)
The symbolic methods guarantee the user-specified accuracy for the input excitation used during the reduction process. The applicability of other inputs of the so far reduced systems, however, has not been a criterion. Therefore, numerical problems may occur while solving the obtained models for another input. Despite this, we also obtain reduced

system	voltage pulse	3 kHz sine wave	sum of sine waves
215 / 1050	106 s	273 s	104 s
159 / 560	55 s	250 s	114 s
68 / 317	36.5 s	65 s	35.9 s
34 / 92	6.6 s	14.1 s	10.5 s

TABLE 4.8. Comparison of the computational cost of simulations of three selected reduced models and the original system.

models that yield good approximations of the original system for other input excitations. This suggests that symbolic reduction methods maintain the "important parts" of the original system.

Furthermore, the time needed for simulations of the so far reduced systems has not been monitored during the reduction process (which in principle is possible, though). Nevertheless, according to experience, the reduced systems in general can be simulated faster than the original one (cf. Table 4.8 and Figure 4.21).

4.7.6. Conclusion. Figure 4.21 summarizes the different reduction approaches and the resulting systems that have been investigated in this section.

Explanation of Figure 4.21. The boxes in this figure contain the number of equations and terms of the corresponding systems, they are used for their identification (cf. Notation 4.9). Moreover, the corresponding errors on the output `V$26` of the original op741 amplifier and the simulation time costs are contained in the boxes. Consequently, the topmost box corresponds to the original system, which contains 215 equations and 1050 terms. Applying the input excitation from Figure 4.12 (a), it can be simulated in ∼26.0 s. The leftmost branch means that the non-hierarchical reduction took ∼10 h 25 min if the $\mathcal{L}^2$-norm error function was used. It resulted in a system with 97 equations and 593 terms. The simulation time of this reduced system is ∼16.0 s with an error of 2.51% measured by the $\mathcal{L}^2$-norm error function. If the new error function is used, the result is slightly better (box at the rightmost branch).

The best results correspond to the two boxes in the middle of the bottom. During the hierarchical reduction, we here used additionally structural subsystem information (cf. Section 4.7.3.2) which is given by the reduced subsystems' number of equations and terms. Subsequently, a non-hierarchical symbolic reduction was applied. Therefore, we call the entire reduction a *hybrid* reduction. The total time needed therefor ranges from ∼2.5 h to ∼4 h and depends on the exact settings of the numerical solver. The resulting systems are rather small (34 equations, 92 terms) and the simulation time is about 2 s with an error of only 5.68% resp. 5.32%.

According to this figure, the hierarchical and hybrid reduction approaches could be applied successfully and with significant savings in computational time to the operational amplifier op741. While the usual non-hierarchical approach needed more than 10 hours,

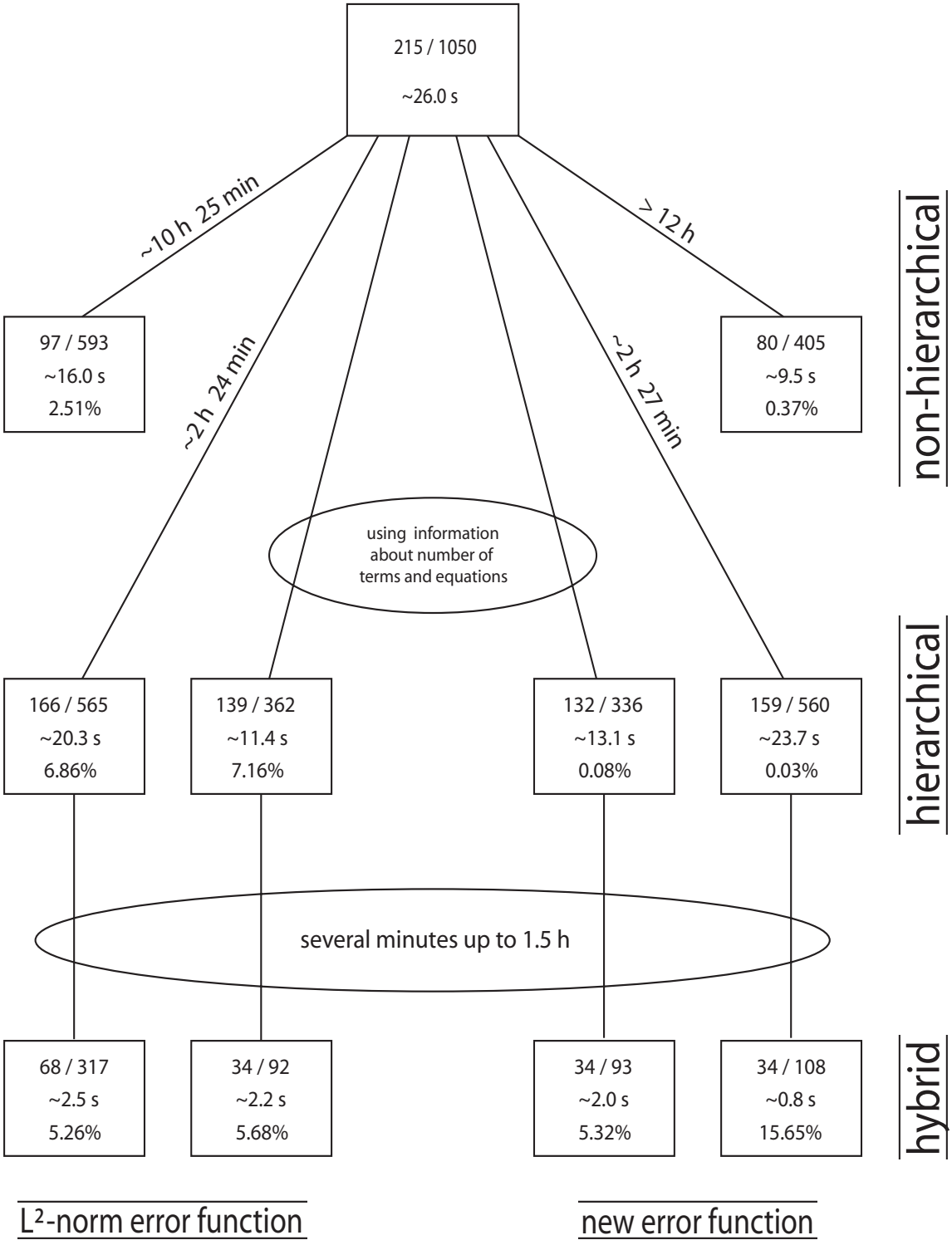


FIGURE 4.21. Summary of the reduced models of the op741 amplifier obtained in this section via different reduction approaches. The boxes contain the number of equations and terms of the reduced models, the time costs of a simulation using the original sine wave excitation in Figure 4.12 (a), and the error on the output `V$26` of the original operational amplifier op741.

the hybrid reduced models could be derived in less than half of that time. Nevertheless, the accuracy of all reduced models is almost the same except for one case. Furthermore, the hybrid reduced model "34/93" (cf. Notation 4.9) can be simulated in ∼2.0 s using the original sine wave excitation, which is only 8% of the time needed for the original system. At the same time, the best non-hierarchically reduced model "80/405" needs ∼9.5 s, which is 37% of the original simulation time of 26.0 s.

According to Figure 4.20, the hierarchical and hybrid approaches yield good models for the op741 amplifier circuit with a much lower complexity than the original circuit[13]. Applying three further test inputs, the hybrid reduced systems "34/92" and "34/93" turned out to be quite a lot faster to simulate than the non-hierarchically reduced ones. Moreover, although the symbolic reductions have been performed by applying a smooth sine wave excitation, the two systems proved to be robust even w.r.t. a highly non-smooth pulse excitation.

Remark 4.12. *Note that we performed* ***symbolic*** *reduction in the* ***transient*** *case. Furthermore, we used full simulations for the term ranking, i.e. for measuring the influence of single terms on the output of the original system. Using approximate simulations might accelerate the entire reduction process, in the hierarchical as well as in the non-hierarchical case.*

The corresponding reduced systems for both the new and the $\mathcal{L}^2$-norm error functions are very similar. Moreover, considering the amplifier's subcircuits (CM1) and (CM2) (cf. Figure 4.11), the number of equations and terms of the corresponding reduced counterparts are equal for both choices of the error function (see Tables 4.2 and 4.4). For the subcircuits (CM3) and (LS) they are at least very similar.

Despite this, the op741 example still shows that a good choice of the error function plays an important role. While the new error function led to errors that either are very small or rather large, the errors obtained by using the $\mathcal{L}^2$-norm error function are more staged. Hence, for certain subcircuits, using the new error function with *different* error bounds sometimes led to the *same* reduced subsystem. For example, any error bound between 1% and 90% resulted in the same reduced subsystem of the (CM2) subcircuit. For some subcircuits this led to only a small number of reduced models. As a consequence, a smaller number of reduced model replacements had to be checked during the hierarchical reduction process. Therefore, the algorithm needed only 25 steps, while it took 51 steps using the $\mathcal{L}^2$-norm error function (cf. Tables 4.3 and 4.5).

However, besides the advantage of a faster processing, this might also lead to subcircuits that will not be replaced by a reduced subsystem at all. If there is a choice of "many" reduced subsystems for a subcircuit, the algorithm can find a suitable one "stepwise". On the other hand, if there is only a small number of subsystems for a subcircuit, "small steps" are maybe impossible and the first attempt to replace a subcircuit fails. This means that the algorithm does not replace the corresponding subcircuit at all. We therefore conclude that the well-established $\mathcal{L}^2$-norm error function is adequate for the

[13]The resulting plots of the corresponding reduced models are very similar. Therefore, we included the plots only once.

more general case, although the new error function is maybe better adjusted to the op741 example at hand and led to smaller systems.

To sum up, the application of the hierarchical reduction approach to the operational amplifier showed the large potential provided by this approach. If one wants to gain insight into functional relations among circuit parameters and components only locally, the remaining parts of the circuit may also be reduced by using numerical methods. This can highly increase the entire reduction process in several magnitudes as it has been shown in Section 4.1.

CHAPTER 5

Implementations

In this chapter, we give an overview of the prototypical implementations made within this thesis and provide brief explanations of the corresponding commands. Basic knowledge of *Mathematica* and *Analog Insydes* is required thereby, we refer to [**MMA, AIman**] for an introduction.

Since the new procedures and commands use some of the already existing functionality implemented in *Analog Insydes*, most of the options refer to internal calls such as the `NDAESolve`-command which solves a system of DAEs.

In the first section of this chapter, we present a new data structure that has been implemented for the use with linear time-invariant systems given in state space formulation. We further present two new procedures that allow for conversions between this new data structure and the suitable existing data structures already available in *Analog Insydes*. The second section then deals with an implementation of the Arnoldi iteration that operates on the new data structure. In section three of this chapter, we introduce two models for transmission line components in a circuit as well as some models for general circuit components that can be modelled via LTI systems in state space formulation. The new implementations in the first three sections of this chapter have been used successfully for the reduction of the differential amplifier in Section 4.1.

Section four then presents implementations that partly realize our hierarchical reduction approach as described in Chapter 4. They have been applied successfully to the operational amplifier op741 in Section 4.7. The fifth section finally provides a brief summary of further implementations such as various error functions from Section 4.6 and extensive development environments used throughout this thesis. Furthermore, these environments also contain the manual checks of those parts of the hierarchical reduction approach that have not yet been implemented.

5.1. GetStateSpace and GetDAE

Symbolic systems of equations in *Analog Insydes* are encapsulated in a data structure

`DAEObject[`*mode*`][{`*eqs, vars*`},{`*dae_options*`}]`.

Besides the equations and variables occuring in the system of equations, it contains information such as design points, reference solutions, initial values, and term rankings. These specifications are stored in *dae_options*. Furthermore, *mode* indicates whether one deals with a linear (AC), a static (DC), or a transient system.

However, we also want to apply numerical reduction methods such as *balanced truncation* or the *Arnoldi iteration* from Section 2.5 to LTI state space systems

$$\begin{aligned} E\dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t) + Du(t). \end{aligned} \tag{5.1}$$

Consequently, we need to have a description of the equations in terms of constant matrices E, A, B, C, D. Therefore, the new data structure

`StateSpaceObject[{`*e, a, b, c, d*`},{`*in, out, states*`},{`*sso_ options*`}]`

as well as new procedures

`GetStateSpace[`*dae, in, out*`]` and `GetDAE[`*sso,* `{`*dae_ options*`}]`

for transformations between these two data structures in both directions have been implemented. While *e, a, b, c, d* correspond to the system matrices E, A, B, C, D of (5.1), *in, out*, and *states* correspond to u, y, and x, respectively. *sso_ options* contains the same information for the underlying system of equations as *dae_ options* in the `DAEObject` data structure.

`GetStateSpace[`*dae, in, out*`]` transforms a *linear* `DAEObject` in AC or transient mode into a `StateSpaceObject`. This is achieved by extracting the system matrices from the equations encapsulated in *dae*. Thereby, the input u and the output y of (5.1) have to be specified by *in* and *out*. The resulting state space system is then stored in our new data structure `StateSpaceObject`, where the *dae_ options* hidden in *dae* are copied to *sso_ options*. Hence, all information such as design points and reference solutions are available in the new state space object.

Similarly, `GetDAE[`*sso, options*`]` transforms a `StateSpaceObject` *sso* into a `DAEObject`. Thereby, *options* are optional and can be used to add or update *sso_ options* encapsulated in *sso*. Finally, the updated *sso_ options* are added as *dae_ options* to the resulting `DAEObject`.

By using these procedures, linear time-invariant (LTI) systems of equations can be preprocessed in order to apply reduction methods such as Arnoldi's algorithm.

5.2. Arnoldi Iteration

In order to reduce a *linear time-invariant* system of DAEs given in state space formulation (5.1) *numerically*, a new procedure

`ReduceArnoldi[`*sso, q,* w_0`]`

has been implemented for the use in *Analog Insydes*. The reduction is performed by applying Arnoldi's iteration as given in Algorithm 2.48. This yields an orthonormalized matrix V which is used to project the system matrices and the states of the LTI system onto a lower dimensional subspace. Finally, the reduced system together with V is returned.

`ReduceArnoldi` is called with three parameters. The first one, *sso*, contains the LTI system encapsulated in the new data structure `StateSpaceObject`. q is the number of

iteration steps that are carried out in the Arnoldi procedure. $s_0 = i \cdot w_0$ is the expansion point which defines the *shifted moments* (cf. page 60).

The procedure firstly extracts the design point π and the system matrices E, A, B, C, D encapsulated in *sso*. Subsequently, all symbolic parameters in these matrices are replaced by the corresponding numerical values specified in π. By an implicit call of our implementation of the actual Arnoldi iteration,

$$\texttt{Arnoldi[}E,\ A,\ B,\ w_0,\ q\texttt{]},$$

the projection matrix V is computed. It is the orthonormalization of the Krylov space $\mathcal{K}_q\big((A - s_0E)^{-1}E, (A - s_0E)^{-1}B\big)$ which can be computed using the parameters committed to `Arnoldi`.

If the *original* LTI system has m inputs and the dimension of its state space is n, then B has size $n \times m$. In that case, the size of V is $n \times mq$, since the Arnoldi iteration is executed q times for each of the m column in B.

By use of the projection matrix V, the reduced state vector z and the reduced system matrices are computed as described in Section 2.5.1.5. The corresponding reduced system is then stored in another `StateSpaceObject` *sso_return*. The return value of `ReduceArnoldi` finally is given by the tuple

$$\texttt{\{}sso_return,\ V\texttt{\}}.$$

The return of V allows for comparisons of the state trajectories $x \approx Vz$ and z to each other. In this notation, x is the state vector of the original system stored in *sso* and z is the state vector of the reduced system which is encapsulated in *sso_return*. Moreover, this procedure has been successfully applied for the reduction of the transmission lines involved in the differential-amplifier circuit that is considered in Section 4.1.

5.3. Models for Transmission Lines and LTI State Space Systems

In *Analog Insydes*, analog electrical circuits are expressed in terms of `Circuit`, `Netlist`, and `Model` objects. For the representation of flat netlists, the `Netlist` object is used. This is a data structure which contains a *netlist entry* for each element in the circuit. Hence, all information necessary for the description of a circuit with flat hierarchy can be stored in a `Netlist` object.

For hierarchical circuit descriptions using subcircuits, a `Circuit` object is used. This is a data structure in which besides a circuit's netlist all its subcircuits and parameters can be stored. Finally, the data structures `Model` and `Subcircuit` serve for the storing of such subcircuits.

In this thesis, we also performed calculations that involved the modelling of transmission lines in an electrical circuit. In order to model these transmission lines as subcircuits, two `Model` objects have been implemented for the use in *Analog Insydes*. The first one applies to the transient case and is based on a semidiscretization of the *telegrapher's equations* (2.36) w.r.t. one-dimensional space. The second model relies on the state space formulation (5.1) of LTI systems using matrices E, A, B, C, and D and applies

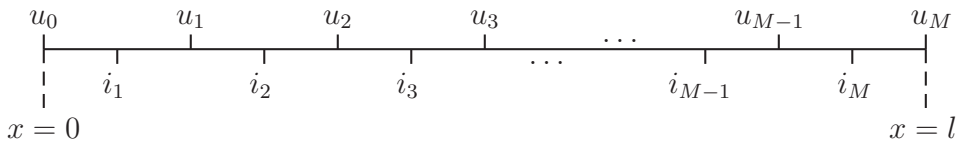


FIGURE 5.1. Spatial discretization grids for the voltages and currents in a transmission line of length l.

to AC, DC, and transient systems. The two `Model` objects are presented in more detail in the following.

Using the notation given in Section 2.2.4.2, an equidistant semidiscretization of the *telegrapher's equations* (2.36) w.r.t. one-dimensional space yields

$$dx \cdot (-C' \cdot \tfrac{\partial}{\partial t} u_k - G' \cdot u_k) = i_{k+1} - i_k, \qquad k = 1, \ldots, M-1, \tag{5.2a}$$

$$dx \cdot (-L' \cdot \tfrac{\partial}{\partial t} i_k - R' \cdot i_k) = u_k - u_{k-1}, \qquad k = 1, \ldots, M, \tag{5.2b}$$

where dx is the length l of the transmission line divided by the number of grid points M. We used two distinct discretization grids for the voltage $u = u(x,t)$ and the current $i = i(x,t)$ that are arranged as in Figure 5.1. Consequently, u_0 and u_M correspond to the voltage potentials at the transmission line's "left" and "right" terminals.

Our first `Model` object for transmission lines sets up the equations in (5.2). In a `Netlist` object, an instance using this model is referenced by

```
{transLine,
   {n1 -> "L", n2 -> "R", n3 -> "N"},
   Model -> "TransmissionLine",
   Selector -> "Transient",
   Length -> length,
   f$ref -> f_ref,
   R -> r,
   C -> c,
   G -> g,
   L -> l,
   M -> m
}.
```

Thereby, `transLine` is the *instance name*. The model has three terminals `"L"`, `"R"`, and `"N"` for connections to the remaining network at nodes *n1*, *n2*, and *n3*. While `"L"` and `"R"` refer to the "left" and "right" end of the transmission line, `"N"` usually is connected to the ground (see also Figure 2.4 on page 32). The entries `Model -> "TransmissionLine"` and `Selector -> "Transient"` uniquely identify our new model as the referenced one. Thus, `transLine` becomes an instance of this subcircuit model.

The remaining entries represent parameters for a detailed specification of `transLine`. While *length* defines its length, r, c, g, and l specify the normalized transmission line parameters R', C', G', and L' as described in Section 2.2.4.2. The *reference frequency* *f_ref* is used to compute the value of L'. It is neglected, if L' is explicitly given via l.

Finally, m is the number of grid points M and describes the refinement of the spatial discretization.

Default values for the above parameters are given by

```
Length -> 4.89,
f$ref -> 50,
R -> 0.337,
C -> 10.18*^-9,
G -> 0,
L -> 0.358/(2*Pi*f$ref),
M -> 15
```

in appropriate units, they are based on experiments with *landlines*[1]. However, for the example involving the differential amplifier in Section 4.1 we use the following parameters:

```
Length -> 5,
R -> 7.5,
C -> 4*^-10,
G -> 2.5*^-3,
L -> 1*^-5,
M -> 20.
```

Note that the scaling of the above magnitudes of course had to be adjusted. For example, while a length of 4.89 as the default value is measured in kilometers, the transmission line's length of 5 for the use in the differential amplifier refers to millimeters. Further, the frequencies are measured in Hz and the remaining parameter values are scaled in an appropriate way, thus describing realistic settings in both cases.

Our second model for transmission line components in an electrical circuit also has three terminals `"L"`, `"R"`, and `"N"` for connections to the remaining network. As in the first model, the first two of its terminals correspond to the left and right ends of the transmission line, while the third one usually is connected to the ground. A transmission line as a component in a `Netlist` object using the second model is referenced as follows:

```
{transLine,
   {n1 -> "L", n2 -> "R", n3 -> "N"},
   Model -> "TransmissionLine",
   Selector -> "StateSpace",
   Matrices -> mat
}.
```

In this model reference, `transLine` again is the instance name of the circuit component. Note that there are no parameters explicitly given as in the reference call of the first model. The entire description of the transmission line and its behavior is encapsulated and exclusively contained in the system matrices *mat* which have to be supplied in

[1]Landlines obviously do not belong to the area of micro- and nanoelectronics. Nevertheless, the modelling principles are the same in both cases.

appropriate dimensions. The vector of states is then set up correspondingly. The shape of *mat* is a list

$$\texttt{\{e -> } E\texttt{, a -> } A\texttt{, b -> } B\texttt{, c -> } C\texttt{, d -> } D\texttt{\}},$$

where E, A, B, C, and D are matrices as in (5.1). In *Analog Insydes* resp. *Mathematica*, they are given as *lists of lists*, i.e. lists of rows, where each row itself is a list of entries. They may contain numerical entries as well as symbolic ones. In the latter case, however, appropriate reference values have to be added to the system's design point.

With the above information, the equations in (5.1) are set up. The input u in this system of equations is hard-coded as the vector of voltage potentials at the terminals `"L"`, `"R"`, and `"N"`. Analogously, the output y is hard-coded as the vector of currents at these terminals directed inwards.

Note that by providing suitable matrices the role of the terminals `"L"`, `"R"`, and `"N"` can be arbitrarily interchanged. Furthermore, we generalized the matrix-based model for the general use with subcircuits that have n terminals and can be described by an LTI state space system with n inputs and n outputs. Its behavior is modelled exclusively by providing a suitable set of matrices. In the example below for $n = 3$, such an "n-gate" in a `Netlist` object is referenced by

```
{nGate,
   {n1 -> "X1", n2 -> "X2", n3 -> "X3"},
   Model -> "StateSpace3Port",
   Selector -> selector,
   Matrices -> mat
}.
```

The parameter *selector* is either `"AC"`, `"DC"`, or `"Transient"`. Depending on this choice, either

$$sEx = Ax + Bu, \qquad Ax = -Bu, \qquad \text{or} \quad E\dot{x} = Ax + Bu$$

together with $y = Cx + Du$ is set up. As for the matrix-based model for transmission lines, u and y are hard-coded as the voltage potentials and currents at the subcircuit's terminals, respectively. This model is currently available for $n = 2, \ldots, 7$, i.e. there are further `"StateSpace2Port"`, `"StateSpace4Port"`, `"StateSpace5Port"`, `"StateSpace6Port"`, and `"StateSpace7Port"` available as model names. In any of these cases, the terminals are named `"X1"`, `"X2"`, and so on, and the matrices in *mat* have to be of appropriate size.

5.4. Subcircuit Reductions

In order to automatize parts of our algorithms for hierarchical reduction, three new procedures have been prototypically implemented.

`RecordNodeVoltages` detects the nodes in the entire circuit to which the terminals of its subcircuits are connected. Then it records the voltage potentials at these nodes during a simulation of the entire circuit. Its return value is a list containing these nodes and the corresponding time-dependent functions describing their voltage potentials.

The second procedure `ReduceHelpCircuit` then operates on the separated subcircuits to which voltage sources are connected that generate the corresponding recorded potentials. Subsequently, the describing system of equations is set up and all reductions specified for the subcircuit at hand are performed sequentially[2]. Thereby, each reduction is started from the original subcircuit. Finally, the reduced subsystems are added to a list which forms the return value of this procedure.

The third procedure `ReduceSubcircuits` internally calls the two ones above. First of all, it detects the subcircuits within the entire circuit and separates them from each other. Then it calls `RecordNodeVoltages` in order to get the recorded voltage potentials of all those nodes that have a connection to a subcircuit's terminal. These are then used to set up "closed subcircuits" by connecting voltage sources to the separated subcircuits that generate the recorded potentials. For the reduction of each closed subcircuit, `ReduceHelpCircuit` is used. For each closed subcircuit, a list of reduced models is returned and appended to the set of models in the entire circuit. Thus, the referenced model for a given subcircuit instance in the entire circuit can easily be switched among its reduced models of different levels of complexity.

In the following, the three new procedures are described in more detail.

The new hierarchical reduction approach presented in Chapter 4 can be divided into several steps. One of the main steps is the separate reduction of single subcircuits. Therefore, we implemented two procedures that realize the workflow for subcircuit reductions described by Algorithm 4.1 in Section 4.2.

According to that workflow, we need to connect the terminals of the subsystem to voltage sources that generate suitable voltage potentials obtained from a previous simulation run in a test bench. Therefore, one of the first steps in the entire hierarchical reduction process is the recording of the voltage potentials of those nodes in the circuit that are connected to subcircuit terminals. This is carried out by the new procedure

`RecordNodeVoltages[`*circuit*`,` *time_interval*`, {`*sim_options*`}]`.

The first parameter *circuit* contains the entire `Circuit` object including the netlist, all subcircuits and models referenced by circuit elements, and all circuit parameters. By making use of the netlist's hierarchical structure, it sets up the list N_{sub} of nodes that are connected to at least one terminal of a subcircuit. Then the entire circuit is simulated on the time interval *time_interval*. It has the shape of `{`*t_start*`,` *t_end*`}` , where *t_start* is the starting time and *t_end* is the ending time of the simulation. The optional list `{`*sim_options*`}` is used to specify certain simulation options.

Note that in this implementation we simulate the entire circuit. This means that for each subcircuit the corresponding remaining circuit acts as a test bench.

During the simulation run, the voltage potentials of the nodes in N_{sub} are recorded which yields a list of time-dependent interpolating functions. For each of the above

[2]Currently, there are only *symbolic* techniques for general nonlinear systems available. First, term reductions in level 0 are performed as described in Section 2.5.2.2. In a second step, the resulting system is compressed by applying algebraic manipulations. These two reduction techniques are hard-coded in this order and usually are sufficient for a significant reduction of a system's level of complexity.

nodes n, a rule n `->` f_n is set up, where f_n is the corresponding interpolating function that describes the node's voltage potential on the interval *time_interval*. The return value of `RecordNodeVoltages` finally is a list of all these rules.

The next step then is the reduction of the separated subcircuits whose terminals are connected to voltage sources that generate suitable potentials. For this purpose, the list returned by `RecordNodeVoltages` is used. The rules contained in this list are used to connect voltage sources to the subcircuits' terminals that generate the voltage potentials recorded during the simulation of the entire system. A subcircuit prepared in this way can then be reduced by applying the second procedure

```
ReduceHelpCircuit[
        help_circ,
        time_interval,
        {perm_errors},
        {add_contr_vars},
        {sim_options}
].
```

help_circ is a `Circuit` object that contains the subcircuit together with the voltage sources connected to its terminals. It is simulated on the time interval *time_interval* and with optional simulation specifications given by {*sim_options*}.

Note that currently this procedure is designed only for the use with *symbolic* model reduction techniques described in Section 2.5.2.2. Therefore, using the notation of Section 2.5.2, {*perm_errors*} prescribes a list of error bounds ε used for symbolic reductions of *help_circ*, i.e. it defines the accuracy of the reduced subsystems that one wants to achieve. While the voltage potentials generated by the sources connected to the subcircuit's terminals act as inputs, the resulting currents at the terminals define the output of the system. They are controlled by comparisons to a reference solution of the original subcircuit. Additional variables that have to be controlled during the reduction process can be specified by {*add_contr_vars*}.

First, MNA is used to set up the describing equations for the original subcircuit encapsulated in *help_circ*. For each error bound ε in {*perm_errors*}, this system of equations is symbolically reduced. This finally yields a list of reduced models with different accuracies. For the symbolic reductions, there are currently two methods from Section 2.5.2.2 hard-coded, namely *term reductions* and *algebraic manipulations*. At first, term reductions on level 0 are applied to the original set of equations for each error bound in {*perm_errors*}. This has the advantage that the costly *transient term ranking* has to be computed only once in the very beginning for the original subsystem. For term reductions of the original subsystem using the different error bounds, it is then still available in the cache memory. After that, the resulting reduced subsystems are further treated by algebraic manipulations. Finally, all the reduced subsystems obtained in this way are arranged in a list which then is returned by `ReduceHelpCircuit` and can be used for replacing the original subsystem in the entire circuit by models of different accuracies.

In order to allow an easy switching among the reduced models of the subcircuits, they are added to the entire circuit as `Subcircuit` objects. Furthermore, the above two

procedures have been integrated in a third one, thus automatizing the steps in between. This third procedure is called via

```
ReduceSubcircuits[
        circuit,
        subcirc_specs,
        time_interval,
        {sim_options}
].
```

Thereby, the parameter *circuit* contains the entire hierarchical `Circuit` object. This means that the entries of the `Netlist` object contained in *circuit* have a shape of

(5.3) `{` *sub_inst*, `{`*conn*`}`, `Subcircuit ->` *name*, `Selector ->` *sel_name* `}`,

where *sub_inst* is a subcircuit instance name, *conn* defines the subcircuit's connection to the nodes of the circuit, and where `Subcircuit ->` *name* and `Selector ->` *sel_name* uniquely reference the subcircuit model of which *sub_inst* is an instance. Of course, there might also be some single circuit components besides these entries which are not part of a subcircuit.

Further, the parameters *time_interval* and `{`*sim_options*`}` are shaped as before. While *time_interval* specifies the time interval on which the entire reduction process is performed, `{`*sim_options*`}` is optional.

The parameter *subcirc_specs* contains all the information that is necessary to compute the reduced models of the subcircuits of the entire circuit. It is a list of rules that assign to each subcircuit instance occuring in *circuit* a list of error bounds and a list of additional variables that have to be controlled. Its shape is as follows:

```
{
   sub_instance1 -> {{perm_errors1}, {add_contr_vars1}},
   sub_instance2 -> {{perm_errors2}, {add_contr_vars2}},
   sub_instance3 -> {{perm_errors3}, {add_contr_vars3}},
      ...
}.
```

In this parameter definition, *perm_errors*$\boldsymbol{i}$ and *add_contr_vars*$\boldsymbol{i}$ are specified in the same way as the parameters *perm_errors* and *add_contr_vars* in `ReduceHelpCircuit`. Furthermore, the parameters *add_contr_vars*$\boldsymbol{i}$ are only optional.

In the first step, `ReduceSubcircuits` calls `RecordNodeVoltages` in order to obtain the recorded voltage potentials at those nodes of the circuit to which the subcircuits are connected. Then it separates the subcircuit instances from each other and connects their terminals to voltage sources that generate the corresponding recorded voltages. This is done by making use of the return value of `RecordNodeVoltages`. In the next step, the so far prepared subcircuit instances are reduced in accordance with the specification given in *subcirc_specs* by using `ReduceHelpCircuit`. The return value of this procedure is a list containing all the resulting reduced models for the committed subcircuit. This list is then added to the subcircuit models contained in *circuit*.

Since a subcircuit instance references a model solely by the specification of *name* and *sel_ name* in 5.3, our proceeding allows for an easy switching of the occuring subcircuit instances among the corresponding original and reduced models of different levels of complexity. Thus, we can quickly create hierarchically reduced entire models of different accuracies.

5.5. Development Environments and Error Functions

In order to show the aptitude of the algorithms and implementations developed in this thesis for circuit design problems of current industrial size, we applied them successfully and with significant savings in computation time to a differential amplifier as well as an operational amplifier typically used in industry (cf. Chapter 4). For this, we implemented extensive development environments. They include tests and checks of our algorithms as well as motivating examples for the hierarchical reduction approach.

Since the choice of an error function used for the reduction of a system is of great importance, we further implemented most of the error functions presented in Section 4.6. They have been used for experiments and computations within the above environments.

Furthermore, our manual computation of subsystem sensitivities and their ranking in the case of the operational amplifier has been performed using the corresponding development environment. Finally, also the manual check of Algorithm 4.5 from Section 4.4 for an optimized order of subsystem reductions using different error functions for measuring the influence of the subcircuits on the performance of the entire circuit has been made within these environments.

CHAPTER 6

Summary and Outlook

In order to envision what has been done in this thesis, the content is summarized below. Furthermore, an outlook to the treated area of research is provided in the second section.

6.1. Summary

In order to cope with the ever increasing size of systems of equations describing the behavior of electrical circuits, a new model reduction approach has been presented in this thesis which exploits the hierarchical structure available on circuit level. Besides a faster processing of smaller subproblems, this further allows for coupling different symbolic and numerical model reduction techniques. While there exist numerical methods that can handle systems of very large size, symbolic techniques allow insights into functional relations and dependences of the circuit's behavior on the dominant parameters of the system.

Chapter 1 provided a general introduction to the topics treated in this thesis. Different methods for system analysis have been presented and a short survey of their historical development was given. By considering the quickly growing complexity of the systems, the need for reduction methods has been motivated. Then, model order reduction has been introduced together with a review of its main application areas and some historical background, particularly in the symbolic case. By considering two practical examples from electrical engineering and weather prediction, both symbolic and numerical analysis for dynamical systems have been motivated. Finally, the aims of this thesis have been defined in the last section of this chapter.

In order to make this thesis self-contained, foundations needed throughout this thesis have been illuminated in more detail in Chapter 2. The section about network analysis showed that the behavior of an electrical circuit is mathematically described by a system of equations composed of the Kirchhoff laws and the current-voltage relations of the circuit components. It further has been described how to set up these equations in an automatized way. Standard techniques such as the *sparse tableau analysis* or the most widely used *modified nodal analysis* have been explained in detail. The thereby resulting systems of equations have been dealt with in the next section. Besides systems of DAEs, also systems of PDAEs have been considered. As practical examples of such systems, we illuminated the *telegrapher's equations* and the *drift-diffusion equations* in detail. They are used in electrical circuits for the modelling of transmission line effects and semiconductor devices, respectively. Then, after a description of the most important numerical analyses to solve systems of DAEs, basic terms and notions from systems and control theory needed throughout the following sections have been presented.

This included basic concepts of stability, passivity, reachability, and observability of dynamical systems.

We then approached the main topic of this thesis, namely, model order reduction. Since our new hierarchical reduction approach offers a possibility for the coupling of different reduction techniques, we reviewed the most popular numerical methods for both linear and nonlinear systems in their basic versions to a large extent. We considered Krylov methods such as the *Arnoldi iteration* and the SVD-based technique of *balanced truncation*. Furthermore, the *proper orthogonal decomposition* and the *trajectory piecewise-linear approach* have been illuminated in detail. But also *symbolic techniques* for both linear and nonlinear systems have been surveyed and – since this area of research is rather new and less well-known – been explained in detail.

Chapter 3 then has focussed on the structure of electrical circuits. First, the general modelling of component-based systems such as electrical circuits as a network of subsystems coupled by a connecting structure has been explained. In this context, the subsystems in the network correspond to the components of the circuit. Then, the concept of coupled and interconnected systems has been described. Since in general they are composed of several *building blocks* such as current mirrors, amplifying stages, or semiconductor devices that have to be modelled via PDEs, electrical circuits can be considered as sets of systems that interact with each other. Hence, electrical circuits in general are interconnected systems. As another example of such systems, an acceleration sensor has been considered and its functionality has been explained briefly. This *micro electro-mechanical system* combines components of an electrical circuit with mechanical parts and, therefore, is an interconnected system.

Next, a concept for structure-preserving model order reduction of interconnected linear time-invariant systems has been reviewed. Structure preservation in this context means the preservation of the block structure of the involved system matrices. However, the considered method is not suitable for general nonlinear systems. Moreover, it is based on the systems-theoretical framework of inputs and outputs to model the subsystems' interaction. In contrast to this, the behavioral approach relies on the *sharing of variables* among the respective subsystems rather than declaring some of them as inputs and others as outputs. This approach has been illuminated in the next section by using some examples from different areas of physics. We then compared the two different concepts to each other by using the example of a differential amplifier.

Finally, the last section in this chapter dealt with the macromodel concept. By a segmentation of the entire circuit and by connecting appropriate voltage or current sources to the "open wires" of the resulting subcircuits, one can take advantage of a faster processing of smaller subproblems. This technique has been adapted for the separate reduction of subcircuits of large electrical circuits in the next chapter.

In Chapter 4, we have reached the core of this thesis, namely, model reduction of electrical circuits that exploits the hierarchical structure available on the circuit level. Besides a faster processing of smaller subsystems, this additionally offers possibilities for the coupling of different symbolic and numerical reduction methods. First, a motivating example has been provided with a view to the great benefits that can be achieved by taking

the hierarchical structure of a circuit into account. By a coupling of *nonlinear symbolic methods* and the *Arnoldi iteration* for suitable subcircuits, a differential amplifier could be reduced within seconds instead of several hours. The resulting hierarchically reduced model only had about half the number of equations of the non-hierarchically reduced one. It further proved to be very accurate and robust w.r.t. different inputs, even for pulse-shaped excitations. Moreover, simulations of the hierarchically reduced model have been accelerated approximately by a factor of 5.

In the next section, we adapted the macromodel concept reviewed in the previous chapter and presented a new workflow for separate reductions of single subsystems. It uses simulations in a test bench to generate suitable voltage potentials at the terminals of the separated subsystems. Then the subsystems prepared in this way can be reduced by applying the usual non-hierarchical reduction techniques, thus yielding reduced models of the subsystems. In order to measure the influence of a single subcircuit on the entire circuit's behavior, we presented a new algorithm. It sequentially replaces a subcircuit in the original connecting structure by its corresponding reduced models of different accuracies. During this process, the resulting error on the entire system's output has been monitored. This led to a new concept of *subsystem sensitivities*. Subsequently, these sensitivities have been exploited in order to derive a *ranking of subsystem reductions*. A new algorithm has been introduced that uses this ranking for deriving a hierarchically reduced model of the entire circuit by performing suitable subsystem reductions in an optimized order. At this point, we want to stress again that the occuring subsystems of the circuit can be reduced by different symbolic and numerical techniques. Hence, the algorithm finally yields a reduced model of the circuit which is obtained by the coupling of different reduction methods. Since error functions for measuring deviations from the reference solutions of a system play a crucial role during the reduction process, a variety of such functions has been provided and newly created.

To conclude this chapter and to show its aptitude for industrial applications, the new hierarchical reduction approach has been applied to an operational-amplifier circuit typically used in industry. Furthermore, the results have been compared to the usual non-hierarchical reduction approach. By merging certain circuit components to suitable building blocks, we reconstructed the subcircuit structure of the amplifier. We then performed symbolic reductions to any of the resulting seven subcircuits by supplying a sweep of thirteen different error bounds. Furthermore, two different error functions from the previous section have come to operation during the reduction process. Proceeding in accordance with the algorithms described above, we obtained hierarchically reduced entire models of the amplifier of high accuracy. The time costs for this were only a fraction of the time needed for the usual non-hierarchical reduction approach.

In order to further improve our approach, we additionally took the number of equations and terms of the reduced subsystem models into account during the hierarchical reduction process. This led to systems of even smaller size and increased simulation performance, while the level of accuracy could be maintained. In a second step of improvement, we then applied further non-hierarchical symbolic reduction methods to the so far obtained *interim* models, thus yielding systems of very low complexity. For one of the "best"

resulting systems, the number of equations could be reduced to 34 (16%) instead of 215, the number of terms decreased from 1050 to 92 (9%), and the simulation performance was up to 19 times faster than the original system. Moreover, the corresponding systems also proved to be very robust w.r.t. different input excitations. Although they have been reduced by using a smooth sine wave excitation to compute the reference solution, they even delivered good results for a highly non-smooth pulse-shaped excitation.

Summing up the above, this example together with the motivating example in the beginning of this chapter have shown the enormous capability of hierarchical model reduction using the coupling of different symbolic and numerical techniques. Furthermore, the new approach is not limited to academic examples, but also adequate for the use in industrial environment.

Finally, Chapter 5 has given an overview of the prototypical implementations made within this thesis. Besides new data structures and procedures for their manipulation, we also implemented some data objects for the modelling of general linear time-invariant state space systems and transmission line components in electrical circuits. Large parts of the newly introduced hierarchical reduction approach have been implemented successfully in prototypical versions. Together with implementations of suitable error functions from Chapter 4, they have come to operation for the hierarchical reduction of the operational amplifier.

6.2. Outlook

Although we only worked on "level 0" of the electrical circuit at hand, the algorithms from Chapter 4 can be adapted recursively to its substructures. This means that suitable subcircuits may also be segmented in accordance with an appropriate interconnecting structure. Thus, one can take advantage of an increased speed of the processing of even smaller subproblems. On the other hand, a segmentation of the entire circuit into too many subsystems increases the cost of administration of the interconnecting network. Consequently, the detection or reconstruction of suitable subsystems in the entire circuit is not trivial. First approaches to an automatized detection of subcircuit structures have been made [**CAD**], they rely on the concept of *pattern matching*. The integration of such algorithms in the hierarchical reduction approach can further automatize the entire process of structure-exploiting coupled symbolic-numerical model reduction of electrical circuits. Consequently, one will not be restricted to circuits that are given by a *hierarchical* netlist description.

Furthermore, in order to guarantee good numerical solvability (stability) of systems obtained by the hierarchical reduction approach, an index monitor may be added to Algorithm 4.5. For example, it could be included somewhere between lines 11 and 13 in this algorithm after the so far reduced entire system is updated by a new replacement of an appropriately reduced subsystem. A simple computation of the index of the system that corresponds to the current subsystem configuration and comparisons to the "old" one then helps to improve the numerical stability of the resulting hierarchically reduced entire system. If the index is raised by the new configuration of subsystems, the current subsystem replacement is rejected and the configuration is reset.

Finally, a relation between the *global error* on the output of the entire system and the errors of the subsystems in the interconnecting structure could be exploited to obtain estimates of the errors of the separate subsystems. Thus, the user-specified error bound of the global error directly yields suitable error bounds for the reductions of the subsystems in the interconnecting structure.

APPENDIX A

Network Theory

Theorem A.1. *A fundamental loop system of a graph $G = (V, E)$ is linearly independent and spans the space of all its loops.*

Proof. First of all, we have to show that the set of loops in a graph has the structure of a linear vector space. For this, assume that the branches of G are numbered from 1 to $b := |E|$, hence $E = \{e_1, \ldots, e_b\}$. Identifying a loop in a graph by its participating branches, one can write it – and hence further identify it – as a b-dimensional vector $v \in \mathbb{Z}_2^b$, where the i-th entry v_i is 1 if branch e_i is contained in the loop and 0 otherwise. Obviously, $\mathbb{Z}_2^b$ with the scalar multiplication over $\mathbb{Z}_2$ and the "normal" componentwise addition has the structure of a linear vector space. The corresponding addition for the set of loops in a graph is given by the symmetric difference

$$l_1 \Delta l_2 := (l_1 \cup l_2) \setminus (l_1 \cap l_2).$$

Furthermore, by the identification above, it is intuitively clear that the symmetric difference of two loops again is a loop, the branch-disjoint union of two loops, or the *zero loop*. Hence, the set of loops of a graph together with the zero loop and the union of branch-disjoint loops is a linear subspace of $\mathbb{Z}_2^b$. (For a detailed proof see [**SwaThu**]).

Now choose a tree T of G. This defines the set of tree and link branches in G. Then a fundamental loop in G is uniquely identified by its only link branch f_k among the participating branches. The link branch f_k does not occur in any of the other fundamental loops, i.e. with the identification from above, the k-th entries in the vectors corresponding to the remaining fundamental loops are 0. Hence, the set of fundamental loops is linearly independent, since no fundamental loop can be written as a linear combination of the remaining ones.

It remains to show that any loop in G can be written as a linear combination of the fundamental loops. So let l be a loop in G and let $f_1, \ldots, f_m$ be the link branches contained in l besides appropriate tree branches. Let $l_1, \ldots, l_m$ be the fundamental loops that are defined by $f_1, \ldots, f_m$ and let $\widetilde{l} = l_1 \Delta \ldots \Delta l_m$. We want to show that $l = \widetilde{l}$.

Since all link branches of l are contained in $\widetilde{l}$ as well, $l \Delta \widetilde{l}$ can only consist of tree branches or be the zero loop. Since we showed that the loops of a graph form a vector space and a loop cannot be built only by tree branches, $l \Delta \widetilde{l}$ must be zero, which is equivalent to $l = \widetilde{l}$. □

Theorem A.2. *Let $G = (V, E)$ be a connected and directed graph with $n := |V|$ nodes $v \in V$ and $b := |E| \geq n - 1$ branches $e \in E$. For the (augmented) nodal incidence matrix A_a, one has* $\operatorname{rank}(A_a) = n - 1$.

PROOF. Recall that the rows of A_a correspond to the nodes and the columns correspond to the branches of G. Hence, A_a is an $n \times b$-matrix with exactly one entry equal to 1 and one entry equal to -1 in each column. Hence, the sum of all its rows is zero. This proves that the rows are linearly dependent and, therefore, $\operatorname{rank}(A_a) < n$.

Now consider a spanning tree T of G with $n-1$ branches $e_1, \ldots, e_{n-1}$ connecting the n nodes of G. For each branch $e = (u, v)$ in G there exists a path from u to v in T using $\leq n-1$ branches. Thus, e is a linear combination of the branches $e_1, \ldots, e_{n-1}$.

If one leaves out one of the branches in T, say $e_k = (u, v)$, due to its connectedness there exists a branch e in G – e.g. e_k itself – which is no linear combination of the branches $e_1, \ldots, e_{k-1}, e_{k+1}, \ldots, e_{n-1}$. Otherwise, the former T would have had loops, which is a contradiction. Thus, $e_1, \ldots, e_{n-1}$ or the corresponding columns in A_a form a minimal generating system. It follows that $\operatorname{rank}(A_a) = n-1$. □

THEOREM A.3. *The reduced nodal incidence matrix A and the transpose B^T of the reduced loop incidence matrix form an exact pair of matrices, i.e.*

$$A \cdot B^T = 0 \qquad \textit{and} \qquad \operatorname{rank}(A) + \operatorname{rank}(B^T) = b.$$

SKETCH OF PROOF. The second condition for exactness of A and B^T is trivial, since $\operatorname{rank} A = n-1$ and $\operatorname{rank} B^T = \operatorname{rank} B = b - n + 1$, hence $\operatorname{rank} A + \operatorname{rank} B^T = b$.

In order to show that $A \cdot B^T = 0$, consider the i-th row of A corresponding to node v_i and the j-th column of B^T, which corresponds to the j-th fundamental loop l_j. l_j either contains no branch that has an origin or terminus v_i, or exactly two such branches. In the first case, the scalar product of the corresponding row and column in A and B^T is zero. In the latter case, one has to distinguish four arrangements of l_j and the two involved branches, since each one of them either can be directed in the same or the opposite direction as l_j.

One finally verifies that in any of these cases the scalar product of the corresponding row and column is zero, hence, $A \cdot B^T = 0$. □

APPENDIX B

Semiconductor Device Modelling – The Drift-Diffusion Model for the Diode

In the literature, there is a huge amount of books and papers concerning the modelling of semiconductor devices. This section mainly follows [**GerKneVog, Gue01, MarRinSch, Sel, Tis**] to give an extensive review of the drift-diffusion equations for semiconductor devices, a parameterized mathematical model for the electron transport in a semiconductor. In the references above, also suitable numerical parameter values for the drift-diffusion equations can be found.

In the case of semi-classical transport completed with balance equations, one obtains the Boltzmann equation and finally, by further simplifications, the drift-diffusion equations. As the name of this model already indicates, the currents in semiconductor devices are mainly steered by *drift* and *diffusion* of charge carriers. While the *drift current* is due to an electric field E formed by free charge carriers, the *diffusion current* is caused by their movement trying to compensate their inhomogeneous concentrations in the interior of the semiconductor crystal.

We focus on the simplest case of a diode. However, also transistors such as the bipolar junction transistor (BJT) in Figure B.1 can be modelled using the drift-diffusion equations. The equations are the same in any case, the type of semiconductor device at hand only defines the preconcentration of so-called *impurity atoms*, i.e. the *doping profile* of the semiconductor, and the position of its contacts.

B.1. Physics of a Diode

This section explains the physics of a diode or, more generally, a semiconductor device. We follow the descriptions given in [**GerKneVog, MarRinSch**].

FIGURE B.1. A pn-diode (left) and a pnp-transistor (right) with their 2-dimensional geometry including the doping regions Ω^+, Ω^- and the (metal) contacts (black).

FIGURE B.2. The valence band and the conduction band in a semiconductor are characterized by different ranges of electron energy.

The atoms of a certain semiconductor material have electrons with different levels of energy contained in their shells. Most of the electrons are *valence electrons*, they are responsible for the chemical compound of the material. *Conduction electrons* are electrons with a higher level of energy than the valence electrons. They can move freely following the influence of an electric field, thus generating an electric current. The different energy levels are depicted in Figure B.2. While conduction electrons have a higher energy potential within the range of the *conduction band*, the energy potential of valence electrons is situated in the lower energy range of the *valence band*. The gap between those two ranges of electron energy is significantly large for semiconductors and usually referred to as the *bandgap*.

The semiconductor crystal, say silicon, is located in the interior of the diode. By diffusion of so-called *impurity atoms* into the silicon crystal and by their implantation with an ion beam, *donor* and *acceptor atoms* are brought into the interior of the semiconductor. Their concentrations are denoted by N_D^+ and N_A^-, respectively, and the *doping regions* where the corresponding impurity atoms are located by Ω^+ ("p-regions") and Ω^- ("n-regions", cf. Figure B.1). They are assumed to be fixed in the semiconductor, which is justified if the impurity concentrations are sufficiently small. Hence, N_D^+ and N_A^- are independent of time and given by functions depending only on the position variable $x \in \mathbb{R}^d$ in the interior of the silicon crystal, where $d = 1, 2, 3$ depends on the modelling dimension. $N(x) := N_D^+(x) - N_A^-(x)$ is the *impurity* or *doping profile*. The n-regions Ω^- of the crystal, where the preconcentration N_A^- of acceptor atoms predominates, are characterized by $N(x) < 0$. Similarly, the p-regions Ω^+ are predominated by donor atoms such that $N(x) > 0$ holds. The boundaries between the p- and n-regions, where N changes its sign, are called *p-n junctions*.

The number of free charge carriers is responsible for the semiconductor's conductivity. While donor atoms try to get rid of the electrons in their outermost atomic shell, acceptor atoms attract and try to fill their outermost shell with electrons. This causes different concentrations of free negative and positive charge carriers, i.e., electrons and *holes*; when the silicon crystal is electrically neutral, then for each *conduction electron* there is a corresponding *hole* in the valence band to which the positive charge $+q := +1.60218 \cdot 10^{-19}\mathrm{A\,s}$ can be assigned. An atom in the semiconductor that lacks one of its valence electrons may attract an electron in the conduction band from another atom. Thus, the movement of this electron can be interpreted as the movement of a hole

in the opposite direction, from one atom to another one. Hence, holes are considered as positive charge carriers, while electrons are negative ones. Since the bandgap usually is significantly large for semiconductors, quite a lot of energy is necessary to transfer electrons from the valence band to the conduction band. This process is called *generation of electron-hole pairs*, i.e. an electron is generated in the conduction band and a hole in the valence band. The inverse process, i.e. the transfer of a conduction electron into the lower energetic valence band, is termed *recombination of electron-hole pairs*.

As a consequence of the above, there is a large incline of the concentrations of electrons and holes in opposing directions at the p-n junction of two oppositionally doped parts of the semiconductor. Conduction electrons in Ω^+ are attracted by and, hence, diffuse to Ω^- in order to equalize the different charge carrier concentrations, while holes in Ω^- diffuse to Ω^+. This process only lasts until the electrostatic field caused by these free charge carriers is strong enough to prevent further diffusion of charge carriers. Then, there is an equilibrium between the *drift current* which is due to the electric field and the *diffusion current*. In this situation, a *depletion region*, i.e. a region with almost no free charge carriers, is formed around the p-n junction and almost no current except for the very small *leakage current* flows.

If a potential difference of an appropriate sign is applied, i.e. a difference in the voltage potentials of the diode's two contacts, then the depletion region around the p-n junction, i.e. the semiconductor's resistance is increased (reverse bias case). Effectively, the depletion region works as an insulator and prevents currents from flowing. If the applied potentials at the contacts are vice versa (forward bias case), the depletion region shrinks. So the semiconductor's resistance decreases, the free charge carriers tend to neutralize the semiconductor's doping concentration, and a current – depending on the applied voltage – flows.

To summarize the above, a diode works similar to a valve only "one-way", i.e. for certain applied voltages at its contacts it behaves like a conductor and lets a current flow, while for applied voltages in *reverse direction* it behaves like an insulator and prevents a current flow.

B.2. Current Density and Continuity Equations

The electric field $E\ \left[\frac{\mathrm{V}}{\mathrm{m}}\right]$ generated by the free charge carriers in the semiconductor crystal is responsible for the *drift current*, the corresponding current densities are given by

$$q\mu_n nE \quad \text{and} \quad q\mu_p pE \quad \left[\frac{\mathrm{A}}{\mathrm{m}^2}\right] \tag{B.1}$$

for electrons and holes, respectively. n and p $\left[\frac{1}{\mathrm{m}^3}\right]$ are their concentrations and q is the elementary charge with a value of approximately $1e := 1.60218 \cdot 10^{-19}\mathrm{A\,s}$. The electron and hole mobilities μ_n and μ_p $\left[\frac{\mathrm{m}^2}{\mathrm{V\,s}}\right]$ are bounded strictly positive functions depending on semiconductor material, doping, temperature, and the electric field E. The choice of an appropriate model for the mobilities depends on the effects one wants to account for.

The densities of the *diffusion currents* caused by the movement of charge carriers trying to equalize their differing concentrations in the semiconductor is proportional to the

gradient of the charge carrier concentration. This yields

$$qD_n \operatorname{grad}(n) \quad \text{and} \quad -qD_p \operatorname{grad}(p) \quad \left[\tfrac{\mathrm{A}}{\mathrm{m}^2}\right] \tag{B.2}$$

for electrons and holes, respectively, where D_n and D_p $\left[\frac{\mathrm{m}^2}{\mathrm{s}}\right]$, the charge carrier diffusivities, in general are bounded strictly positive functions depending on semiconductor material, doping, and temperature. For non-degenerate semiconductors in thermal equilibrium[1], the mobilities μ_n, μ_p and the diffusivities D_n, D_p are related to each other by the *Einstein relations*

$$D_n = \frac{kT}{q}\mu_n \quad \text{and} \quad D_p = \frac{kT}{q}\mu_p, \tag{B.3}$$

where T $[K]$ is the temperature and $k = 8.617134 \cdot 10^{-5} eV/K$ is the Boltzmann constant. ($V_T := \frac{kT}{q}$ is the *thermal voltage*.)

Since the electrostatic potential V $[V]$ and the electric field E are related by

$$E = -\operatorname{grad}(V), \tag{B.4}$$

summing up all the contributions above, one obtains for the current densities J_n and J_p of electrons and holes

$$J_n = -q\mu_n n \operatorname{grad}(V) + qD_n \operatorname{grad}(n), \tag{B.5a}$$

$$J_p = -q\mu_p p \operatorname{grad}(V) - qD_p \operatorname{grad}(p). \tag{B.5b}$$

They are measured in $\left[\frac{\mathrm{A}}{\mathrm{m}^2}\right]$ and depend on the time t and the position $x \in \mathbb{R}^d$, $d = 1, 2, 3$. In the case of the application of a magnetic field to the semiconductor, an additional current has to be taken into account, which, however, usually can be neglected for devices used in integrated circuits.

The relations between the electron and hole concentrations and the corresponding current densities are given by the *continuity equations* describing particle conservation and have to be added. Therefore, one has

$$-q\partial_t n + \operatorname{div} J_n = qR, \tag{B.6a}$$

$$q\partial_t p + \operatorname{div} J_p = -qR, \tag{B.6b}$$

where $R = R(x,t)$ $\left[\frac{1}{\mathrm{m}^3\,\mathrm{s}}\right]$ is the rate of generation and recombination of charge carriers; if an electron is moved from the *valence band* to the higher energetic *conduction band* (cf. Figure B.2) under energetic costs, a free electron and a hole are generated simultaneously (electron/hole emission). If the opposite happens, i.e., an electron from the conduction band falls back and recombines with a hole in the valence band, the two charges are neutralized. The electron's energy is either transferred to another electron in the conduction band (electron capture) or to another hole in the valence band (hole capture).

[1] If one brings together two different materials, one usually has differing electrostatic potentials. Hence, electrons and holes of both parts start to diffuse in the respective other part. As soon as this diffusion process stops, the state obtained is called *thermal equilibrium*.

The choice of the model for the generation-recombination rate R also depends on the effects that one wants to take into account. They are not further specified here, see, e.g., [**MarRinSch**] for an overview.

B.3. The Poisson Equation

Due to the free charge carriers, an electric field E exists in the interior of the semiconductor. To obtain a self-consistent formulation, equations (B.5) and (B.6) are completed by the third Maxwell equation relating E to the electric charges. Since the local charge in a semiconductor consists of electrons, holes, and donor and acceptor atoms, by using equation (B.4), one has the *Poisson equation*

$$\operatorname{div}(-\varepsilon \operatorname{grad} V) = q(p - n + N), \tag{B.7}$$

where $N(x) = N_D^+(x) - N_A^-(x)$ denotes the doping concentration depending only on the position variable x. ε is the so-called *permittivity constant*, its value in silicon is approximately $10^{-10}\frac{\mathrm{A\,s}}{\mathrm{V\,m}}$.

The set of equations (B.5)–(B.7) form the ***drift-diffusion model equations***, originally due to van Roosbroeck [**vRoo**], where the unknowns are the electrostatic potential $V = V(x,t)$, the charge carrier concentrations $n = n(x,t)$ and $p = p(x,t)$, and the current densities $J_n = J_n(x,t)$ and $J_p = J_p(x,t)$. Since J_n and J_p are given by (B.5), inserting these in the remaining equations yields a system in the primary variables V, n, and p only.

Note that the drift-diffusion model equations are a set of five coupled PDEs, where the Poisson equation (B.7) is of *elliptic type*, while the two continuity equations in (B.6) are of *parabolic type* [**Jos**].

B.4. Initial and Boundary Conditions

Many semiconductor devices have metal contacts with low resistance which are called *Ohmic contacts*. We will restrict ourselves here to this kind of semiconductors.

Let $\Omega = \Omega^- \cup \Omega^+ \subseteq \mathbb{R}^d$, $d = 1, 2, 3$, with doping regions Ω^- and Ω^+ denote the d-dimensional geometry of a pn-diode. Figure B.3 shows the 2-dimensional case. In general, the pn-diode has two types of bounding materials: its contacts to the surrounding network, denoted by Γ_1 and Γ_2 in Figure B.3, and insulating materials like oxides which

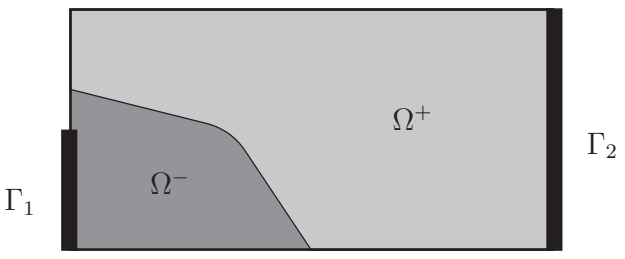


FIGURE B.3. The pn-diode as a 2-dimensional model including the doping regions Ω^+ and Ω^- and two metal contacts Γ_1 and Γ_2.

we simply denote by Γ_N. Hence, one has a disjoint union of the bounding materials $\partial\Omega = \Gamma_N \cup \Gamma_1 \cup \Gamma_2$.

At the Ohmic contacts, the space charge vanishes, therefore

$$n(x,t) - p(x,t) - N(x) = 0 \quad \text{for } x \in \Gamma_1 \cup \Gamma_2 \text{ and for all } t. \tag{B.8}$$

We consider the Ohmic contacts to be *ideal*, i.e. they have a very high doping of charge carriers. Thus, with their resistance tending to zero, one has

$$n \cdot p = n_i^2 \quad \text{for } x \in \Gamma_1 \cup \Gamma_2 \text{ and for all } t \tag{B.9}$$

with the *intrinsic concentration* $n_i = n_i(x)\ \left[\frac{1}{\mathrm{m}^3}\right]$. Since n_i depends on material and temperature[2], we consider the system to be in thermal equilibrium. This means that the applied voltages and, thus, also the currents are considered not to be too large. Hence, using the last two equations, one easily obtains the *Dirichlet boundary conditions* for n and p

$$n(x,t) \;=\; n(x) = \frac{1}{2}\left(\sqrt{N^2(x) + 4n_i^2(x)} + N(x)\right), \tag{B.10a}$$

$$p(x,t) \;=\; p(x) = \frac{1}{2}\left(\sqrt{N^2(x) + 4n_i^2(x)} - N(x)\right) \tag{B.10b}$$

at the Ohmic contacts, i.e. $x \in \Gamma_1 \cup \Gamma_2$, and for all t. The electrostatic potential V at the Ohmic contacts is given by

$$V(x,t) = V_{\mathrm{ap}}(t) + V_{\mathrm{bi}}(x) \quad \text{for } x \in \Gamma_1 \cup \Gamma_2, \tag{B.11}$$

where $V_{\mathrm{bi}}(x) = V_T \cdot \ln\left(\frac{n(x)}{n_i(x)}\right) = -V_T \cdot \ln\left(\frac{p(x)}{n_i(x)}\right)$, $x \in \Gamma_1 \cup \Gamma_2$, is the so-called *built-in voltage* and V_{ap} is the applied voltage potential. Hence, the boundary values of n, p, and V for the diode's Ohmic contacts are given by (B.10) and (B.11).

We also consider the insulating *Neumann bounding parts*[3] Γ_N of the diode to be ideal, such that there is no current flow and a zero electric field in the normal direction of Γ_N (Figure B.4). This is accounted for by the equations

$$J_n(x,t) \cdot \nu = 0, \tag{B.12a}$$

$$J_p(x,t) \cdot \nu = 0, \tag{B.12b}$$

$$\operatorname{grad}(V) \cdot \nu = 0, \tag{B.12c}$$

where $x \in \Gamma_N$ and ν is the unit outward normal vector on the Neumann bounding parts Γ_N. The equations are obtained as follows: The current I leaving the diode at Γ_N is given by

$$I = \int_{\Gamma_N} (J_n + J_p - \varepsilon\, \partial_t \operatorname{grad} V) \cdot \nu \, d\gamma,$$

[2]The value of n_i in silicon at room temperature is approximately of the order of magnitude $10^{11}\,\mathrm{cm}^{-3}$.

[3]They are called *Neumann* bounding parts, since they involve partial derivatives of the electrostatic potential V. The boundary conditions at the metal contacts are of *Dirichlet type* because the charge carrier concentrations n and p as well as the electrostatic potential V (via V_{ap} and V_{bi}) are prescribed there.

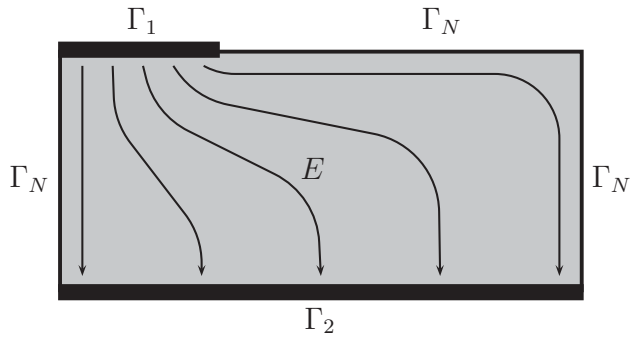


FIGURE B.4. The electrostatic field E between the two contacts of a diode. With $E = -\operatorname{grad}(V)$ and ν normal on Γ_N, one has $\operatorname{grad}(V) \cdot \nu = 0$.

where J_n and J_p are the current densities of electrons and holes, respectively, and $\varepsilon\, \partial_t E = -\varepsilon\, \partial_t \operatorname{grad} V$ is the displacement current density which is due to the electrostatic potential V. Since Γ_N is an insulator at any point and preventing *any* current from passing through, one has

$$\int_{\gamma_N} J_n \cdot \nu \, d\gamma = \int_{\gamma_N} J_p \cdot \nu \, d\gamma = \int_{\gamma_N} -\varepsilon \partial_t \operatorname{grad} V \cdot \nu \, d\gamma = 0$$

for any $\gamma_N \subseteq \Gamma_N$ arbitrary small. This finally yields (B.12).

Further, the initial values of the charge carrier concentrations n and p are prescribed,

$$n(x, t_0) = n_{\text{init}}(x), \tag{B.13a}$$

$$p(x, t_0) = p_{\text{init}}(x), \tag{B.13b}$$

where $x \in \Omega$ is the position in the interior of the semiconductor and t_0 is the initial time.

The entire initial-boundary value problem for the diode is then given by the drift-diffusion equations (B.5) – (B.7), the boundary conditions for the charge carrier concentrations and the electrostatic potential at the Ohmic contacts (B.10) and (B.11) and those for the current densities and the electrostatic potential at the insulating parts of the semiconductor (B.12), and finally the initial values for the charge carrier concentrations (B.13) at initial time t_0.

B.5. Coupling Conditions

This section shows how to couple the entire initial-boundary value problem with the remaining network equations.

Let Γ_j, $j = 1, 2$, be the metal (Ohmic) contacts of the diode as shown in Figure B.3. Then the current i_j *leaving* the diode and flowing through Γ_j consists of three components, since there are the currents of electrons and holes and the displacement current which is due to the electrostatic potential. Integrating the corresponding current densities over the contact Γ_j yields

$$i_j = \int_{\Gamma_j} (J_n + J_p - \varepsilon \partial_t \operatorname{grad} V) \cdot \nu \, d\gamma, \tag{B.14}$$

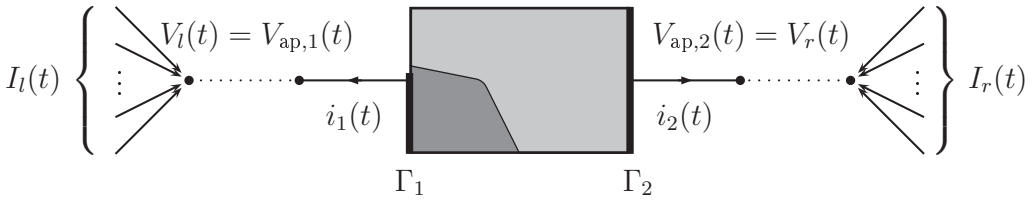


FIGURE B.5. Connecting the drift-diffusion model equations for the diode to the remaining network equations.

where ν is the unit outward normal vector on Γ_j. From equation (B.11) one has $V(x,t) = V_{\mathrm{ap},j}(t) + V_{\mathrm{bi}}(x)$ for the electrostatic potential V with $x \in \Gamma_j$ and time t, where $V_{\mathrm{ap},j}$ is the applied voltage potential at Γ_j. If Γ_j is connected to node k with potential v_k in the entire network, then

$$v_k(t) = V_{\mathrm{ap},j}(t) \tag{B.15}$$

holds for all t. Denote the sums of all the remaining *incoming* currents of the nodes to which the *left* and *right* contacts Γ_1 and Γ_2 of the diode are connected by I_l and I_r (cf. Figure B.5) and their respective voltage potentials by V_l and V_r. Then

$$\begin{aligned} V_{\mathrm{ap},1}(t) &= V_l(t), & i_1(t) &= -I_l(t), \\ V_{\mathrm{ap},2}(t) &= V_r(t), & i_2(t) &= -I_r(t) \end{aligned} \tag{B.16}$$

are the coupling conditions between the network and the drift-diffusion model for the diode.

APPENDIX C

The Operational Amplifier op741

In Section 4.7, an example application for the new hierarchical reduction approach is given. It reduces the describing system of equations for the operational amplifier op741 shown in Figures 4.11 and C.1, respectively, by exploiting its hierarchical structure. Here, a somewhat more detailed description of the functionalities of the amplifier's components and subcircuits is given from an electrical-engineering point of view.

First of all, recall the occuring subcircuits in the operational amplifier. We have four *current mirror* subcircuits (CM1–4), a *differential pair* subcircuit (DP), a *Darlington pair* subcircuit (DAR), and finally a *Level-Shift* and a *Push-Pull* subcircuit (LS) and (PP), respectively.

The entire circuit has to be thought of as divided into three stages which are separated from each other by two vertical lines. The first line has to be drawn dividing (CM2) into two parts such that (CM1) is completely in the left part of the circuit and (CM3) completely in the right part. A second vertical line has to be drawn between (CM3) and (LS). While the third stage on the right is the *output stage* of the operational amplifier, the two leftmost stages are *amplification stages*. The transistors $Q1$ and $Q2$ form the actual differential pair of the first amplification stage, i.e. the leftmost stage.

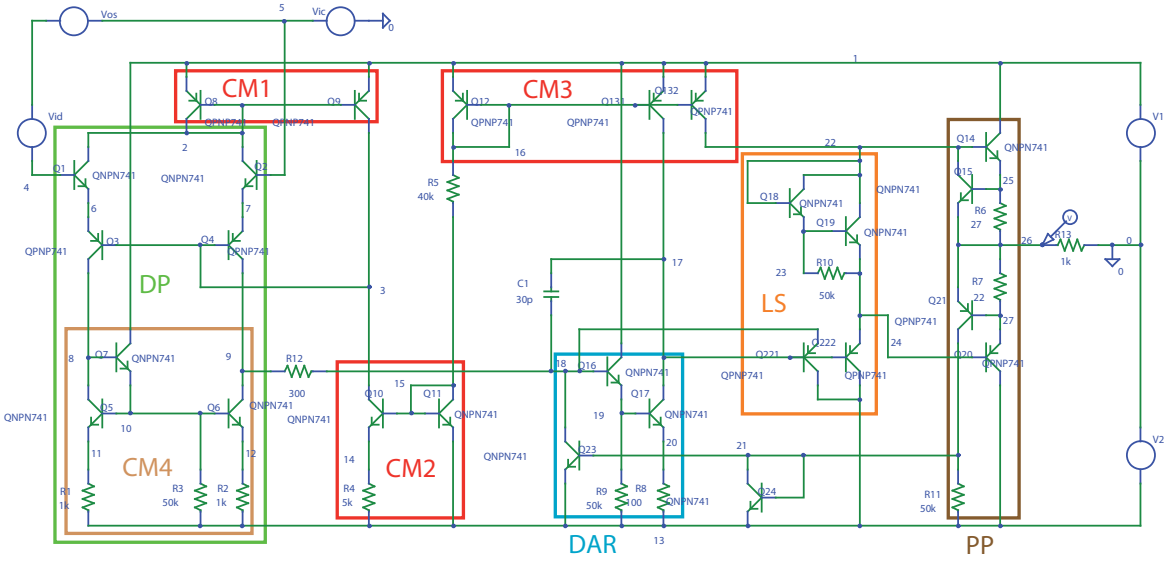


FIGURE C.1. Segmented operational amplifier op741 with a differential pair (DP) including a current mirror (CM4), three more current mirrors (CM1–3), a Darlington pair (DAR), a Level-Shift pair (LS), and a Push-Pull pair (PP) of transistors.

Usually, a current mirror is used to *mirror* a current from one of its sides to the other one, as the name already induces. For (CM1–3), one of the involved transistors has its base terminal short-circuited with either the collector or the emitter terminal. From that side of the current mirror, the current is mirrored to its other side. Therefore, (CM2), for example, mirrors a current from the amplifier's second stage back to the first stage, from where it runs towards node 3.

However, (CM2) together with (CM1) has a further functionality in the op741 at hand, since they form a so-called *polarization loop* which is used to preserve the polarization of the differential pair $Q1$ and $Q2$ in the first stage. The collector terminals of these two transistors are connected to (CM1) which mirrors the sum of the corresponding currents to its other "side", where this current, say I, runs to node 3. I and the mirrored current from (CM2), which has an adjusting effect, are led back to the differential pair passing by transistors $Q3$ and $Q4$, whose function is explained below. *Without* the polarization loop, one would have big changes for the voltages at the differential pair, even for very small changes in the supply voltages `V1` and `V2`. But *with* this loop built in the amplifier, the voltages are stable.

In the lower part of (DP), the transistors $Q5$ and $Q6$ – together with $Q7$ which improves the *mirror factor* [1] – form another type of a current mirror (CM4). In this case, however, it works as a load for the differential pair. Thus, the transistors $Q1$ and $Q2$ of the differential pair are able to work by only a small voltage drop.

The function of the transistors $Q3$ and $Q4$ is to protect the differential pair, e.g. from connection errors in a laboratory, and stabilizing the above polarization loop. The principal reason for having this pair of transistors built in the op741 is as follows: without that pair, the "load current mirror" (CM4) would have to be placed *above* the differential pair. In a consequence, the *stage output* of the first stage, i.e. the voltage potential at node 9, would be closer to the positive voltage supply thus forcing the usage of transistors of the pnp-type for the differential pair and the Darlington pair in (DAR) instead of npn. However, pnp-transistors usually work less well than those of the npn-type.

The output of the amplifier's first stage is transferred to the second stage via the connection from node 9 to node 18, passing resistor `R12`. Let the voltage potential at node 18 be denoted by V_{in}. Then the Darlington pair of transistors $Q16$ and $Q17$ in (DAR) works as a second amplifying stage,

$$V_{out} = c \cdot V_{in},$$

where V_{out} denotes the voltage potential at node 17. Similar to (CM4) in (DP), the current mirror (CM3) thereby works as an active load for (DAR). Furthermore, the role of the capacitor `C1` is compensation; the op741 amplifier works unstably without `C1`.

The amplified voltage V_{out} is then transferred to the third stage, the output stage. The actual "output" of the op741 amplifier is taken over by the Push-Pull subcircuit (PP), whose first function is to give current to the *load*, i.e. the component connected to the op741's output terminal at node 26. The second function of the (PP) subcircuit is a *second separation* between the amplifying stages and the output stage. Thus, it is

[1] The mirror factor simply is the ratio of the current and its mirrored counterpart.

hidden whether the load is connected in parallel or in series to the amplified voltage. Otherwise the amplification could be "killed".

A *first separation* of the amplifying stages from the output stage is realized by transistor $Q222$ in the Level-Shift subcircuit (LS). The pair of transistors $Q18$ and $Q19$ together with resistor R10 prevent the Push-Pull transistors from not working if the transient output signal is close to 0. Otherwise, since transistors need a certain *operating voltage*, an output signal such as a sine wave-like shaped one could get some "dents" in those regions, where it passes the time axis, i.e. where its value is close to zero.

Bibliography

[AdaAroKre71] Adamjan, V.M., Arov, D.Z., Krein, M.G.: Analytic properties of Schmidt pairs for a Hankel operator and the generalized Schur-Takagi problem. Math. USSR Sbornik **15**, 31–73 (1971)

[AdaAroKre78] Adamjan, V.M., Arov, D.Z., Krein, M.G.: Infinite block Hankel matrices and related extension problems. Amer. Math. Soc. Trans.**111**, 133–156 (1978)

[AI] Analog Insydes website: `http://www.analog-insydes.de`

[AIman] Halfmann, T., Hennig, E., Thole, M., Wichmann, T.: Analog Insydes 2 Manual. Fraunhofer ITWM, Kaiserslautern (2001)

[AliBarGueTis] Alí, G., Bartel, A., Günther, M., Tischendorf, C.: Elliptic partial differential-algebraic multiphysics models in electrical network design. Mathematical Models and Methods in Applied Sciences 13:9, 1261–1278 (2003)

[AndAnt] Anderson, B., Antoulas, A.: Rational interpolation and state-variable realizations. Linear Algebra Appl. **137-138**, 479–509 (1990)

[answers] Answers.com: `http://www.answers.com`

[Ant] Antoulas, A.C.: Approximation of Large-Scale Dynamical Systems. SIAM, Philadelphia, PA (2005)

[Ant05] Antoulas, A.C.: A new result on passivity preserving model reduction. Systems Control Lett. **54**, 361–374 (2005)

[AntSor] Antoulas, A.C., Sorensen, D.C.: Approximation of large-scale dynamical systems: An overview. Special Issue in Numerical Analysis and System Theory, Edited by S.L. Campbell. International J. of Applied Mathematics and Computational Science, 11, 1093–1121 (2001)

[ArnLetHer] Arnold, E., Letavic, T., Herko, S.: High-Field Electron Velocity in Silicon Surface-Accumulation Layers. IEEE Electron Device Letters, Vol. **20**, No. **9**, 490–492 (1999)

[Bai] Bai, Z.: Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems. Appl. Numer. Math. **43**, 9–44 (2002)

[BaiSloSmiYe] Bai, Z., Slone, R., Smith, W., Ye, Q.: Error bound for reduced system model by Padé approximation via the Lanczos process. IEEE Trans. Comput. Aided Design **18**, 133–141 (1999)

[BakGra] Baker, G., Jr., Graves-Morris, P.: Padé Approximants. Second edition. Encyclopedia of Mathematics and its Applications **59**, Cambridge University Press, Cambridge (1996)

[Bec] Bechtold, T.: Model Order Reduction of Electro-Thermal MEMS. Ph.D. Thesis, Albert-Ludwigs Universität Freiburg (2005)

[BecRudKor] Bechtold, T., Rudnyi, E.B., Korvink, J.G.: Error Indicators for Fully Automatic Extraction of Heat-Transfer Macromodels for MEMS. Journal of Micromechanics and Microengineering, Vol. 15, N3, 430–440 (2005)

[BenMehSor] Benner, P., Mehrmann, V., Sorensen, D.: Dimension Reduction of Large-Scale Systems. Vol. 45 of Lecture Notes in Computational Science and Engineering, Springer Verlag Berlin Heidelberg (2005)

[BodTis] Bodestedt, M., Tischendorf, C.: PDAE models of integrated circuits and perturbation analysis. Math. Comput. Model. Dyn. Syst. 13(1), 1–17 (2007)

[Bor97] Borchers, C.: Automatische Generierung von Verhaltensmodellen für nichtlineare Analogschaltungen. Ph.D. Thesis, Institut für Mikroelektronische Systeme, Universität Hannover (1997)

[Bor98] Borchers, C.: The Symbolic Behavioral Model Generation of Nonlinear Analog Circuits. IEEE Transactions on Circuits and Systems II, **45:10**, 1362–1371 (1998)

[Bra] Branin, Jr., F.H.: A Sparse Matrix Modification of Kron's Method of Piecewise Analysis. Proc. IEEE Intl. Symp. on Circ. Theory, 383–386, Boston (1975)

[BreCamPet] Brenan, K.E., Campbell, S.L., Petzold, L.R.: The Numerical Solution of Initial Value Problems in Ordinary Differential-Algebraic Equations. North Holland Publishing Co. (1989)

[CAD] Cadence website: `http://www.cadence.com`

[CamMar] Campbell, S.L., Marszalek, W.: The Index of an Infinite Dimensional Implicit System. Mathematical and Computer Modelling of Dynamical Systems, Vol. 5, Issue **1**, 18–42 (1999)

[Cha] Chatterjee, A.: An introduction to the proper orthogonal decomposition. Current Science, Vol. **78**, No. **7**, 808–817 (2000)

[ChaKan] Chang, E.C., Kang, S.-M.: Transient Simulation of Lossy Coupled Transmission Lines Using Iterative Linear Least Square Fitting and Piecewise Recursive Convolution. IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, Vol. **43**, No. **11**, 923–932 (1996)

[ChaMcKayWie] Chang, S.M., MacKay, J.F., Wierzba, G.M.: Matrix Reduction and Numerical Approximation During Computation Techniques for Symbolic Analog Circuit Analysis. Proc. IEEE International Symposium on Circuits and Systems 1992 (ISCAS'92), San Diego, USA, 1153–1156 (1992)

[ChuChe] Chua, L.O., Chen, L.K.: Nonlinear Diakoptics. Proc. IEEE Intl. Symp. on Circ. Theory, 373–376, Boston (1975)

[ClaHol] Clark, J., Holton, D.A.: A First Look At Graph Theory. World Scientific Publishing Company (1991)

[DroSomHor] Dröge, G., Sommer, R., Horneber, E.-H.: Obtaining Compact Network Equations by Reduction of Matrices. Proc. 2nd International Workshop on Symbolic Methods and Applications in Circuit Design 1992 (SMACD'92), Florence, Italy, 83–93 (1992)

[EckYou] Eckart, C., Young, G.: The approximation of one matrix by another of lower rank. Psychometrika **1**, 211–218 (1936)

[FasBen] Faßbender, H., Benner, P.: Numerische Methoden zur passivitätserhaltenden Modellreduktion. at – Automatisierungstechnik **54**, 4, 153–160, Oldenbourg (2006)

[FelFre] Feldmann, P., Freund, R.: Efficient linear circuit analysis by Padé approximation via the Lanczos process. IEEE Trans. Computer-Aided Design **14**, 639–649 (1995)

[FelParFar] Felippa, C., Park, C., Farhat, C.: Partitioned analysis of coupled mechanical systems. Comput. Methods Appl. Mech. Engrg. 190, 3247–3270 (2001)

[FerRodHue] Fernández, F.V., Rodríguez-Vázquez, A., Huertas, J.L.: A Tool for Symbolic Analysis of Analog Integrated Circuits Including Pole/Zero Extraction. Proc. 10th European Conference on Circuit Theory and Design 1991 (ECCTD'91), Copenhagen, Denmark, 752–761 (1991)

[Fre03] Freund, R.W.: Model reduction methods based on Krylov subspaces. Acta Numerica **12**, 267–319 (2003)

[Fre04] Freund, R.W.: SPRIM: structure-preserving reduced-order interconnect macromodeling. Technical Digest of the 2004 IEEE/ACM International Conference on Computer-Aided Design, Los Alamos, CA, 80–87 (2004)

[Fre08] Freund, R.W.: Structure-Preserving Model Order Reduction of RCL Circuit Equations. In: van der Vorst, H., Schilders, W., Rommes, J. (eds.), Model Order Reduction: Theory, Research Aspects and Applications, Mathematics in Industry. Springer Berlin/Heidelberg, Germany (2008)

[FreFel] Freund, R., Feldmann, P.: The SyMPVL algorithm and its applications in interconnect simulation. Proceedings of the 1997 International Conference on Simulation of Semiconductor Processes and Devices, New York, 113–116 (1997)

[GalGriDoo] Gallivan, K., Grimme, E., van Dooren, P.: A rational Lanczos algorithm for model reduction. Numerical Algorithms **12**, 33-63 (1996)

[Gea] Gear, C.W.: The simultaneous numerical solution of differential-algebraic equations. IEEE Trans. Circ. Theor. CT-18, 89–95 (1971)

[GerKneVog] Gerthsen, Kneser, Vogel: Physik. Springer, Berlin Heidelberg NewYork, 14. Auflage (1982)

[GieSan] Gielen, G., Sansen, W.: Symbolic Analysis for Automated Design of Analog Integrated Circuits. Kluwer Academic Publishers, Boston, USA (1991)

[Gra] Grabinski, H.: Theorie und Simulation von Leitbahnen. Signalverhalten auf Leitungssystemen in der Mikroelektronik. Springer Verlag, Berlin (1991)

[GraMey] Gray, P.R., Meyer, R.G.: Analysis and Design of Analog Integrated Circuits. John Wiley & Sons, Inc. (1993)

[Gri] Grimme, E.J.: Krylov Projection Methods for Model Reduction. Ph.D. Thesis, University of Illinois, Urbana-Champaign (1997)

[Gue00] Günther, M.: A Joint DAE/PDE Model for Interconnected Electrical Networks. Mathematical and Computer Modelling of Dynamical Systems, Vol. 6, No. 2, 114–128 (2000)

[Gue01] Günther, M.: Partielle differential-algebraische Systeme in der numerischen Zeitbereichsanalyse elektrischer Schaltungen. Fortschritt-Berichte VDI, Reihe 20, Nr. **343**. VDI-Verlag, Düsseldorf (2001)

[GueFel] Günther, M., Feldmann, U.: The DAE-index in electric circuit simulation. Mathematics and Computers in Simulation, Vol. 39, 573–582 (1995)

[Gug03] Gugercin, S.: Projection Methods for Model Reduction of Large-Scale Dynamical Systems. Ph.D. Thesis, Rice University, Houston, Texas (2003)

[Gug08] Gugercin, S.: An iterative SVD-Krylov based method for model reduction of large-scale dynamical systems. Linear Algebra and its Applications, Vol. 428, Issues 8–9, 1964–1986 (2008)

[HacBraGus] Hachtel, G.D., Brayton, R.K., Gustavson, F.G.: The sparse tableau approach to network analysis and design. IEEE Trans. Circuit Theory 18, 101–113 (1971)

[HalWic03] Halfmann, T., Wichmann, T.: Overview of Symbolic Methods in Industrial Analog Circuit Design. Berichte des Fraunhofer ITWM, Nr. **44** (2003)

[HalWic06] Halfmann, T., Wichmann, T.: Symbolic Methods in Industrial Analog Circuit Design. In: Anile, A.M., Ali, G., Mascali, G. (eds.), Scientific Computing in Electrical Engineering. Springer (2006)

[HeeVerSeg] Heemink, A.W., Verlaan, M., Segers, A.J.: Variance reduced ensemble Kalman filter. Report 00-03, Department of Applied Mathematical Analysis, Delft University of Technology, Delft, The Netherlands (2000)

[HeiSorSun] Heinkenschloss, M., Sorensen, D.C., Sun, K.: Balanced Truncation Model Reduction for a Class of Descriptor Systems with Application to the Oseen Equations. SIAM Journal on Scientific Computing, Vol. 30, No. 2, 1038–1063 (2008)

[Hen] Hennig, E.: Symbolic Approximation and Modeling Techniques for Analysis and Design of Analog Circuits. Ph.D. Thesis, Universität Kaiserslautern, Shaker, Aachen (2000)

[HenTweSom] Hennig, E., Tweer, J.M., Sommer, R.: Enhanced Symbolic Matrix Approximation Techniques. Proc. 5th International Workshop on Symbolic Methods and Applications in Circuit Design 1998 (SMACD'98), Kaiserslautern, Germany, 199–206 (1998)

[Her] Herkt, S.: Model Reduction of Nonlinear Problems in Structural Mechanics: Towards a Finite Element Tyre Model for Multibody Simulation. Ph.D. Thesis, Technische Universität Kaiserslautern (2008)

[HinPri] Hinrichsen, D., Pritchard, A.J.: Mathematical Systems Theory I - Modelling, State Space Analysis, Stability and Robustness. Texts in Applied Mathematics, Vol. 48, Springer (2005)

[HoRueBre] Ho, C.-W., Ruehli, A.E., Brennan, P.A.: The modified nodal approach to network analysis. IEEE Trans. Circuit Systems 22, 505–509 (1975)

[HsiRab80] Hsieh, H.Y., Rabbat, N.B.: A Multilevel Newton Algorithm for Nested Macromodel Analysis of Bipolar Networks. Proc. IEEE Intl. Symp. on Circ. and Syst. (1980)

[HsiRab82] Hsieh, H.Y., Rabbat, N.B.: An Efficient Large System Analysis Method Using System Dynamic Behavior. Circuits, Systems and Signal Processing, Vol. 1, No. **1**, Birkhäuser Boston Inc. (1982)

[HsiRabRue] Hsieh, H.Y., Rabbat, N.B., Ruehli, A.E.: Macromodelling and Macrosimulation Techniques. Proc. IEEE, ISCAS 16th DA Conf., 229–235 (1979)

[IkoHar] Ikoma, T., Hara, K.: Temperature Dependence of Hole Saturation Velocity in Silicon. Appl. Phys. **3**, 431–432, Springer (1974)

[Jef] Jeffrey, A.: Quasilinear Hyperbolic Systems and Waves. Pitman, London (1976)

[Jos] Jost, J.: Partielle Differentialgleichungen. Springer, Berlin Heidelberg New York (1998)

[JosKel] Joshi, S.M., Kelkar, A.G.: Robust Stabilization of a Class of Passive Nonlinear Systems. NASA Technical Memorandum 110287 (1996)

[Kal] Kalman, R.E.: On partial realizations, transfer functions, and canonical forms. Acta Polytech. Scand. Math. **31**, 9–32 (1979)

[Kol] Kole, M.E.: Algorithms for Symbolic Circuit Analysis Based on Determinant Calculations. Ph.D. dissertation, University of Twente, Twente, The Netherlands (1996)

[Kro] Kronecker, L.: Algebraische Reduction der Schaaren bilinearer Formen. Sitzungsber. Akad. der Wiss., 763–776, Berlin (1890)

[KunMeh] Kunkel, P., Mehrmann, V.: Differential-Algebraic Equations. EMS Textbooks in Mathematics (2006)

[Kuy] Kuypers, F.: Klassische Mechanik. VCH Verlag, Weinheim, 2. Auflage (1989)

[LamMarTis] Lamour, R., März, R., Tischendorf, C.: PDAEs and Further Mixed Systems as Abstract Differential Algebraic Systems. Applied Numerical Mathematics, Vol. 42, Issues 1–3, 315–335 (2002)

[LasWil] Lassaux, G., Willcox, K.: Model reduction for active control design using multiple-point Arnoldi methods. AIAA Paper 2003-0616, presented at 41st Aerospace Sciences Meeting & Exhibit, Reno, NV, (2003)
`http://hdl.handle.net/1721.1/3702`

[Li] Li, J.-R.: Model Reduction of Large Linear Systems via Low Rank System Gramians. Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA (2000)

[LiBai] Li, R.-C., Bai, Z.: Structure-Preserving Model Reduction using a Krylov Subspace Projection Formulation. Comm. Math. Sci., Vol. 3, No. **2**, 179-–199 (2005)

[LiNorHsi] Lin, P.-M., Nordsieck, A.W., Hsieh, H.Y.: A Convergence Rate Investigation of a Multilevel Newton Algorithm for Large Circuit Analysis. Proc. Mid Western Symposion on Circ. and Syst., Mexico (1983)

[LiWhi] Li, J.-R., White, J.: Efficient model reduction of interconnect via approximate system Gramians. Proceedings of the IEEE/ACM International Conference on Computer-Aided Design, 380–383 (1999)

[MarBar] Martinson, W.S., Barton, P.I.: A Differentiation Index for Partial Differential-Algebraic Equations. SIAM Journal on Scientific Computing, Vol. 21, No. **6**, 2295–2315, SIAM Philadelphia, PA, USA (1999)

[MarRinSch] Markovich, P.A., Ringhofer, C.A., Schmeiser, C.: Semiconductor Equations. Springer, Wien New York (1990)

[MayAnt] Mayo, A.J., Antoulas, A.C.: A framework for the solution of the generalized realization problem. Linear Algebra and its Applications **425**, 634–662 (2007)

[Mir] Mirsky, L.: Symmetric gauge functions and unitarily invariant norms. Quart. J. Math. **11**, 50–59 (1960)

[Miri] Miri, A.M.: Ausgleichsvorgänge in Elektroenergiesystemen. Springer (2000)

[MMA] Wolfram, S.: The Mathematica Book, fifth ed. Wolfram Media Incorporated (2003)

[Moo] Moon, J.W.: Counting Labelled Trees. Canadian Mathematical Congress, 1970, Montreal, William Clowes and Sons, London, UK (1970)

[MOR] Modelreduction website: `http://modelreduction.com`

[ObiAnd] Obinata, G., Anderson, B.: Model Reduction for Control System Design. Springer Verlag, London (2001)

[OdaCelPil] Odabasioglu, A., Celik, M., Pileggi, L.: PRIMA: Passive reduced-order interconnect macromodeling algorithm. IEEE Trans. Circuits and Systems **17**, 645–654 (1998)

[PopHarHedBar] Popp, R., Hartong, W., Hedrich, L., Barke, E.: Error Estimation on Symbolic Behavioral Models of Nonlinear Analog Circuits. SMACD 1998: 5th International Conference on Symbolic Methods and Applications in Circuit Design, Kaiserslautern, Germany, 223–226 (1998)

[PurEA] Puri, R.S., Morrey, D., Durodola, J.F., Morgans, R.C., Howard, C.Q.: A Comparison of Structural-Acoustic Coupled Reduced Order Models (ROMS): Modal Coupling and Implicit Moment Matching via Arnoldi. Proceedings of the 14th International Congress on Sound and Vibration, Cairns, Australia (2007)

[RabHsi] Rabbat, N.B., Hsieh, H.Y.: A Latent Macromodular Approach to Large-Scale Sparse Networks. IEEE Trans. CAS-23, 745–752 (1976)

[RabPed] Rabiei, P., Pedram, M.: Model order reduction of large circuits using balanced truncation. Proceedings of the ASP-DAC Design Automation Conference, Vol. **1**, 237–240 (1999)

[RabSanHsi] Rabbat, N.G.B., Sangiovanni-Vincentelli, A.L., Hsieh, H.Y.: A Multilevel Newton Algorithm with Macromodelling and Latency for the Analysis of Large-Scale Nonlinear Circuits in the Time Domain. IEEE Trans. CAS-26, 733–741 (1979)

[RasNicLoz] Răsvan, V., Niculesco, S.-I., Lozano, R.: Input-Output Passive Framework for Delay Systems. Proceedings of the 39th IEEE Conference on Decision and Control, Vol. 3, 2823–2828 (2000)

[Reic] Reich, S.: Transmission line equations as Hamiltonian partial differential equations: theory and implications for circuit simulation. Kleinheubacher Berichte **40**, 304–312 (1996)

[Reis05] Reis, T.: Model Reduction for a Class of PDAE Systems. Proc. Appl. Math. Mech. **5**, 179–180 (2005)

[Reis06] Reis, T.: Systems Theoretic Aspects of PDAEs and Applications to Electrical Circuits. Shaker, Aachen (2006)

[ReiSty07] Reis, T., Stykel, T.: Stability Analysis and Model Order Reduction for Coupled Systems. Math. Comput. Model. Dyn. Syst., 13(5), 413–436 (2007)

[ReiSty08] Reis, T., Stykel, T.: A Survey On Model Reduction Of Coupled Systems. In: Schilders, W.H.A., van der Voorst, H.A., Rommes, J.: Model Order Reduction: Theory, Research Aspects and Applications, Mathematics in Industry **13**, 133–155, Springer (2008)

[Rew] Rewieński, M.J.: A Trajectory Piecewise-Linear Approach to Model Order Reduction of Nonlinear Dynamical Systems. Ph.D. Thesis, Massachusetts Institute of Technology (2003)

[RewWhi01] Rewieński, M.J., White, J.: A Trajectory Piecewise-Linear Approach to Model Order Reduction and Fast Simulation of Nonlinear Circuits and Micromachined Devices. Proceedings of the International Conference on Computer-Aided Design, 252–257 (2001)

[RewWhi02] Rewieński, M.J., White, J.: Improving Trajectory Piecewise-Linear Approach to Nonlinear Model Order Reduction for Micromachined Devices Using an Aggregated Projection Basis. Proceedings of the 5th International Conference on Modeling and Simulation of Microsystems, 128–131 (2002)

[RewWhi03] Rewieński, M.J., White, J.: A Trajectory Piecewise-Linear Approach to Model Order Reduction and Fast Simulation of Nonlinear Circuits and Micromachined Devices. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. **22**, No. **2**, 155-170 (2003)

[RezGer] Reznik, D., Gerlach, W.: Generalised drift-diffusion model of bipolar transport in semiconductors. Electrical Engineering **70**, 219–225, Springer (1996)

[RodEA] Rodríguez-García, J.D., Guerra, O., Roca, E., Fernández, F.V., Rodríguez-Vázquez, A.: A New Simplification Before and During Generation Algorithm. Proc. 5th International Workshop on Symbolic Methods and Applications in Circuit Design 1998 (SMACD'98), Kaiserslautern, Germany, 110–124 (1998)

[SchE] Schmidt, E.: Zur Theorie der linearen und nichtlinearen Integralgleichungen, Teil 1: Entwicklung willkürlicher Funktionen nach System vorgeschriebener. Math. Ann. **63**, 433-476 (1907)

[SchG] Schmidt, G.: Grundlagen der Regelungstechnik. Springer (1982)

[SchHalLan] Schmidt, O., Halfmann, T., Lang, P.: Coupling of numerical and symbolic techniques for model order reduction in circuit design. Lecture Notes in Electrical Engineering, Springer, to appear.

[SchSty] Schröder, C., Stykel, T.: Passivation of LTI systems. Technical Report 368, DFG Research Center Matheon, Technische Universität Berlin, Berlin, Germany (2007)

[SchVorRom] Schilders, W.H., van der Vorst, H.A., Rommes, J. (Eds.): Model Order Reduction: Theory, Research Aspects and Applications. Mathematics in Industry, Vol. 13, Springer Berlin Heidelberg (2008)

[SedDegFic] Seda, S.J., Degrauwe, M.G.R., Fichtner, W.: A Symbolic Analysis Tool for Analog Circuit Design Automation. Proc. IEEE International Conference on Computer-Aided Design 1988 (ICCAD'88), 488–491 (1988)

[Sel] Selberherr, S.: Analysis and Simulation of Semiconductor Devices. Springer, Wien New York (1984)

[Sho] Shockley, W.: The Theory of p-n Junctions in Semiconductors and p-n Junction Transistors. Bell Syst. Tech. J. **28**, 435 (1949)

[SidGri] Sidi-Ali-Cherif, S., Grigoriadis, K.M.: Efficient model reduction of large scale systems using Krylov-subspace iterative methods. International Journal of Engineering Science, Vol. **41**, 507–520 (2003)

[Slo] Slone, R.D.: Removing the frequency restriction on Padé Via Lanczos with an error bound. Proceedings of the 1998 IEEE International Symposium on Electromagnetic Compatibility, 202–207 (1998)

[SomEA06] Sommer, R., Platte, D., Broz, J., Dreyer, A., Halfmann, T., Barke, E.: Automatic Nonlinear Behavioral Model Generation using Sequential Equation Structures. Proc. 9th International Workshop on Symbolic Methods and Applications in Circuit Design (SMACD 2006), Florence, Italy (2006)

[SomHalBro] Sommer, R., Halfmann, T., Broz, J.: Automated behavioral modeling and analytical model-order reduction by application of symbolic circuit analysis for multi-physical systems. Simulation Modelling Practice and Theory **16**, 1024–1039 (2008)

[SomHen] Sommer, R., Hennig, E.: Rechnergestützte Berechnung elektrischer Schaltungen. Lecture Notes, Universität Kaiserslautern, Kaiserslautern, Germany (1998)

[SomHenDroHor] Sommer, R., Hennig, E., Dröge, G., Horneber, E.-H.: Equation-Based Symbolic Approximation by Matrix Reduction with Quantitative Error Prediction. Alta Frequenza - Rivista di Elettronica, Vol. 5, No. 6, 29–37 (1993)

[SomHenEA] Sommer, R., Hennig, E., Nitsche, G., Schwarz, P., Broz, J.: Automatic Nonlinear Behavioral Model Generation Using Symbolic Circuit Analysis. In: Fakhfakh, M., Tlelo-Cuautle, E., Fernández, F.V. (eds.), Design of Analog Circuits through Symbolic Analysis, to appear.

[SomKraEA] Sommer, R., Kraße, D., Schäfer, E., Hennig, E.: Application of Symbolic Circuit Analysis for Failure Detection and Optimization of Industrial Integrated Circuits. In: Fakhfakh, M., Tlelo-Cuautle, E., Fernández, F.V. (eds.), Design of Analog Circuits through Symbolic Analysis, to appear.

[Sor] Sorensen, D.: Passivity preserving model reduction via interpolation of spectral zeros. Systems Control Lett. **54**, 347–360 (2005)

[SorAnt] Sorensen, D.C., Antoulas, A.C.: The Sylvester equation and approximate balanced reduction. Linear Algebra and its Applications, Vol. **351-352**, 671–700 (2002)

[Spi83] Spiro, H.: Überlegungen zu Rechenzeit und Speicherplatz für spärlich besetzte Matrizen und Systemzerlegungen in Macros. Elektronische Rechenanlagen, Hefte **3** und **4**. R. Oldenbourg Verlag, München (Juni und August 1983)

[Spi90] Simulation integrierter Schaltungen. Oldenbourg, Wien, 2. Auflage (1990)

[SteSun] Stewart, G.W., Sun, J.: Matrix perturbation theory. Academic Press, Boston (1990)

[Sty] Stykel, T.: Balancing-Related Model Reduction of Circuit Equations Using Their Topological Structure. In: Benner, P., Hinze, M., ter Maten, J. (eds.), Model Reduction for Circuit Simulation, Springer-Verlag, Berlin/Heidelberg, to appear.

[SwaThu] Swamy, M.N.S., Thulasiraman, K.: Graphs, Networks, and Algorithms. Wiley, New York (1981)

[SyreNe] SyreNe website: `http://www.syrene.org`

[Tis] Tischendorf, C.: Coupled Systems of Differential Algebraic and Partial Differential Equations in Circuit and Device Simulation. Modeling and Numerical Analysis, Habilitation Thesis, Institut für Mathematik, Humboldt-Universität zu Berlin, Berlin, Germany (2003)

[Ung] Unger, H.G.: Theorie der Leitungen. Vieweg, Braunschweig (1967)

[VanVDoo] Vandendorpe, A., Van Dooren, P.: Model Reduction of Interconnected Systems. In: Schilders, W., van der Vorst, H., Rommes, J. (eds.), Model Order Reduction: Theory, Research Aspects and Applications, Mathematics in Industry Vol. 13, 305–321, Springer Berlin Heidelberg (2008)

[Ver] Verlaan, M.: Efficient Kalman filtering algorithms for hydrodynamic models. Ph.D. Thesis, Technische Universiteit, Delft, The Netherlands (1998)

[VilSchSil] Villena, J.F., Schilders, W.H.A., Silveira, L.M.: Parametric Structure-Preserving Model Order Reduction. In: Reis, R., Mooney, V., Hasler, P. (eds.), IFIP International Federation for Information Processing, Volume **291**, VLSI-SoC: Advanced Topics on Systems on a Chip, 69-–88, Springer (2009)

[VlaSin] Vlach, J., Singhal, K.: Computer Methods for Circuit Analysis and Design. Van Nostrand Reinhold, New York, 2. Auflage (1993)

[vRoo] van Roosbroeck, W.V.: Theory of Flow of Electrons and Holes in Germanium and Other Semiconductors. Bell Syst. Techn. J. **29**, 560–607 (1950)

[Wal] Walter, W.: Gewöhnliche Differentialgleichungen, 6. Auflage, Springer (1996)

[WalGieSan] Walscharts, H., Gielen, G., Sansen, W.: Symbolic Simulation of Analog Circuits in s- and z-Domain. Proc. IEEE International Symposium on Circuits and Systems 1989 (ISCAS'89), Portland, USA, 814–817 (1989)

[Wam] Wambacq, P.: Symbolic Analysis of Large and Weakly Nonlinear Analog Integrated Circuits. Ph.D. dissertation, Katholieke Universiteit Leuven, Leuven, Belgium (1996)

[Wei58] Weierstraß, K.: Über ein die homogenen Funktionen zweiten Grades betreffendes Theorem, nebst Anwendungen desselben auf die Theorie der kleinen Schwingungen. Monatsh. Akad. der Wissensch., 207–220, Berlin (1858)

[Wei67] Weierstraß, K.: Zur Theorie der bilinearen quadratischen Formen. Monatsh. Akad. der Wissensch., 310–338, Berlin (1867)

[Wic01] Wichmann, T.: Simplification of Nonlinear DAE Systems with Index Tracking. ECCTD '01 – European Conference on Circuit Theory and Design, **II**, Espoo, Finland, 173–175 (2001)

[Wic04] Wichmann, T.: Symbolische Reduktionsverfahren für nichtlineare DAE-Systeme. Shaker, Aachen (2004)

[WicEA] Wichmann, T., Popp, R., Hartong, W., Hedrich, L.: On the Simplification of Nonlinear DAE Systems in Analog Circuit Design. Computer Algebra in Scientific Computing - CASC 99, V.G. Ganzha et al., Springer Verlag, Berlin, Germany (1999)

[wikipedia] Wikipedia: `http://en.wikipedia.org`

[Wil72] Willems, J.C.: Dissipative Dynamical Systems Part I: General Theory. Arch. Rational Mech. Anal., Vol. 45., Issue 5, 321–351, Springer, Berlin Heidelberg (1972)

[Wil97] Willems, J.C.: On Interconnections, Control, and Feedback. IEEE Transactions on Automatic Control, Vol. 42, No. 3, 326–339 (1997)

[Wil07] Willems, J.C.: The Behavioral Approach to Open and Interconnected Systems. IEEE Control Systems Magazine **27**, 46–99 (2007)

[YanHajTri] Yang, P., Hajj, I.N., Trick, T.N.: SLATE, a Circuit Simulation Program with Latency Exploitation and Node Tearing. IEEE Proc. ICCC **80**, Vol. 1, 353–355 (1980)

[Zer] Zerz, E.: Introduction to systems and control theory. Technical report, Universität Kaiserslautern, Kaiserslautern (2002)

[ZhoLiCaiGuo] Zhou, D., Li, W., Cai, W., Guo, N.: An efficient balanced truncation realization algorithm for interconnect model order reduction. Proceedings of the 2001 IEEE International Symposium on Circuits and Systems, Vol. **5**, 383–386 (2001)

Index

AC
  analysis, 40, 69
  system, 39, 129
acceleration sensor, 79
admittance formulation, 20–22, 30
amplifier
  differential, 79, 82, 88, 90–92, 95, 104, 129, 131, 133, 138, 140, 141
  operational, 82, 87, 104–106, 108, 115, 119, 122, 124–126, 128, 129, 138, 141, 142, 155
analysis
  AC, 40, 69
  DC, 38, 69
  DT, 38
  hand, 2
  loop, 22, 23
  modified nodal, 5, 8, 20, 21, 31, 69, 77, 88, 94, 107, 136, 139
  network, 11, 139
  nodal, 21, 22
  numerical, 2, 139
  sensitivity, 119, 120
  sparse tableau, 19, 20, 31, 69, 77, 139
  symbolic, 3, 139
  system, 1, 139
  transient, 37, 69
Arnoldi, 52, 60
  algorithm, 61, 90, 91, 93, 129–131, 140, 141

backwards steps, 116, 117
balanced truncation, 55, 80, 130, 140
balancing transformation, 55, 64
band
  conduction, 148
  valence, 148
bandgap, 148, 149
behavioral approach, 78, 81, 140
branch, 13
  link, 14, 145
  tree, 13, 145

Cayley-Hamilton, 43
clustering, 74, 87, 95, 97, 101
complexity, 3, 6, 72, 77, 84, 100, 112, 121, 124, 135, 139

DC
  analysis, 38, 69
  system, 38, 129
depletion region, 149
design point, 36, 129, 130, 134
differential-algebraic equations, 23, 24, 79
  partial, 23, 29
dissipativity, 46, 47
drift-diffusion equations, 33, 34, 139, 147, 151, 154
DT analysis, 38
dyadic decomposition, 57

Ebers-Moll equations, 18
Einstein relations, 150
electron
  conduction, 148
  valence, 148
error function, 8, 69, 74, 87, 102, 104, 106, 112, 116, 117, 127–129, 138, 141, 142
  generalized, 104

frequency response, 6, 57, 70
fundamental loop, 14, 15, 145
  system, 14, 145

graph, 12
  circuit, 13, 79
  connected, 12, 145
  directed, 13, 145
  finite, 12
  loop, *see* loop
  path, 12, 146
  tree, 13, 145, 146
Gummel-Poon equations, 19, 107

Hankel
  norm, 54
  norm approximant, 51
  norm approximation, 54

singular value, 56, 57
$\mathbb{H}_\infty$-norm, 56, 80

impulse response, 47, 59
incidence matrix
loop, 15, 146
nodal, 14, 15, 21, 30, 145, 146
index, 27, 75, 98, 142
initial value, 24, 129
consistent, 24

Kirchhoff
law, 15, 22, 33, 79, 85, 139
space, 16, 17
Krylov
methods, 52, 58, 140
space, 4, 52, 54, 59, 80, 131

Lanczos, 52, 60
algorithm, 62, 63
leakage current, 18, 32, 149
Lebesgue space, 46
linearization, 39
loop, 12
current, 16
fundamental, *see* fundamental loop
impedance matrix, 23
incidence matrix, 15
space, 15, 145
LTI system, *see* system
Lyapunov
equations, 50, 51, 55, 64
stability, 48

macro, 77, 84
macromodel concept, 78, 84, 85, 94, 140, 141
Markov parameter, 60, 62
micro electro-mechanical system, 79, 140
modified nodal analysis, *see* analysis
moment-matching, 59–62, 80
Moore's Law, 77

nodal admittance matrix, 22
node, 13
potential, 16, 20, 84

observability, 44, 55, 56, 140
Gramian, 44, 55, 64
matrix, 45
Ohm space, 19
operating point, 38
output space, 41

p-n junction, 148, 149
Padé approximation problem, 60
partial differential equations, 28, 79
implicit, 29
partial differential-algebraic equations, *see* differential-algebraic equations
partial realization problem, 60
passive, 3, 47
passivity, 45, 59, 68, 140
pattern matching, 102, 142
positive real, 47
projection
Galerkin, 53, 58, 60
Petrov-Galerkin, 53, 62
proper orthogonal decomposition, 52, 57, 58, 140

ranking, 138
absolute, 96
error, 96, 97
of subsystem reductions, 9, 87, 95, 98, 100, 102, 141
perfect, 96, 97
relative, 97
term, 69, 71, 74, 87, 89, 95, 96, 107, 123, 127, 129, 136
rational interpolation, 60
reachability, 42, 55, 56, 140
Gramian, 42, 50, 55, 64
matrix, 42
reduction information, 99, 102, 108, 110
reference solution, 69, 72, 74, 89, 93, 94, 105, 129, 130, 136, 141

Schmidt-Mirsky/Eckart-Young, 54, 58
sensitivity, 95, 100, 102
absolute, 98
analysis, 119, 120
relative, 99
subsystem, 87, 98, 99, 108, 113, 117, 141
shifted moment, 60, 131
Shockley equation, 18, 33
singular value, 54
decomposition, 4, 54
Hankel, *see* Hankel singular value
snapshot matrix, 57
sparse tableau analysis, *see* analysis
sparsity, 19, 64
stability, 48, 59, 68, 140
asymptotic, 48, 55, 56
bounded-input bounded-output, 50
input-output, 50

internal, 50
$\mathcal{L}^q$, 50
Lyapunov, 48
numerical, 142
state
space, 19, 41, 131
space system, *see* system
trajectory, 43, 48, 64, 65, 131
transition map, 41
state-to-output map, 41
SVD-Krylov methods, 63
Sylvester equations, 64
symbolic methods, 5, 127, 139, 140
linear, 70
nonlinear, 72
system
AC, 39, 129
analysis, 139
component-based, 69, 78, 140
coupled, 78, 79, 140
DC, 38, 129
fundamental loop, 14, 145
interconnected, 78, 79, 140
LTI, 19, 22, 25, 46, 49, 51, 55, 78, 82, 84, 129–131, 134, 140, 142
micro electro-mechanical, 79, 140
quasi-linear, 26
semi-explicit, 25
state space, 26, 48, 53, 55, 59, 90, 129–131, 134, 142
transient, 129

telegrapher's equations, 32, 88, 131, 132, 139
terminal, 11
test bench, 94, 95, 108, 135, 141
thermal equilibrium, 150, 152
thermal voltage, 18, 33
TPWL model, 66
training trajectory, 52, 67, 68
trajectory piecewise-linear approach, 52, 64, 140
transfer function, 4, 6, 46, 51, 59, 67, 69, 70
transient
analysis, 37, 69
system, 129
transmission line, 8, 31, 32, 79, 88, 90, 93, 129, 131–134, 139, 142

voltage potential, *see* node potential

Curriculum Vitae

Name	Oliver Schmidt	
Date of Birth	July 22, 1978	
Place of Birth	Wiesbaden, Germany	
School Education	1985–1989	Elementary School in Wiesbaden-Nordenstadt
	1989–1998	Oranienschule, Wiesbaden
	June 1998	Abitur
Civilian Service	1998–1999	Arbeiterwohlfahrt, Wiesbaden
Academic Vitae	1999–2006	Student of Mathematics and Computer Science at the Johannes Gutenberg-University Mainz, Germany
	Sept. 2006	Diploma in Mathematics
	2006–2010	PhD fellowship of the DFG-Graduiertenkolleg "Mathematik und Praxis" at the University of Kaiserslautern, Germany
	July 2010	PhD in Mathematics

Lebenslauf

Name	Oliver Schmidt	
Geburtsdatum	22. Juli 1978	
Geburtsort	Wiesbaden, Deutschland	
Schulbildung	1985–1989	Grundschule Wiesbaden-Nordenstadt
	1989–1998	Oranienschule Wiesbaden (Gymnasium)
	Juni 1998	Abitur
Zivildienst	1998–1999	Arbeiterwohlfahrt, Wiesbaden
Studium	1999–2006	Student der Mathematik und Informatik an der Johannes Gutenberg-Universität Mainz
	Sept. 2006	Diplom in Mathematik
	2006–2010	Doktorand der Mathematik und Stipendiat des DFG-Graduiertenkollegs "Mathematik und Praxis" an der Technischen Universität Kaiserslautern
	Juli 2010	Abschluss der Promotion in Mathematik